Appendix D describes three metamodels for the Observer pattern: a PSM metamodel based on the Eiffel platform, a PSM metamodel based on a Java platform and a ISM metamodel based on the Java platform.

D.1. EIFFEL-PSM OBSERVER METAMODEL

Description of the Metaclasses

AssocEndEffectiveObserver

Description
This end-association connects an association ObserverSubject, of which is member, with an EffectiveObserver.

Generalizations

- AssociationEnd (from PSM-Eiffel)
Appendix D

Figure 1. Eiffel PSM observer metamodel

Associations

- association: ObserverSubject [1] It denotes the association of which this end-association is member. It redefines Property::association.
- participant: EffectiveObserver [1] It denotes the class that participates in the association.

Constraints

[1] It has a multiplicity n1..n2 (n1 >= 0 and n2 >= 1) self.lower >= 0 and self.upper >= 1
AssocEndEffectiveSubject

**Description**
It connects an association ObserverSubject, of which is member, with a class EffectiveSubject.

**Generalizations**
- AssociationEnd (from PSM-Eiffel)

**Associations**
- association: ObserverSubject [1] It denotes the association of which the association-end is member. It redefines Property::association.
- participant: EffectiveSubject [1] It denotes the class that participates in the association.

**Constraints**
1. It has a multiplicity n1..n2 (n1>= 0 and n2>=1) self.lower >= 0 and self.upper >=1
2. It must be navigable self.isNavigable()

**Additional operations**
1. isNavigable denotes whether the association-end is navigable. The association-end is member of a binary association then, to be navigable, it must be own part of a class. isNavigable(): Boolean isNavigable() = not self.class -> isEmpty()

AssocEndObserver

**Description**
It connects an association SubjectObserver, of which is member, with a class Observer.

**Generalizations**
- AssociationEnd (from PSM-Eiffel)

**Associations**
- association: SubjectObserver [1] It denotes the association of which this association-end is member. It redefines Property::association.
- participant: Observer [1] It denotes the class that participates in the association.

**Constraints**
1. It has a multiplicity n1..n2 (n1>= 0 and n2>=1) self.lower >= 0 and self.upper >= 1
2. It must be navigable. self.isNavigable()
Appendix D

AssocEndSubject

Description
It connects an association SubjectObserver, of which is member, with a class Subject.

Generalizations

- AssociationEnd (from PSM-Eiffel)

Associations

- association: SubjectObserver[1] It denotes the association of which this association-end is member. It redefines Property::association.
- participant: Subject [1] It denotes the class that participates in the association.

Constraints

[1] It has a multiplicity n1..n2 (n1>= 0 and n2>=1). self.lower >= 0 and self.upper >=1

Attach

Description
It defines a routine that is declared by Subject.

Generalizations

- Routine (from PSM-Eiffel)

Associations

- subject: Subject [1] It denotes the class that declares this routine. It redefines Routine::class.

Constraints

[1] This routine changes the state of the subject. not self.isQuery
[2] This routine has a non-empty set of parameters being one of them an input parameter (direction = #in) whose type is Observer. self.ownedParameter -> notEmpty () and self.ownedParameter -> select (par | par.direction = #in and par.type = oclIsKindOf (Observer)) -> size () = 1
[3] Its visibility must be public. self.visibility = #public
Appendix D

Detach

**Description**
It defines a routine that is declared by Subject.

**Generalizations**

- Routine (from PSM-Eiffel)

**Associations**

- subject: Subject [1] It denotes the class that is declared by this routine. It redefines Routine::class.

**Constraints**

[1] This routine changes the state of the subject. not self.isQuery
[2] It has a non-empty set of parameters being one of them an input parameter whose type is Observer. self. 
   ownedParameter -> notEmpty () and self.ownedParameter -> select (par | par.direction = #in 
   and par.type = oclIsKindOf (Observer)) -> size () = 1
[3] Its visibility must be public. self.visibility = #public

EffectiveObserver

**Description**
This metaclass specifies the features that a class with the behavior of an effective observer must have in the model of the pattern Observer.

**Generalizations**

- EiffelClass (from PSM-Eiffel)

**Associations**

- assocEndEffectiveObserver: AssocEndEffectiveObserver[1] It denotes the association-end of the association ObserverSubject in which this classifier participates.
- relationshipObserver:RelationshipObserver[1] It denotes a generalization where EffectiveObserver takes the role of heir. It is a subset of Classifier::generalization.

**Constraints**

[1] Instances of effective observers should not be a deferred class. not self.isDeferred
EffectiveSubject

Description
This metaclass specifies the features that must have a class taking the role of effective subject in the model of the Observer pattern.

Generalizations

• EiffelClass (from PSM-Eiffel)

Associations

• assocEndEffectiveSubject:AssocEndEffectiveSubject [1] It denotes the association-end of the association ObserverSubject in which this classifier participates.
• getState: GetState [1..*] Every instance of EffectiveSubject must have one or more operation instances of GetState. They can be own or inherited. It is a subset of NameSpace::member.
• relationshipSubject:RelationshipSubject [1] It denotes a generalization where EffectiveSubject takes the role of heir. It is a subset of Classifier::generalization.
• setState: SetState [1..*] Every instance of EffectiveSubject must have one or more operation instances of SetState. They can be own or inherited. It is a subset of NameSpace::member.
• state: Attribute [1..*] It specifies a non-empty set that contains all attributes of EffectiveSubject. They can be own or inherited. It is a subset of NameSpace::member.

