RBAC-MAS & SODA: Experimenting RBAC in AOSE

Ambra Molesini & Enrico Denti & Andrea Omicini
{ambra.molesini, enrico.denti, andrea.omicini}@unibo.it

Alma Mater Studiorum—Università di Bologna

ESAW 2008, Saint-Étienne, France, 25th September 2008

The Objective of this Paper

- Our work is aimed at discussing the methodological support provided by SODA, an AOSE methodology, for a particular security issue: the access control
- In order to do this we
  - present the *Role-Based Access Control* (RBAC) standard and its extension for MAS infrastructures (RBAC-MAS)
  - extract requirements for engineering an RBAC system
  - show how SODA supports these requirements
  - apply SODA to the engineering of a concrete case study— the management of the access control to a university building

The Objective of this Paper

- Our work is aimed at discussing the methodological support provided by SODA, an AOSE methodology, for a particular security issue: the access control
- In order to do this we
  - present the *Role-Based Access Control* (RBAC) standard and its extension for MAS infrastructures (RBAC-MAS)
  - extract requirements for engineering an RBAC system
  - show how SODA supports these requirements
  - apply SODA to the engineering of a concrete case study— the management of the access control to a university building

The Objective of this Paper

- Our work is aimed at discussing the methodological support provided by SODA, an AOSE methodology, for a particular security issue: the access control
- In order to do this we
  - present the *Role-Based Access Control* (RBAC) standard and its extension for MAS infrastructures (RBAC-MAS)
  - extract requirements for engineering an RBAC system
  - show how SODA supports these requirements
  - apply SODA to the engineering of a concrete case study— the management of the access control to a university building

The Objective of this Paper

- Our work is aimed at discussing the methodological support provided by SODA, an AOSE methodology, for a particular security issue: the access control
- In order to do this we
  - present the *Role-Based Access Control* (RBAC) standard and its extension for MAS infrastructures (RBAC-MAS)
  - extract requirements for engineering an RBAC system
  - show how SODA supports these requirements
  - apply SODA to the engineering of a concrete case study— the management of the access control to a university building

The Objective of this Paper

- Our work is aimed at discussing the methodological support provided by SODA, an AOSE methodology, for a particular security issue: the access control
- In order to do this we
  - present the *Role-Based Access Control* (RBAC) standard and its extension for MAS infrastructures (RBAC-MAS)
  - extract requirements for engineering an RBAC system
  - show how SODA supports these requirements
  - apply SODA to the engineering of a concrete case study— the management of the access control to a university building

The Objective of this Paper

- Our work is aimed at discussing the methodological support provided by SODA, an AOSE methodology, for a particular security issue: the access control
- In order to do this we
    - present the *Role-Based Access Control* (RBAC) standard and its extension for MAS infrastructures (RBAC-MAS)
    - extract requirements for engineering an RBAC system
    - show how SODA supports these requirements
    - apply SODA to the engineering of a concrete case study— the management of the access control to a university building

Access Control

- Access control is aimed at enabling (only) the authorised users to access the system resources in a controlled and supervised way
- Key aspect: the clear separation between
  - the access policy used to decide whether access to a resource should or not be granted for a given user
  - the hardware & software mechanisms actually enforcing such rules
- Such a separation is useful for two main reasons:
  - to uncouple the definition of a policy from its implementation, so that the latter is not affected by policy changes
  - to more easily identify the basic properties that any access control system should satisfy (complete mediation, default deny, minimum privilege, . . .)

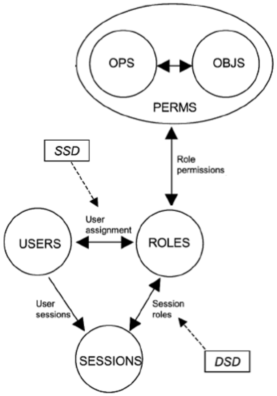
Access Control

- Access control is aimed at enabling (only) the authorised users to access the system resources in a controlled and supervised way
- Key aspect: the clear separation between
  - the access policy used to decide whether access to a resource should or not be granted for a given user
  - the hardware & software mechanisms actually enforcing such rules
- Such a separation is useful for two main reasons:
  - to uncouple the definition of a policy from its implementation, so that the latter is not affected by policy changes
  - to more easily identify the basic properties that any access control system should satisfy (complete mediation, default deny, minimum privilege, . . .)

