Identifying Security Aspects in Early Development Stages

March 5th, 2008

*Takao OKUBO†‡ Hidehiko TANAKA‡
†Fujitsu Laboratories ltd.
‡Institute of Information Security
Japan

Table of Contents

FUJITSU

1.Motivation

Backgrounds

FUJITSU

Problems with software security

- Insufficient security expertise
  - The root of all evil
- Low Security coverage
  - ⇒ Vulnerability
- Low maintainability/reusability
  - ⇒ Development cost escalation

–AOSD- A Possible Solution

FUJITSU

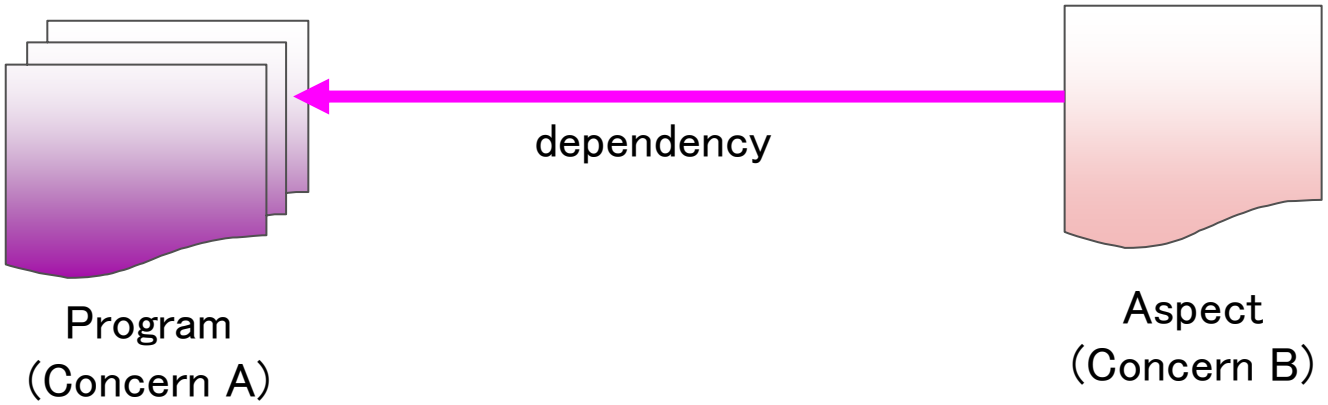
AOSD (Aspect-Oriented Software Development)

－Suitable for Non-functional Requirements (NFR)

⇒Silver bullet?

Our research: Attempts to verify the assumption

Aspect-Oriented Programming (AOP)

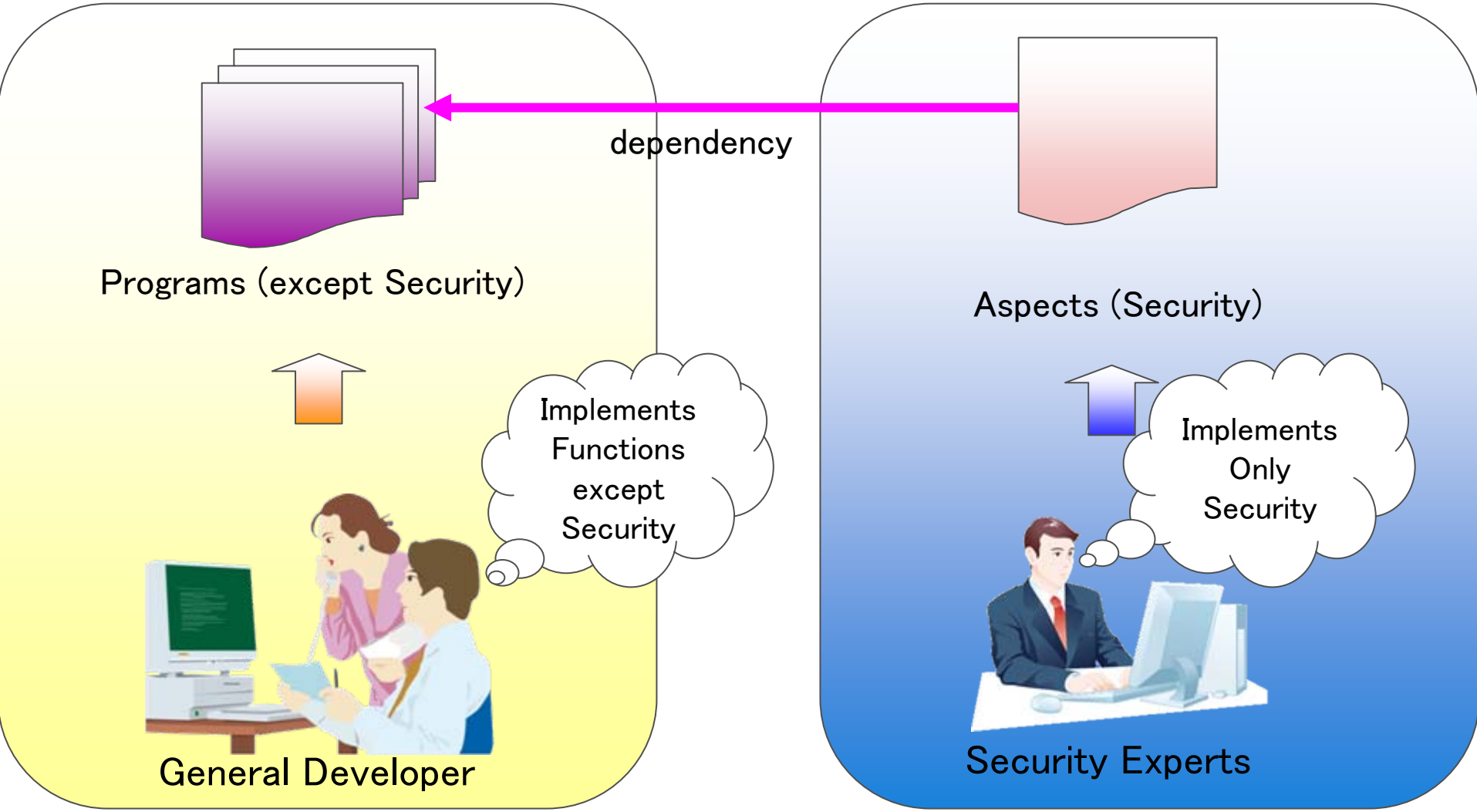


- Crosscutting concern/Dependency Injection (DI)
  A concern (aspect) is injected
  into other concerns (programs)
- Modularity, Coverage, Reusability

