THE UNIVERSITY OF BRITISH COLUMBIA

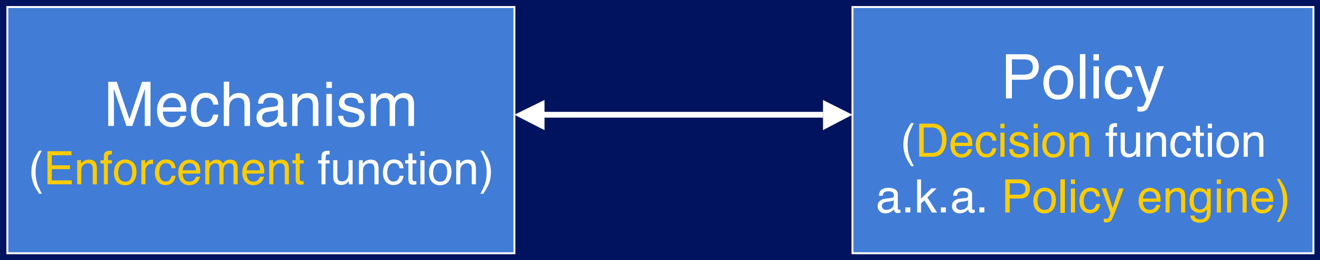
On the Benefits of Decomposing Policy Engines into Components

Konstantin Beznosov
Electrical and Computer Engineering
University of British Columbia

Outline

- Problem motivation
- Proposed solution
- Feasibility demonstration
  - policy engine architecture
  - examples
- Summary

Problem Motivation



Mechanism
(Enforcement function)

Policy
(Decision function
a.k.a. Policy engine)

Distributed app. developers/admins have limited choices:

1. Pre-built policy engines with limited capabilities
   - e.g., JAAS default policy file, COM+, EJB authorization
   - Limited support for non-trivial or application-specific policies
2. Pre-built policy engines “one size fits all” generic
   - e.g., CORBA
   - Unnecessary complex and expensive to use
3. “plug-in” APIs for creating custom “do-it-yourself” engines
   - e.g., CORBA Sec. Replaceable, JSR 115, SiteMinder and alike
   - Hard to do it right

Premise

- common policy elements
  - e.g., authorizations based on roles, groups, location
- differences in
  - the weight and composition
    - e.g., location || (role && group) vs.
      role || (location && group)
  - application-specific factors
    - e.g., relations, certification, license

What Could Be Done About It?

Assemble policy engines out of pre-built and custom components, i.e.,

Policy engines as Component Frameworks

Expected Benefits

- wide range of supported policies
- “pay as you go” cost of supporting a policy
  - determined by required policy
    - not by policy engine complexity
  - incremental changes proportional to policy Δ-s
    - addition/removal/re-composition of policy components
    - re-use of existing policy logic by developers/administrators

