

Requirements Diagram

Introduction

- ❖ One of the most important steps in a system development is collecting the requirements and finalize each one of them. Skipping this phase or completing it in a neglected way might heart the whole system development process.
- ❖ SysML defines several elements that assist with describing and modeling the requirements.

Requirements Types

- ❖ There are different types of requirements (e.g. response time, size, specific functionality etc.).

Functional requirements are usually described using the use case diagrams. None functional requirements are usually described using the requirements diagram.

Requirement SysML Definition

- ❖ A requirement describes one or more properties or behaviors of a system that always have to be met. A requirement is a stereotype <<requirement>> of the UML element class.

(System Engineering with SysML/UML by Tim Weilkiens)

The Requirements Contract

- ❖ Each requirement describes a contract between the principals and those that design and develop the system.

The Requirement Notation

- ❖ Each requirement is represented using the UML class notation together with the <<requirement>> stereotype.
- ❖ Each requirement notation has a unique identifier (ID) and a descriptive text that describes it.

SysML doesn't specify any specific structure the ID should follow. The descriptive text can be a reference for an external resource such as a specific data record in a requirements management tool. Apart of these two attributes we cannot add operations or other attributes. Nevertheless, it is possible to extend the requirement in order to add more properties, such as 'priority'.

The Requirement Notation

<code><<requirement>></code> W3C XHTML Standard
<code>+id = "XHTML 1.0"</code> <code>+text = "http://www.w3.org/TR/xhtml1/"</code>

The Requirement Notation

- ❖ In terms of the model we have in the background of our requirement diagram, each class is considered as an abstract one.

It wouldn't make sense to create instances of a requirement class.

- ❖ Requirements cannot be generalized.

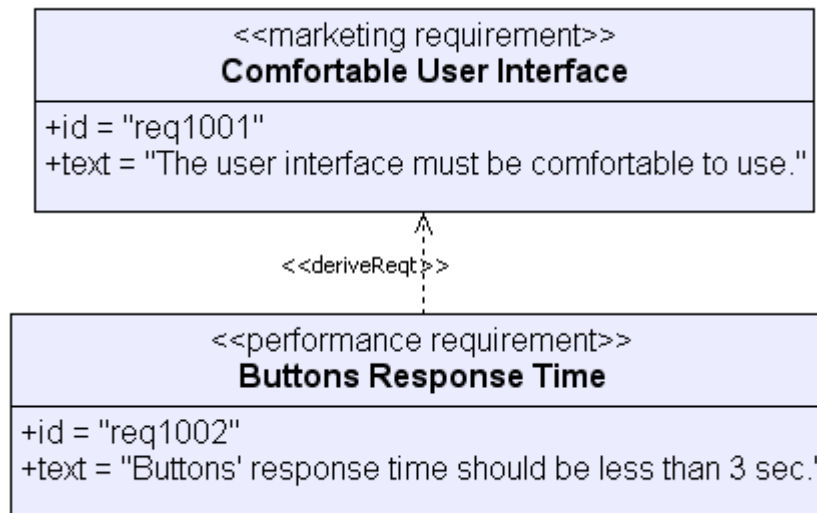
There is no sense in generalizing a requirement as it doesn't have any operation or attribute to inherit.

The Derive Relationship

- ❖ The derive requirement relationship describes a requirement that derives from another requirement. We reflect one requirement deriving from another by using the `<<deriveReq>>` stereotype together with the dependency dashed arrow pointing at the requirement from which the other derives.

(System Engineering with SysML/UML by Tim Weilkiens)

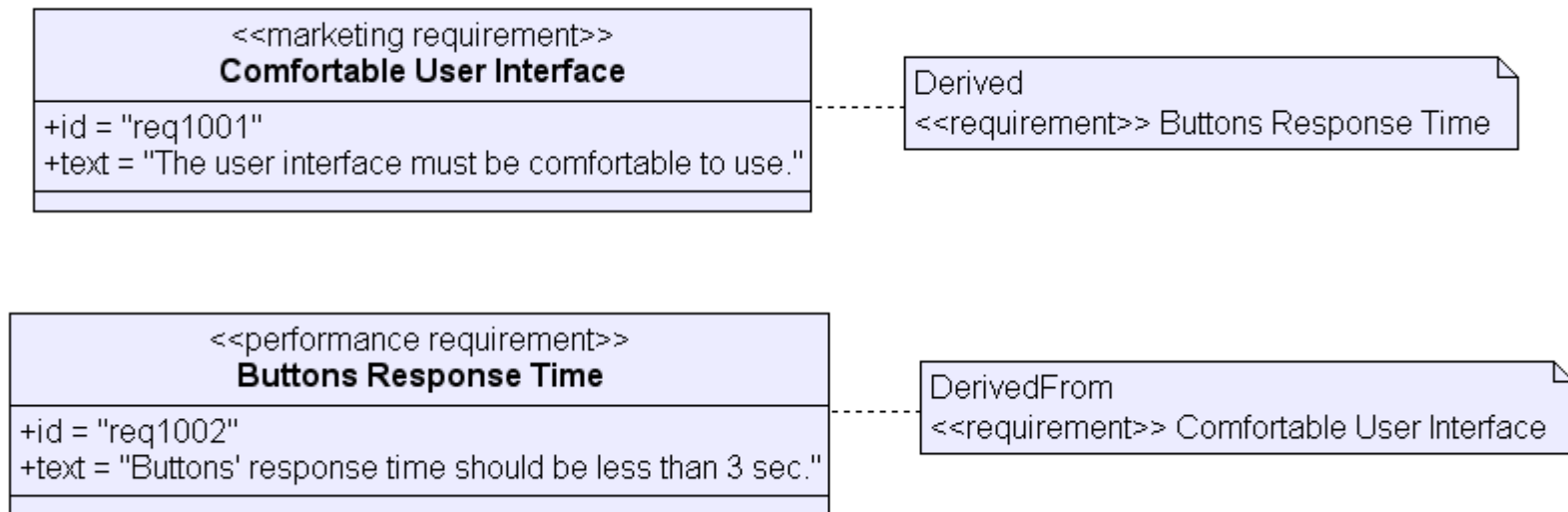
The Derive Relationship



The Derive Relationship

- ❖ We can show that one requirement derives from another by adding comments with either the “Derived” or the “DerivedFrom” titles.

