

# *OA&D CORBAfacility Interface Definition*

5

This chapter specifies the interfaces for a CORBAfacility for Object Analysis & Design, consistent with the Unified Modeling Language, version 1.0. An OA&D Facility is a repository for models expressed in the UML. The facility enables the creation, storage, and manipulation of UML models. The facility enables clients to be developed that provide a wide variety of model-based development capabilities, including:

- Drawing and animation of UML models in UML and other notations
- Enforcement of process and method style guidelines
- Metrics, queries, and reports
- Automation of certain development lifecycle activities (e.g., through design wizards and code generation).

## *Contents*

This chapter contains the following sections.

<b>Section Title</b>	<b>Page</b>
“Service Description”	5-2
“Mapping of UML Semantics to Facility Interfaces”	5-4
“Facility Implementation Requirements”	5-9
“IDL Modules”	5-10

## 5.1 *Service Description*

There are two sets of interfaces provided: 1) generic and 2) tailored. Both sets of interfaces enable the creation and traversal of UML model elements. The generic interfaces are included in the Reflective module.

This is a set of general-purpose interfaces that provide utility for browser type functionality and as a base for the tailored interfaces. They are more fully described in the Meta-Object Facility (MOF) specification.

A set of tailored interfaces that are specifically typed to the UML metamodel elements is defined. The tailored interfaces inherit from the generic interfaces. The tailored interfaces provide capabilities necessary to instantiate, traverse, and modify UML model elements in the facility, directly in terms of the UML metamodel, with type safety. The specifications of the tailored interfaces were generated by applying a set of transformations to the UML semantic metamodel. Because the tailored interfaces were generated consistently from a set of patterns (described more fully in the MOF specification), they are easy to understand and program against. It is feasible to generate automatically the implementation for the OA&D facility, for the most part, because of these patterns and because the UML metamodel is strictly structural.

The UML is designed with a layered architecture. Implementers can choose which layers to implement, and whether to implement only the generic interfaces or the generic and tailored interfaces.

One of the primary goals was to advance the state of the industry by enabling OO modeling tool interoperability. This OA&D facility defines a set of interfaces to provide that tool interoperability. However, enabling meaningful exchange of model information between tools requires agreement on semantics and their visualization. The metamodel documenting the UML semantics and notation is defined in the UML Semantics chapter. Most of the IDL defined in this document is a direct mapping of the UML v1.0 metamodel, based on the IDL mapping defined in the MOF specification. Because the UML semantics are sufficiently complex, they are documented separately in the UML Semantics chapter, whereas this chapter is void of explanations of semantics.

### 5.1.1 Tool Sharing Options

A major goal is to achieve semantic interoperability between OA&D tools. Figure 5-1 depicts several viable alternatives to exchanging model information between tools.

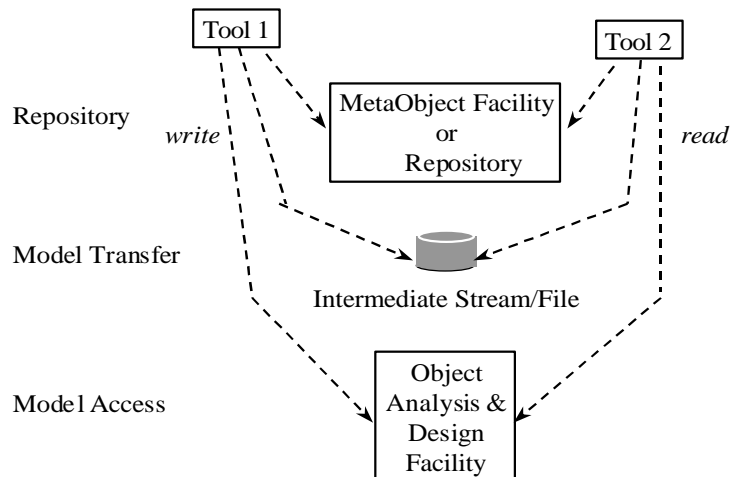


Figure 5-1 Model Sharing Alternatives between OA&D Tools

#### *General-purpose Repository*

Two tools could interface to the same repository and access a model there. The MetaObject Facility (MOF) could be this repository. This mapping is very implementation dependent, since the MOF cannot necessarily enforce the richer semantics defined in a UML-compliant tool. This approach is not described in this response, although the mapping to the MOF is described in the Preface.

#### *Model Transfer*

Two tools could understand the same stream format and exchange models via that stream, which could be a file. This is referred to as an "import facility." Having this interface would be necessary to provide a path for tools that are not implemented in an API (CORBA or non-CORBA) or repository environment. The Preface discusses stream format and CDIF further.

#### *Model Access*

Two tools could exchange models on a detail-by-detail basis. This is referred to as a "connection facility." Although this would not be the most efficient method for sharing an entire model, this type of access enables semantic interoperability to the greatest

degree and is extremely useful for client applications. This, too, is a repository, but its interfaces are specific to the OA&D domain. A set of IDL interfaces is defined in this document to provide model access.

In summary, the OA&D Facility defines IDL interfaces for clients to use in a Model Access mode. The interface is consistent with the UML metamodel contained in this response.

## 5.2 *Mapping of UML Semantics to Facility Interfaces*

Understanding the process used to generate the IDL for this facility is helpful in understanding the resulting IDL. The process was as follows:

1. Converted the UML Semantics Metamodel into the Interface Metamodel, making necessary refinements for CORBA interfaces.
2. Stored the Interface Metamodel into a MetaObject Facility prototype as an instance of the MOF meta-metamodel elements.
3. Generated IDL from the MOF, based on the mapping defined in the MOF proposal.

### 5.2.1 *Transformation of UML Semantics Metamodel into Interfaces Metamodel*

A model was created representing the interfaces required on the OA&D Facility. This interface metamodel is nearly identical to the UML Semantics metamodel, so it is not documented explicitly. The following list summarizes the conversions made from the UML Semantics metamodel:

- Named associations and their ends, where names were missing.
- Deleted derived associations, since they would have resulted in redundant interfaces.
- Mapped all UML data types and select classes to CORBA data types.
- Transformed association classes into more fundamental structures. (A goal of the MOF was simplicity, so it does not support association classes.)
- Combined the UML DataTypes and Extension Mechanisms packages into Core, resulting in easier-to-use name scoping for the interfaces.
- Renamed certain classifiers, association ends, and attributes to avoid conflicts with words reserved in Reflective interfaces, CORBA, and MOF.
- Set navigability for associations to be uni-directional between classifiers that crossed packages. This was necessary to permit implementations of the more fundamental packages/modules without requiring a full UML implementation<sup>1 2</sup>.

---

1. All other navigability is assumed to be useful. For example, although an implementation environment class should not know about its child classes, it is useful for an OA&D tool.

- Renamed AssociationClass to UmlAssociationClass. (The MOF IDL generation creates a FooClass for every Foo, so the UML class 'Association' would have created an 'AssociationClass' interface which would have clashed.)
- Renamed enumeration literal names so they would be unique within the resulting IDL modules.

The IDL generation from the MOF assures that all root classes in the interface metamodel are specializations of `Reflective::RefObject`, so this relationship is assumed to be present in the interface metamodel.

The remainder of this chapter describes the transformation of Association Classes as in the UML Semantics metamodel to the interface metamodel, summarizes the usage of CORBA data types, and summarizes the source and purpose of the MOF Reflective interfaces.

### *Transformation for Association Classes*

Since the MOF does not represent the semantics of association classes directly, we needed to convert the association classes in the UML into a simple class and add necessary relationships to enable complete navigation (in the resulting facility IDL). Figure 5-2 shows an example association class as it would appear in the semantic metamodel.

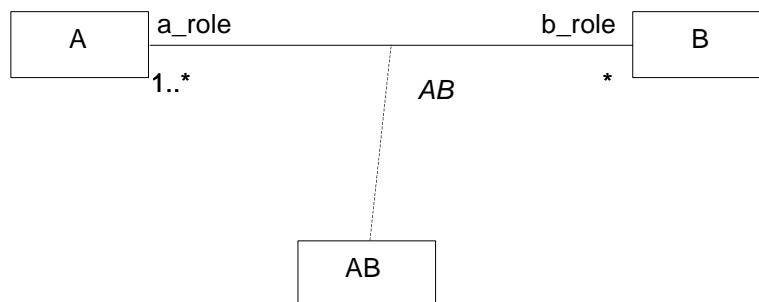


Figure 5-2 An Association Class in a Semantic Metamodel

2. The OA&D facility interfaces exclude navigation from classes in base packages/modules to classes in dependent packages/modules. This is deliberate. For example, an instance of a UML class does not know about a State Machine that might be attached to it. Vendors should consider adding interfaces to support such navigation when implementing multiple modules.

Figure 5-3 shows the corresponding transformed structure in the interface model.

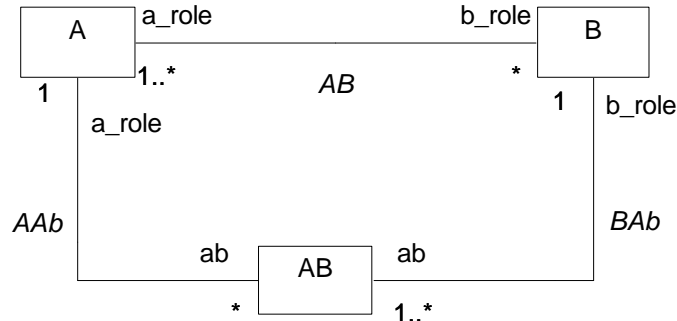


Figure 5-3 Corresponding Association Class in an Interface Metamodel

### *MOF Generic Interfaces*

The MOF specification fully describes the generic interfaces. As a summary, the generic interfaces in the Reflective module provide the following:

- consistent treatment of type information,
- exception handling (including constraint violations, missing parameters, etc.), and
- generic creation and traversal of objects.

---

**Note** – The MOF specification replaces the definition of the Reflective module contained in this specification.

---

### *DataTypes for Interface*

UML itself is platform independent; therefore, during the translation to the interface model, specific CORBA data types were selected as the structural base. These are listed in Table 5-1.

Table 5-1 Data Types

UML DataType	IDL Declaration
String	//string
Integer	typedef short Integer;
Uninterpreted	typedef any Uninterpreted;
Time	typedef float Time;
Name	struct Name { string body ; };
GraphicMarker	struct GraphicMarker { any body ; } ;
Geometry	struct Geometry { any body ; } ;
TimeExpression	struct TimeExpression { Name language ; any body ; } ;

Table 5-1 Data Types

ObjectSetExpression	struct ObjectSetExpression { Name language ; any body ; } ;
ProcedureExpression	struct ProcedureExpression { Name language ; any body ; } ;
Expression	struct Expression { Name language ; any body ; } ;
BooleanExpression	struct BooleanExpression { Name language ; any body ; } ;
Mapping	struct Mapping { any body ; } ;
MultiplicityRange	struct MultiplicityRange { lower short; upper short ; } ;
Multiplicity	sequence < MultiplicityRange > Multiplicity;
ChangeableKind	enum ChangeableKind { ck_none, ck_frozen, ck_addOnly } ;
OperationDirectionKind	enum OperationDirectionKind { odk_provide, odk_require } ;
ParameterDirectionKind	enum ParameterDirectionKind { pdk_in, pdk_inout, pdk_out, pdk_return};
MessageDirectionKind	enum MessageDirectionKind { mdk_activation, mdk_return } ;
SynchronousKind	enum SynchronousKind { sk_synchronous, sk_asynchronous};
ScopeKind	enum ScopeKind {sk_instance, sk_type};
VisibilityKind	enum VisibilityKind { vk_public, vk_protected, vk_private } ;
PseudostateKind	enum PseudostateKind { pk_initial, pk_final, pk_shallowHistory, pk_deepHistory, pk_join, pk_fork, pk_branch, pk_or } ;
CallConcurrencyKind	enum CallConcurrencyKind { cck_sequential, cck_guarded, cck_concurrent } ;
AggregationKind	enum AggregationKind {ak_none, ak_shared, ak_composite};

### 5.2.2 Mapping of Interface Model into MOF

The UML metamodel elements can be expressed as instances of MOF meta-metamodel elements. This mapping is summarized in the table below for the relevant elements in the interface metamodel.

