Proposal for Enhancing UTP2 with Test Aspects

Makoto Nakakuki
TEF-Do

Shizuka Ban
NPO ASTER

Noriyuki Mizuno
TEF-Do

Koki Abe
Tosei Systems Co., Ltd

Hiroki Iseri
NPO ASTER

Tomohiro Odan
Mamezou Co., Ltd.

Hitoshi Ando
TEF-Do

Kumiko Iseri
TIS Inc.

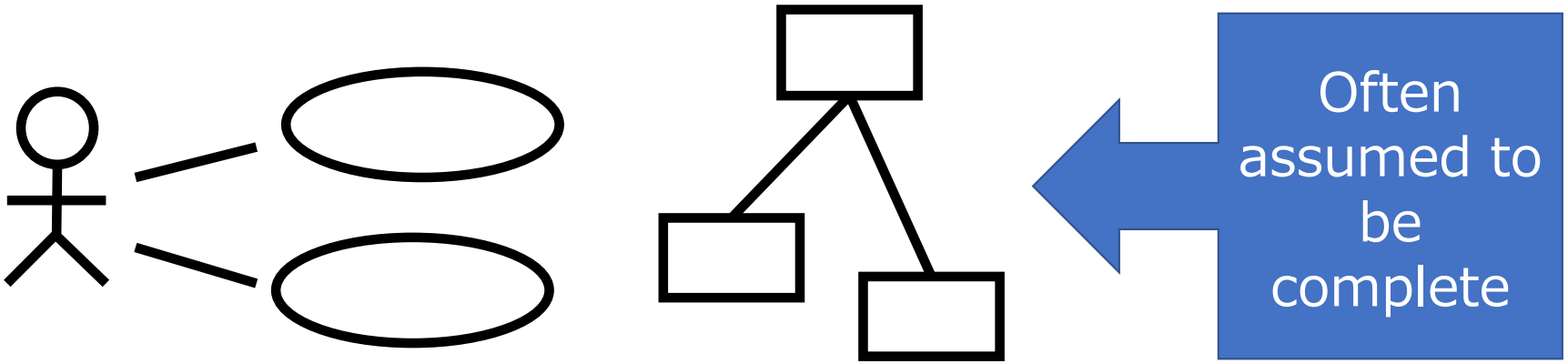
Akiharu Satoh
Hitachi Information Academy Inc.

Contents

INTRODUCTION

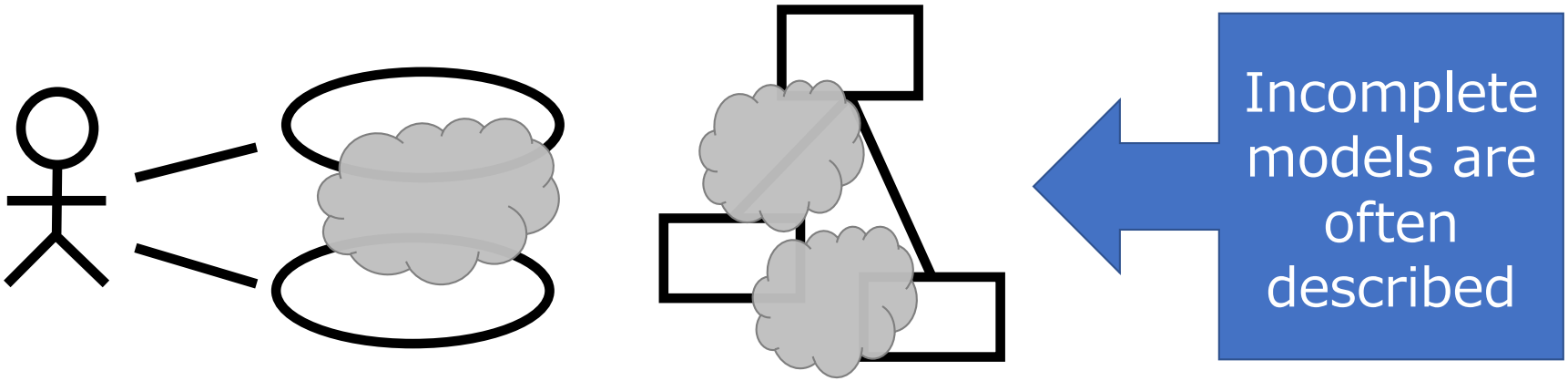
Introduction

Various models are developed prior to software testing.



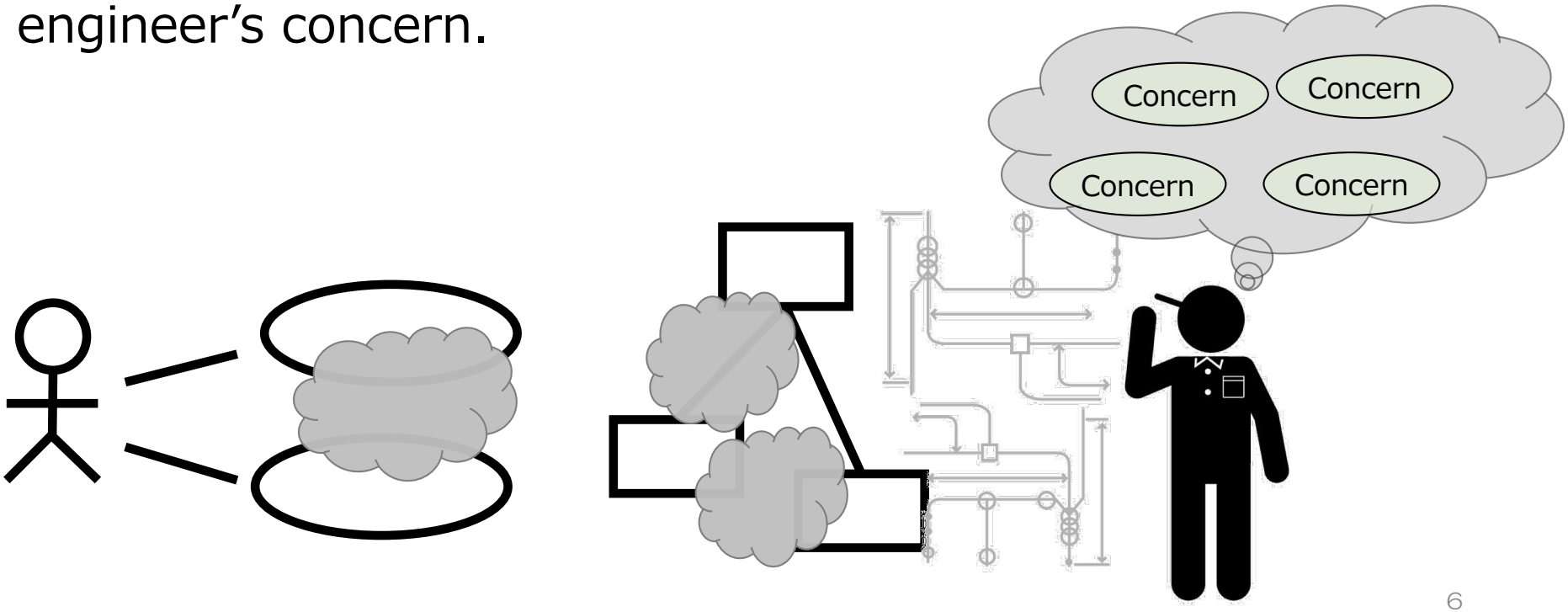
Introduction

These incomplete models that have omitted information are usually revised by modeling for software testing.



Introduction

Other aspects of development model are created from test engineer’s concern.



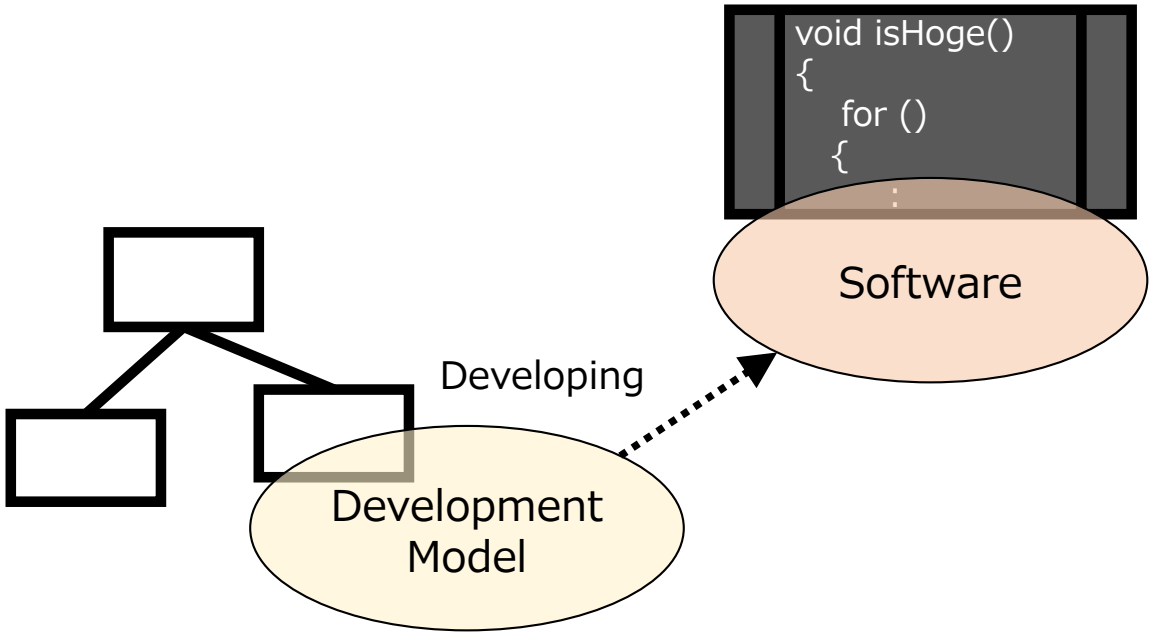
PROPOSAL FOR TEST ASPECT MODEL

Common Problem

Development Models are usually Incomplete...

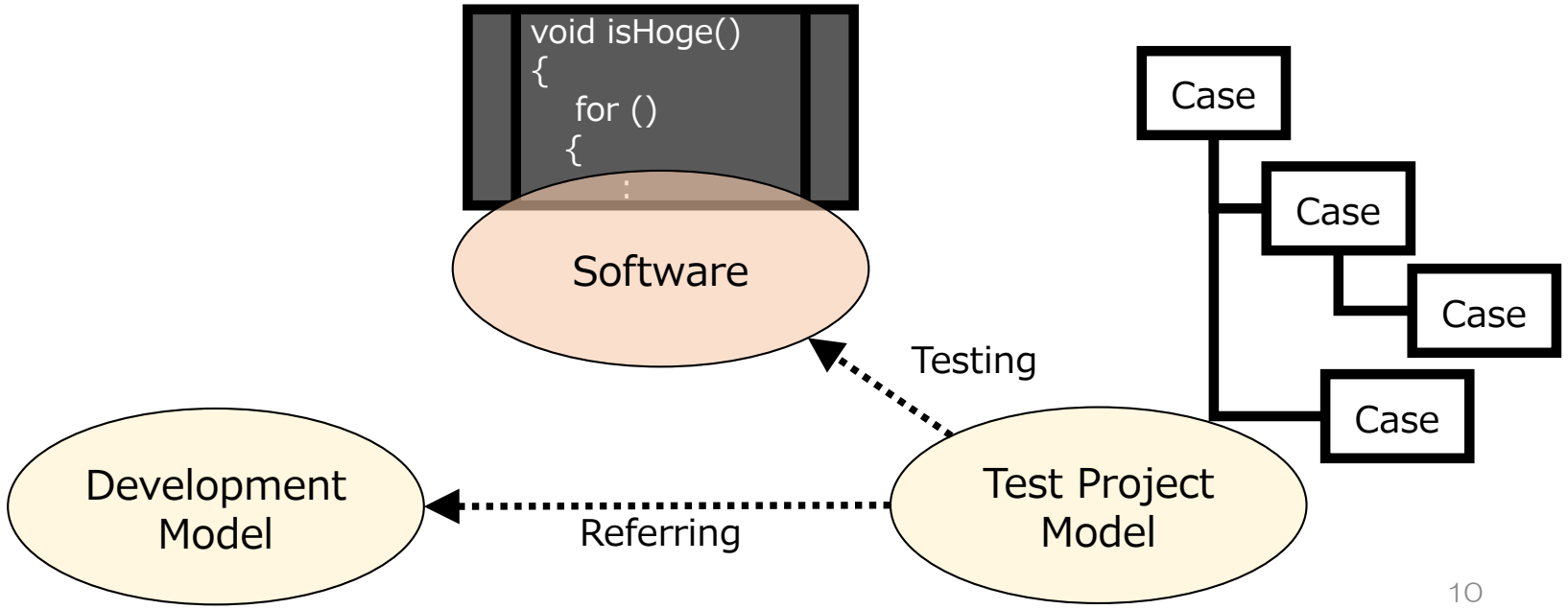
To Complement a Development Model

The development model can be created using modeling notations such as UML and SysML.



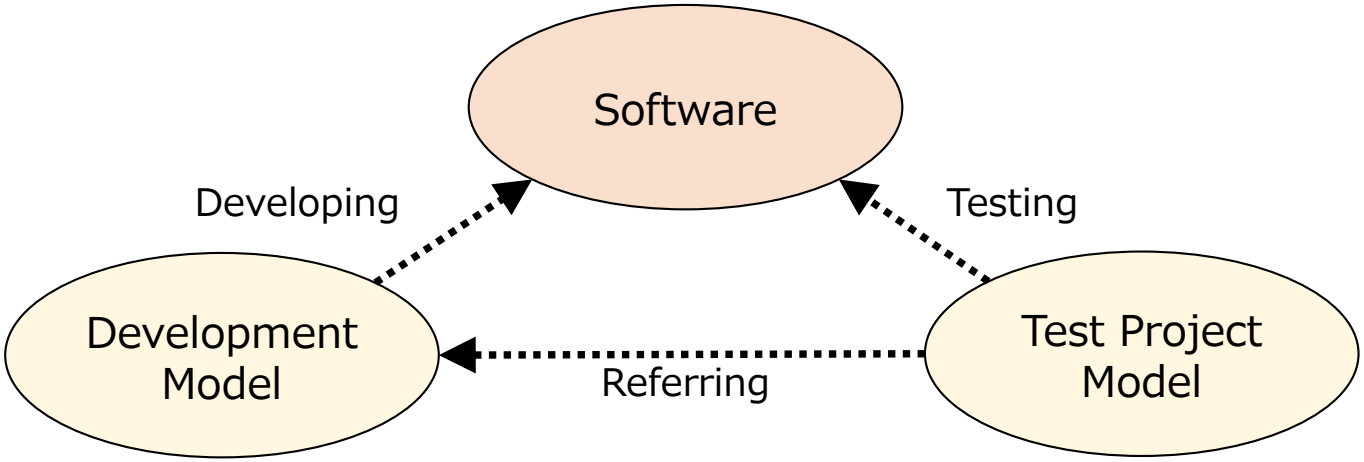
To Complement a Development Model

The test design process creates a test project model.
“Test project model” has test cases.