The Derive Relationship



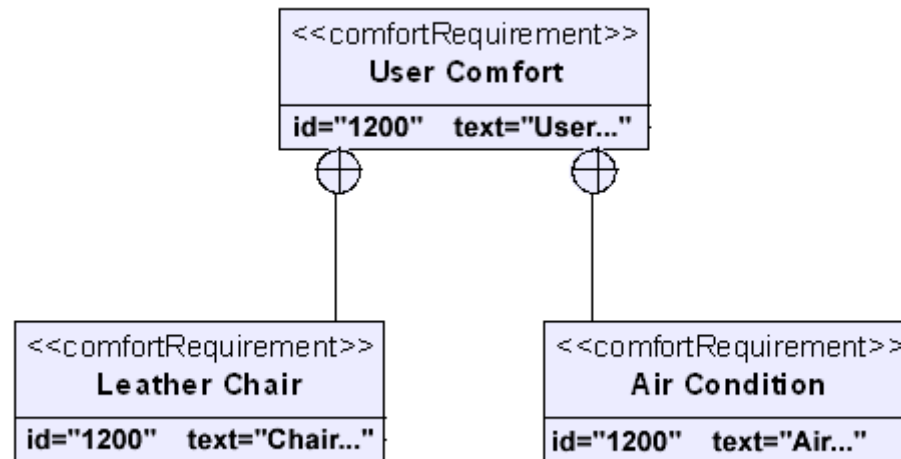
The Namespace Containment

- ❖ The namespace containment describes a case in which one requirement is contained within another.

(System Engineering with SysML/UML by Tim Weilkiens)

When the master requirement is satisfied when all of its sub requirements are satisfied we get a containment relationship between the two. Each requirement cannot contain requirements with the same name and each requirement cannot be part of more than one other requirement at the same time. The notation is the same notation used to represent inner classes.

The Namespace Containment



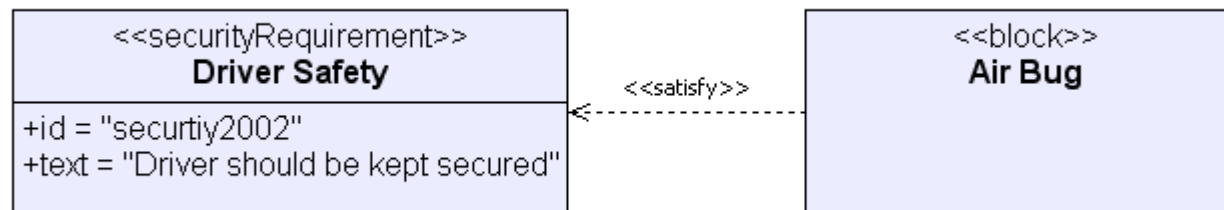
The Satisfy Relationship

- ❖ The satisfy relationship describes a design element that satisfies a requirement. It is a stereotype <<satisfy>> of the UML relationship realization.

(System Engineering with SysML/UML by Tim Weilkiens)

Each design element in our model satisfies a requirement (either directly or indirectly). We use a dashed arrow and the <<satisfy>> keyword.

The Satisfy Relationship



The Copy Relationship

- ❖ The copy relationship describes a requirement which is a copy of another requirement.

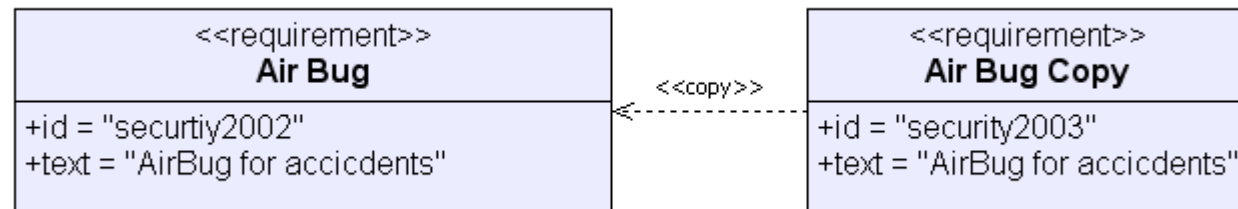
(System Engineering with SysML/UML by Tim Weilkiens)

When there is a need to reuse a requirement within another context we can use the <<copy>> relationship in order to enable that. This way we overcome the limit to have each requirement within one context at the most.

The Copy Relationship

- ❖ The copy we get is denoted as a dashed arrow pointing from the copy to the original and with the `<<copy>>` stereotype.
- ❖ The name and the ID might be different. The descriptive text **MUST** be identical.

The Copy Relationship



Test Cases

- ❖ “A test case is a flow that checks whether or not the system satisfies a requirement.”
(System Engineering with SysML/UML by Tim Weilkiens)
- ❖ The notation used to represent a test case is a simple rectangle with the name of the test case written within it together with the `<<testCase>>` stereotype.

Test Cases

<<testCase>>

Send Testing SMS

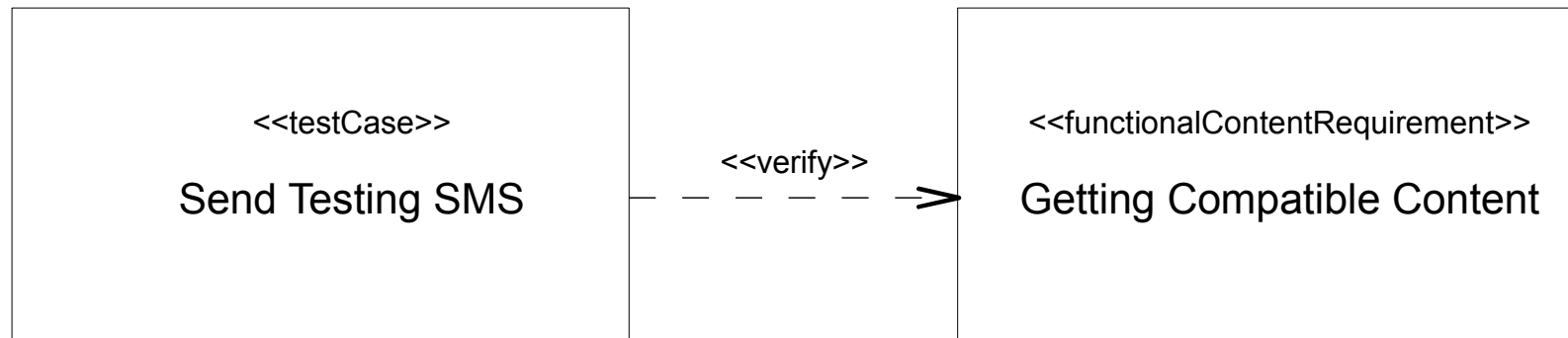
The Verify Relationship

- ❖ The verify relationship connects a test case with the requirement that is verified by that test case.
(System Engineering with SysML/UML by Tim Weilkiens)
- ❖ The verify relationship describes which test case tests each requirement.

The Verify Relationship

- ❖ The notation is a dashed arrow with the <<verify>> stereotype pointing at the requirement.
- ❖ We will use the <<verify>> relationship no matter whether the test verifies the requirement in a partial or a complete way.