Access Control

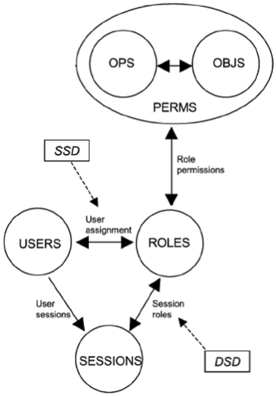
- Access control is aimed at enabling (only) the authorised users to access the system resources in a controlled and supervised way
- Key aspect: the clear separation between
  - the access policy used to decide whether access to a resource should or not be granted for a given user
  - the hardware & software mechanisms actually enforcing such rules
- Such a separation is useful for two main reasons:
  - to uncouple the definition of a policy from its implementation, so that the latter is not affected by policy changes
  - to more easily identify the basic properties that any access control system should satisfy (complete mediation, default deny, minimum privilege, . . .)

RBAC



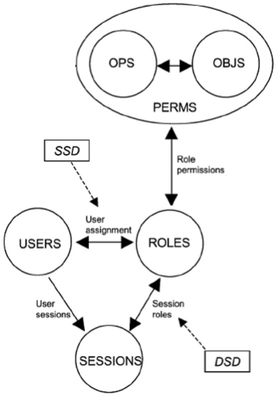
- Role is a job function with associated authority and responsibilities conferred to the User

RBAC



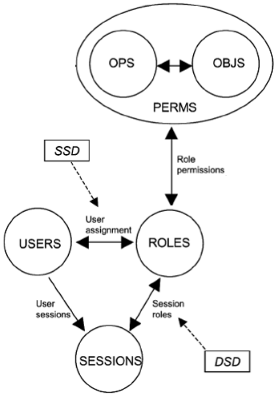
- Role is a job function with associated authority and responsibilities conferred to the User
- Permission is an approval to perform an Operation on some protected Objects

RBAC



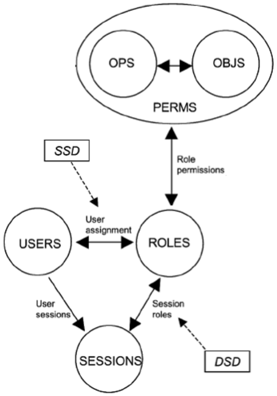
- Role is a job function with associated authority and responsibilities conferred to the User
- Permission is an approval to perform an Operation on some protected Objects
- Session is a mapping between a user and the subset of its currently active roles

RBAC



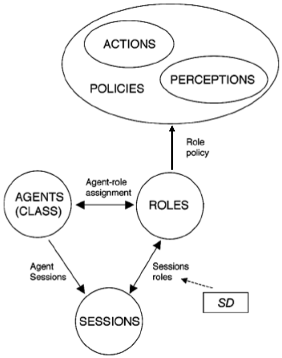
- Role is a job function with associated authority and responsibilities conferred to the User
- Permission is an approval to perform an Operation on some protected Objects
- Session is a mapping between a user and the subset of its currently active roles
- Static separation of duty (SSD) is obtained by enforcing constraints on the assignment of users to roles

RBAC



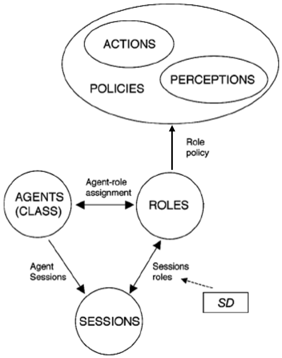
- Role is a job function with associated authority and responsibilities conferred to the User
- Permission is an approval to perform an Operation on some protected Objects
- Session is a mapping between a user and the subset of its currently active roles
- Static separation of duty (SSD) is obtained by enforcing constraints on the assignment of users to roles
- Dynamic separation of duty (DSD) is achieved by placing constraints on the roles that can be activated within or across the given users' session

RBAC-MAS



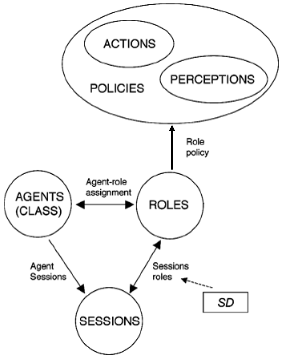
- RBAC Users are represented in RBAC-MAS by Agent Classes

RBAC-MAS



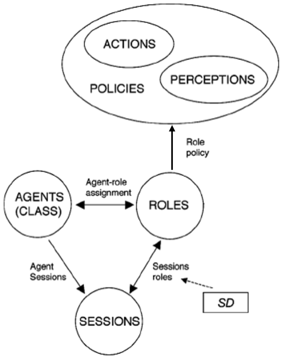
- RBAC Users are represented in RBAC-MAS by Agent Classes
- The behaviour of each role is defined in terms of Actions and Perceptions used, respectively, to affect and perceive the computational environment