Table 5-2 Relevant Elements in the Interface Metamodel

<b>Package</b>	<b>Package, Contains</b>
Class	Class, Contains
Attribute	Attribute, Contains, IsOfType
DataType	DataType
Association	Association, AssociationEnd, Reference, Contains, RefersTo, IsOfType
Generalization	Generalizes

These mappings are relatively straightforward, with the exception of how an Association is instantiated. Figure 5-4 on page 5-8 shows an example association as it would appear in the interface model. Figure 5-5 on page 5-8 illustrates a relevant view of the MOF meta-metamodel.

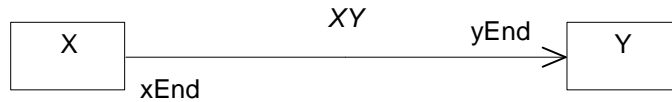


Figure 5-4 Association at Meta-Model Level Example

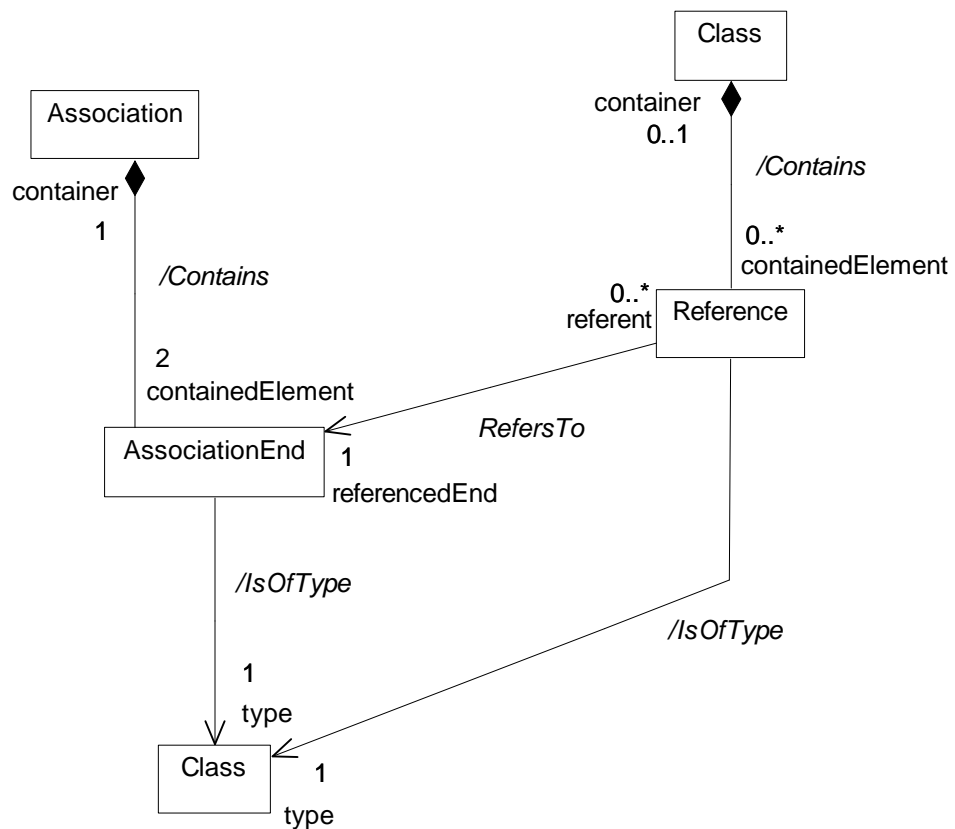


Figure 5-5 Projection of MOF Meta-Metamodel

Figure 5-6 on page 5-9 is a collaboration diagram showing how the association would be instantiated in terms of the MOF.



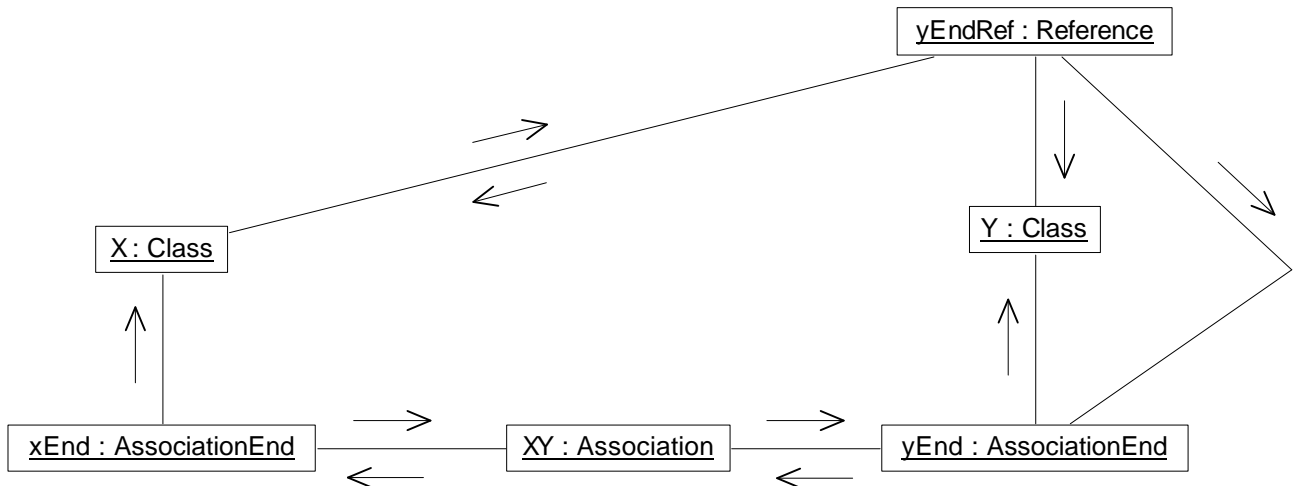


Figure 5-6 Collaboration Diagram showing Association Instantiated in Terms of MOF

In Figure 5-6, the message arrows are based on the navigation in the MOF meta-model and indicate structural knowledge and potential messaging using the resulting interface.

### 5.2.3 Mapping from MOF to IDL

The description for the mapping from instances of models stored in the MOF is described in detail in the MOF specification. The result of this mapping is the generated IDL in this specification

## 5.3 Facility Implementation Requirements

Although this chapter focuses on defining the interfaces for the facility and leaves implementation decisions up to the creativity of vendors, there are some implementation requirements.

The UML Standard Elements (stereotypes, constraints, and tags) must be known to a facility implementation, or provided via a load. This is necessary so that the interoperability of these elements can be achieved. The semantics of the standard elements (e.g., containment restrictions) must be enforced. The Standard Elements are documented in the UML Semantics chapter.

The facility interfaces inherit from generic interfaces defined in the Reflective module. These interfaces provide common operations, such as `verify()`. The `verify()` operation should be implemented to return well-formedness violations numbered equal to the well-formedness rule following the meta-class definition in the UML Semantics

chapter. This includes the semantics for the UML Standard Elements. The Reflective interfaces and exception handling is described in the Meta Object Facility (MOF) specification.

