Designing Fulfilling Test Cases with Test Aspect Model

Akiharu Satoh

NPO ASTER

Minato-ku, Tokyo, Japan

Yuzuru Harayama

Panasonic ITS Co., Ltd.

Yokohama, Kanagawa, Japan

Shizuka Ban

NPO ASTER

Minato-ku, Tokyo, Japan

Kunio Yamamoto

NPO ASTER

Minato-ku, Tokyo, Japan

Contents

1. Introduction
2. Use case of Test Aspect Model
3. Definition of Test Aspect and relationships
4. Examples of Test Aspect Model
5. Conclusion

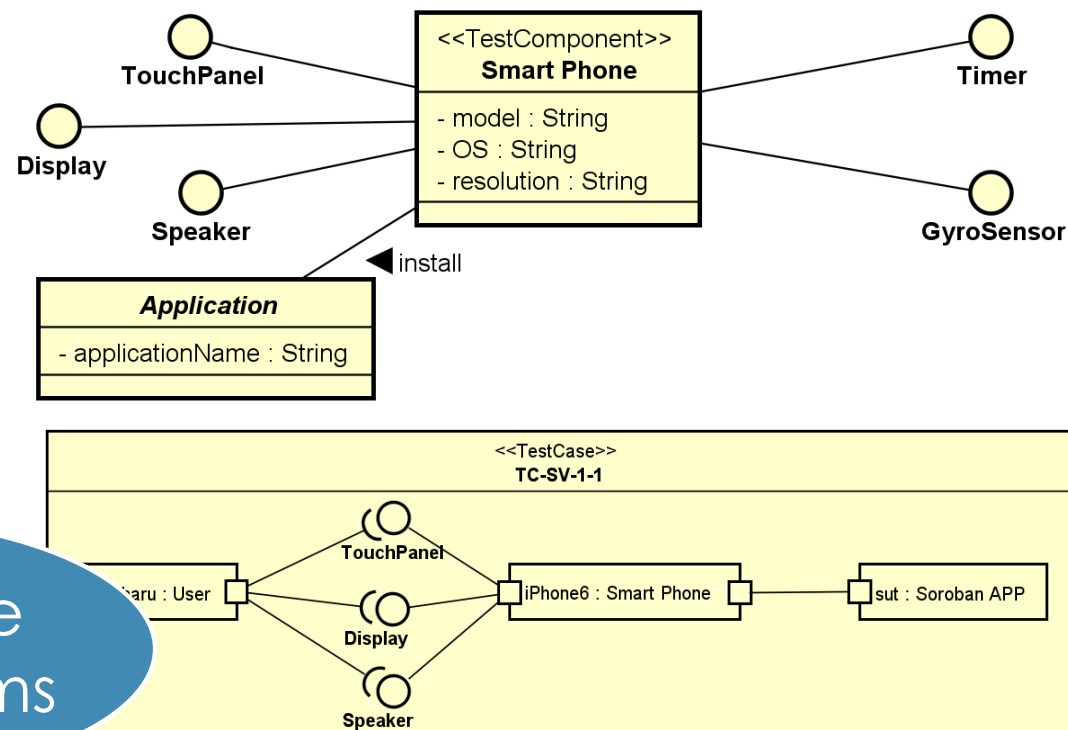# 1. Introduction

Background of our proposal

UML testing profile 2.0(UTP2) allows test engineers to describe the architecture of test systems in more detail. However, UTP2 is not sufficient to describe the architecture of test suites.
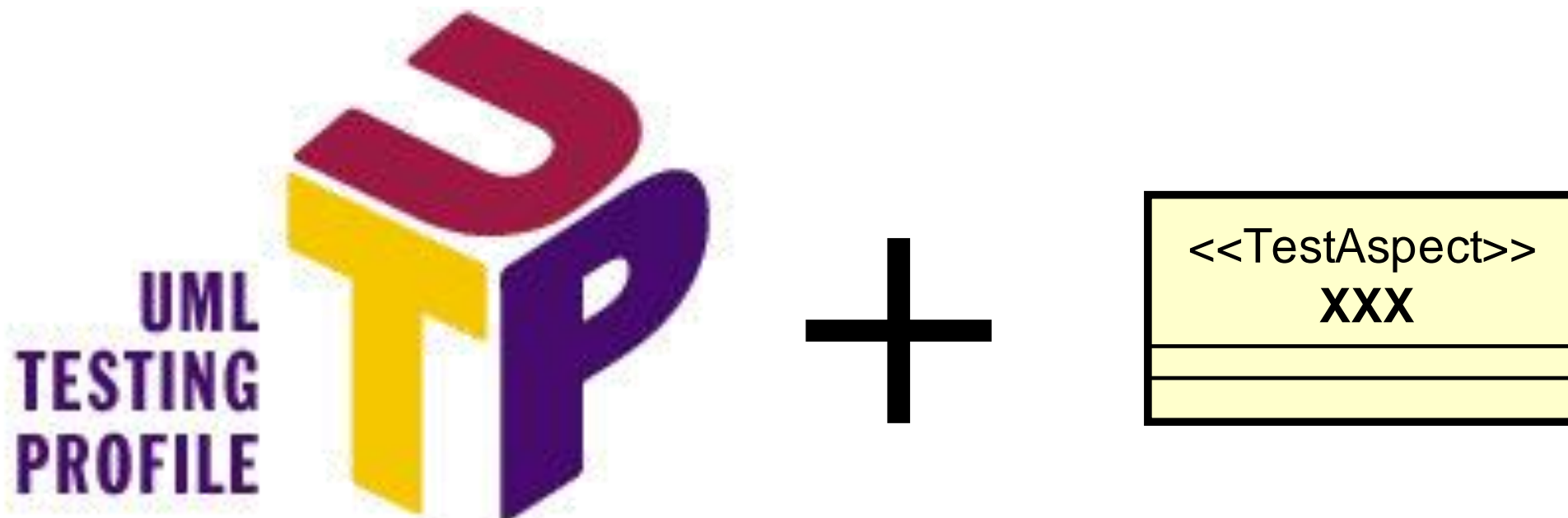
Background of our proposal

So, a year ago, we proposed the **test aspect**, which reveals the aspect to be tested in InSTA 2018. One of the features of the test aspect is it has UTP-enhanced notation.

Our new proposal

In InSTA 2018, our proposal gave examples of a test aspect model only, but not use cases which derive test cases from a test aspect model.
We will give more examples of the use cases of a test aspect model, and show the merit of considering a test aspect model in the test design process.
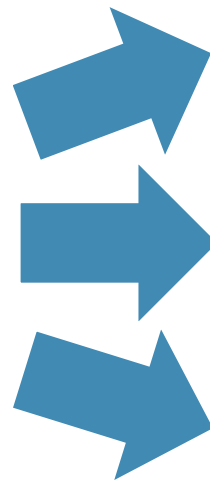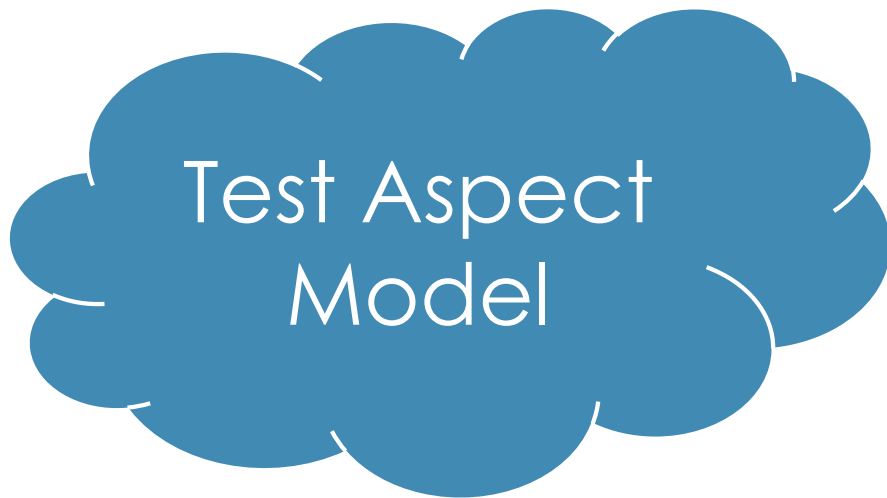
Test Aspect Model

