



# **Web Service Security**

Anja Fischer  
Technical University of Ilmenau, Germany  
Computer Science and Automation Faculty  
Distributed Systems and Operating Systems Chair

# Outline

- **Web Services and their Drawbacks**
- **Objectives**
- **Approach**
  - Security Requirements
  - Security Policy
  - Implementation
- **Evaluation**
  - Reference Monitor Principles
  - Trusted Computing Base
- **Conclusion**



■ **Web Services and their Drawbacks**

■ **Objectives**

■ **Approach**

- Security Requirements
- Security Policy
- Implementation

■ **Evaluation**

- Reference Monitor Principles
- Trusted Computing Base

■ **Conclusion**

# Web Services

- name is badly chosen
- independent, modular software components, which encapsulate well defined functions and provide standardized interfaces
- goals
  - interoperability
  - openness
  - easy usage
- protocol stack is based on XML and standardized
  - communication (SOAP)
  - interface description
    - Web Service Description Language (WSDL)
  - service directory
    - Universal Directory, Discovery and Integration (UDDI)

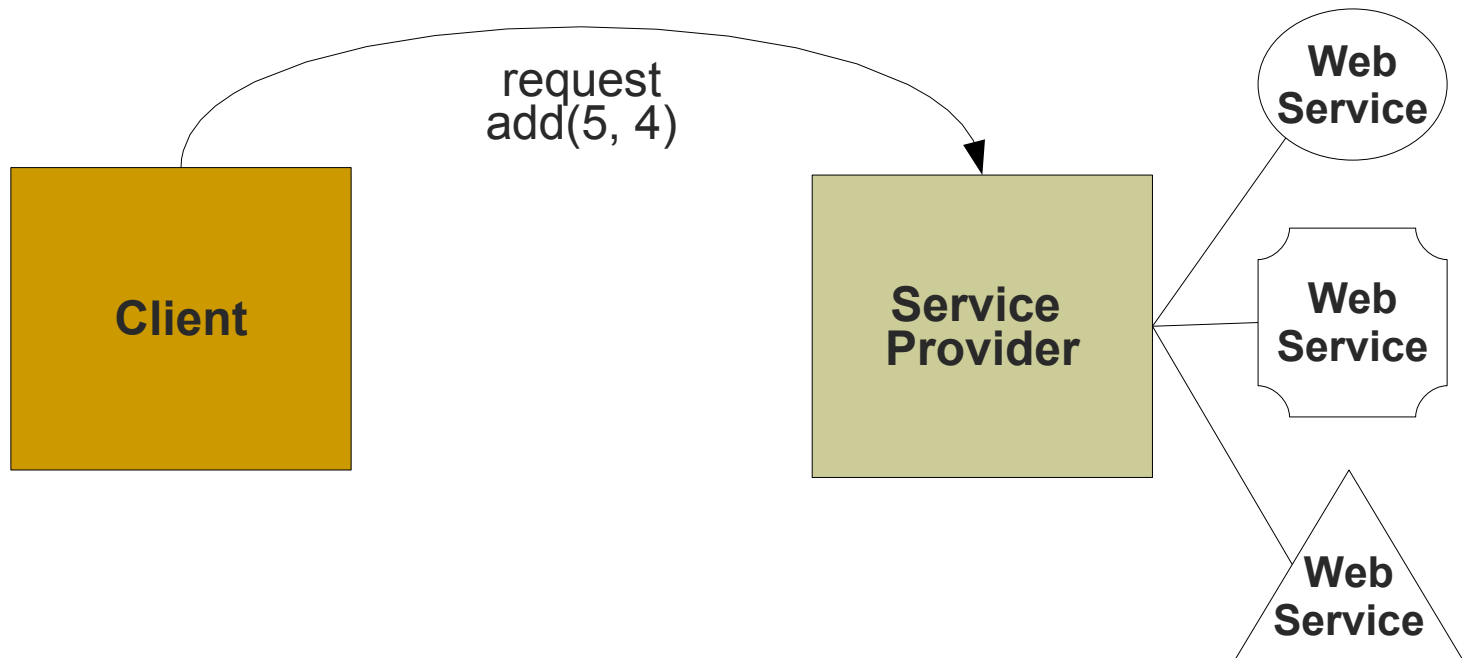
# Web Services

## Remote Procedure Call



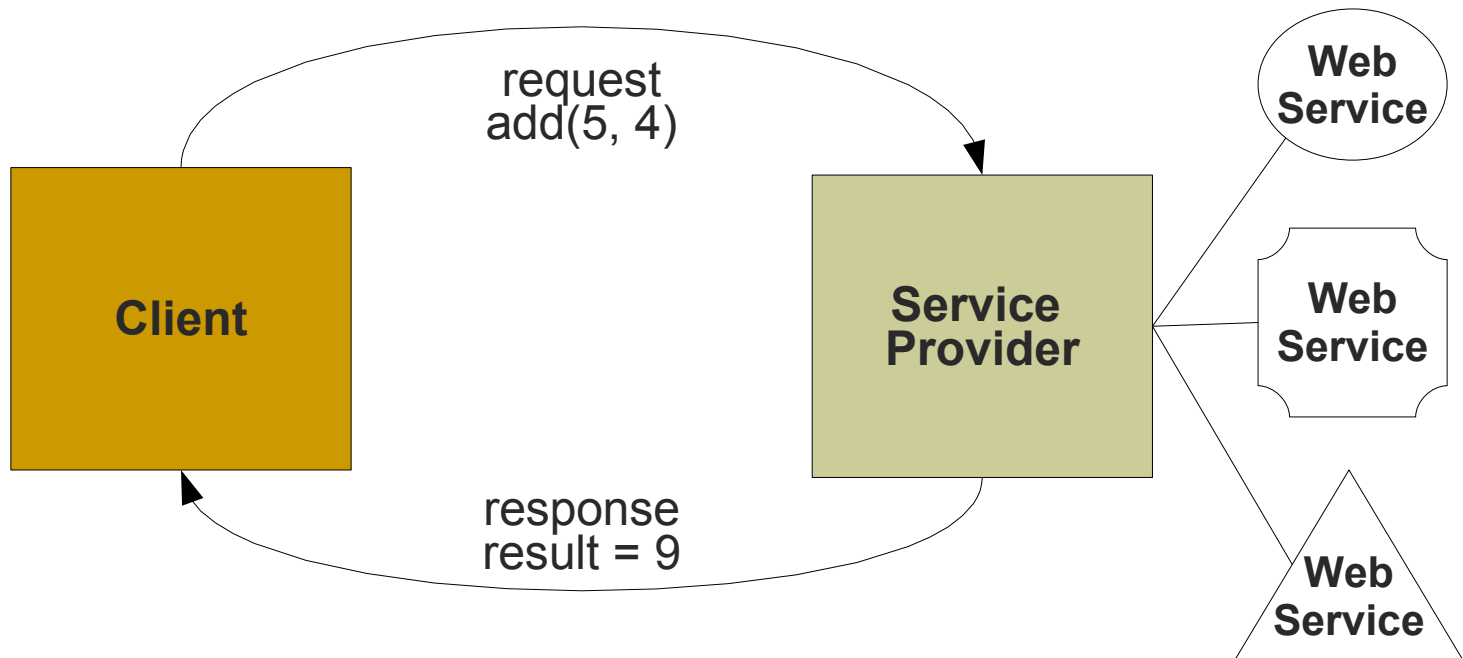
# Web Services

## Remote Procedure Call



# Web Services

## Remote Procedure Call



# Drawbacks

- web services are today's standard platform for heterogenous distributed systems
  - predominantly used for stateless, security neglecting scenarios
  - reasons: missing concepts and technologies
  - numerous security specifications driven by standard setting bodies (World Wide Web Consortium (W3C), Organization for the Advancement of Structured Information Standards (OASIS))
- unsatisfactory
- limited abilities in integrating security properties into today's web service platform
  - isolated solutions



- **Web Services and their Drawbacks**

- **Objectives**

- **Approach**

- Security Requirements
- Security Policy
- Security Model

- **Evaluation**

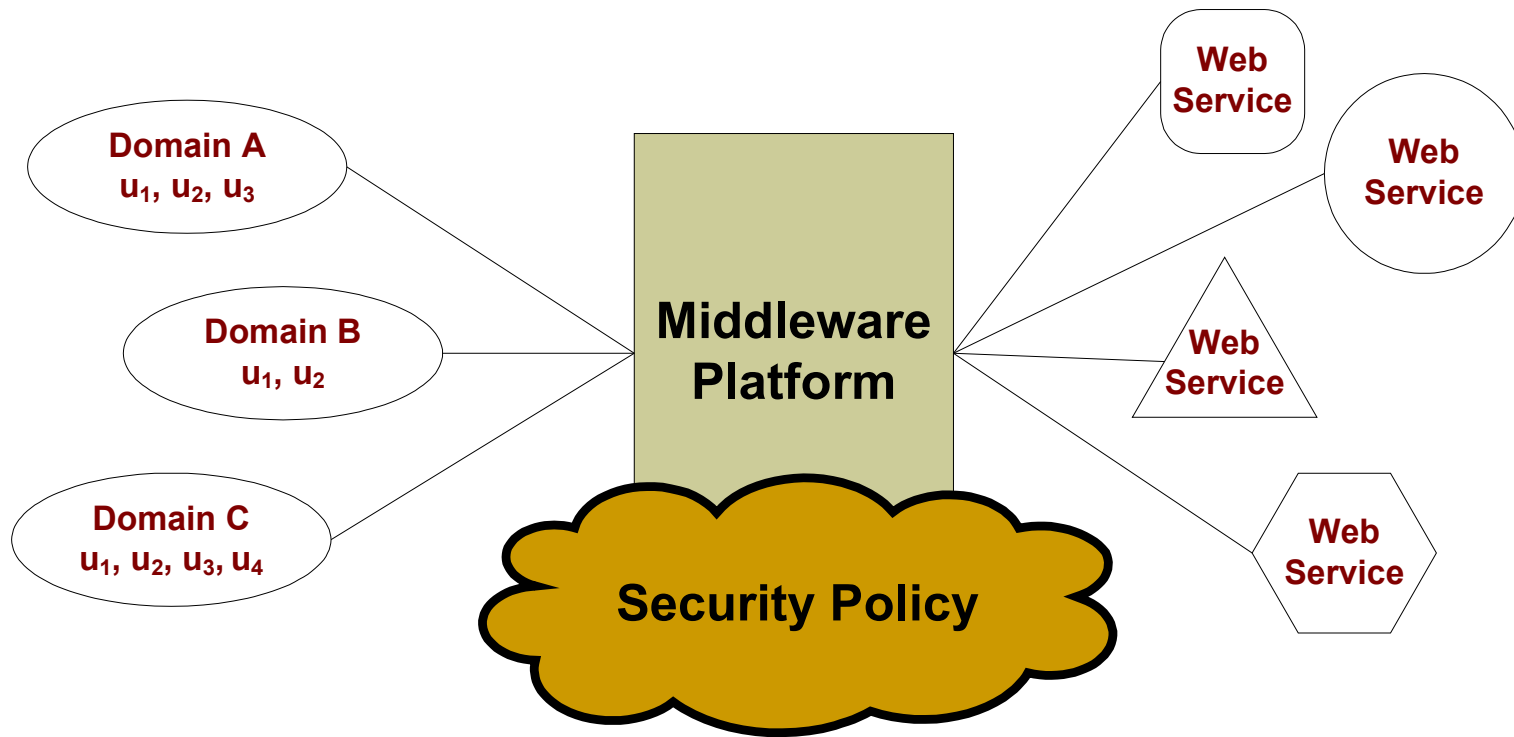
- Reference Monitor Principles
- Trusted Computing Base