Ideal World with Aspects(1)

FUJITSU

Solution for Security Expertise Issues



Issues with AOP

FUJITSU

Add Security Aspects for Completed Web Server Programs

- Forceful Browsing
- Session Attacks

Issues with Coverage and Reusability

- Low Detection Accuracy
  - ←ad-hoc, implementation
- Low Reusability
  - ←dependent on specific codes

Our Goal

FUJITSU

Aspects should be analyzed, and designed in earlier development stages

Today's presentation

- Analysis methods
  - How to identify security requirements
  - How to achieve sufficient security coverage
    - For finding pointcut-candidate

2.Related Works

Eliciting Security Requirements

FUJITSU

Threat modeling (Microsoft)

- A famous threat analysis method
- Precise analysis with DFD
  - Much cost needed
  - Architecture must be detailed
- Unexpressed attackers
  - Difficult to identify threats caused by various types of attackers

Analyzing Aspects

FUJITSU

- I.Jacobson and P.Ng, ”Aspect-oriented software development with UML”, Addison-Wesley, 2004.
- S.Clarke and E.Baniassad, ”Aspect-oriented analysis and design”, Addison-Wesley, 2005.

- Methods for identifying aspects
- Insufficient reference to:
  - How to identify enough security concerns
  - Security coverage of aspects (All the threats must be covered)

Misuse Case ([Sindre00])

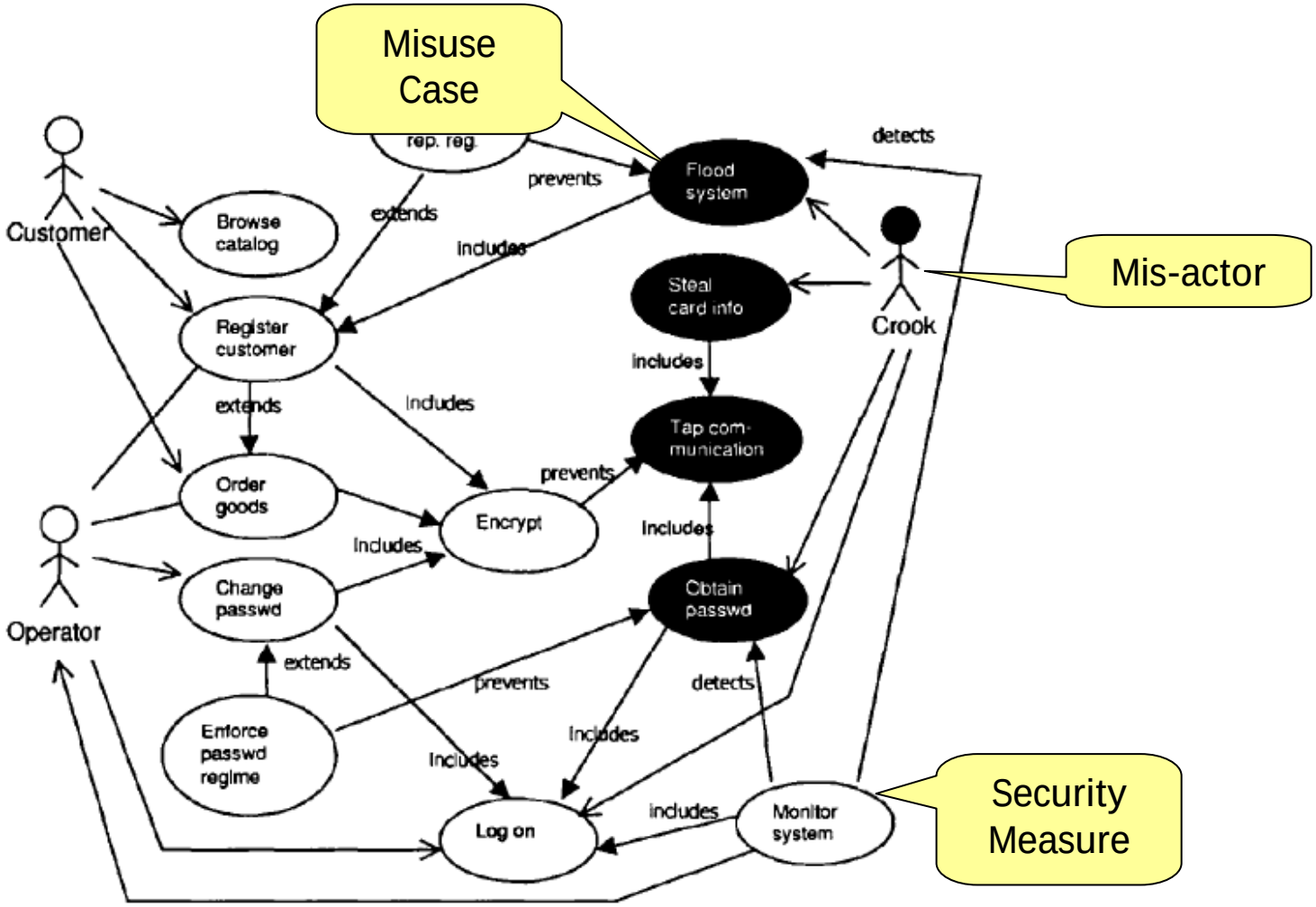


Figure from [Sindre00]

Advantages of Misuse Cases

FUJITSU

- Visualized analysis
  - UML-style diagram
  - Easy to understand
- Correspondence between threats and measures
- Security measures for aspect-candidates