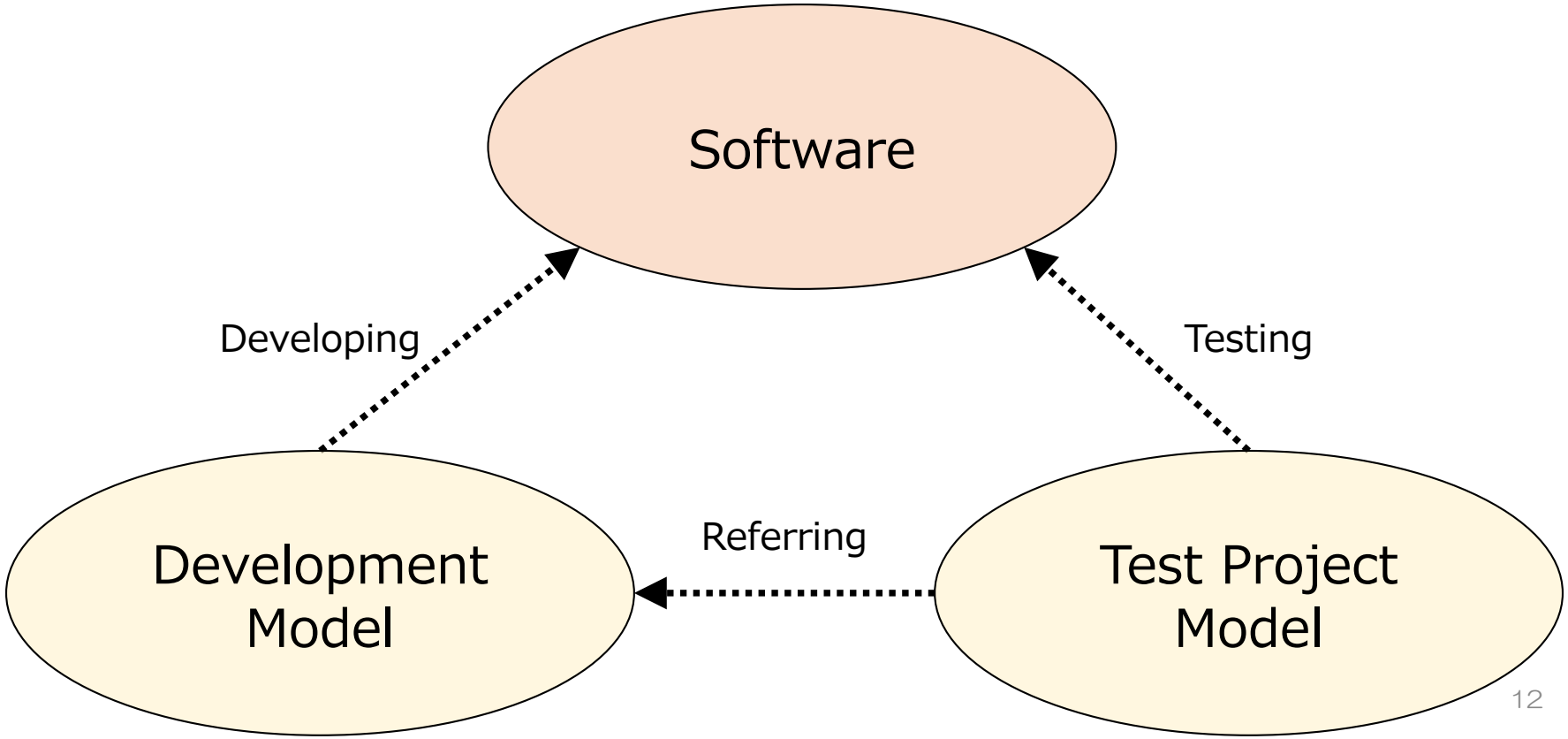


To Complement a Development Model

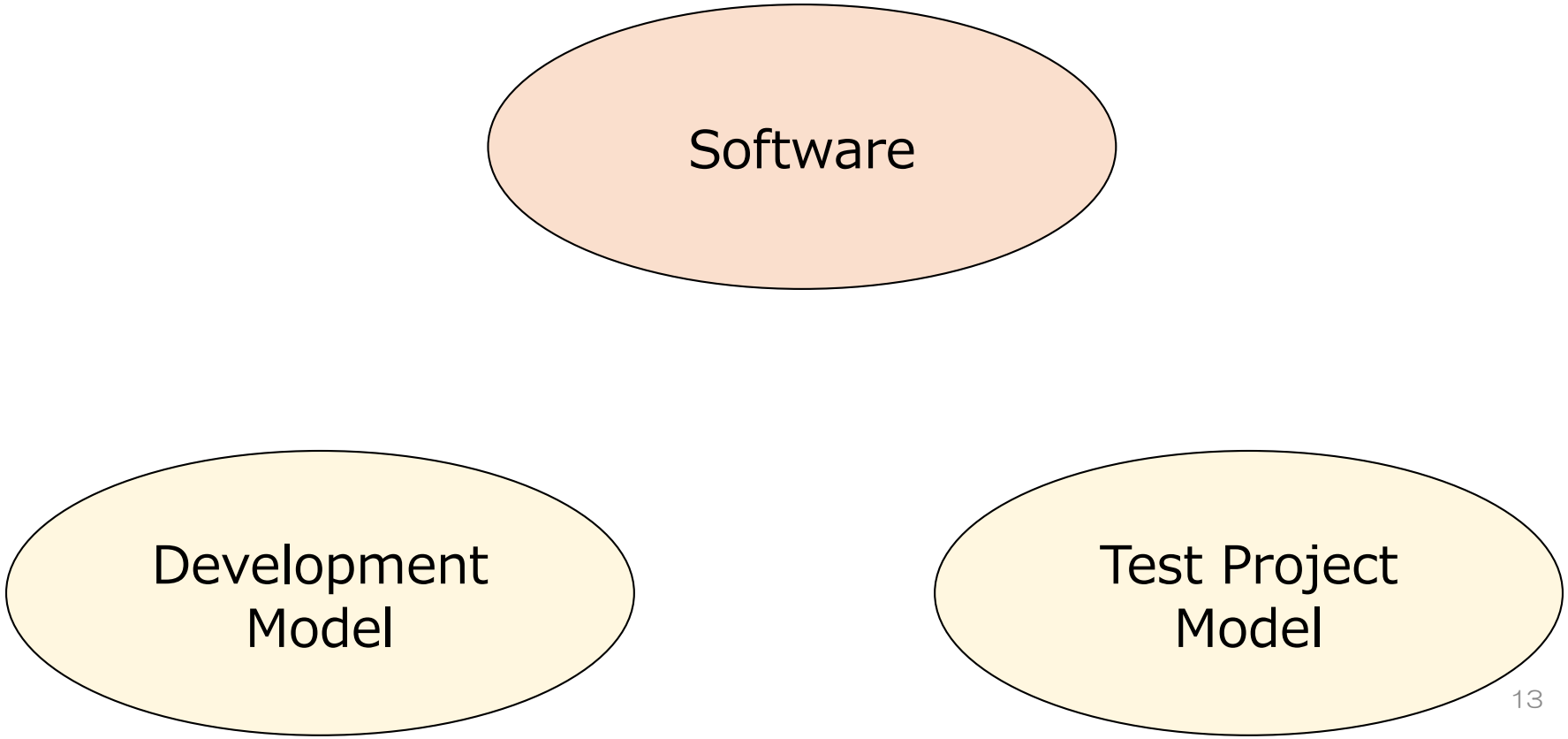
Relationship among software, development model and test project model.



