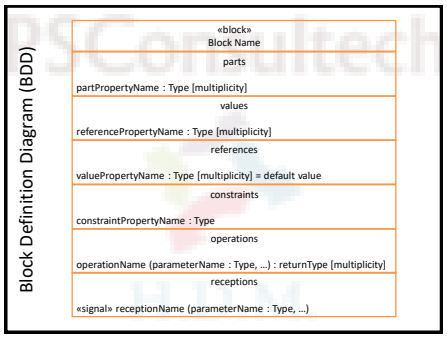
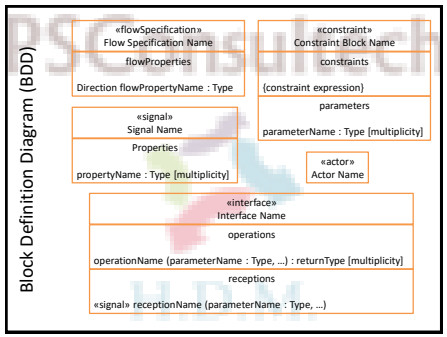
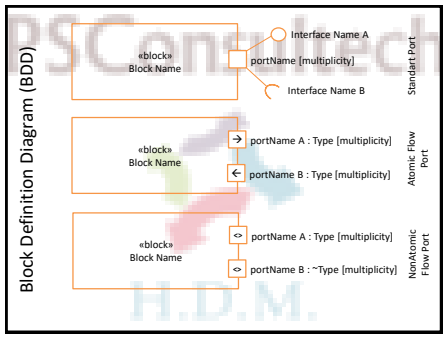
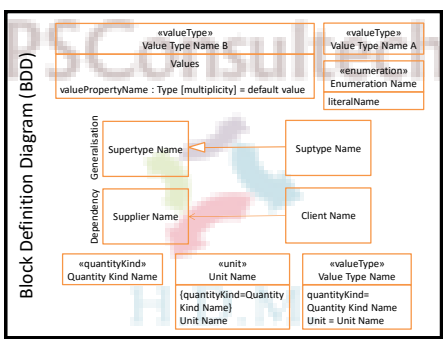
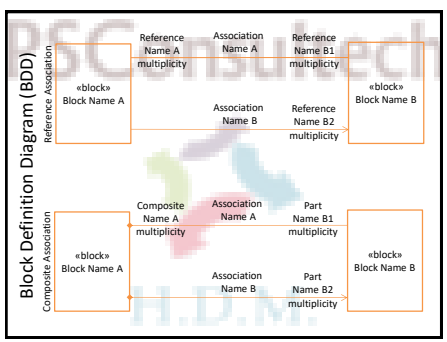
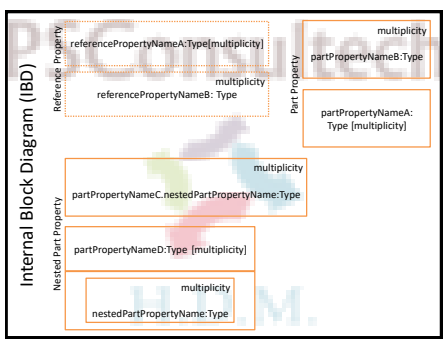

Block Definition Diagram (BDD)
«block»
Block Name
parts
partPropertyName : Type [multiplicity]
values
referencePropertyName : Type [multiplicity]
references
valuePropertyName : Type [multiplicity] = default value
constraints
constraintPropertyName : Type
operations
operationName (parameterName : Type, ...) : returnType [multiplicity]
receptions
«signal» receptionName (parameterName : Type, ...)