RBAC-MAS



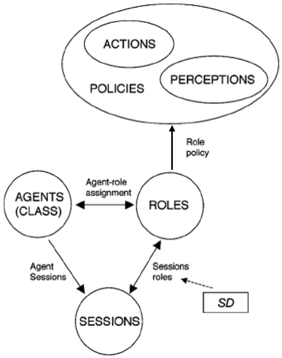
- RBAC Users are represented in RBAC-MAS by Agent Classes
- The behaviour of each role is defined in terms of Actions and Perceptions used, respectively, to affect and perceive the computational environment
- Policies constrain the admissible interaction histories of an agent playing a specific role, and are used to model the organisational rules

RBAC-MAS



- RBAC Users are represented in RBAC-MAS by Agent Classes
- The behaviour of each role is defined in terms of Actions and Perceptions used, respectively, to affect and perceive the computational environment
- Policies constrain the admissible interaction histories of an agent playing a specific role, and are used to model the organisational rules
- Agents' session starts with no activated roles

RBAC-MAS



- RBAC Users are represented in RBAC-MAS by Agent Classes
- The behaviour of each role is defined in terms of Actions and Perceptions used, respectively, to affect and perceive the computational environment
- Policies constrain the admissible interaction histories of an agent playing a specific role, and are used to model the organisational rules
- Agents' session starts with no activated roles
- Dynamics of role activation is constrained by the DSD rules

RBAC-MAS Requirements

- Role — supporting the modelling and design of both the user roles and the administrative roles
- Organisation — supporting the modelling and design of agent societies and the rules that govern them
- Object — hiding a lot of complexity:
  - able to model the environment of the MAS. . .
  - provide the physical and logical control to prevent unauthorised access. . .
  - . . . so, model and design both the topological structure and the resources that populate the environment
- Action and Perception — supporting the modelling and design of the actions that roles can perform over the objects and of the perceptions of the environment
- Policy — supporting the design of rules concerning the abstractions

RBAC-MAS Requirements

- Role — supporting the modelling and design of both the user roles and the administrative roles
- Organisation — supporting the modelling and design of agent societies and the rules that govern them
- Object — hiding a lot of complexity:
  - able to model the environment of the MAS. . .
  - provide the physical and logical control to prevent unauthorised access. . .
  - . . . so, model and design both the topological structure and the resources that populate the environment
- Action and Perception — supporting the modelling and design of the actions that roles can perform over the objects and of the perceptions of the environment
- Policy — supporting the design of rules concerning the abstractions

RBAC-MAS Requirements

- Role — supporting the modelling and design of both the user roles and the administrative roles
- Organisation — supporting the modelling and design of agent societies and the rules that govern them
- Object — hiding a lot of complexity:
  - able to model the environment of the MAS...
  - provide the physical and logical control to prevent unauthorised access...
  - ... so, model and design both the topological structure and the resources that populate the environment
- Action and Perception — supporting the modelling and design of the actions that roles can perform over the objects and of the perceptions of the environment
- Policy — supporting the design of rules concerning the abstractions

RBAC-MAS Requirements

- Role — supporting the modelling and design of both the user roles and the administrative roles
- Organisation — supporting the modelling and design of agent societies and the rules that govern them
- Object — hiding a lot of complexity:
  - able to model the environment of the MAS. . .
  - provide the physical and logical control to prevent unauthorised access. . .
  - . . . so, model and design both the topological structure and the resources that populate the environment
- Action and Perception — supporting the modelling and design of the actions that roles can perform over the objects and of the perceptions of the environment
- Policy — supporting the design of rules concerning the abstractions

RBAC-MAS Requirements

- Role — supporting the modelling and design of both the user roles and the administrative roles
- Organisation — supporting the modelling and design of agent societies and the rules that govern them
- Object — hiding a lot of complexity:
  - able to model the environment of the MAS. . .
  - provide the physical and logical control to prevent unauthorised access. . .
  - . . . so, model and design both the topological structure and the resources that populate the environment
- Action and Perception — supporting the modelling and design of the actions that roles can perform over the objects and of the perceptions of the environment
- Policy — supporting the design of rules concerning the abstractions

Policy and Mechanism Separation Requirements

- The separation between policy and mechanism introduces further constraints:
  - while such two sub-systems can be designed separately
  - they are indirectly coupled by the *representation language* of the access policies, since these are designed by one sub-system, but enforced by the other
  - it is not necessary to know the specific policy during the mechanism design phase: knowing how the policy is represented is relevant to choose the most appropriate storage and to decide the most adequate enforcing implementation
- The mechanism sub-system should manage the association between users and roles in a dynamic way:
  - support and implement policies changes with no need to stop or reset