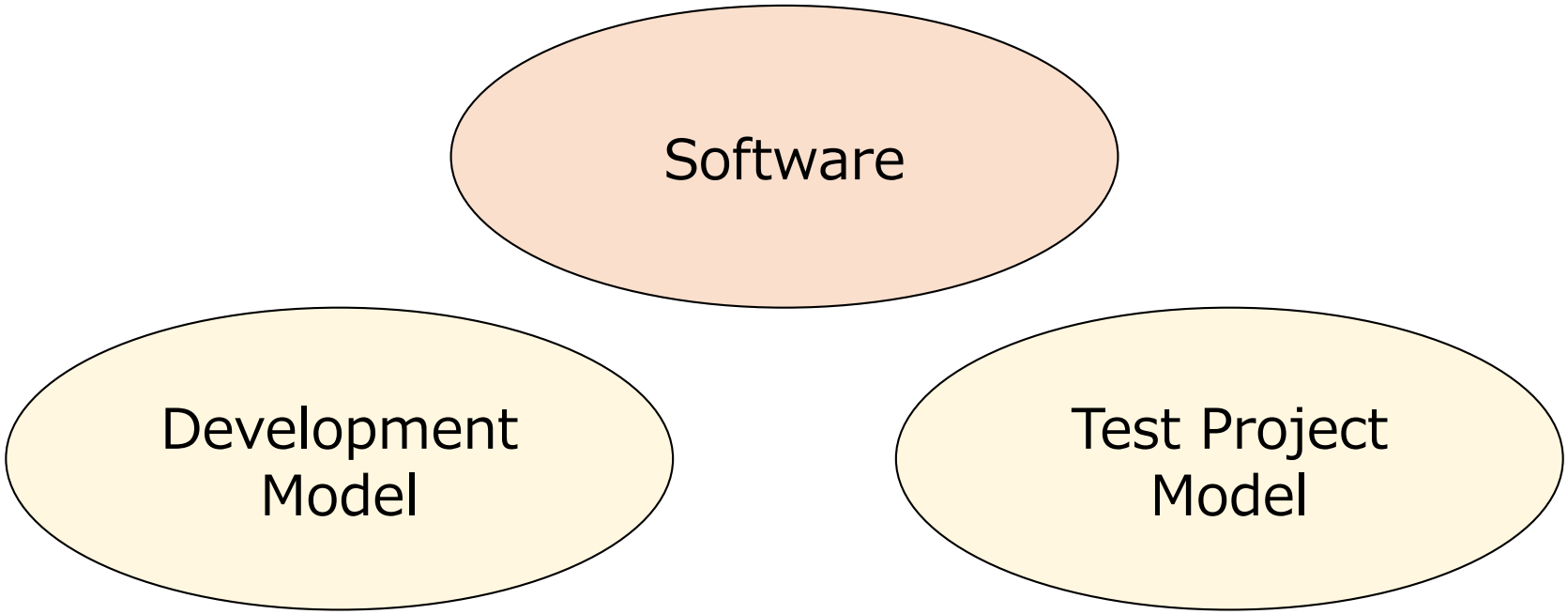
Case 1:
Development model is perfect



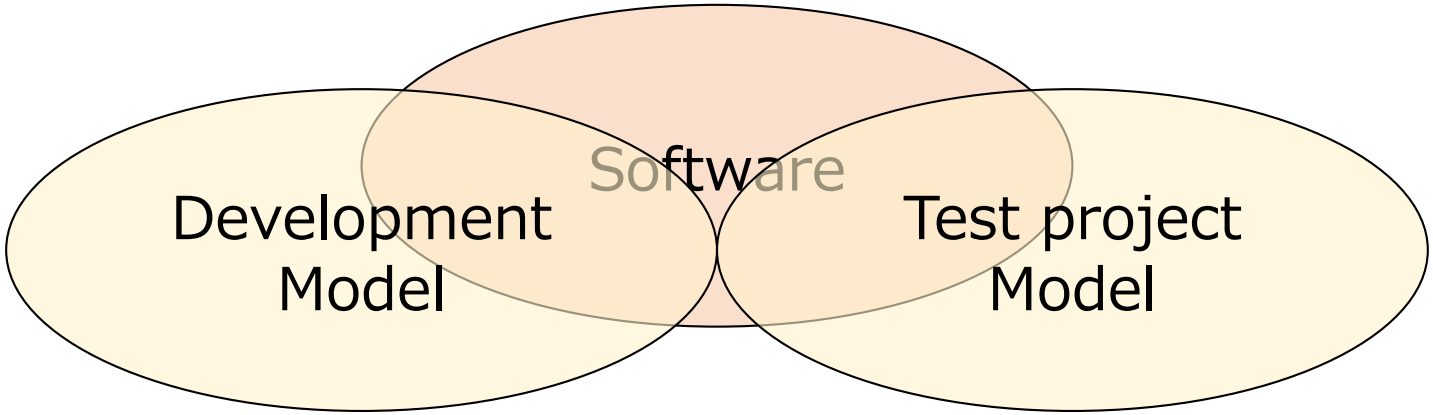
Case 1:
Development model is perfect



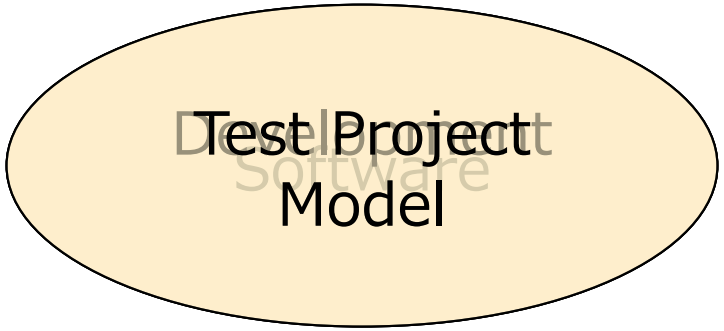
Case 1:
Development model is perfect



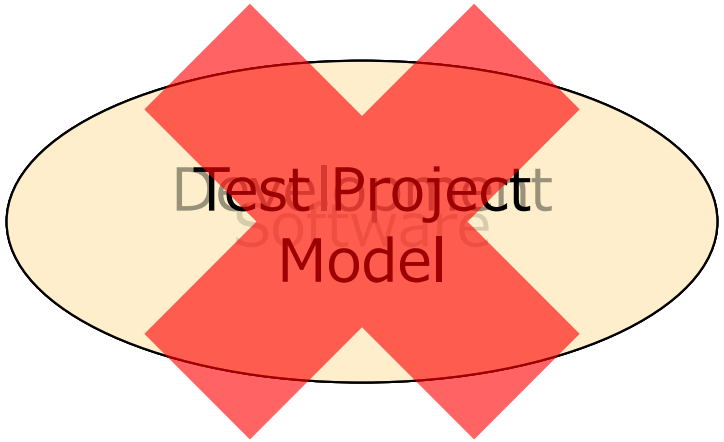
Case 1:
Development model is perfect



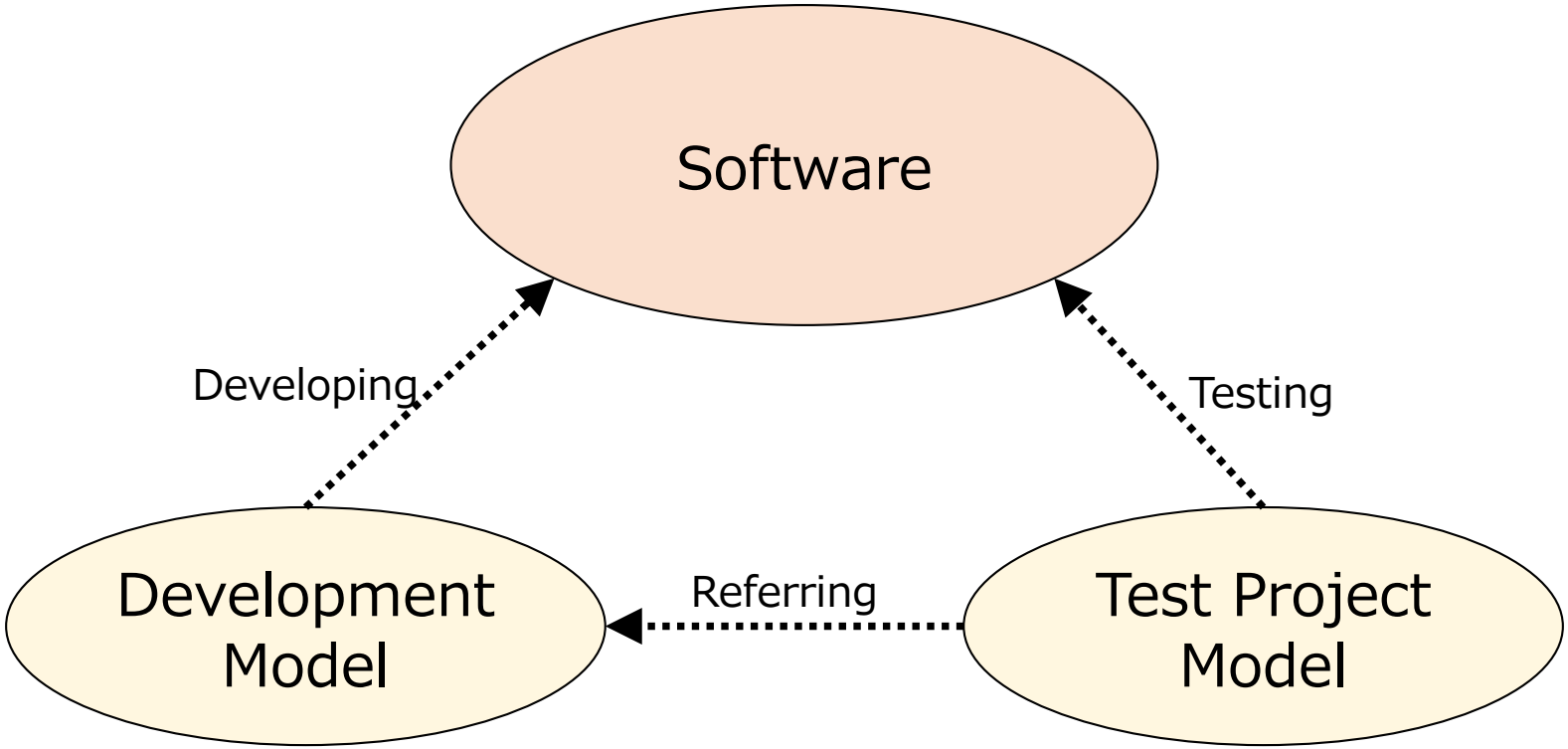
Case 1:
Development model is perfect



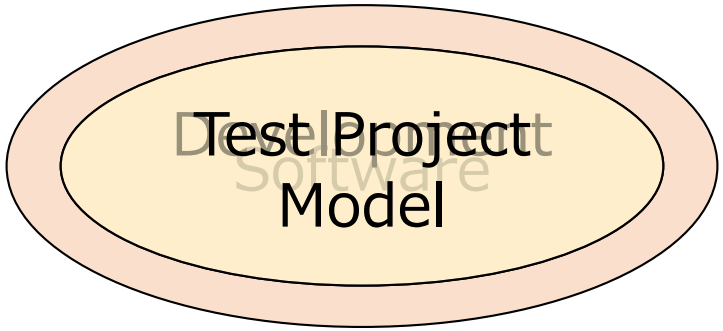
Development Model is Incomplete



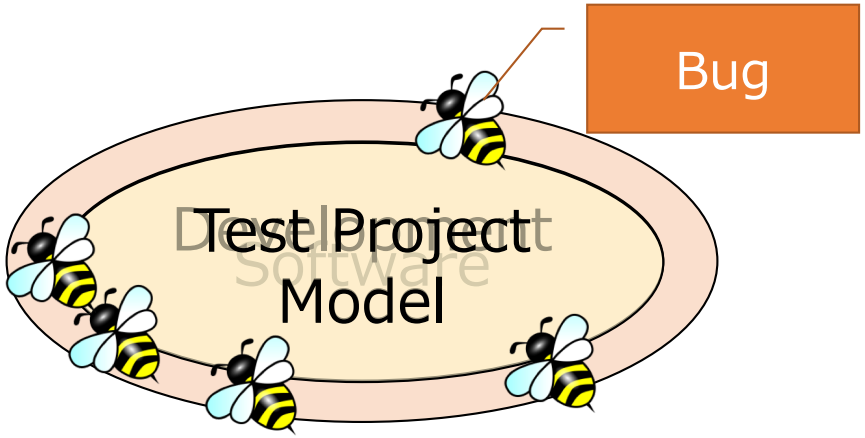
Case 2:
Development Model is Incomplete



Case 2:
Development Model is Incomplete

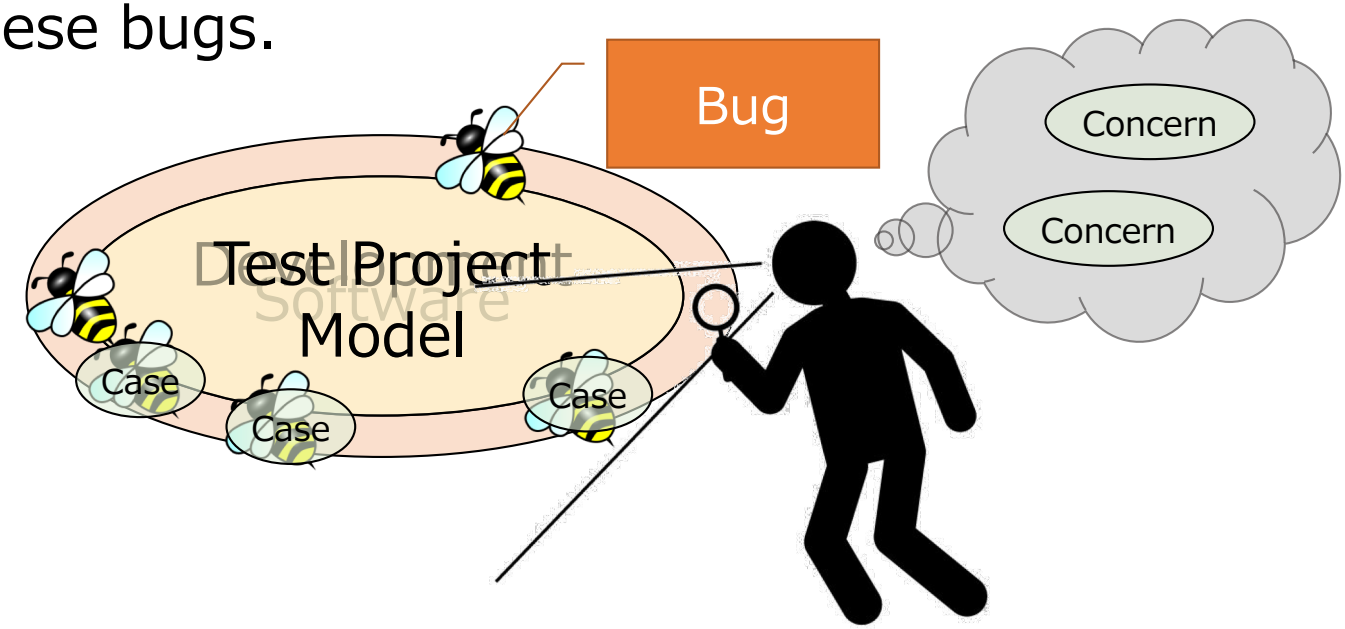


Case 2:
Development Model is Incomplete



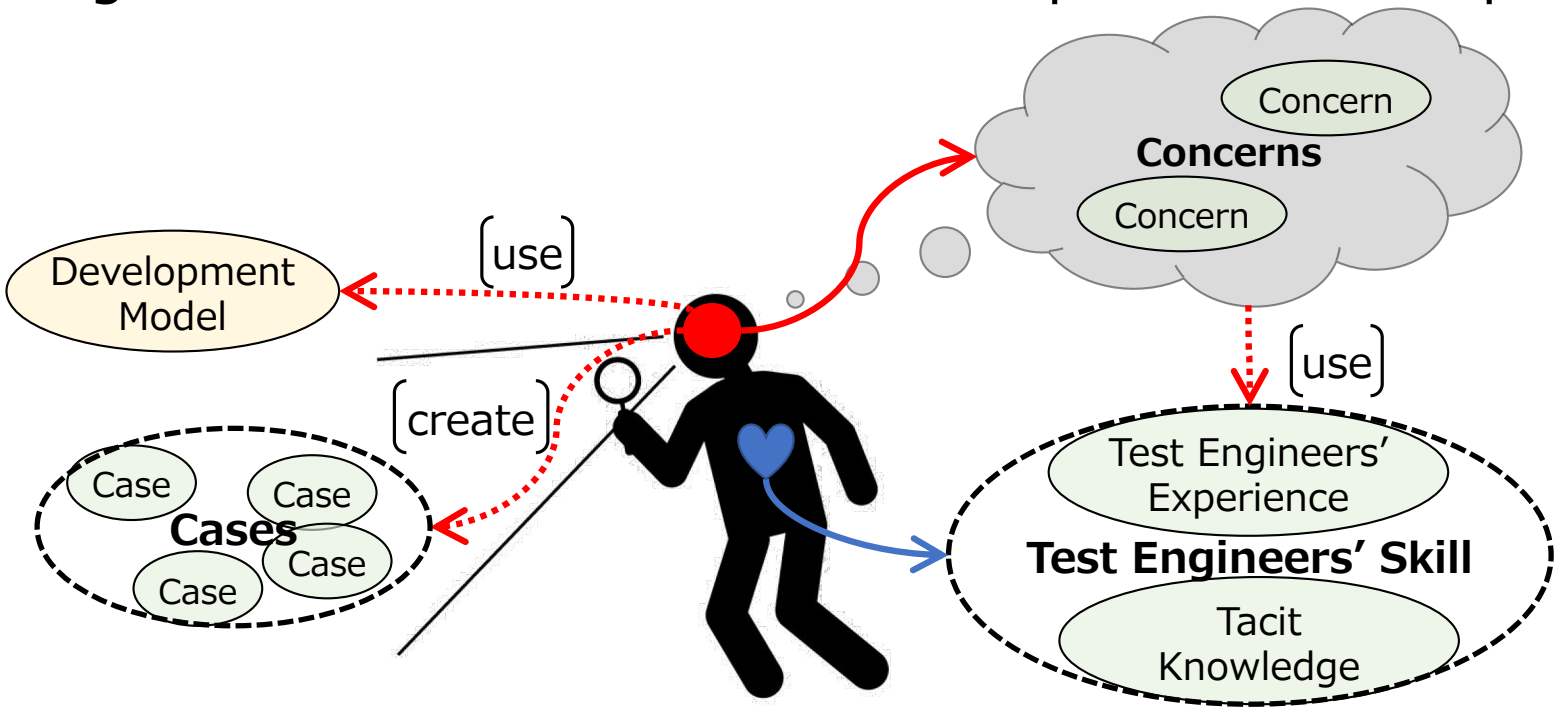
Case 2:
Development Model is Incomplete

Test cases designed based on test engineer's concern usually detect these bugs.

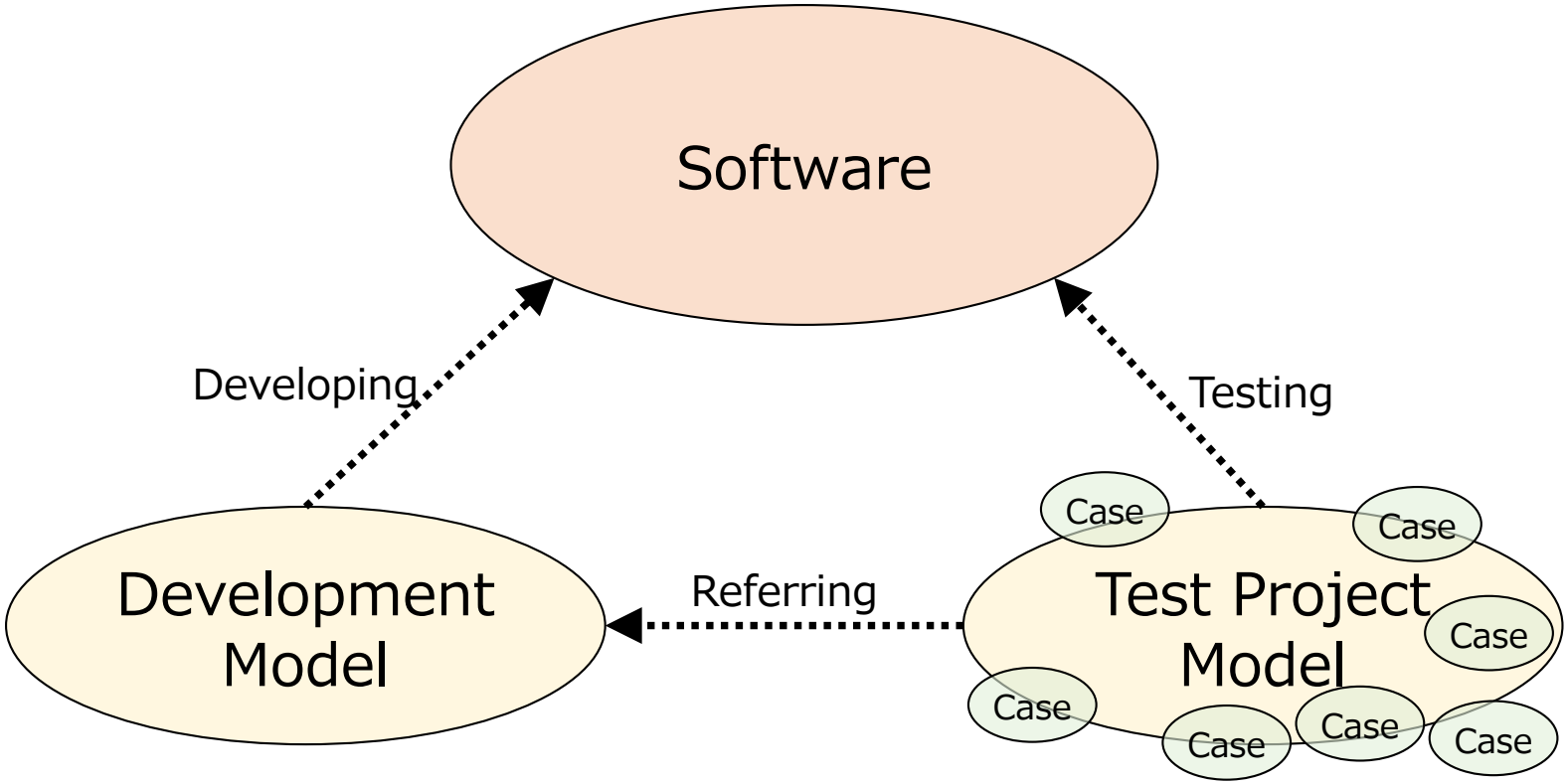


Test Depends on the Test Engineers' Skill

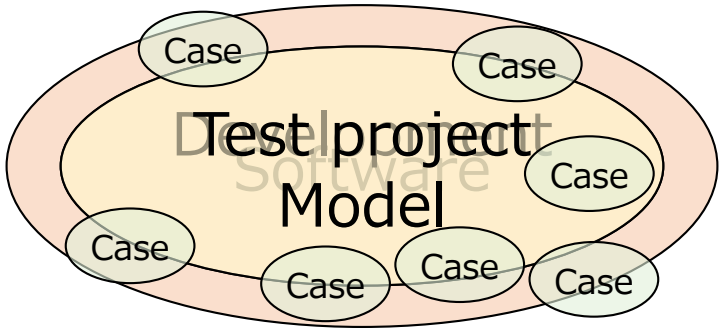
Test engineers sometimes rely on some information such as "bugs" detected in the software components in the past.



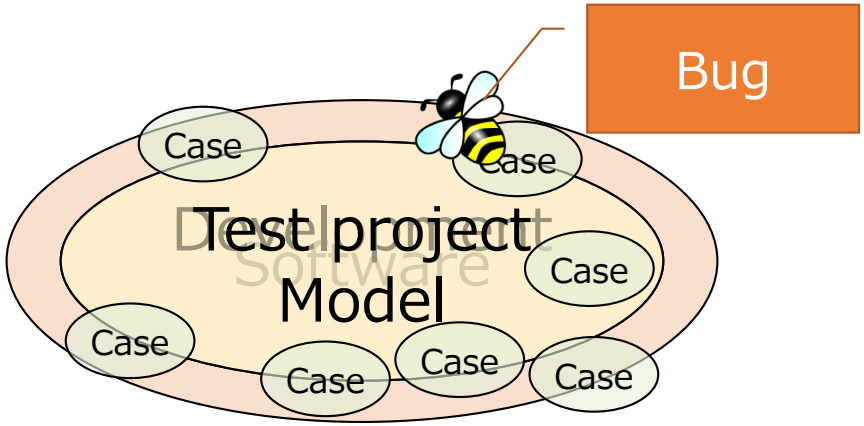
Case 3:
Test Depends on the Test Engineers' Skill



Case 3:
Test Depends on the Test Engineers' Skill



Case 3:
Test Depends on the Test Engineers' Skill

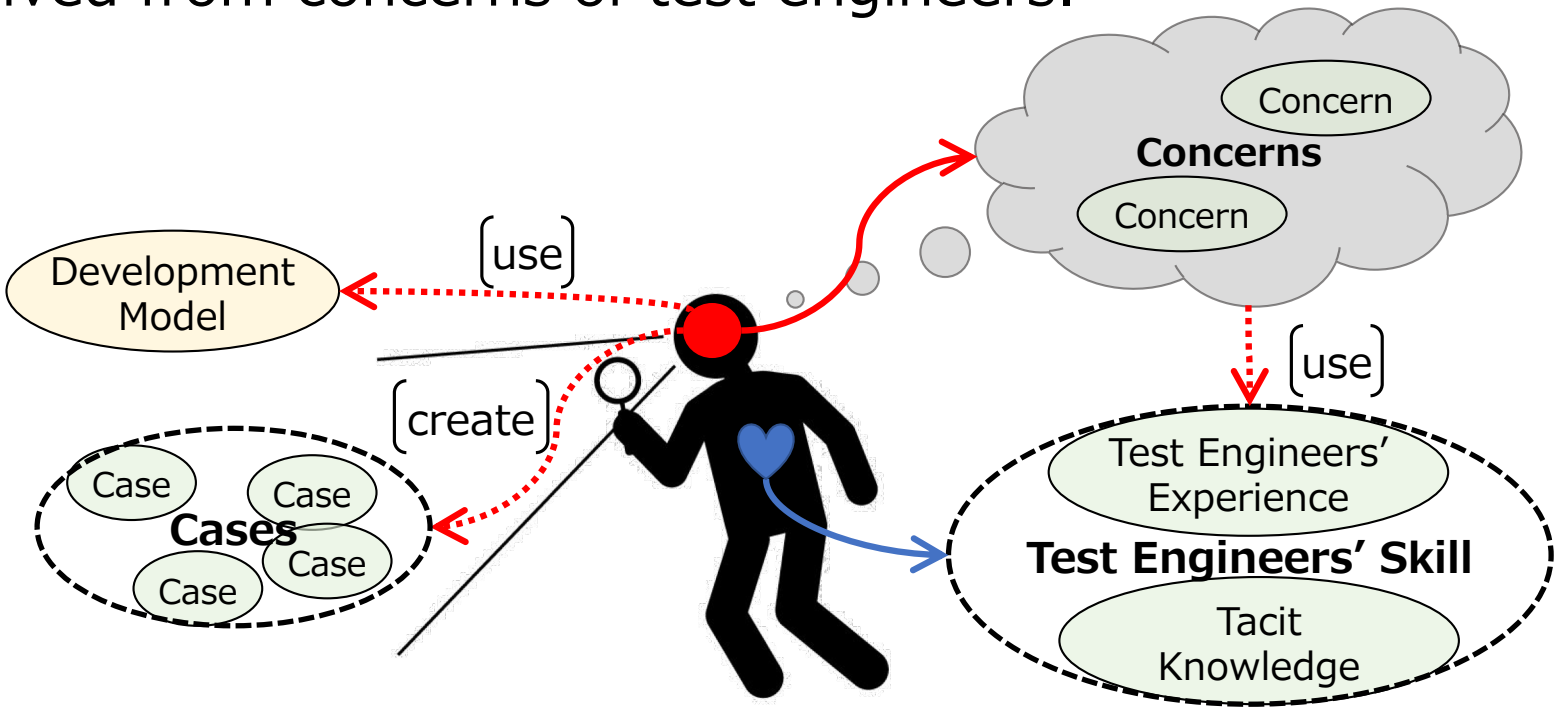


Use Aspect Model

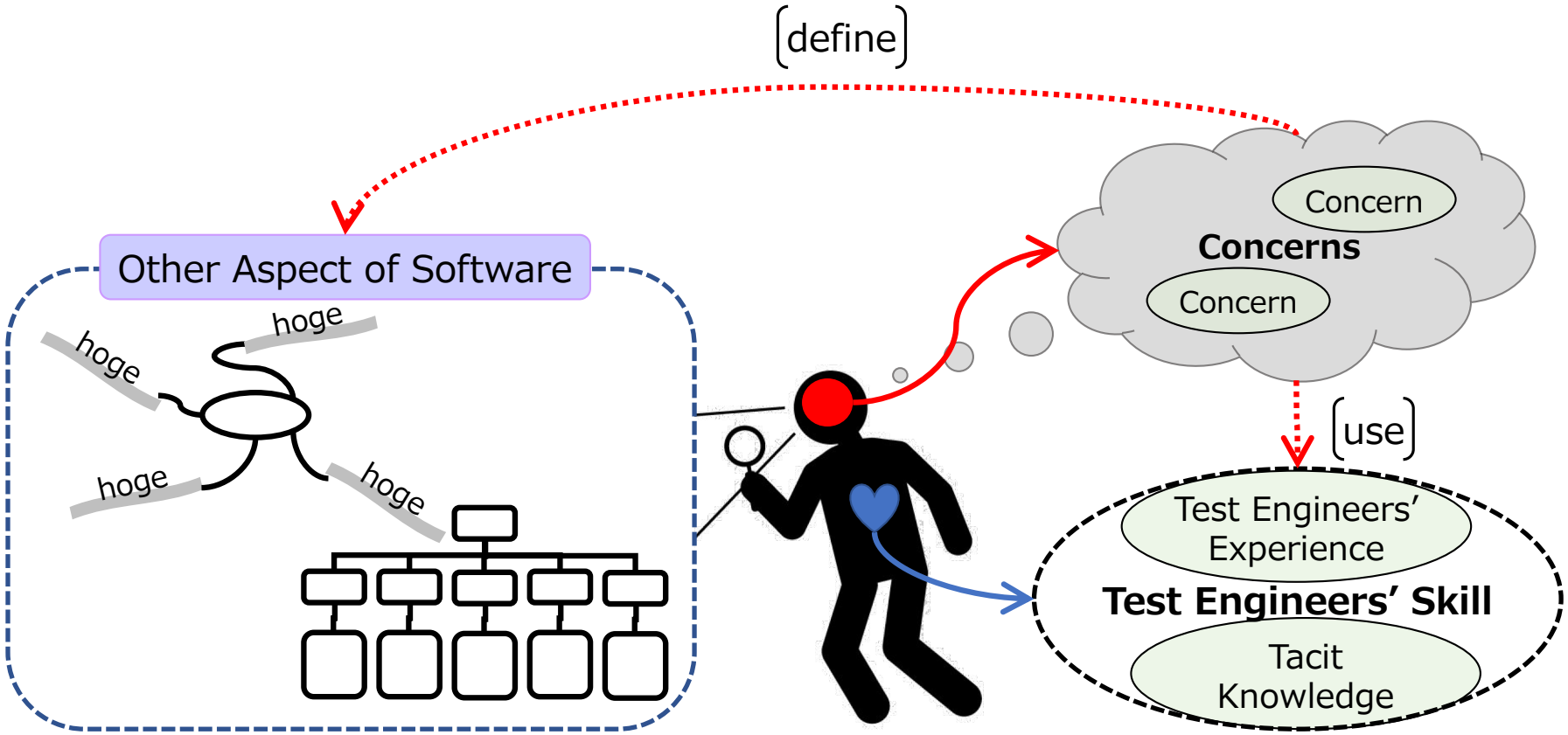
Concerns of test engineers are usually tacit knowledge...

Concerns Tend to be Tacit and Subjective

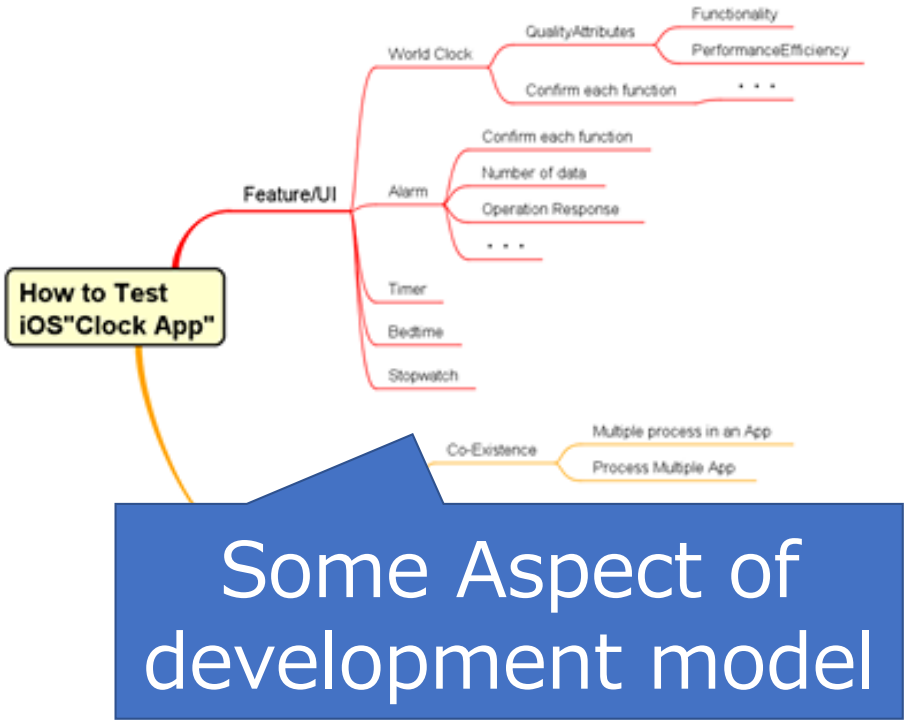
We consider that much of information on testing will be derived from concerns of test engineers.