- **Conclusion**

# Objectives

## Middleware Platform For Web Services

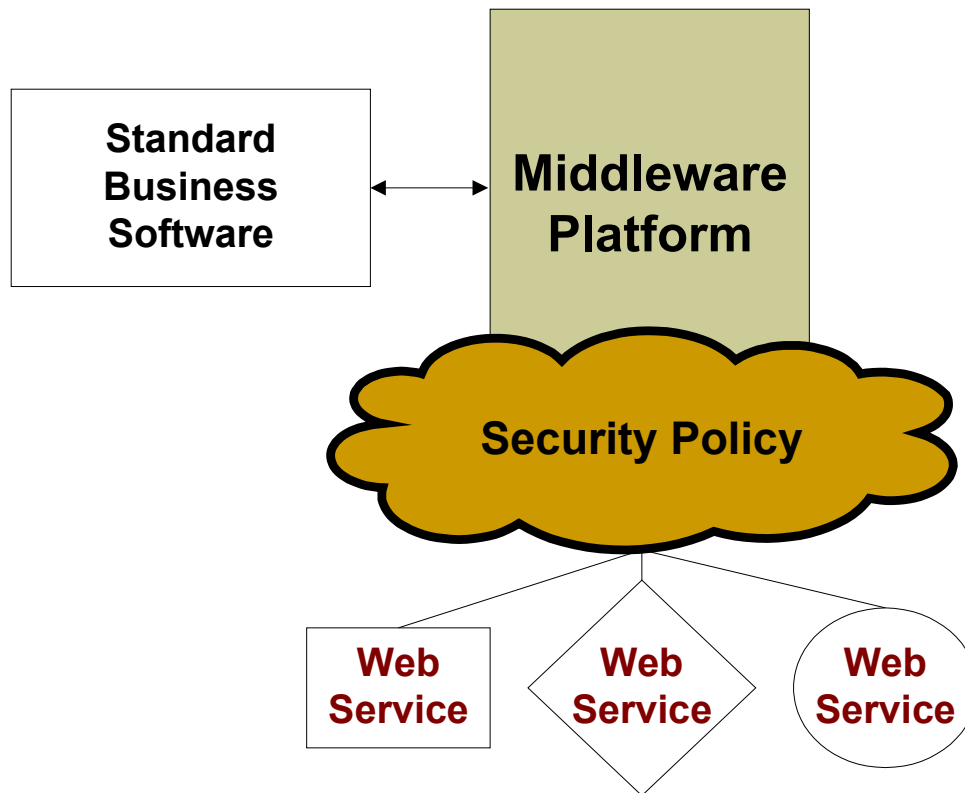
- building a security framework to establish a uniform, technological platform for system integration