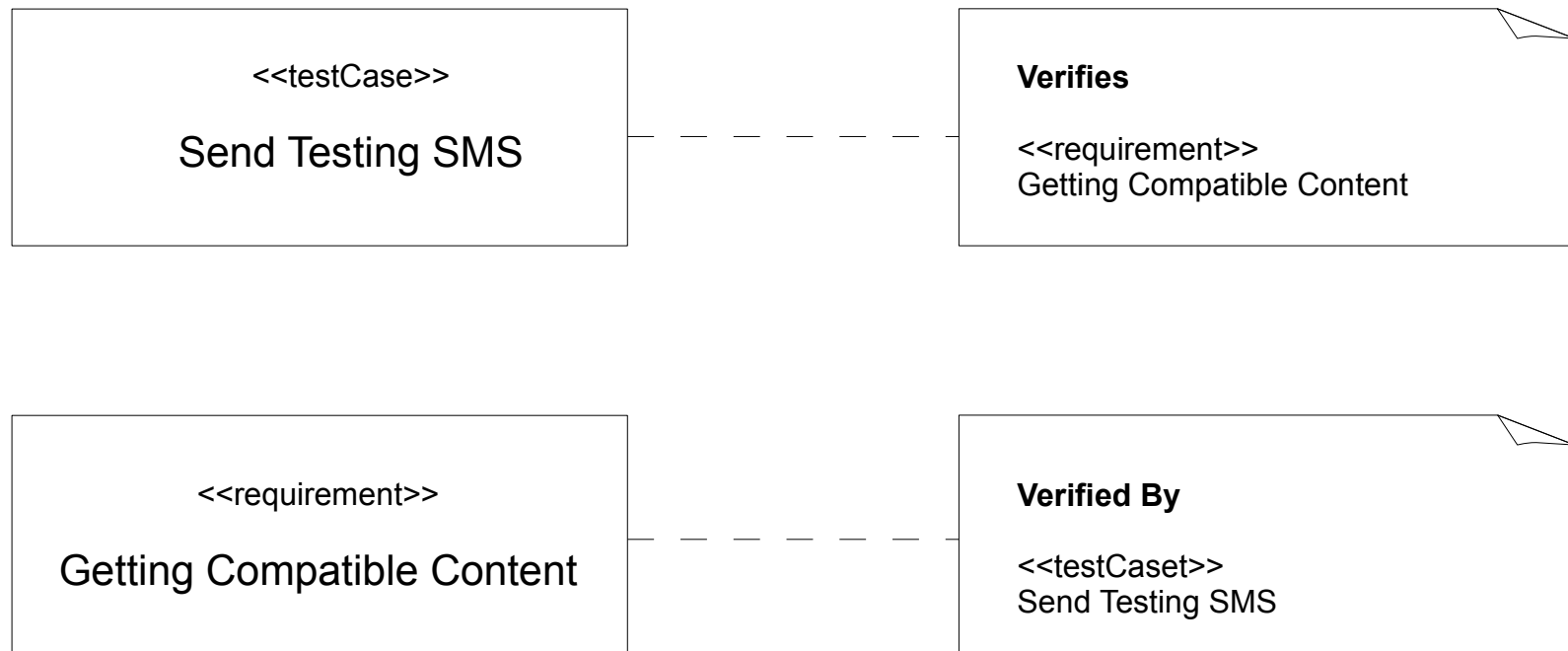
The Verify Relationship



The Verify Relationship

- ❖ Alternative notation for showing the verify relationship includes the usage of the comment symbol.

The Verify Relationship



The Refine Relationship

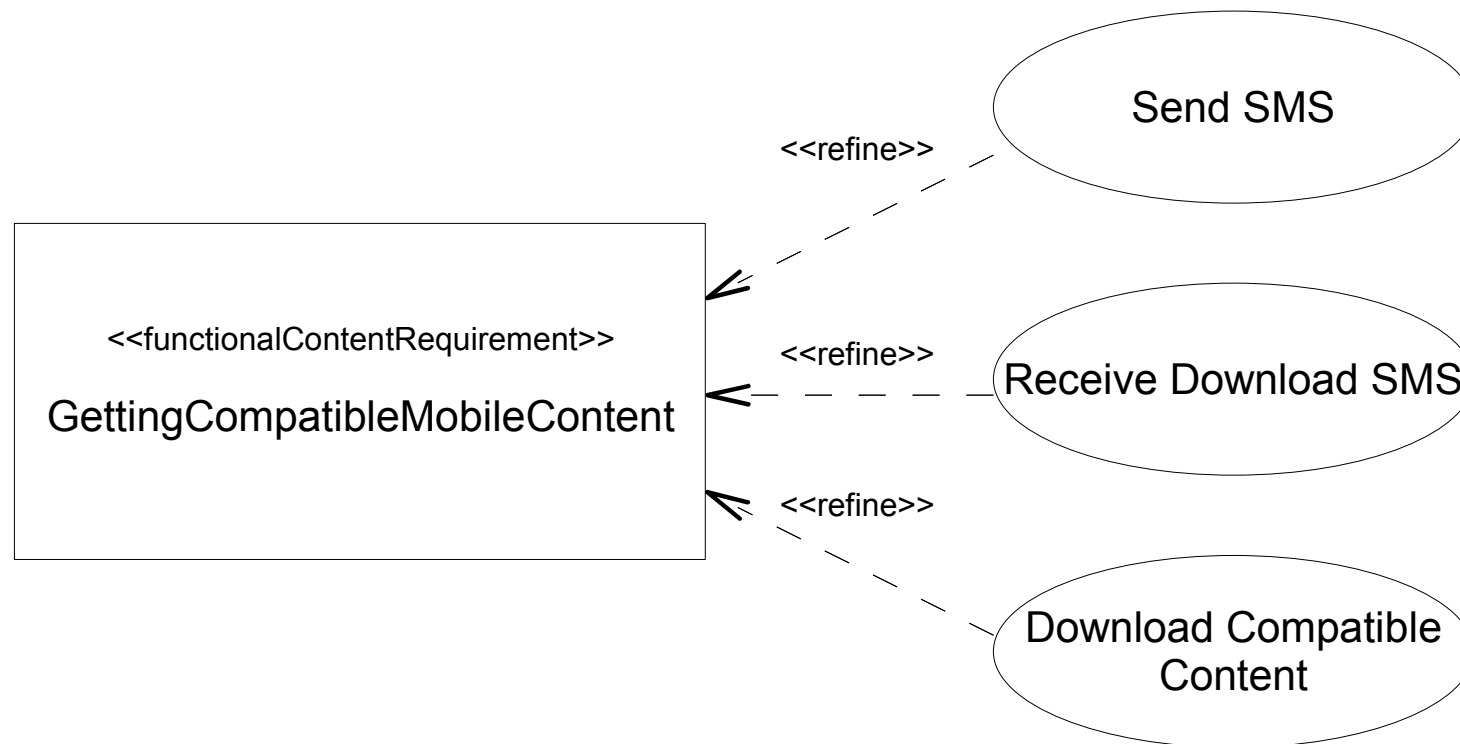
- ❖ “A refine relationship specifies that a model element describes the properties of a requirement in more detail.”