## 5.4 IDL Modules

### 5.4.1 Reflective

```
1 #ifndef REFLECTIVE_IDL
2 #define REFLECTIVE_IDL
3
4 // #include <orb.idl>
5 module Reflective {
6
7  interface RefBaseObject;
8
9  interface RefObject;
10 typedef sequence < RefObject > RefObjectUList;
11 typedef sequence < RefObject > RefObjectSet;
12
13 interface RefAssociation;
14 interface RefPackage;
15
16 typedef RefObject DesignatorType;
17 typedef any ValueType;
18 typedef sequence < ValueType > ValueTypeList;
19 typedef sequence < RefObject, 2 > Link;
20 typedef sequence < ValueType > ErroneousValues;
21
22 const string UNDERFLOW_VIOLATION = "underflow";
23 const string OVERFLOW_VIOLATION = "overflow";
24 const string DUPLICATE_VIOLATION = "duplicate";
25 const string TYPE_CLOSURE_VIOLATION = "type closure";
26 const string COMPOSITION_VIOLATION = "composition";
27 const string INVALID_OBJECT_VIOLATION = "invalid object";
28
29 struct StructuralViolation {
30  string violation_kind;
31  RefObject element_designator;
32  ErroneousValues offending_values;
33 };
```

```
34 typedef sequence < StructuralViolation > StructuralViolationSet;
35 exception StructuralError {
36   StructuralViolationSet violations;
37 };
38 struct ConstraintViolation {
39   RefObject constraint_designator;
40   ErroneousValues offending_values;
41   string explanation_text;
42 };
43 exception ConstraintError {
44   ConstraintViolation violation;
45 };
46 struct ErrorDescription {
47   string error_name;
48   ErroneousValues offending_values;
49   string explanation_text;
50 };
exception SemanticError {
52   ErrorDescription error;
53 };
54
55 exception NotFound {};
56 exception NotSet {};
57 exception BadPosition {};
58 exception AlreadyCreated {};
59 exception InvalidLink {};
60 exception InvalidDesignator {
61   DesignatorType designator;
62   string element_kind;
63 };
64 exception InvalidValue {
65   DesignatorType designator;
66   string element_kind;
67   ValueType value;
68   CORBA::TypeCode type_expected;
69 };
70 exception InvalidObject {
71   DesignatorType designator;
72   RefObject obj;
73   CORBA::TypeCode type_expected;
74 };
```

```
75 exception MissingParameter {
76   DesignatorType designator;
77 };
78 exception TooManyParameters {};
79 exception OtherException {
80   DesignatorType exception_designator;
81   ValueTypeList exception_values;
82 };
83
84 interface RefBaseObject {
85   DesignatorType meta_object ();
86   boolean itself (in RefBaseObject other_object);
87   RefBaseObject repository_container ();
88 }; // end of RefBaseObject
89
90
91 interface RefObject : RefBaseObject {
92   boolean is_instance_of (in DesignatorType obj_type,
93     in boolean consider_subtypes);
94   RefObject create_instance (in ValueTypeList args)
95     raises (TooManyParameters,
96       MissingParameter,
97       InvalidValue,
98       AlreadyCreated,
99       StructuralError,
100       ConstraintError,
101       SemanticError);
102   RefObjectSet all_objects (in boolean include_subtypes);
103   void set_value (in DesignatorType feature,
104     in ValueType value)
105     raises (InvalidDesignator,
106       InvalidValue,
107       StructuralError,
108       ConstraintError,
109       SemanticError);
110   ValueType value (in DesignatorType feature)
111     raises (InvalidDesignator,
112       SemanticError);
113   void add_value (in DesignatorType feature,
114     in ValueType value)
115     raises (InvalidDesignator,
```

```
116     InvalidValue,
117     StructuralError,
118     ConstraintError,
119     SemanticError);
120 void add_value_before (in DesignatorType feature,
121                       in ValueType value,
122                       in ValueType existing_value)
123 raises (InvalidDesignator,
124        InvalidValue,
125        NotFound,
126        StructuralError,
127        ConstraintError,
128        SemanticError);
129 void add_value_at (in DesignatorType feature,
130                  in ValueType value,
131                  in long position)
132 raises (InvalidDesignator,
133        InvalidValue,
134        BadPosition,
135        StructuralError,
136        ConstraintError,
137        SemanticError);
138 void modify_value (in DesignatorType feature,
139                  in ValueType existing_value,
140                  in ValueType new_value)
141 raises (InvalidDesignator,
142        InvalidValue,
143        NotFound,
144        StructuralError,
145        ConstraintError,
146        SemanticError);
147 void modify_value_at (in DesignatorType feature,
148                      in ValueType new_value,
149                      in long position)
150 raises (InvalidDesignator,
151        InvalidValue,
152        BadPosition,
153        StructuralError,
154        ConstraintError,
155        SemanticError);
156 void remove_value (in DesignatorType feature,
```

```
157         in ValueType existing_value)
158     raises (InvalidDesignator,
159           InvalidValue,
160           NotFound,
161           StructuralError,
162           ConstraintError,
163           SemanticError);
164 void remove_value_at (in DesignatorType feature,
165                     in long position)
166     raises (InvalidDesignator,
167           InvalidValue,
168           BadPosition,
169           NotFound,
170           StructuralError,
171           ConstraintError,
172           SemanticError);
173 ValueType invoke_operation (in DesignatorType requested_operation,
174                             in ValueTypeList args)
175     raises (InvalidDesignator,
176           TooManyParameters,
177           MissingParameter,
178           InvalidValue,
179           OtherException,
180           ConstraintError,
181           SemanticError);
182 }; // end of interface RefObject
183
184 interface RefAssociation : RefBaseObject {
185     boolean link_exists (in Link some_link)
186     raises (InvalidLink,
187           SemanticError);
188     RefObjectUList query (in DesignatorType query_end,
189                          in RefObject query_object)
190     raises (InvalidDesignator,
191           InvalidObject,
192           SemanticError);
193     void add_link (in Link new_link)
194     raises (InvalidLink,
195           StructuralError,
196           ConstraintError,
197           SemanticError);
```

```
198 void add_link_before (in Link new_link,
199                       in DesignatorType position_end,
200                       in RefObject position_value)
201 raises (InvalidDesignator,
202        InvalidObject,
203        InvalidLink,
204        NotFound,
205        StructuralError,
206        ConstraintError,
207        SemanticError);
208 void modify_link (in Link existing_link,
209                  in DesignatorType position_end,
210                  in RefObject position_value)
211 raises (InvalidDesignator,
212        InvalidObject,
213        InvalidLink,
214        NotFound,
215        StructuralError,
216        ConstraintError,
217        SemanticError);
218 void remove_link (in Link existing_link)
219 raises (InvalidLink,
220        NotFound,
221        StructuralError,
222        ConstraintError,
223        SemanticError);
224 }; // end of interface RefAssociation
225
226 interface RefPackage : RefBaseObject {
227   RefObject get_class_ref (in DesignatorType type)
228   raises (InvalidDesignator);
229   RefAssociation get_association (in DesignatorType association)
230   raises (InvalidDesignator);
231   RefPackage get_nested_package (in DesignatorType nested_package)
232   raises (InvalidDesignator);
233 }; // end of interface RefPackage
234 }; // end of module Reflective
235
236 #endif
```

UMLCore

```
237 #include "Reflective.idl"
238
239 module UmlCore {
240   interface UmlCorePackage;
241   interface Enumeration;
242   interface EnumerationClass;
243   typedef sequence<Enumeration> EnumerationUList;
244   interface Generalization;
245   interface GeneralizationClass;
246   typedef sequence<Generalization> GeneralizationUList;
247   typedef sequence<Generalization> GeneralizationSet;
248   interface Class;
249   interface ClassClass;
250   typedef sequence<Class> ClassUList;
251   interface Dependency;
252   interface DependencyClass;
253   typedef sequence<Dependency> DependencyUList;
254   typedef sequence<Dependency> DependencySet;
255   interface Parameter;
256   interface ParameterClass;
257   typedef sequence<Parameter> ParameterUList;
258   interface GeneralizableElement;
259   interface GeneralizableElementClass;
260   typedef sequence<GeneralizableElement> GeneralizableElementUList;
261   interface Constraint;
262   interface ConstraintClass;
263   typedef sequence<Constraint> ConstraintUList;
264   typedef sequence<Constraint> ConstraintSet;
265   interface ModelElement;
266   interface ModelElementClass;
267   typedef sequence<ModelElement> ModelElementUList;
268   typedef sequence<ModelElement> ModelElementSet;
269   interface ElementOwnership;
270   interface ElementOwnershipClass;
271   typedef sequence<ElementOwnership> ElementOwnershipUList;
272   typedef sequence<ElementOwnership> ElementOwnershipSet;
273   interface Classifier;
274   interface ClassifierClass;
275   typedef sequence<Classifier> ClassifierUList;
276   typedef sequence<Classifier> ClassifierSet;
277   interface UmlAttribute;
```



```
278 interface UmlAttributeClass;
279 typedef sequence<UmlAttribute> UmlAttributeUList;
280 interface EnumerationLiteral;
281 interface EnumerationLiteralClass;
282 typedef sequence<EnumerationLiteral> EnumerationLiteralUList;
283 interface Element;
284 interface ElementClass;
285 typedef sequence<Element> ElementUList;
286 interface Namespace;
287 interface NamespaceClass;
288 typedef sequence<Namespace> NamespaceUList;
289 interface Primitive;
290 interface PrimitiveClass;
291 typedef sequence<Primitive> PrimitiveUList;
292 interface UmlAssociationClass;
293 interface UmlAssociationClassClass;
294 typedef sequence<UmlAssociationClass> UmlAssociationClassUList;
295 interface StructuralFeature;
296 interface StructuralFeatureClass;
297 typedef sequence<StructuralFeature> StructuralFeatureUList;
298 interface Feature;
299 interface FeatureClass;
300 typedef sequence<Feature> FeatureUList;
301 typedef sequence<Feature> FeatureSet;
302 interface Stereotype;
303 interface StereotypeClass;
304 typedef sequence<Stereotype> StereotypeUList;
305 typedef sequence<Stereotype> StereotypeSet;
306 interface Association;
307 interface AssociationClass;
308 typedef sequence<Association> AssociationUList;
309 interface TaggedValue;
310 interface TaggedValueClass;
311 typedef sequence<TaggedValue> TaggedValueUList;
312 typedef sequence<TaggedValue> TaggedValueSet;
313 interface AssociationEnd;
314 interface AssociationEndClass;
315 typedef sequence<AssociationEnd> AssociationEndUList;
316 typedef sequence<AssociationEnd> AssociationEndSet;
317 interface Operation;
318 interface OperationClass;
```

```
319 typedef sequence<Operation> OperationUList;
320 interface BehavioralFeature;
321 interface BehavioralFeatureClass;
322 typedef sequence<BehavioralFeature> BehavioralFeatureUList;
323 typedef sequence<BehavioralFeature> BehavioralFeatureSet;
324 interface Request;
325 interface RequestClass;
326 typedef sequence<Request> RequestUList;
327 interface Method;
328 interface MethodClass;
329 typedef sequence<Method> MethodUList;
330 typedef sequence<Method> MethodSet;
331 interface DataType;
332 interface DataTypeClass;
333 typedef sequence<DataType> DataTypeUList;
334 interface UmlInterface;
335 interface UmlInterfaceClass;
336 typedef sequence<UmlInterface> UmlInterfaceUList;
337 interface Structure;
338 interface StructureClass;
339 typedef sequence<Structure> StructureUList;
340 enum AggregationKind { ak_none, ak_shared, ak_composite };
341 enum CallConcurrencyKind { cck_sequential, cck_guarded, cck_concurrent };
342 enum ChangeableKind { ck_none, ck_frozen, ck_addOnly };
343 typedef short Integer;
344 enum MessageDirectionKind { mdk_activation, mdk_return };
345 enum OperationDirectionKind { odk_provide, odk_require };
346 enum ParameterDirectionKind { pdk_in, pdk_inout, pdk_out, pdk_return };
347 enum ScopeKind { sk_instance, sk_type };
348 enum SynchronousKind { sk_synchronous, sk_asynchronous };
349 typedef float Time;
350 typedef any Uninterpreted;
351 enum VisibilityKind { vk_public, vk_protected, vk_private };
352 struct Name { string body; };
353 struct Expression { Name language; any body; };
354 struct BooleanExpression { Name language; any body; };
355 struct ObjectSetExpression { Name language; any body; };
356 struct ProcedureExpression { Name language; any body; };
357 struct TimeExpression { Name language; any body; };
358 struct Geometry { any body; };
359 struct GraphicMarker { any body; };
```

```
360 struct MultiplicityRange { short lower; short upper; };
361 enum PseudostateKind { pk_initial,
362     pk_final,
363     pk_shallowHistory,
364     pk_deepHistory,
365     pk_join,
366     pk_fork,
367     pk_branch,
368     pk_or };
369 typedef sequence <MultiplicityRange> Multiplicity;
370 struct Mapping { any body; };
371
372 interface ElementClass : Reflective::RefObject {
373     readonly attribute ElementUList all_of_kind_element;
374 };
375
376 interface Element : ElementClass { };
377
378 interface TaggedValueClass : ElementClass {
379     readonly attribute TaggedValueUList all_of_kind_tagged_value;
380     readonly attribute TaggedValueUList all_of_type_tagged_value;
381     TaggedValue create_tagged_value (in Name tag,
382         in Uninterpreted uml_value)
383     raises (Reflective::SemanticError);
384 };
385
386 interface TaggedValue : TaggedValueClass, Element {
387     Name tag ()
388     raises (Reflective::SemanticError);
389     void set_tag (in Name new_value)
390     raises (Reflective::SemanticError);
391     Uninterpreted uml_value ()
392     raises (Reflective::SemanticError);
393     void set_uml_value (in Uninterpreted new_value)
394     raises (Reflective::SemanticError);
395 };
396
397 interface EnumerationLiteralClass : ElementClass {
398     readonly attribute EnumerationLiteralUList all_of_kind_enumeration_literal;
399     readonly attribute EnumerationLiteralUList all_of_type_enumeration_literal;
400     EnumerationLiteral create_enumeration_literal (in UmlCore::Name name)
```

```
401     raises (Reflective::SemanticError);
402 };
403
404 interface EnumerationLiteral : EnumerationLiteralClass, Element {
405     UmlCore::Name name ()
406     raises (Reflective::SemanticError);
407     void set_name (in UmlCore::Name new_value)
408     raises (Reflective::SemanticError);
409     UmlCore::Enumeration enumeration ()
410     raises (Reflective::SemanticError);
411     void set_enumeration (in UmlCore::Enumeration new_value)
412     raises (Reflective::SemanticError);
413 };
414
415 interface ModelElementClass : ElementClass {
416     readonly attribute ModelElementUList all_of_kind_model_element;
417 };
418
419 interface ModelElement : ModelElementClass, Element {
420     UmlCore::Name name ()
421     raises (Reflective::SemanticError);
422     void set_name (in UmlCore::Name new_value)
423     raises (Reflective::SemanticError);
424     UmlCore::Namespace namespace ()
425     raises (Reflective::NotSet, Reflective::SemanticError);
426     void set_namespace (in UmlCore::Namespace new_value)
427     raises (Reflective::SemanticError);
428     void unset_namespace ()
429     raises (Reflective::SemanticError);
430     DependencySet provision ()
431     raises (Reflective::NotSet, Reflective::SemanticError);
432     void add_provision (in DependencySet new_value)
433     raises (Reflective::StructuralError, Reflective::SemanticError);
434     void remove_provision ()
435     raises (Reflective::SemanticError);
436     UmlCore::TaggedValueSet tagged_value ()
437     raises (Reflective::NotSet, Reflective::SemanticError);
438     void add_tagged_value (in UmlCore::TaggedValueSet new_value)
439     raises (Reflective::StructuralError, Reflective::SemanticError);
440     void remove_tagged_value ()
441     raises (Reflective::SemanticError);
```

```

442 UmlCore::ConstraintSet constraint ()
443     raises (Reflective::NotSet, Reflective::SemanticError);
444 void add_constraint (in UmlCore::ConstraintSet new_value)
445     raises (Reflective::StructuralError, Reflective::SemanticError);
446 void remove_constraint ()
447     raises (Reflective::SemanticError);
448 DependencySet requirement ()
449     raises (Reflective::NotSet, Reflective::SemanticError);
450 void add_requirement (in DependencySet new_value)
451     raises (Reflective::StructuralError, Reflective::SemanticError);
452 void remove_requirement ()
453     raises (Reflective::SemanticError);
454 ModelElement template ()
455     raises (Reflective::NotSet, Reflective::SemanticError);
456 void set_template (in ModelElement new_value)
457     raises (Reflective::SemanticError);
458 void unset_template ()
459     raises (Reflective::SemanticError);
460 ModelElementUList template_parameter ()
461     raises (Reflective::NotSet, Reflective::SemanticError);
462 void add_template_parameter (in ModelElementUList new_value)
463     raises (Reflective::StructuralError, Reflective::SemanticError);
464 void add_template_parameter_before (in ModelElement new_value,
465                                     in ModelElement before)
466     raises (Reflective::StructuralError,
467             Reflective::NotFound,
468             Reflective::SemanticError);
469 void remove_template_parameter ()
470     raises (Reflective::SemanticError);
471 ElementOwnership namespace1 ()
472     raises (Reflective::NotSet, Reflective::SemanticError);
473 void set_namespace1 (in ElementOwnership new_value)
474     raises (Reflective::SemanticError);
475 void unset_namespace1 ()
476     raises (Reflective::SemanticError);
477 };
478
479 interface FeatureClass : ModelElementClass {
480     readonly attribute FeatureUList all_of_kind_feature;
481 };
482