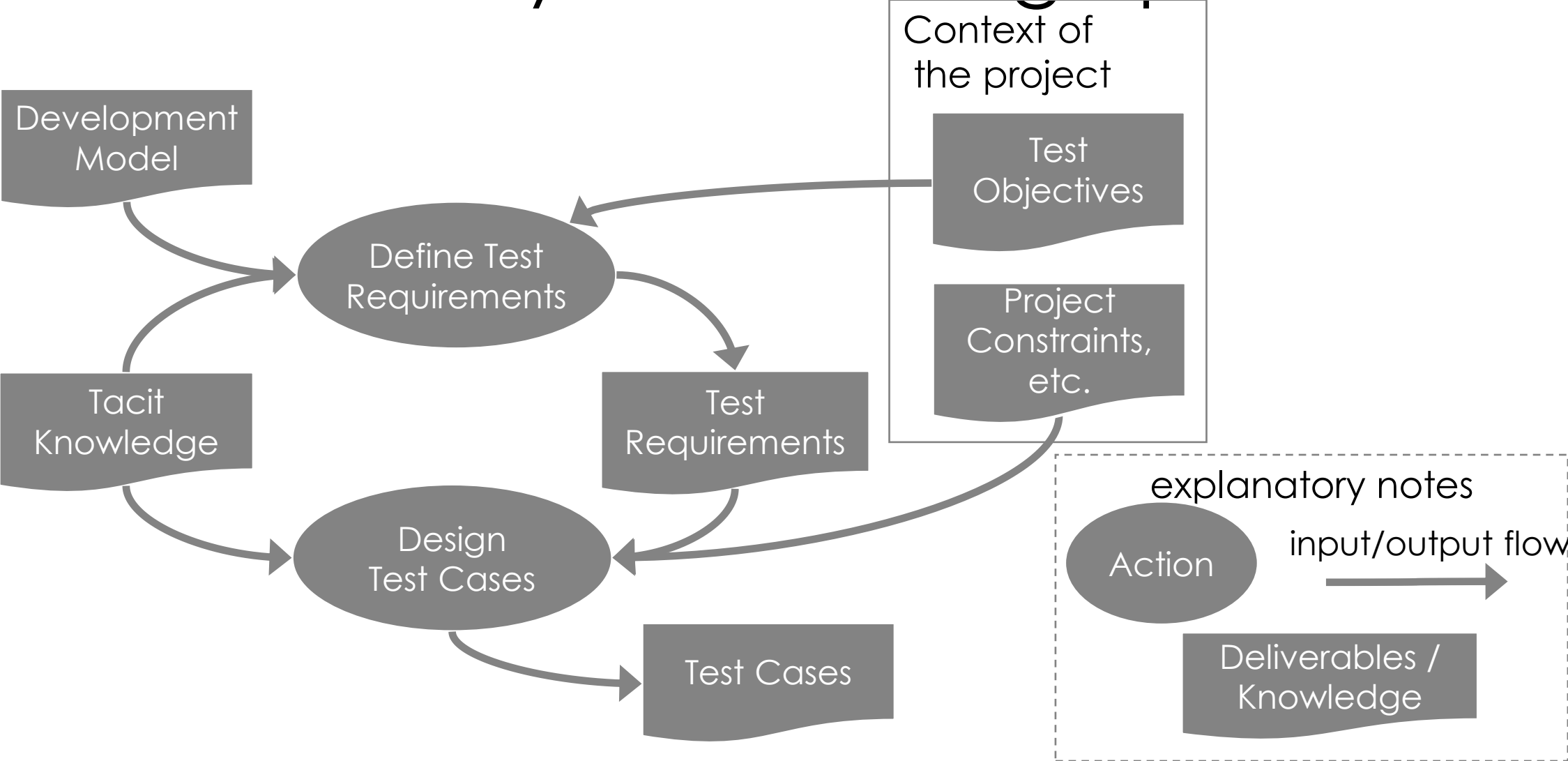Sharing knowledge about test objects

Extracting new test requirements and making test cases fulfilling

Complementing development models

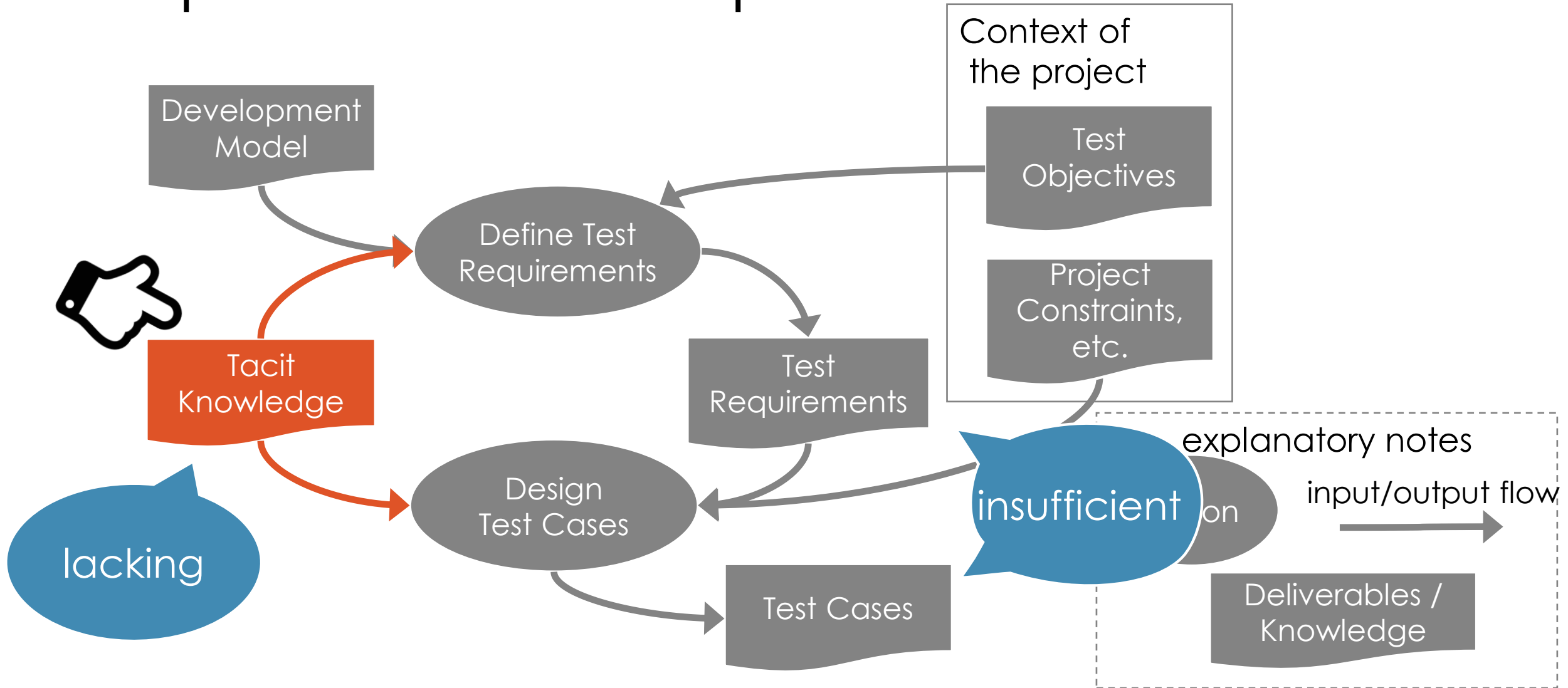(These use cases are shown later.)

2. Use case of Test Aspect Model

Usual test analysis and design process

A problem of the process

Solution: making tacit knowledge explicit

Test engineers have various concerns for the development model from their own tacit knowledge. These concerns helps in extracting aspects to be tested.

tacit knowledge

information on bugs found in the past

experiences as user

experiences as developer

concerns

development model
(ex. login page specification)

look in detail

whether the web system allows multiple login

what will happen if a user enters an expired password

whether user input is encrypted
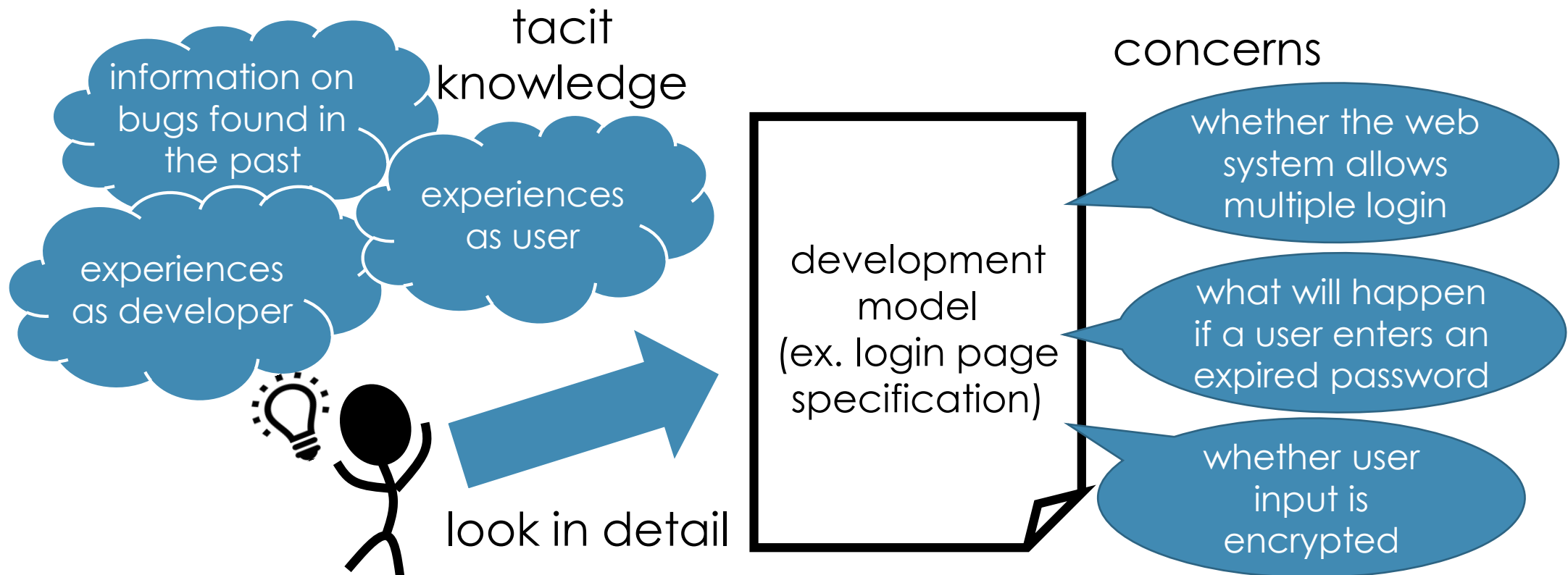
The *test aspect* and *test aspect model*

The *test aspect* is the aspect to be tested derived from the test engineers' concerns.
The *test aspect model* is the set of test aspects equipped with their relationships.
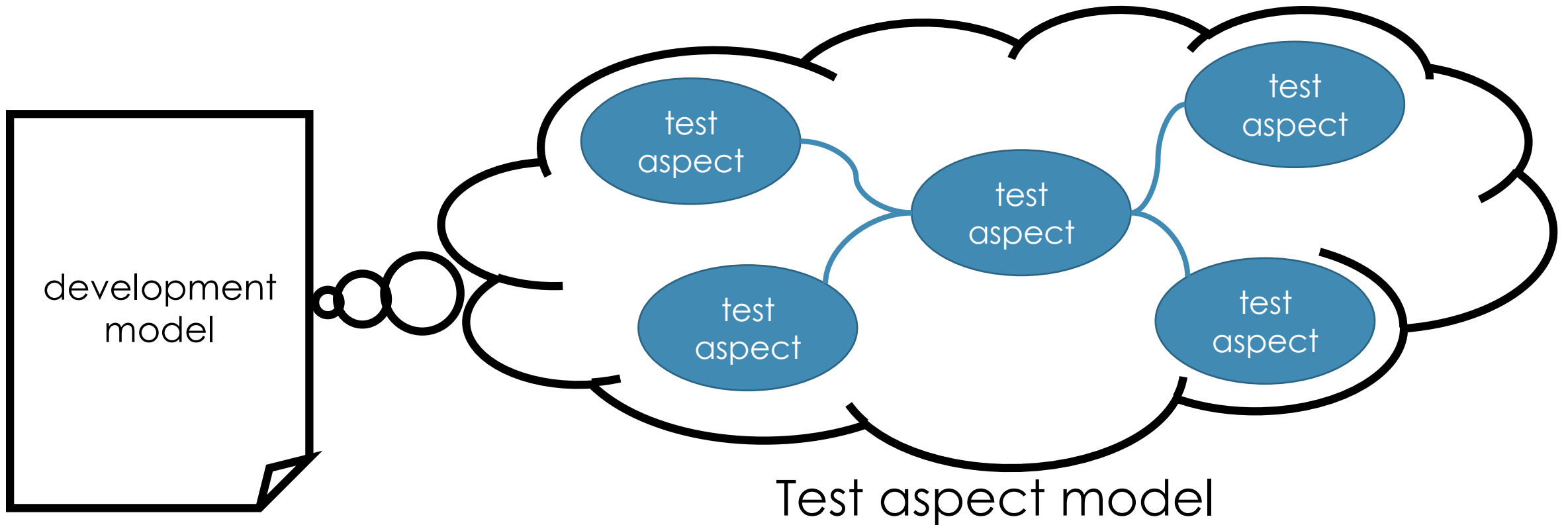
Test aspect model