Issues of Misuse Cases

FUJITSU

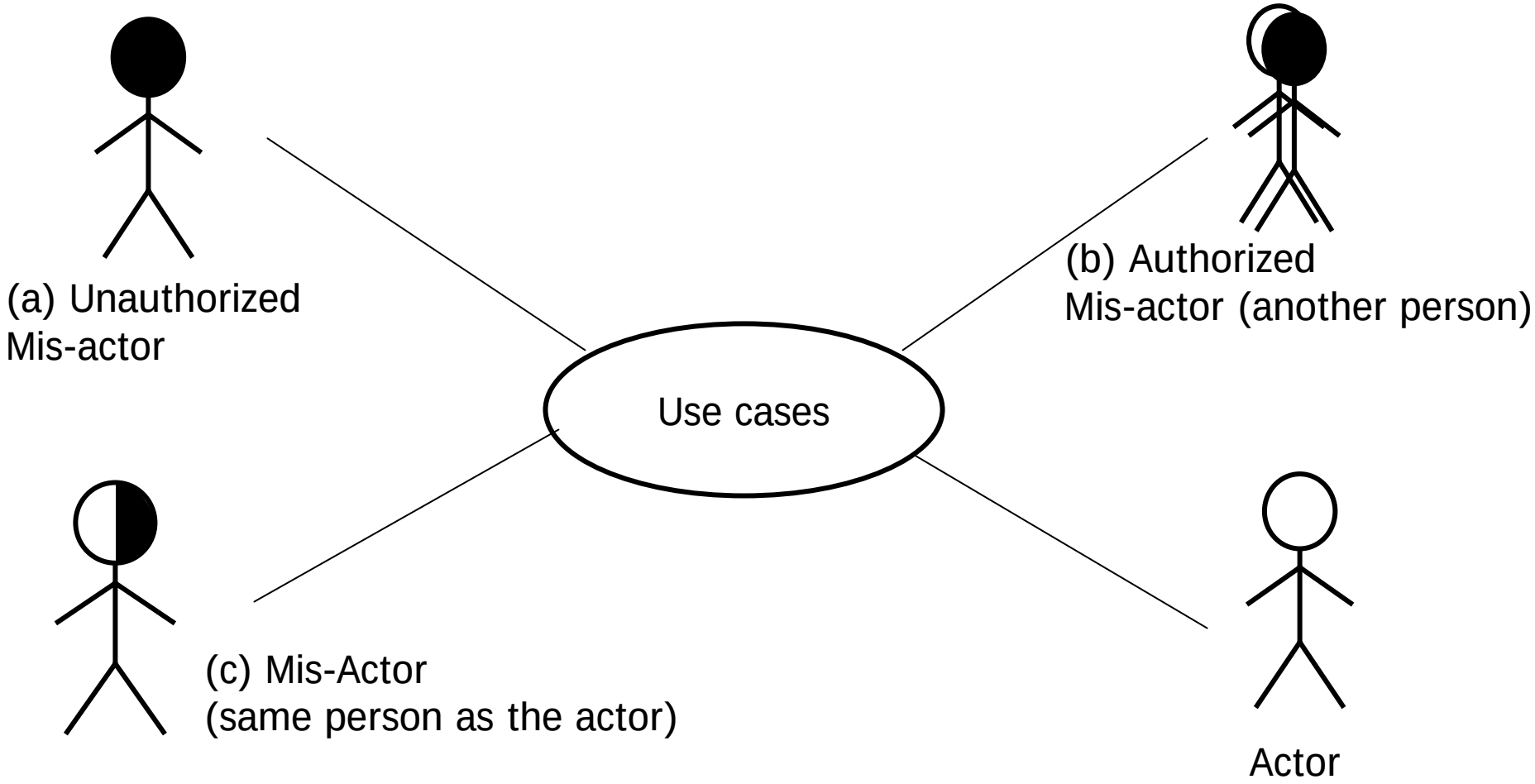
- Security expertise required for eliciting mis-actors & misuse cases
  - Data assets unexpressed
  - Different types of mis-actor unexpressed
- Difficulty for designing aspects
  - Specifying crosscutting points (Pointcuts)
  - Coverage

3.Proposed Approach

Proposed Approach

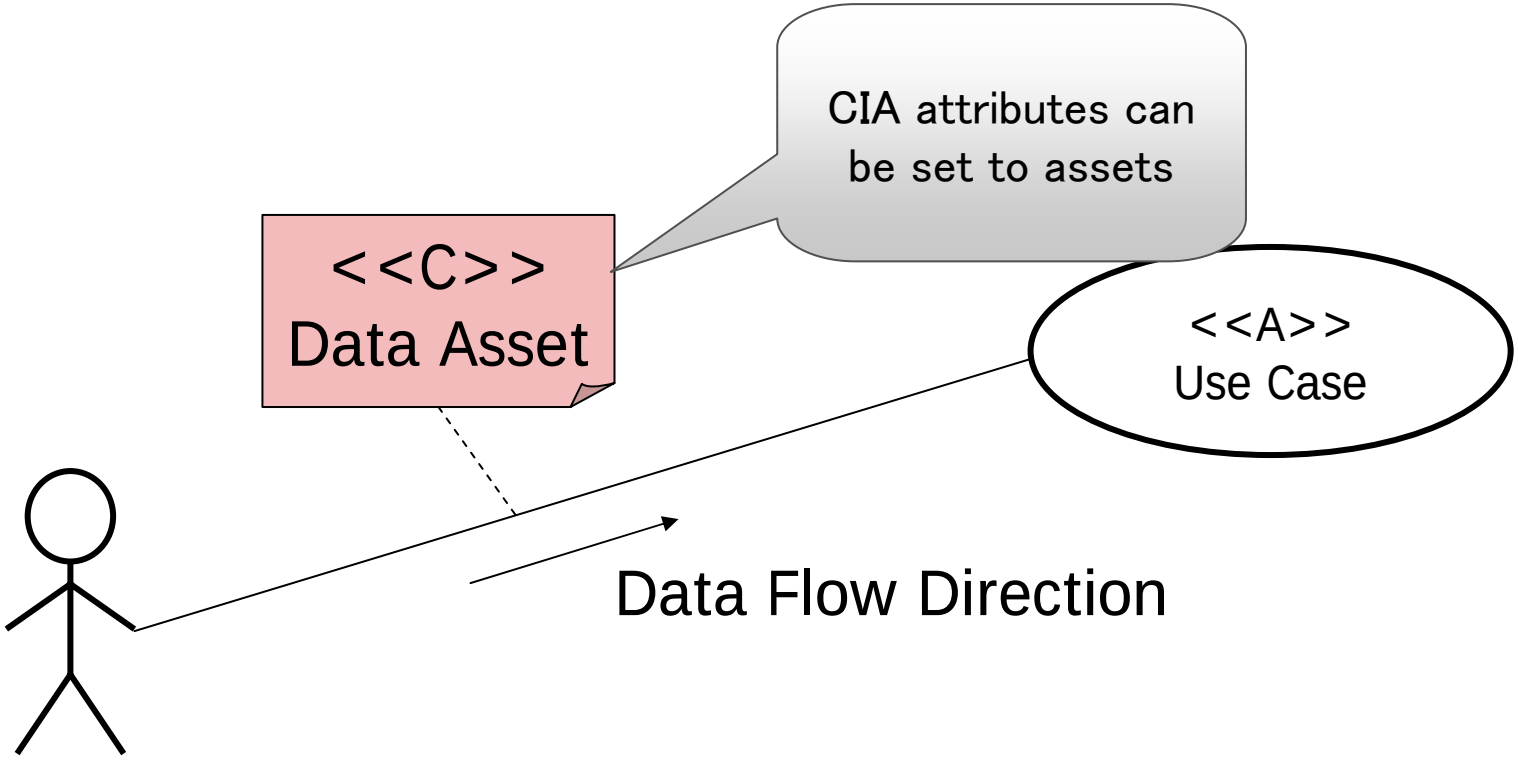
- Extension of misuse cases
  - Mis-actor type extension
  - Data asset extension
  - Misuse case endpoint extension
- Procedure of identifying aspects and pointcuts