Constraints

[1] Instances of effective subjects should not be a deferred class. not self.isDeferred

GetState

Description
It defines a routine member of EffectiveSubject. It specifies a service that can be required from another object.

Generalizations

• Routine (from PSM-Eiffel)

Associations
No additional associations.

Constraints

[1] It is not a deferred routine and does not change the state of the subject not self.isDeferred and self.isQuery
The state should return to the subject, therefore within the set of arguments must have at least one parameter whose address is out or return. self.ownedParameter -> notEmpty () and self.ownedParameter -> select (par | par.direction = #return or par.direction = #out) -> size () >= 1

Its visibility must be public. self.visibility = #public

Notify

Description
It defines a routine that is declared by Subject.

Generalizations

• Routine (from PSM-Eiffel)

Associations

• Subject: Subject [1] It denotes the class that is declared by this routine. It redefines Routine::class.

Constraints

[1] This routine does not change the subject state. self.isQuery
[2] Its visibility must be public. self.visibility = #public

Observer

Description
This metaclasse specifies the features that must have every class taking the role of observer in the model of the Observer pattern in the Eiffel platform.

Generalizations

• EiffelClass (from PSM-Eiffel)

Associations

• assocEndObserver: AssocEndObserver [1] It denotes the association-end of the association SubjectObserver in which this class participates.
• update: Update [1..*] Every instance of Observer must have at least one routine instance of Update. It is a subset of EiffelClass::ownedRoutine.

Constraints
No additional constraints.
Appendix D

ObserverSubject

Description
This metaclass specifies a binary association between instances of EffectiveObserver and EffectiveSubject.

Generalizations

• Association (from Kernel)

Associations

• assocEndEffectiveObserver: AssocEndEffectiveObserver [1] It represents a connection with the class EffectiveObserver. It is a subset of Association::memberEnd.
• assocEndEffectiveSubject: AssocEndEffectiveSubject [1] It represents a connection with the class EffectiveSubject. It is a subset of Association::memberEnd.

Constraints

[1] It has two association-ends. self.memberEnd -> size () =2

RelationshipObserver

Description
This class specifies the inheritance relation (Generalization) between an observer (Observer) and an effective observer (EffectiveObserver) in the model of the Observer pattern.

Generalizations

• Generalization (from Kernel)

Associations

• heir: EffectiveObserver [1] It denotes the element that takes the role of heir in the relation. It redefines Generalization::specific.
• parent: Observer [1] It denotes the element that takes the role of parent in the relation. It redefines Generalization::general.

Constraints
No additional constraints.
RelationshipSubject

Description
This metaclass specifies an inheritance relation (Generalization) between a subject (Subject) and an effective subject (EffectiveSubject) in the model of the Observer pattern.

Generalizations
• Generalization (from Kernel)

Associations
• heir: EffectiveSubject [1] It denotes the element that takes the role of heir in the relation. It redefines Generalization::specific.
• parent: Subject [1] It denotes the element that takes the role of parent in the relation. It redefines Generalization::general.

Constraints
No additional constraints.

SetState

Descriptions
It defines a routine member of EffectiveSubject. It specifies a service that can be required from another object.

Generalizations
• Routine (from PSM-Eiffel)

Associations
No additional associations.

Constraints
[1] It is not a deferred routine and modifies the state of the subset. not self.isDeferred and not self.isQuery
[2] The set of arguments is non-empty and one of them must be an input parameter. self.ownedParameter ->notEmpty () and self.ownedParameter ->select (par | par.direction= #in) -> size () >=1
[3] Its visibility must be public. self.visibility = #public
Appendix D

Subject

Description
This metaclass specifies the features that must have an Eiffel class taking the role of subject in the model of the Observer pattern in the Eiffel platform.

Generalizations

• EiffelClass (from PSM-Eiffel)

Associations

• assocEndSubject: AssocEndSubject [1] It denotes the association-end of the association SubjectObserver in which this class participates.
• attach: Attach [1..*] Every instance of Subject must have at least a routine instance of Attach. It is a subset of Subconjunto de EiffelClass::ownedRoutine.
• detach: Detach [1..*] Every instance of Subject must have at least a routine instance of Detach. It is a subset of EiffelClass::ownedRoutine.
• notify: Notify[1..*] Every instance of Subject must have at least a routine instance of Notify. It is a subset of EiffelClass::ownedRoutine.

Constraints
No additional constraints.

SubjectObserver

Description
This metaclass specifies a binary association between the instances Subject and Observer.

Generalizations

• Association (from Kernel)

Associations

• assocEndObserver: AssocEndObserver [1] It represents a connection with the class Observer. It is a subset of Association::memberEnd.
• assocEndSubject: AssocEndSubject [1] It represents a connection with the class Subject. It is a subset of Association::memberEnd.

Constraints

[1] It has two association-ends. self.memberEnd -> size () =2
Appendix D

Update

Description
It defines the routine that is declared by the Observer specifying the required services by another object.

Generalizations

- Routine (from PSM-Eiffel)

Associations

- observer: Observer [1] It denotes the class that declares this operation. It is a subset of Routine::ownedRoutine.

