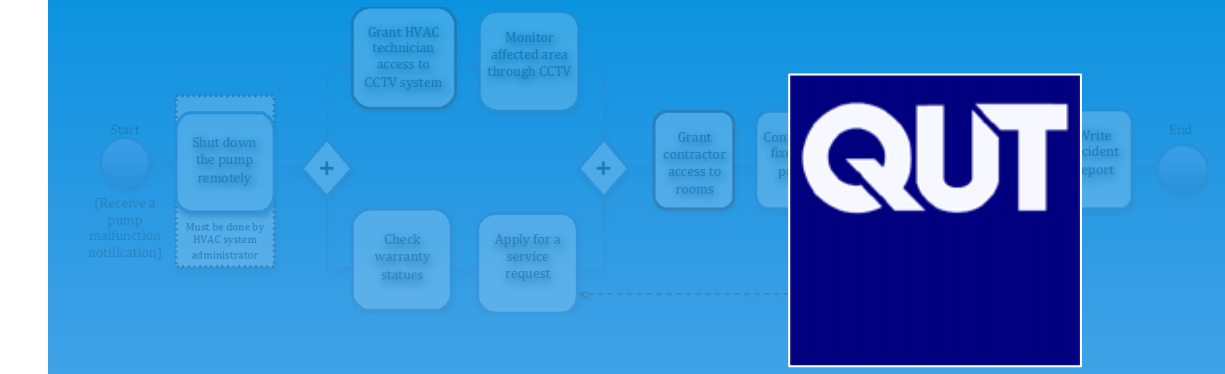
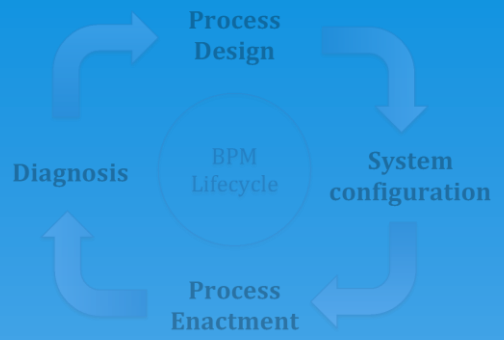




Authorisation Management in Business Process Environments

Khalid Adnan Alissa

Supervisors

Emeritus Prof. Ed Dawson

Dr. Jason Reid

Airports of the Future

QUT ife Institute for Future Environments

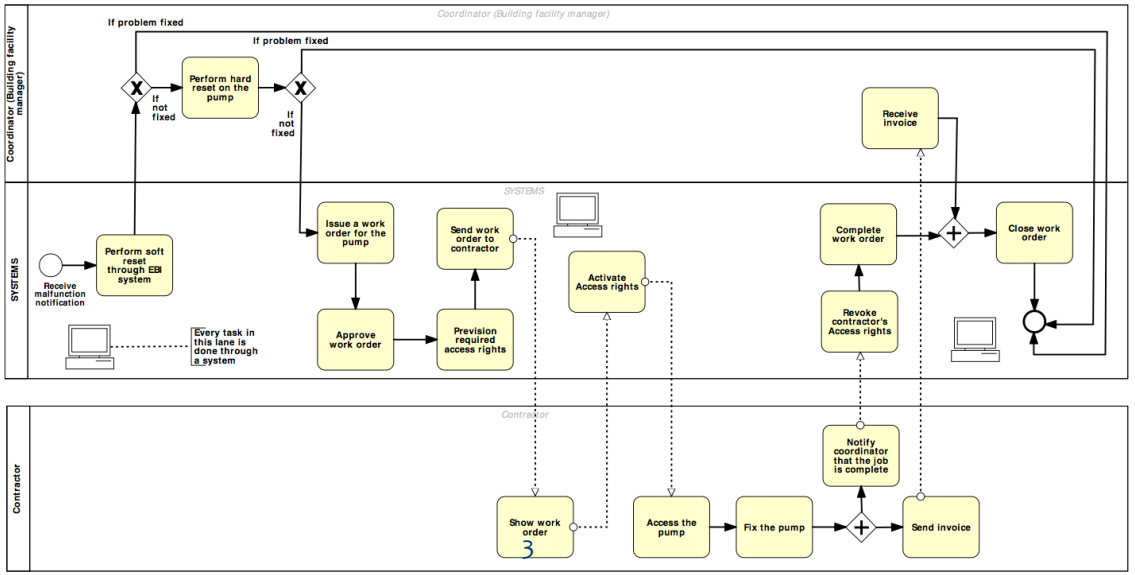
Agenda

* Introduction
  * Background
  * Area of research
  * Aims
  * Contributions
* Authorisation management for BPM
  * Example scenario
  * Characteristics analysis
  * Literature review
* BP-TRBAC
  * Concept
  * Formalisation
  * Example
  * Discussion
  * review
* SPCC
  * YAWL
  * SPCC
  * Implementation
  * Results
* BP-XACML
  * Policy structure
  * Policy Model
  * Policy semantics
  * Discussion
* Conclusion
  * Contribution summery
  * Future direction

Introduction

Business Process Management

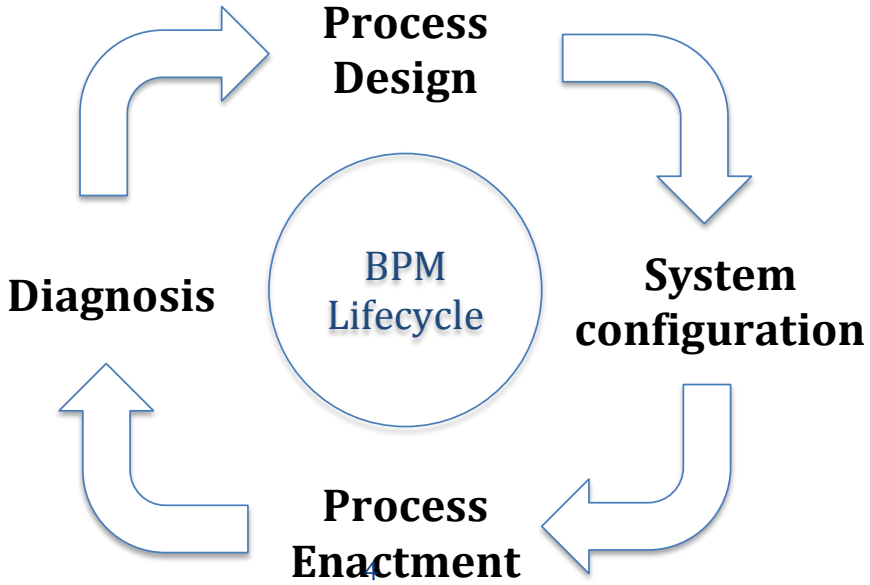
* Business processes is a set of tasks performed in a structured flow that supports the business goals.
* Business processes are repeatable, and consistent.



Business Process Management

* BPM includes concepts, methods,

and techniques to support the design, administration, configuration, and analysis of business processes



Authorisation Management

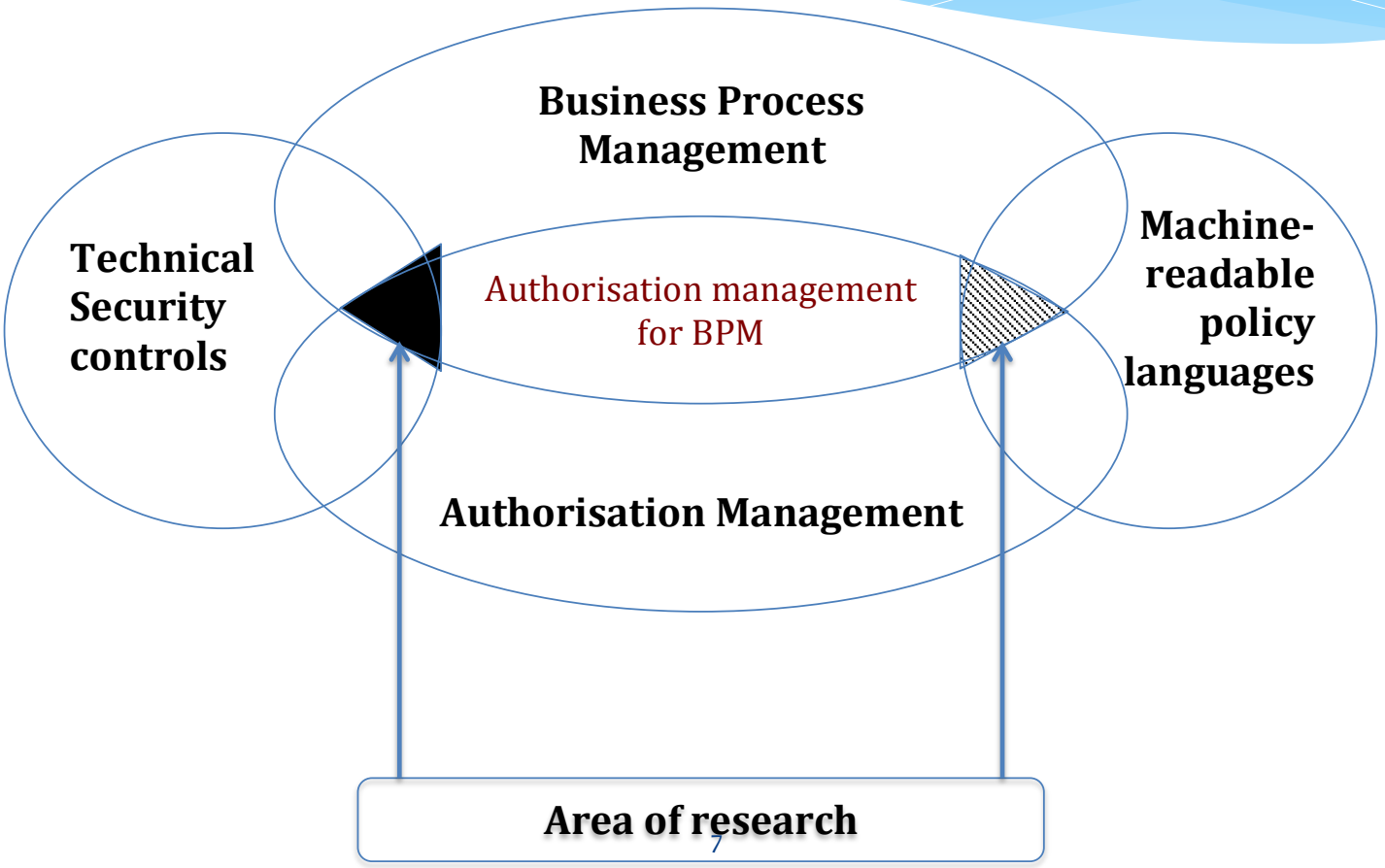
* Authorisation is a process that involves granting or denying permission to an authenticated entity to access a resource in a particular way (e.g. reading or writing a file)

Authorisation Management

* Authorisation Management is concerned with writing an authorisation policy, storing the policy, managing the policy, and enforcing the policy.