(System Engineering with SysML/UML by Tim Weilkiens)

The Refine Relationship

- ❖ We can refine a requirement using different types of elements.
- ❖ It is common to refine a requirement using use cases.

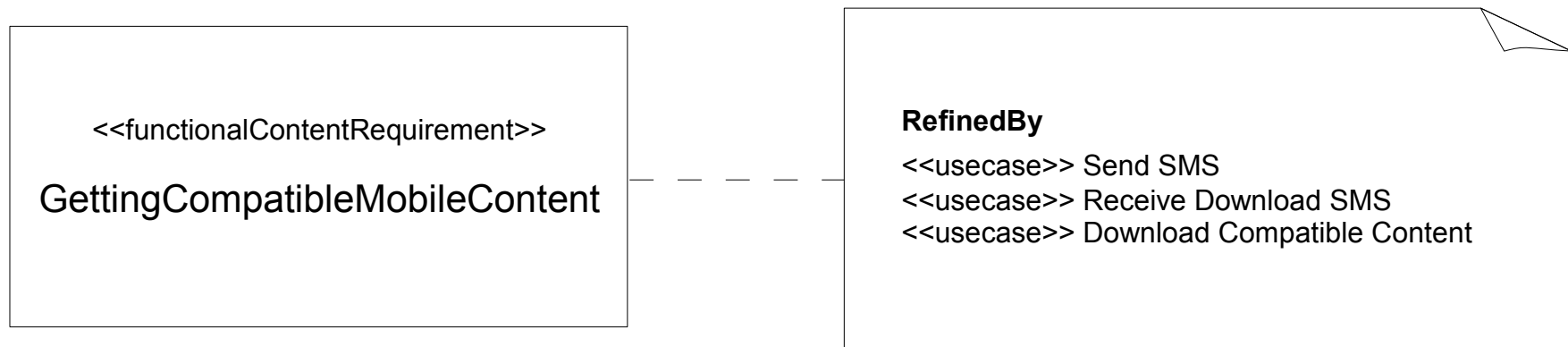
The Refine Relationship



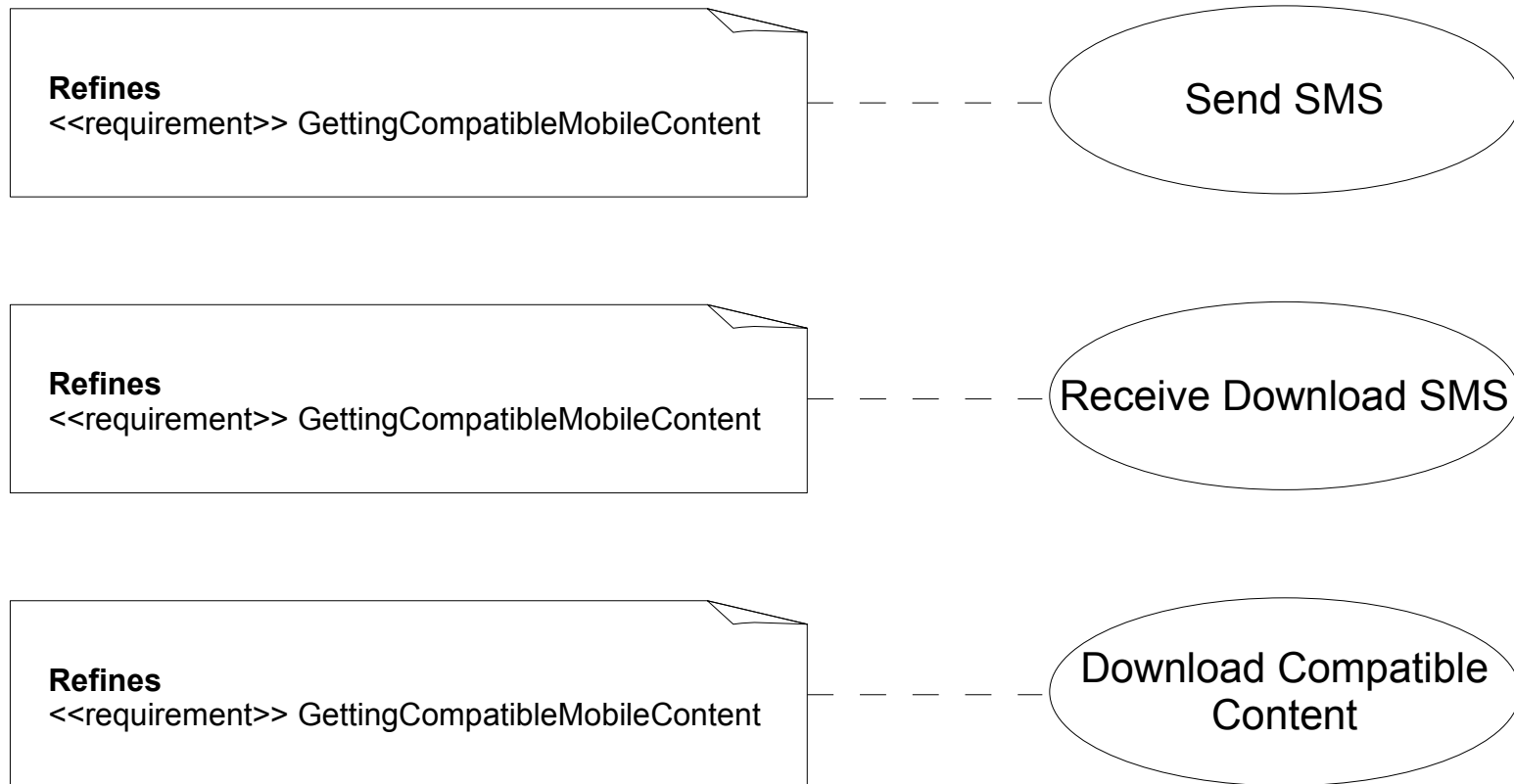
The Refine Relationship

- ❖ We can alternatively denote the refine relationship using the comment symbol.

The Refine Relationship



The Refine Relationship



The Trace Relationship

- ❖ “A trace relationship is a relationship between a requirement and an arbitrary model element; it describes a general relation for traceability reasons only.”