Constraints

[1] It is a routine that does not change the observer state. self.isQuery
[2] Its visibility must be public. self.visibility = #public

D.2. JAVA-PSM OBSERVER METAMODEL

Description of Metaclasses

AssocEndConcreteObserver

Description
This association-end connects an association ObserverSubject, of which is member, with a ConcreteObserver.

Generalizations

- AssociationEnd (from PSM-Java)

Associations

- association: ObserverSubject [1] It denotes the association of which this association-end is member. It redefines Property::association
- participant: ConcreteObserver [1] It denotes the classifier that participates in the association.
**Constraints**

[1] It has a multiplicity n1..n2 (n1 >= 0 and n2 >= 1). self.lower >= 0 and self.upper > 0

**AssocEndConcreteSubject**

**Description**
This association-end connects an association ObserverSubject, of which is member, with a class ConcreteSubject.
Figure 3. Java PSM observer metamodel: Abstract subject: Operations

Figure 4. Java PSM observer metamodel: Abstract observer: Operations
Generalizations

• AssociationEnd (from PSM-Java)

Associations

• association: ObserverSubject [1] It denotes the association of which this association-end is member. It redefines Property::association.
• participant: Concretesubject [1] It denotes the classifier that participates in the association.

Constraints

[1] It has a multiplicity n1..n2 (n1 >= 0 and n2 >= 1). self.lower >= 0 and self.upper > 0
[2] It is navigable. self.isNavigable ()

Additional operations

[3] The observer operation isNavigable determines whether this association-end is navigable. Due to it is member of a binary association, to be navigable must be an own association-end of a class. isNavigable(): Boolean isNavigable() = not self.class -> isEmpty ()

AssocEndObserver

Description
It connects an association SubjectObserver, of which is member, with a class Observer.
Generalizations

- AssociationEnd (from PSM-Java)

Associations

- association: SubjectObserver [1] It denotes the association of which this association-end is member. It redefines Property::association.
- participant: Observer [1] It denotes the classifier that participates in the association.

Constraints

[1] It has a multiplicity n1..n2 (n1 >= 0 and n2 >=1). self.lower >= 0 and self.upper > 0
[2] It must be navigable self.isNavigable ()

AssocEndSubject

Description
It connects an association SubjectObserver, of which is member, with a class Subject.

Generalizations

- AssociationEnd (from PSM-Java)

Associations

- association: SubjectObserver [1] It denotes the association of which this association-end is member. It redefines Property::association.
- participant: Subject [1] It denotes the classifier that participates in the association.

Constraints

[1] It has a multiplicity n1..n2 (n1 >= 0 and n2 >=1). self.lower >= 0 and self.upper > 0

Attach

Description
It defines a method that is declared by a subject.

Generalizations

- Method (from PSM-Java)
Appendix D

Associations

- classSubject: ClassSubject [0..1] It denotes the class declaring this method. It redefines JavaOperation::class.
- interfaceSubject: InterfaceSubject [0..1] It denotes the interface declaring this method. It redefines Method::interface.

Constraints

[1] This method changes the subject state. not self.isQuery
[2] This method has a non-empty set of parameters, being one of them an input parameter whose type is Observer. self.ownedParameter -> notEmpty () and self.ownedParameter -> select (param | param.direction= #in and param.type = oclIsKindOf(Observer)) -> size () = 1
[3] Its visibility must be public. self.visibility = #public

ClassObserver

Description
A metaclass ClassObserver specifies the features that must have a Java class taking the role of observer in the model of the Observer pattern.

Generalizations

- Observer, JavaClass (from PSM-Java)

Associations

- update: Update [1..*] Every instance of ClassObserver must have at least a method instance of Update. It is a subset of JavaClass::javaOperation.

Constraints
No additional constraints.

ClassSubject

Description
This metaclass specifies the features that must have a Java class taking the role of subject in the model of the pattern Observer.

Generalizations

- Subject, JavaClass (from PSM-Java)
Appendix D

Associations

- **attach**: Attach [1..*] Every instance of ClassSubject has at least a method instance of Attach. It is a subset of JavaClass:: javaOperation.
- **detach**: Detach [1..*] Every instance of ClassSubject has at least a method instance of Detach. It is a subset of JavaClass:: javaOperation.
- **notify**: Notify[1..*] Every instance of ClassSubject has at least a method instance of Notify. It is a subset of JavaClass:: javaOperation.

Constraints
No additional constraints.

ConcreteObserver

Description
This metaclass specifies the features that must have a Java class with the behavior of a concrete observer in the model of the pattern Observer.

Generalizations
- JavaClass (from PSM-Java)

Associations

- **assocEndConcreteObserver**: AssocEndConcreteObserver [1] It denotes the association-end of the association ObserverSubject in which this classifier participates.
- **generalizationObserver**: GeneralizationObserver [0..1] It denotes a generalization where ConcreteObserver takes the role of child (specific). It redefines Classifier::generalization.
- **interfaceRealizationObserver**: InterfaceRealizationObserver [0..1] It denotes an interface realization where ConcreteObserver takes the role of the classifier implementing the contract (implementingClassifier). It is a subset of BehavioredClassifier::interfaceRealization.