THE UNIVERSITY OF BRITISH COLUMBIA

Demonstrating Feasibility

Architecture Used for Demonstration

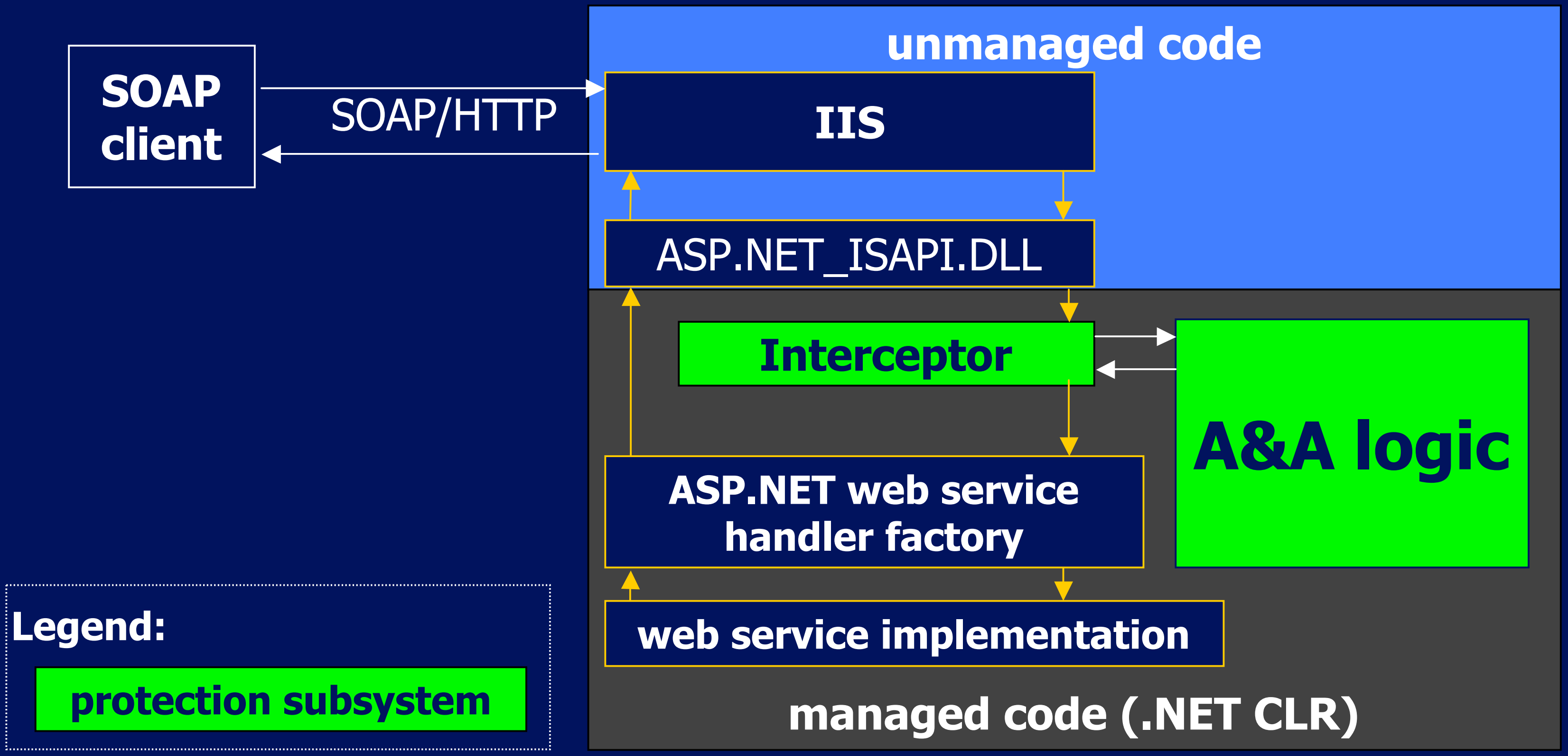
What is it?

Authentication and Authorization (A&A) architecture for ASP.NET Web services

Key features

1. Simplifies creation of custom authorization logic, and avoids generic authorization engine
2. Enables incremental modifications to the policy engine
3. Enables fine-grained replaceable authorization modules