# Objectives

## Example Of Use (1)

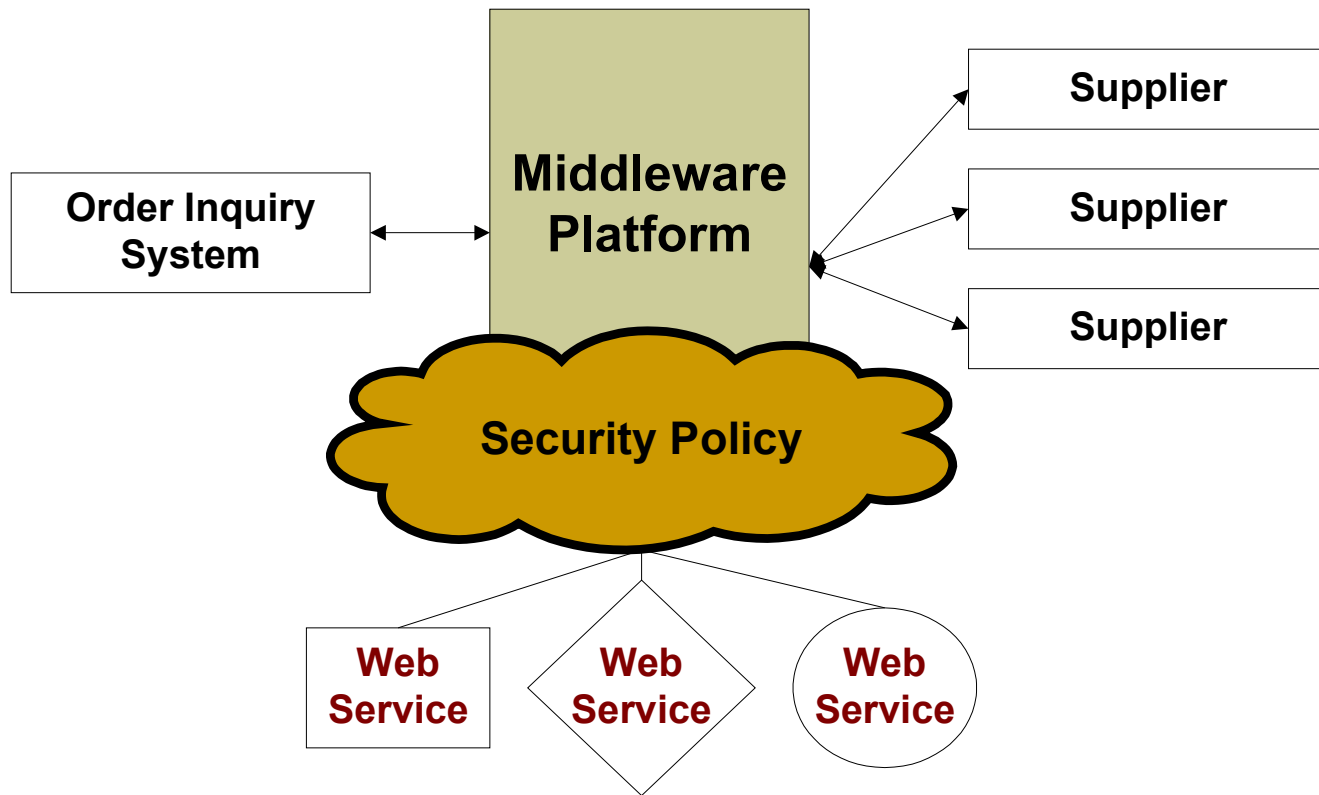
extending functions by using logistics services



# Objectives

## Example Of Use (2)

cooperative availability check (ATP – available to promise)



- 
- **Web Services and their Drawbacks**
  - **Objectives**
  - **Approach**
    - Security Requirements
    - Security Policy
    - Implementation
  - **Evaluation**
    - Reference Monitor Principles
    - Trusted Computing Base
  - **Conclusion**

# Approach

## Security Requirements

- general security properties (confidentiality, integrity, authenticity)
- concrete, application specific requirements
  - role based access control model (RBAC)
    - standard business software (e.g. ERP systems)
    - prospective users are already familiar
  - delegation of tasks and necessary permissions to fulfill tasks
  - non-repudiation
    - accounting presentational and legally binding business transactions are essential parts of business processes
  - availability
    - contract penalty

# Approach

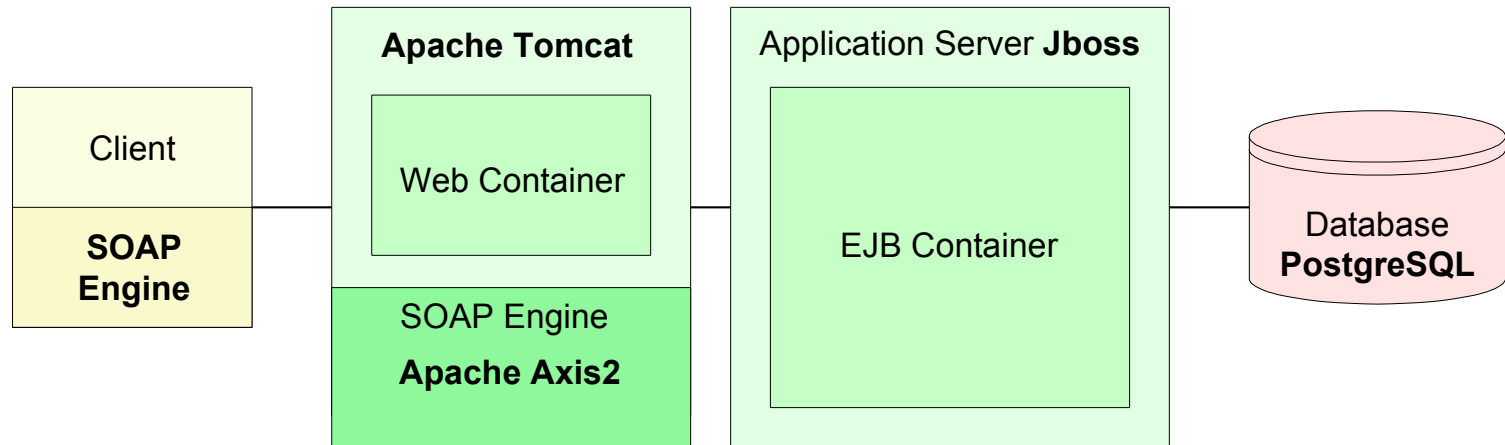
## Security Policy

- administrative role based access control (ARBAC) [Sandhu1996]
  - role hierarchies
  - constraints (separation of duty)
  - differentiating between administrative and user roles
- based on access control lists (ACL)
- fine-grained (methods)
- user premissions: see, execute, delegate, revoke
- development of a concept to realize reference monitor principles as close as possible