```

```
483 interface Feature : FeatureClass, ModelElement {
484     ScopeKind owner_scope ()
485     raises (Reflective::SemanticError);
486     void set_owner_scope (in ScopeKind new_value)
487     raises (Reflective::SemanticError);
488     VisibilityKind visibility ()
489     raises (Reflective::SemanticError);
490     void set_visibility (in VisibilityKind new_value)
491     raises (Reflective::SemanticError);
492     Classifier owner ()
493     raises (Reflective::SemanticError);
494     void set_owner (in Classifier new_value)
495     raises (Reflective::SemanticError);
496 };
497
498 interface GeneralizationClass : ModelElementClass {
499     readonly attribute GeneralizationUList all_of_kind_generalization;
500     readonly attribute GeneralizationUList all_of_type_generalization;
501     Generalization create_generalization (in UmlCore::Name name,
502     in UmlCore::Name discriminator)
503     raises (Reflective::SemanticError);
504 };
505
506 interface Generalization : GeneralizationClass, ModelElement {
507     UmlCore::Name discriminator ()
508     raises (Reflective::SemanticError);
509     void set_discriminator (in UmlCore::Name new_value)
510     raises (Reflective::SemanticError);
511     GeneralizableElement subtype ()
512     raises (Reflective::SemanticError);
513     void set_subtype (in GeneralizableElement new_value)
514     raises (Reflective::SemanticError);
515     GeneralizableElement supertype ()
516     raises (Reflective::SemanticError);
517     void set_supertype (in GeneralizableElement new_value)
518     raises (Reflective::SemanticError);
519 };
520
521 interface NamespaceClass : ModelElementClass {
522     readonly attribute NamespaceUList all_of_kind_namespace;
523     readonly attribute NamespaceUList all_of_type_namespace;
```

```

524 Namespace create_namespace (in UmlCore::Name name)
525     raises (Reflective::SemanticError);
526 };
527
528 interface Namespace : NamespaceClass, ModelElement {
529     ModelElementSet owned_element ()
530     raises (Reflective::NotSet, Reflective::SemanticError);
531     void add_owned_element (in ModelElementSet new_value)
532     raises (Reflective::StructuralError, Reflective::SemanticError);
533     void remove_owned_element ()
534     raises (Reflective::SemanticError);
535     UmlCore::ElementOwnershipSet element_ownership ()
536     raises (Reflective::NotSet, Reflective::SemanticError);
537     void add_element_ownership (in UmlCore::ElementOwnershipSet new_value)
538     raises (Reflective::StructuralError, Reflective::SemanticError);
539     void remove_element_ownership ()
540     raises (Reflective::SemanticError);
541 };
542
543 interface ParameterClass : ModelElementClass {
544     readonly attribute ParameterUList all_of_kind_parameter;
545     readonly attribute ParameterUList all_of_type_parameter;
546     Parameter create_parameter (in UmlCore::Name name,
547                                in Expression default_value,
548                                in ParameterDirectionKind kind)
549     raises (Reflective::SemanticError);
550 };
551
552 interface Parameter : ParameterClass, ModelElement {
553     Expression default_value ()
554     raises (Reflective::SemanticError);
555     void set_default_value (in Expression new_value)
556     raises (Reflective::SemanticError);
557     ParameterDirectionKind kind ()
558     raises (Reflective::SemanticError);
559     void set_kind (in ParameterDirectionKind new_value)
560     raises (Reflective::SemanticError);
561     UmlCore::BehavioralFeature behavioral_feature ()
562     raises (Reflective::NotSet, Reflective::SemanticError);
563     void set_behavioral_feature (in UmlCore::BehavioralFeature new_value)
564     raises (Reflective::SemanticError);

```

```
565 void unset_behavioral_feature ()
566     raises (Reflective::SemanticError);
567 Classifier type ()
568     raises (Reflective::SemanticError);
569 void set_type (in Classifier new_value)
570     raises (Reflective::SemanticError);
571 };
572
573 interface ConstraintClass : ModelElementClass {
574     readonly attribute ConstraintUList all_of_kind_constraint;
575     readonly attribute ConstraintUList all_of_type_constraint;
576     Constraint create_constraint (in UmlCore::Name name,
577                                 in BooleanExpression body)
578     raises (Reflective::SemanticError);
579 };
580
581 interface Constraint : ConstraintClass, ModelElement {
582     BooleanExpression body ()
583     raises (Reflective::SemanticError);
584     void set_body (in BooleanExpression new_value)
585     raises (Reflective::SemanticError);
586     ModelElementUList constrained_element ()
587     raises (Reflective::SemanticError);
588     void add_constrained_element (in ModelElementUList new_value)
589     raises (Reflective::StructuralError, Reflective::SemanticError);
590     void add_constrained_element_before (in ModelElement new_value,
591                                         in ModelElement before)
592     raises (Reflective::StructuralError,
593            Reflective::NotFound,
594            Reflective::SemanticError);
595     void remove_constrained_element ()
596     raises (Reflective::SemanticError);
597 };
598
599 interface DependencyClass : ModelElementClass {
600     readonly attribute DependencyUList all_of_kind_dependency;
601 };
602
603 interface Dependency : DependencyClass, ModelElement {
604     string description ()
605     raises (Reflective::SemanticError);
```



```

606 void set_description (in string new_value)
607     raises (Reflective::SemanticError);
608 ModelElementSet supplier ()
609     raises (Reflective::NotSet, Reflective::SemanticError);
610 void add_supplier (in ModelElementSet new_value)
611     raises (Reflective::StructuralError, Reflective::SemanticError);
612 void remove_supplier ()
613     raises (Reflective::SemanticError);
614 ModelElementSet client ()
615     raises (Reflective::NotSet, Reflective::SemanticError);
616 void add_client (in ModelElementSet new_value)
617     raises (Reflective::StructuralError, Reflective::SemanticError);
618 void remove_client ()
619     raises (Reflective::SemanticError);
620 };
621
622 interface RequestClass : ModelElementClass {
623     readonly attribute RequestUList all_of_kind_request;
624     readonly attribute RequestUList all_of_type_request;
625     Request create_request (in UmlCore::Name name)
626     raises (Reflective::SemanticError);
627 };
628
629 interface Request : RequestClass, ModelElement { };
630
631 interface GeneralizableElementClass : UmlCore::NamespaceClass {
632     readonly attribute GeneralizableElementUList
633     all_of_kind_generalizable_element;
634 };
635
636 interface GeneralizableElement : GeneralizableElementClass,
637     UmlCore::Namespace {
638     boolean is_root ()
639     raises (Reflective::SemanticError);
640     void set_is_root (in boolean new_value)
641     raises (Reflective::SemanticError);
642     boolean is_leaf ()
643     raises (Reflective::SemanticError);
644     void set_is_leaf (in boolean new_value)
645     raises (Reflective::SemanticError);
646     boolean is_abstract ()