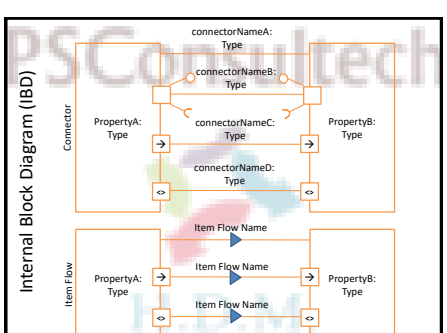

Block Definition Diagram (BDD)
«flowSpecification»
Flow Specification Name
flowProperties
Direction flowPropertyName : Type
«constraint»
Constraint Block Name
constraints
{constraint expression}
parameters
parameterName : Type [multiplicity]
«signal»
Signal Name
Properties
propertyName : Type [multiplicity]
«actor»
Actor Name
«interface»
Interface Name
operations
operationName (parameterName : Type, ...) : returnType [multiplicity]
receptions
«signal» receptionName (parameterName : Type, ...)

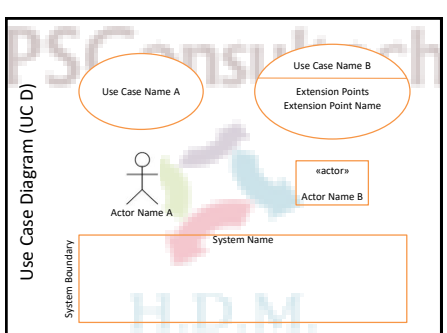

Block Definition Diagram (BDD)
«block»
Block Name
Interface Name A
portName [multiplicity]
Interface Name B
Standart Port
«block»
Block Name
portName A : Type [multiplicity]
portName B : Type [multiplicity]
Atomic Flow Port
«block»
Block Name
portName A : Type [multiplicity]
portName B : ~Type [multiplicity]
NonAtomic Flow Port

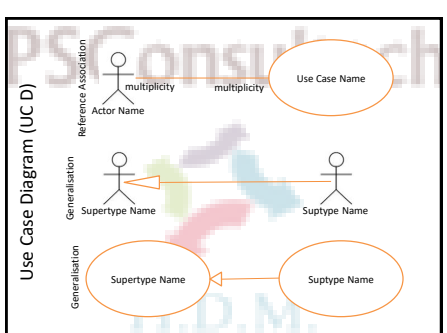

Block Definition Diagram (BDD)
«valueType»
Value Type Name B
Values
valuePropertyName : Type [multiplicity] = default value
«valueType»
Value Type Name A
«enumeration»
Enumeration Name
literalName
Generalisation
Supertype Name
Suptype Name
Dependency
Supplier Name
Client Name
«quantityKind»
Quantity Kind Name
«unit»
Unit Name
{quantityKind=Quantity Kind Name}
Unit Name
«valueType»
Value Type Name
quantityKind= Quantity Kind Name
Unit = Unit Name

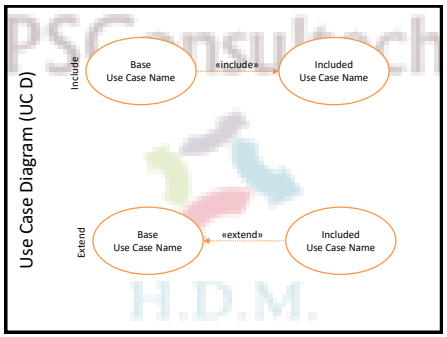

Block Definition Diagram (BDD)
Reference Association
«block»
Block Name A
Reference Name A multiplicity
Association Name A
Reference Name B1 multiplicity
«block»
Block Name B
Association Name B
Reference Name B2 multiplicity
Composite Association
«block»
Block Name A
Composite Name A multiplicity
Association Name A
Part Name B1 multiplicity
«block»
Block Name B
Association Name B
Part Name B2 multiplicity