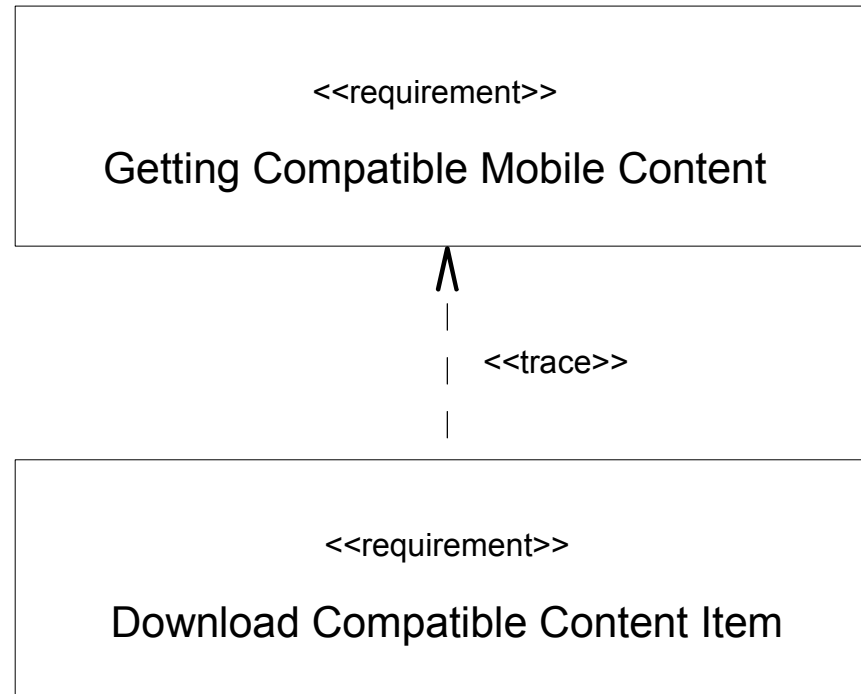
(System Engineering with SysML/UML by Tim Weilkiens)

The Trace Relationship

- ❖ The trace relationship is a very general one. It doesn't specify the nature of the relationship and serves to express a general relationship only.

The only purpose of this relationship is to assist us understanding the whole design by knowing of the general connection between the various elements in our model.

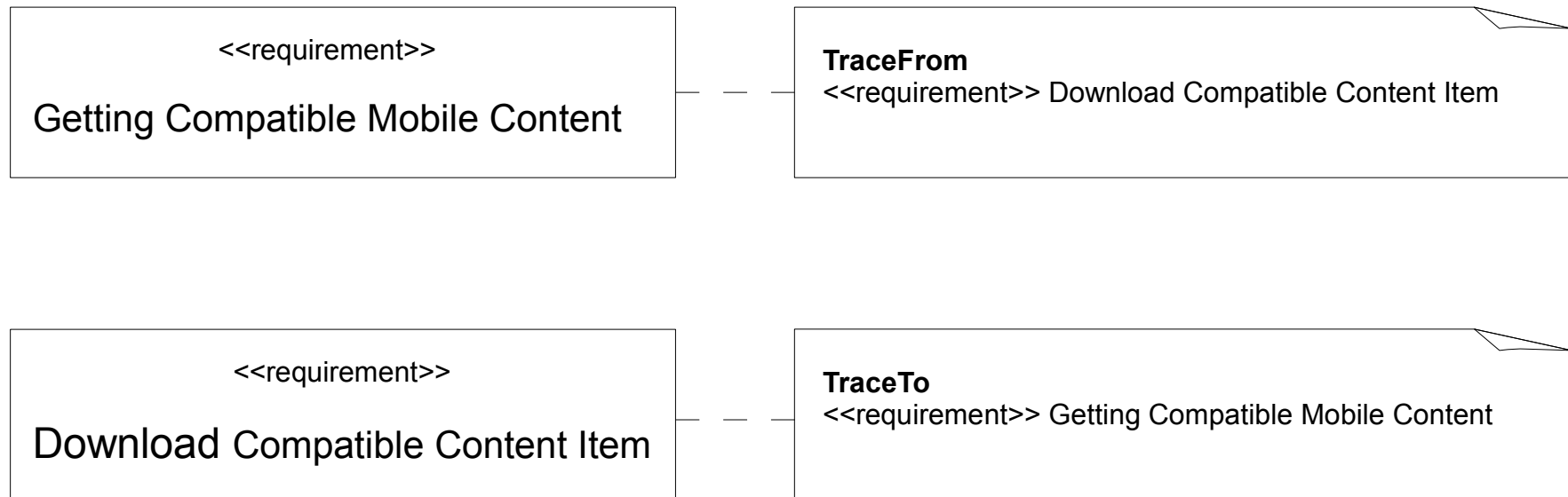
The Trace Relationship



The Trace Relationship

- ❖ Alternative notation used for showing a trace relationship includes the usage of the comment symbol together with one of the following keywords: “TraceTo” and “TraceFrom”.

The Trace Relationship



The Requirements Table Notation

- ❖ SysML allows representing the requirements using the table notation (in a textual form).

The Requirements Table Notation

ID	Name	Text
req100 1	Get Compatible Mobile Content	When user sends SMS to pay for a content item he should get the version compatible with his mobile telephone.
req100 2	Get Non Available Message	When user sends SMS to pay for a content title for which there isn't a compatible version for his mobile telephone he should get a proper message.
req100 3	Get Download URL Available for 30 Min	The download URL the user receives should be available for 30 min.

The Requirements Table Notation

- ❖ We can also create a table that shows the requirements and the relationships between them.

The Requirements Table Notation

ID	Name	Relation	ID	Name
req1002	Get Non Available Message	deriveReqt	req1001	Get Compatible Mobile Content
req1003	Get Download URL Available for 30 Min	deriveReqt	req1009	Prevent Piracy

Requirements Diagram

05/10/10

© 2008 Haim Michael. All Rights Reserved.

1

Introduction

- ❖ One of the most important steps in a system development is collecting the requirements and finalize each one of them. Skipping this phase or completing it in a neglected way might heart the whole system development process.
- ❖ SysML defines several elements that assist with describing and modeling the requirements.

Requirements Types

- ❖ There are different types of requirements (e.g. response time, size, specific functionality etc.).

Functional requirements are usually described using the use case diagrams. None functional requirements are usually described using the requirements diagram.

Requirement SysML Definition

- ❖ A requirement describes one or more properties or behaviors of a system that always have to be met. A requirement is a stereotype <<requirement>> of the UML element class.
(System Engineering with SysML/UML by Tim Weilkiens)

The Requirements Contract

- ❖ Each requirement describes a contract between the principals and those that design and develop the system.

The Requirement Notation

- ❖ Each requirement is represented using the UML class notation together with the <<requirement>> stereotype.
- ❖ Each requirement notation has a unique identifier (ID) and a descriptive text that describes it.

SysML doesn't specify any specific structure the ID should follow. The descriptive text can be a reference for an external resource such as a specific data record in a requirements management tool. Apart of these two attributes we cannot add operations or other attributes. Nevertheless, it is possible to extend the requirement in order to add more properties, such as 'priority'.