Mis-actor Type Extension



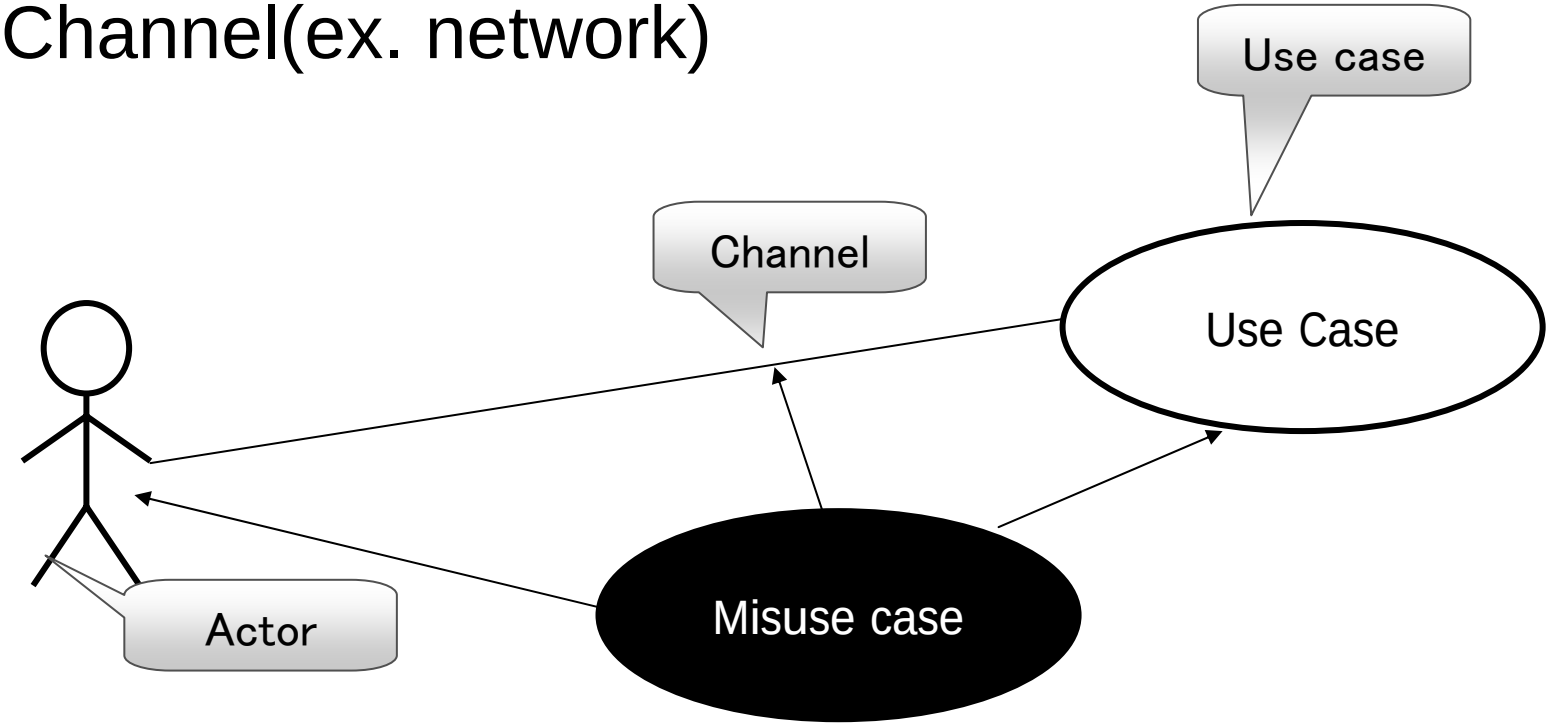
Data Asset Extension

- Data asset description
  ⇒Data-oriented threat analysis



Endpoint Extension

- 3 Types of Endpoint
  - Use case
  - Actor(client)
  - Channel(ex. network)