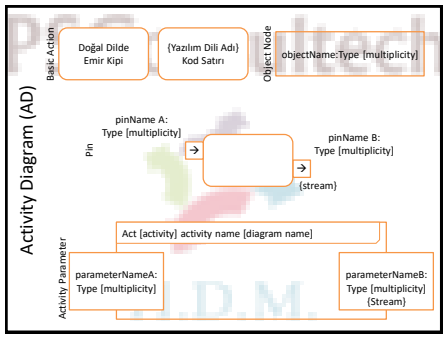

Internal Block Diagram (IBD)
Reference Property
referencePropertyNameA:Type[multiplicity]
multiplicity
referencePropertyNameB: Type
Part Property
multiplicity
partPropertyNameB:Type
partPropertyNameA: Type [multiplicity]
Nested Part Property
multiplicity
partPropertyNameC.nestedPartPropertyName:Type
partPropertyNameD:Type [multiplicity]
multiplicity
nestedPartPropertyName:Type

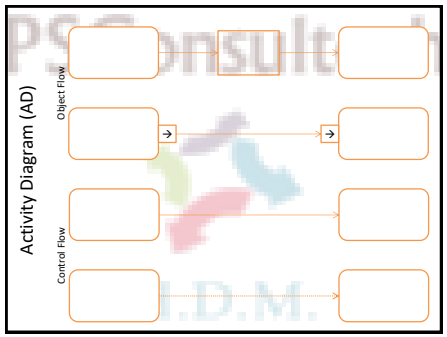

Internal Block Diagram (IBD)
connectorNameA: Type
connectorNameB: Type
Connector
PropertyA: Type
connectorNameC: Type
PropertyB: Type
connectorNameD: Type
Item Flow Name
Item Flow
PropertyA: Type
Item Flow Name
PropertyB: Type
Item Flow Name

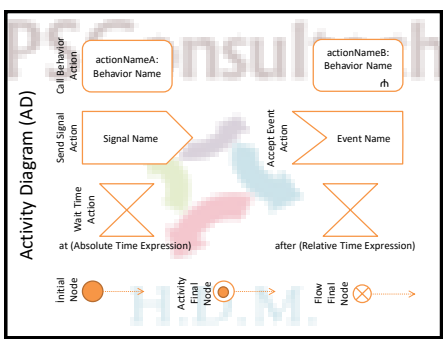

Use Case Diagram (UC D)
Use Case Name A
Use Case Name B
Extension Points
Extension Point Name
«actor»
Actor Name B
Actor Name A
System Name
System Boundary

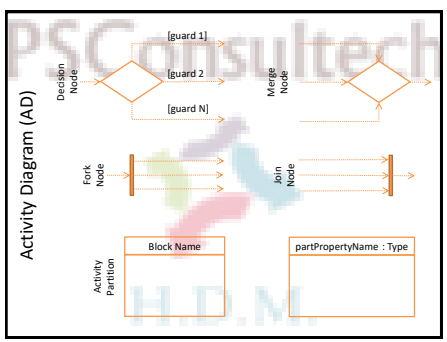

Use Case Diagram (UC D)
Reference Association
multiplicity
multiplicity
Actor Name
Use Case Name
Generalisation
Supertype Name
Suptype Name
Generalisation
Supertype Name
Suptype Name

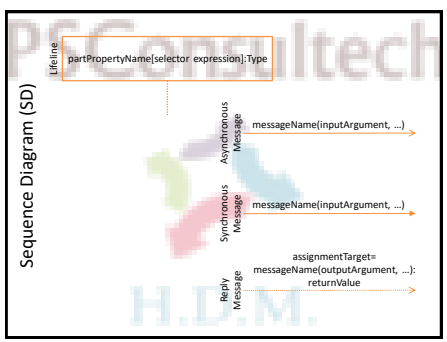

Use Case Diagram (UC D)
Include
Base Use Case Name
«include»
Included Use Case Name
Extend
Base Use Case Name
«extend»
Included Use Case Name