* Our focus is on:
  - Enforcing the policy: an authorisation model that enforces the policy
  - Writing the policy: the language for the authorisation policy.

Research Area



Aims

This research aims to enable unified authorisation management in environments that include business process systems

Aims

* This research has two sub-aims:
  - To provide an authorisation model for business process environments that control authorisation requests and enforces the authorisation policies.
  - To provide a structured, machine-readable language that has the ability to represent authorisation policies for business processes.

Contributions

The first contribution is BP-TRBAC:

* BP-TRBAC extends RBAC to support business process authorisation constraints.
* It is a a unified independent enterprise-wide authorisation model
* Khalid Alissa, Jason Reid, Farzad Salim, and Ed Dawson. Business Process Task-Role-Baseed Access Control Model (BP-TRBAC). submitted to the International Journal of Cooperative Information Systems. World Scientific Publishing Company.

Contributions

The second contribution is BP-XACML:

* BP-XACML extends XACML to support business process authorisation policies.
* Also includes a policy model for BP-XACML.
* Khalid Alissa, Jason Reid, Ed Dawson, and Farzad Salim. BP-XACML: an authorisation policy language for business processes. In Douglas Stebila and Ernest Foo, The 20th Australasian Conference on Information Security and Privacy. Brisbane. Springer. 2015.

Agenda

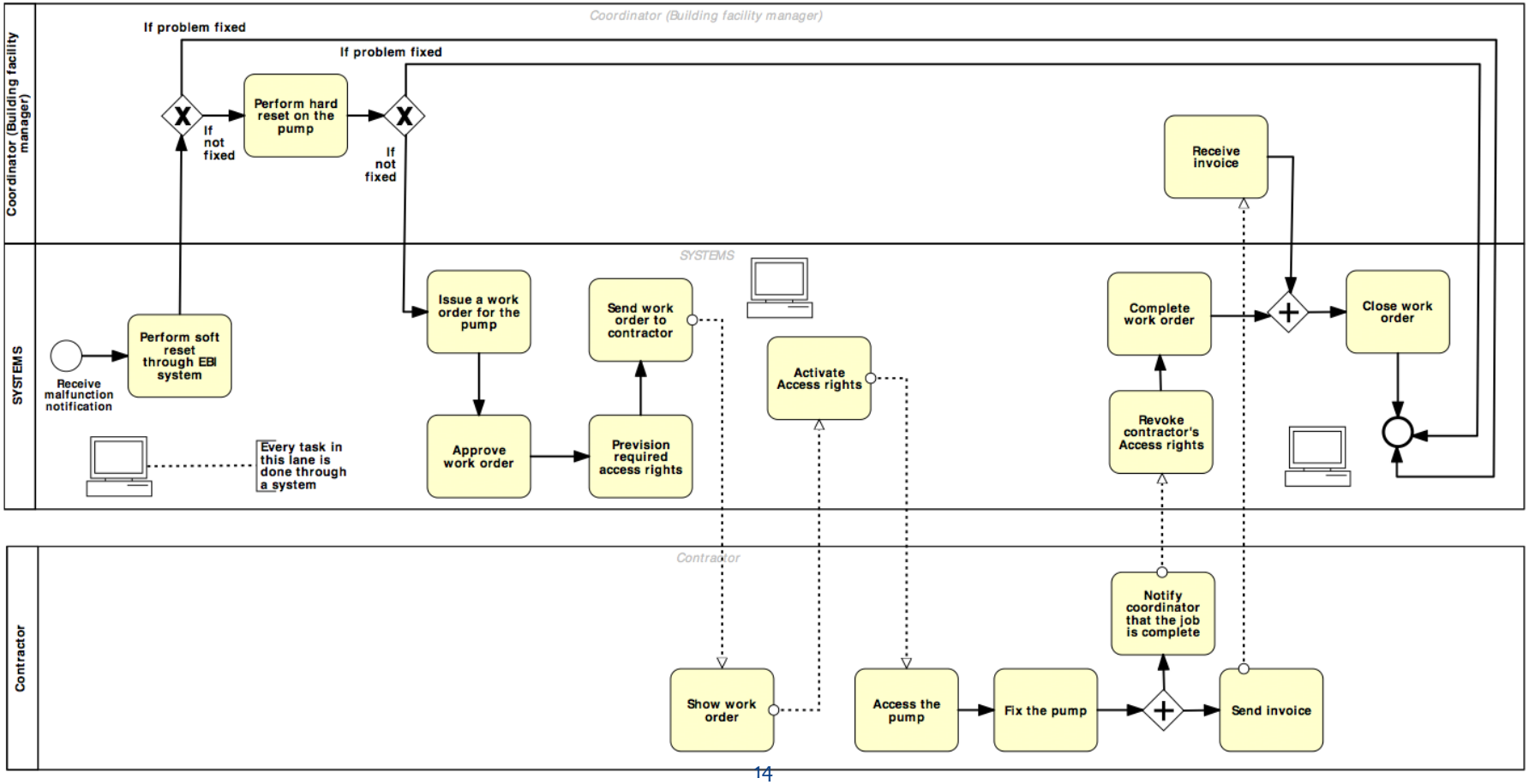
* Introduction
  * Background
  * Area of research
  * Aims
  * Contributions
* Authorisation management for BPM
  * Example scenario
  * Characteristics analysis
  * Literature review
* BP-TRBAC
  * Concept
  * Formalisation
  * Example
  * Discussion
  * review
* SPCC
  * YAWL
  * SPCC
  * Implementation
  * Results
* BP-XACML
  * Policy structure
  * Policy Model
  * Policy semantics
  * Discussion
* Conclusion
  * Contribution summery
  * Future direction

Example scenario

* Real-life example business process
* Security-sensitive organisation
* Gathered data from process stakeholders

Authorisation management for BPM

Example scenario



Example scenario

Business rules and authorisation policies associated with this process are:

* The task ‘soft reset’ should not be performed unless a malfunction notification was received.
* Only the role ‘coordinator’ is allowed to issue work orders.
* A work order can be closed only after receiving both work-order completion and an invoice.
* Only the person who issued a certain work order is allowed to close it.
* No person is allowed to perform ‘issue work order’ and ‘approve work order’ for the same ‘work order’.
* No person is allowed to have both roles ‘coordinator’ and ‘contractor’.

Characteristics analysis

A business process access control model should support the following characteristics:

- Role-based access control.
- Static and dynamic separation of duties.
- Active access control.
- Instance-level restrictions.
- Task-based access control.
- Supports workflow and non-workflow tasks.

Authorisation management for BPM

Literature Review

Authorisation Model

	Compression criteria	RBAC*	WAM	FWAM	SRBWM	T-RBAC	W-RBAC	MSoD	WSession	SOWAC	Str. & Mend.	AW-RBAC
1	An independent access control model	Yes	Yes	Yes	Yes	Yes	No	No	No	Yes	No	Yes
2	Supports Role-based access control	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
3	Supports Task-based authorisation	No	No	No	Yes	Yes	No	No	Yes	Yes	Yes	No
4	Supports Active access control	No	Yes	Yes	Yes	Yes	No	No	No	Yes	Yes	No
5	Supports SoD on instance-level	No	No	No	No	Yes	Yes	Yes	Yes	No	Yes	No
6	Supports History-based SoD on IL**	No	No	No	No	No	Yes	Yes	No	No	Yes	No
7	Supports BoD on instance-level	No	No	No	No	No	Yes	Yes	Yes	No	Yes	No
8	Support non-workflow tasks	Yes	No	No	No	Yes	No	Yes	No	No	No	Yes
9	Support workflow tasks	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

* The RBAC model used in this table is NIST RBAC2

** IL= Instance-Level

Characteristics analysis

Policy Language

Authorisation policy language for business processes should be able to represent:

- Users.
- Roles
- Operations.
- Tasks.
- Tasks instance.
- Instance-level restrictions.
- Role-level-SoD restrictions.

Literature Review

Policy Language

* Authorisation policies are initially authored in plain, human language.
* To enforce the policy it needs to be translated to a machine-enforceable language, so systems are able to interpret the policy.
* There have been a number of investigations and studies on machine readable structured language
* The formalism of languages differs:
  * For example, some languages are logical based languages, while other languages are rule-based languages.

Literature Review

Policy Language

None of the languages is suitable for business process authorisation policies.

Either propose a new language or extend a language that already exists to be able to support authorisation policies for business processes.