# Approach

## Implementation

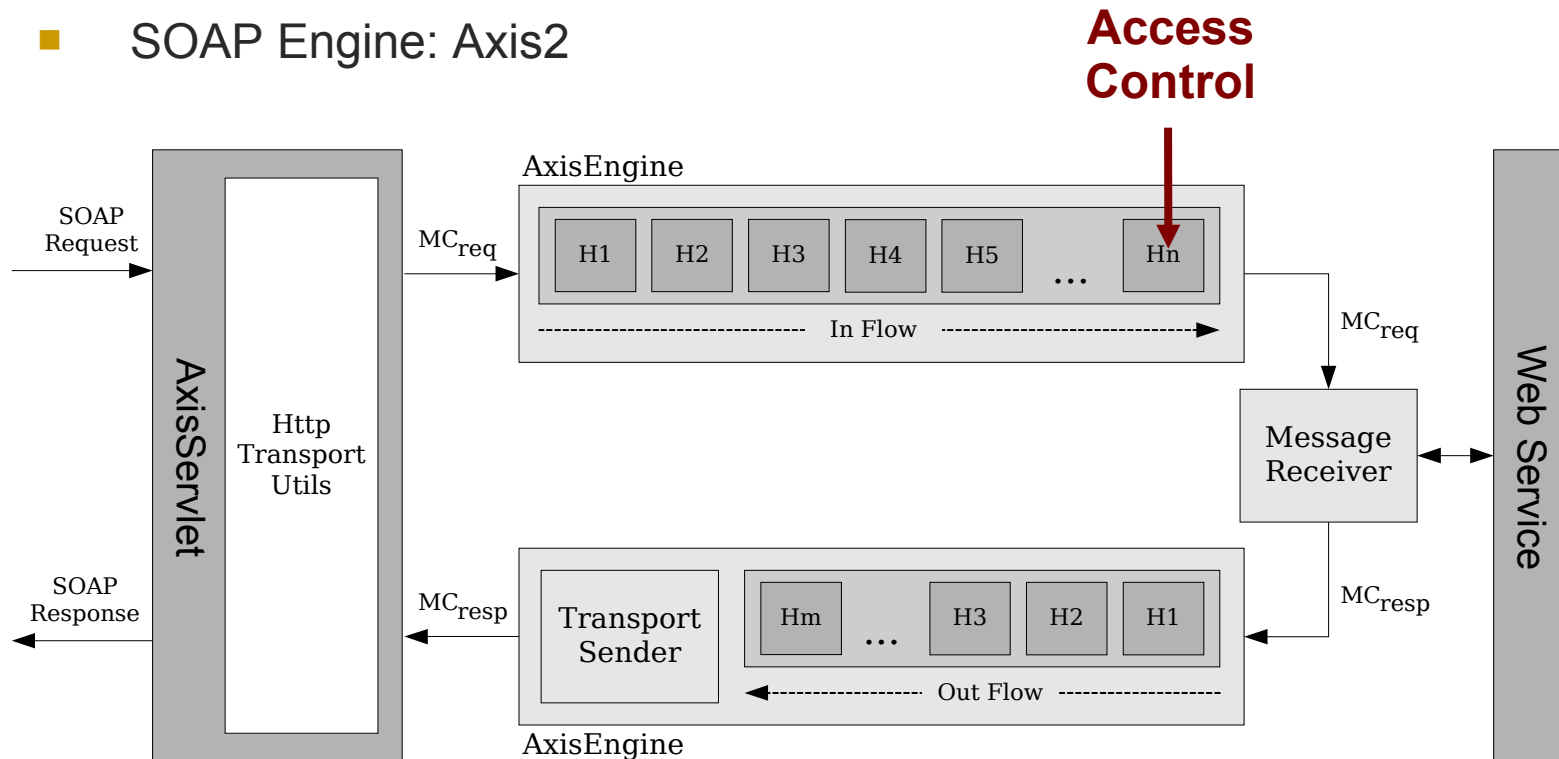
- proposed architecture



# Approach

## Implementation

- SOAP Engine: Axis2