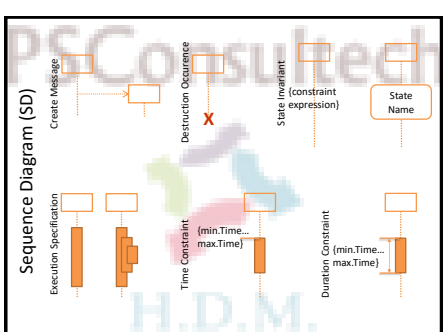

Activity Diagram (AD)
Basic Action
Doğal Dilde Emir Kipi
{Yazılım Dili Adı} Kod Satırı
Object Node
objectName:Type [multiplicity]
Pin
pinName A: Type [multiplicity]
pinName B: Type [multiplicity]
{stream}
Activity Parameter
Act [activity] activity name [diagram name]
parameterNameA: Type [multiplicity]
parameterNameB: Type [multiplicity] {Stream}

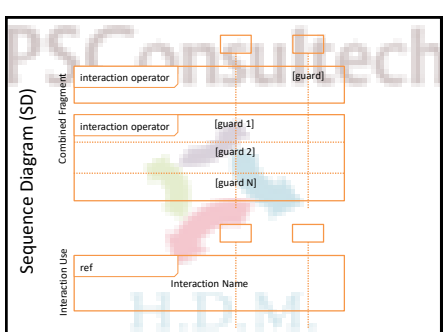

Activity Diagram (AD)
Object Flow
Control Flow

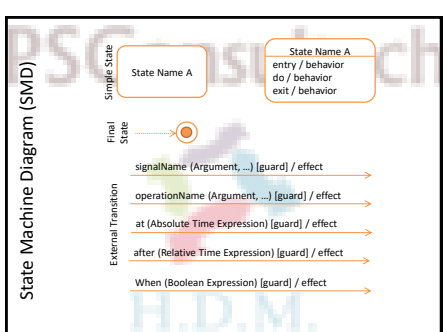

Activity Diagram (AD)
Call Behavior Action
actionNameA: Behavior Name
actionNameB: Behavior Name
Send Signal Action
Signal Name
Accept Event Action
Event Name
Wait Time Action
at (Absolute Time Expression)
after (Relative Time Expression)
Initial Node
Activity Final Node
Flow Final Node

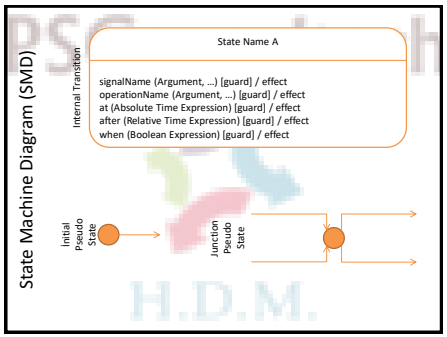

Activity Diagram (AD)
Decision Node
[guard 1]
[guard 2
[guard N]
Merge Node
Fork Node
Join Node
Activity Partition
Block Name
partPropertyName : Type

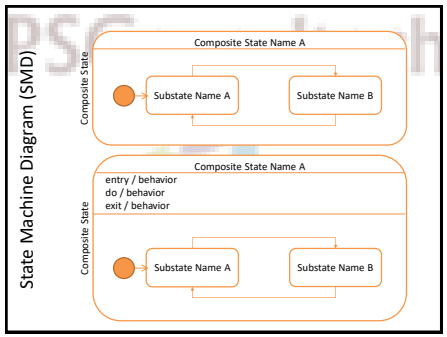

Sequence Diagram (SD)
Lifeline
partPropertyName[selector expression]:Type
Asynchronous Message
messageName(inputArgument, ...)
Synchronous Message
messageName(inputArgument, ...)
Reply Message
assignmentTarget= messageName(outputArgument, ...): returnValue