Define Concerns as Aspects

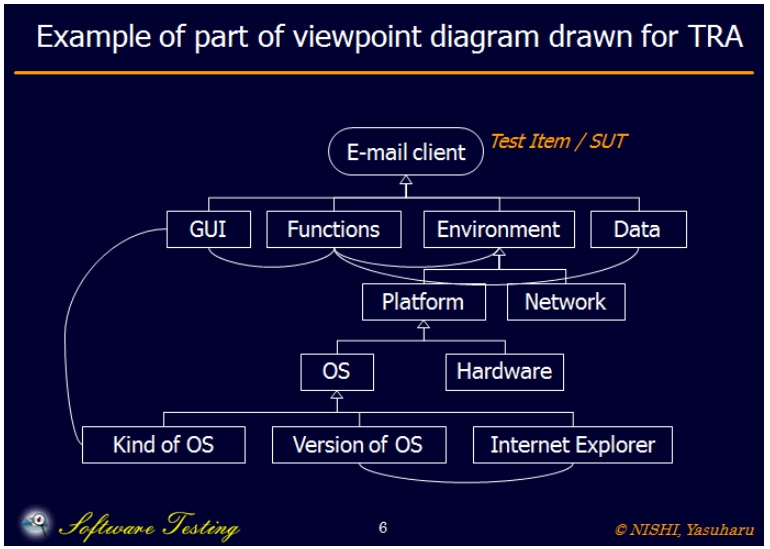


Example
Define Method



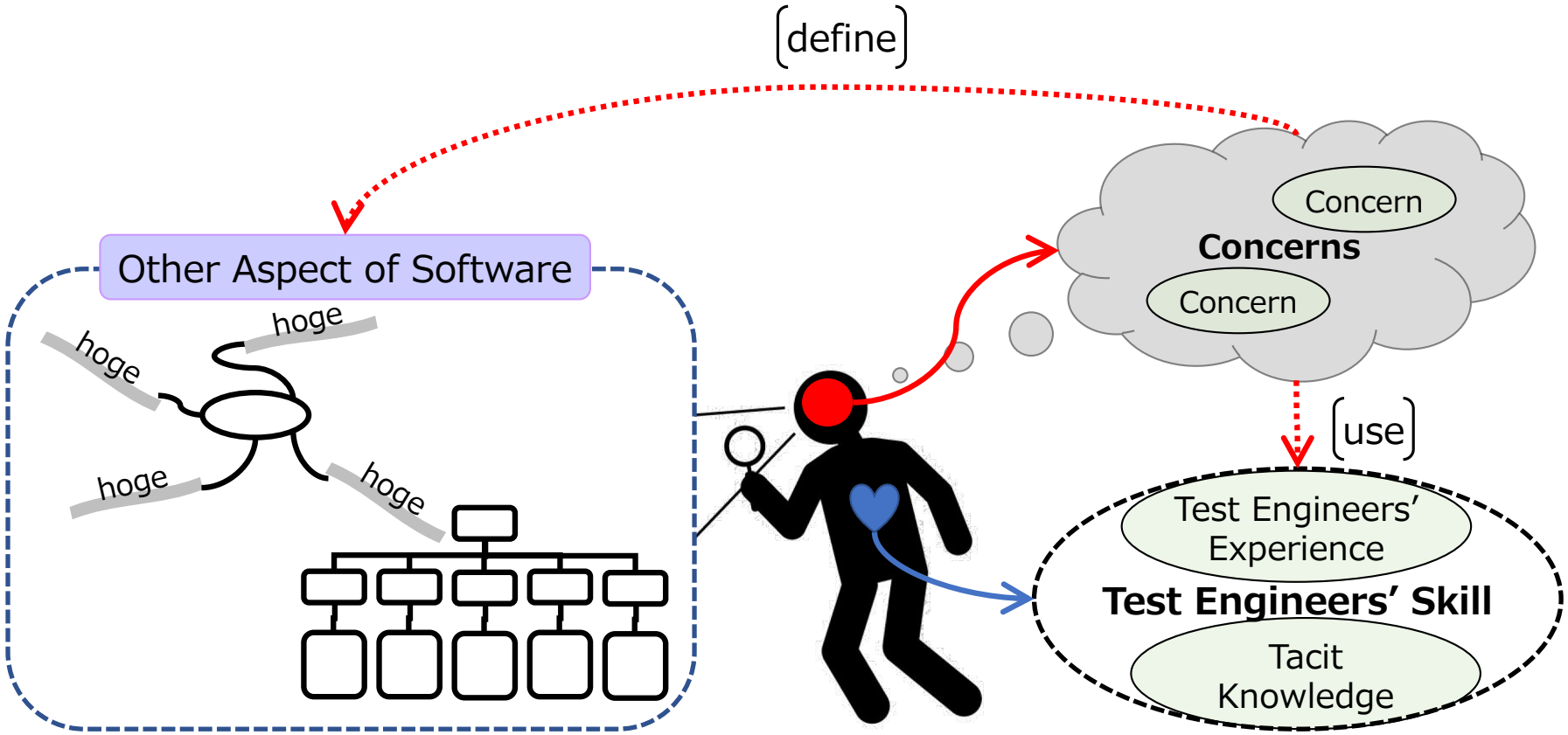
Some Aspect of development model

Viewpoint Diagram

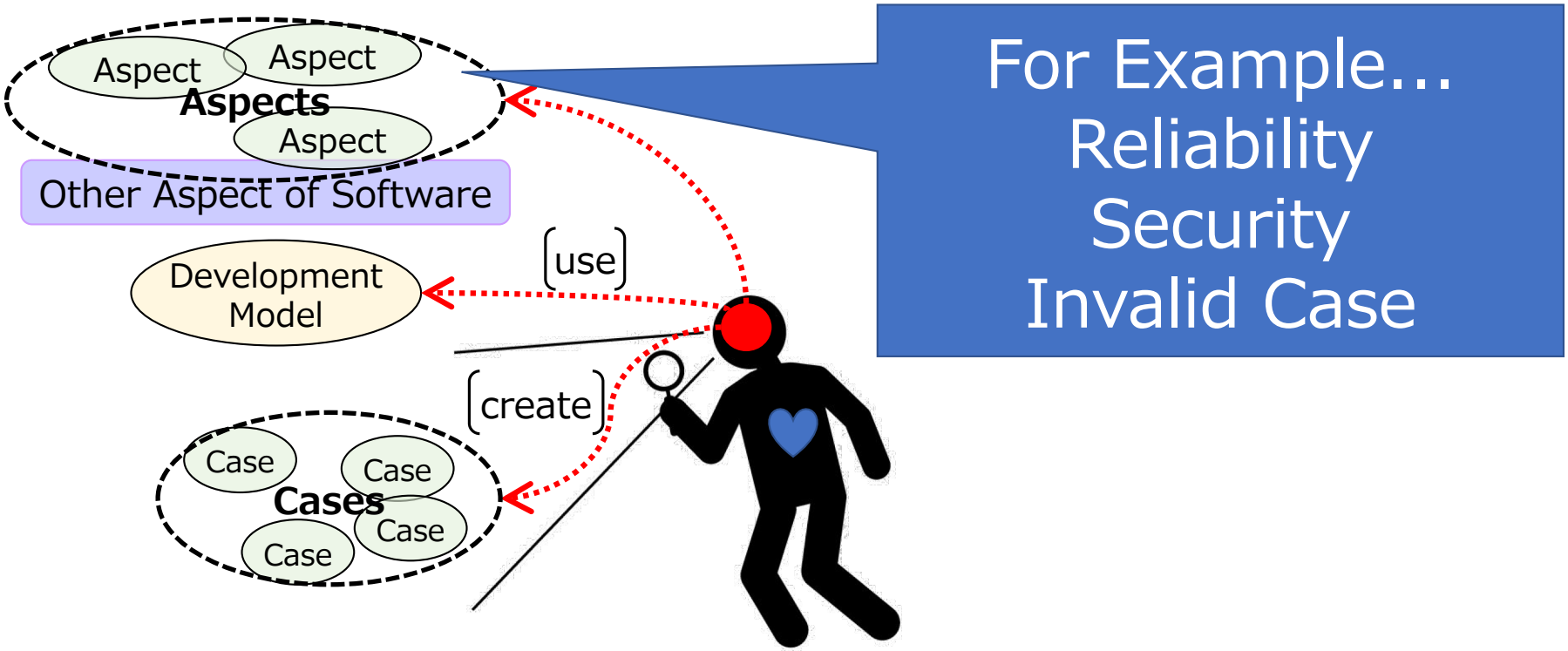


Y.Nishi, "Design principles in Test Suite Architecture," International Workshop on Software Test Architecture (InSTA 2015), Graz, Austria, April 2015.