```

```
647     raises (Reflective::SemanticError);
648 void set_is_abstract (in boolean new_value)
649     raises (Reflective::SemanticError);
650 UmlCore::GeneralizationSet generalization ()
651     raises (Reflective::NotSet, Reflective::SemanticError);
652 void add_generalization (in UmlCore::GeneralizationSet new_value)
653     raises (Reflective::StructuralError, Reflective::SemanticError);
654 void remove_generalization ()
655     raises (Reflective::SemanticError);
656 UmlCore::GeneralizationSet specialization ()
657     raises (Reflective::NotSet, Reflective::SemanticError);
658 void add_specialization (in UmlCore::GeneralizationSet new_value)
659     raises (Reflective::StructuralError, Reflective::SemanticError);
660 void remove_specialization ()
661     raises (Reflective::SemanticError);
662 };
663
664 interface BehavioralFeatureClass : FeatureClass {
665     readonly attribute BehavioralFeatureUList all_of_kind_behavioral_feature;
666 };
667
668 interface BehavioralFeature : BehavioralFeatureClass, Feature {
669     boolean is_query ()
670         raises (Reflective::SemanticError);
671     void set_is_query (in boolean new_value)
672         raises (Reflective::SemanticError);
673     UmlCore::ParameterUList parameter ()
674         raises (Reflective::NotSet, Reflective::SemanticError);
675     void add_parameter (in UmlCore::ParameterUList new_value)
676         raises (Reflective::StructuralError, Reflective::SemanticError);
677     void add_parameter_before (in UmlCore::Parameter new_value,
678                               in UmlCore::Parameter before)
679         raises (Reflective::StructuralError,
680               Reflective::NotFound,
681               Reflective::SemanticError);
682     void remove_parameter ()
683         raises (Reflective::SemanticError);
684 };
685
686 interface ClassifierClass : GeneralizableElementClass {
687     readonly attribute ClassifierUList all_of_kind_classifier;
```

```
688 };
689
690 interface Classifier : ClassifierClass, GeneralizableElement {
691     UmlCore::FeatureUList feature ()
692     raises (Reflective::NotSet, Reflective::SemanticError);
693     void add_feature (in UmlCore::FeatureUList new_value)
694     raises (Reflective::StructuralError, Reflective::SemanticError);
695     void add_feature_before (in UmlCore::Feature new_value,
696                             in UmlCore::Feature before)
697     raises (Reflective::StructuralError,
698            Reflective::NotFound,
699            Reflective::SemanticError);
700     void remove_feature ()
701     raises (Reflective::SemanticError);
702     StructuralFeatureUList features ()
703     raises (Reflective::NotSet, Reflective::SemanticError);
704     void add_features (in StructuralFeatureUList new_value)
705     raises (Reflective::StructuralError, Reflective::SemanticError);
706     void add_features_before (in StructuralFeature new_value,
707                              in StructuralFeature before)
708     raises (Reflective::StructuralError,
709            Reflective::NotFound,
710            Reflective::SemanticError);
711     void remove_features ()
712     raises (Reflective::SemanticError);
713     UmlCore::ParameterUList parameter ()
714     raises (Reflective::NotSet, Reflective::SemanticError);
715     void add_parameter (in UmlCore::ParameterUList new_value)
716     raises (Reflective::StructuralError, Reflective::SemanticError);
717     void add_parameter_before (in UmlCore::Parameter new_value,
718                               in UmlCore::Parameter before)
719     raises (Reflective::StructuralError,
720            Reflective::NotFound,
721            Reflective::SemanticError);
722     void remove_parameter ()
723     raises (Reflective::SemanticError);
724     UmlCore::AssociationEndSet participant ()
725     raises (Reflective::NotSet, Reflective::SemanticError);
726     void add_participant (in UmlCore::AssociationEndSet new_value)
727     raises (Reflective::StructuralError, Reflective::SemanticError);
728     void remove_participant ()
```

```
729     raises (Reflective::SemanticError);
730 ClassifierSet realization ()
731     raises (Reflective::NotSet, Reflective::SemanticError);
732 void add_realization (in ClassifierSet new_value)
733     raises (Reflective::StructuralError, Reflective::SemanticError);
734 void remove_realization ()
735     raises (Reflective::SemanticError);
736 ClassifierSet specification ()
737     raises (Reflective::NotSet, Reflective::SemanticError);
738 void add_specification (in ClassifierSet new_value)
739     raises (Reflective::StructuralError, Reflective::SemanticError);
740 void remove_specification ()
741     raises (Reflective::SemanticError);
742 UmlCore::AssociationEndSet association_end ()
743     raises (Reflective::NotSet, Reflective::SemanticError);
744 void add_association_end (in UmlCore::AssociationEndSet new_value)
745     raises (Reflective::StructuralError, Reflective::SemanticError);
746 void remove_association_end ()
747     raises (Reflective::SemanticError);
748 };
749
750 interface OperationClass : BehavioralFeatureClass {
751     readonly attribute OperationUList all_of_kind_operation;
752     readonly attribute OperationUList all_of_type_operation;
753     Operation create_operation (in UmlCore::Name name,
754                               in ScopeKind owner_scope,
755                               in VisibilityKind visibility,
756                               in boolean is_query,
757                               in Uninterpreted specification,
758                               in boolean is_polymorphic,
759                               in CallConcurrencyKind concurrency)
760     raises (Reflective::SemanticError);
761 };
762
763 interface Operation : OperationClass, BehavioralFeature {
764     Uninterpreted specification ()
765     raises (Reflective::SemanticError);
766     void set_specification (in Uninterpreted new_value)
767     raises (Reflective::SemanticError);
768     boolean is_polymorphic ()
769     raises (Reflective::SemanticError);
```

```

770 void set_is_polymorphic (in boolean new_value)
771     raises (Reflective::SemanticError);
772 CallConcurrencyKind concurrency ()
773     raises (Reflective::SemanticError);
774 void set_concurrency (in CallConcurrencyKind new_value)
775     raises (Reflective::SemanticError);
776 UmlCore::MethodSet method ()
777     raises (Reflective::NotSet, Reflective::SemanticError);
778 void add_method (in UmlCore::MethodSet new_value)
779     raises (Reflective::StructuralError, Reflective::SemanticError);
780 void remove_method ()
781     raises (Reflective::SemanticError);
782 };
783
784 interface StereotypeClass : GeneralizableElementClass {
785     readonly attribute StereotypeUList all_of_kind_stereotype;
786     readonly attribute StereotypeUList all_of_type_stereotype;
787     Stereotype create_stereotype (in UmlCore::Name name,
788         in boolean is_root,
789         in boolean is_leaf,
790         in boolean is_abstract,
791         in Geometry icon)
792     raises (Reflective::SemanticError);
793 };
794
795 interface Stereotype : StereotypeClass, GeneralizableElement {
796     Geometry icon ()
797     raises (Reflective::SemanticError);
798     void set_icon (in Geometry new_value)
799     raises (Reflective::SemanticError);
800     UmlCore::TaggedValueSet required_tag ()
801     raises (Reflective::NotSet, Reflective::SemanticError);
802     void add_required_tag (in UmlCore::TaggedValueSet new_value)
803     raises (Reflective::StructuralError, Reflective::SemanticError);
804     void remove_required_tag ()
805     raises (Reflective::SemanticError);
806     ModelElementSet extended_element ()
807     raises (Reflective::NotSet, Reflective::SemanticError);
808     void add_extended_element (in ModelElementSet new_value)
809     raises (Reflective::StructuralError, Reflective::SemanticError);
810     void remove_extended_element ()

```

```
811     raises (Reflective::SemanticError);
812     UmlCore::ConstraintSet stereotype_constraint ()
813     raises (Reflective::NotSet, Reflective::SemanticError);
814     void add_stereotype_constraint (in UmlCore::ConstraintSet new_value)
815     raises (Reflective::StructuralError, Reflective::SemanticError);
816     void remove_stereotype_constraint ()
817     raises (Reflective::SemanticError);
818 };
819
820 interface StructuralFeatureClass : FeatureClass {
821     readonly attribute StructuralFeatureUList all_of_kind_structural_feature;
822 };
823
824 interface StructuralFeature : StructuralFeatureClass, Feature {
825     UmlCore::Multiplicity multiplicity ()
826     raises (Reflective::SemanticError);
827     void set_multiplicity (in UmlCore::Multiplicity new_value)
828     raises (Reflective::SemanticError);
829     ChangeableKind changeable ()
830     raises (Reflective::SemanticError);
831     void set_changeable (in ChangeableKind new_value)
832     raises (Reflective::SemanticError);
833     ScopeKind target_scope ()
834     raises (Reflective::SemanticError);
835     void set_target_scope (in ScopeKind new_value)
836     raises (Reflective::SemanticError);
837     Classifier type ()
838     raises (Reflective::SemanticError);
839     void set_type (in Classifier new_value)
840     raises (Reflective::SemanticError);
841 };
842
843 interface DataTypeClass : ClassifierClass {
844     readonly attribute DataTypeUList all_of_kind_data_type;
845     readonly attribute DataTypeUList all_of_type_data_type;
846     DataType create_data_type (in UmlCore::Name name,
847                               in boolean is_root,
848                               in boolean is_leaf,
849                               in boolean is_abstract)
850     raises (Reflective::SemanticError);
851 };
```

```

852
853 interface DataType : DataTypeClass, Classifier { };
854
855 interface UmlInterfaceClass : ClassifierClass {
856     readonly attribute UmlInterfaceUList all_of_kind_uml_interface;
857     readonly attribute UmlInterfaceUList all_of_type_uml_interface;
858     UmlInterface create_uml_interface (in UmlCore::Name name,
859                                     in boolean is_root,
860                                     in boolean is_leaf,
861                                     in boolean is_abstract)
862     raises (Reflective::SemanticError);
863 };
864
865 interface UmlInterface : UmlInterfaceClass, Classifier { };
866
867 interface UmlAttributeClass : StructuralFeatureClass {
868     readonly attribute UmlAttributeUList all_of_kind_uml_attribute;
869     readonly attribute UmlAttributeUList all_of_type_uml_attribute;
870     UmlAttribute create_uml_attribute (in UmlCore::Name name,
871                                     in ScopeKind owner_scope,
872                                     in VisibilityKind visibility,
873                                     in UmlCore::Multiplicity multiplicity,
874                                     in ChangeableKind changeable,
875                                     in ScopeKind target_scope,
876                                     in Expression initial_value)
877     raises (Reflective::SemanticError);
878 };
879
880 interface UmlAttribute : UmlAttributeClass, StructuralFeature {
881     Expression initial_value ()
882     raises (Reflective::SemanticError);
883     void set_initial_value (in Expression new_value)
884     raises (Reflective::SemanticError);
885     UmlCore::AssociationEnd association_end ()
886     raises (Reflective::NotSet, Reflective::SemanticError);
887     void set_association_end (in UmlCore::AssociationEnd new_value)
888     raises (Reflective::SemanticError);
889     void unset_association_end ()
890     raises (Reflective::SemanticError);
891 };
892

```

```
893 interface AssociationEndClass : ModelElementClass {
894     readonly attribute AssociationEndUList all_of_kind_association_end;
895     readonly attribute AssociationEndUList all_of_type_association_end;
896     AssociationEnd create_association_end (
897         in UmlCore::Name name,
898         in boolean is_navigable,
899         in boolean is_ordered,
900         in AggregationKind aggregation,
901         in ScopeKind target_scope,
902         in UmlCore::Multiplicity multiplicity,
903         in ChangeableKind changeable)
904     raises (Reflective::SemanticError);
905 };
906
907 interface AssociationEnd : AssociationEndClass, ModelElement {
908     boolean is_navigable ()
909     raises (Reflective::SemanticError);
910     void set_is_navigable (in boolean new_value)
911     raises (Reflective::SemanticError);
912     boolean is_ordered ()
913     raises (Reflective::SemanticError);
914     void set_is_ordered (in boolean new_value)
915     raises (Reflective::SemanticError);
916     AggregationKind aggregation ()
917     raises (Reflective::SemanticError);
918     void set_aggregation (in AggregationKind new_value)
919     raises (Reflective::SemanticError);
920     ScopeKind target_scope ()
921     raises (Reflective::SemanticError);
922     void set_target_scope (in ScopeKind new_value)
923     raises (Reflective::SemanticError);
924     UmlCore::Multiplicity multiplicity ()
925     raises (Reflective::SemanticError);
926     void set_multiplicity (in UmlCore::Multiplicity new_value)
927     raises (Reflective::SemanticError);
928     ChangeableKind changeable ()
929     raises (Reflective::SemanticError);
930     void set_changeable (in ChangeableKind new_value)
931     raises (Reflective::SemanticError);
932     UmlCore::Association association ()
933     raises (Reflective::SemanticError);
```



```

934 void set_association (in UmlCore::Association new_value)
935     raises (Reflective::SemanticError);
936 UmlAttributeUList qualifier ()
937     raises (Reflective::NotSet, Reflective::SemanticError);
938 void add_qualifier (in UmlAttributeUList new_value)
939     raises (Reflective::StructuralError, Reflective::SemanticError);
940 void add_qualifier_before (in UmlAttribute new_value,
941                          in UmlAttribute before)
942     raises (Reflective::StructuralError,
943           Reflective::NotFound,
944           Reflective::SemanticError);
945 void remove_qualifier ()
946     raises (Reflective::SemanticError);
947 Classifier type2 ()
948     raises (Reflective::SemanticError);
949 void set_type2 (in Classifier new_value)
950     raises (Reflective::SemanticError);
951 ClassifierSet specification ()
952     raises (Reflective::NotSet, Reflective::SemanticError);
953 void add_specification (in ClassifierSet new_value)
954     raises (Reflective::StructuralError, Reflective::SemanticError);
955 void remove_specification ()
956     raises (Reflective::SemanticError);
957 };
958
959 interface AssociationClass : GeneralizableElementClass {
960     readonly attribute AssociationUList all_of_kind_association;
961     readonly attribute AssociationUList all_of_type_association;
962     Association create_association (in UmlCore::Name name,
963                                   in boolean is_root,
964                                   in boolean is_leaf,
965                                   in boolean is_abstract)
966     raises (Reflective::SemanticError);
967 };
968
969 interface Association : AssociationClass, GeneralizableElement {
970     AssociationEndUList connection ()
971     raises (Reflective::SemanticError);
972     void add_connection (in AssociationEndUList new_value)
973     raises (Reflective::StructuralError, Reflective::SemanticError);
974     void add_connection_before (in AssociationEnd new_value,