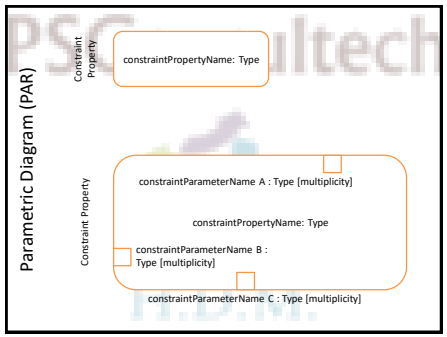

Sequence Diagram (SD)
Create Message
Destruction Occurence
X
State Invariant
{constraint expression}
State Name
Execution Specification
Time Constraint
{min.Time... max.Time}
Duration Constraint
{min.Time... max.Time}

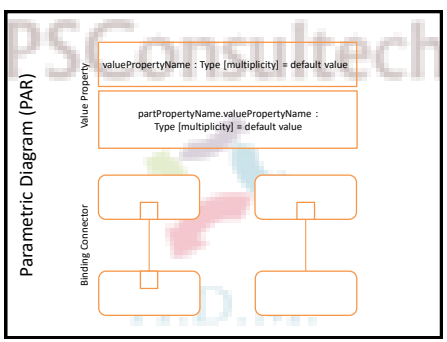

Sequence Diagram (SD)
Combined Fragment
interaction operator
[guard]
interaction operator
[guard 1]
[guard 2]
[guard N]
Interaction Use
ref
Interaction Name

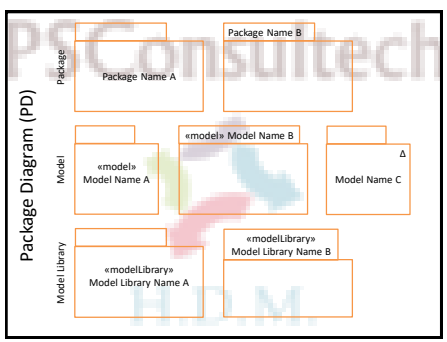

State Machine Diagram (SMD)
Simple State
State Name A
State Name A
entry / behavior
do / behavior
exit / behavior
Final State
External Transition
signalName (Argument, ...) [guard] / effect
operationName (Argument, ...) [guard] / effect
at (Absolute Time Expression) [guard] / effect
after (Relative Time Expression) [guard] / effect
When (Boolean Expression) [guard] / effect

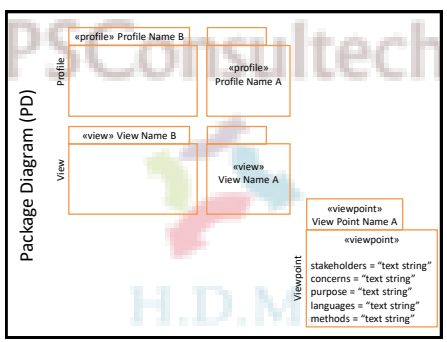

State Machine Diagram (SMD)
Internal Transition
State Name A
signalName (Argument, ...) [guard] / effect
operationName (Argument, ...) [guard] / effect
at (Absolute Time Expression) [guard] / effect
after (Relative Time Expression) [guard] / effect
when (Boolean Expression) [guard] / effect
Initial Pseudo State
Junction Pseudo State

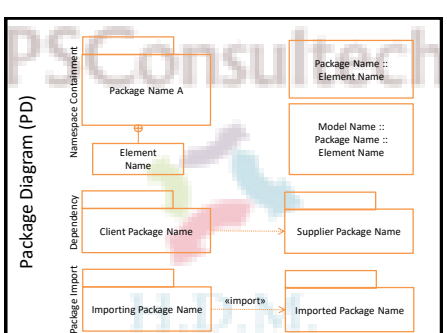

State Machine Diagram (SMD)
Composite State
Composite State Name A
Substate Name A
Substate Name B
Composite State
Composite State Name A
entry / behavior
do / behavior
exit / behavior
Substate Name A
Substate Name B