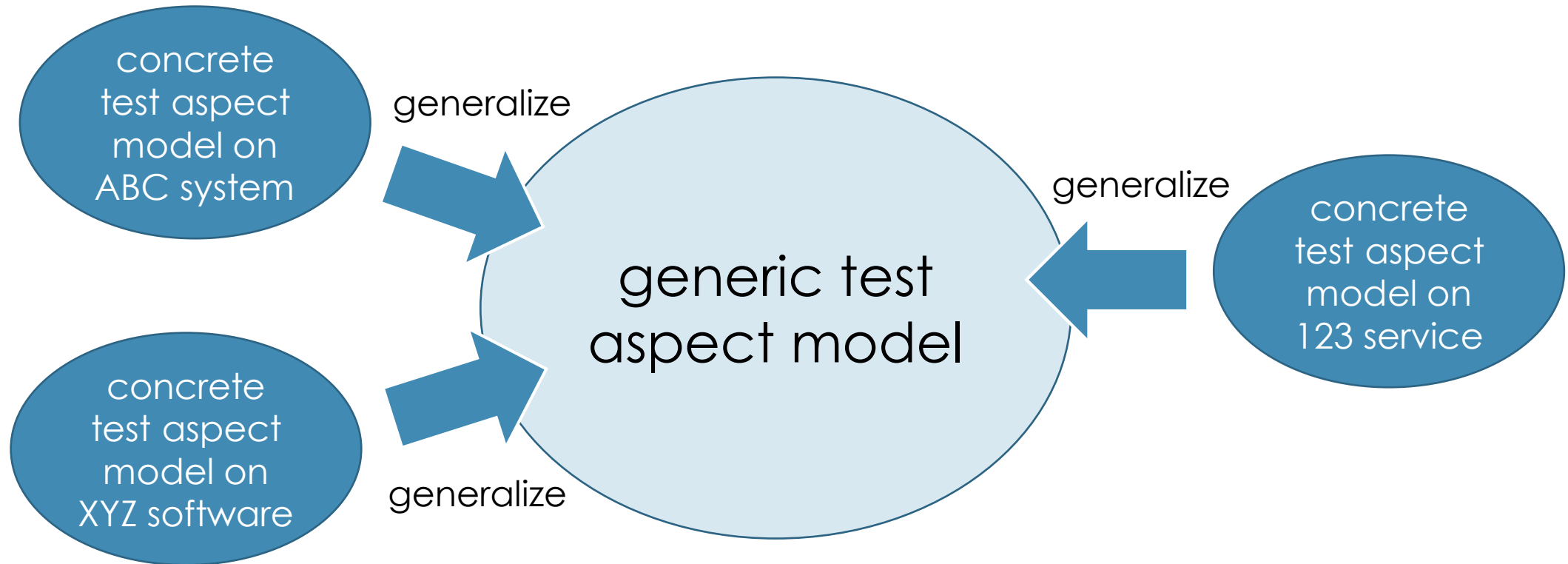# The *generic* test aspect model

Sharing knowledge more effectively, test aspect model should not be specialized in a certain test project. Then, we propose the *generic test aspect model*.
The generic test aspect model does not contain the information about certain test projects.

concrete test aspect model on ABC system

generalize

concrete test aspect model on XYZ software

generalize

generic test aspect model

generalize

concrete test aspect model on 123 service

Use cases of test aspect model
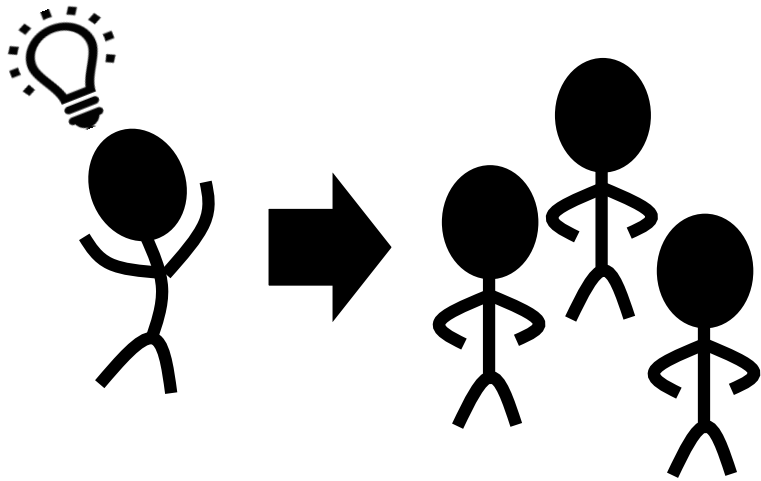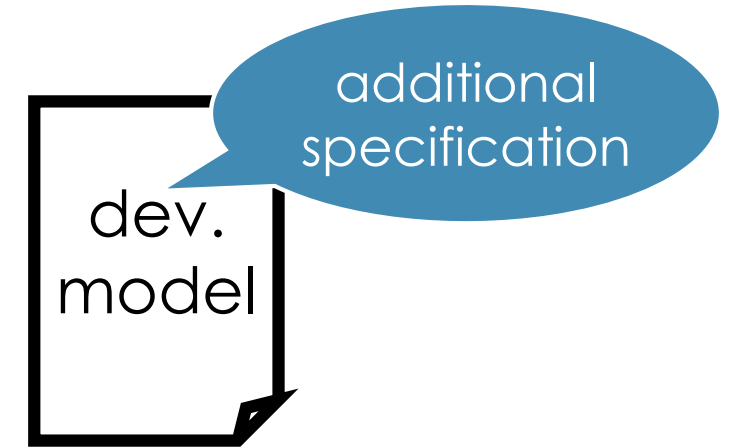
Proposal of new test analysis and design process with test aspect model
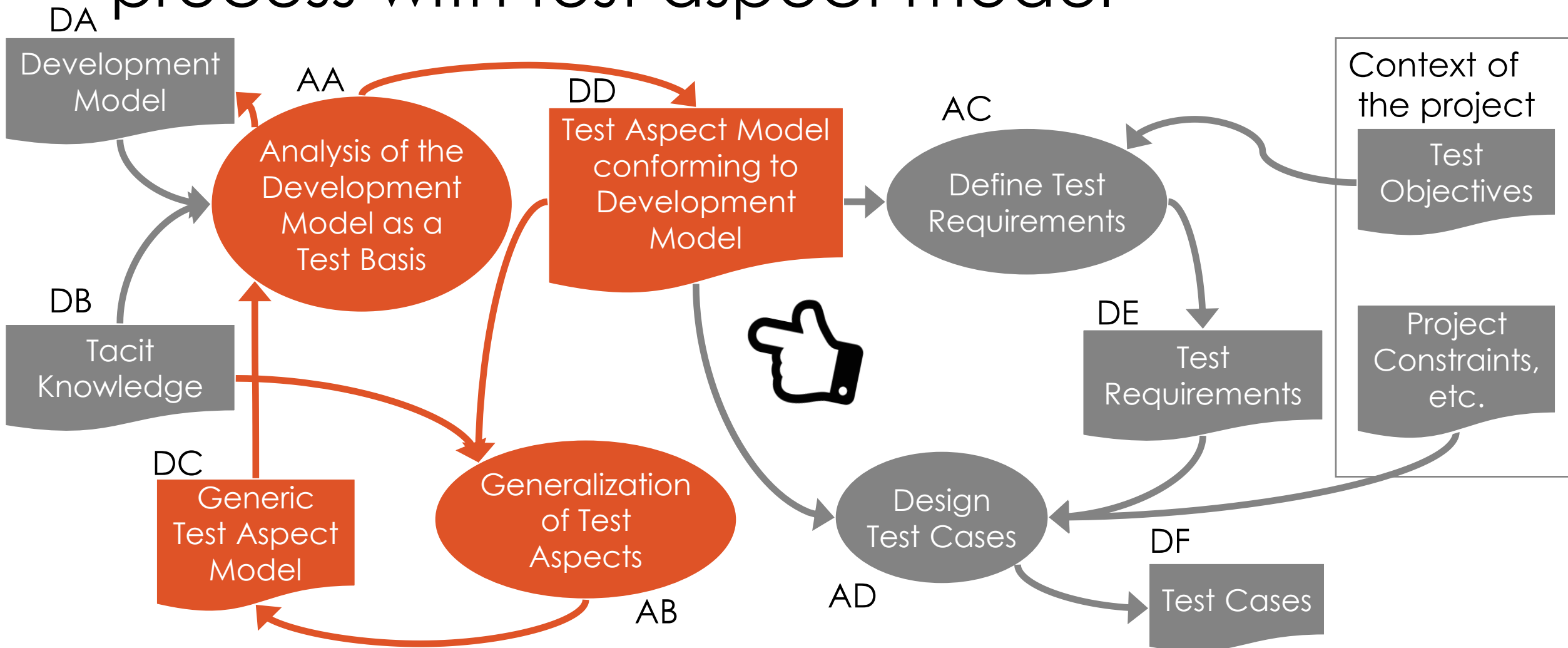
3. Definition of Test Aspect and relationships

Notation of *test aspect*

The *test aspect* is the aspect to be tested derived from the test engineers' concerns.
In the UML notation, a test aspect is represented by class with stereotype <<TestAspect>>