```

```
975         in AssociationEnd before)
976     raises (Reflective::StructuralError,
977           Reflective::NotFound,
978           Reflective::SemanticError);
979 void modify_connection (in AssociationEnd old_value,
980                       in AssociationEnd new_value)
981     raises (Reflective::StructuralError,
982           Reflective::NotFound,
983           Reflective::SemanticError);
984 void remove_connection ()
985     raises (Reflective::StructuralError, Reflective::SemanticError);
986 };
987
988 interface MethodClass : BehavioralFeatureClass {
989     readonly attribute MethodUList all_of_kind_method;
990     readonly attribute MethodUList all_of_type_method;
991     Method create_method (in UmlCore::Name name,
992                          in ScopeKind owner_scope,
993                          in VisibilityKind visibility,
994                          in boolean is_query,
995                          in ProcedureExpression body)
996     raises (Reflective::SemanticError);
997 };
998
999 interface Method : MethodClass, BehavioralFeature {
1000     ProcedureExpression body ()
1001     raises (Reflective::SemanticError);
1002     void set_body (in ProcedureExpression new_value)
1003     raises (Reflective::SemanticError);
1004     Operation specification ()
1005     raises (Reflective::SemanticError);
1006     void set_specification (in Operation new_value)
1007     raises (Reflective::SemanticError);
1008 };
1009
1010 interface EnumerationClass : DataTypeClass {
1011     readonly attribute EnumerationUList all_of_kind_enumeration;
1012     readonly attribute EnumerationUList all_of_type_enumeration;
1013     Enumeration create_enumeration (in UmlCore::Name name,
1014                                    in boolean is_root,
1015                                    in boolean is_leaf,
```

```
1016             in boolean is_abstract)
1017     raises (Reflective::SemanticError);
1018 };
1019
1020 interface Enumeration : EnumerationClass, DataType {
1021     EnumerationLiteralUList literal ()
1022     raises (Reflective::SemanticError);
1023     void add_literal (in EnumerationLiteralUList new_value)
1024     raises (Reflective::StructuralError, Reflective::SemanticError);
1025     void add_literal_before (in EnumerationLiteral new_value,
1026                             in EnumerationLiteral before)
1027     raises (Reflective::StructuralError,
1028            Reflective::NotFound,
1029            Reflective::SemanticError);
1030     void remove_literal ()
1031     raises (Reflective::SemanticError);
1032 };
1033
1034 interface ClassClass : ClassifierClass {
1035     readonly attribute ClassUList all_of_kind_class;
1036     readonly attribute ClassUList all_of_type_class;
1037     Class create_class (in UmlCore::Name name,
1038                       in boolean is_root,
1039                       in boolean is_leaf,
1040                       in boolean is_abstract,
1041                       in boolean is_active)
1042     raises (Reflective::SemanticError);
1043 };
1044
1045 interface Class : ClassClass, Classifier {
1046     boolean is_active ()
1047     raises (Reflective::SemanticError);
1048     void set_is_active (in boolean new_value)
1049     raises (Reflective::SemanticError);
1050 };
1051
1052 interface PrimitiveClass : DataTypeClass {
1053     readonly attribute PrimitiveUList all_of_kind_primitive;
1054     readonly attribute PrimitiveUList all_of_type_primitive;
1055     Primitive create_primitive (in UmlCore::Name name,
1056                               in boolean is_root,
```

```
1057             in boolean is_leaf,
1058             in boolean is_abstract)
1059     raises (Reflective::SemanticError);
1060 };
1061
1062 interface Primitive : PrimitiveClass, DataType { };
1063
1064 interface StructureClass : DataTypeClass {
1065     readonly attribute StructureUList all_of_kind_structure;
1066     readonly attribute StructureUList all_of_type_structure;
1067     Structure create_structure (in UmlCore::Name name,
1068                               in boolean is_root,
1069                               in boolean is_leaf,
1070                               in boolean is_abstract)
1071     raises (Reflective::SemanticError);
1072 };
1073
1074 interface Structure : StructureClass, DataType { };
1075
1076 interface UmlAssociationClassClass : ClassClass, AssociationClass {
1077     readonly attribute UmlAssociationClassUList
1078         all_of_kind_uml_association_class;
1079     readonly attribute UmlAssociationClassUList
1080         all_of_type_uml_association_class;
1081     UmlAssociationClass create_uml_association_class (in UmlCore::Name name,
1082                                                       in boolean is_root,
1083                                                       in boolean is_leaf,
1084                                                       in boolean is_abstract,
1085                                                       in boolean is_active)
1086     raises (Reflective::SemanticError);
1087 };
1088
1089 interface UmlAssociationClass : UmlAssociationClassClass,
1090                               Class, Association { };
1091
1092 interface ElementOwnershipClass : ElementClass {
1093     readonly attribute ElementOwnershipUList all_of_kind_element_ownership;
1094     readonly attribute ElementOwnershipUList all_of_type_element_ownership;
1095     ElementOwnership create_element_ownership (in VisibilityKind visibilty)
1096     raises (Reflective::SemanticError);
1097 };
```

```

1098
1099 interface ElementOwnership : ElementOwnershipClass, Element {
1100     VisibilityKind visibilty ()
1101     raises (Reflective::SemanticError);
1102     void set_visibilty (in VisibilityKind new_value)
1103     raises (Reflective::SemanticError);
1104     UmlCore::Namespace namespace ()
1105     raises (Reflective::SemanticError);
1106     void set_namespace (in UmlCore::Namespace new_value)
1107     raises (Reflective::SemanticError);
1108     ModelElement owned_element ()
1109     raises (Reflective::SemanticError);
1110     void set_owned_element (in ModelElement new_value)
1111     raises (Reflective::SemanticError);
1112 };
1113
1114 struct AssociationOwnsAssociationEndLink {
1115     UmlCore::Association association;
1116     AssociationEnd connection;
1117 };
1118 typedef sequence <AssociationOwnsAssociationEndLink>
1119     AssociationOwnsAssociationEndLinkSet;
1120
1121 interface AssociationOwnsAssociationEnd : Reflective::RefAssociation {
1122     readonly attribute UmlCorePackage enclosing_package_ref;
1123     AssociationOwnsAssociationEndLinkSet
1124     all_association_owns_association_end_links();
1125     boolean exists (in UmlCore::Association association,
1126                   in AssociationEnd connection);
1127     UmlCore::Association with_connection (in AssociationEnd connection);
1128     AssociationEndUList with_association (in UmlCore::Association association);
1129     void add (in UmlCore::Association association,
1130              in AssociationEnd connection)
1131     raises (Reflective::StructuralError, Reflective::SemanticError);
1132     void add_before_connection (in UmlCore::Association association,
1133                                in AssociationEnd connection,
1134                                in AssociationEnd before)
1135     raises (Reflective::StructuralError,
1136            Reflective::SemanticError,
1137            Reflective::NotFound);
1138     void modify_association (in UmlCore::Association association,

```

```
1139             in AssociationEnd connection,
1140             in UmlCore::Association new_association)
1141     raises (Reflective::StructuralError,
1142           Reflective::SemanticError,
1143           Reflective::NotFound);
1144 void modify_connection (in UmlCore::Association association,
1145                       in AssociationEnd connection,
1146                       in AssociationEnd new_connection)
1147     raises (Reflective::StructuralError,
1148           Reflective::SemanticError,
1149           Reflective::NotFound);
1150 void remove (in UmlCore::Association association,
1151            in AssociationEnd connection)
1152     raises (Reflective::StructuralError,
1153           Reflective::SemanticError,
1154           Reflective::NotFound);
1155 };
1156
1157 struct ClassifierOwnsFeatureLink {
1158     Classifier owner;
1159     UmlCore::Feature feature;
1160 };
1161 typedef sequence <ClassifierOwnsFeatureLink> ClassifierOwnsFeatureLinkSet;
1162
1163 interface ClassifierOwnsFeature : Reflective::RefAssociation {
1164     readonly attribute UmlCorePackage enclosing_package_ref;
1165     ClassifierOwnsFeatureLinkSet all_classifier_owns_feature_links();
1166     boolean exists (in Classifier owner, in UmlCore::Feature feature);
1167     Classifier with_feature (in UmlCore::Feature feature);
1168     UmlCore::FeatureUList with_owner (in Classifier owner);
1169     void add (in Classifier owner, in UmlCore::Feature feature)
1170         raises (Reflective::StructuralError, Reflective::SemanticError);
1171     void add_before_feature (in Classifier owner,
1172                          in UmlCore::Feature feature,
1173                          in UmlCore::Feature before)
1174         raises (Reflective::StructuralError,
1175               Reflective::SemanticError,
1176               Reflective::NotFound);
1177     void modify_owner (in Classifier owner,
1178                    in UmlCore::Feature feature,
1179                    in Classifier new_owner)
```

```

1180     raises (Reflective::StructuralError,
1181             Reflective::SemanticError,
1182             Reflective::NotFound);
1183 void modify_feature (in Classifier owner,
1184                     in UmlCore::Feature feature,
1185                     in UmlCore::Feature new_feature)
1186     raises (Reflective::StructuralError,
1187             Reflective::SemanticError,
1188             Reflective::NotFound);
1189 void remove (in Classifier owner, in UmlCore::Feature feature)
1190     raises (Reflective::StructuralError,
1191             Reflective::SemanticError,
1192             Reflective::NotFound);
1193 };
1194
1195 struct MethodIsSpecifiedByOperationLink {
1196     Operation specification;
1197     UmlCore::Method method;
1198 };
1199 typedef sequence <MethodIsSpecifiedByOperationLink>
1200     MethodIsSpecifiedByOperationLinkSet;
1201
1202 interface MethodIsSpecifiedByOperation : Reflective::RefAssociation {
1203     readonly attribute UmlCorePackage enclosing_package_ref;
1204     MethodIsSpecifiedByOperationLinkSet
1205     all_method_is_specified_by_operation_links();
1206     boolean exists (in Operation specification, in UmlCore::Method method);
1207     Operation with_method (in UmlCore::Method method);
1208     UmlCore::MethodSet with_specification (in Operation specification);
1209     void add (in Operation specification, in UmlCore::Method method)
1210         raises (Reflective::StructuralError, Reflective::SemanticError);
1211     void modify_specification (in Operation specification,
1212                               in UmlCore::Method method,
1213                               in Operation new_specification)
1214         raises (Reflective::StructuralError,
1215                 Reflective::SemanticError,
1216                 Reflective::NotFound);
1217     void modify_method (in Operation specification,
1218                        in UmlCore::Method method,
1219                        in UmlCore::Method new_method)
1220         raises (Reflective::StructuralError,