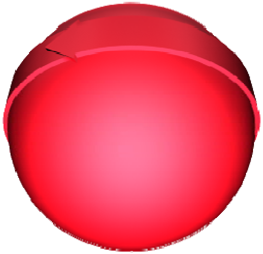
Policy and Mechanism Separation Requirements

- The separation between policy and mechanism introduces further constraints:
  - while such two sub-systems can be designed separately
  - they are indirectly coupled by the *representation language* of the access policies, since these are designed by one sub-system, but enforced by the other
  - it is not necessary to know the specific policy during the mechanism design phase: knowing how the policy is represented is relevant to choose the most appropriate storage and to decide the most adequate enforcing implementation
- The mechanism sub-system should manage the association between users and roles in a dynamic way:
  - support and implement policies changes with no need to stop or reset

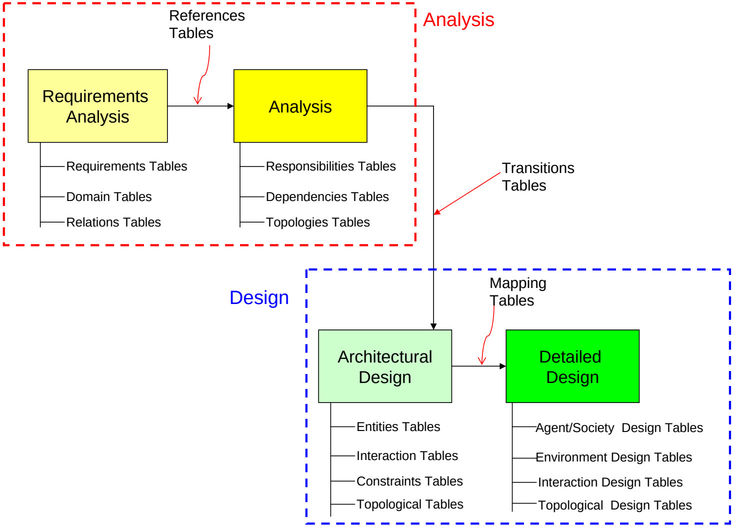
SODA: Societies in Open and Distributed Agent spaces

SODA . . .

- . . . is an agent-oriented methodology for the analysis and design of agent-based systems
- . . . focuses on inter-agent issues, like the engineering of societies and environment for MAS
- . . . adopts agents and artifacts – after the A&A meta-model – as the main building blocks for MAS development
- . . . introduces a simple *layering* principle in order to cope with the complexity of system description
- . . . adopts a tabular representation



SODA: Overview



A&A Meta-model

- Agents model individual and social activities
- Artifacts *glue* agents together, as well as MAS and the environment
  - artifacts mediate between individual agents and MAS
  - artifacts build up agent societies
  - artifacts wrap up the resources of MAS and bring them to the cognitive level of agents
- Workspaces structure agents and artifacts organisation & interaction

A&A Meta-model

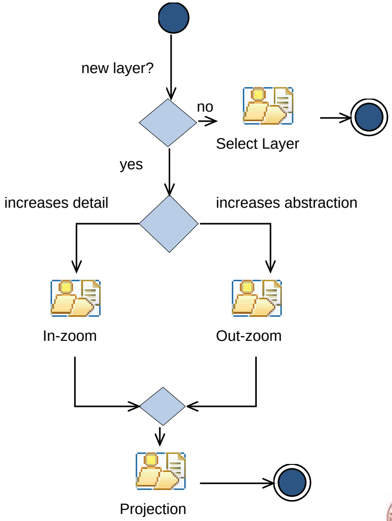
- Agents model individual and social activities
- Artifacts *glue* agents together, as well as MAS and the environment
  - artifacts mediate between individual agents and MAS
  - artifacts build up agent societies
  - artifacts wrap up the resources of MAS and bring them to the cognitive level of agents
- Workspaces structure agents and artifacts organisation & interaction

A&A Meta-model

- Agents model individual and social activities
- Artifacts *glue* agents together, as well as MAS and the environment
    - artifacts mediate between individual agents and MAS
    - artifacts build up agent societies
    - artifacts wrap up the resources of MAS and bring them to the cognitive level of agents
- Workspaces structure agents and artifacts organisation & interaction

Layering in SODA

- The layering principle is achieved by means of the zoom and projection mechanisms
- Two kinds of zoom
  - in-zoom — from an abstract to a more detailed layer
  - out-zoom — from a detailed to a more abstract layer
- The *projection mechanism* projects entities from one to another layer