<<TestAspect>>
test aspect

Relationships between two test aspects

is-a
"B is a A"

```
<<TestAspect>>
A
```

△

```
<<TestAspect>>
B
```

A and B are
in the relation of
taxonomic hierarchy

part-of (has-a)
"A has a B"

```
<<TestAspect>>
A
```

◇

```
<<TestAspect>>
B
```

A and B are
in the relation of
possessive hierarchy
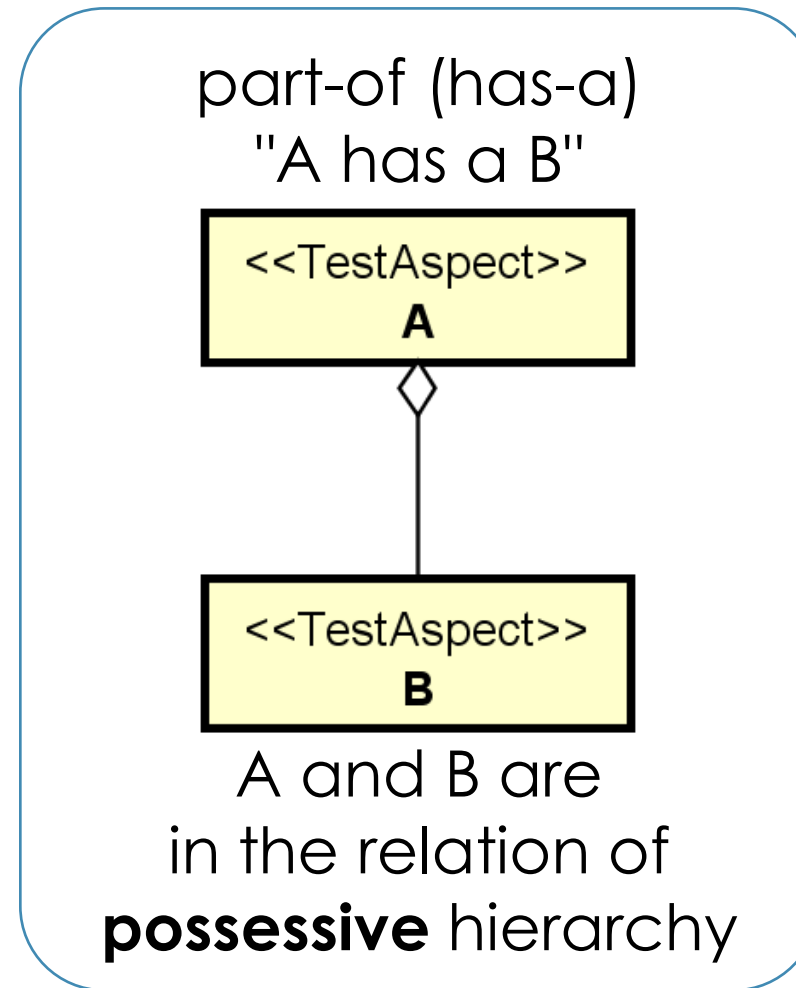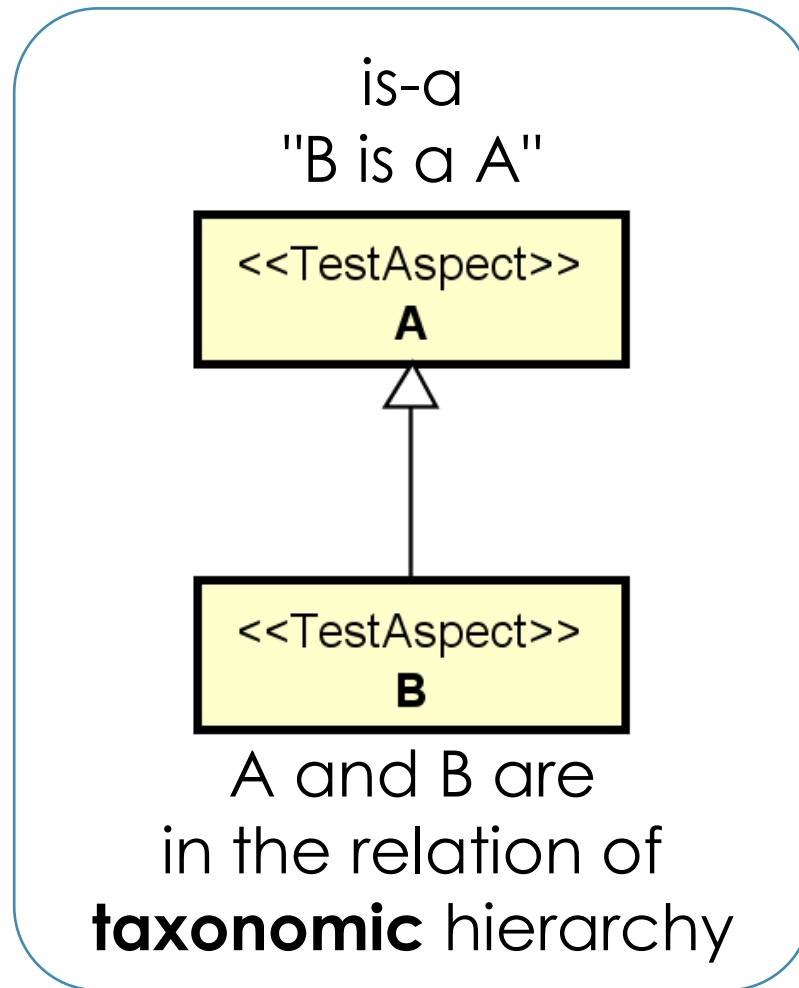
Relationships between test aspects and other elements

We define two kinds of relationships, *extract* and *suggest*, as relationships between a test aspect and one of other model elements.
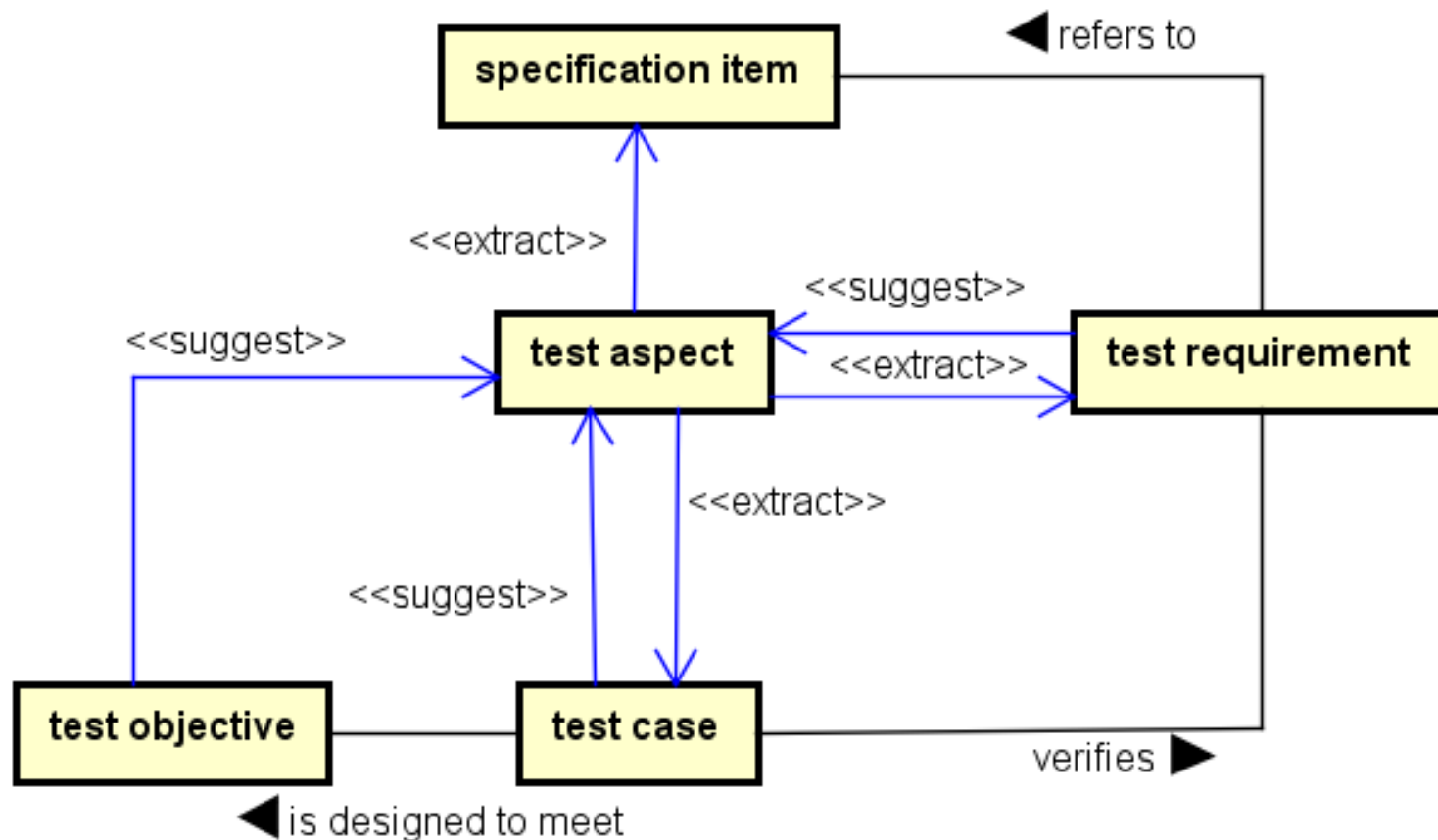
4. Examples of Test Aspect Model

The test base

The test base of our example is the ATM in the InterBank Exchange Network (IBEN) shown in the UTP2 document.
We assume that the test level is the integration test.

country A bank network

SWIFT network

ATM

country B bank network

ATM

Test scope
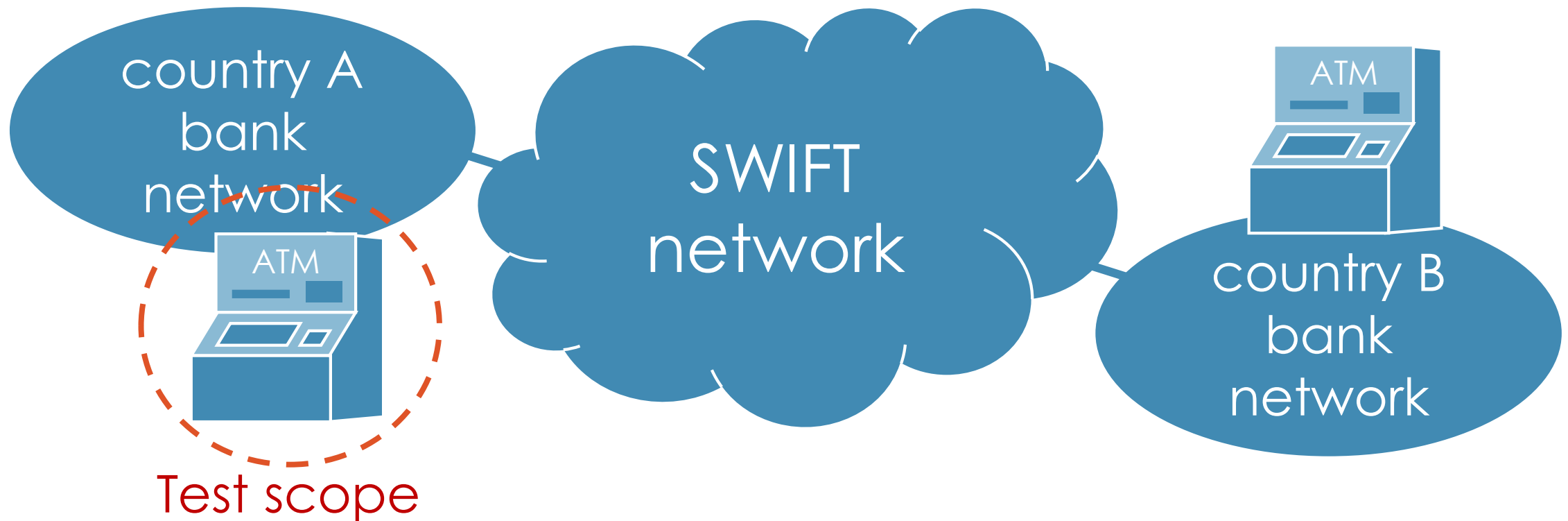
The Development Model of the ATM

The ATM software consists of 5 packages as follows.
Then, class diagrams are given for every package.