Some languages can be extended

For example XACML does not support RBAC, then a new profile (RBAC-XACML) extended XACML to support RBAC

Literature Review

Policy Language

In our case we need an extension to support business process authorisation policies.

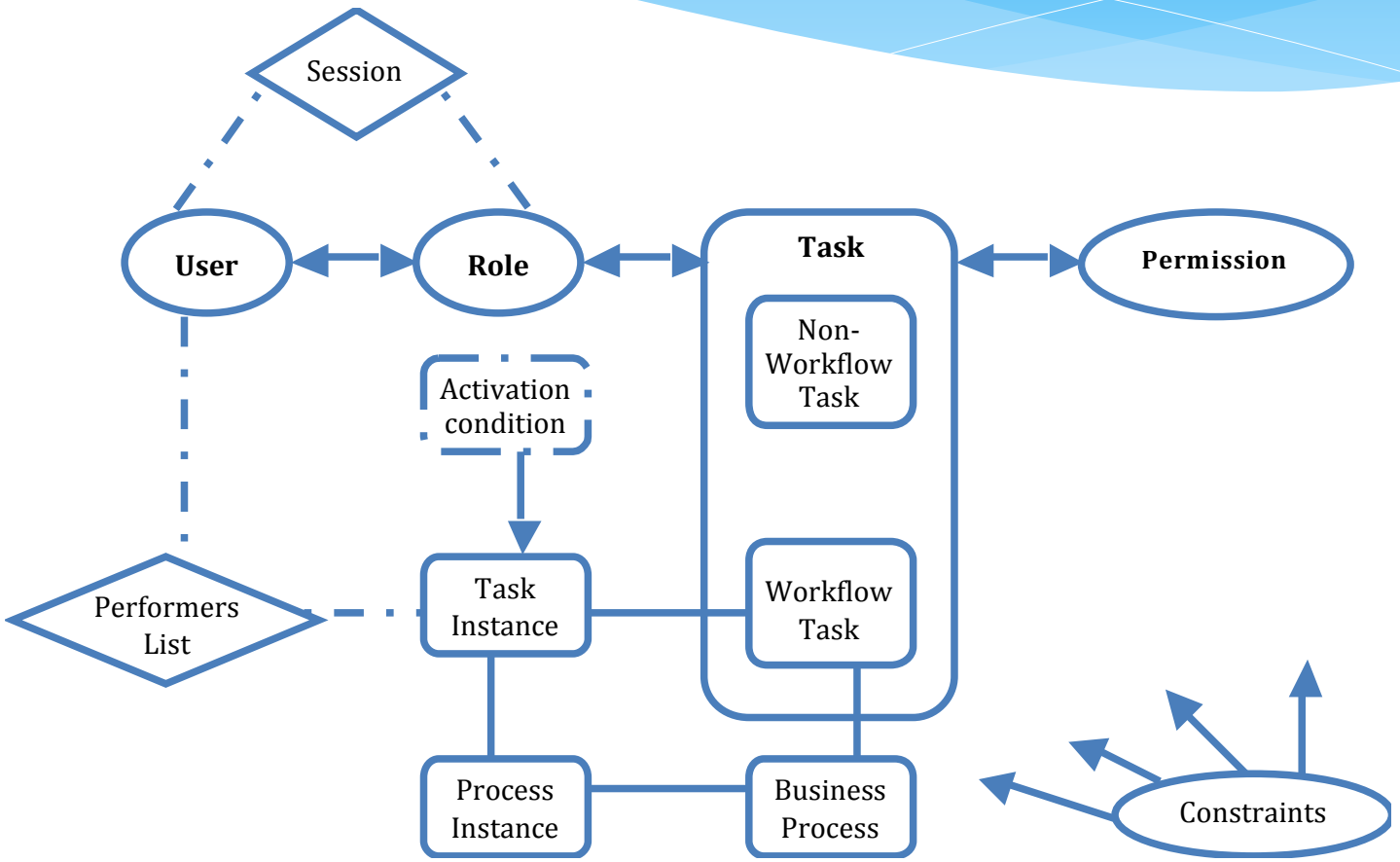
Currently there is no published work that aims to extend XACML to support business process policies

There are several published works that extend XACML to support different models but none of them focus on 'business processes'.

Agenda

* Introduction
  * Background
  * Area of research
  * Aims
  * Contributions
* Authorisation management for BPM
  * Example scenario
  * Characteristics analysis
  * Literature review
* BP-TRBAC
  * Concept
  * Formalisation
  * Example
  * Discussion
  * review
* SPCC
  * YAWL
  * SPCC
  * Implementation
  * Results
* BP-XACML
  * Policy structure
  * Policy Model
  * Policy semantics
  * Discussion
* Conclusion
  * Contribution summery
  * Future direction

Business Process Task-Role-Based access control (BP-TRBAC)



Conceptualisation

* We have conceptualised BP-TRBAC using ORM.
* Helped to identify details of model elements and relationships.
* ORM model can be found in your handouts.

BP-TRBAC

Formalisation

* A formal description of BP-TRBAC using set theory.
* Described BP-TRBAC and its aspects in formal way
* Including users, roles, permissions, tasks, task-instances, instance-level restrictions, activation conditions, and role-level SoD.
* We will show some aspects, for more refer to the handouts.

BP-TRBAC

Formalisation

Task and Task-instance:

* Task type (ty) can be either workflow task : (ty=w), or non-workflow task : (type=n)
* Task (t) is a tuple of id∈N, task type ty∈Ty, and a set of permissions p⊆P

$$t = (id, ty, p) \in N \times T y \times P$$

* Permission-task assignment (PTA) is a many-to-many mapping permissions-to-tasks assignment relation.

$$PTA \subseteq P \times T$$

* Task-role assignment (TRA) is a many-to-many mapping tasks-to-roles assignment relation.

$$T RA \subseteq T \times R$$

* A task instance (ti) is a tuple of st ∈ ST : ST = {unassigned, active, completed}, task t ∈ T, and a number n ∈ N. The set of all task instances is 'TI'.

$$ti = (st, t, n) : \forall (bp, n'), n = n'$$

Formalisation

Instance-level Restrictions:

* An ‘ir’ rule is written as a tuple of the two task instances and the type of the restriction (type). The set of all instance-level restrictions in a system is referred to as ‘IR’.

$$\mathbf{ir = (ti_1, ti_2, type) : ti_1 \Lambda ti_2 \in T\ I \text{ and } type \in \{SoD, BoD\}, ir \in IR}$$

* An ir means that the user can not be part of the performers list of both tasks instance, unless each belongs to different instance.

$$\mathbf{ir = (t1_i, t4_j, SoD) \rightarrow u \in pl(t1_i) \text{ and } u \in pl(t4_j) \text{ iff } I \neq j}$$

BP-TRBAC

Example

Rules from the example with expression using the introduced formal representations:

* The task ‘soft reset’ should not be performed unless malfunction notification was received.

(AC(soft reset)= true iff st(receive malfunction notification)=completed).

* Only the role “coordinator” is allowed to issue work orders.

((issue work order, coordinator) ∈ T RA)

* No work order can be closed until receiving both a ‘work order completion’ and an invoice.

(AC(close work order)=true iff (st(complete work order)=completed ∧ st(receive invoice)=completed)).

Example

* Only a person who issued a certain work order is allowed to close it

(ir=(issue work order, close work order, BoD)).

$u \in pl(\text{close work order}_x)$ iff $u \in pl(\text{issue work order}_x)$.

* No person is allowed to perform 'issue work order' and 'review work order' for the same 'work order'.

(ir=(issue work order, review work order, SoD)).

if $u \in pl(\text{issue work order}_x) \rightarrow u \notin pl(\text{review work order}_x)$.

Discussion

* Point of strength for BP-TRBAC: combining all characteristics, and maintaining RBAC
* BP-TRBAC is a unified model that is in charge of all authorisation requests' decisions
* In terms of implementation, activation conditions can be part of the authorisation system itself, or through a cooperative interaction between workflow system and authorisation system

BP-TRBAC

Discussion

	Compression criteria	RBAC*	WAM	FWAM	SRBWM	T-RBAC	W-RBAC	MSoD	WSession	SOWAC	Str. & Mend.	AW-RBAC	BP-TRBAC
1	An independent access control model	Yes	Yes	Yes	Yes	Yes	No	No	No	Yes	No	Yes	Yes
1	Supports Role-based access control	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
2	Supports Task-based authorisation	No	No	No	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes
3	Supports Active access control	No	Yes	Yes	Yes	Yes	No	No	No	Yes	Yes	No	Yes
4	Supports SoD on instance-level	No	No	No	No	Yes	Yes	Yes	Yes	No	Yes	No	Yes
5	Supports History-based SoD on IL**	No	No	No	No	No	Yes	Yes	No	No	Yes	No	Yes
6	Supports BoD on instance-level	No	No	No	No	No	Yes	Yes	Yes	No	Yes	No	Yes
7	Support non-workflow tasks	Yes	No	No	No	Yes	No	Yes	No	No	No	Yes	Yes
8	Support workflow tasks	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

* The RBAC model used in this table is NIST RBAC2

** IL= Instance-Level

Agenda

* Introduction
  * Background
  * Area of research
  * Aims
  * Contributions
* Authorisation management for BPM
  * Example scenario
  * Characteristics analysis
  * Literature review
* BP-TRBAC
  * Concept
  * Formalisation
  * Example
  * Discussion
  * review
* SPCC
  * YAWL
  * SPCC
  * Implementation
  * Results
* BP-XACML
  * Policy structure
  * Policy Model
  * Policy semantics
  * Example
  * Discussion
* Conclusion
  * Contribution summery
  * Future direction

A use case

* We have identified a shortcoming in YAWL as a workflow system.
  * Process modeler has the ability to assign roles to task, who is not a security expert. The assignments might not be in compliance with the policy.
* We have identified a subset of BP-TRBAC to produce a security policy compliance checker (SPCC).
* SPCC is a use-case of BP-TRBAC, it work as a compliance checker.