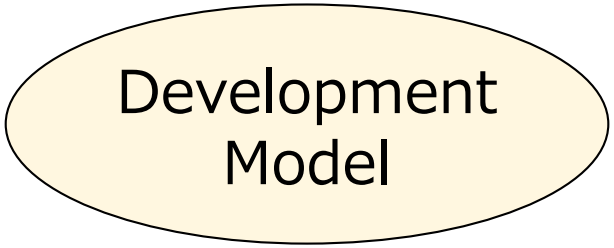
Define Concerns as Aspects



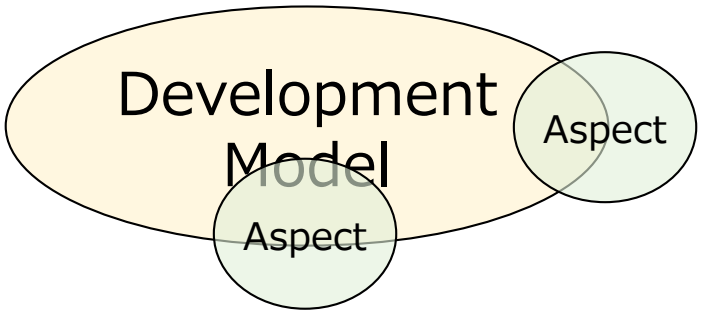
Use Test Aspect



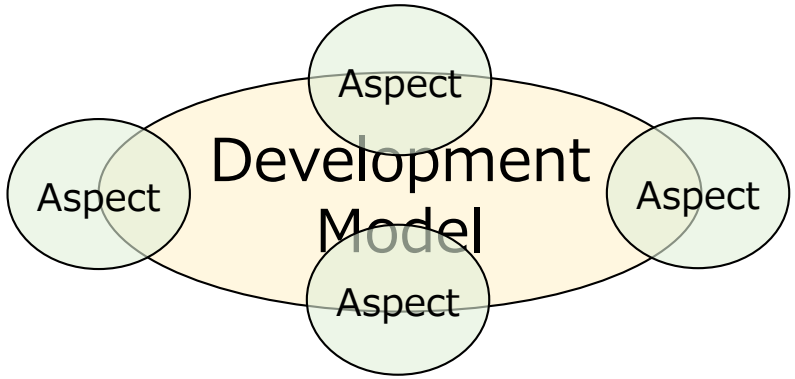
Proposal for Test Aspect Model



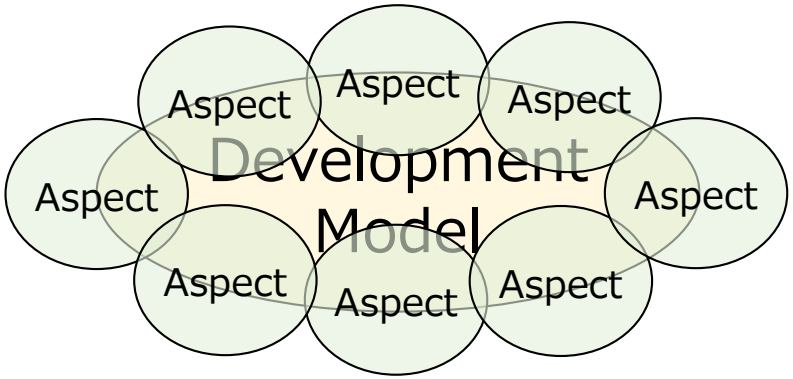
Proposal for Test Aspect Model



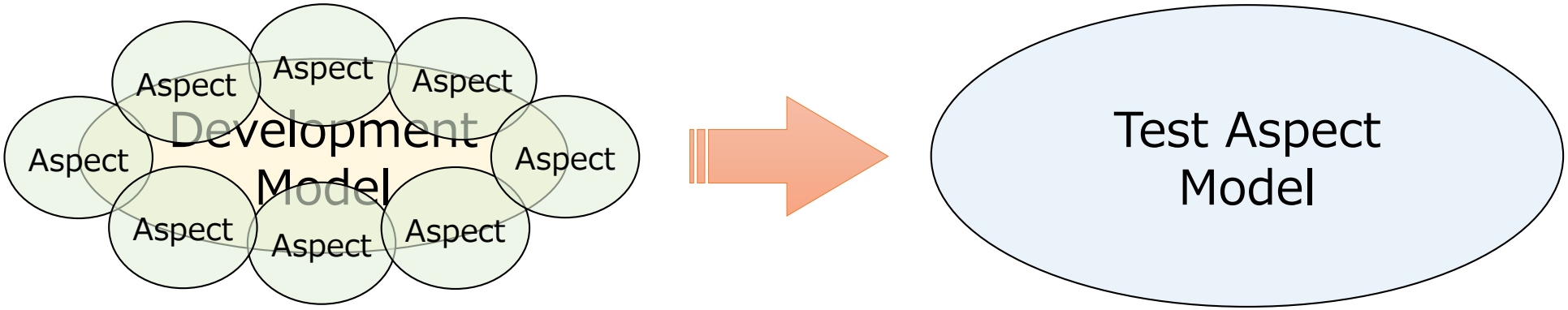
Proposal for Test Aspect Model



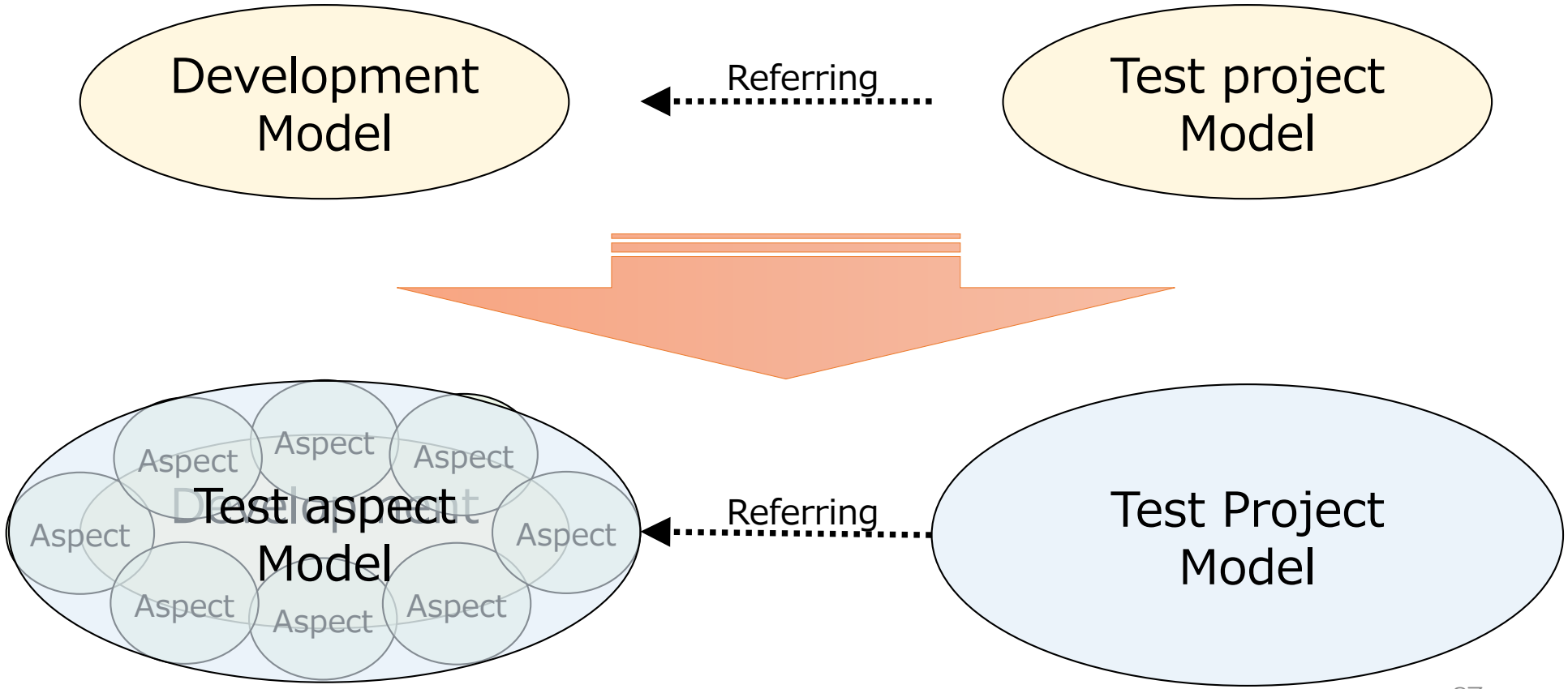
Proposal for Test Aspect Model



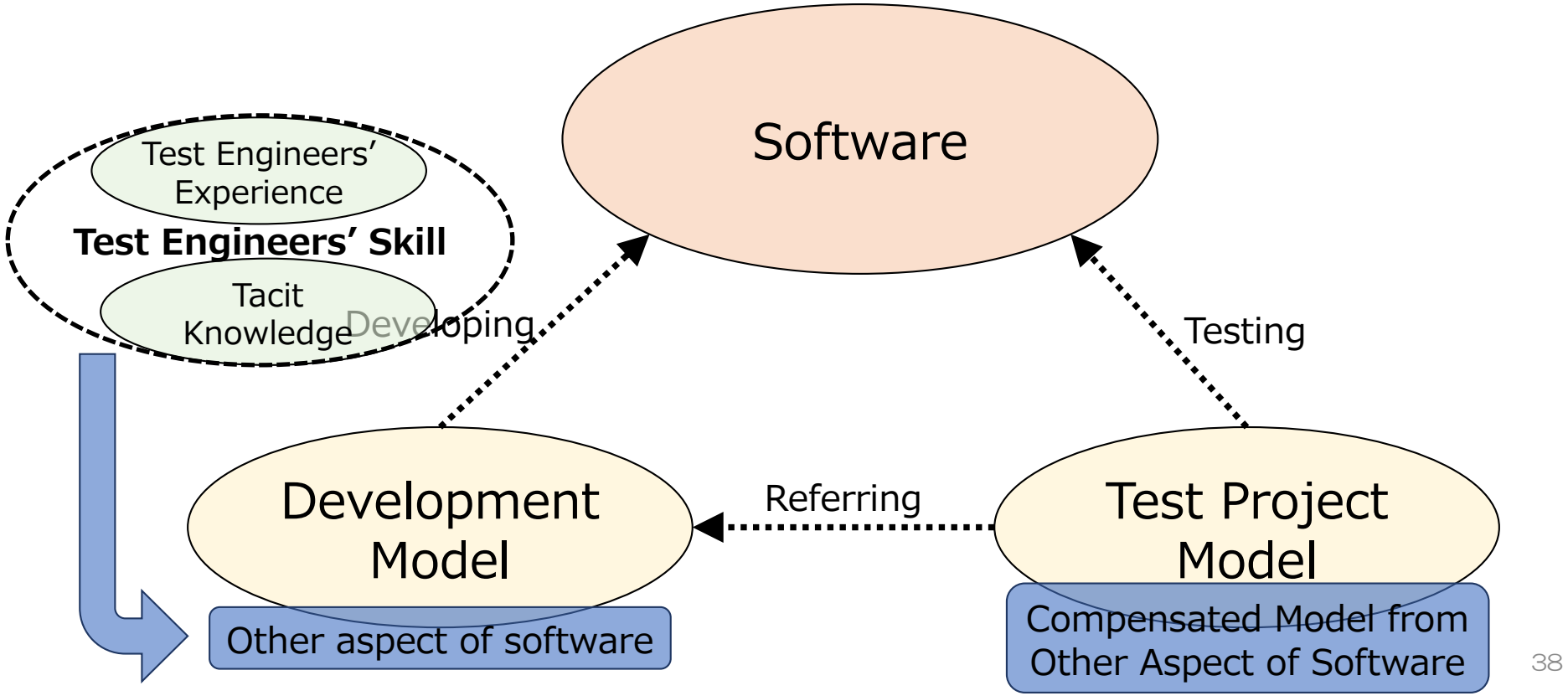
Proposal for Test Aspect Model



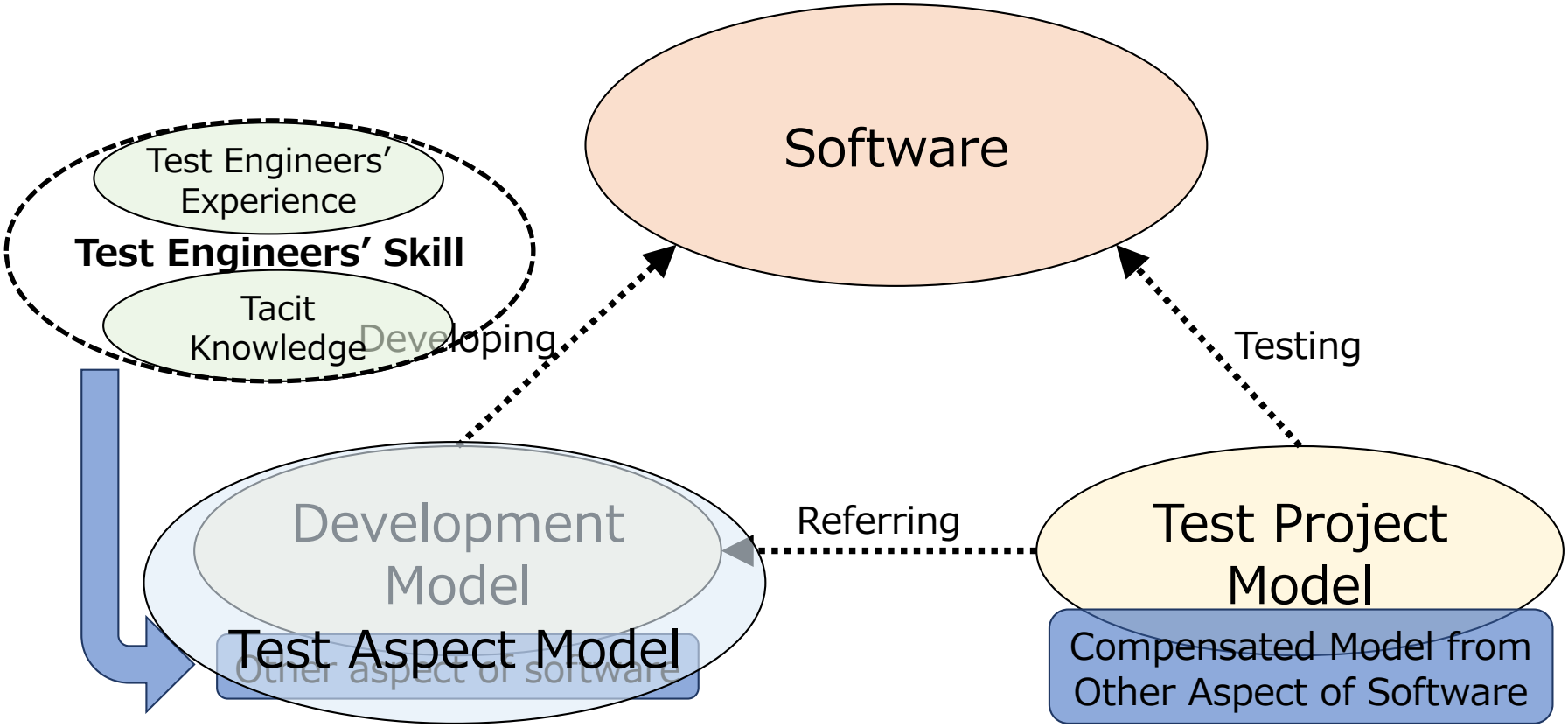
Proposal for Test Aspect Model



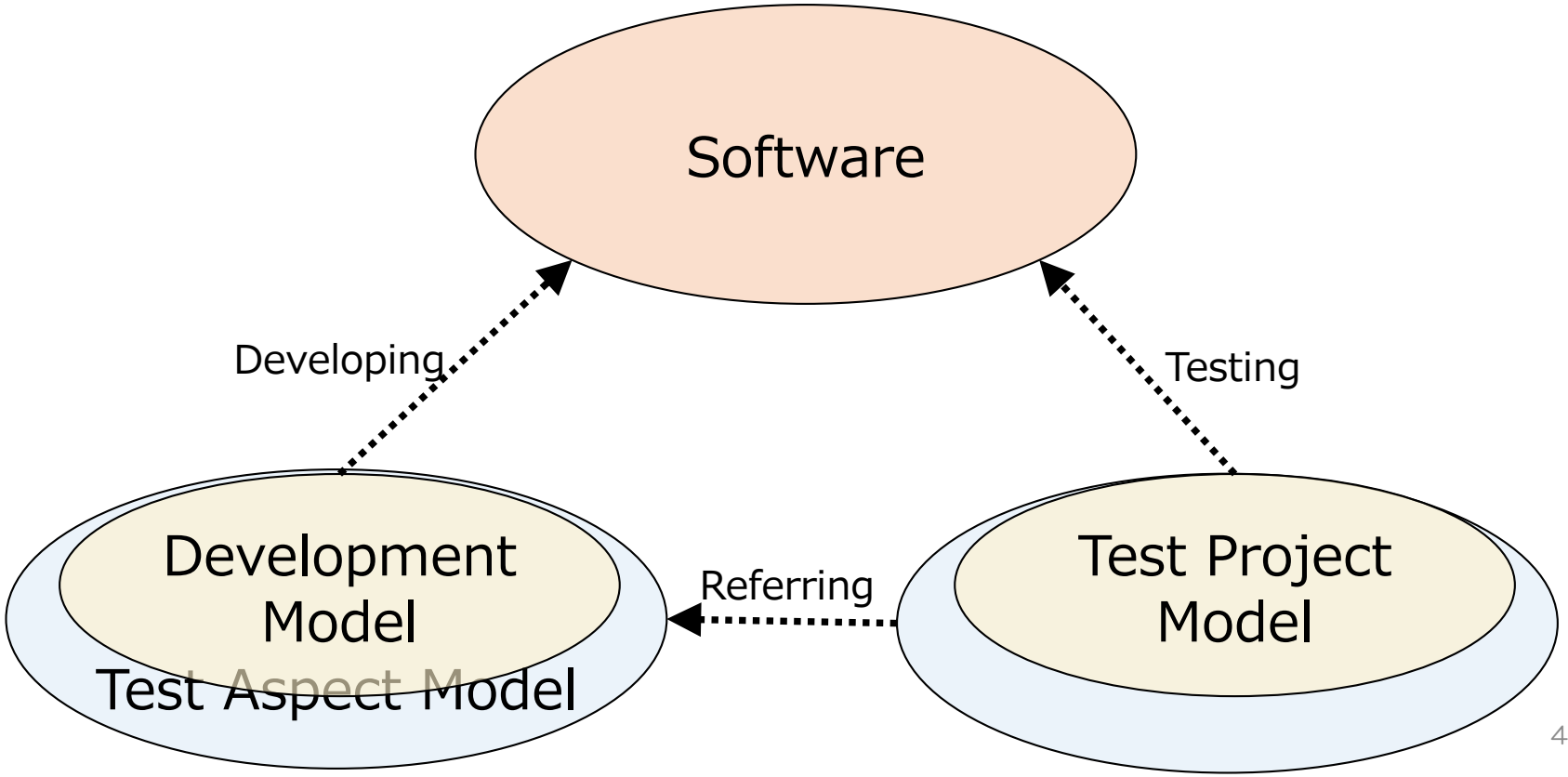
Proposal for Test Aspect Model



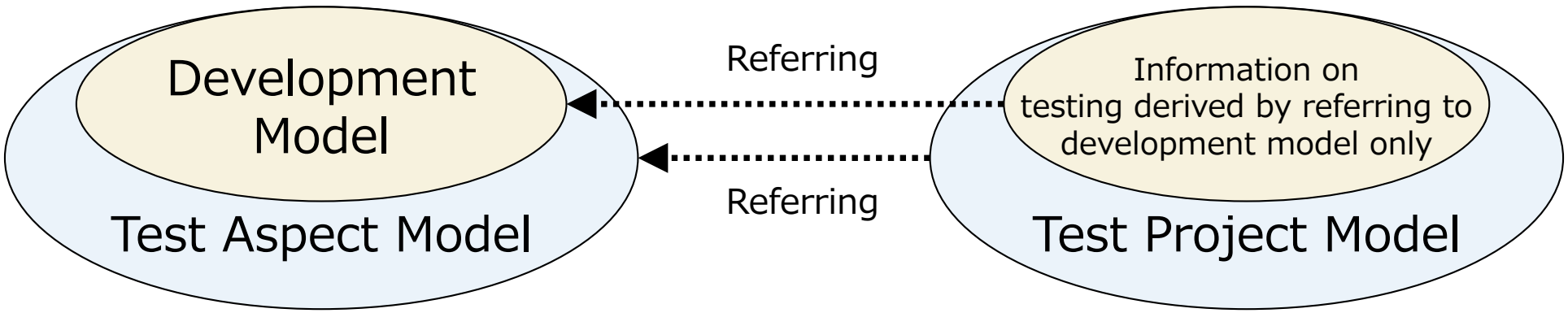
Proposal for Test Aspect Model



Proposal for Test Aspect Model

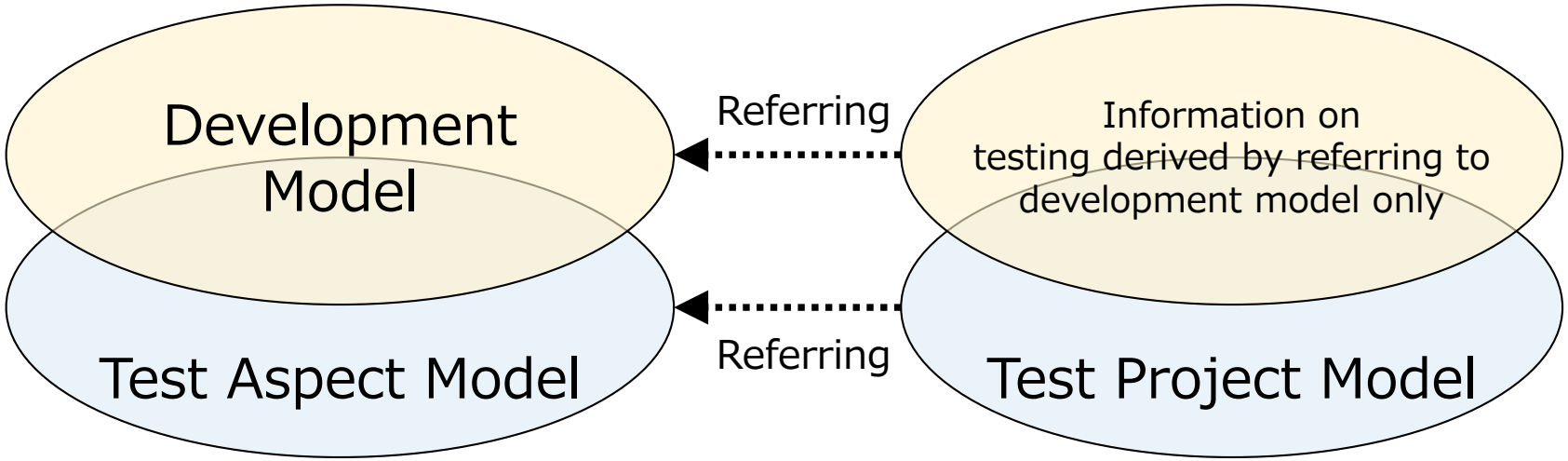


Proposal for Test Aspect Model



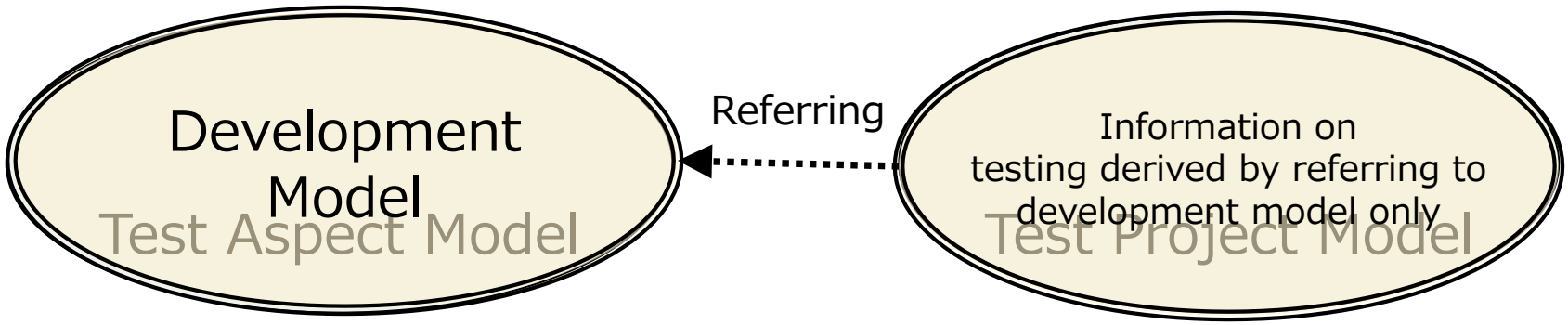
Case 1:
Development Model is Perfect

Development Model = Test Aspect Model ∈ Test project Model

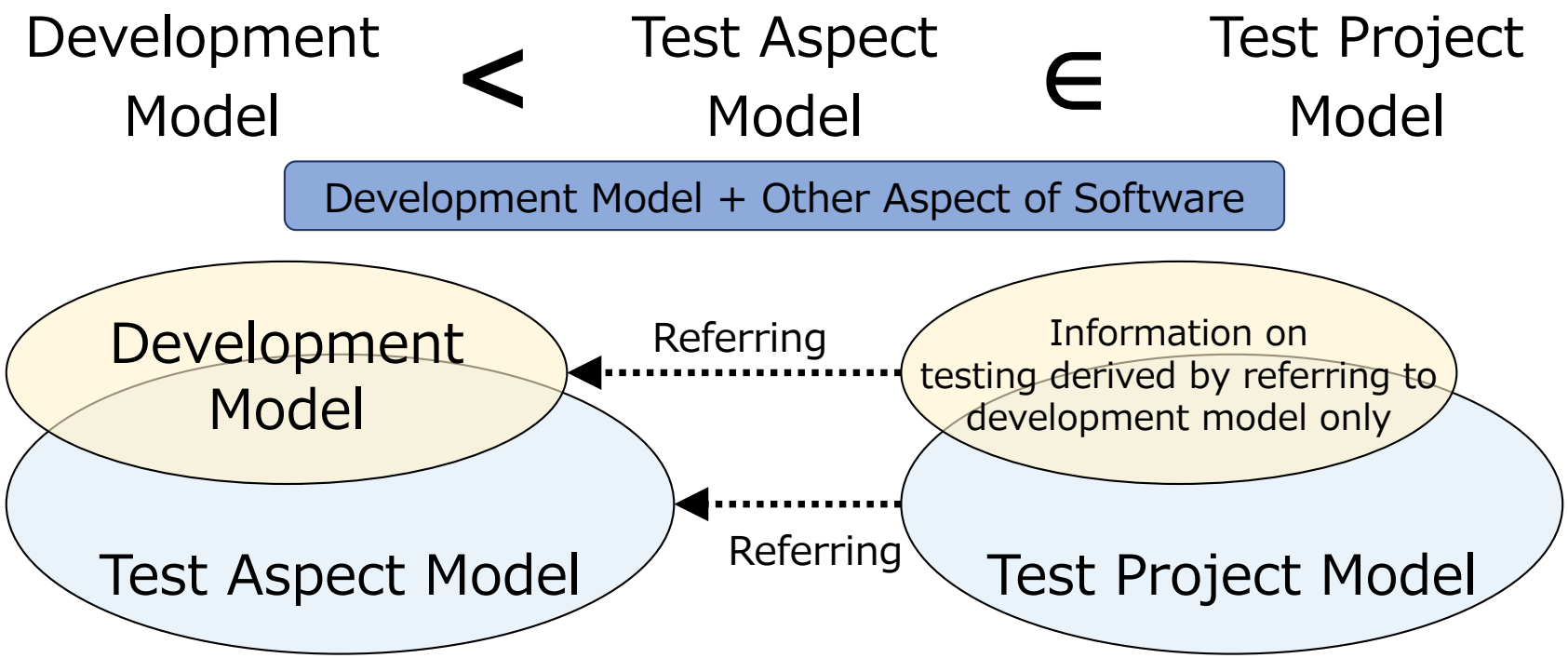


Case 1:
Development Model is Perfect

Development Model = Test Aspect Model ∈ Test project Model

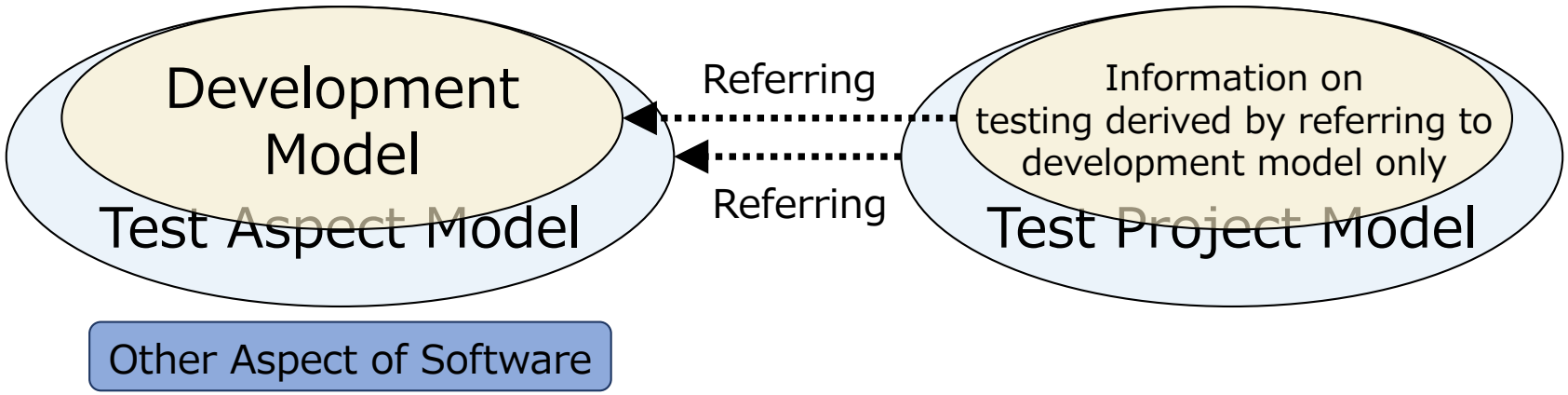


Case 2:
Development Model is Incomplete

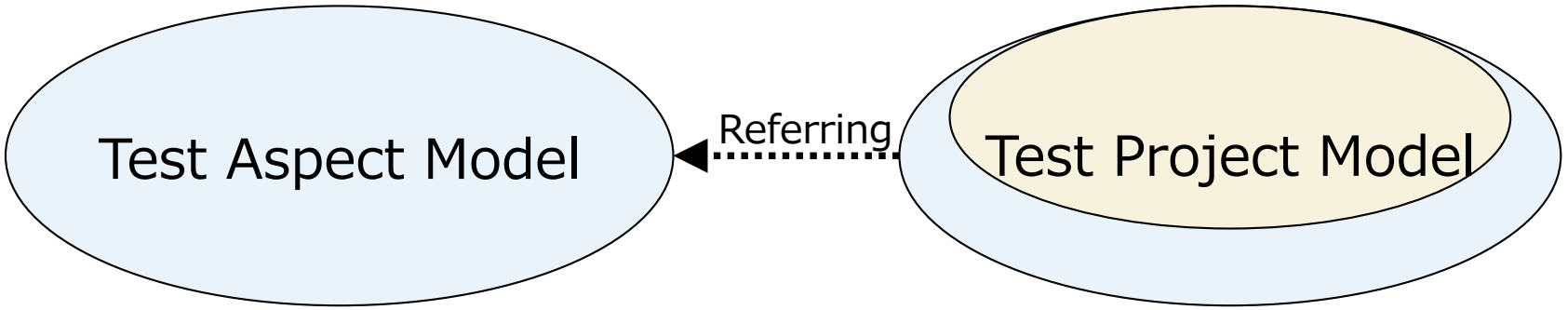


Case 2:
Development Model is Incomplete

Development Model < Test Aspect Model ∈ Test Project Model



Also Enrich Test Project Model



Using the Test Aspect Model

Furthermore, we suggest to share information and agreement among stakeholders, and to grasp the whole picture, using the test aspect model.

This approach will smoothen the problems when test designs rely on test engineer's skills.

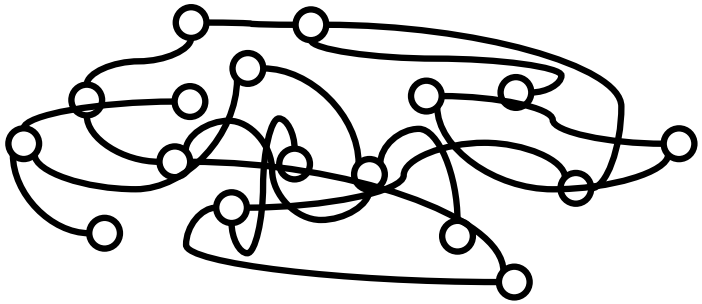
PROPOSAL FOR THE NOTATION

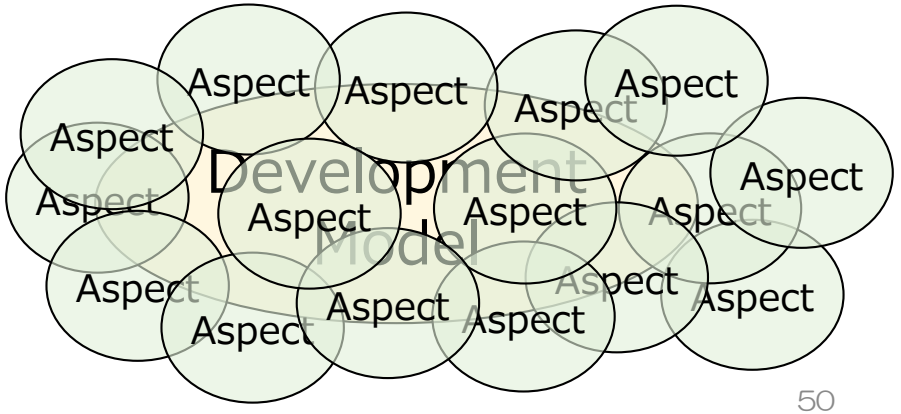
Next Problem

Complicated system has many test aspects...

Next Problem..

As the software itself is bigger and more complicated now.
→ Test aspect models also become complicated.





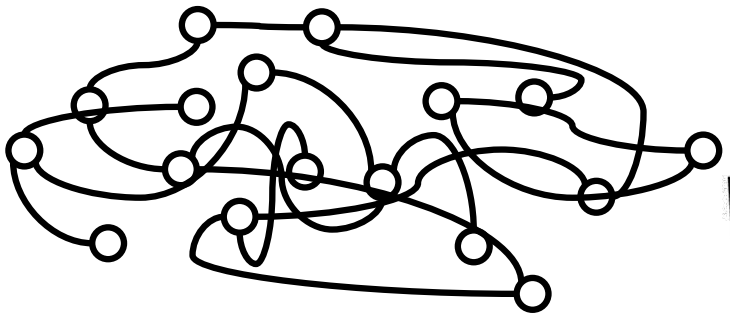
Next Problem..

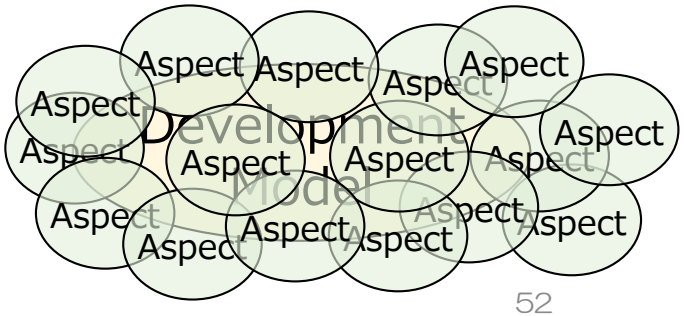
These test aspects are usually tacit knowledge.

Next Problem → Necessity for the Notation

- to share the information with other test engineers
- to grasp a holistic model
- to find the omitted test aspect information

The unified notation is essential…