The Development Model of the ATM
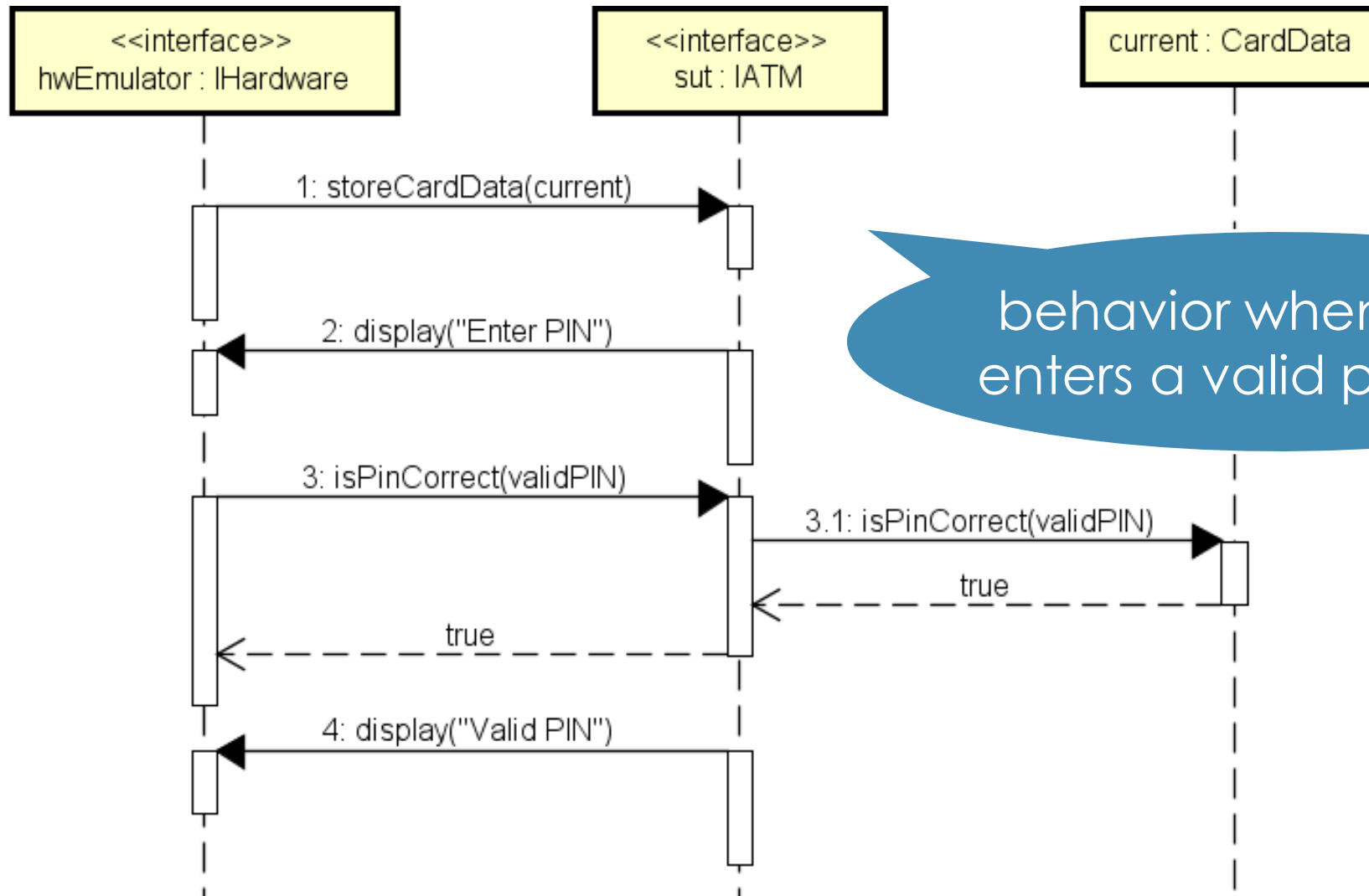
Trivial test requirements and test cases

These are trivial test requirements and test cases lead from the development model without the test aspect model.

<<TestContext>>
Integration Test of Banking

| <<TestCase>> **validPIN_TC** | <<TestCase>> **invalidPIN_TC** | <<TestCase>> **retryEnterPIN_TC** | <<TestCase>> **validWiring_TC** | <<TestCase>> **authorizeCard_TC** |

<<verifies>>   <<verifies>>   <<verifies>>   <<verifies>>   <<verifies>>

<<TestRequirement>> **approveAuthorization**
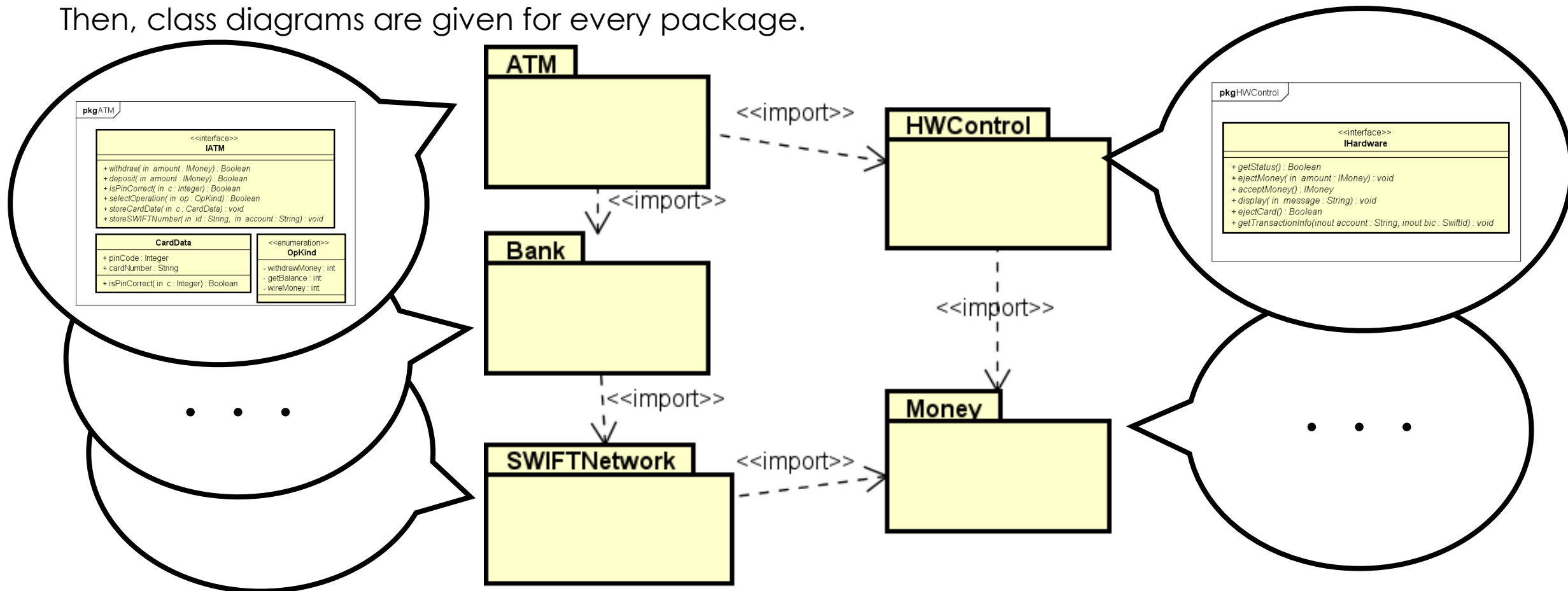
<<TestRequirement>> **approveTransaction**
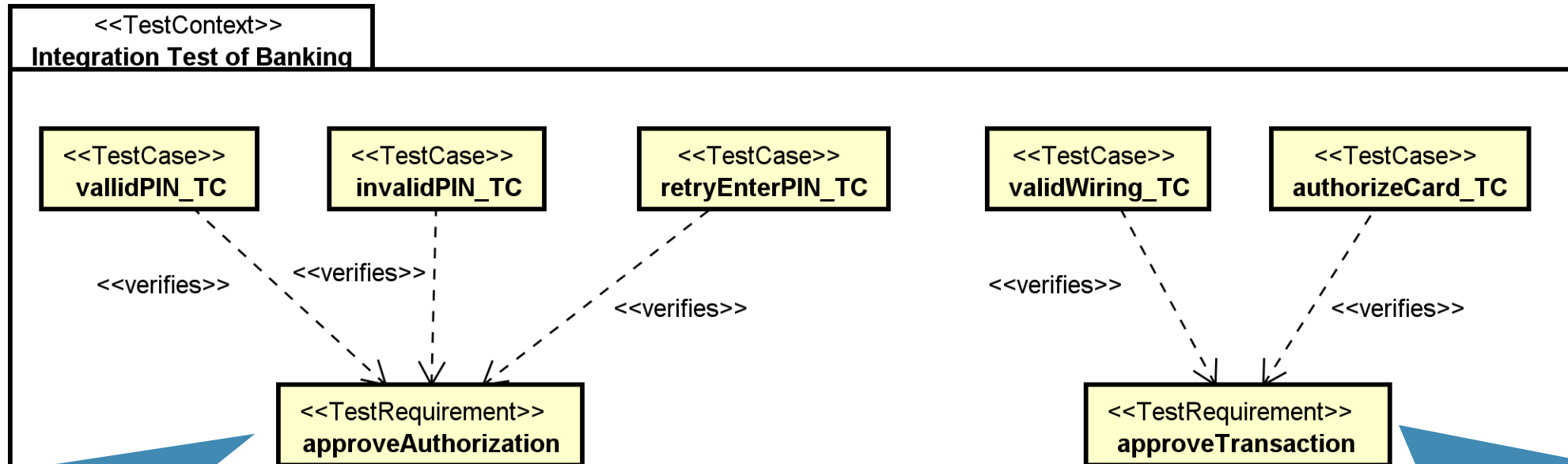
The hardware terminal (HWControl) provides user's card and user's pin-code. The ATM shall authorize this card and its pin-code.

After a successful authorization of user's data, money shall be deposited into the bank. The ATM shall ensure a correct transaction communication with the Bank.