Separation of Enforcements & Decisions



SOAP client

SOAP/HTTP

unmanaged code

IIS

ASP.NET_ISAPI.DLL

Interceptor

A&A logic

ASP.NET web service handler factory

web service implementation

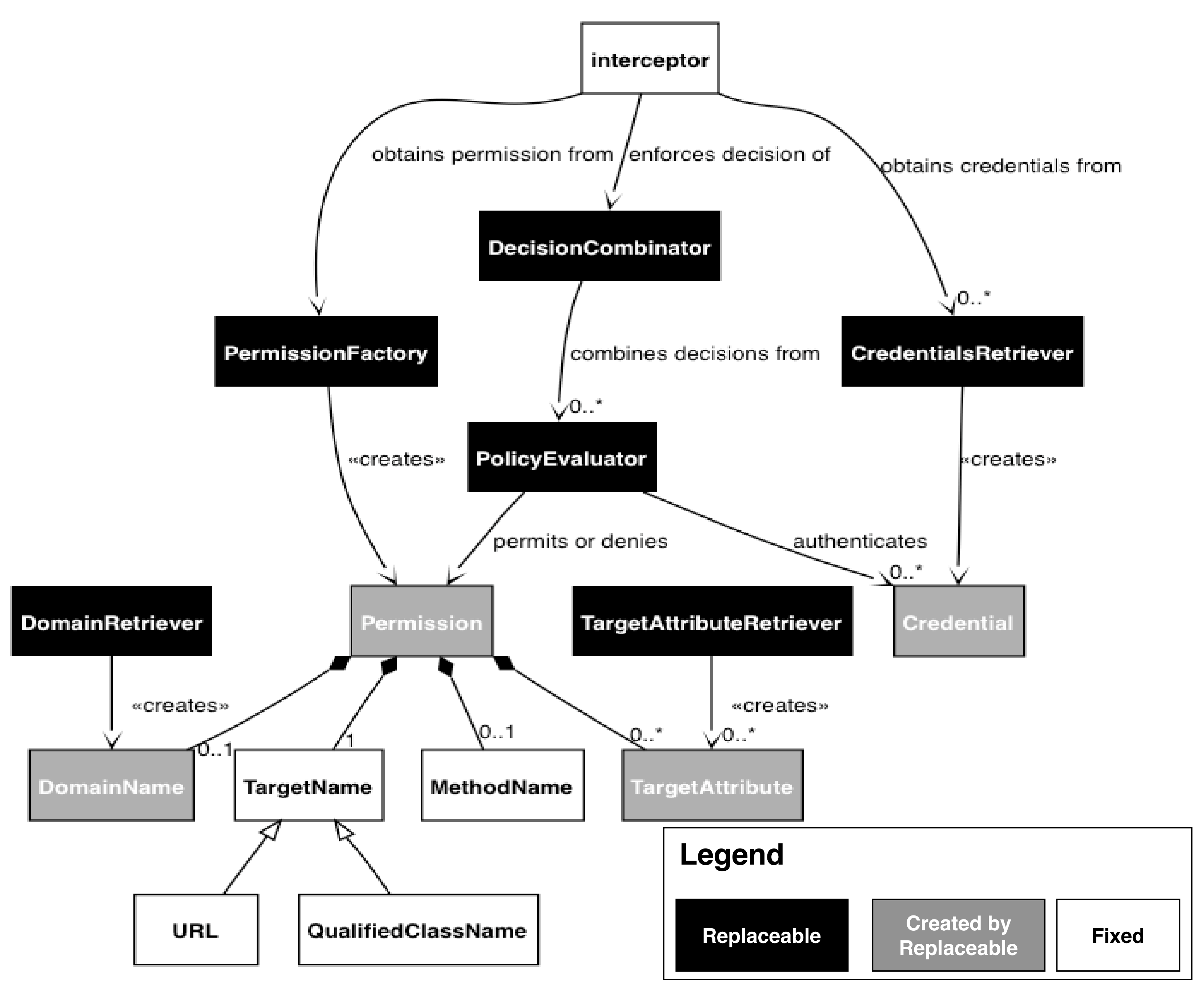
managed code (.NET CLR)

Legend:

🟩 protection subsystem

- Interceptor enforces
- Decisions made in “A&A logic”