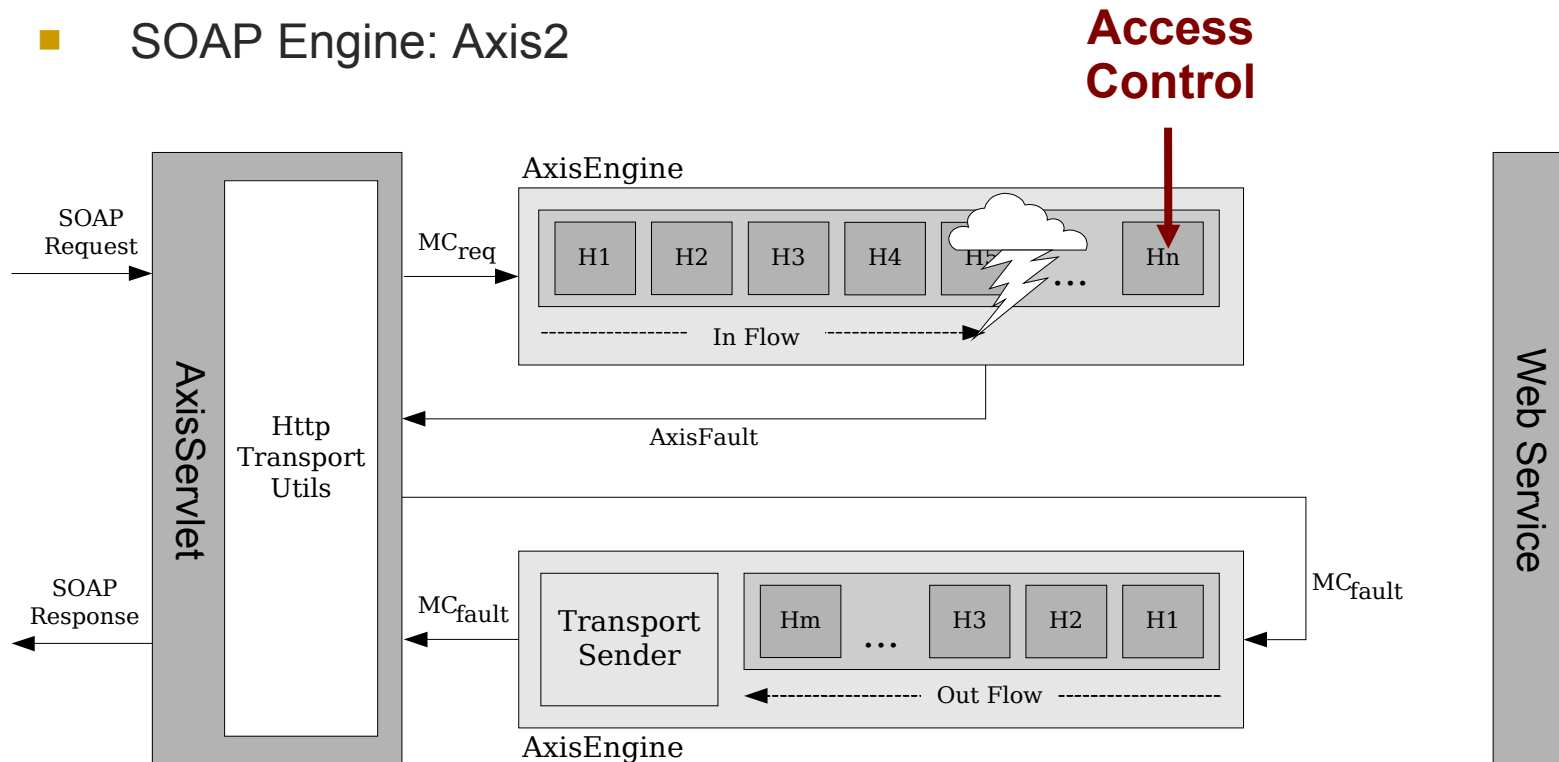


[Frotscher2007]

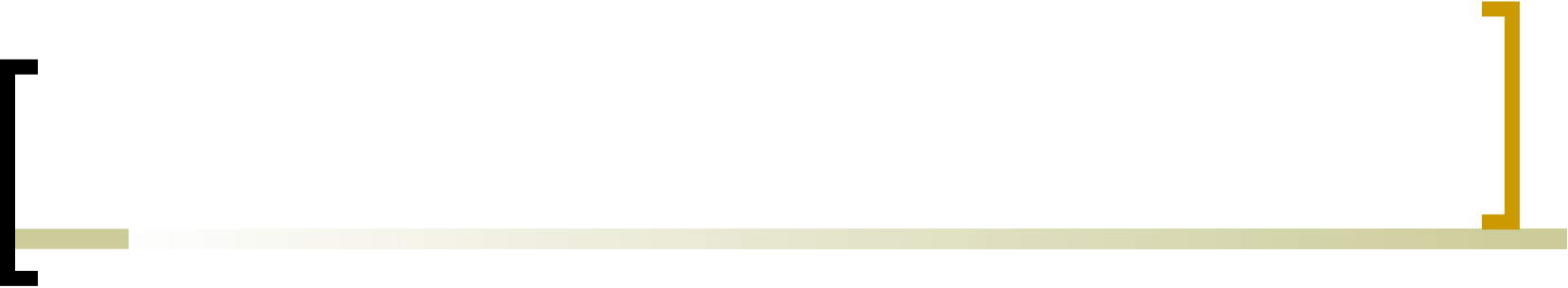
# Approach

## Implementation

- SOAP Engine: Axis2



[Frotscher2007]