Trying the improved test analysis process
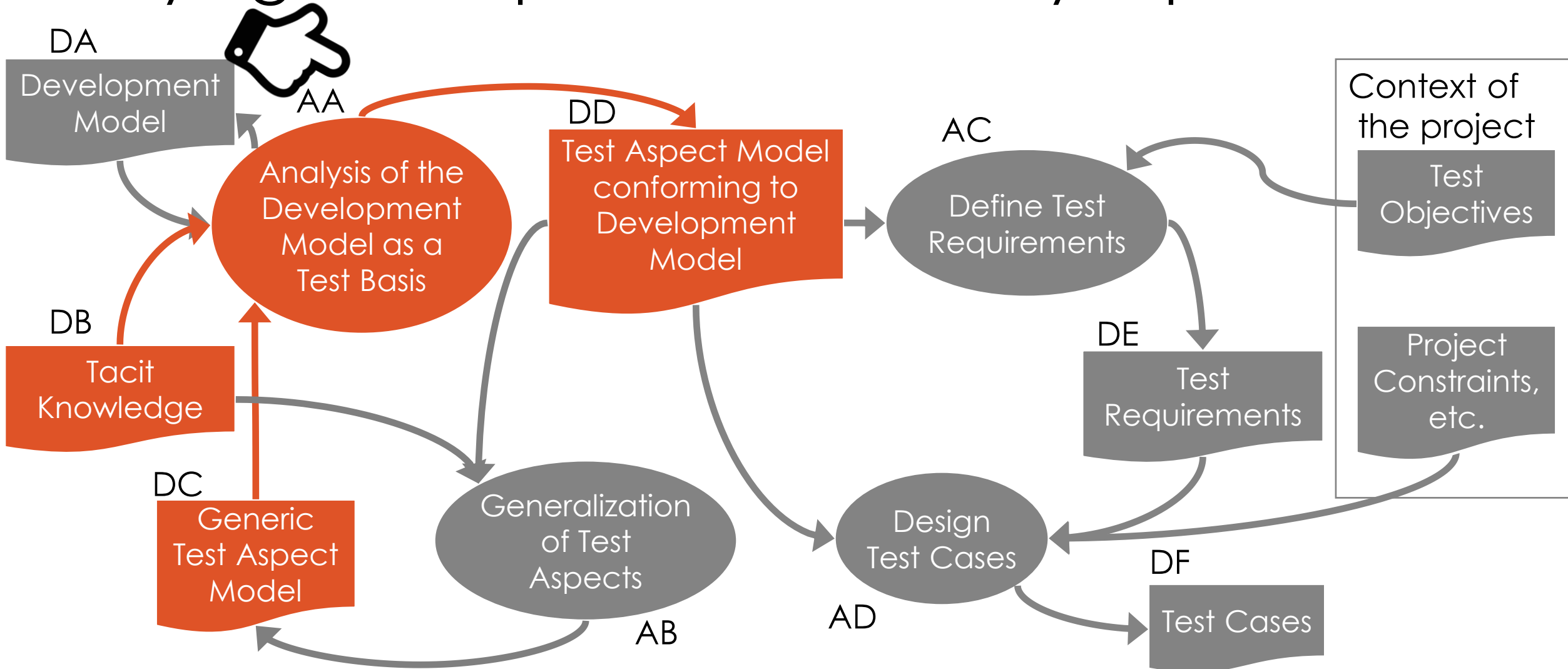
Discovering concerns

Looking at the specifications described in the test base, test engineers indicate their concerns.

concern

looking in detail

Test base
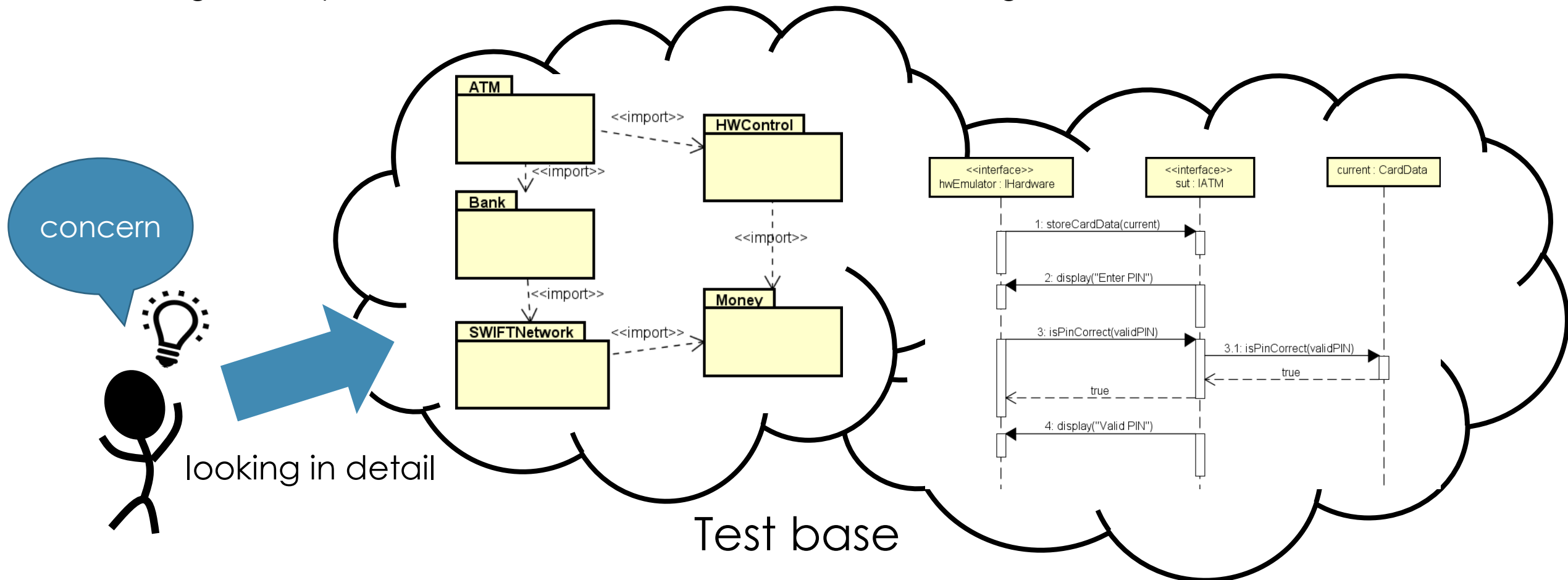
From concerns to test aspects

Test engineers indicate their concerns as test aspects in the class diagram.

How is the response speed of each method such as *isPinCorrect* appropriate?

→ <<TestAspect>>
response time

What is "invalid" of an invalid card? And how does an invalid card behave?

→ <<TestAspect>>
what is 'invalid' ?

Is it possible to conjecture personal information from error messages or not?

→ <<TestAspect>>
inference of personal data

and so on ...
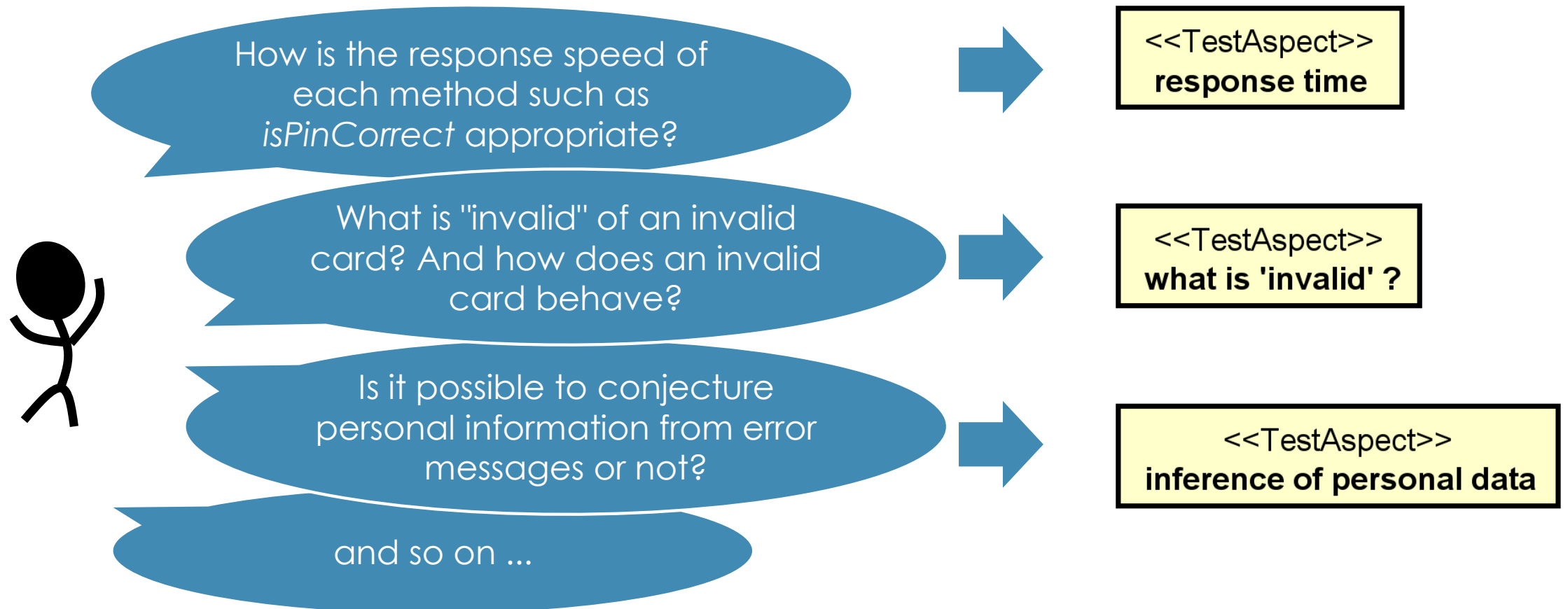
Creating a test aspect model
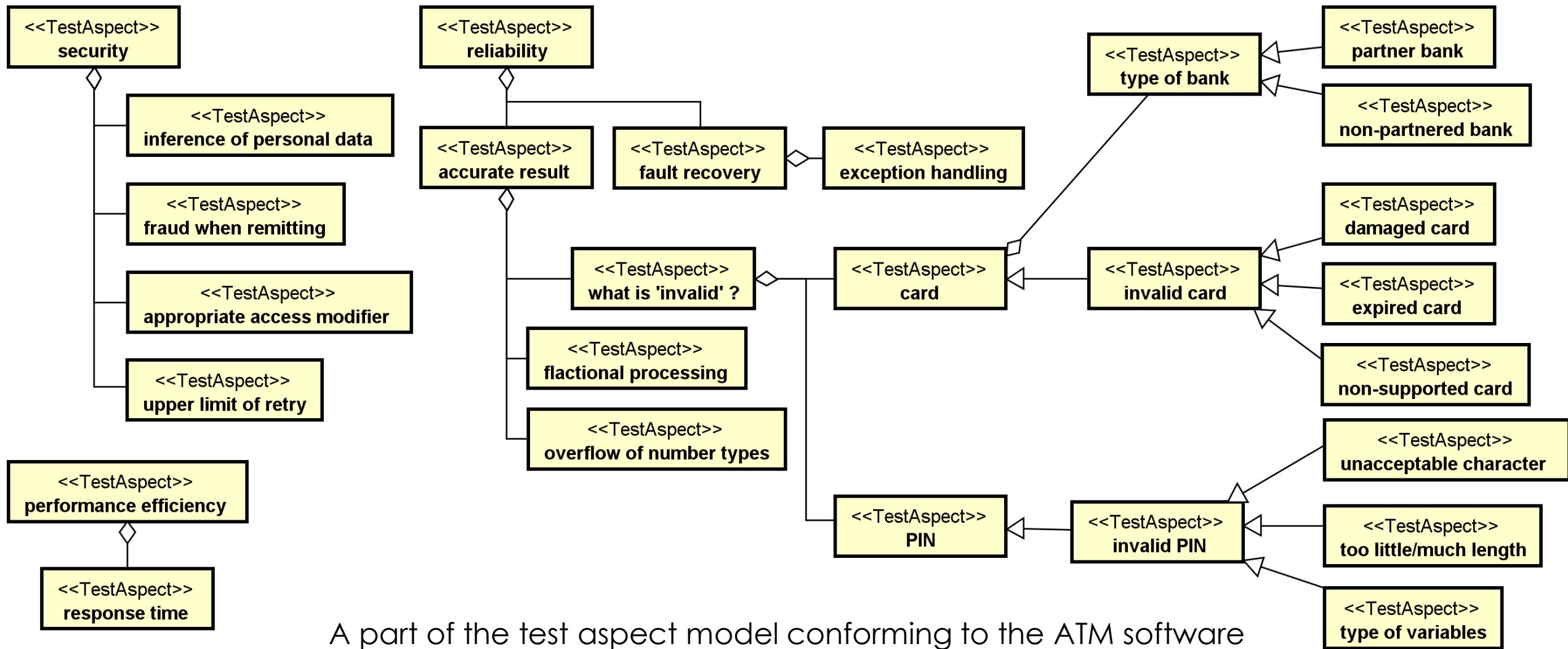
A part of the test aspect model conforming to the ATM software