Component Framework for A&A Policy Engine



THE UNIVERSITY OF BRITISH COLUMBIA

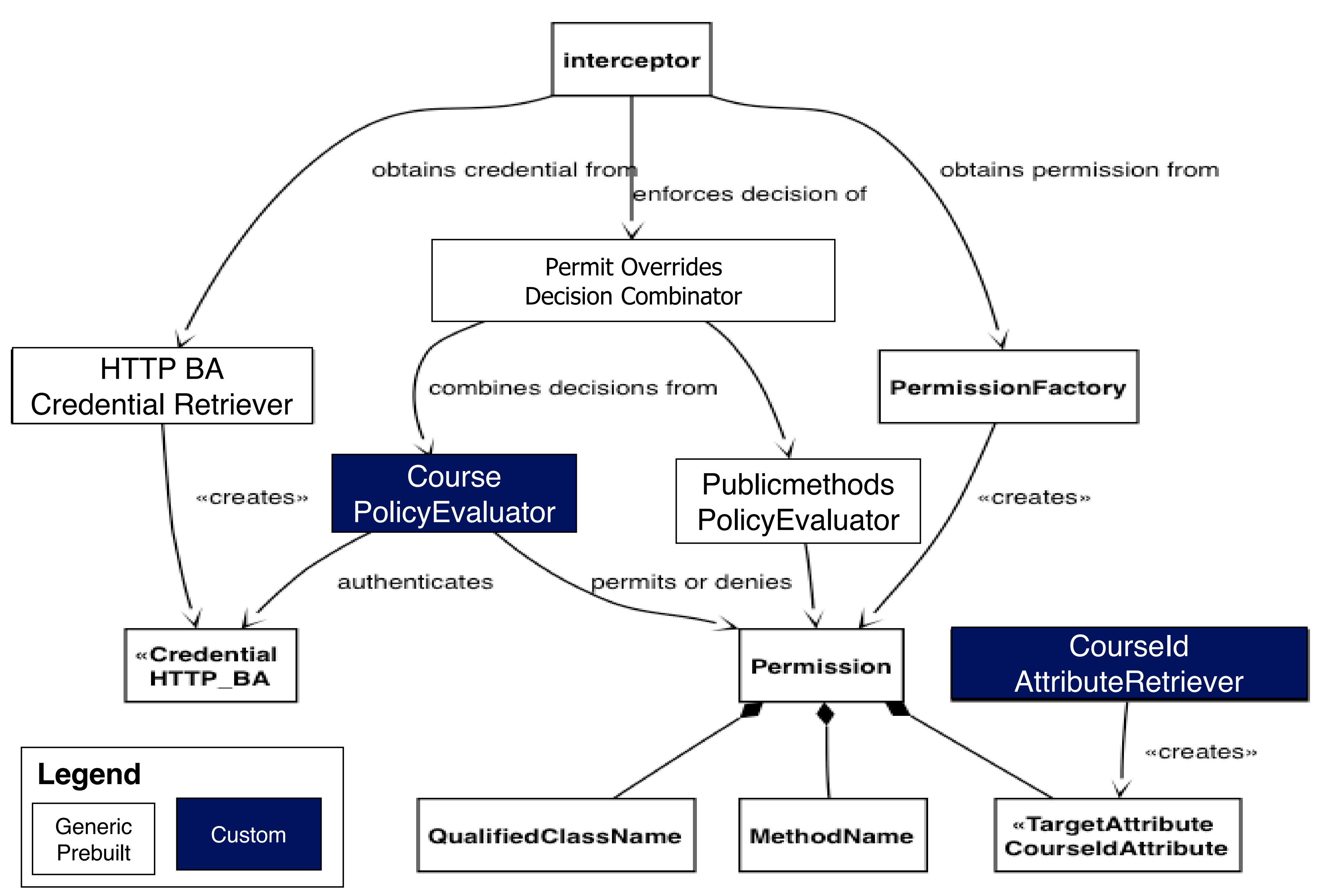
Example 1

University Course Web Service

University Course Web Service **Policy**

1. Anyone can lookup course descriptions.
2. All users should authenticate using HTTP-BA.
3. Registration clerks can list students registered for the course and (un)register students.
4. The course instructor can list registered students as well as manage course content.
5. Registered for the course students can download assignments and course material, as well as submit assignments.