Necessity for the Notation.

The requirement for the notation
In UML Testing Profile...

Necessity for the Notation

- to share the information with several test engineers

- to grasp a holistic model
- to find the omitted aspect information.

Requirements for the Notation

- to share the information with several test engineers
 → **The notation should be easy to be shared and to be learned by test engineers.**

- to grasp a holistic model
- to find the omitted aspect information.
 → **The notation should have capacity to organize complicated information.**

Example of Test Aspect

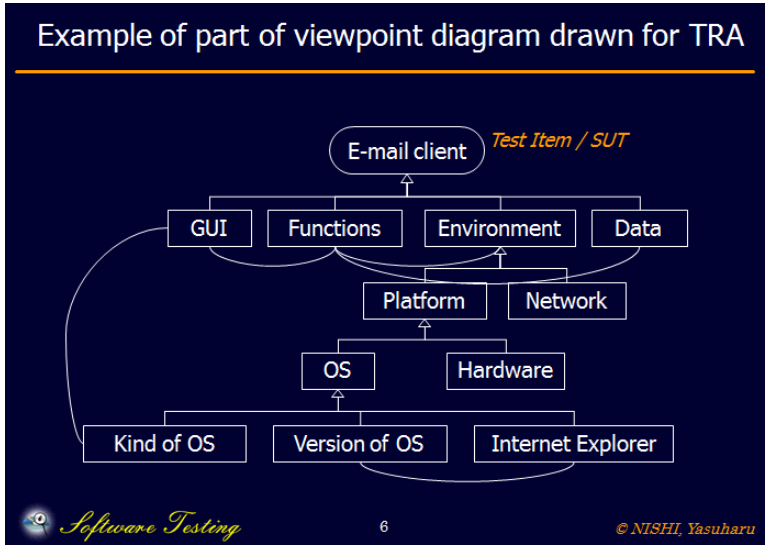
Test for ATM “deposit” function.
Following items(test aspects) may be considered for designing test cases.

Insert Money (Test)
- Valid Money
- Invalid Money
- ...

Reliability (Test)
- Repeated transaction
- Contingency approach
- ...

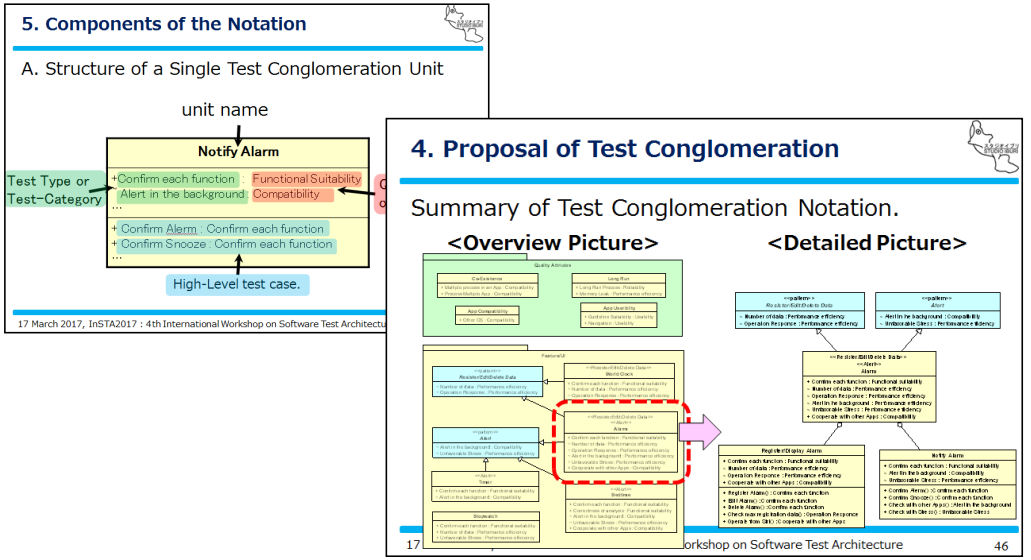
References of other Notations

Viewpoint Diagram



Y.Nishi, “Design principles in Test Suite Architecture,” International Workshop on Software Test Architecture (InSTA 2015), Graz, Austria, April 2015.

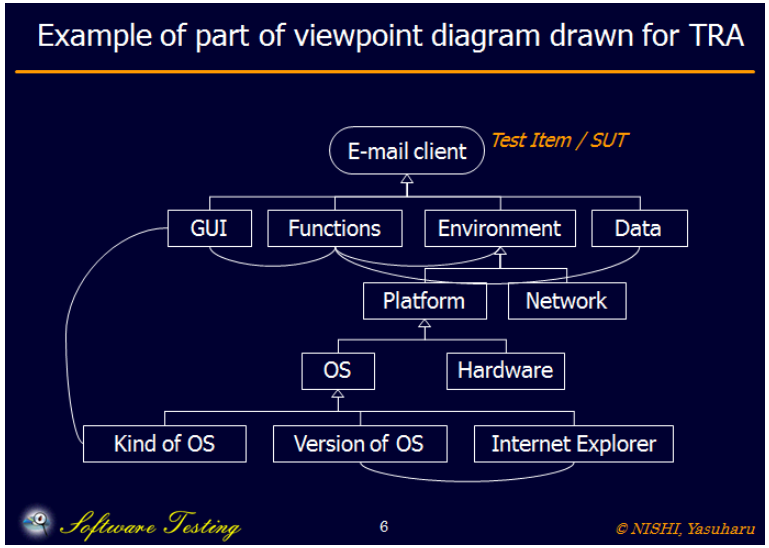
Test Conglomeration

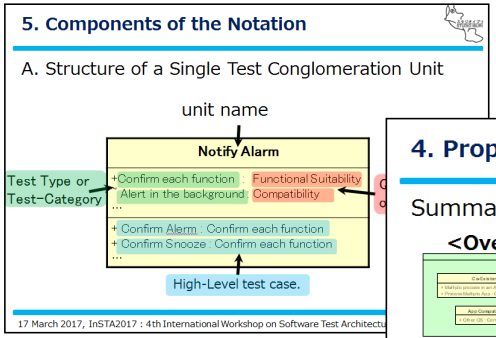


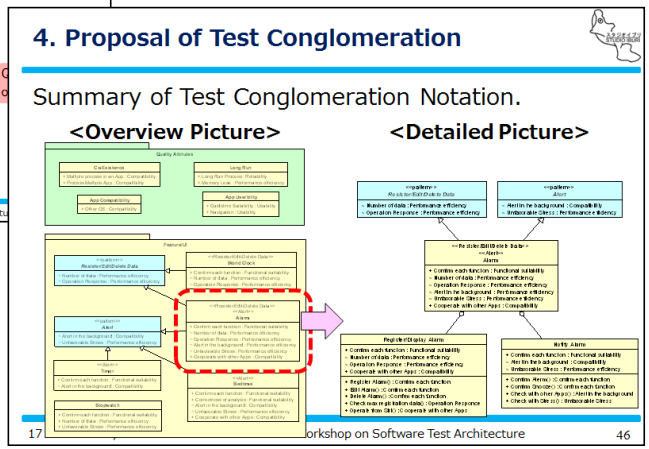
N.Mizuno, M.Nakakuni and Y.Seino, “Test Conglomeration - Proposal for Test Design Notation like Class Diagram,” International Workshop on Software Test Architecture (InSTA2017), March 2017.

Requirements for the Notation

Each element has relationship with others.





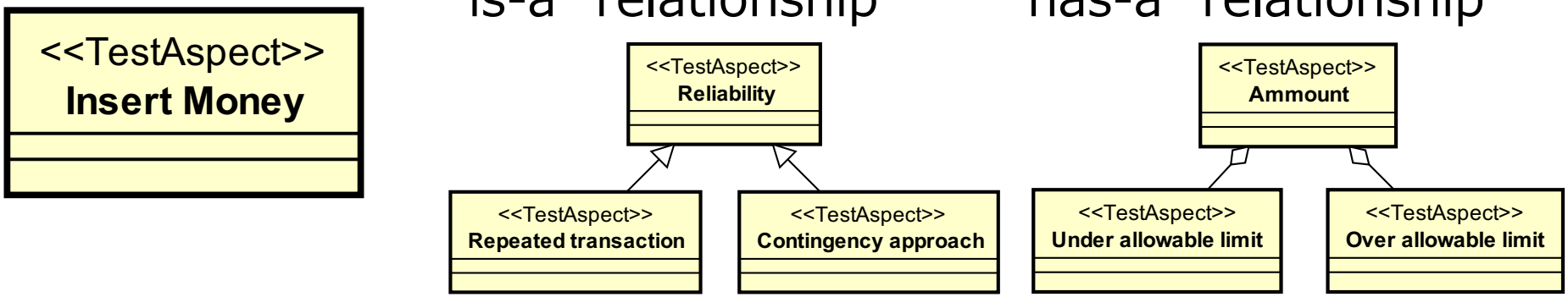


Requirements for the Notation

Each element has relationship with others.

In the notation, “has-a”,“is-a ” relationships are useful.

→ UML metaclass “class” is suitable
for describing test aspect.



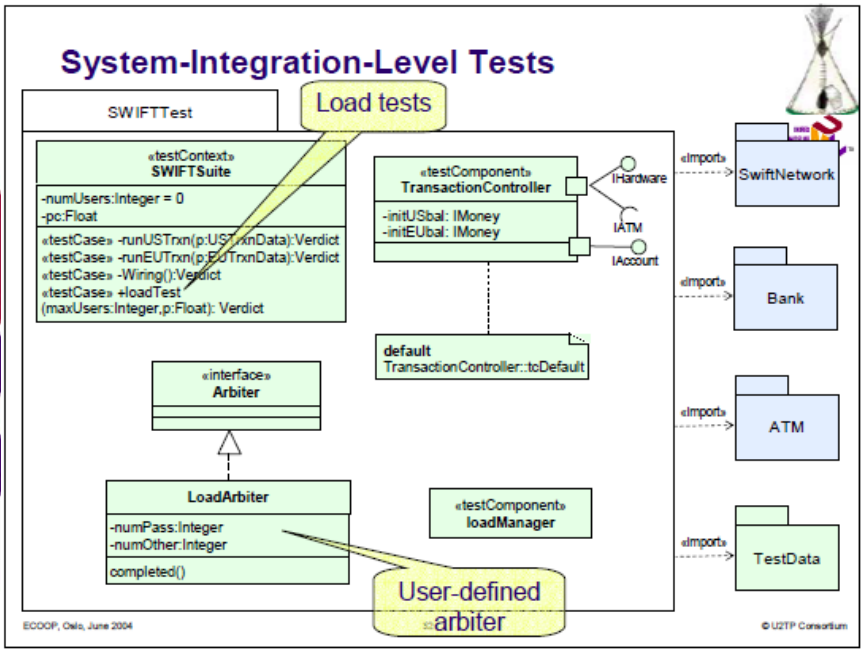
In UML Testing Profile

http://utp.zen-tools.com/

In UML Testing Profile

Test System Architecture

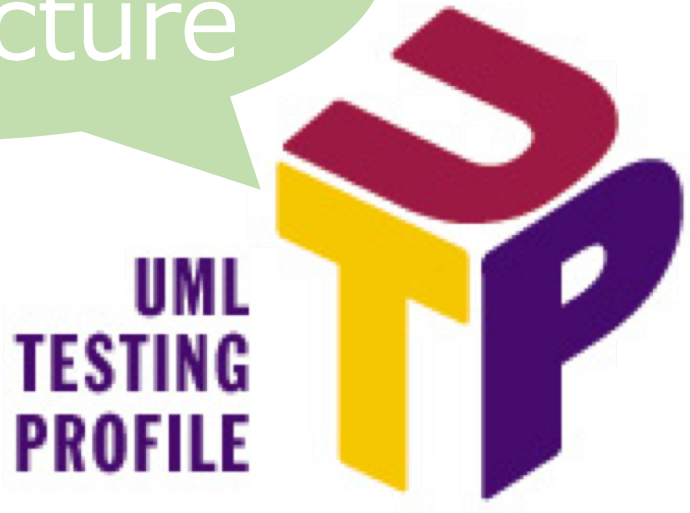
UML TESTING PROFILE



http://utp.zen-tools.com/ The UML Testing Profile - Tutorial at the ECOOP 2004 - http://folk.uio.no/oysteinh/Schieferdecker-Haugen-ECOOP2004-U2TP.pdf

In UML Testing Profile

Test System Architecture



http://utp.zen-tools.com/

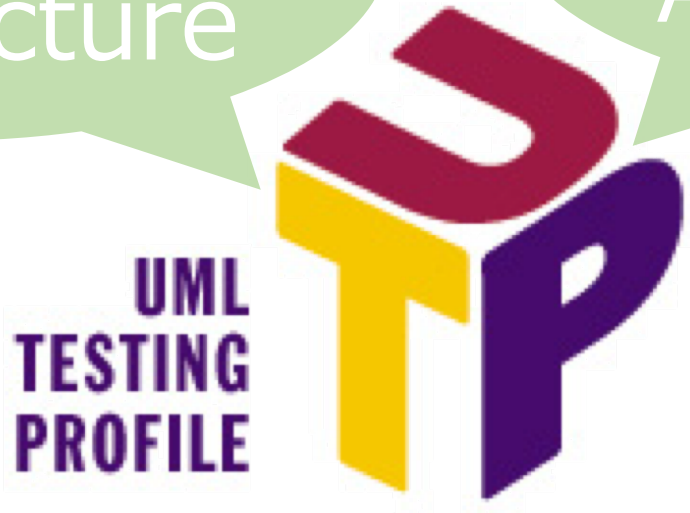
It has no concept to describe an element like the test aspect.