Constraints

[1] An instance of a concrete observer can not be an abstract class, not self.isAbstract
[2] If an instance of a concrete observer participates in an interface realization, then it must be a BehavioredClassifier. self.interfaceRealizationObserver -> notEmpty () implies self.oclIsKindOf (BehavioredClassifier)

ConcreteSubject

Description
This metaclass specifies the features that must have a class taking the role of subject in the model of the Observer pattern.
Appendix D

Generalizations

- **JavaClass** (from PSM-Java)

Associations

- **assocEndConcreteSubject:** AssocEndConcreteSubject [1] It denotes the association-end of the association ObserverSubject in which this classifier participates.
- **generalizationSubject:** GeneralizationSubject [0..1] It denotes a generalization where ConcreteSubject takes the role of child (specific). It redefines Classifier::generalization.
- **getState:** GetState [1..*] Every instance of ConcreteSubject must have one or more method instances of GetState. They may be own or inherited. It is a subset of NameSpace::member.
- **interfaceRealizationSubject:** InterfaceRealizationSubject [0..1] It denotes an interface realization where ConcreteSubject takes the role of the classifier implementing the contract (implementing Classifier). It is a subset of BehavioredClassifier::interfaceRealization.
- **setState:** GetState [1..*] Every instance of ConcreteSubject must have one or more method instances of SetState. They may be own or inherited. It is a subset of NameSpace::member.
- **state:** Field [1..*] It specifies a non-empty set of all attributes of ConcreteSubject. They may be own or inherited. It is a subset of NameSpace::member.

Constraints

[1] An instance of the concrete subject can not be an abstract class. **not** self.isAbstract
[2] If an instance of a concrete subject participates in an interface realization, then it must be a BehavioredClassifier. self.interfaceRealizationSubject -> notEmpty () **implies** self.ooclIsKindOf (BehavioredClassifier))

Detach

Description
It defines a method that is declared by a subject.

Generalizaciones

- **Method** (from PSM-Java)

Associations

- **classSubject:** ClassSubject [0..1] It denotes the class that declares this method. It redefines JavaOperation::class.
- **interfaceSubject:** InterfaceSubject [0..1] It denotes the interface that declares this method. It redefines Method::interface.
Appendix D

Constraints

[1] This method changes the subject state. **not** self.isQuery
[2] This method has a non-empty set of parameters being one of them an input parameter of type Observer. self.ownedParameter -> notEmpty () **and** self.ownedParameter -> select (param | param. direction= #in **and** param.type = oclIsKindOf (Observer)) -> size() = 1
[3] Its visibility must be public. self.visibility = #public

GeneralizationObserver

**Description**
This metaclass specifies a generalization between an observer (ClassObserver) and a concrete observer (ConcreteObserver) in the model of the pattern **Observer**.

**Generalizations**

- Generalization (from Kernel)

**Associations**

- classObserver: ClassObserver [1] It denotes the general element of this relation. It redefines Generalization::general.
- concreteObserver: ConcreteObserver [1] It denotes the specific element of this relation. It redefines Generalization::specific.

**Constraints**
No additional constraints.

GeneralizationSubject

**Description**
This metaclass specifies a generalization between a subject (ClassSubject) and a concrete subject (ConcreteSubject) in the model of the **Observer** pattern.

**Generalizations**

- Generalization (from Kernel)

**Associations**

- classSubject: ClassSubject [1] It denotes the general element of this relation. It redefines Generalization::general.
- concreteSubject: ConcreteSubject [1] It denotes the specific element of this relation. It redefines Generalization::specific.
Appendix D

Constraints
No additional restrictions.

GetState

Description
It defines a method that is member of ConcretetSubject. It specifies a service that can be required from another object.

Generalizations

- Method (from PSM-Java)

Associations
No additional associations.

Constraints

[1] It is an observer and concrete method. self.isQuery and not self.isAbstract
[2] Because it has to return the state of the subject, the set of parameters should not be empty and at least, must have one of them whose direction is out or return. self.ownedParameter -> notEmpty () and self.ownedParameter ->select (par | par.direction= #return or par.direction = #out) -> size () >=1
[3] Its visibility must be public. self.visibility = #public

InterfaceObserver

Description
The metaclass InterfaceObserver specifies the features that must have a Java interface taking the role of abstract observer in the model of the Observer pattern.

Generalizations

- Observer, JavaInterface (from PSM-Java)

Associations

- update: Update [1..*] Every instance of InterfaceObserver must have at least one operation instance of Update. It is a subset of JavaInterface::method.

Constraints
No additional constraints.
InterfaceSubject

**Description**
This metaclass specifies the features that must have a Java interface taking the role of abstract subject in the model of the pattern *Observer*.

**Generalizations**
- Subject, JavaInterface (from PSM-Java)

**Associations**
- attach: Attach [1..*] Every instance of InterfaceSubject must have at least a method instance of Attach. It is a subset of JavaInterface::method.
- detach: Detach [1..*] Every instance of InterfaceSubject must have at least a method instance of Detach. It is a subset of JavaInterface::method.
- notify: Notify [1..*] Every instance of InterfaceSubject must have at least a method instance of Notify. It is a subset of JavaInterface::method.

**Constraints**
No additional constraints.

InterfaceRealizationObserver

**Description**
This metaclass specifies an interface realization between an abstract observer (InterfaceObserver) and a concrete observer (ConcreteObserver) in the model of the pattern *Observer*.

**Generalizations**
- InterfaceRealization (from Kernel)

**Associations**
- concreteObserver: ConcreteObserver [1] It denotes the element implementing the contract in this relation. It redefines InterfaceRealization::implementingClassifier.
- interfaceObserver: InterfaceObserver [1] It denotes the element that defines the contract in this relation. It redefines InterfaceRealization::contract.