```

```
1221         Reflective::SemanticError,
1222         Reflective::NotFound);
1223 void remove (in Operation specification, in UmlCore::Method method)
1224     raises (Reflective::StructuralError,
1225           Reflective::SemanticError,
1226           Reflective::NotFound);
1227 };
1228
1229 struct StructuralFeatureIsOfTypeClassifierLink {
1230     StructuralFeature features;
1231     Classifier type;
1232 };
1233 typedef sequence <StructuralFeatureIsOfTypeClassifierLink>
1234     StructuralFeatureIsOfTypeClassifierLinkSet;
1235
1236 interface StructuralFeatureIsOfTypeClassifier : Reflective::RefAssociation {
1237     readonly attribute UmlCorePackage enclosing_package_ref;
1238     StructuralFeatureIsOfTypeClassifierLinkSet
1239     all_structural_feature_is_of_type_classifier_links();
1240     boolean exists (in StructuralFeature features, in Classifier type);
1241     StructuralFeatureUList with_type (in Classifier type);
1242     Classifier with_features (in StructuralFeature features);
1243     void add (in StructuralFeature features, in Classifier type)
1244         raises (Reflective::StructuralError, Reflective::SemanticError);
1245     void add_before_features (in StructuralFeature features,
1246                             in Classifier type,
1247                             in StructuralFeature before)
1248         raises (Reflective::StructuralError,
1249               Reflective::SemanticError,
1250               Reflective::NotFound);
1251     void modify_features (in StructuralFeature features,
1252                         in Classifier type,
1253                         in StructuralFeature new_features)
1254         raises (Reflective::StructuralError,
1255               Reflective::SemanticError,
1256               Reflective::NotFound);
1257     void modify_type (in StructuralFeature features,
1258                    in Classifier type,
1259                    in Classifier new_type)
1260         raises (Reflective::StructuralError,
1261               Reflective::SemanticError,
```



```

1262         Reflective::NotFound);
1263     void remove (in StructuralFeature features, in Classifier type)
1264         raises (Reflective::StructuralError,
1265             Reflective::SemanticError,
1266             Reflective::NotFound);
1267 };
1268
1269 struct NamespaceOwnsModelElementLink {
1270     UmlCore::Namespace namespace;
1271     ModelElement owned_element;
1272 };
1273 typedef sequence <NamespaceOwnsModelElementLink>
1274     NamespaceOwnsModelElementLinkSet;
1275
1276 interface NamespaceOwnsModelElement : Reflective::RefAssociation {
1277     readonly attribute UmlCorePackage enclosing_package_ref;
1278     NamespaceOwnsModelElementLinkSet all_namespace_owns_model_element_links();
1279     boolean exists (in UmlCore::Namespace namespace,
1280         in ModelElement owned_element);
1281     UmlCore::Namespace with_owned_element (in ModelElement owned_element);
1282     ModelElementSet with_namespace (in UmlCore::Namespace namespace);
1283     void add (in UmlCore::Namespace namespace, in ModelElement owned_element)
1284         raises (Reflective::StructuralError, Reflective::SemanticError);
1285     void modify_namespace (in UmlCore::Namespace namespace,
1286         in ModelElement owned_element,
1287         in UmlCore::Namespace new_namespace)
1288         raises (Reflective::StructuralError,
1289             Reflective::SemanticError,
1290             Reflective::NotFound);
1291     void modify_owned_element (in UmlCore::Namespace namespace,
1292         in ModelElement owned_element,
1293         in ModelElement new_owned_element)
1294         raises (Reflective::StructuralError,
1295             Reflective::SemanticError,
1296             Reflective::NotFound);
1297     void remove (in UmlCore::Namespace namespace,
1298         in ModelElement owned_element)
1299         raises (Reflective::StructuralError,
1300             Reflective::SemanticError,
1301             Reflective::NotFound);
1302 };

```

```
1303
1304 struct BehavioralFeatureOwnsParameterLink {
1305     UmlCore::BehavioralFeature behavioral_feature;
1306     UmlCore::Parameter parameter;
1307 };
1308 typedef sequence <BehavioralFeatureOwnsParameterLink>
1309     BehavioralFeatureOwnsParameterLinkSet;
1310
1311 interface BehavioralFeatureOwnsParameter : Reflective::RefAssociation {
1312     readonly attribute UmlCorePackage enclosing_package_ref;
1313     BehavioralFeatureOwnsParameterLinkSet
1314     all_behavioral_feature_owns_parameter_links();
1315     boolean exists (in UmlCore::BehavioralFeature behavioral_feature,
1316                   in UmlCore::Parameter parameter);
1317     UmlCore::BehavioralFeature with_parameter (
1318         in UmlCore::Parameter parameter);
1319     UmlCore::ParameterUList with_behavioral_feature (
1320         in UmlCore::BehavioralFeature behavioral_feature);
1321     void add (in UmlCore::BehavioralFeature behavioral_feature,
1322             in UmlCore::Parameter parameter)
1323         raises (Reflective::StructuralError, Reflective::SemanticError);
1324     void add_before_parameter (
1325         in UmlCore::BehavioralFeature behavioral_feature,
1326         in UmlCore::Parameter parameter,
1327         in UmlCore::Parameter before)
1328         raises (Reflective::StructuralError,
1329             Reflective::SemanticError,
1330             Reflective::NotFound);
1331     void modify_behavioral_feature (
1332         in UmlCore::BehavioralFeature behavioral_feature,
1333         in UmlCore::Parameter parameter,
1334         in UmlCore::BehavioralFeature new_behavioral_feature)
1335         raises (Reflective::StructuralError,
1336             Reflective::SemanticError,
1337             Reflective::NotFound);
1338     void modify_parameter (in UmlCore::BehavioralFeature behavioral_feature,
1339                           in UmlCore::Parameter parameter,
1340                           in UmlCore::Parameter new_parameter)
1341         raises (Reflective::StructuralError,
1342             Reflective::SemanticError,
1343             Reflective::NotFound);
```

```

1344 void remove (in UmlCore::BehavioralFeature behavioral_feature,
1345               in UmlCore::Parameter parameter)
1346   raises (Reflective::StructuralError,
1347          Reflective::SemanticError,
1348          Reflective::NotFound);
1349 };
1350
1351 struct ParameterIsOfTypeClassifierLink {
1352   Classifier type;
1353   UmlCore::Parameter parameter;
1354 };
1355 typedef sequence <ParameterIsOfTypeClassifierLink>
1356   ParameterIsOfTypeClassifierLinkSet;
1357
1358 interface ParameterIsOfTypeClassifier : Reflective::RefAssociation {
1359   readonly attribute UmlCorePackage enclosing_package_ref;
1360   ParameterIsOfTypeClassifierLinkSet
1361     all_parameter_is_of_type_classifier_links();
1362   boolean exists (in Classifier type, in UmlCore::Parameter parameter);
1363   Classifier with_parameter (in UmlCore::Parameter parameter);
1364   UmlCore::ParameterUList with_type (in Classifier type);
1365   void add (in Classifier type, in UmlCore::Parameter parameter)
1366     raises (Reflective::StructuralError, Reflective::SemanticError);
1367   void add_before_parameter (in Classifier type,
1368                             in UmlCore::Parameter parameter,
1369                             in UmlCore::Parameter before)
1370     raises (Reflective::StructuralError,
1371            Reflective::SemanticError,
1372            Reflective::NotFound);
1373   void modify_type (in Classifier type,
1374                   in UmlCore::Parameter parameter,
1375                   in Classifier new_type)
1376     raises (Reflective::StructuralError,
1377            Reflective::SemanticError,
1378            Reflective::NotFound);
1379   void modify_parameter (in Classifier type,
1380                         in UmlCore::Parameter parameter,
1381                         in UmlCore::Parameter new_parameter)
1382     raises (Reflective::StructuralError,
1383            Reflective::SemanticError,
1384            Reflective::NotFound);