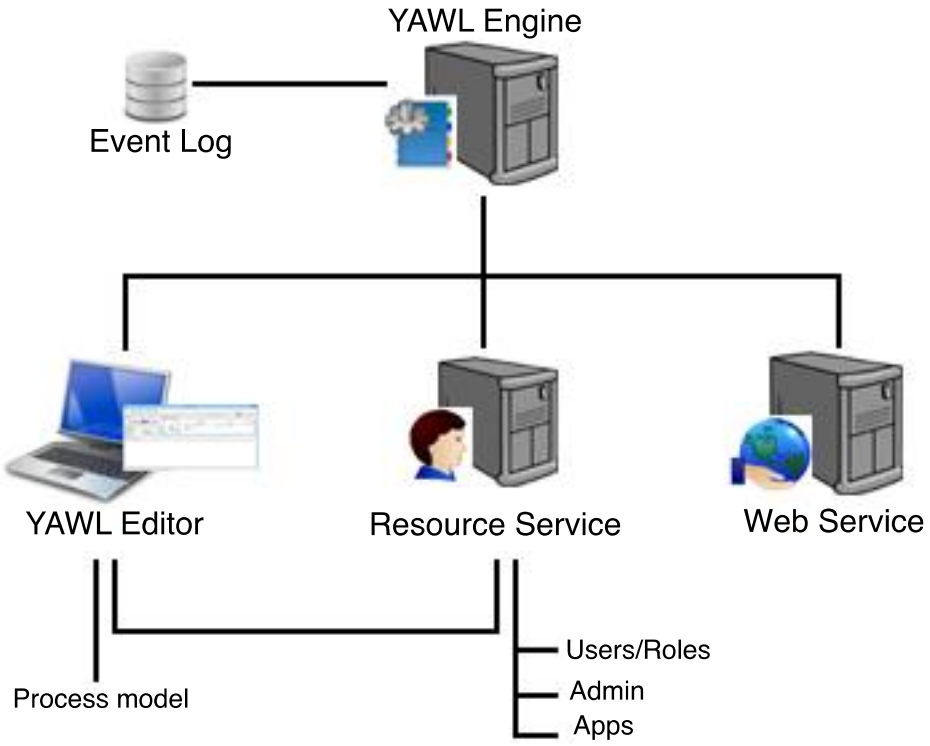
SPCC

YAWL

* YAWL is a workflow system
* Uses YAWL modeling language.
* Language can automatically be translated to a working system.

SPCC

YAWL

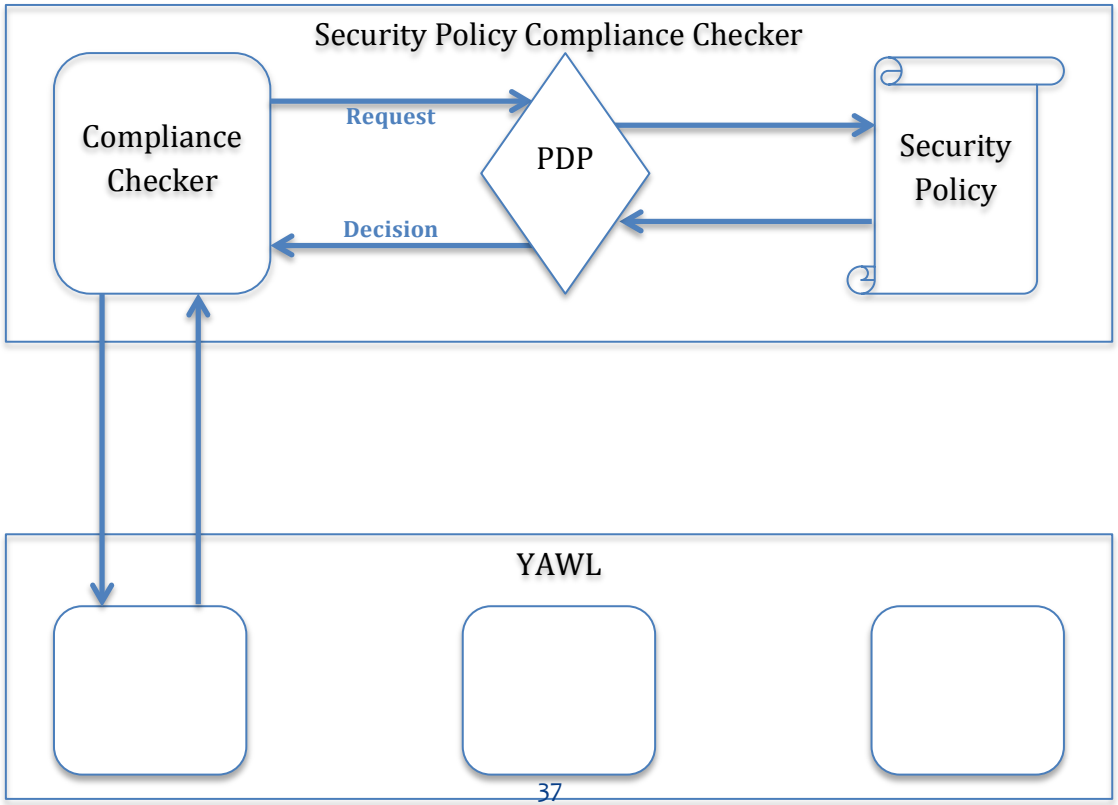

YAWL Engine
Event Log
YAWL Editor
Resource Service
Web Service
Process model
Users/Roles
Admin
Apps

Security Policy Compliance Checker

SPCC takes in the process modeler's assignment of a role to perform a task, and checks if that assignment is in compliance with the authorisation policy.

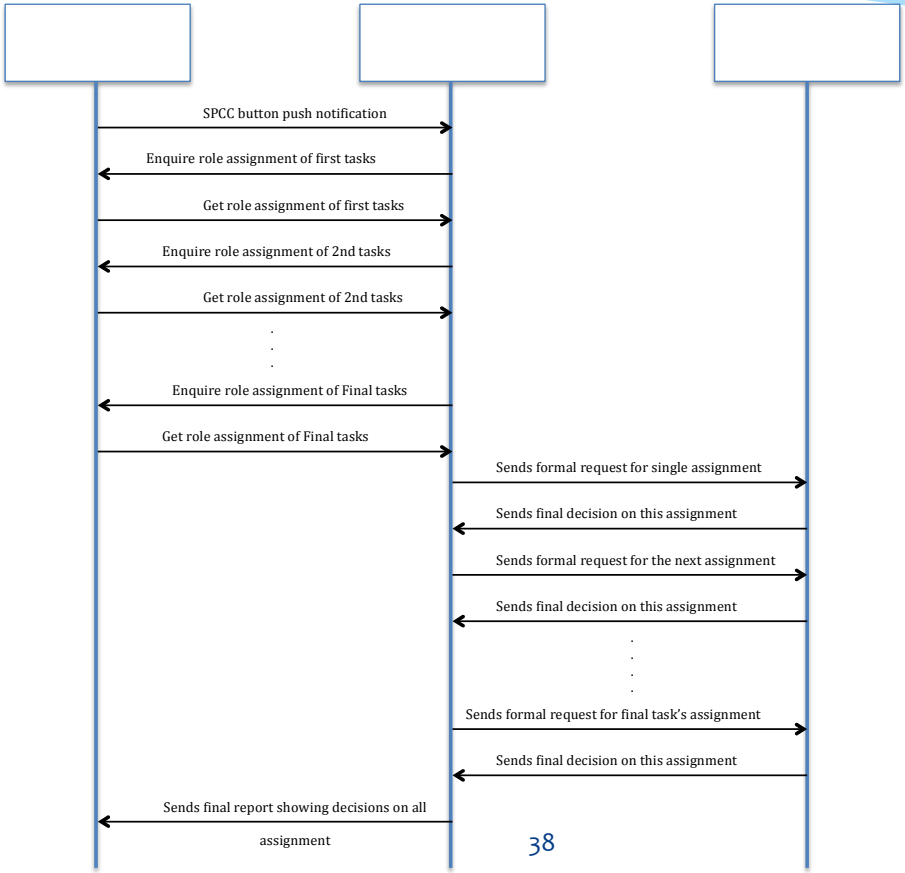
Security Policy Compliance Checker (SPCC)

Architecture:



Security Policy Compliance Checker (SPCC)

Sequence:



Security Policy Compliance Checker (SPCC)