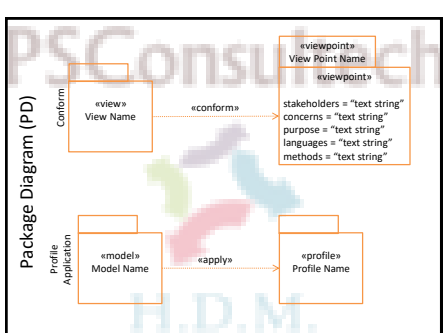

Parametric Diagram (PAR)
Constraint Property
constraintPropertyName: Type
Constraint Property
constraintParameterName A : Type [multiplicity]
constraintPropertyName: Type
constraintParameterName B :
Type [multiplicity]
constraintParameterName C : Type [multiplicity]

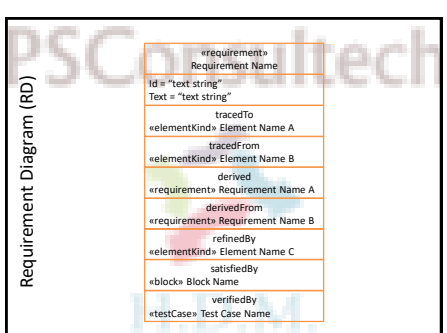

Parametric Diagram (PAR)
Value Property
valuePropertyName : Type [multiplicity] = default value
partPropertyName.valuePropertyName :
Type [multiplicity] = default value
Binding Connector

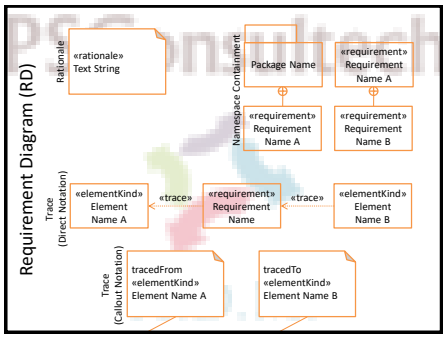

Package Diagram (PD)
Package
Package Name A
Package Name B
Model
«model»
Model Name A
«model» Model Name B
Model Name C
Model Library
«modelLibrary»
Model Library Name A
«modelLibrary»
Model Library Name B

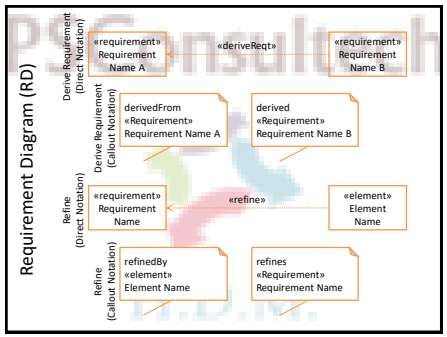

Package Diagram (PD)
Profile
«profile» Profile Name B
«profile»
Profile Name A
View
«view» View Name B
«view»
View Name A
Viewpoint
«viewpoint»
View Point Name A
«viewpoint»
stakeholders = "text string"
concerns = "text string"
purpose = "text string"
languages = "text string"
methods = "text string"

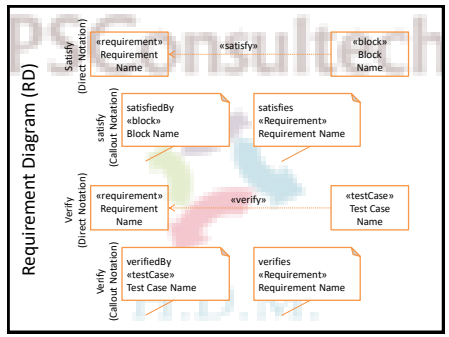

Package Diagram (PD)
Namespace Containment
Package Name A
Element
Name
Package Name ::
Element Name
Model Name ::
Package Name ::
Element Name
Dependency
Client Package Name
Supplier Package Name
Package Import
Importing Package Name
«import»
Imported Package Name