Nor it has no way to describe structure of test cases.

In UML Testing Profile

Test System Architecture

Test Suite Architecture?



http://utp.zen-tools.com/

It has no concept to describe an element like the test aspect.

Nor it has no way to describe structure of test cases.

We propose for

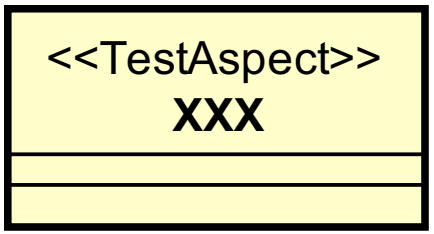
http://utp.zen-tools.com/

We propose for

Enhancing UTP2 with “Test Aspects”

http://utp.zen-tools.com/

+



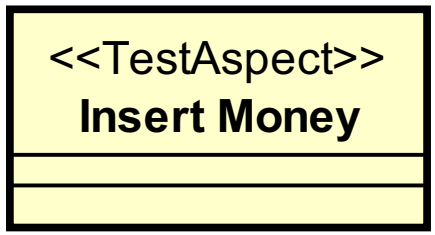
Explanation and Example of the Notation for Test Aspect

The notation of Test Aspect

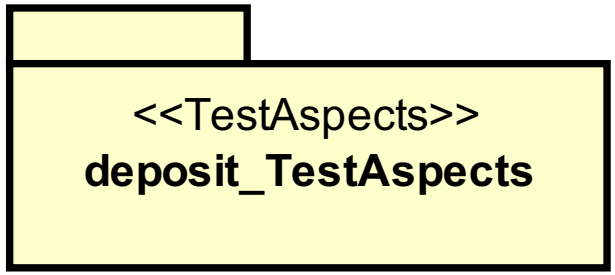
Example of Test Aspect Model

The Notation for Test Aspect

Metaclass “class” with stereotype “TestAspect”

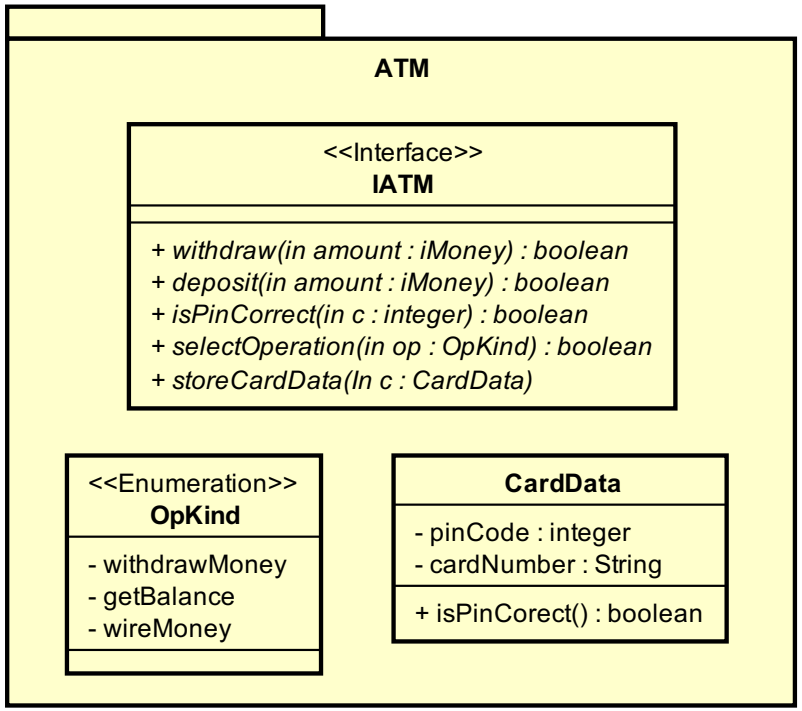


Metaclass “package” with stereotype “TestAspects”



Example of Test Aspect Model

Example for ATM “deposit” test

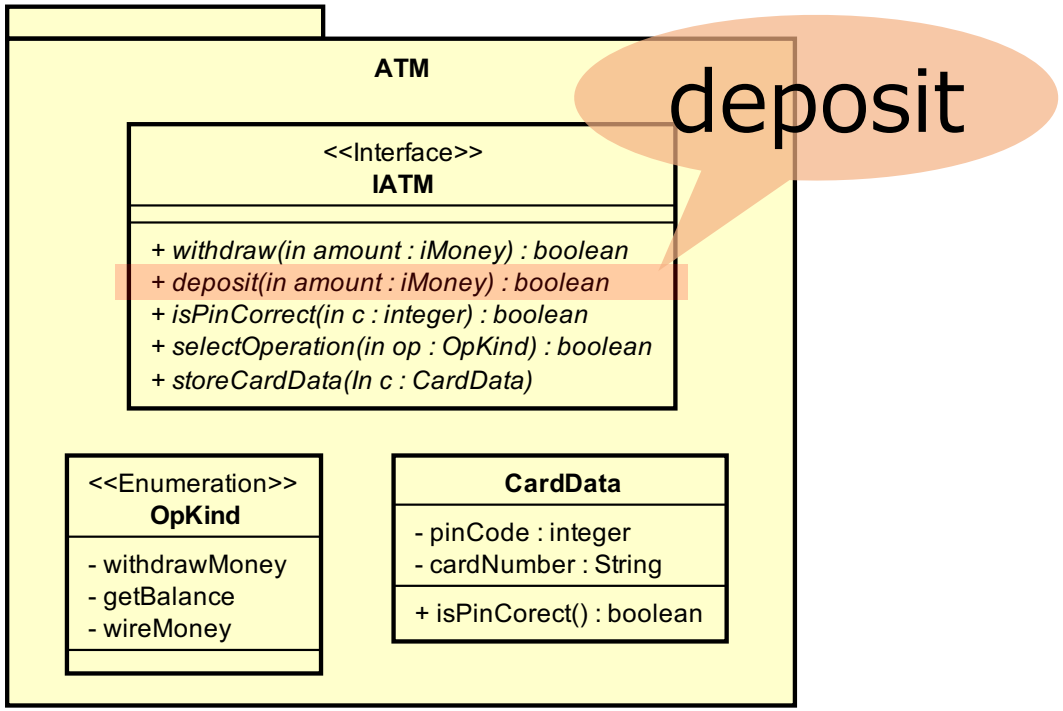


described in UTP2.0

https://www.omg.org/spec/UTP/About-UTP/

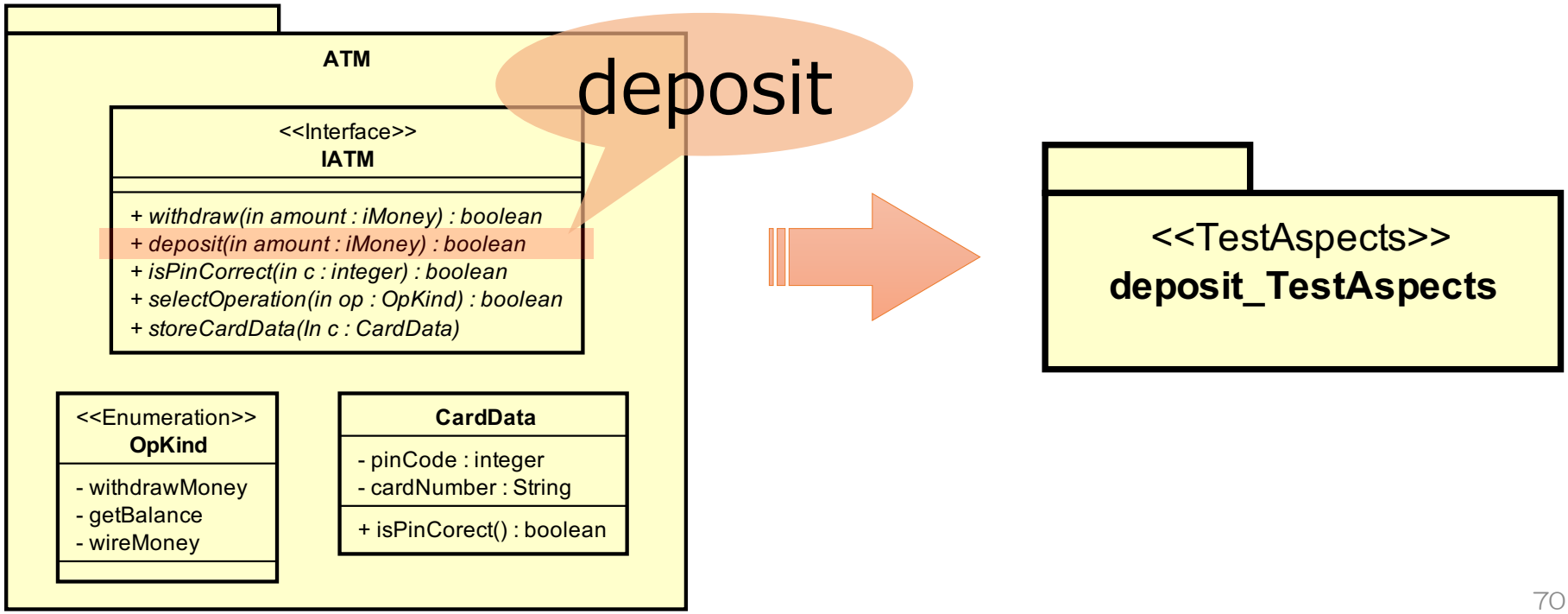
Example of Test Aspect Model

"Test Aspects" deposit package is extracted.



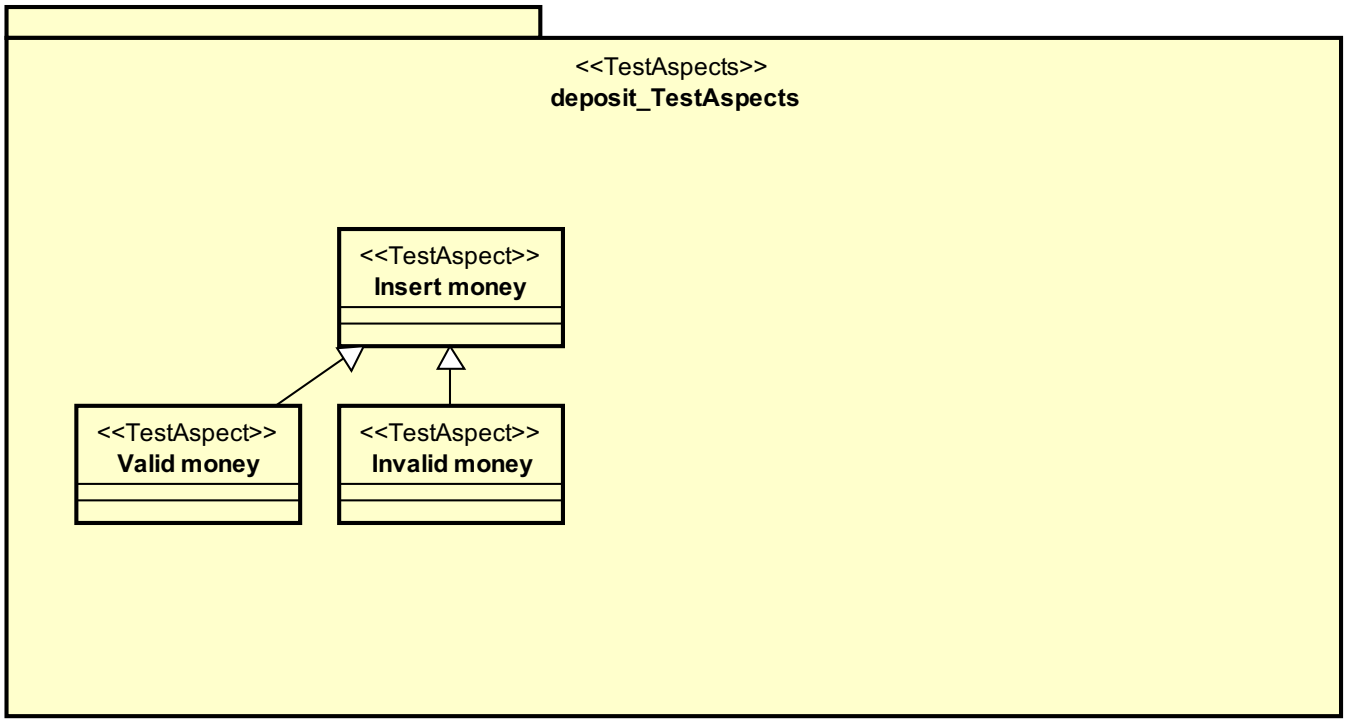
Example of Test Aspect Model

"Test Aspects" deposit package is extracted.



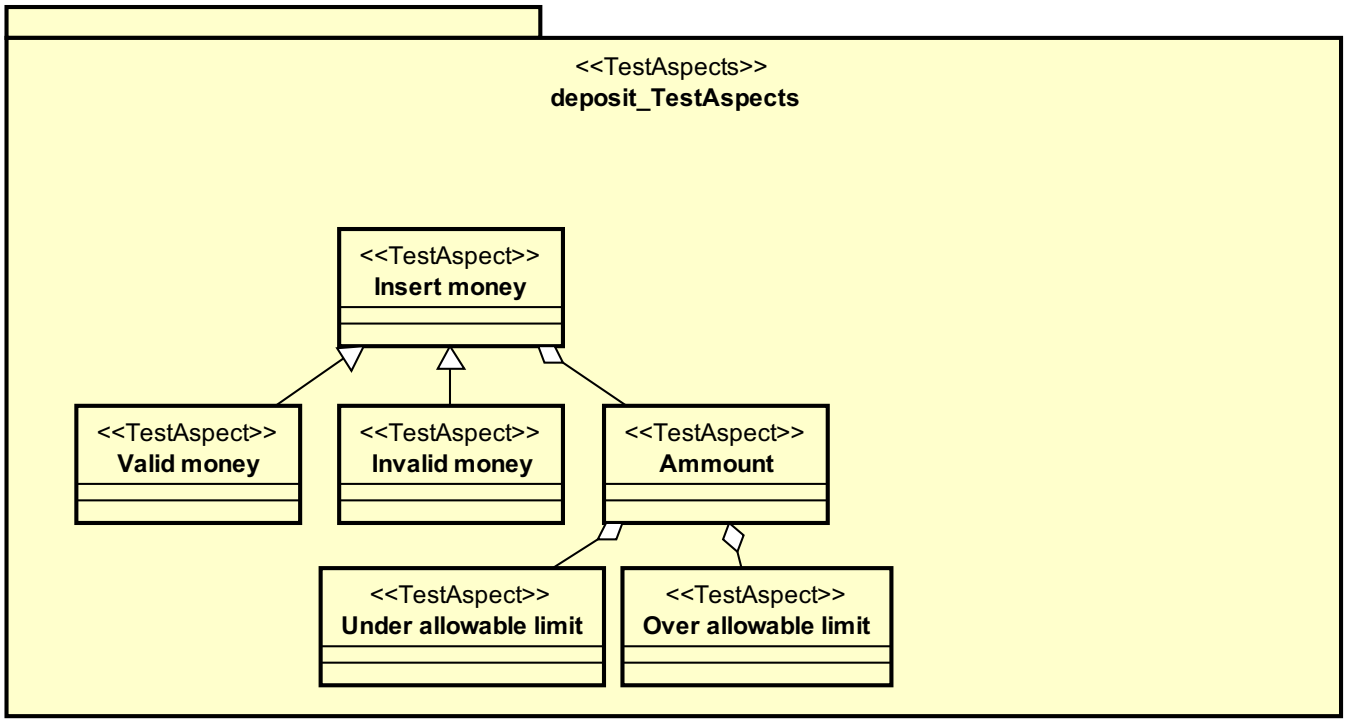
Example of Test Aspect Model

Deposit has following test aspects.



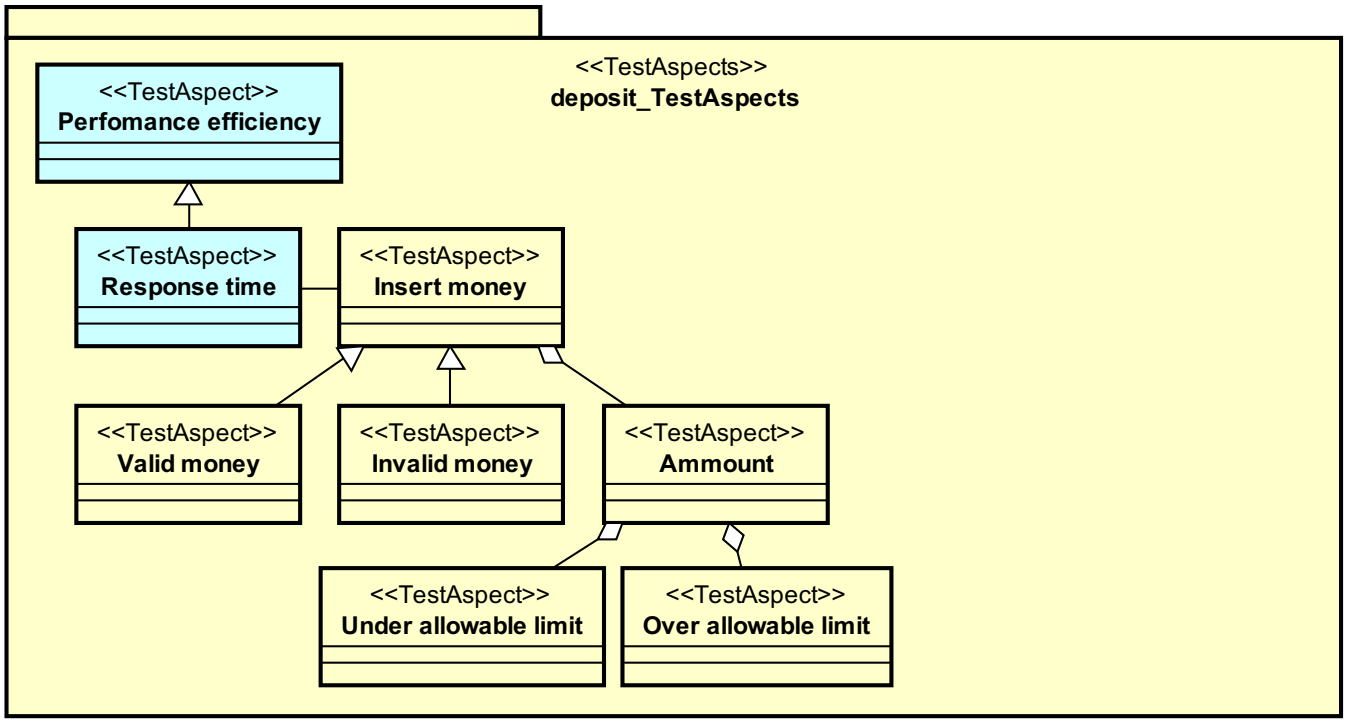
Example of Test Aspect Model

Deposit has following test aspects.