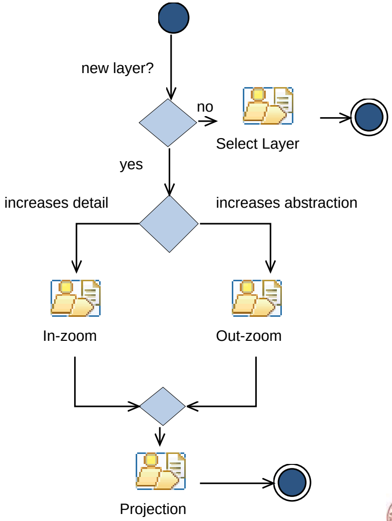


Layering in SODA

- The layering principle is achieved by means of the zoom and projection mechanisms
- Two kinds of zoom

in-zoom	— from an abstract to a more detailed layer
out-zoom	— from a detailed to a more abstract layer

- The *projection mechanism* projects entities from one to another layer

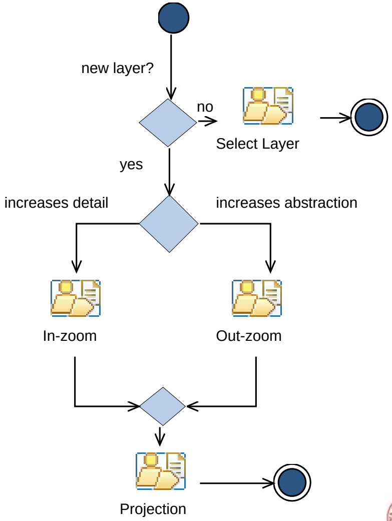


Layering in SODA

- The layering principle is achieved by means of the zoom and projection mechanisms
- Two kinds of zoom

in-zoom	— from an abstract to a more detailed layer
out-zoom	— from a detailed to a more abstract layer

- The *projection mechanism* projects entities from one to another layer



Outline

SODA&RBAC-MAS Requirements

RBAC-MAS	SODA
Role	*role*
Organisation	*societies and rules*
Object	*legacy-system, function, resource, artifact topology, space, workspace*
Action and Perception	*action, uses manifests*
Policy	*rule, artifact*
Policy language	*orthogonal to any language*
User-role association	*artifact*

- In the design of the mechanism sub-system only the reactive abstractions are involved
- In the design of the policy sub-system only the interactions and rules abstractions are used
- The active abstractions are just modelled: from the RBAC design perspective, roles are an input of the policy sub-system

SODA&RBAC-MAS Requirements

RBAC-MAS	SODA
Role	*role*
Organisation	*societies and rules*
Object	*legacy-system, function, resource, artifact topology, space, workspace*
Action and Perception	*action, uses manifests*
Policy	*rule, artifact*
Policy language	*orthogonal to any language*
User-role association	*artifact*

- In the design of the mechanism sub-system only the reactive abstractions are involved
- In the design of the policy sub-system only the interactions and rules abstractions are used
- The active abstractions are just modelled: from the RBAC design perspective, roles are an input of the policy sub-system

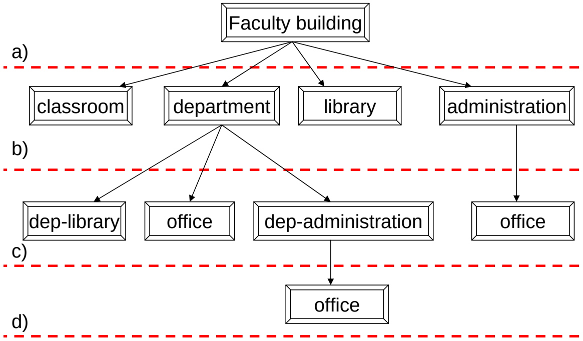
SODA&RBAC-MAS Requirements

RBAC-MAS	SODA
Role	*role*
Organisation	*societies and rules*
Object	*legacy-system, function, resource, artifact topology, space, workspace*
Action and Perception	*action, uses manifests*
Policy	*rule, artifact*
Policy language	*orthogonal to any language*
User-role association	*artifact*

- In the design of the mechanism sub-system only the reactive abstractions are involved
- In the design of the policy sub-system only the interactions and rules abstractions are used
- The active abstractions are just modelled: from the RBAC design perspective, roles are an input of the policy sub-system

The Case Study

- Management of the access control to a university building
- Key system aspect:



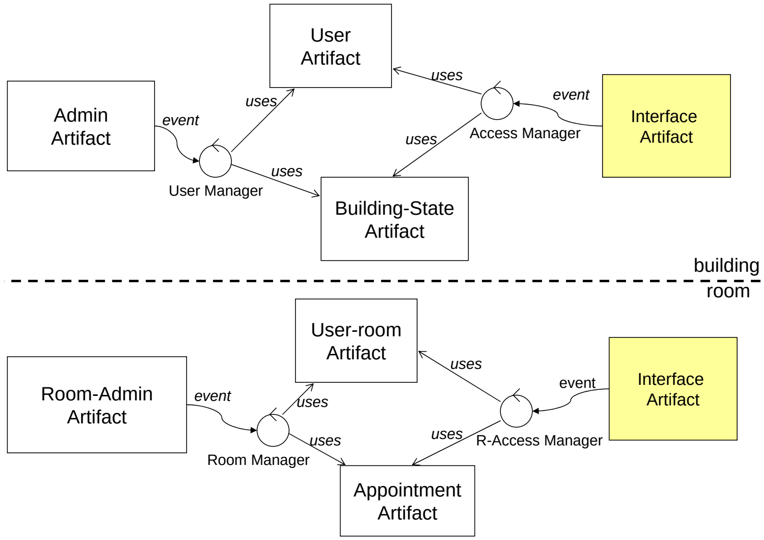
Outline

SODA's Tables

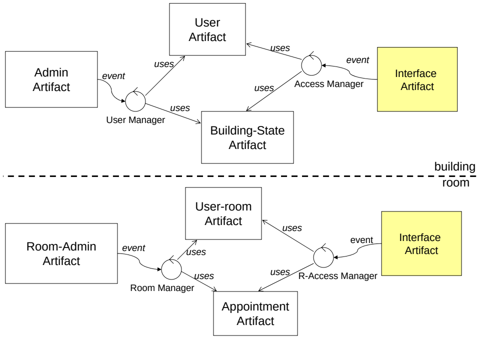
Space	Description
Faculty	the whole building
Classroom	the student space
Library	the faculty library
Department	the research centre
Administration	the faculty bureaucracy centre
Dep-Library	the department library
Dep-Administration	the department bureaucracy centre
Office	the rooms for employees

Space	Connection
Faculty	Classroom, Library, Department, Administration
Administration	Office
Department	Dep-Library, Dep-Administration, Office
Dep-Administration	Office

The mechanism

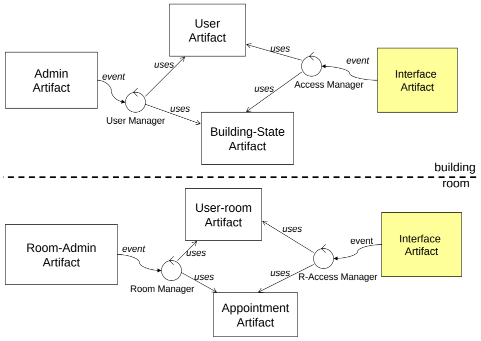


Mechanism’s Artifacts and Agents



- Interface Artifacts represent the wrappers to the hardware resources capturing the user credentials
- (Room-)Access Manager agents check whether such an access can be authorised
- User(-room) Artifacts store all the roles permanently qualified to access the building (room), along with their access privileges
- Building-State Artifact traces the people inside the building

Mechanism's Artifacts and Agents



- Appointment Artifact manages the users' appointments, storing the list of the appointments for a given room
- User Manager and Room Manager agents manage the system users
- (Room-)Admin Artifacts are used by the system administrator to introduce or delete roles and to edit the policies over time, or to handle appointments.

Outline

The Policy sub-system

- RBAC policies are designed during SODA's architectural design step
- The constraints that shape the role interaction spaces drive the design of the organisational rules
- The environment needs not to be explicitly designed
  - it is already represented in the mechanism sub-system
  - we need to model it in the analysis phase
  - so as to identify the relationships and the interactions between the two sub-systems
- Also the topological structure is implicit in the mechanism
- So, we now focus only on the design of the interaction and organisational rule entities

The Policy sub-system

- RBAC policies are designed during SODA's architectural design step
- The constraints that shape the role interaction spaces drive the design of the organisational rules
- The environment needs not to be explicitly designed
  - it is already represented in the mechanism sub-system
  - we need to model it in the analysis phase
  - so as to identify the relationships and the interactions between the two sub-systems
- Also the topological structure is implicit in the mechanism
- So, we now focus only on the design of the interaction and organisational rule entities

The Policy sub-system

- RBAC policies are designed during SODA's architectural design step
- The constraints that shape the role interaction spaces drive the design of the organisational rules
- The environment needs not to be explicitly designed
  - it is already represented in the mechanism sub-system
  - we need to model it in the analysis phase
  - so as to identify the relationships and the interactions between the two sub-systems
- Also the topological structure is implicit in the mechanism
- So, we now focus only on the design of the interaction and organisational rule entities

The Policy sub-system

- RBAC policies are designed during SODA's architectural design step
- The constraints that shape the role interaction spaces drive the design of the organisational rules
- The environment needs not to be explicitly designed
    - it is already represented in the mechanism sub-system
    - we need to model it in the analysis phase
    - so as to identify the relationships and the interactions between the two sub-systems
- Also the topological structure is implicit in the mechanism
- So, we now focus only on the design of the interaction and organisational rule entities