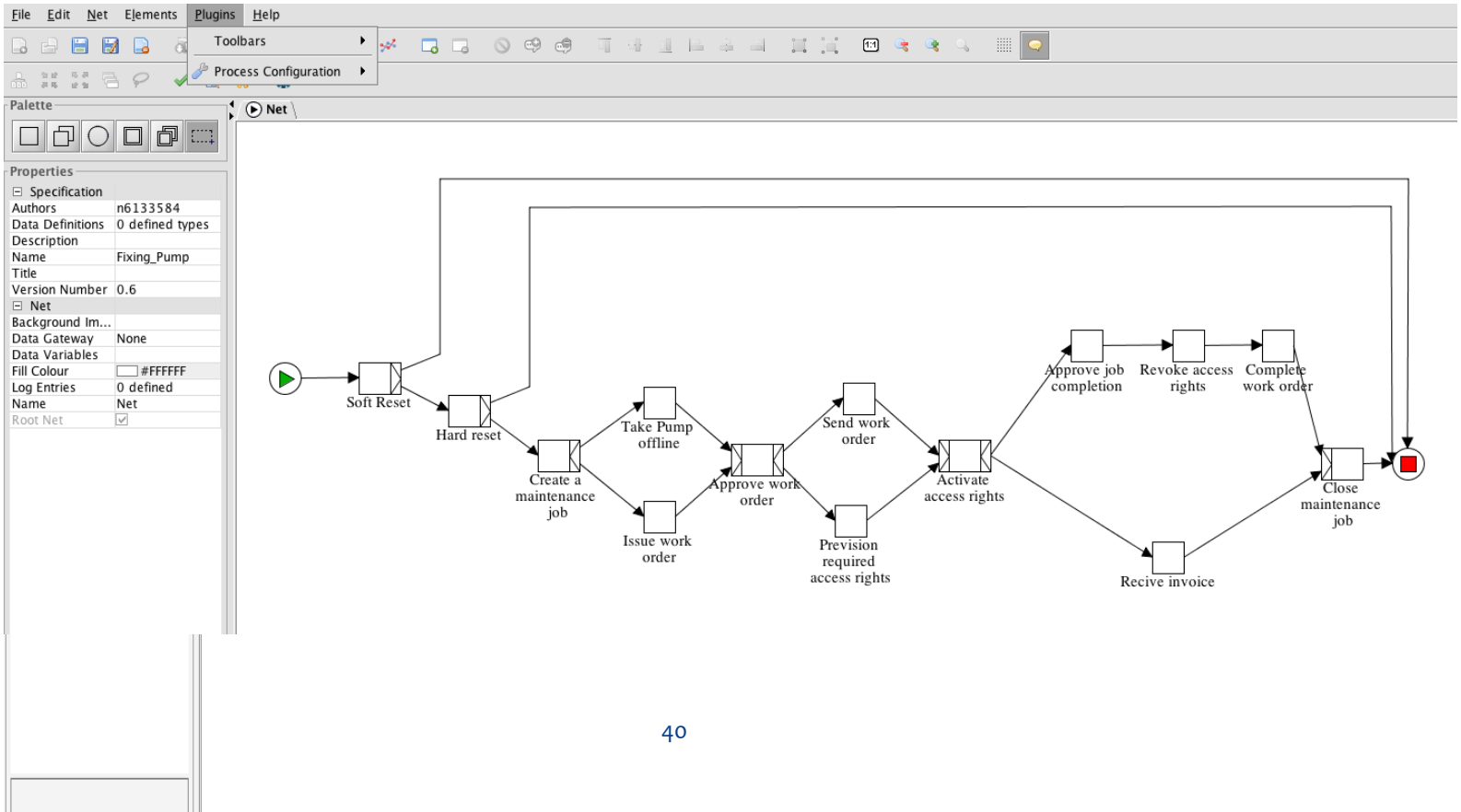
Data Structure:

* Implementation was to test SPCC. We defined our own structure
* Building this policy structure helped us to recognise the need for a structured language.
* This opportunity helped us realise the elements and the structure of such a language

SPCC

Implementation

* YAWL Plug-in interface.



SPCC

Results

SPCC

Security Policy Compliance Checker

* SPCC as an addition to YAWL, helps in making sure that the modeler assignments are in compliance with the policy before run time.

* SPCC originally is meant to check assignments requests against the organisation's actual policy. There is a need for a standardised, structured language, such as XACML for BPM authorisation policies.

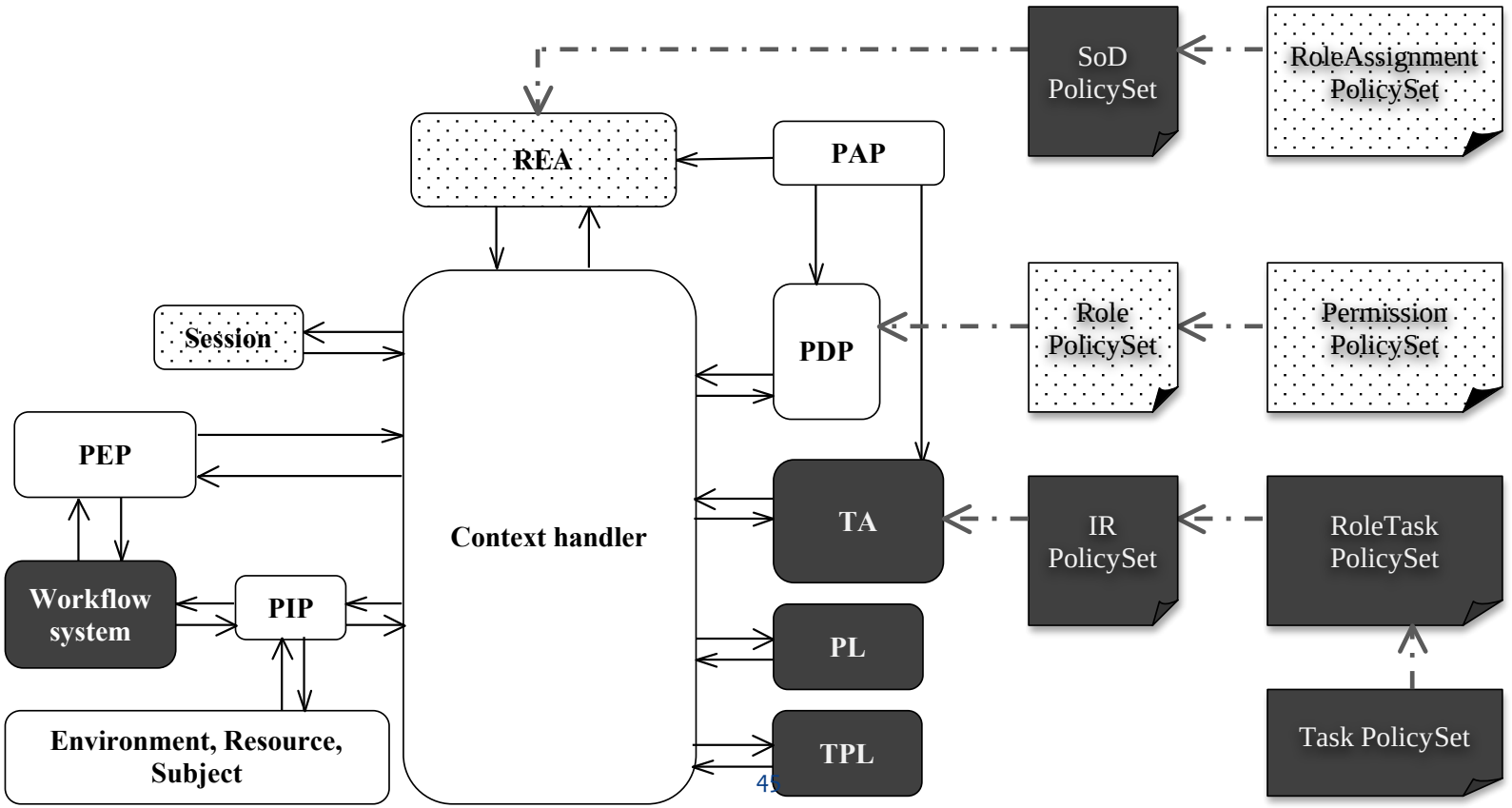
Agenda

* Introduction
  * Background
  * Area of research
  * Aims
  * Contributions
* Authorisation management for BPM
  * Example scenario
  * Characteristics analysis
  * Literature review
* BP-TRBAC
  * Concept
  * Formalisation
  * Example
  * Discussion
  * review
* SPCC
  * YAWL
  * SPCC
  * Implementation
  * Results
* BP-XACML
  * Policy structure
  * Policy Model
  * Policy semantics
  * Discussion
* Conclusion
  * Contribution summery
  * Future direction

Business Process Authorisation Policy Language

* There is a need for a structured machine-readable authorisation policy language for business process environments.
* XACML is a structured machine-readable authorisation policy language, but it does not support business process.
* XACML is platform independent authorisation policy language, consistent, enterprise-wide policies enforcement
* RBAC-XACML supports RBAC but not business processes.
* There is a need for a new profile that extends XACML to support business process authorisation policies.

XACML, RBAC-XACML, and BP-XACML



Policy Structure

Requests & Decisions:

* The Request (RQ) is in the form of {S,O,A}
* In BP-XACML there are three types of resource (roles, tasks, normal resources) whose related policies are defined in three different policy sets
* In the context of the request, the interpretation of S, O and A are different for each type.
* Because of this, each type of request is processed by a different authority.
* The decision (DS) will be either {Allow}, {Deny}, or {Not applicable} if no matching policies are found.

Policy Structure

* An authorisation policy may contain multiple authorisation rules (AR), which are the basic building blocks for stating authorisation restrictions.
* Each AR consists of four elements: Subject, Object, Action, and Condition, the evaluation of which results in a Allow or Deny decision

$$\mathbf{AR = \{S, O, A, C\} \rightarrow \{Allow, Deny\}}$$

* Action (A) is implementation specific. Condition (C) is a boolean expression that is evaluated based on the value of variables determined at run time as either true or false.

Policy Structure

Policy Sets:

* 'Policy sets' are used to group related policies, which groups related access control rules.
* A 'Policy set' also contains a target, and a policy-combining algorithm. It may also contain other policy sets included by reference
* BP-XACML includes seven types 'Policy sets'
* Three 'Policy sets' adopted from RBAC-XACML (Role<PolicySet>, Permission<PolicySet>, and RoleAssignment<PolicySet>)
* Four 'Policy sets' newly introduced (SoD<PolicySet>, Task<PolicySet>, RoleTask<PolicySet>, IR<PolicySet>).