Procedure in an Analysis Stage

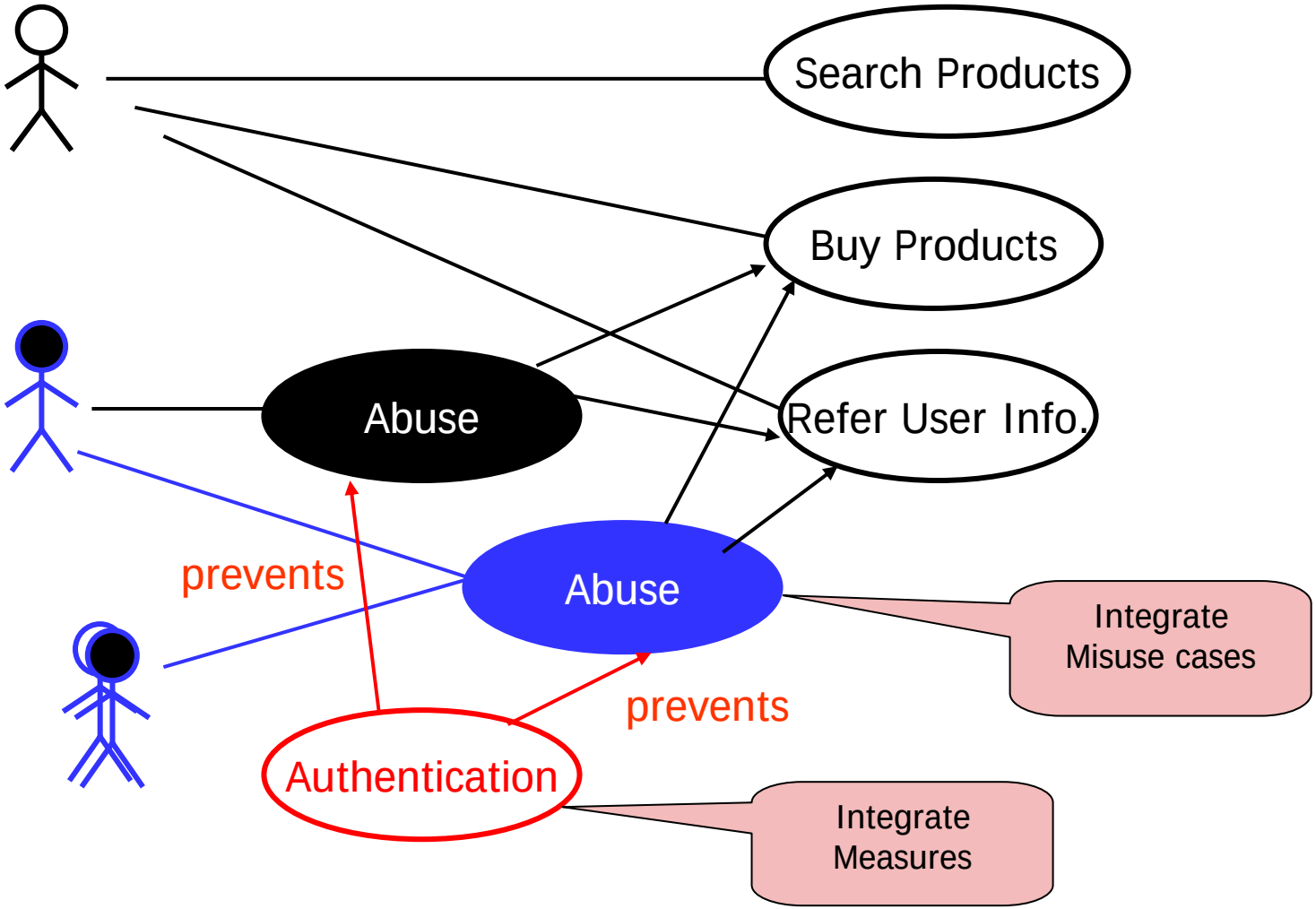
FUJITSU

1. Describe use cases
2. Add data assets
3. Identify threats
   - Adding mis-actors and misuse cases
4. Identify security measures
   - Adding measure use cases
5. Identify aspects and pointcuts