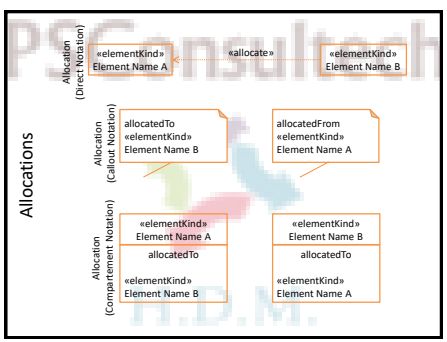

Package Diagram (PD)
Conform
«view»
View Name
«conform»
«viewpoint»
View Point Name
«viewpoint»
stakeholders = "text string"
concerns = "text string"
purpose = "text string"
languages = "text string"
methods = "text string"
Profile Application
«model»
Model Name
«apply»
«profile»
Profile Name

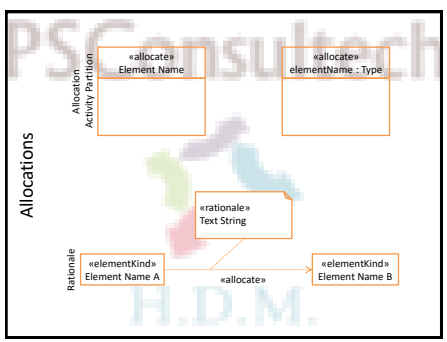

Requirement Diagram (RD)
«requirement»
Requirement Name
Id = "text string"
Text = "text string"
tracedTo
«elementKind» Element Name A
tracedFrom
«elementKind» Element Name B
derived
«requirement» Requirement Name A
derivedFrom
«requirement» Requirement Name B
refinedBy
«elementKind» Element Name C
satisfiedBy
«block» Block Name
verifiedBy
«testCase» Test Case Name


Requirement Diagram (RD)
Rationale
«rationale»
Text String
Namespace Containment
Package Name
«requirement»
Requirement
Name A
«requirement»
Requirement
Name A
«requirement»
Requirement
Name B
Trace
(Direct Notation)
«elementKind»
Element
Name A
«trace»
«requirement»
Requirement
Name
«trace»
«elementKind»
Element
Name B
Trace
(Callout Notation)
tracedFrom
«elementKind»
Element Name A
tracedTo
«elementKind»
Element Name B


Requirement Diagram (RD)
Derive Requirement
(Direct Notation)
«requirement»
Requirement
Name A
«deriveReqt»
«requirement»
Requirement
Name B
Derive Requirement
(Callout Notation)
derivedFrom
«Requirement»
Requirement Name A
derived
«Requirement»
Requirement Name B
Refine
(Direct Notation)
«requirement»
Requirement
Name
«refine»
«element»
Element
Name
Refine
(Callout Notation)
refinedBy
«element»
Element Name
refines
«Requirement»
Requirement Name


Requirement Diagram (RD)
Satisfy
(Direct Notation)
«requirement»
Requirement
Name
«satisfy»
«block»
Block
Name
satisfy
(Callout Notation)
satisfiedBy
«block»
Block Name
satisfies
«Requirement»
Requirement Name
Verify
(Direct Notation)
«requirement»
Requirement
Name
«verify»
«testCase»
Test Case
Name
Verify
(Callout Notation)
verifiedBy
«testCase»
Test Case Name
verifies
«Requirement»
Requirement Name


Allocations
Allocation
(Direct Notation)
«elementKind»
Element Name A
«allocate»
«elementKind»
Element Name B
Allocation
(Callout Notation)
allocatedTo
«elementKind»
Element Name B
allocatedFrom
«elementKind»
Element Name A
Allocation
(Compartement Notation)
«elementKind»
Element Name A
allocatedTo
«elementKind»
Element Name B
«elementKind»
Element Name B
allocatedTo
«elementKind»
Element Name A


Allocations
Allocation
Activity Partition
«allocate»
Element Name
«allocate»
elementName : Type
«rationale»
Text String
Rationale
«elementKind»
Element Name A
«allocate»
«elementKind»
Element Name B