The Policy sub-system

- RBAC policies are designed during SODA's architectural design step
- The constraints that shape the role interaction spaces drive the design of the organisational rules
- The environment needs not to be explicitly designed
    - it is already represented in the mechanism sub-system
    - we need to model it in the analysis phase
    - so as to identify the relationships and the interactions between the two sub-systems
- Also the topological structure is implicit in the mechanism
- So, we now focus only on the design of the interaction and organisational rule entities

The Roles

- From the viewpoint of sub-system requirements our scenario involves six different roles
    - Professors, Technicians, and Administrative staff can freely access the building at any time
    - Students can
        - access the building only during the regular opening hours
        - access the Administrative staffs' and Professors' offices only if they must have an appointment
    - Visitors cannot access the building without a Guide, who is a member of the University
- Beyond these roles, the user management activity highlights the need of a new service role – the System administrator – for modifying the access privileges and managing the users' credentials

The Roles

- From the viewpoint of sub-system requirements our scenario involves six different roles
  - Professors, Technicians, and Administrative staff can freely access the building at any time
  - Students can
    - access the building only during the regular opening hours
    - access the Administrative staffs' and Professors' offices only if they must have an appointment
  - Visitors cannot access the building without a Guide, who is a member of the University
- Beyond these roles, the user management activity highlights the need of a new service role – the System administrator – for modifying the access privileges and managing the users' credentials

The Roles

- From the viewpoint of sub-system requirements our scenario involves six different roles
    - Professors, Technicians, and Administrative staff can freely access the building at any time
    - Students can
        - access the building only during the regular opening hours
        - access the Administrative staffs' and Professors' offices only if they must have an appointment
    - Visitors cannot access the building without a Guide, who is a member of the University
- Beyond these roles, the user management activity highlights the need of a new service role – the System administrator – for modifying the access privileges and managing the users' credentials

The Roles

- From the viewpoint of sub-system requirements our scenario involves six different roles
  - Professors, Technicians, and Administrative staff can freely access the building at any time
  - Students can
    - access the building only during the regular opening hours
    - access the Administrative staffs' and Professors' offices only if they must have an appointment
  - Visitors cannot access the building without a Guide, who is a member of the University
- Beyond these roles, the user management activity highlights the need of a new service role – the System administrator – for modifying the access privileges and managing the users' credentials

The Roles

- From the viewpoint of sub-system requirements our scenario involves six different roles
    - Professors, Technicians, and Administrative staff can freely access the building at any time
    - Students can
        - access the building only during the regular opening hours
        - access the Administrative staffs' and Professors' offices only if they must have an appointment
    - Visitors cannot access the building without a Guide, who is a member of the University
- Beyond these roles, the user management activity highlights the need of a new service role – the System administrator – for modifying the access privileges and managing the users' credentials

Roles & Actions

Role	**Action**
Visitor	enter, exit, ask_appointment
Student	enter, exit, ask_appointment
Professor	enter, exit, canc_appointment, set_appointment change_policy, insert_role, canc_role
Administrative staff	enter, exit, canc_appointment, set_appointment change_policy, insert_role, canc_role
Technician	enter, exit, canc_appointment, set_appointment change_policy, insert_role, canc_role
Guide	enter, exit
System administrator	enter, exit, change_policy insert_role, canc_role

Rules

Rule	Description
Guide-Rule	Guide cannot be activated together other roles (DSD constraint)
Visitor-Rule	Visitor cannot be activated together other roles (SSD constraint)
Admin-Rule	The Administrator can modify the access rules for the whole building but cannot modify the access rules for the offices
Prof-Admin-Rule	The Professor can modify the access rules for his/her office
Staff-Admin-Rule	The Administrative staff can modify the access rules for their office
Visit-Rule	Visitor can access the building only together a Guide
Building-Rule	The access to the building is possible only when the building is open to the public
Uni-Build-Rule	Professor, Technician, Administrative staff and System administrator can always access the building
App-Rule	The access to an office is granted only if the Student has an an appointment and the Professor/Administrative staff is in the office
Administration-Rule	The access to the staff office is possible only when the office is open to the public
ClassRoom-Rule	The access to a classroom is not granted during a lecture
Library-Rule	The access to the library is possible only when the library is open to the public
Lab-Rule	The access to the laboratory is possible only when the laboratory is open to the public
Department-Rule	The access to the department is possible only if the destination room grants the access