Policy Engine Assembly for Example 1



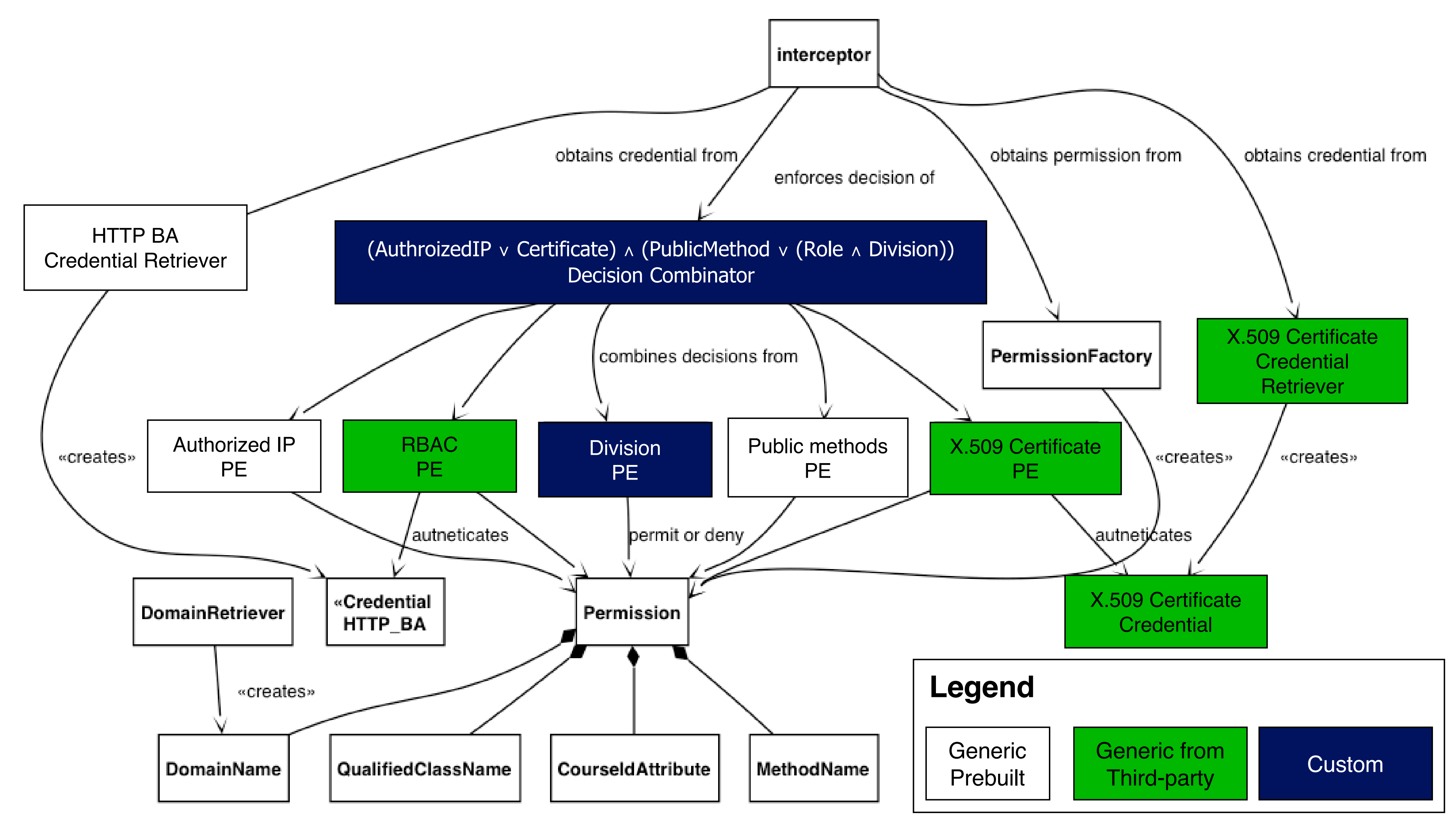
Example 2

Human Resources Web Service for an International Organization

HR Web Service Policy

1. Only users within the company's intranet or those who access the service over SSL and have valid X.509 certificates issued by the company should access.
2. Anybody in the company can look up any employee and get essential information about her/him.
3. HR employees can modify contact information and review salary information of any employee from the same division.
4. HR managers can modify any information about the employees of the same division.