The Requirement Notation

<<requirement>>
W3C XHTML Standard
+id = "XHTML 1.0"
+text = "http://www.w3.org/TR/xhtml1/"

The Requirement Notation

- ❖ In terms of the model we have in the background of our requirement diagram, each class is considered as an abstract one.

It wouldn't make sense to create instances of a requirement class.

- ❖ Requirements cannot be generalized.

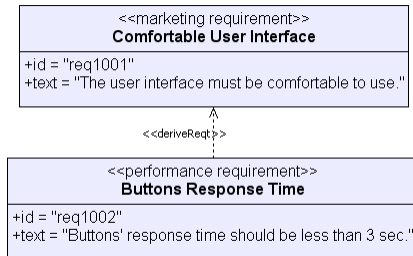
There is no sense in generalizing a requirement as it doesn't have any operation or attribute to inherit.

The Derive Relationship

- ❖ The derive requirement relationship describes a requirement that derives from another requirement. We reflect one requirement deriving from another by using the `<<deriveReq>>` stereotype together with the dependency dashed arrow pointing at the requirement from which the other derives.

(System Engineering with SysML/UML by Tim Weilkiens)

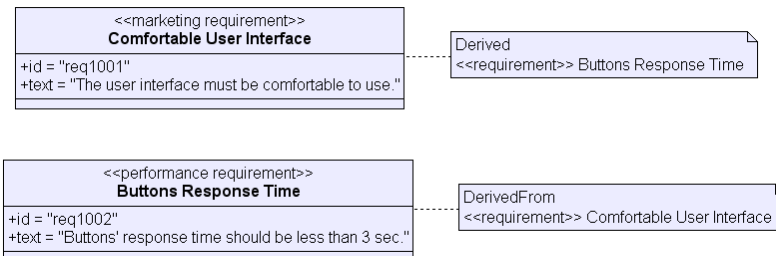
The Derive Relationship



The Derive Relationship

- ❖ We can show that one requirement derives from another by adding comments with either the “Derived” or the “DerivedFrom” titles.

The Derive Relationship



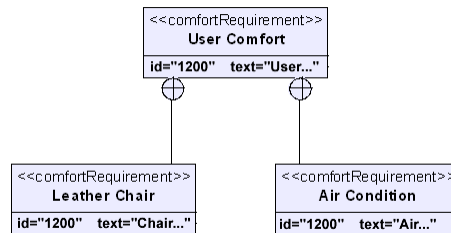
The Namespace Containment

- ❖ The namespace containment describes a case in which one requirement is contained within another.

(System Engineering with SysML/UML by Tim Weilkiens)

When the master requirement is satisfied when all of its sub requirements are satisfied we get a containment relationship between the two. Each requirement cannot contain requirements with the same name and each requirement cannot be part of more than one other requirement at the same time. The notation is the same notation used to represent inner classes.

The Namespace Containment



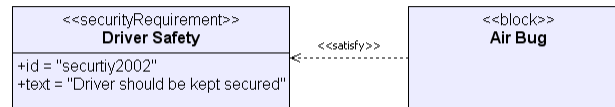
The Satisfy Relationship

- ❖ The satisfy relationship describes a design element that satisfies a requirement. It is a stereotype <<satisfy>> of the UML relationship realization.

(System Engineering with SysML/UML by Tim Weilkiens)

Each design element in our model satisfies a requirement (either directly or indirectly). We use a dashed arrow and the <<satisfy>> keyword.

The Satisfy Relationship



The Copy Relationship

- ❖ The copy relationship describes a requirement which is a copy of another requirement.

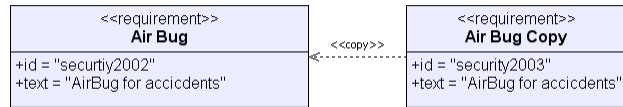
(System Engineering with SysML/UML by Tim Weilkiens)

When there is a need to reuse a requirement within another context we can use the <<copy>> relationship in order to enable that. This way we overcome the limit to have each requirement within one context at the most.

The Copy Relationship

- ❖ The copy we get is denoted as a dashed arrow pointing from the copy to the original and with the `<<copy>>` stereotype.
- ❖ The name and the ID might be different. The descriptive text MUST be identical.

The Copy Relationship



Test Cases

- ❖ “A test case is a flow that checks whether or not the system satisfies a requirement.”
(System Engineering with SysML/UML by Tim Weilkiens)
- ❖ The notation used to represent a test case is a simple rectangle with the name of the test case written within it together with the `<<testCase>>` stereotype.

Test Cases

<<testCase>>

Send Testing SMS

05/10/10

© 2008 Haim Michael. All Rights Reserved.

21

The Verify Relationship

- ❖ The verify relationship connects a test case with the requirement that is verified by that test case.
(System Engineering with SysML/UML by Tim Weilkiens)
- ❖ The verify relationship describes which test case tests each requirement.

The Verify Relationship

- ❖ The notation is a dashed arrow with the `<<verify>>` stereotype pointing at the requirement.
- ❖ We will use the `<<verify>>` relationship no matter whether the test verifies the requirement in a partial or a complete way.

The Verify Relationship

`<<testCase>>`
Send Testing SMS `<<verify>>` `<<functionalContentRequirement>>`
Getting Compatible Content

```
graph LR; A["<<testCase>>  
Send Testing SMS"] -.->|<<verify>>| B["<<functionalContentRequirement>>  
Getting Compatible Content"]
```