Policy Structure

Policy Sets:

* Standard RBAC request (through PDP):

  Role<PolicySet> & Permission<PolicySet>

* Role Activation request (through REA):

  SoD<PolicySet> & RoleAssignment<PolicySet>

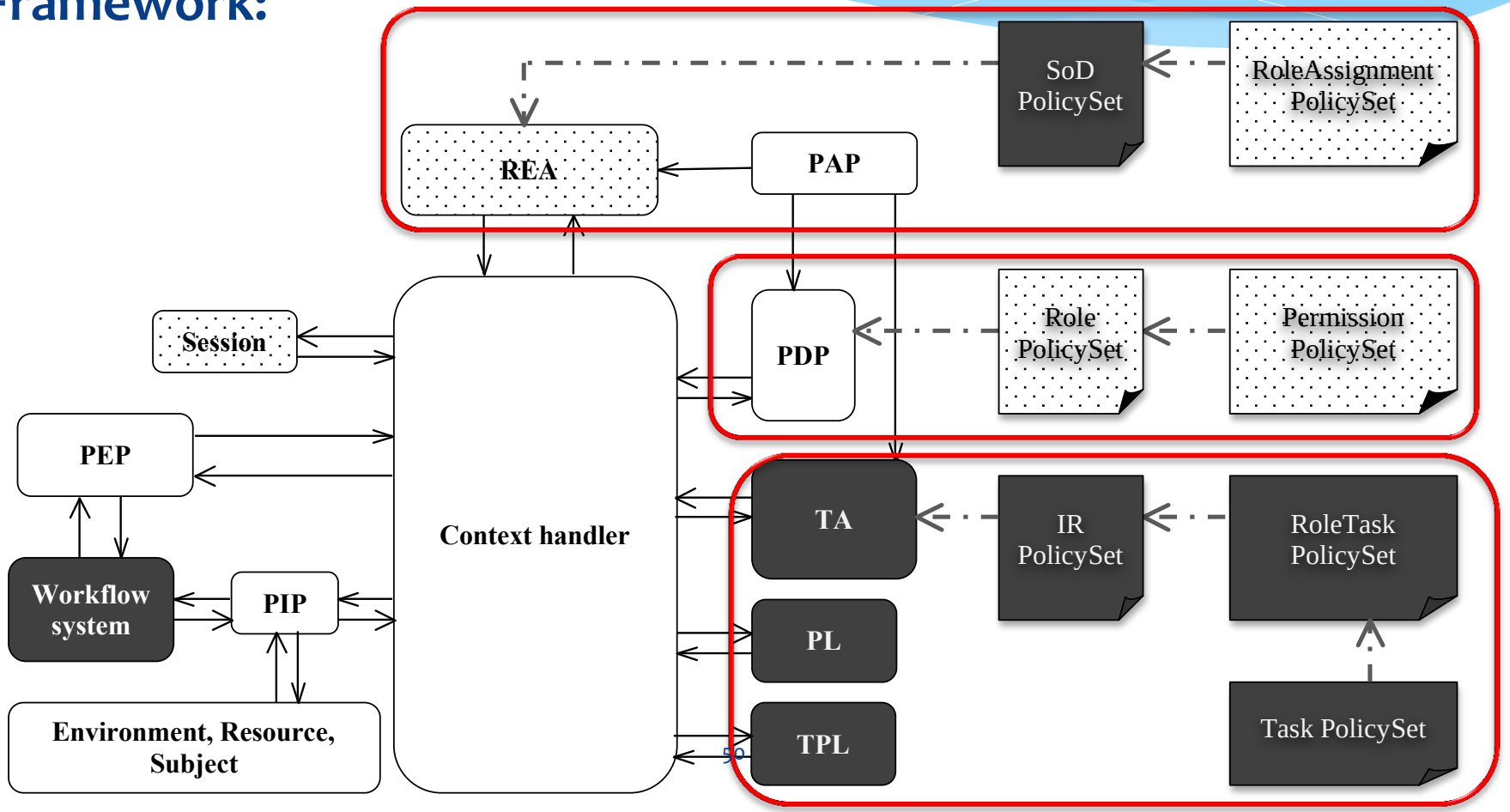
* Task Performance requests (through TA):

  IR<PolicySet>, RoleTask<PolicySet>, & Task<PolicySet>

BP-XACML

Policy Model

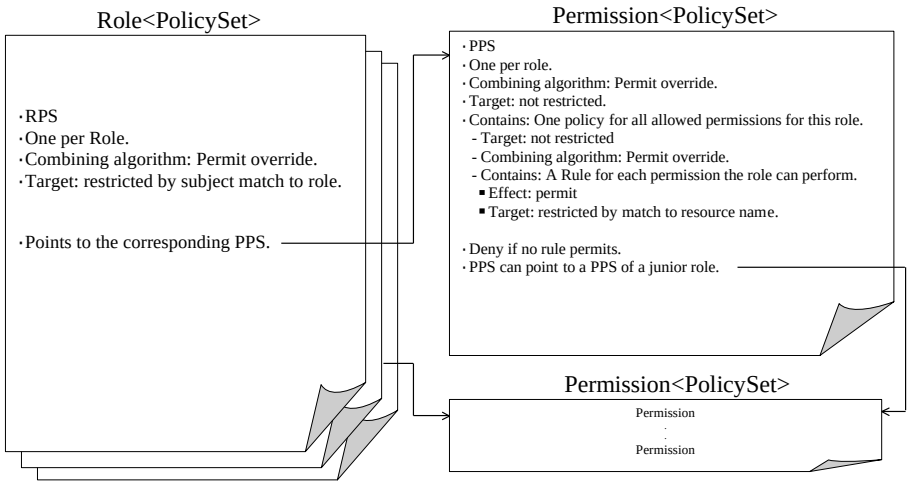
Framework:



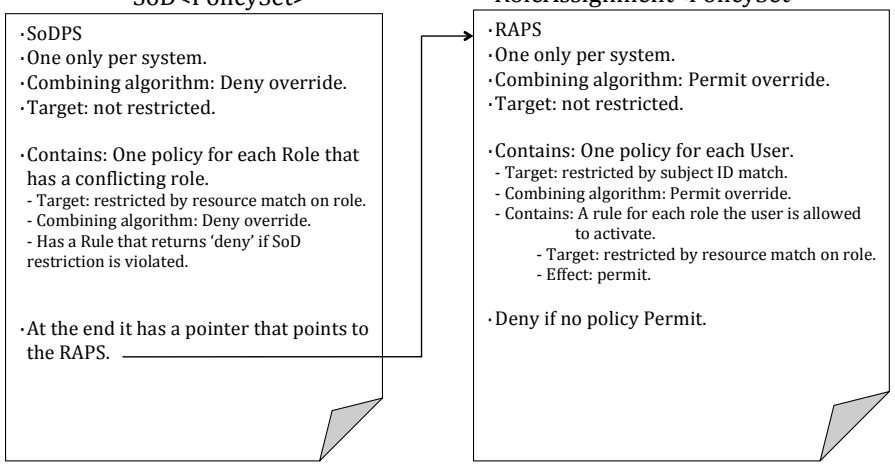
BP-XACML

Policy Structure

Standard RBAC Requests



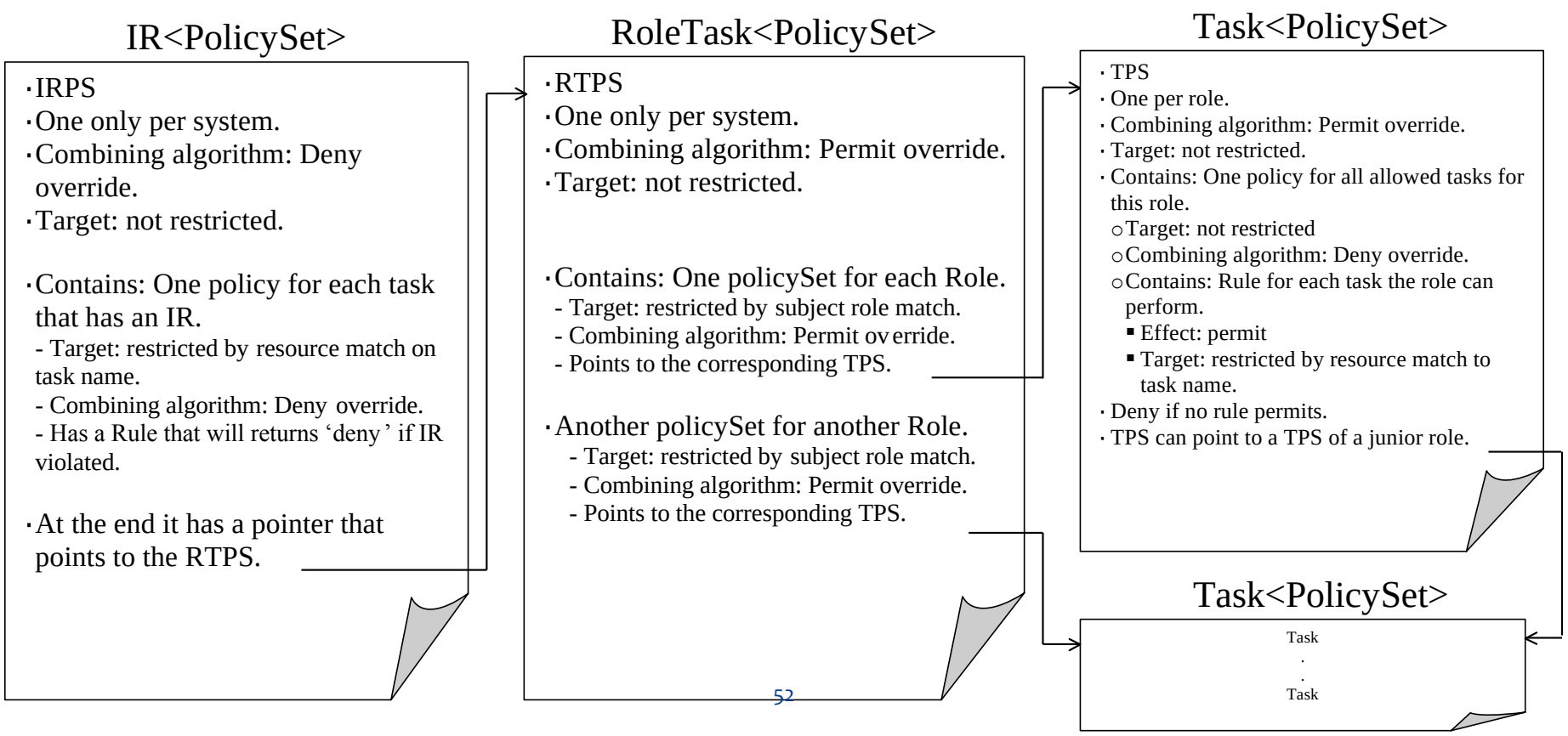
Role activation Requests



BP-XACML

Policy Structure

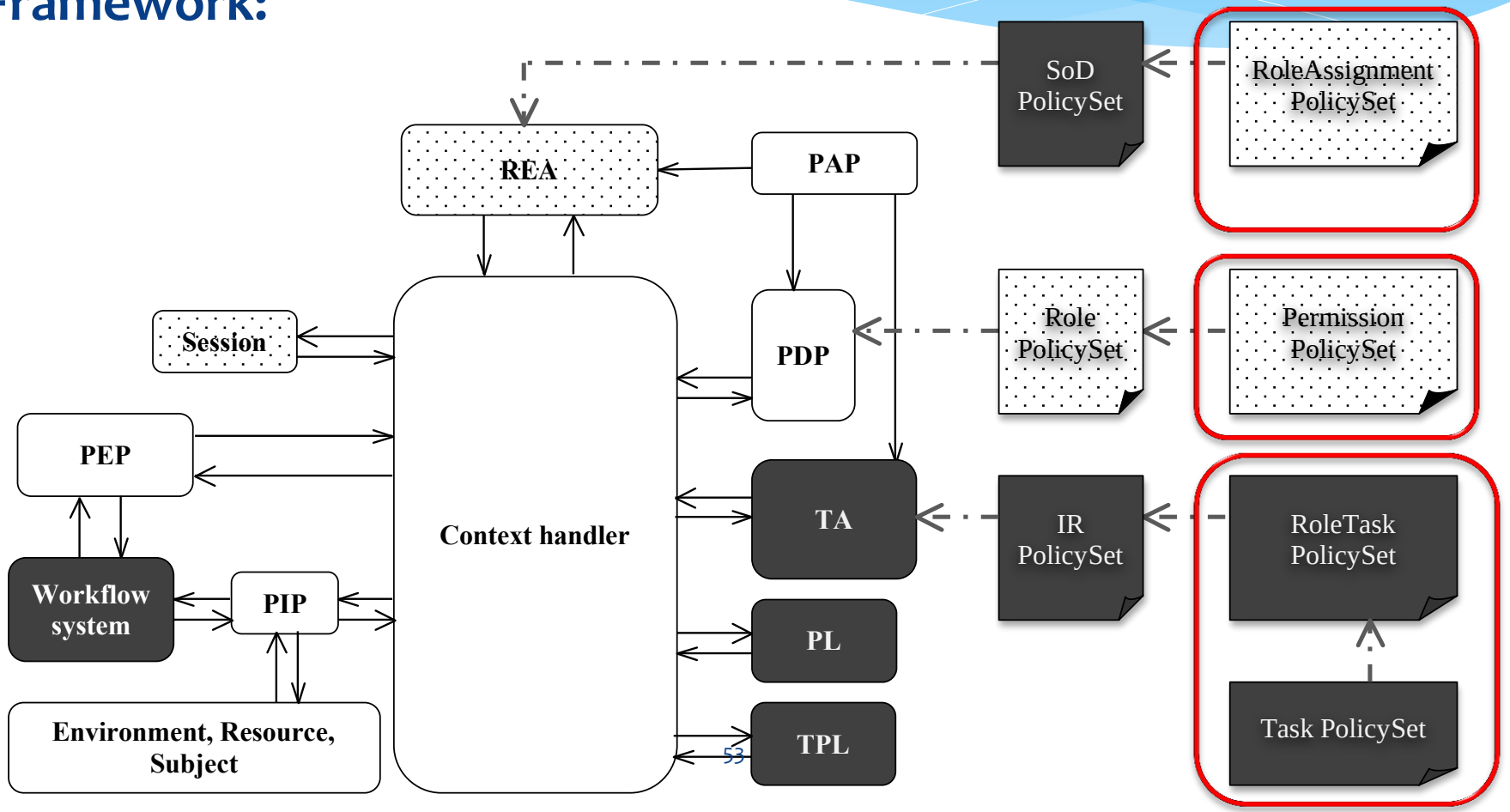
Task Performance Requests



BP-XACML

Policy Model

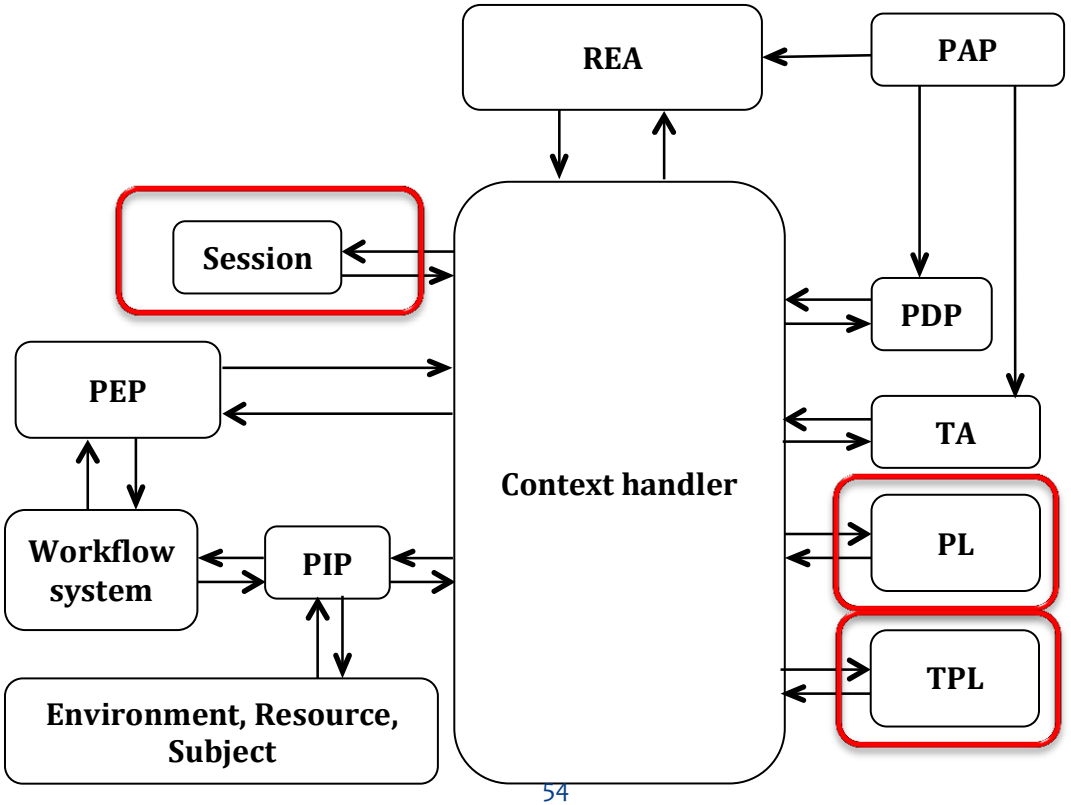
Framework:



BP-XACML

Policy Model

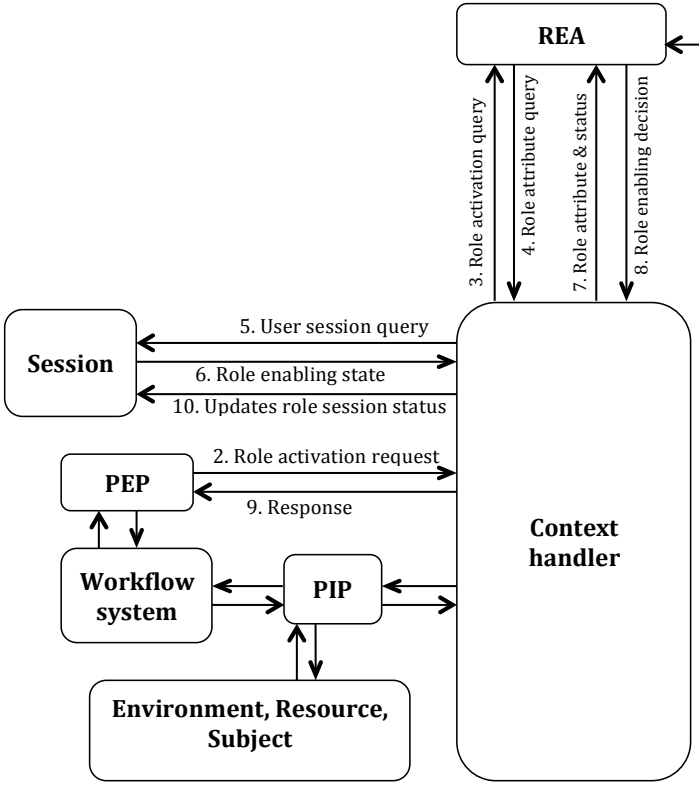
Authorities and Repositories:



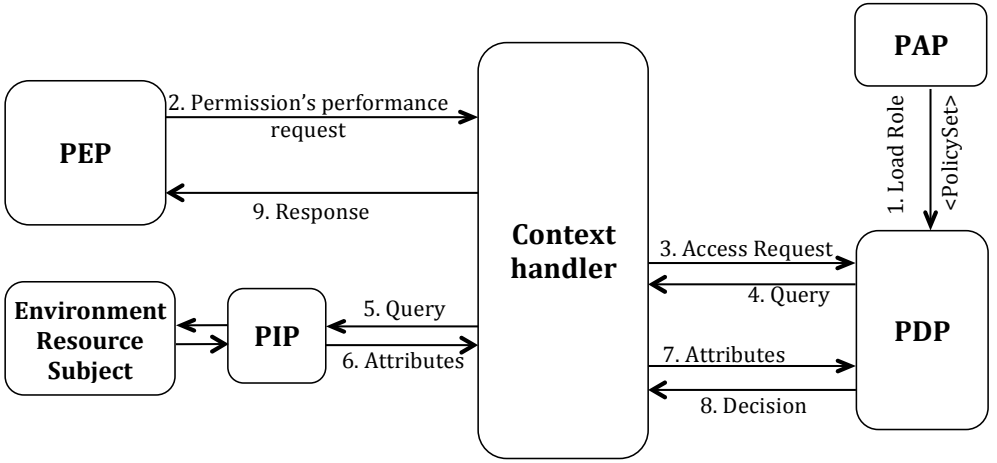
BP-XACML

Policy Model

Role activation Request:



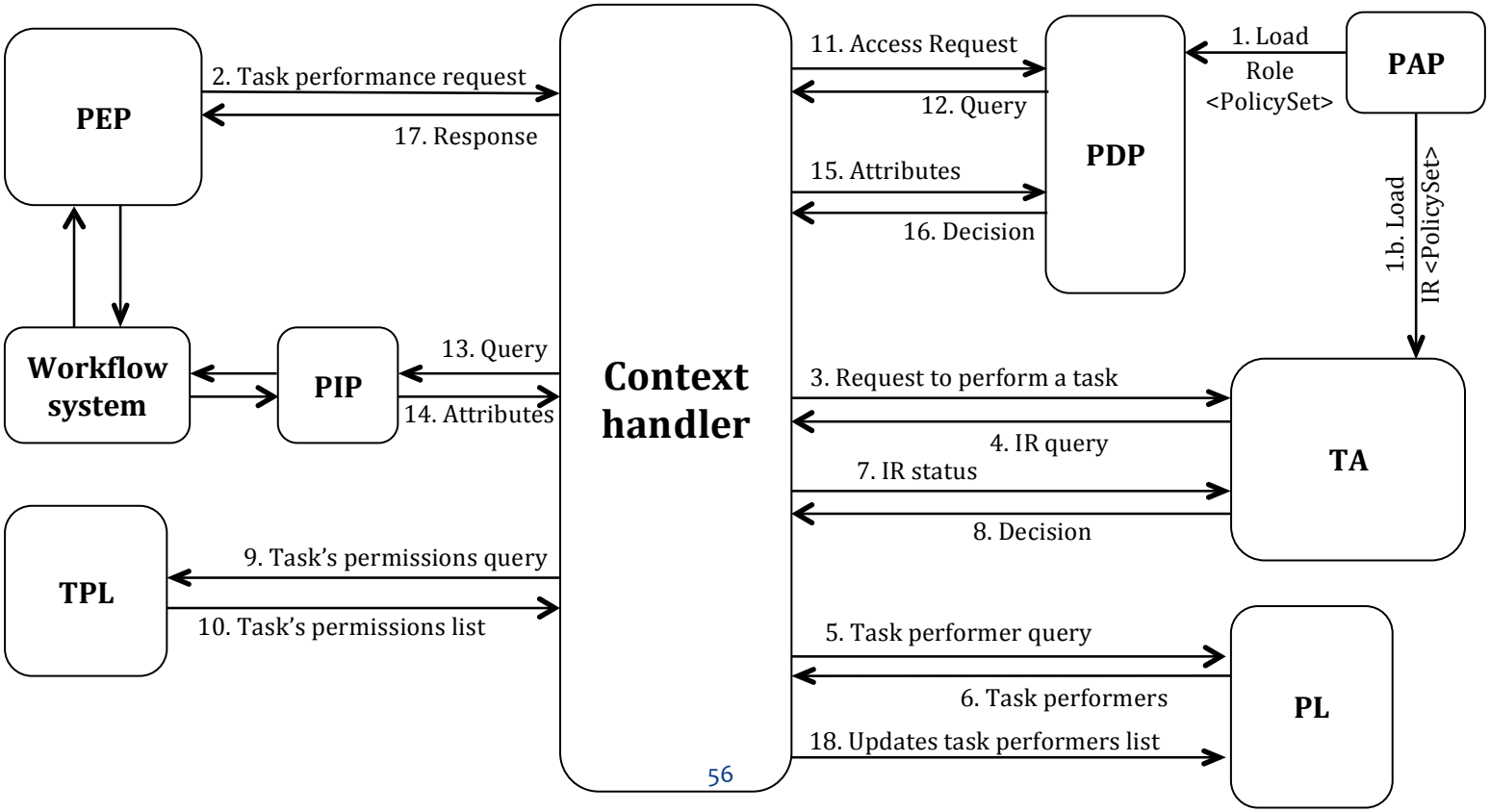
Standard RBAC Request:



BP-XACML

Policy Model

Task Performance Request:



BP-XACML

Policy Semantics

* Extended the RBAC-XACML semantics.
* Task and task instances are not supported in RBAC- XACML
* Tasks are expressed as an XACML Resource. Task instance are expressed as a resource attribute called instance.

Policy Semantics

Role level SoD:

* SoD is expressed as rules in the SoD <PolicySet>.
* The function 'Session' is a new function. It takes one argument of data-type "..#string", which is the user's ID, and returns a list of all roles currently activated for this user

Instance-level Restrictions:

* IR expressed as rules in the IR<PolicySet>.
* The function 'PL' is a new function that retrieves the performers list of a specific task for a specific instance. It takes two arguments of data-type "..#string", which are a task name and an instance number. It returns a list of all users who performed the task for this instance.

Discussion

* BP-XACML support the notions of task, task instance, instance-level restrictions, and role-level SoD.
* BP-XACML support sessions and the idea of having multiple roles active at the same time.
* BP-XACML can be used with workflow authorisation model.
* Because of backward compatibility, BP-XACML can be used with NIST-RBAC authorisation models.

Discussion

* Final Decision is based on the current organisation policies
* TA decision is not enough
* Task-Permissions List (TPL): a function that takes in a task ID and sends back a list of permissions associated with the task.

Agenda

* Introduction
  * Background
  * Area of research
  * Aims
  * Contributions
* Authorisation management for BPM
  * Example scenario
  * Characteristics analysis
  * Literature review
* BP-TRBAC
  * Concept
  * Formalisation
  * Example
  * Discussion
  * review
* SPCC
  * YAWL
  * SPCC
  * Implementation
  * Results
* BP-XACM
  * Policy structure
  * Policy Model
  * Policy semantics
  * Example
  * Discussion
* Conclusion
  * Contribution summary
  * Future direction

Conclusion

Summary

* BPM is growing in use
* Authorisation policies need to be enforced all the time
* Business process environments has special authorisation constraints
* Current authorisation models are not suitable for business process environments
* Current authorisation policy languages do not have the ability to represent business process authorisation constraints.
* There is a need for enforcing authorisation policies in business process environments
* BP-TRBAC is a model to enforce authorisation policies in mixed-environments, that include business process and non-business process systems.
* BP-XACML is a structured, machine-readable language to write the authorisation policies

Conclusion

Contributions

* A case scenario from a real life security-sensitive environment: After collecting data and interviewing stakeholders we were able to provide the scenario.
* The characteristics of an authorisation model for business process environments, and the characteristics that a business process authorisation language should satisfy.

Conclusion

Contributions

* Proposed BP-TRBAC, a unified organisation-wide business process authorisation model. BP-TRBAC is designed to support all the required characteristics.
* A use-case implementation is provided. The use-case is intended to check design time assignments. The implementation showed that SPCC was able to communicate with YAWL.

Conclusion

Contributions

* Proposed BP-XACML an authorisation policy language for business processes. It can also support standard RBAC policies. BP-XACML is designed to support all the required characteristics. The policy language is generic.
* A policy model for BP-XACML is provided. It showed how using this language a system can handle and evaluate authorisation requests. The policy model is in compliance with the authorisation model BP-TRBAC.

Future research directions

* Full BP-TRBAC implementation in an operational environment.
  * Testing non-functional capabilities.
* Implementing SPCC using BP-XACML as the policy language.
* Implement the BP-XACML policy model

Acknowledgment

* I would like to acknowledge the contribution made by many stakeholders involved in the Airports of the Future project (www.airportsofthefuture. qut.edu.au). Part of this research was undertaken as a part of the Airport of the Future project (LP0990135), which is funded by the Australian Re- search Council Linkage Project scheme.
* Special thanks to Dr. Farzad Salim for his help, supervision, and professional support throughout my studies.
* I would like to thank Dr. Michael Adams, Nishchal Kush, and Nicholas Rodofile for their help with the SPCC implementation.

..:: Thank you ::..