**Constraints**
No additional constraints.
Appendix D

InterfaceRealizationSubject

Description
This metaclass specifies an interface realization between an abstract subject (InterfaceSubject) and a concrete subject (ConcreteSubject) in the model of the pattern Observer.

Generalizations
- InterfaceRealization (from Kernel)

Associations
- concreteSubject: ConcreteSubject [1] It denotes the element implementing the contract in this relation. It redefines InterfaceRealization::implementingClassifier.
- interfaceSubject: InterfaceSubject [1] It denotes the element that defines the contract in this relation. It redefines InterfaceRealization::contract.

Constraints
No additional constraints.

Notify

Description
It defines a method that is declared by the subject.

Generalizations
- Method (from PSM-Java)

Associations
- classSubject: ClassSubject [0..1] It denotes the class that declares the method. It redefines JavaOperation::class.
- interfaceSubject: InterfaceSubject [0..1] It denotes the interface that declares this method. It redefines Method::interface.

Constraints
[1] It is a method that changes the subject state. self.isQuery
[2] Its visibility is public. self.visibility = #public
Appendix D

Observer

Description
An observer is a specialized classifier that specifies the features of observers in the model of the pattern Observer. It is an abstract metaclass.

Generalizations
- Classifier (from Kernel)

Associations
- assocEndObserver:AssocEndObserver [0..1] It denotes the association-end of the association SubjectObserver in which this classifier participates.

Constraints
No additional constraints.

ObserverSubject

Description
This metaclass specifies a binary association between two instances of Observer and Subject.

Generalizations
- Association (from Kernel)

Associations
- assocEndConcreteObserver:AssocEndConcreteObserver [1] It represents a connection with the classifier ConcreteObserver. It is a subset of Association::memberEnd.
- assocEndConcreteSubject:AssocEndConcreteSubject [1] It represents a connection with the classifier ConcreteSubject. It is a subset of Association::memberEnd.

Constraints
[1] It has two association-ends. self.memberEnd->size() = 2

SetState

Description
It defines an operation that is member of ConcreteSubject. It specifies a service that can be required from another object.
Generalizaciones

Method (from PSM-Java)

Associations
No additional associations.

Constraints

1. It is a concrete method that modify the subject state. not self.isAbstract and not self.isQuery
2. It has a non-empty set of parameters and one of them, at least must be an input parameter. self. OwnedParameter ->notEmpty () and self.OwnedParameter -> select (param | param.direction = #in) -> size () >= 1
3. Its visibility must be public. self.visibility = #public

Subject

Description
This metaclass is a specialized classifier that specifies the features that must have instances taking the role of subject in the model of the pattern Observer. It is an abstract metaclass.

Generalizations

- Classifier (from Kernel)

Associations

- assocEndSubject: AssocEndSubject [0..1] It denotes the association-end of the association SubjectObserver in which the classifier participates.

Constraints
No additional constraints.

SubjectObserver

Description
This metaclass specifies a binary association between two classifiers: Subject y Observer.

Generalizations

- Association (from Kernel)
Appendix D

Associations

- assocEndObserver: AssocEndObserver [1] It represents a connection with the classifier Observer. It is a subset of Association::memberEnd.
- assocEndSubject: AssocEndSubject [1] It represents a connection with the classifier Subject. It is a subset of Association::memberEnd.

Constraints

[1] It has two association-ends. self.memberEnd -> size () = 2

Update

Description
It defines a method that is declared by an observer. This method specifies a service that can be required by another object.

Generalizations

- Method (from PSM-Java)

Associations

- classObserver: ClassObserver [0..1] It denotes the class that declares this operation. It redefines JavaOperation::class.
- interfaceObserver: InterfaceObserver [0..1] It denotes the interface that declares this operation. It redefines Method::interface.

Constraints

[1] It is a method that does not change the observer state. self.isQuery
[2] Its visibility must be public. self.visibility = #public

D. 3. JAVA-ISM OBSERVER METAMODEL

Description of the Metaclasses

AddLink

Description
It defines a method that is declared by an instance of the class SubjectObserverAssociation.
Appendix D

Figure 6. Java ISM observer metamodel

Generalizations

- Method (from ISM-Java)

Associations

- subjectObserverAssociation:SubjectObserverAssociation [1] It denotes the class that declares this method. It redefines JavaOperation::class.
Figure 7. Java ISM observer metamodel: Abstract subject: Operations

Figure 8. Java ISM observer metamodel: Abstract observer: Operations
Appendix D

Figure 9. Java ISM observer metamodel: Concrete subject: Operations and attributes

Figure 10. Java ISM observer metamodel: SubjectObserverAssociation: Operations

Constraints

[1] This method changes the state of the instance defining it. **not** self.isQuery

[2] It has a non-empty set of parameters being one of them an input parameter. **self.parameter** -> notEmpty () **and** self. parameter -> select (param | param.direction = #in **and** param.type = oclIsKindOf (Observer)) -> size () = 1

[3] Its visibility must be public. **self.visibility** = #public
Appendix D

Attach

Description
It defines a method that is declared by a subject.

Generalizations

- Method (from ISM-Java)

Associations

- classSubject: ClassSubject [0..1] It denotes the class that declares this method. It redefines JavaOperation::class.
- interfaceSubject: InterfaceSubject [0..1] It denotes the interface that declares this method. It redefines Method::interface.