05/10/10

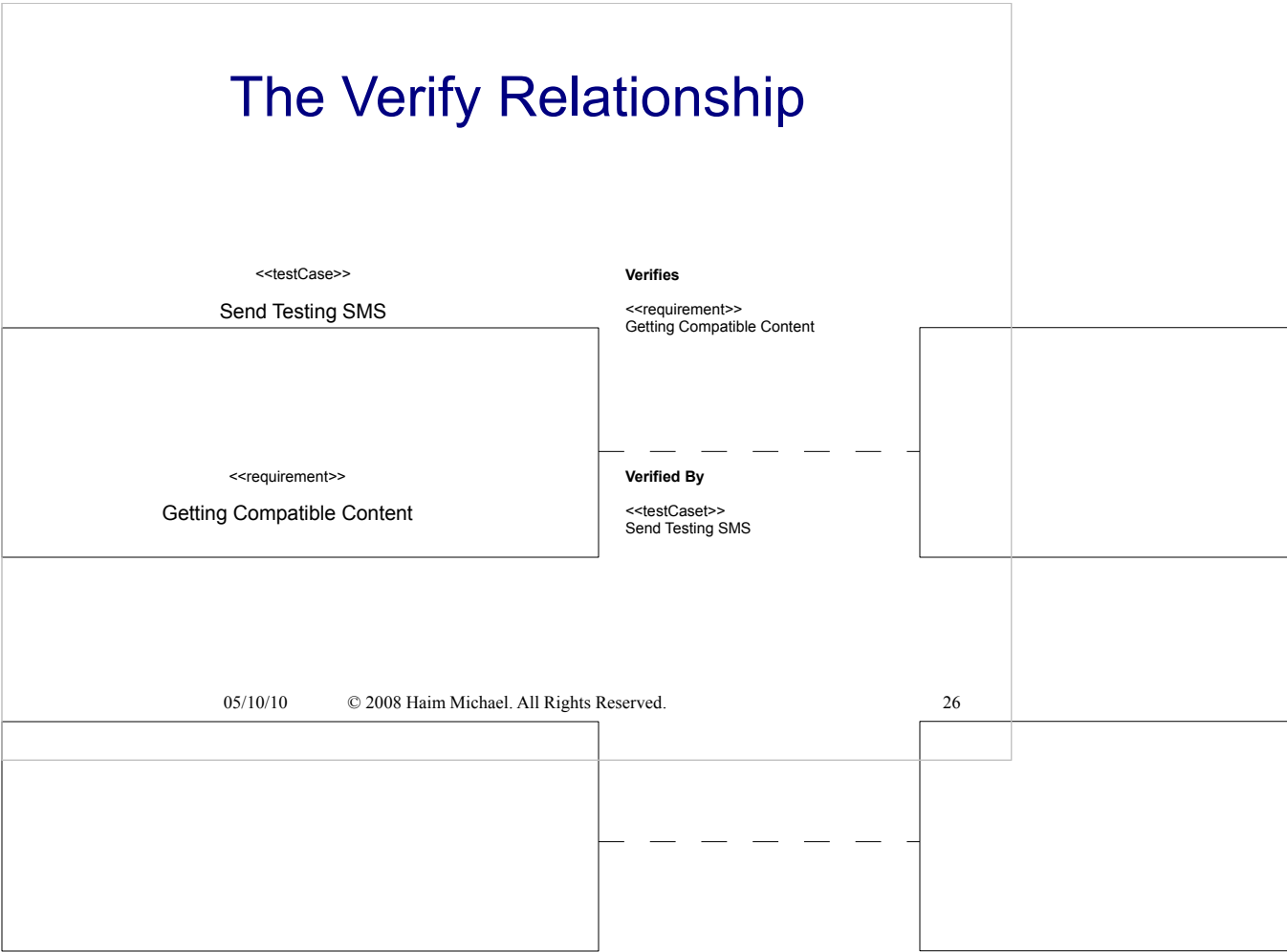
© 2008 Haim Michael. All Rights Reserved.

24

The Verify Relationship

- ❖ Alternative notation for showing the verify relationship includes the usage of the comment symbol.

The Verify Relationship



The Refine Relationship

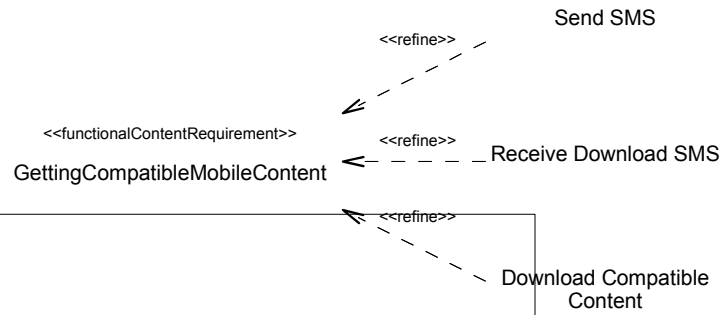
- ❖ “A refine relationship specifies that a model element describes the properties of a requirement in more detail.”

(System Engineering with SysML/UML by Tim Weilkiens)

The Refine Relationship

- ❖ We can refine a requirement using different types of elements.
- ❖ It is common to refine a requirement using use cases.

The Refine Relationship



05/10/10

© 2008 Haim Michael. All Rights Reserved.

29

The Refine Relationship

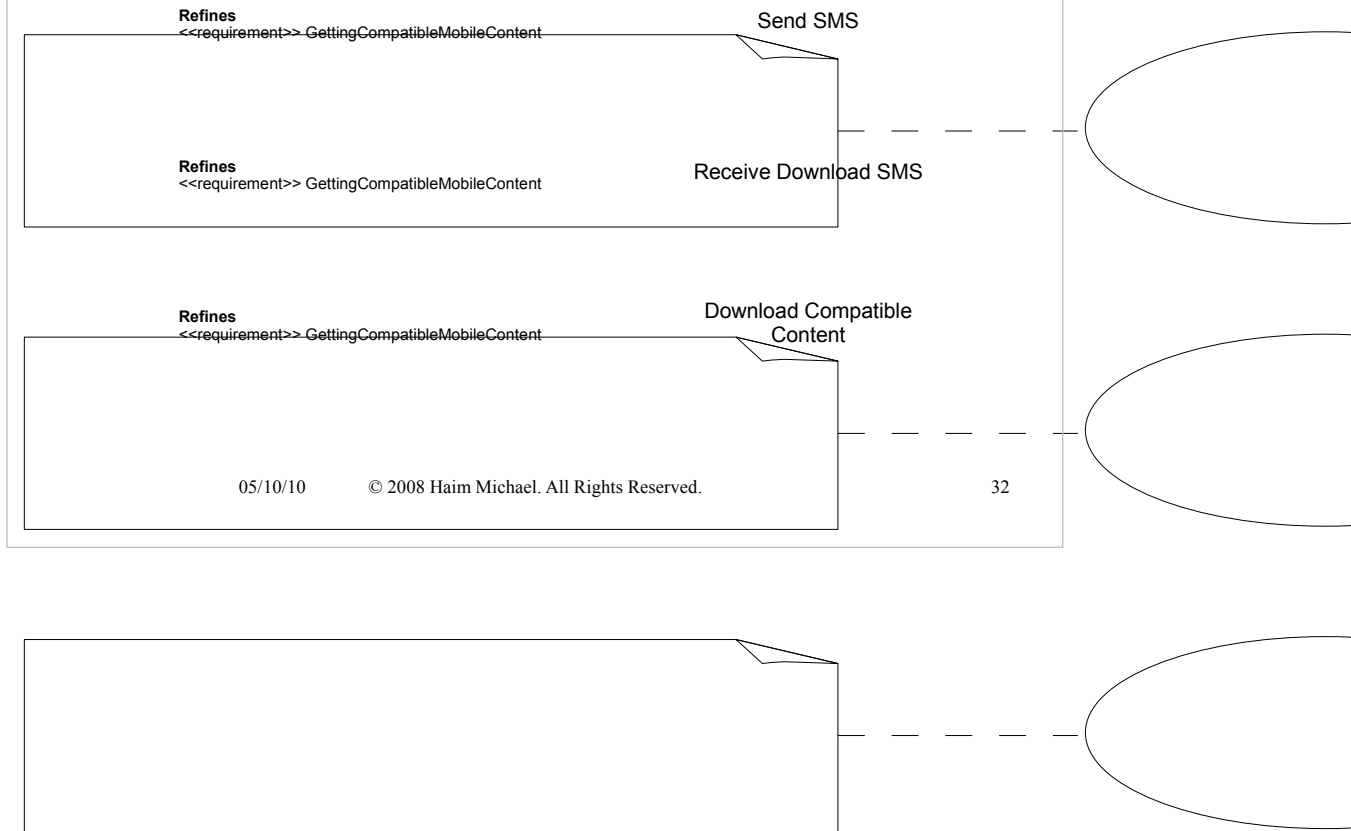
- ❖ We can alternatively denote the refine relationship using the comment symbol.