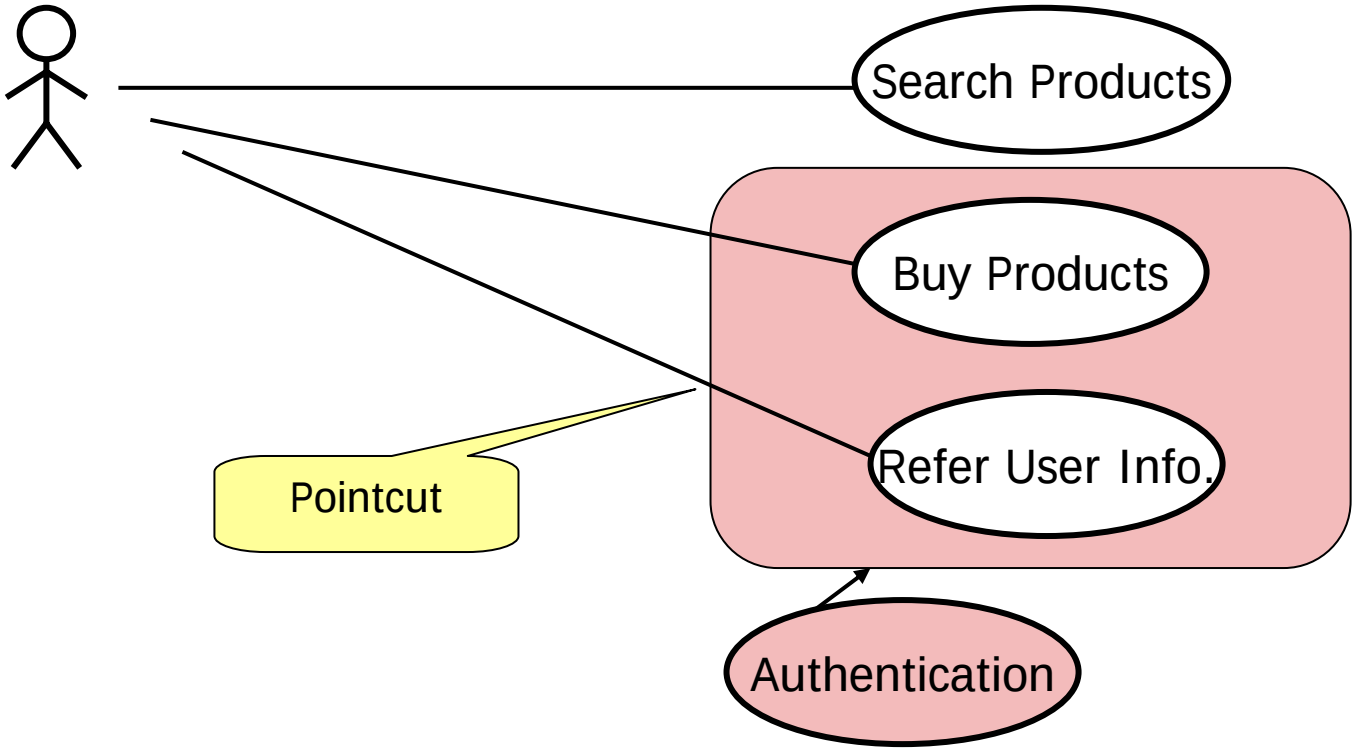
Identifying Aspects

1. Integrate the same kind of threats
2. Integrate the same kind of measures
   ⇒Aspects
3. Identify pointcuts

Integration of Misuse Cases



Identifying Pointcuts



Specifying Pointcuts

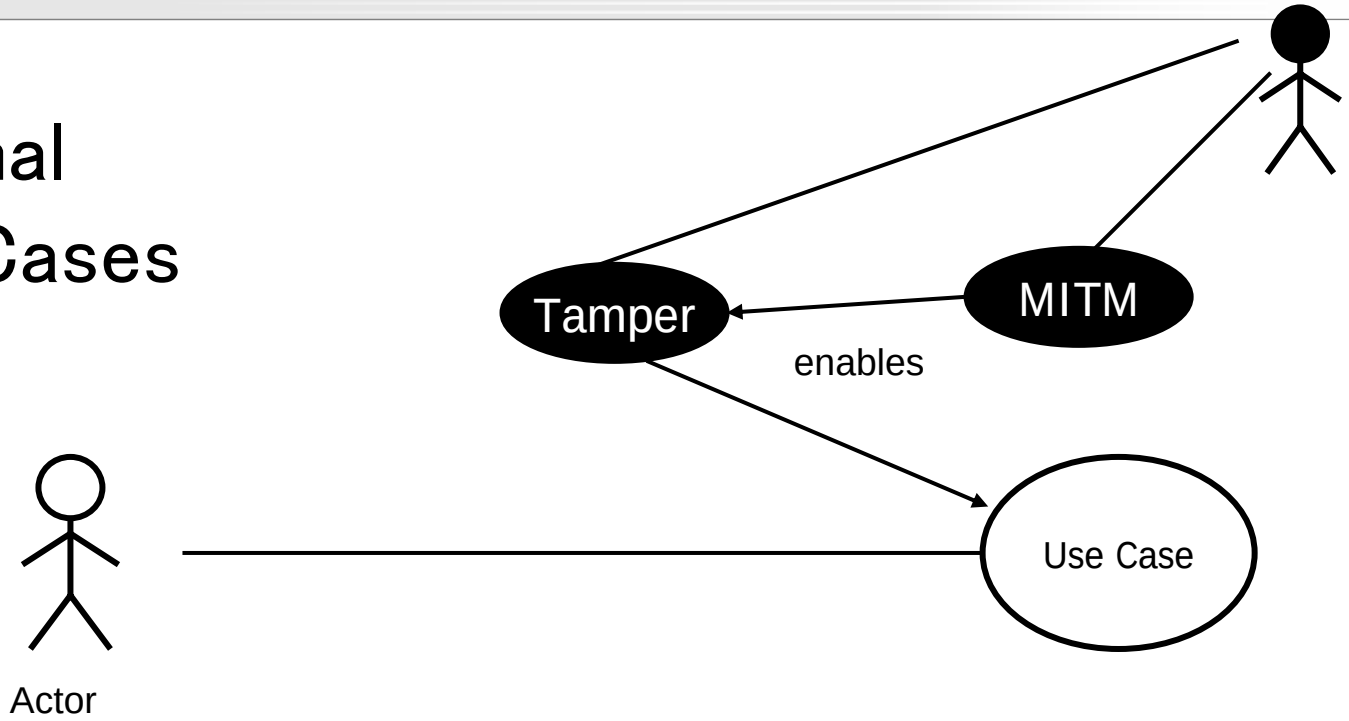
FUJITSU

- Specify the timing that the measure must be injected
  - Before (the use case is executed)
    - Authentication
  - Around
    - Encryption
  - After
    - Logoff
  - Not specified

4.Evaluation

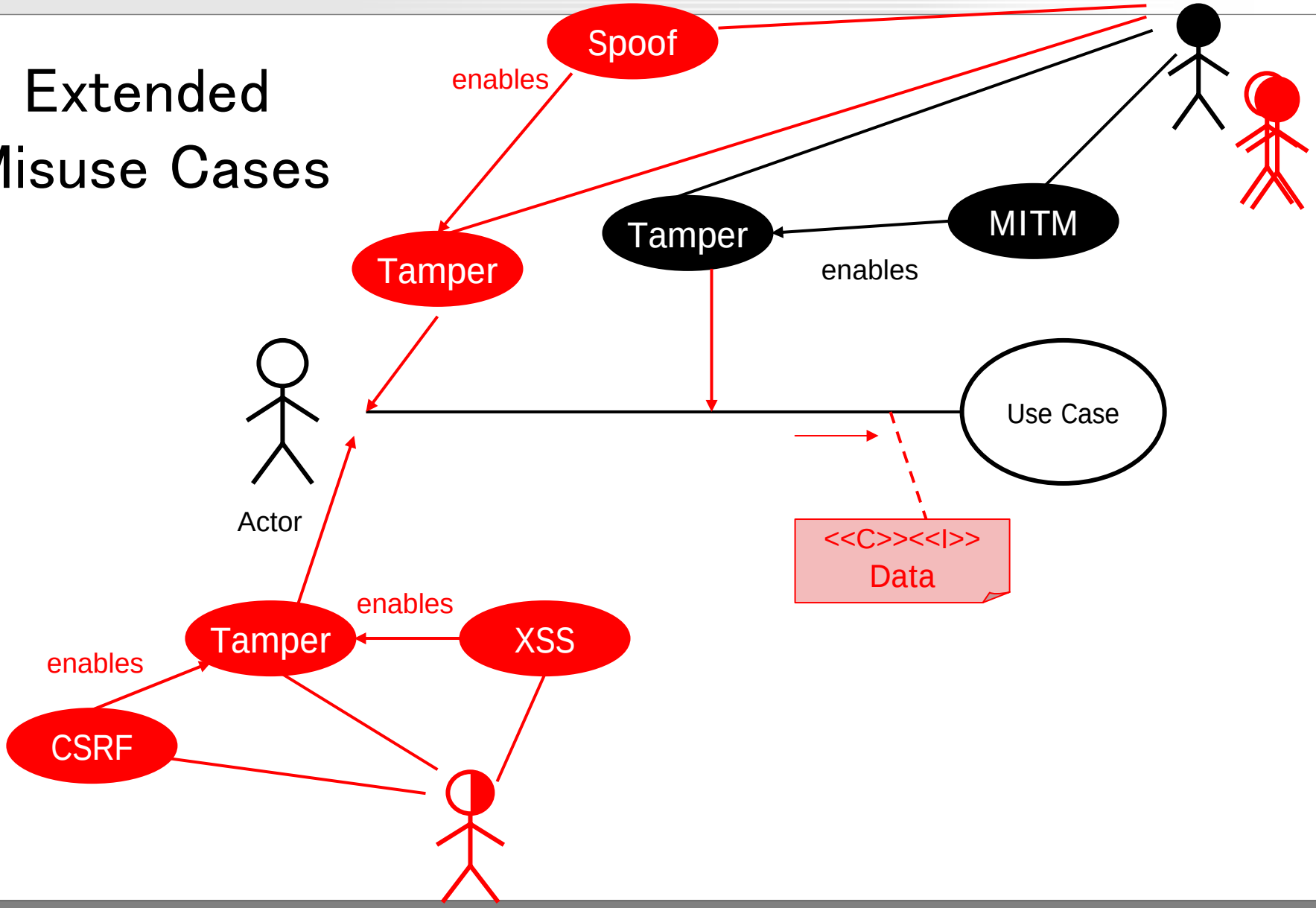
Expressiveness(1)

Original
Misuse Cases



Expressiveness(2)