Constraints

[1] This method changes the state of the subject. not self.isQuery
[2] It has a non-empty set of parameters, one of them is an input parameter of type Observer. self. parameter -> notEmpty () and self. parameter -> select (param | param.direction = #in and param. type = oclIsKindOf (Observer)) -> size () = 1
[3] Its visibility must be public. self.visibility = #public
[4] If the subject that declares this routine has a reference to a class SubjectObserverAssociation, then this routine will delegate this task to this class, by invoking to a routine that is an instance of AddLink. not self.subject.subjectObserverReference -> isEmpty implies self.invokedRoutine -> exists (r | r.oclIsTypeOf (AddLink))

ClassObserver

Description
A metaclass ClassObserver specifies the features of a class Java taking the role of observer in the model of a pattern Observer.

Generalizations

- Observer, JavaClass (from ISM-Java)

Associations

- update: Update [1..*] Every instance of ClassObserver must have at least a method instance of Update. It is a subset of JavaClass::javaOperation.
Appendix D

Constraints
No additional constraints.

ClassSubject

Description
This metaclass specifies the features that must have a Java class taking the role of subject in the model of the pattern Observer.

Generalizations

- Subject, JavaClass (from ISM-Java)

Associations

- attach: Attach [1..*] Every instance of ClassSubject has at least a method instance of Attach. It is a subset of JavaClass::javaOperation.
- detach: Detach [1..*] Every instance of ClassSubject has at least a method instance of Detach. It is a subset of JavaClass::javaOperation
- notify: Notify [1..*] Every instance of ClassSubject has at least a method instance of Notify. It is a subset of JavaClass::javaOperation.
- observerReference: ObserverReference [0..1] It denotes the attribute that allows the subject to maintain a reference to its observers. It is a subset of JavaClass::field.
- subjectObserverReference:SubjectObserverReference [0..1] It denotes the attribute, which is a reference to a class that maintains the relation subject-observers. It is a subset of JavaClass::field.

Constraints
No additional constraints.

ConcreteObserver

Description
This metaclass specifies the features that must have a Java class with the behavior of a concrete observer in the model of the pattern Observer.

Generalizations

- JavaClass (de ISM-Java)

Associations

- generalizationObserver: GeneralizationObserver [0..1] It denotes a generalization where ConcreteObserver takes the role of child (specific). It redefines Classifier::generalization.
• interfaceRealizationObserver: InterfaceRealizationObserver [0..1] It denotes an interface realization where ConcreteObserver takes the role of the classifier implementing the contract. It is a subset of BehavioredClassifier::interfaceRealization.

• subjectReference: SubjectReference [0..1] It denotes a reference to subjects that are observed by the observer. It is a subset of JavaClass::field.

Constraints

[1] An instance of a concrete observer can not be an abstract class. not self.isAbstract

[2] If an instance of a concrete observer participates in an interface realization, then it must be a BehavioredClassifier. self.interfaceRealizationObserver -> notEmpty () implies self.superClass -> exists (c | c.ooclIsTypeOf (BehavioredClassifier))

ConcreteSubject

Description
This metaclass specifies the features that must have a class taking the role of concrete subject in the model of the pattern Observer.

Generalizations

• JavaClass (from ISM-Java)

Associations

• generalizationSubject: GeneralizationSubject [0..1] It denotes a generalization where ConcreteSubject takes the role of child (specific). It redefines Classifier::generalization.

• getState: GetState [1..*] Every instance of ConcreteSubject must have one or more method instances of GetState. They can be own or inherited. It is a subset of NameSpace::member.

• interfaceRealizationSubject: InterfaceRealization [0..1] It denotes an interface realization where ConcreteSubject takes the role of the classifier implementing the contract (implementingClassifier). It is a subset of BehavioredClassifier::interfaceRealization.

• observerReference: ObserverReference [0..1] It denotes the attribute that allows the subject to maintain a reference to its observers. It is a subset of JavaClass::field.

• setState: GetState [1..*] Every instance of ConcreteSubject must have one or more method instances of SetState. They can be own or inherited. It is a subset of NameSpace::member.

• state: Property [1..*] It specifies a non-empty set of all attributes of ConcreteSubject. They can be own or inherited. It is a subset of NameSpace::member.

• subjectObserverReference: SubjectObserverReference [0..1] It denotes the attribute, which is a reference to a class maintaining the relation subject-observers. It is a subset of JavaClass::field.

Constraints

[1] An instance of a concrete subject can not be an abstract class. not self.isAbstract
Appendix D

[2] If an instance of a concrete subject participates in an interface realization, then it must be a BehavioredClassifier.

\[
\text{self.interfaceRealizationSubject} \rightarrow \text{notEmpty} \implies \text{self.superClass} \rightarrow \exists (c | c.\text{oclIsTypeOf}(\text{BehavioredClassifier}))
\]

[3] If an instance of a concrete subject is subclass of ClassSubject, it will inherit the field ObserverReference or the field subjectObserverReference, therefore it does not need to declare any references to its observers. On the contrary, if it implements the interface InterfaceSubject, then it must declare a field whose type is ObserverReference or SubjectObserverReference to maintain information of its observers. 

\[
\text{not self.generalizationSubject} \rightarrow \text{isEmpty} \implies \text{self.observerReference} \rightarrow \text{isEmpty} \land \text{self.subjectObserverReference} \rightarrow \text{isEmpty} \land \text{not self.interfaceRealizationSubject} \rightarrow \text{isEmpty} \implies \text{not self.observerReference} \rightarrow \text{isEmpty} \lor \text{not self.observerReference} \rightarrow \text{isEmpty}
\]

Detach

Description
It defines a method that is declared by a subject.