Policy Engine Assembly for Example 2



Summary

Problem

Affordable support for diverse policies

Proposed solution

Policy engines as component frameworks

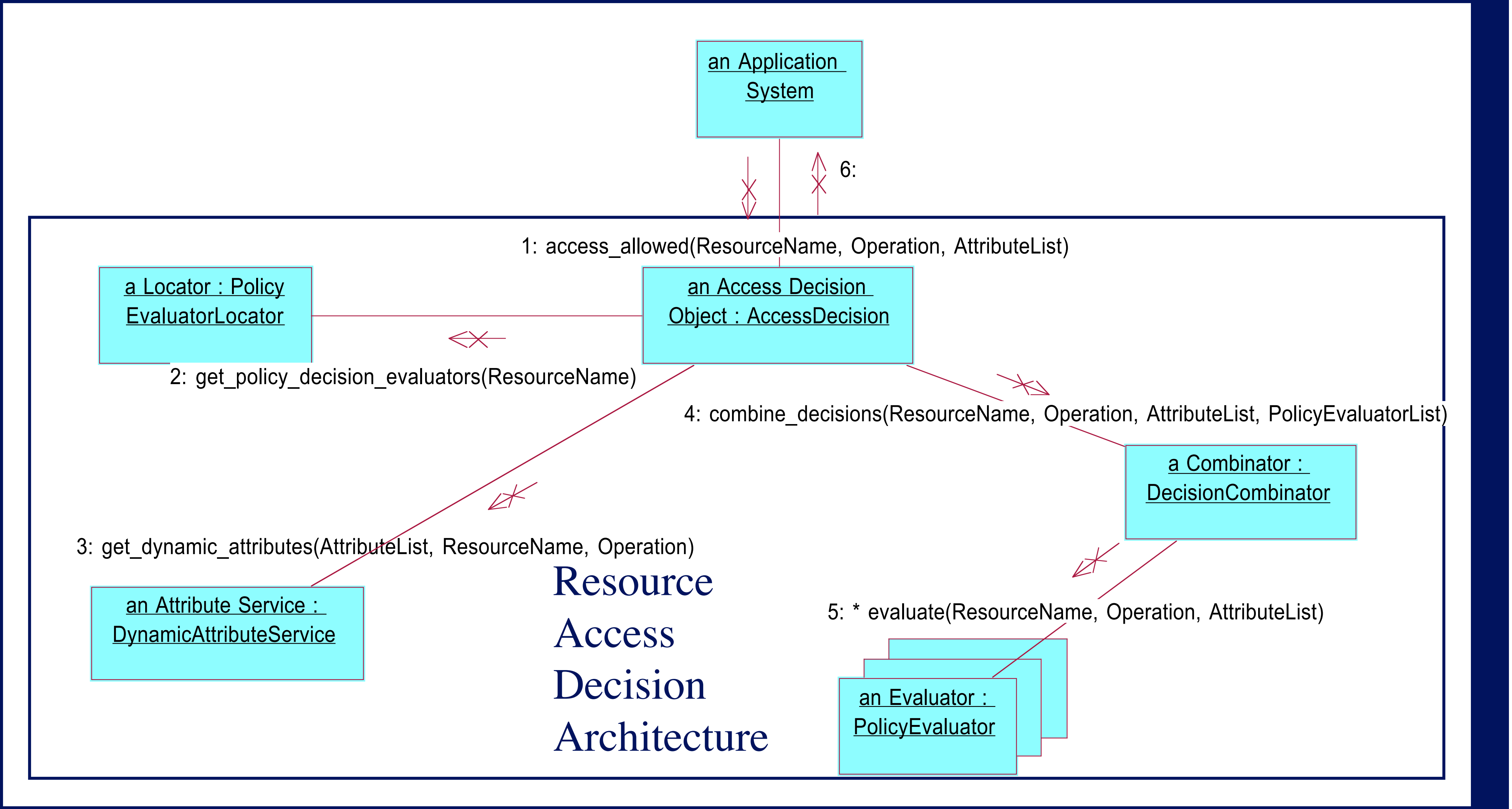
Contributions

1. Proposes CF-based to approach policy engine designs
2. Demonstrates the feasibility with a protection architecture for ASP.NET Web services