Extended Misuse Cases



Coverage

FUJITSU

Application to web systems

- Typical threats & measures can be identified (Including Vulnerability with Programming)
  - Injection attacks
  - XSS
  - CSRF
- Aspects & pointcuts can be specified (at the use case level)

5.Conclusion

Conclusion

FUJITSU

- Aspect may be good for security
  - not ad-hoc AOP.
  - Analysis in early stages needed
- Security requirements (aspects) identification with extending misuse cases
  - For easier threat-identification
  - Clarified correspondence between threats and measures
  - Methods for identifying crosscutting points
- Application to the web domain
  - Threats at programming are predictable
  - Able to patterning in the web domain

Future Works

- Security framework with aspects
  - UML+Java+AspectJ
- Developing security patterns
  - Security analysis patterns
  - Security design patterns

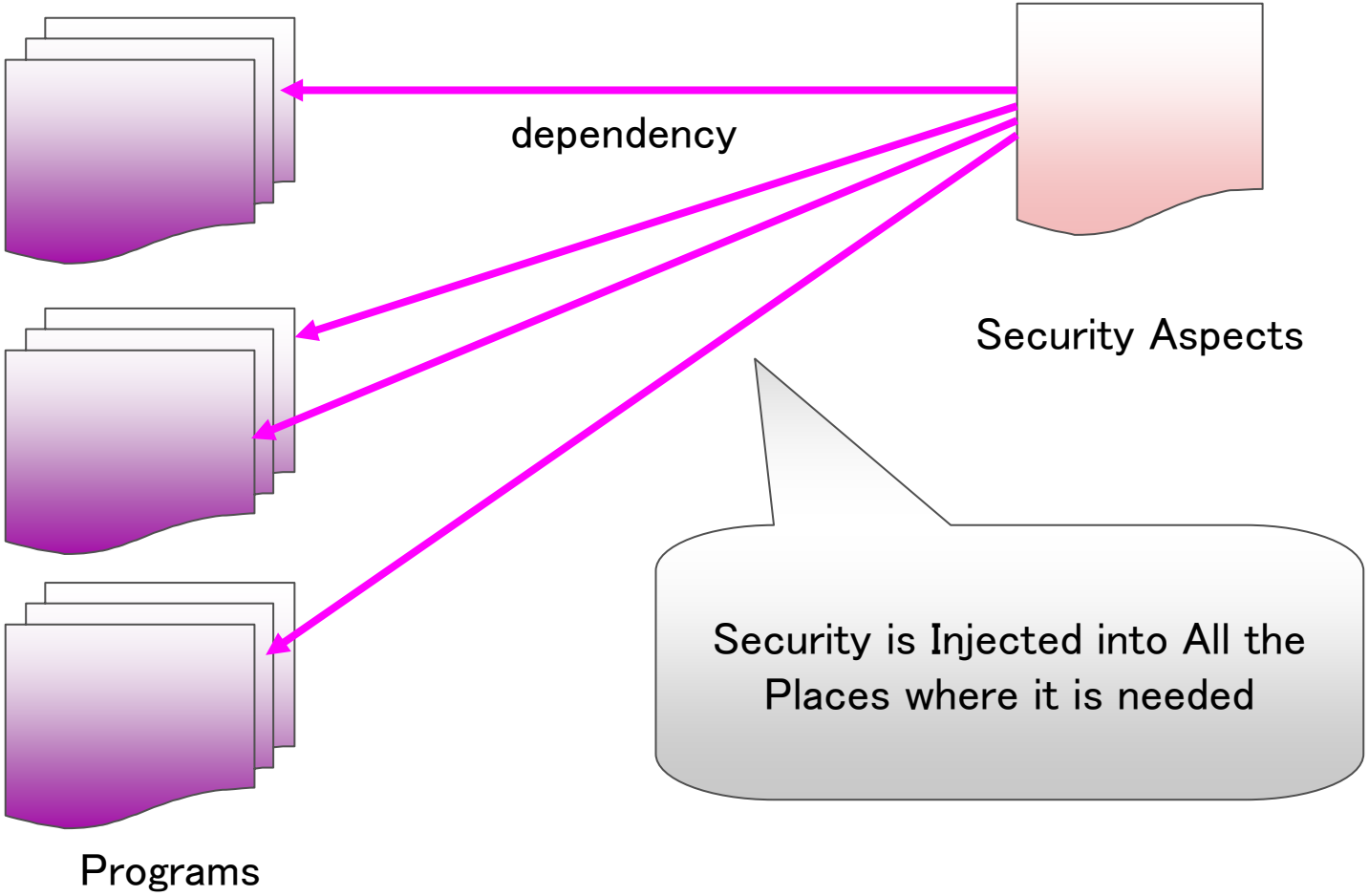
FUJITSU

THE POSSIBILITIES ARE INFINITE

Ideal World with Aspects(2)