Rules&Artifacts

Rule	Description	
Guide-Rule	Guide cannot be activated together other roles (DSD constraint)	→ User Artifact
Visitor-Rule	Visitor cannot be activated together other roles (SSD constraint)	
Admin-Rule	The Administrator can modify the access rules for the whole building but cannot modify the access rules for the offices	
Prof-Admin-Rule	The Professor can modify the access rules for his/her office	→ (Room-)Admin Artifact
Staff-Admin-Rule	The Administrative staff can modify the access rules for their office	
Visit-Rule	Visitor can access the building only together a Guide	
Building-Rule	The access to the building is possible only when the building is open to the public	→ User Artifact
Uni-Build-Rule	Professor, Technician, Administrative staff and System administrator can always access the building	
App-Rule	The access to an office is granted only if the Student has an an appointment and the Professor/Administrative staff is in the office	
Administration-Rule	The access to the staff office is possible only when the office is open to the public	
ClassRoom-Rule	The access to a classroom is not granted during a lecture	User-room Artifact
Library-Rule	The access to the library is possible only when the library is open to the public	→
Lab-Rule	The access to the laboratory is possible only when the laboratory is open to the public	Appointment Artifact
Department-Rule	The access to the department is possible only if the destination room grants the access	

Conclusion

- In this work we have shown how an AOSE methodology supports the design of an RBAC-MAS system
- with the purpose of
  - identifying the RBAC-MAS requirements
  - showing the suitability of the separation between policy and mechanism:
    - the mechanism sub-system is designed as general as possible, since its structure is basically stable and reusable as is in other applications
    - policies are generally tied to the application domain, so they have to be re-designed each time
  - testing the suitability of SODA in the engineering process of an RBAC system

Conclusion

- In this work we have shown how an AOSE methodology supports the design of an RBAC-MAS system
- with the purpose of
  - identifying the RBAC-MAS requirements
  - showing the suitability of the separation between policy and mechanism:
    - the mechanism sub-system is designed as general as possible, since its structure is basically stable and reusable as is in other applications
    - policies are generally tied to the application domain, so they have to be re-designed each time
  - testing the suitability of SODA in the engineering process of an RBAC system

Conclusion

- In this work we have shown how an AOSE methodology supports the design of an RBAC-MAS system
- with the purpose of
  - identifying the RBAC-MAS requirements
  - showing the suitability of the separation between policy and mechanism:
    - the mechanism sub-system is designed as general as possible, since its structure is basically stable and reusable as is in other applications
    - policies are generally tied to the application domain, so they have to be re-designed each time
  - testing the suitability of SODA in the engineering process of an RBAC system

Conclusion

- In this work we have shown how an AOSE methodology supports the design of an RBAC-MAS system
- with the purpose of
  - identifying the RBAC-MAS requirements
  - showing the suitability of the separation between policy and mechanism:
    - the mechanism sub-system is designed as general as possible, since its structure is basically stable and reusable as is in other applications
    - policies are generally tied to the application domain, so they have to be re-designed each time
  - testing the suitability of SODA in the engineering process of an RBAC system

Future Work

- Improving the methodology in several directions:
  - to support the design of secure agent-oriented systems since the earliest Requirement Analysis step
  - to develop a language for SODA rules able to capture all the relevant RBAC permissions and constraints
  - to study more deeply the access control issues related to artifacts

Future Work

- Improving the methodology in several directions:
  - to support the design of secure agent-oriented systems since the earliest Requirement Analysis step
  - to develop a language for SODA rules able to capture all the relevant RBAC permissions and constraints
  - to study more deeply the access control issues related to artifacts

Future Work

- Improving the methodology in several directions:
  - to support the design of secure agent-oriented systems since the earliest Requirement Analysis step
  - to develop a language for SODA rules able to capture all the relevant RBAC permissions and constraints
  - to study more deeply the access control issues related to artifacts

Future Work

- Improving the methodology in several directions:
  - to support the design of secure agent-oriented systems since the earliest Requirement Analysis step
  - to develop a language for SODA rules able to capture all the relevant RBAC permissions and constraints
  - to study more deeply the access control issues related to artifacts

Acknowledgements

This work has been supported by the “Social networks and knowledge construction promotion in e-learning contexts” project (229692-CP-1-2006-1-IT MINERVA-M, http://projects.deis-ce.unibo.it/minerva/). This is an e-learning project funded with support from Minerva action of the Socrates programme of the European Commission. This work reflects the views only of the authors, and the Commission can not be held responsible for any use which be made of the information contained therein

RBAC-MAS & SODA: Experimenting RBAC in AOSE

Ambra Molesini & Enrico Denti & Andrea Omicini
{ambra.molesini, enrico.denti, andrea.omicini}@unibo.it

ALMA MATER STUDIORUM—Università di Bologna

ESAW 2008, Saint-Étienne, France, 25th September 2008