Creating a test aspect model
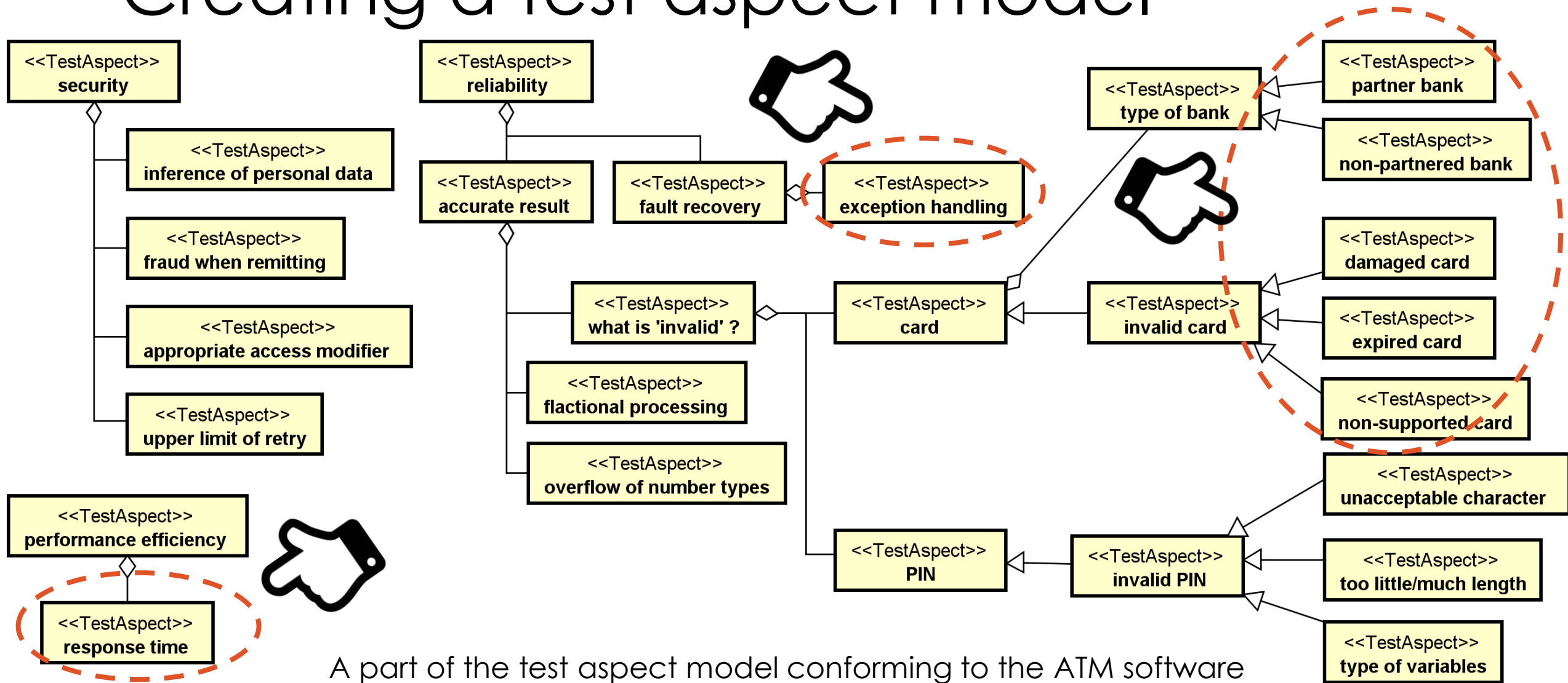
A part of the test aspect model conforming to the ATM software

Use case 1: Extracting new test requirements

The consideration of test aspects extracts new test requirements.

<<TestAspect>>
Test Aspect Model

<<TestAspect>>
exception handling

<<TestAspect>>
non-partnered bank

<<extract>>

<<extract>>

Test Requirements

<<TestRequirement>>
exceptionHandling

<<TestRequirement>>
cardCompatibility

<<TestRequirement>>
approveAuthorization

<<TestRequirement>>
approveTransaction

If one of the methods returns neither true nor false, an appropriate exception handling will occur.

If an incompatible card is entered in ATM, ATM software will return false.
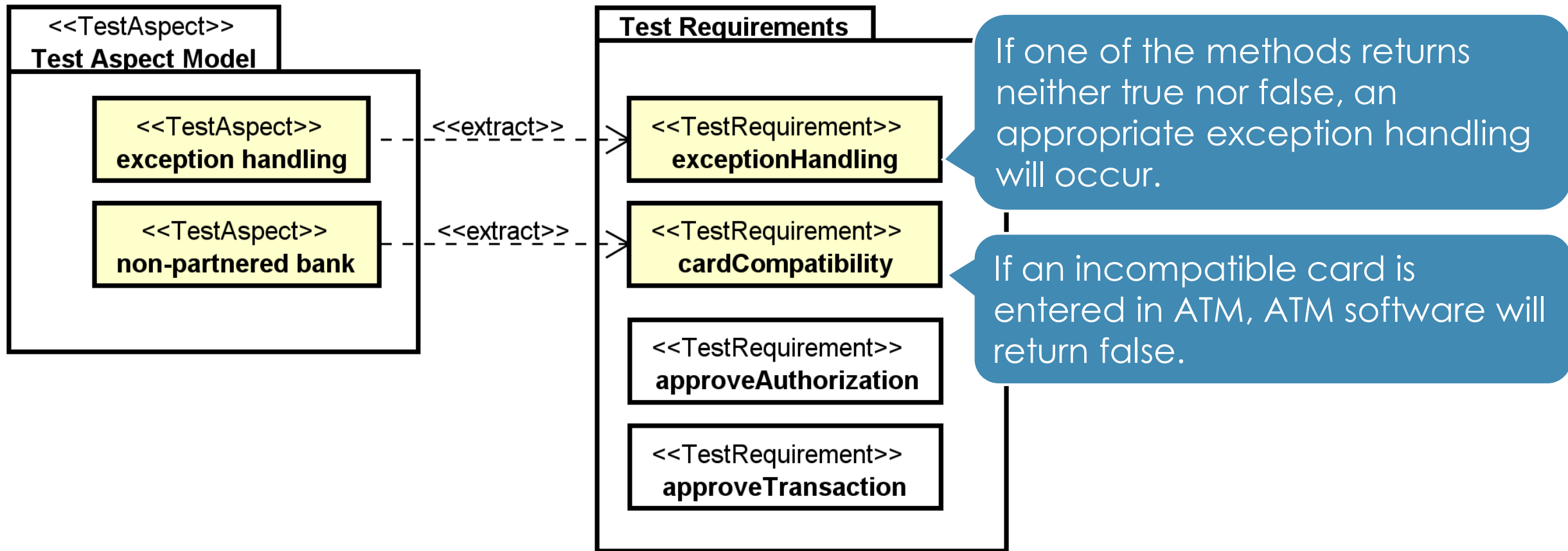
Use case 1: Extracting new test requirements

In this way, test engineers can enhance the test context.
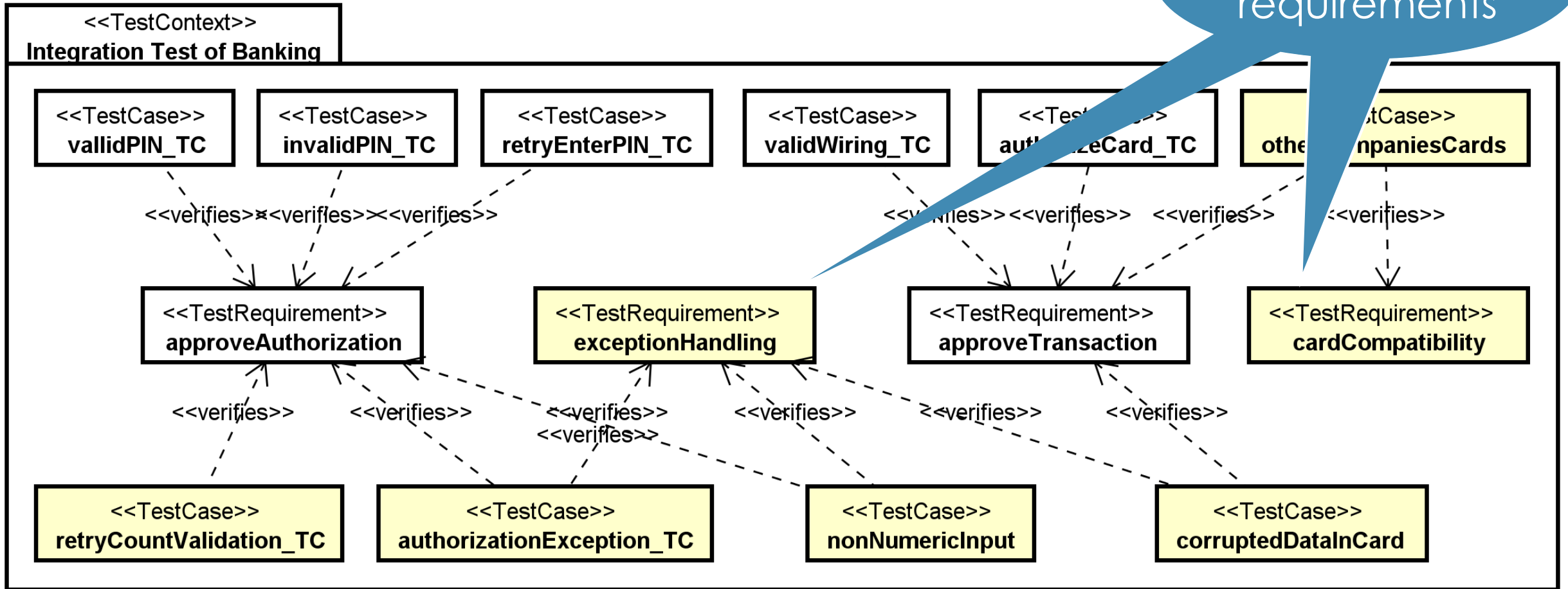
Use case 2: Extracting new test cases

Looking at test aspects in detail, we sometimes find the aspects that show factors and levels of parameters. Test engineers can use such test aspects when designing test cases.