Generalizations

- Method (from ISM-Java)

Associations

- classSubject: ClassSubject [0..1] It denotes the class that declares this method. It redefines JavaOperation::class.
- interfaceSubject: InterfaceSubject [0..1] It denotes the interface that declares this method. It redefines Method::interface.

Constraints

[1] This method changes the subject state. not self.isQuery
[2] It has a non-empty set of parameters being one of them an input parameter whose type is Observer.

\[
\text{self.parameter} \rightarrow \text{notEmpty} \land \text{self.parameter} \rightarrow \exists (\text{param} | \text{param}.\text{direction} = \#\text{in} \land \text{param}.\text{type} = \text{oclIsKindOf}(\text{Observer})) \rightarrow \text{size}() = 1
\]
[3] Its visibility must be public. self.visibility = #public
[4] If the subject that declares this routine has a reference to a SubjectObserverAssociation class, this routine will delegate its task to this class, by invoking to a routine instance of RemoveLink.

\[
\text{not self.subject.subjectObserverReference} \rightarrow \text{isEmpty} \implies \text{self.invokedRoutine} \rightarrow \exists (r | r.\text{oclIsTypeOf}(\text{RemoveLink}))
\]
GeneralizationObserver

**Description**
This metaclass specifies a generalization between an abstract observer (ClassObserver) and a concrete observer (ConcreteObserver) in the model of the pattern Observer.

**Generalizations**
- Generalization (from Kernel)

**Associations**
- classObserver: ClassObserver [1] It denotes the general element of this relation. It redefines Generalization::general.
- concreteObserver: ConcreteObserver [1] It denotes the specific element of this relation. It redefines Generalization::specific.

**Constraints**
No additional constraints.

GeneralizationSubject

**Description**
This metaclass specifies a generalization between an abstract subject (ClassSubject) and a concrete subject (ConcreteSubject) in the model of the pattern Observer.

**Generalizations**
- Generalization (from Kernel)

**Associations**
- classSubject: ClassSubject [1] It denotes a general element of this relation. It redefines Generalization::general.
- concreteSubject: ConcreteSubject [1] It denotes the specific element of this relation. It redefines Generalization::specific.

**Constraints**
No additional constraints.
Appendix D

GetState

Description
This metaclass specifies a generalization between an abstract subject (ClassSubject) and a concrete subject (ConcreteSubject) in the model of the pattern Observer.

Generalizations

- Method (from ISM-Java)

Associations
No additional associations.

Constraints

[1] It is an observer and concrete method. self.isQuery and not self.isAbstract
[2] As it must return the subject state, the set of parameters must not be empty, and at least, one of the parameters must have a direction equal to out or return. self.parameter -> notEmpty () and self. parameter -> select (par | par.direction = #return or par.direction = #out) -> size () >=1
[3] Its visibility must be public. self.visibility = #public

InterfaceObserver

Description
An InterfaceObserver specifies the features that must have a Java interface taking the role of abstract observer in the model of the pattern Observer.

Generalizations

- Observer, JavaInterface (from ISM-Java)

Associations

- update: Update [1..*] Every instance of InterfaceObserver has at least an operation instance of Update. It is a subset of JavaInterface::method.

Constraints
No additional constraints.

InterfaceSubject

Description
This metaclass specifies the features that must have a Java interface taking the role of abstract subject in the model of a pattern Observer.
Appendix D

Generalizations

- Subject, JavaInterface (from ISM-Java)

Associations

- attach: Attach [1..*] Every instance of InterfaceSubject has at least a method instance of Attach. It is a subset of JavaInterface::method.
- detach: Detach [1..*] Every instance of InterfaceSubject has at least a method instance of Detach. It is a subset of JavaInterface::method.
- notify: Notify[1..*] Every instance of InterfaceSubject has at least a method instance of Notify. It is a subset of JavaInterface::method.

Constraints

No additional constraints.

InterfaceRealizationObserver

Description

This metaclass specifies an interface realization between an abstract observer (InterfaceObserver) and a concrete observer (ConcreteObserver) in the model of the pattern Observer.

Generalizations

- InterfaceRealization (from Kernel)

Associations

- concreteObserver: ConcreteObserver [1] It denotes the element that implements the contract in this relation. It redefines InterfaceRealization::implementingClassifier.
- interfaceObserver: InterfaceObserver [1] It denotes the element that defines the contract in this relation. It redefines InterfaceRealization::contract.

Constraints

No additional constraints.

InterfaceRealizationSubject

Description

This metaclass specifies an interface realization between an abstract subject (InterfaceSubject) and a concrete subject (ConcreteSubject) in the model of the pattern Observer.
Appendix D

Generalizations

- InterfaceRealization (from Kernel)

Associations

- concreteSubject: ConcreteSubject [1] It denotes the element that implements the contract in this relation. It redefines InterfaceRealization::implementingClassifier.
- interfaceSubject: InterfaceSubject [1] It denotes the element that defines the contract in this relation. It redefines InterfaceRealization::contract.