- 
- **Web Services and their Drawbacks**
  - **Objectives**
  - **Approach**
    - Security Requirements
    - Security Policy
    - Implementation
  - **Evaluation**
    - Reference Monitor Principles
    - Trusted Computing Base
  - **Conclusion**

# Evaluation

## Reference Monitor Principles

### Isolation

- goal: „it must be tamperproof“ [GASSER1988]
- in microkernel based architectures isolation is based on:
  - tamperproofness, correctness, and completeness of memory management mechanisms [DoD1985]
  - using a type safe programming language [Aiken2006]

# Evaluation

## Reference Monitor Principles

### Isolation

- of the security policy implementing handler and Axis2 Engine is realized by using a type safe programming language
- due to performance all handlers of one Axis2 Engine are running in the same address space
  - no hardware supported isolation
  - whole Axis2 Engine is part of Trusted Computing Base
- conceptually possible to reduce size of Trusted Computing Base by running each handler as a separate process (address space)

# Evaluation

## Reference Monitor Principles

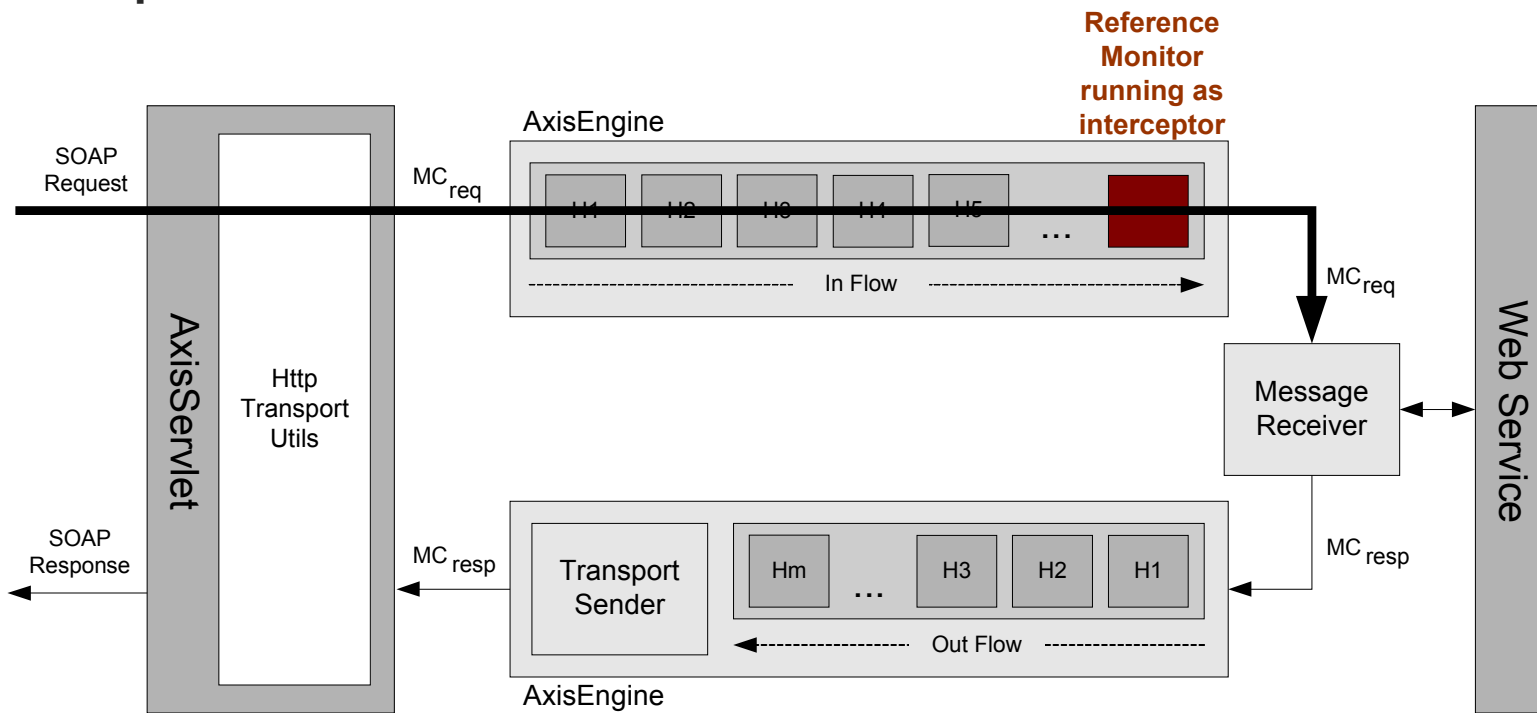
### Completeness

- goal: „it must be impossible to bypass“ [GASSER1988]
  - a subject may not reference an object without invoking the reference monitor
- is based on:
  - complete isolation of subjects and objects
  - complete interaction control between subjects and objects

# Evaluation

## Reference Monitor Principles

### Completeness



# Evaluation

## Reference Monitor Principles

### Completeness

- *precondition*: tamperproofness of all TCB components
- Axis2 framework realizes „total mediation“-property
  - realizes completeness principle
- handlers run as interceptors
  - algorithmic presentation of a security model
  - complete control of the information flow between subjects and objects
  - globally defined
- ignoring all communications attempts other than the communication paradigm of the middleware platform