			1	2	3	4	5	6	7	8
Conditions	Type of Bank	partner bank	Y	Y	Y	Y				
		non-partnered bank					Y	Y	Y	Y
	Card Status	damaged		Y				Y		
		expired			Y				Y	
		non-supported				Y				Y
Expected Actions	Card Validations	valid	X							
		invalid		X	X	X	X	X	X	X
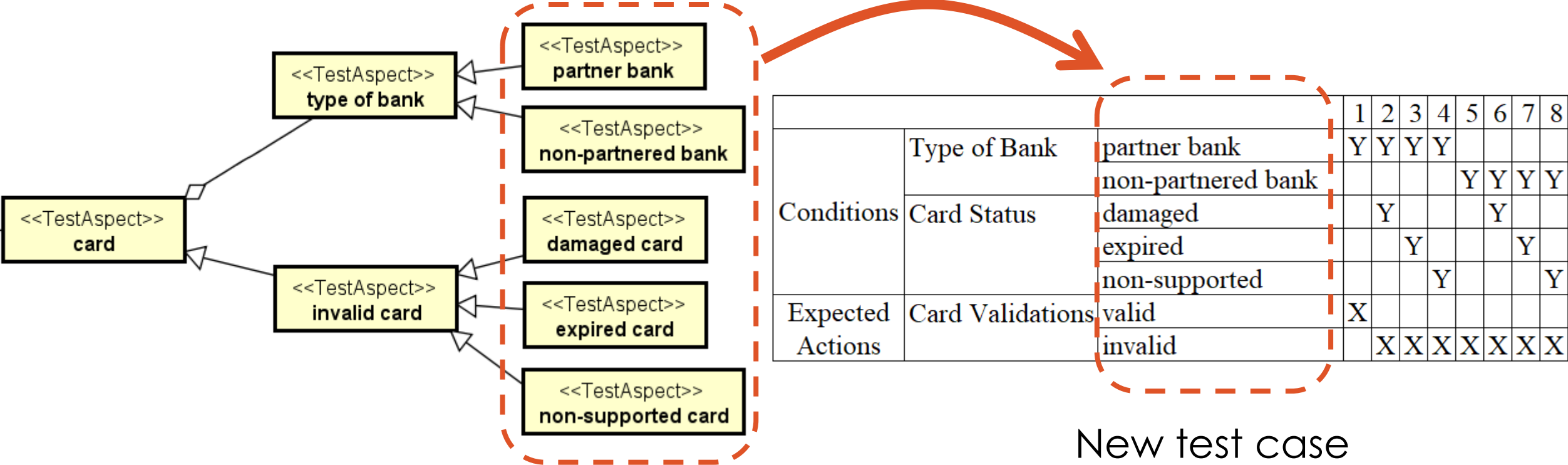
Test aspects about *card*

New test case

31

Use case 3: Complementing the development model

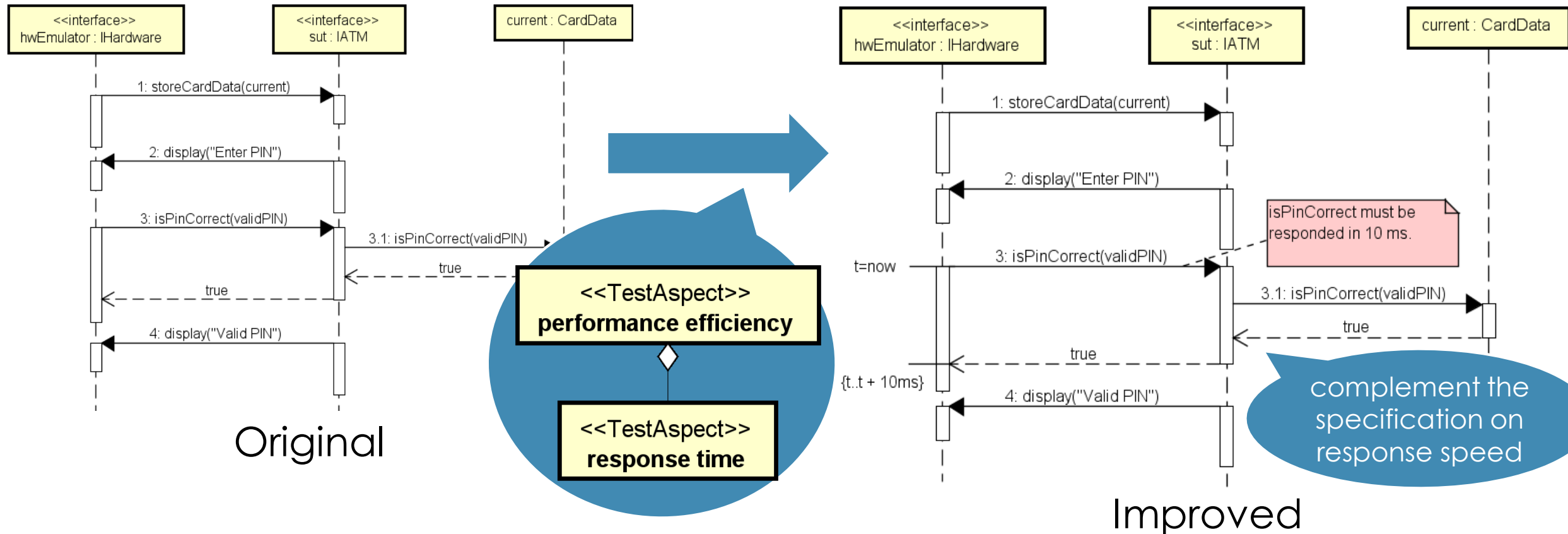The consideration of test aspects complements the shortcoming of the development model.
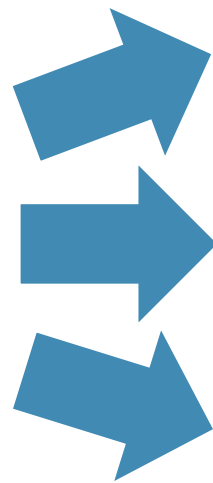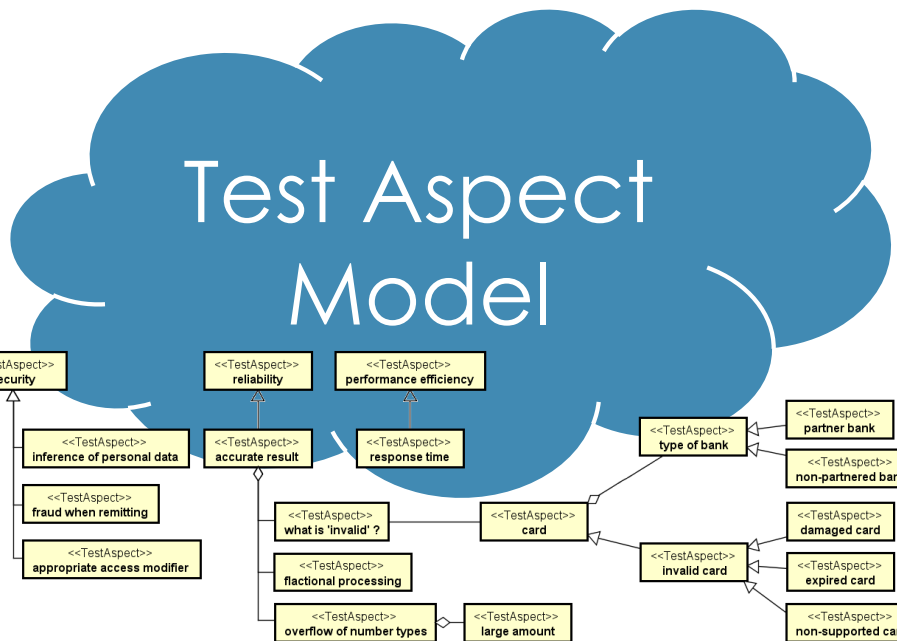

Original

Improved

5. Conclusion

Utilizing tacit knowledge with UTP

Test engineers' concerns on a test object is important to prevent from insufficient testing.
We gave the methodology of explicating these concerns with UTP, that is the *test aspect* and the *test aspect model*.
The effects of considering test aspects will be as follows:

Test Aspect Model

Extracting new test requirements and making test cases fulfilling

Sharing knowledge about test objects

Complementing development models

<<TestAspect>>
security

<<TestAspect>>
reliability

<<TestAspect>>
performance efficiency

<<TestAspect>>
inference of personal data

<<TestAspect>>
accurate result

<<TestAspect>>
response time

<<TestAspect>>
type of bank

<<TestAspect>>
partner bank

<<TestAspect>>
fraud when remitting

<<TestAspect>>
non-partnered bank

<<TestAspect>>
appropriate access modifier

<<TestAspect>>
what is 'invalid' ?

<<TestAspect>>
card

<<TestAspect>>
damaged card

<<TestAspect>>
flactional processing

<<TestAspect>>
invalid card

<<TestAspect>>
expired card

<<TestAspect>>
overflow of number types

<<TestAspect>>
large amount

<<TestAspect>>
non-supported card

Future tasks

- Examination of relationships between other methodologies.
  - We will examine the relationship between the test aspect model and other methodologies of the test development.
- How to create a better test aspect model.
  - Where, we think "better" means as follows:
    - the model leads to more fulfilling test cases
    - the model can be used in a more general purpose

Special Thanks

Satomi-Juku

What is Satomi-Juku ?

"Satomi-Juku" is a research group of test architecture in NPO ASTER.
It is a place to disclose more advanced test development methodologies
through debate, and share them.