THE UNIVERSITY OF BRITISH COLUMBIA

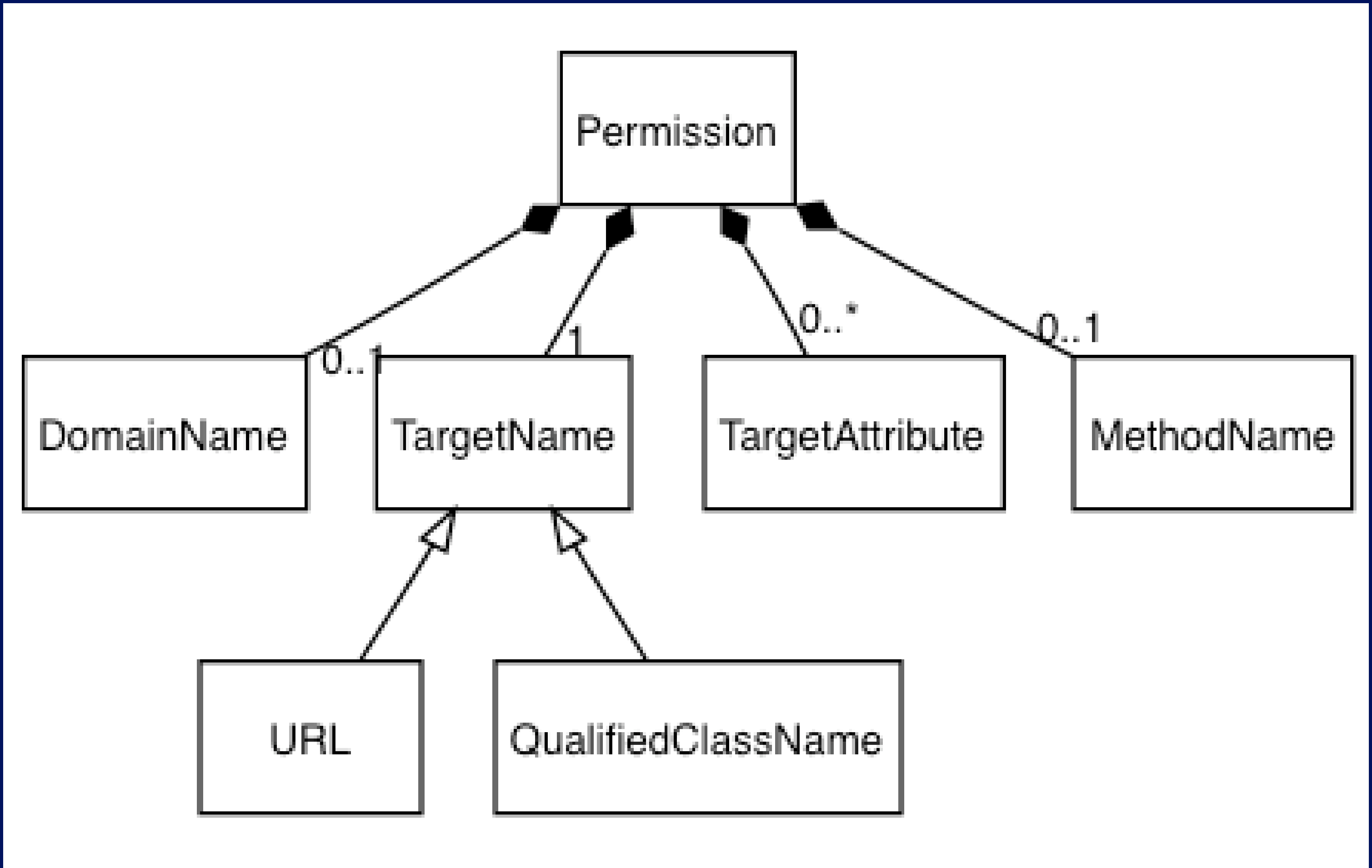
Additional Slides

Custom Composition of Authorization Logic


an Application System
6:
1: access_allowed(ResourceName, Operation, AttributeList)
a Locator : Policy EvaluatorLocator
an Access Decision Object : AccessDecision
2: get_policy_decision_evaluators(ResourceName)
4: combine_decisions(ResourceName, Operation, AttributeList, PolicyEvaluatorList)
a Combinator : DecisionCombinator
3: get_dynamic_attributes(AttributeList, ResourceName, Operation)
Resource
Access
Decision
Architecture
5: * evaluate(ResourceName, Operation, AttributeList)
an Attribute Service : DynamicAttributeService
an Evaluator : PolicyEvaluator