# Evaluation

## Reference Monitor Principles

### Verifiability

- goal: „it must be shown to be properly implemented“ [GASSER1988]
- formal mathematical specification and verification
  - formal specification of the security policy: HRU model
  - analyzing the security model: HRU safety
- code inspection and thorough testing
  - state-of-the-art software engineering techniques
    - high-order language (Java 5)
    - object-oriented
    - structured design
    - modularity
    - information hiding
  - minimizing the size of the handler by excluding functions

# Evaluation

## Reference Monitor Principles

### HRU Model

- application oriented level of abstraction

$\delta(q, (\text{delegateSeePermission}(s_s.\text{domain}), (s_s, s_d, o)))) ::=$   
if  $\text{delegateSeePermission} \in M(s_s, o)$   
and  $\text{seeWSMethodPermission} \in M(s_s, o)$   
and  $s_s.\text{domain} \in M(s_d, \text{domain})$   
and  $\text{seeWSMethodPermission} \notin M(s_d, o)$  then  
enter  $(\text{seeWSMethodPermission}, ts_n)$  into  $M(s_d, o)$   
end if

- machine oriented level of abstraction  
→ advantages of HRU calculus are preserved

# Evaluation

## Reference Monitor Principles

### HRU Safety

- Given a protection system, is it possible that a specific subject can ever get a specific permission for a specific object?

in other words:

- Given a protection system, is it possible that a permission is entered at some place in the access matrix where it did not exist before?

# Evaluation

## Reference Monitor Principles

### analysis of HRU Safety – results

- model provides permissions and commands, which enable its usage in real computer systems

HOWEVER: this is critical for security properties of the model

- only by definition trustworthy instances (administrators) are allowed to own specific permissions
  - *createUserPermission, assignUserToRolePermission*

# Evaluation

## Reference Monitor Principles

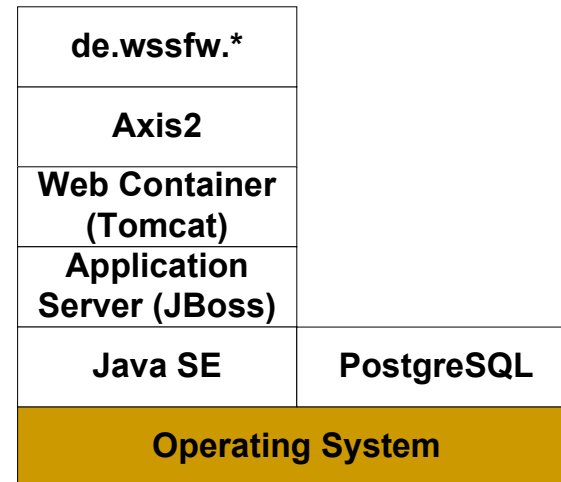
### Analysis of HRU Safety – results

- permissions to delegate permissions influence HRU Safety
  - HRU Safety of a configuration (Q) concerning one specific generic right depends on the domain of a subject
- leakage of permissions possible only within a single application domain
- not even administrators are able to get permission for another application domain

# Evaluation

## Trusted Computing Base

- very large
- numerous dependencies
- further components:
  - net infrastructure
    - filter
    - router
  - clients
    - operating system
    - browser
    - SOAP engine



- 
- **Web Services and their Drawbacks**
  - **Objectives**
  - **Approach**
    - Requirements engineering / risk analysis
    - Security policy
    - Implementation
  - **Evaluation**
    - Reference monitor principles
    - Transparency of the security policy
    - Trusted computing base
  - **Conclusion**

# Conclusion

- missing technologies and concepts for web services concerning security
- framework combines web services with state-of-the-art security mechanisms
- definition of application-oriented scenarios
  - cooperative order processing based on web services
- framework invisible for web services and regular users
- HRU model: efficient and effective instrument
- framework applied the reference monitor principles
- precisely defined trusted computing base

A horizontal line with a gradient from light green to white, spanning the width of the slide. A black left square bracket is positioned on the left side, and a gold right square bracket is on the right side.

**Thank you for your attention!**

# Literature

- Frotscher2007      Frotscher, Thilo; Teufel Marc; Wang, Dapeng; Java Web Services mit Apache Axis2; Entwickler.press 2007 ISBN 978-3-935042-81-9
- DoD1985            Department of Defense: Trusted Computer System Evaluation Criteria, Dezember 1985
- Aiken2006           Aiken, Mark; Fähndrich Manuel; Hawblitzel, Chris; Hunt, Galen; Larus, James: Deconstructing Process Isolation, MSPC 06: Proceedings of the 2006 workshop on Memory system performance and correctness, New York, USA, ISBN 3-411-01271-4

# Literature

- Gasser1988      Gasser, Morrie: Building a secure computer system, 1988, New York, Van Nostrand Reinhold, ISBN 0-442-23022-2
- HRU1976      Harrison, Michael; Ruzzo, Walter; Ullman Jeffrey; Protection in Operating Systems, Communications of ACM, August 1976, Volume 19, Number 8
- Sandhu1996      Sandhu, Ravi; Coyne, Edward; Feinstein, Hal; Youman, Charles; Role-Based Access Control Models, IEEE Computer, 1996, Volume 22, Number 2