Constraints

No additional constraints.

Notify

Description

It defines a method that is declared by a subject.

Generalizations

- Method (from ISM-Java)
- classSubject: ClassSubject [0..1] It denotes the class that declares this method. It redefines JavaOperation::class.
- interfaceSubject: InterfaceSubject [0..1] It denotes the interface that declares this method. It redefines Method::interface.

Constraints

[1] It is a method that does not change the subject state. self.isQuery
[2] Its visibility must be public. self.visibility = #public
[3] If the subject that declares this routine has a reference to a SubjectObserverAssociation class, this routine will delegate its task to this class by invoking this routine instance of NotifyObserver. not self.subject.subjectObserverReference -> isEmpty implies self.invokedRoutine -> exists (r | r.oclIsTypeOf(NotifyObserver))

NotifyObservers

Description

It defines a method that is declared by an instance of SubjectObserverAssociation.

Generalizations

- Method (from ISM-Java)
Appendix D

Associations

- subjectObserverAssociation:SubjectObserverAssociation [1] It denotes the class that declares this method. It redefines JavaOperation::class.

Constraints

[1] This is a method that does not change the state of the instance that defines it. self.isQuery
[2] Its visibility must be public. self.visibility = #public

Observer

Description
An observer is a specialized classifier that specifies a classifier whose role is observer in the model of the pattern Observer. It is an abstract metaclass.

Generalizations

- Classifier (from Kernel)

Associations
No additional associations.

Constraints
No additional constraints.

ObserverReference

Description
This field represents a reference to observers of a subject.

Generalizations

- Field (from ISM-Java)

Associations

- subject: Subject [1] It denotes the class that declares this field. It redefines Field::class.

Constraints

[1] The type of this field must correspond to some of the collections of the Java library. self.type.oclIsKindOf (JavaCollection) and self.type.parameter -> size () = 1 and self.type.parameter.ownedParameteredElement.oclIsTypeOf (Observer)
Appendix D

[2] Its visibility must be private or protected. self.visibility = #private or self.visibility = #protected

RemoveLink

Description
It defines a method that is declared by an instance of SubjectObserverAssociation.

Generalizations

• Method (from ISM-Java)

Associations

• subjectObserverAssociation:SubjectObserverAssociation [1] It denotes the class that declares this method. It redefines JavaOperation::class.

Constraints

[1] This method changes the state of instances defining it. not self.isQuery
[2] It has a non-empty set of parameters being one of them an input parameter whose type is observer. self.parameter -> notEmpty() and self.parameter -> select (param | param.direction = #in and param.type = oclIsKindOf (Observer)) -> size () = 1
[3] Its visibility must be public. self.visibility = #public

SetState

Description
It defines an operation member of ConcreteSubject. It specifies a service that can be required from another object.

Generalizations

• Method (from ISM-Java)

Associations
No additional associations.

Constraints

[1] This is a method that is concrete and modifies the state of the subject. not self.isAbstract and not self.isQuery
[2] It has a non-empty set of parameters being at least one of them an input parameter. self.parameter -> notEmpty () and self.parameter ->select (param | param.direction = #in) -> size () >= 1
[3] Its visibility must be public. self.visibility = #public
Subject

Description
This metaclass specifies a specialized classifier that takes the role of subject in the model of the pattern Observer. It is an abstract metaclass.

Generalizations

• Classifier (from Kernel)

Associations
No additional associations.

Constraints
No additional constraints.

SubjectObserverAssociation

Description
This metaclass specifies the features of the class that maintains the relation between a subject and its observers.

Generalizations

• JavaClass (from ISM-Java)

Associations

• mapping: SubjectObserverMapping [1] It specifies an own attribute. It is a subset of JavaClass::field.
• notify: NotifyObservers [1..*] It specifies an own operation. It is a subset of JavaClass::method.
• register: AddLink [1..*] It specifies an own operation. It is a subset of JavaClass::method.
• unregister: RemoveLink [1..*] It specifies an own operation. It is a subset of JavaClass::method.

Constraints
No additional constraints.

SubjectObserverMapping

Description
This metaclass specifies the class attribute SubjectObserverAssociation, that maintains the mapping between a subject and its observers.
Appendix D

Generalizations

• Field (from ISM-Java)

Associations

• subjectObserverAssociation: SubjectObserverAssociation [1] It denotes the class that declares this attribute. It redefines Field::Class.

Constraints

[1] Its visibility must be private or protected. self.visibility = #private or self.visibility = #protected

SubjectObserverReference

Description
This attribute is a reference to a class SubjectObserverAssociation, which maintain the relation between a subject and its observers.

Generalizations

• Field (from ISM-Java)

Associations

• subject: Subject [1] It denotes the class that declares this attribute. It redefines Attribute:: class.
• type: SubjectObserverAssociation [1] It refers to the type of this attribute. It redefines Attribute::type.

Constraints
No additional constraints.

Update

Description
It defines a method that is declared by an observer which specifies a service that is required by another object.

Generalizations

• Method (from ISM-Java)
Associations

- classObserver: ClassObserver [0..1] It denotes the class that declares this operation. It redefines JavaOperation::class.
- interfaceObserver: InterfaceObserver [0..1] It denotes the interface that declares this operation. It redefines Method::interface.

Constraints

- [1] This is a method that does not change the state of the observer. self.isQuery
- [2] Its visibility must be public. self.visibility = #public