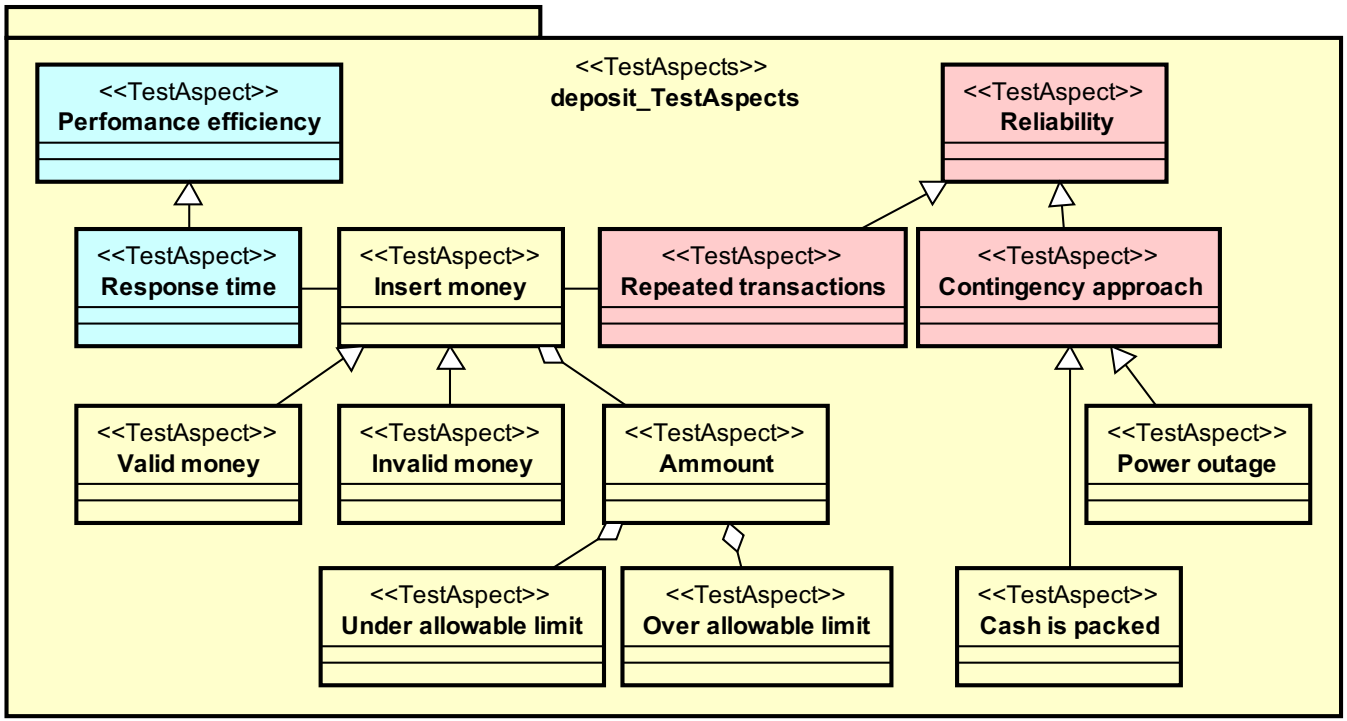
Example of Test Aspect Model

Deposit has following test aspects.



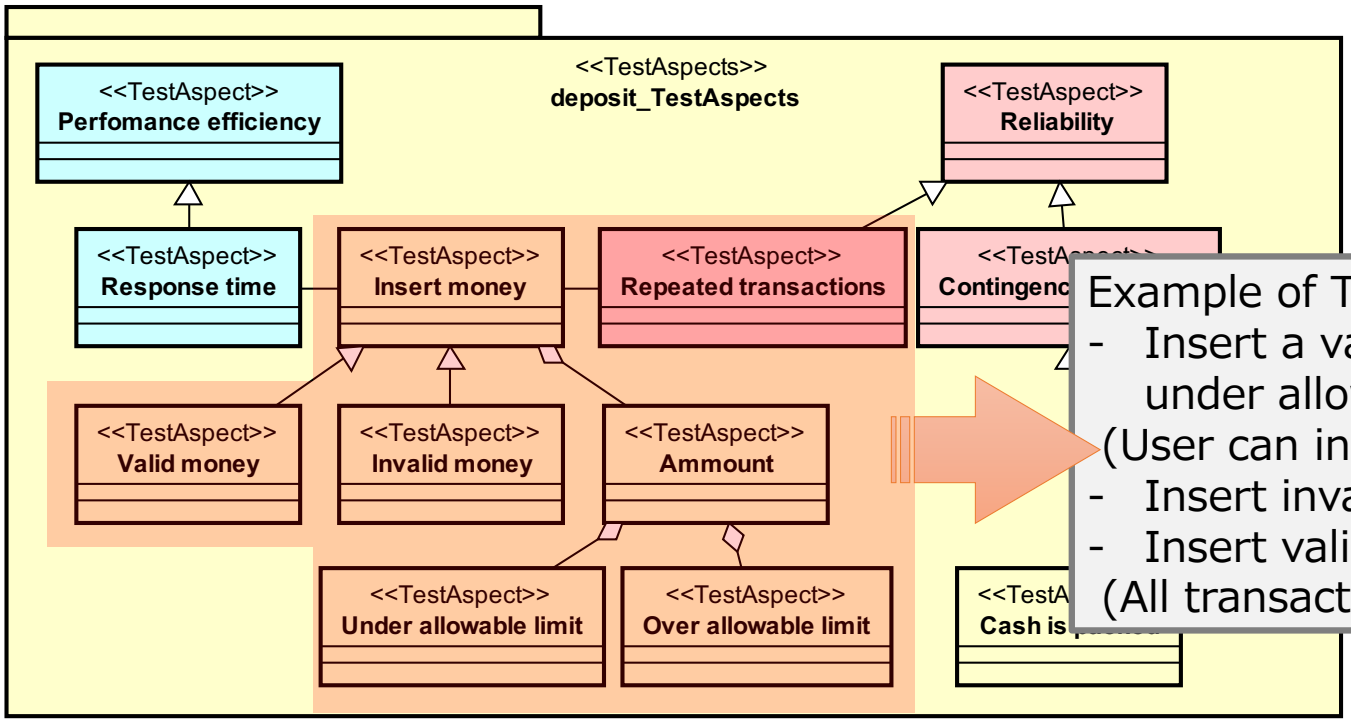
Example of Test Aspect Model

Deposit has following test aspects.



Example of Test Aspect Model

Test cases are designed from the part of these test aspects.

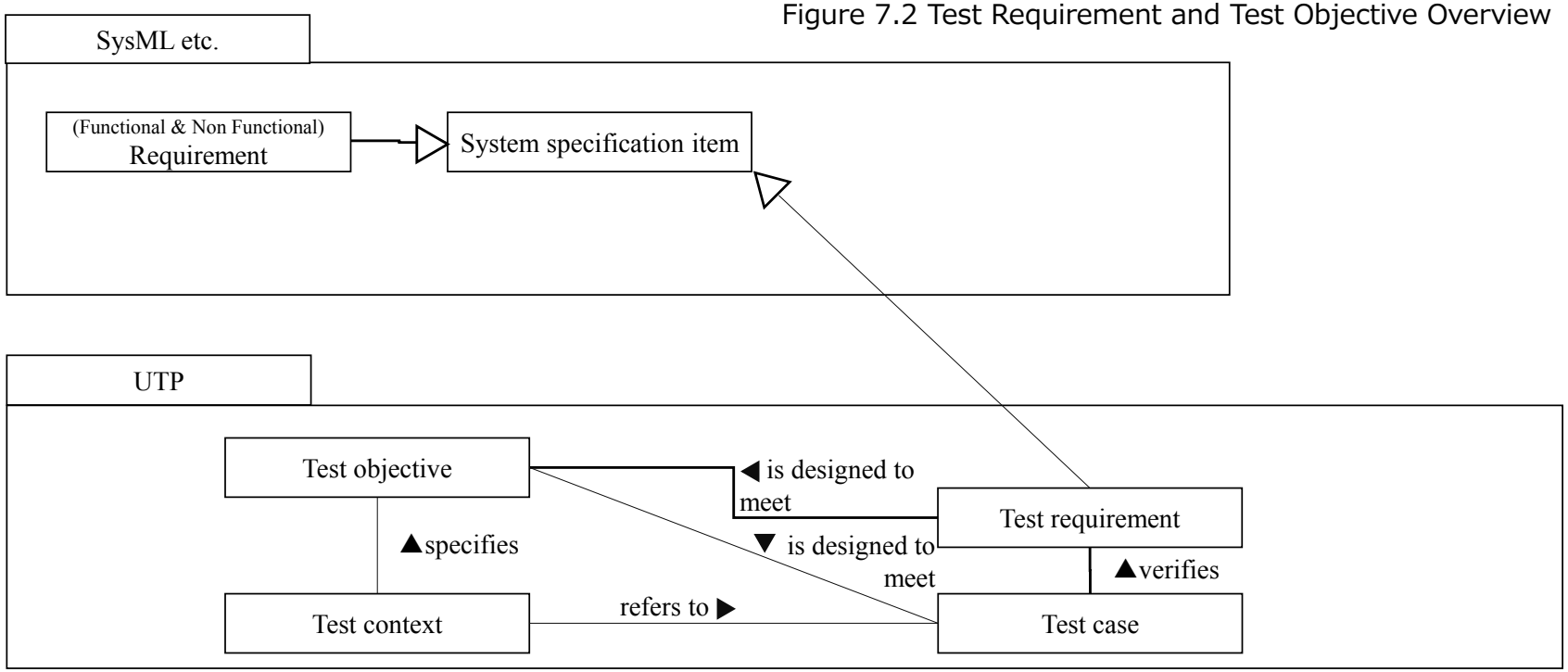


Example of Test Cases...
- Insert a valid money that amount is under allowable limit
(User can insert money successfully)
- Insert invalid money (Error will occur)
- Insert valid money repeatedly
(All transactions success normally)

Position of Test Aspect in UTP2

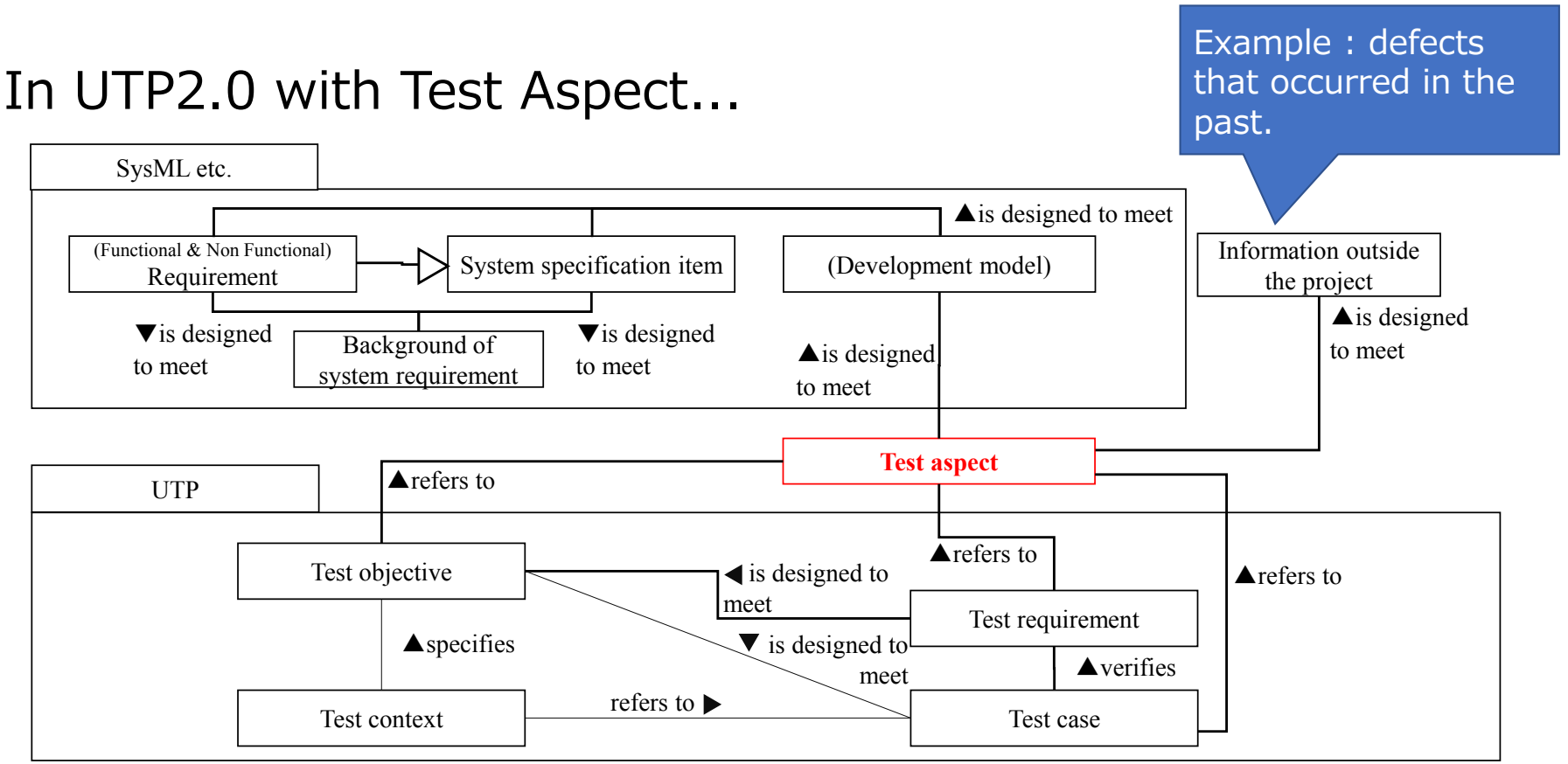
In UTP2.0...

UML Testing Profile (UTP) Version 2.0 – Beta
Publication Date: September 2017
Figure 7.2 Test Requirement and Test Objective Overview



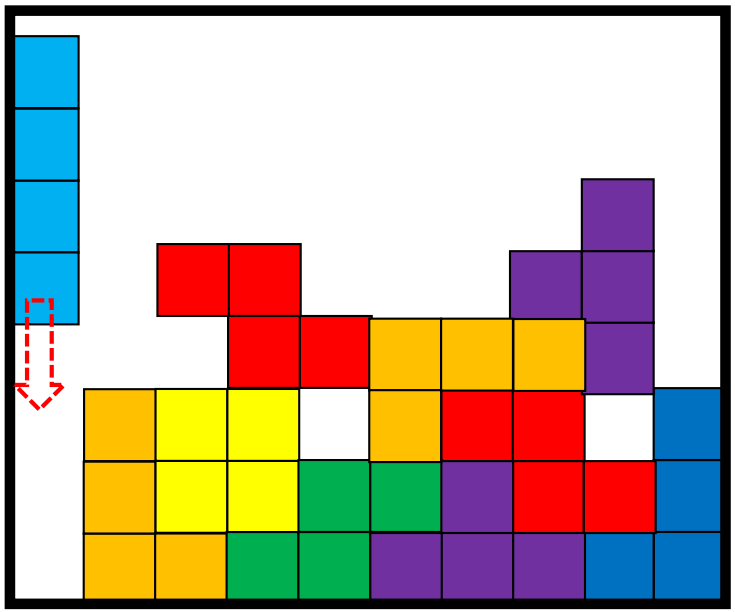
Position of Test Aspect in UTP2

In UTP2.0 with Test Aspect...



Advantage of Test Aspect

The test aspect can complement development models.



Advantage of Test Aspect

- Test aspects can be used to organize the information in the test design process. This characteristic can improve systemized knowledge of the test design.
- The relationship among test aspects enables to grasp the intention of the test design.
- Test engineers can maintain the structure of test aspects using the test aspect model.

As a result, maintainability of the test suite improves.

CONCLUSION

Conclusions

- The test aspect can complement the development model.
- The test aspect can indicate missing information in the development model.
- Test aspect model can describe relationships among test aspects.
- Test aspects organized appropriately in the test aspect model can complement necessary information that is not found in development models.
- This model enables to reduce test case omissions in test design process.

Thank you!

Any Questions?

We are not good at English.
So please speak clearly and slowly.

Special Thanks

Satomi-Juku

What is Satomi-Juku ?

"Satomi-juku" is a research group of test architecture.
It is a place to disclose & share each advanced test development methods them through discussion.

References

References

[1] OMG, “UML Testing Profile (UTP) Version 2.0 - Beta,” http://www.omg.org/spec/UTP/2.0/, September 2017.

[2] Y.Nishi, “Design principles in Test Suite Architecture,” International Workshop on Software Test Architecture (InSTA 2015), Graz, Austria, April 2015.

[3] Y. Nishi, T. Katayama and S. Yoshizawa, “Combinatorial Test Architecture Design Using Viewpoint diagram,” International Workshop on Combinatorial Testing (IWCT2013), Luxembourg, March 2013, CD-ROM.

[4] OMG, “UML Version 2.4.1,” http://www.uml.org/, 2012.

References

[5] OMG, “SysML Version 1.5,” http://www.omgsysml.org/, 2017.

[6] ISO/IEC 25010:2011, “Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE)–System and software quality models,” 2011.

[7] Glenford J. Myers, Tom Badgett, Corey Sandler, "The Art of Software Testing 3rd edition," Wiley, 2011.

[8] N.Mizuno, M.Nakakuni and Y.Seino, “Test Conglomeration - Proposal for Test Design Notation like Class Diagram,” International Workshop on Software Test Architecture (InSTA2017), March 2017.

References

[9] T.Yumoto, T.Matsuodani, and K.Tsuda, ” A Study on the effectiveness of Test-Categories based test analysis” , International Workshop on Software Test Architecture (InSTA2016), Chicago, April 2016.

[10] Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides, “Design Patterns: Elements of Reusable Object-Oriented Software,” Addison-Wesley Professional, 1994