- RAD architectural style
- No monolithic general-purpose authorization engine

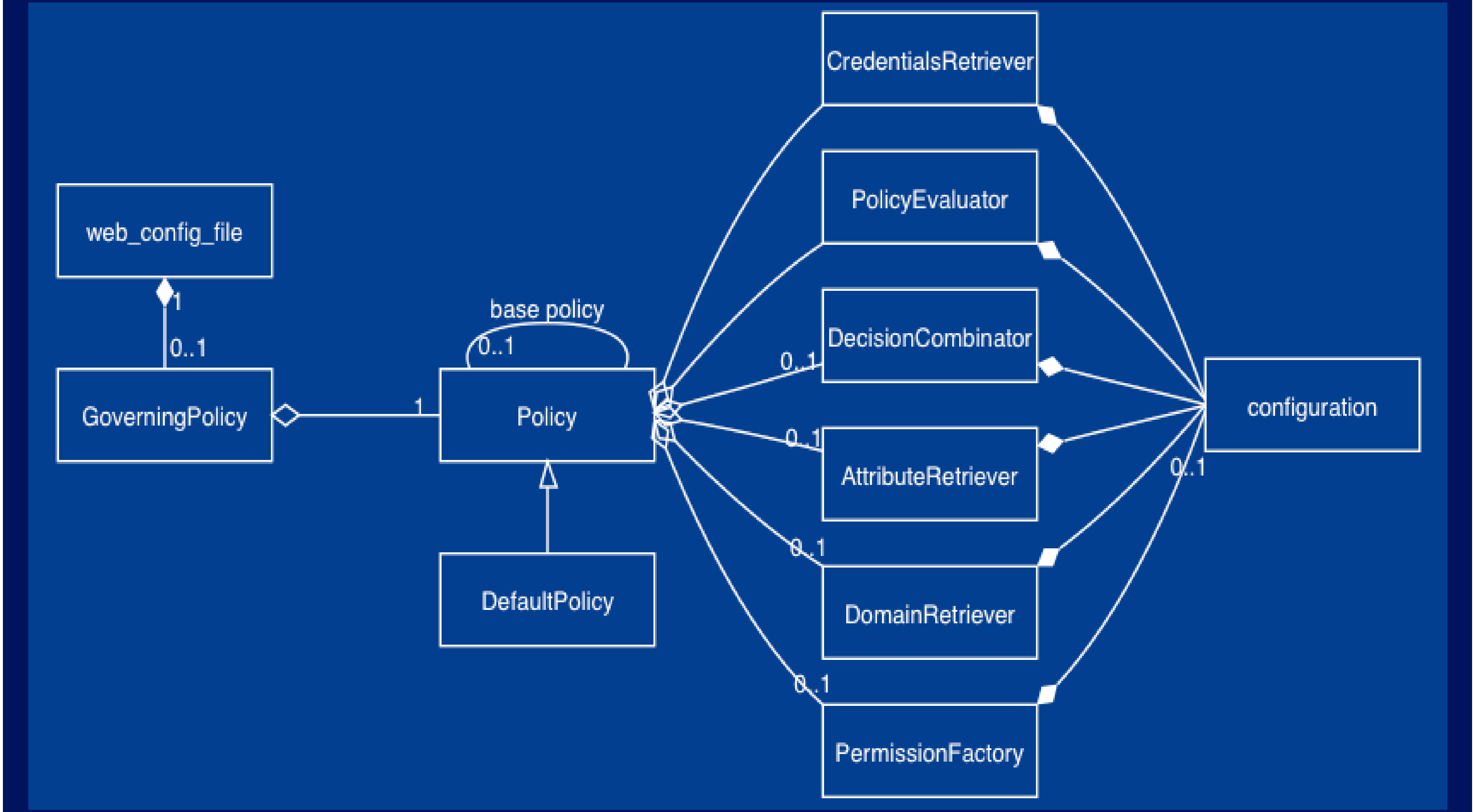
Adaptable Construction of Data Used for Authorizing Access



Permission Examples

Permission Example	Explanation
http://foobank.com/bar.asmx	Only the URL is used
com.foobank.ws.Sbar/m1	Class and method names
D1/com.foobank.ws.Sbar/m1	Same but in domain “D1”
com.foobank.ws.Sbar/owner=smith	Class name and attribute
D1/com.foobank.ws.Sbar/owner=smith /m1	Domain, class, attribute, method

Configuration Scalability, Extensibility, and Reuse


CredentialsRetriever
PolicyEvaluator
web_config_file
1
0..1
base policy
0..1
DecisionCombinator
0..1
GoverningPolicy
1
Policy
configuration
0..1
AttributeRetriever
0..1
0..1
DefaultPolicy
DomainRetriever
0..1
PermissionFactory