The Refine Relationship

<<functionalContentRequirement>>
GettingCompatibleMobileContent

RefinedBy
<<usecase>> Send SMS
<<usecase>> Receive Download SMS
<<usecase>> Download Compatible Content

The Refine Relationship



The Trace Relationship

- ❖ “A trace relationship is a relationship between a requirement and an arbitrary model element; it describes a general relation for traceability reasons only.”

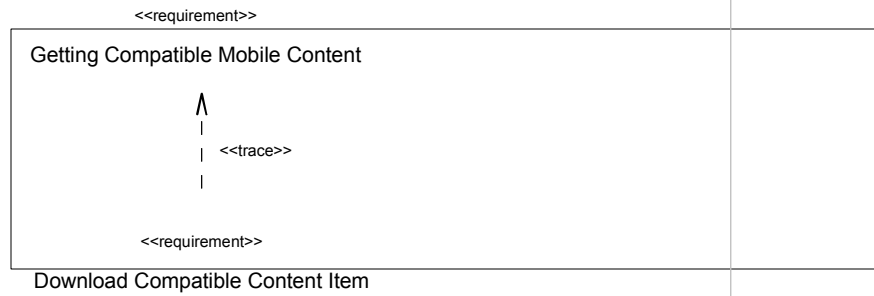
(System Engineering with SysML/UML by Tim Weilkiens)

The Trace Relationship

- ❖ The trace relationship is a very general one. It doesn't specify the nature of the relationship and serves to express a general relationship only.

The only purpose of this relationship is to assist us understanding the whole design by knowing of the general connection between the various elements in our model.

The Trace Relationship



05/10/10

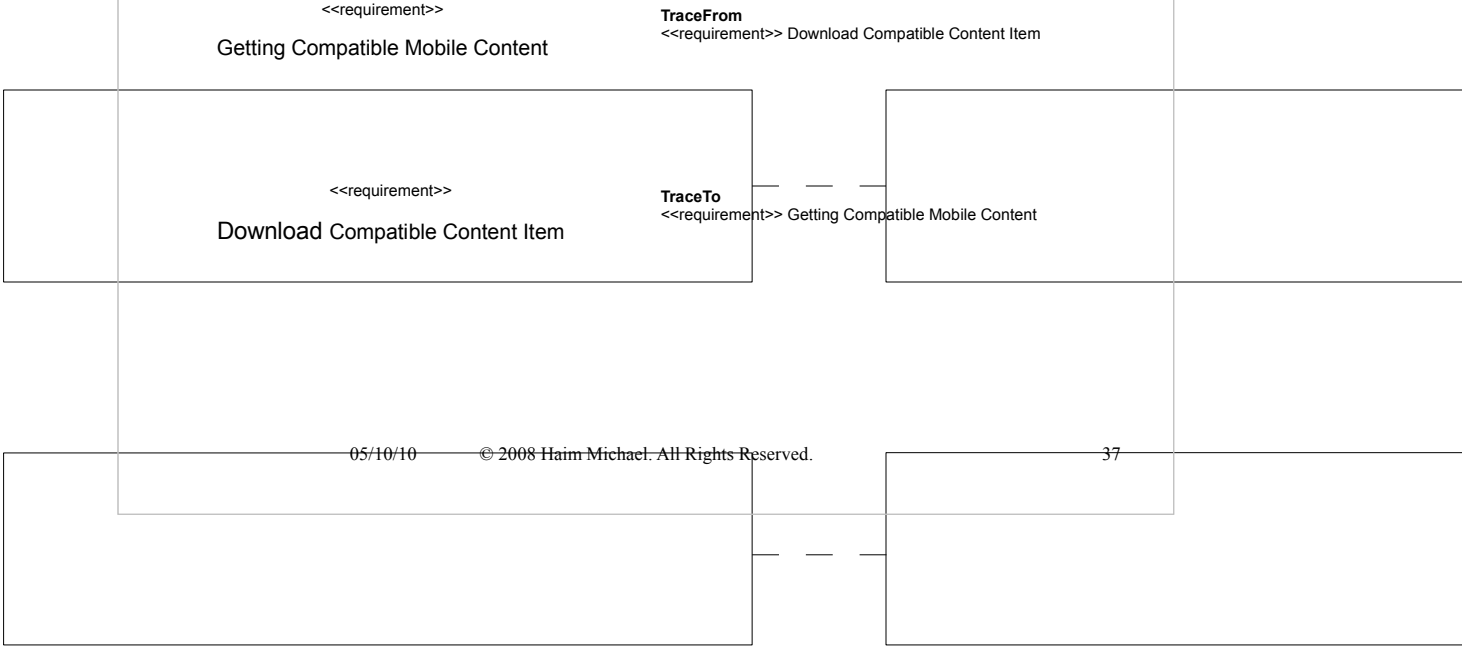
© 2008 Haim Michael. All Rights Reserved.

35

The Trace Relationship

- ❖ Alternative notation used for showing a trace relationship includes the usage of the comment symbol together with one of the following keywords: “TraceTo” and “TraceFrom”.

The Trace Relationship



The Requirements Table Notation

- ❖ SysML allows representing the requirements using the table notation (in a textual form).

The Requirements Table Notation

ID	Name	Text
req1001	Get Compatible Mobile Content	When user sends SMS to pay for a content item he should get the version compatible with his mobile telephone.
req1002	Get Non Available Message	When user sends SMS to pay for a content title for which there isn't a compatible version for his mobile telephone he should get a proper message.
req1003	Get Download URL Available for 30 Min	The download URL the user receives should be available for 30 min.

The Requirements Table Notation

- ❖ We can also create a table that shows the requirements and the relationships between them.

The Requirements Table Notation

ID	Name	Relation	ID	Name
req1002	Get Non Available Message	deriveReq	req1001	Get Compatible Mobile Content
req1003	Get Download URL Available for 30 Min	deriveReq	req1009	Prevent Piracy