FUJITSU

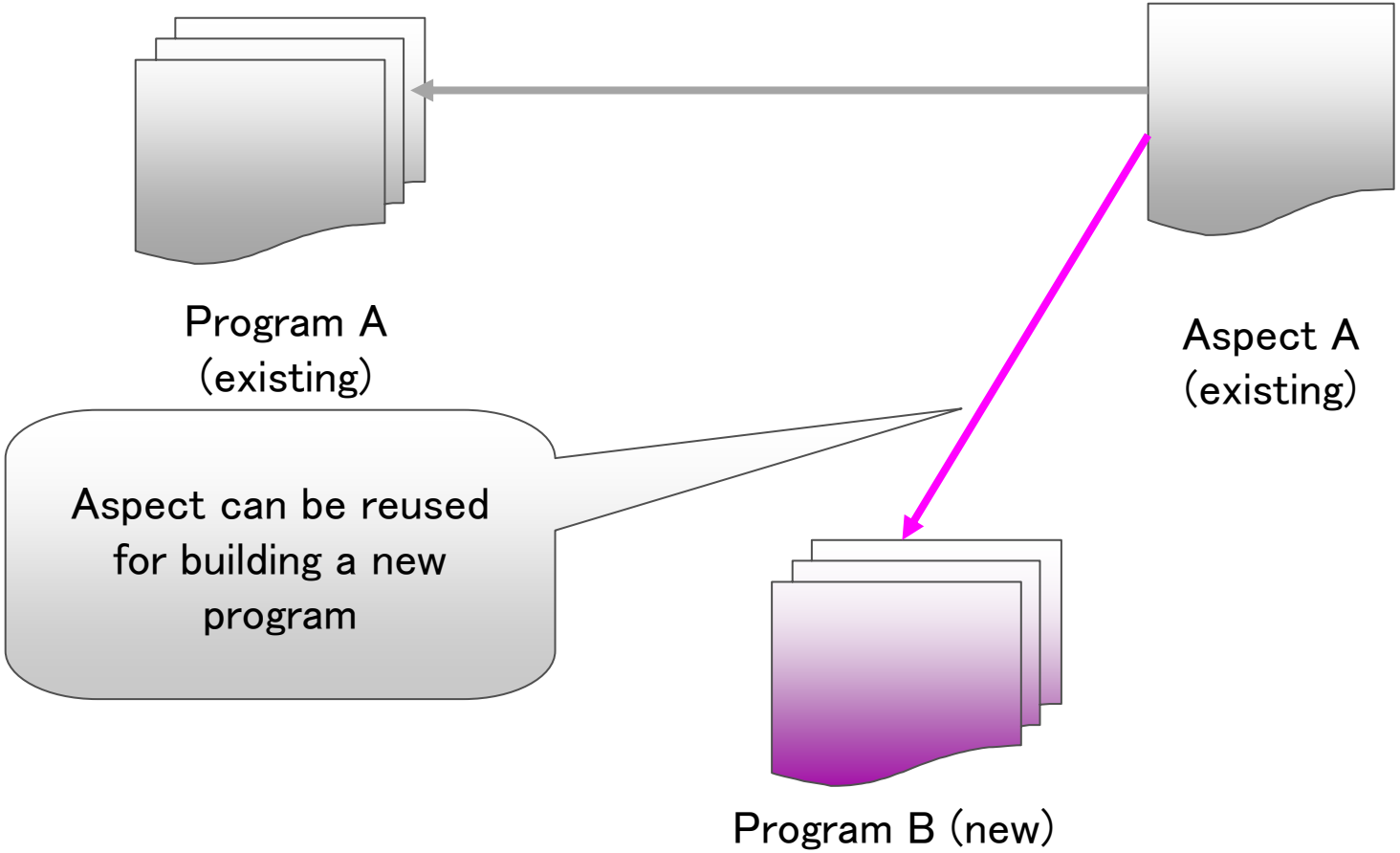
Coverage



Ideal World with Aspects(3)

FUJITSU

Reusability



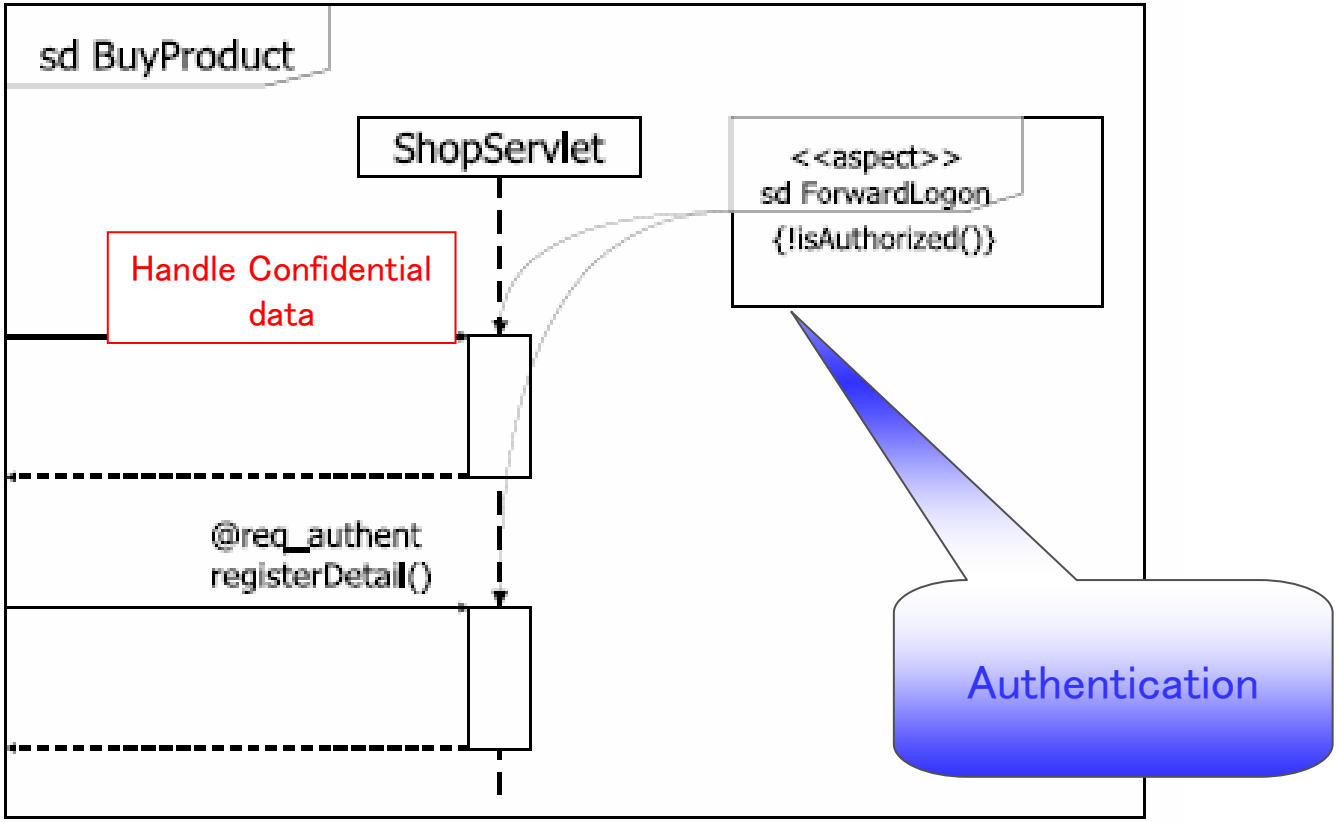
Requirements in the Analysis Stage

- All the potential threats must be identified
- Security measures (aspects) must be identified for all the threats
- Pointcut-candidates must be specified

Designing Security Aspects(1)

FUJITSU

- Designing and Maintain Crosscutting Points (for Coverage and Reusability)



Designing Security Aspects(2)

FUJITSU

- Automatic Code Generation for More Coverage & Reusability
- Developing Security Design Pattern

Backgrounds of Backgrounds

FUJITSU

Need for Programming Control

"Only necessary to use that library?"

"Only to program in that manner?"

Why Analysis/Design/Testing is important?

- Persons Make Programs
- Programmers obtain freedom and power
- Most users are not the programmers