```

```
1385 void remove (in Classifier type, in UmlCore::Parameter parameter)
1386     raises (Reflective::StructuralError,
1387           Reflective::SemanticError,
1388           Reflective::NotFound);
1389 };
1390
1391 struct GeneralizableElementIsSubtypeInGeneralizationLink {
1392     GeneralizableElement subtype;
1393     UmlCore::Generalization generalization;
1394 };
1395 typedef sequence <GeneralizableElementIsSubtypeInGeneralizationLink>
1396     GeneralizableElementIsSubtypeInGeneralizationLinkSet;
1397
1398 interface GeneralizableElementIsSubtypeInGeneralization :
1399     Reflective::RefAssociation {
1400     readonly attribute UmlCorePackage enclosing_package_ref;
1401     GeneralizableElementIsSubtypeInGeneralizationLinkSet
1402     all_generalizable_element_is_subtype_in_generalization_links();
1403     boolean exists (in GeneralizableElement subtype,
1404                   in UmlCore::Generalization generalization);
1405     GeneralizableElement with_generalization (
1406         in UmlCore::Generalization generalization);
1407     UmlCore::GeneralizationSet with_subtype (in GeneralizableElement subtype);
1408     void add (in GeneralizableElement subtype,
1409             in UmlCore::Generalization generalization)
1410     raises (Reflective::StructuralError, Reflective::SemanticError);
1411     void modify_subtype (in GeneralizableElement subtype,
1412                         in UmlCore::Generalization generalization,
1413                         in GeneralizableElement new_subtype)
1414     raises (Reflective::StructuralError,
1415           Reflective::SemanticError,
1416           Reflective::NotFound);
1417     void modify_generalization (in GeneralizableElement subtype,
1418                               in UmlCore::Generalization generalization,
1419                               in UmlCore::Generalization new_generalization)
1420     raises (Reflective::StructuralError,
1421           Reflective::SemanticError,
1422           Reflective::NotFound);
1423     void remove (in GeneralizableElement subtype,
1424                in UmlCore::Generalization generalization)
1425     raises (Reflective::StructuralError,
```

```
1426         Reflective::SemanticError,
1427         Reflective::NotFound);
1428     };
1429
1430     struct GeneralizableElementIsSupertypeInGeneralizationLink {
1431         GeneralizableElement supertype;
1432         Generalization specialization;
1433     };
1434     typedef sequence <GeneralizableElementIsSupertypeInGeneralizationLink>
1435         GeneralizableElementIsSupertypeInGeneralizationLinkSet;
1436
1437     interface GeneralizableElementIsSupertypeInGeneralization :
1438         Reflective::RefAssociation {
1439         readonly attribute UmlCorePackage enclosing_package_ref;
1440         GeneralizableElementIsSupertypeInGeneralizationLinkSet
1441         all_generalizable_element_is_supertype_in_generalization_links();
1442         boolean exists (in GeneralizableElement supertype,
1443             in Generalization specialization);
1444         GeneralizableElement with_specialization (
1445             in Generalization specialization);
1446         GeneralizationSet with_supertype (in GeneralizableElement supertype);
1447         void add (in GeneralizableElement supertype,
1448             in Generalization specialization)
1449             raises (Reflective::StructuralError, Reflective::SemanticError);
1450         void modify_supertype (in GeneralizableElement supertype,
1451             in Generalization specialization,
1452             in GeneralizableElement new_supertype)
1453             raises (Reflective::StructuralError,
1454                 Reflective::SemanticError,
1455                 Reflective::NotFound);
1456         void modify_specialization (in GeneralizableElement supertype,
1457             in Generalization specialization,
1458             in Generalization new_specialization)
1459             raises (Reflective::StructuralError,
1460                 Reflective::SemanticError,
1461                 Reflective::NotFound);
1462         void remove (in GeneralizableElement supertype,
1463             in Generalization specialization)
1464             raises (Reflective::StructuralError,
1465                 Reflective::SemanticError,
1466                 Reflective::NotFound);
```

```
1467 };
1468
1469 struct AssociationEndOwnsQualifierAttributeLink {
1470     UmlAttribute qualifier;
1471     UmlCore::AssociationEnd association_end;
1472 };
1473 typedef sequence <AssociationEndOwnsQualifierAttributeLink>
1474     AssociationEndOwnsQualifierAttributeLinkSet;
1475
1476 interface AssociationEndOwnsQualifierAttribute : Reflective::RefAssociation {
1477     readonly attribute UmlCorePackage enclosing_package_ref;
1478     AssociationEndOwnsQualifierAttributeLinkSet
1479     all_association_end_owns_qualifier_attribute_links();
1480     boolean exists (in UmlAttribute qualifier,
1481                   in UmlCore::AssociationEnd association_end);
1482     UmlAttributeUList with_association_end (
1483         in UmlCore::AssociationEnd association_end);
1484     UmlCore::AssociationEnd with_qualifier (in UmlAttribute qualifier);
1485     void add (in UmlAttribute qualifier,
1486             in UmlCore::AssociationEnd association_end)
1487         raises (Reflective::StructuralError, Reflective::SemanticError);
1488     void add_before_qualifier (in UmlAttribute qualifier,
1489                               in UmlCore::AssociationEnd association_end,
1490                               in UmlAttribute before)
1491         raises (Reflective::StructuralError,
1492               Reflective::SemanticError,
1493               Reflective::NotFound);
1494     void modify_qualifier (in UmlAttribute qualifier,
1495                           in UmlCore::AssociationEnd association_end,
1496                           in UmlAttribute new_qualifier)
1497         raises (Reflective::StructuralError,
1498               Reflective::SemanticError,
1499               Reflective::NotFound);
1500     void modify_association_end (
1501         in UmlAttribute qualifier,
1502         in UmlCore::AssociationEnd association_end,
1503         in UmlCore::AssociationEnd new_association_end)
1504         raises (Reflective::StructuralError,
1505               Reflective::SemanticError,
1506               Reflective::NotFound);
1507     void remove (in UmlAttribute qualifier,
```

```
1508         in UmlCore::AssociationEnd association_end)
1509     raises (Reflective::StructuralError,
1510           Reflective::SemanticError,
1511           Reflective::NotFound);
1512 };
1513
1514 struct AssociationEndIsOfTypeClassifierLink {
1515     Classifier type2;
1516     AssociationEnd participant;
1517 };
1518 typedef sequence <AssociationEndIsOfTypeClassifierLink>
1519     AssociationEndIsOfTypeClassifierLinkSet;
1520
1521 interface AssociationEndIsOfTypeClassifier : Reflective::RefAssociation {
1522     readonly attribute UmlCorePackage enclosing_package_ref;
1523     AssociationEndIsOfTypeClassifierLinkSet
1524     all_association_end_is_of_type_classifier_links();
1525     boolean exists (in Classifier type2, in AssociationEnd participant);
1526     Classifier with_participant (in AssociationEnd participant);
1527     AssociationEndSet with_type2 (in Classifier type2);
1528     void add (in Classifier type2, in AssociationEnd participant)
1529         raises (Reflective::StructuralError, Reflective::SemanticError);
1530     void modify_type2 (in Classifier type2,
1531                       in AssociationEnd participant,
1532                       in Classifier new_type2)
1533         raises (Reflective::StructuralError,
1534               Reflective::SemanticError,
1535               Reflective::NotFound);
1536     void modify_participant (in Classifier type2,
1537                             in AssociationEnd participant,
1538                             in AssociationEnd new_participant)
1539         raises (Reflective::StructuralError,
1540               Reflective::SemanticError,
1541               Reflective::NotFound);
1542     void remove (in Classifier type2, in AssociationEnd participant)
1543         raises (Reflective::StructuralError,
1544               Reflective::SemanticError,
1545               Reflective::NotFound);
1546 };
1547
1548 struct ClassifierIsRealizedByClassifierLink {
```

```
1549 Classifier realization;
1550 Classifier specification;
1551 };
1552 typedef sequence <ClassifierIsRealizedByClassifierLink>
1553 ClassifierIsRealizedByClassifierLinkSet;
1554
1555 interface ClassifierIsRealizedByClassifier : Reflective::RefAssociation {
1556     readonly attribute UmlCorePackage enclosing_package_ref;
1557     ClassifierIsRealizedByClassifierLinkSet
1558     all_classifier_is_realized_by_classifier_links();
1559     boolean exists (in Classifier realization, in Classifier specification);
1560     ClassifierSet with_specification (in Classifier specification);
1561     ClassifierSet with_realization (in Classifier realization);
1562     void add (in Classifier realization, in Classifier specification)
1563     raises (Reflective::StructuralError, Reflective::SemanticError);
1564     void modify_realization (in Classifier realization,
1565                             in Classifier specification,
1566                             in Classifier new_realization)
1567     raises (Reflective::StructuralError,
1568            Reflective::SemanticError,
1569            Reflective::NotFound);
1570     void modify_specification (in Classifier realization,
1571                               in Classifier specification,
1572                               in Classifier new_specification)
1573     raises (Reflective::StructuralError,
1574            Reflective::SemanticError,
1575            Reflective::NotFound);
1576     void remove (in Classifier realization, in Classifier specification)
1577     raises (Reflective::StructuralError,
1578            Reflective::SemanticError,
1579            Reflective::NotFound);
1580 };
1581
1582 struct AssociationEndIsSpecifiedByClassifierLink {
1583     UmlCore::AssociationEnd association_end;
1584     Classifier specification;
1585 };
1586 typedef sequence <AssociationEndIsSpecifiedByClassifierLink>
1587 AssociationEndIsSpecifiedByClassifierLinkSet;
1588
1589 interface AssociationEndIsSpecifiedByClassifier :
```



```

1590             Reflective::RefAssociation {
1591     readonly attribute UmlCorePackage enclosing_package_ref;
1592     AssociationEndIsSpecifiedByClassifierLinkSet
1593     all_association_end_is_specified_by_classifier_links();
1594     boolean exists (in UmlCore::AssociationEnd association_end,
1595                   in Classifier specification);
1596     UmlCore::AssociationEndSet with_specification (
1597                   in Classifier specification);
1598     ClassifierSet with_association_end (
1599                   in UmlCore::AssociationEnd association_end);
1600     void add (in UmlCore::AssociationEnd association_end,
1601             in Classifier specification)
1602     raises (Reflective::StructuralError, Reflective::SemanticError);
1603     void modify_association_end (
1604                   in UmlCore::AssociationEnd association_end,
1605                   in Classifier specification,
1606                   in UmlCore::AssociationEnd new_association_end)
1607     raises (Reflective::StructuralError,
1608           Reflective::SemanticError,
1609           Reflective::NotFound);
1610     void modify_specification (in UmlCore::AssociationEnd association_end,
1611                              in Classifier specification,
1612                              in Classifier new_specification)
1613     raises (Reflective::StructuralError,
1614           Reflective::SemanticError,
1615           Reflective::NotFound);
1616     void remove (in UmlCore::AssociationEnd association_end,
1617                in Classifier specification)
1618     raises (Reflective::StructuralError,
1619           Reflective::SemanticError,
1620           Reflective::NotFound);
1621 };
1622
1623 struct ModelElementIsSupplierInDependencyLink {
1624     ModelElement supplier;
1625     Dependency provision;
1626 };
1627 typedef sequence <ModelElementIsSupplierInDependencyLink>
1628     ModelElementIsSupplierInDependencyLinkSet;
1629
1630 interface ModelElementIsSupplierInDependency : Reflective::RefAssociation {

```

```
1631     readonly attribute UmlCorePackage enclosing_package_ref;
1632     ModelElementIsSupplierInDependencyLinkSet
1633     all_model_element_is_supplier_in_dependency_links();
1634     boolean exists (in ModelElement supplier, in Dependency provision);
1635     ModelElementSet with_provision (in Dependency provision);
1636     DependencySet with_supplier (in ModelElement supplier);
1637     void add (in ModelElement supplier, in Dependency provision)
1638         raises (Reflective::StructuralError, Reflective::SemanticError);
1639     void modify_supplier (in ModelElement supplier,
1640                          in Dependency provision,
1641                          in ModelElement new_supplier)
1642         raises (Reflective::StructuralError,
1643               Reflective::SemanticError,
1644               Reflective::NotFound);
1645     void modify_provision (in ModelElement supplier,
1646                           in Dependency provision,
1647                           in Dependency new_provision)
1648         raises (Reflective::StructuralError,
1649               Reflective::SemanticError,
1650               Reflective::NotFound);
1651     void remove (in ModelElement supplier, in Dependency provision)
1652         raises (Reflective::StructuralError,
1653               Reflective::SemanticError,
1654               Reflective::NotFound);
1655 };
1656
1657 struct ModelElementOwnsTaggedValueLink {
1658     UmlCore::ModelElement model_element;
1659     UmlCore::TaggedValue tagged_value;
1660 };
1661 typedef sequence <ModelElementOwnsTaggedValueLink>
1662     ModelElementOwnsTaggedValueLinkSet;
1663
1664 interface ModelElementOwnsTaggedValue : Reflective::RefAssociation {
1665     readonly attribute UmlCorePackage enclosing_package_ref;
1666     ModelElementOwnsTaggedValueLinkSet
1667     all_model_element_owns_tagged_value_links();
1668     boolean exists (in UmlCore::ModelElement model_element,
1669                   in UmlCore::TaggedValue tagged_value);
1670     UmlCore::ModelElement with_tagged_value (
1671         in UmlCore::TaggedValue tagged_value);
```

```

1672 UmlCore::TaggedValueSet with_model_element (
1673         in UmlCore::ModelElement model_element);
1674 void add (in UmlCore::ModelElement model_element,
1675         in UmlCore::TaggedValue tagged_value)
1676     raises (Reflective::StructuralError, Reflective::SemanticError);
1677 void modify_model_element (in UmlCore::ModelElement model_element,
1678         in UmlCore::TaggedValue tagged_value,
1679         in UmlCore::ModelElement new_model_element)
1680     raises (Reflective::StructuralError,
1681         Reflective::SemanticError,
1682         Reflective::NotFound);
1683 void modify_tagged_value (in UmlCore::ModelElement model_element,
1684         in UmlCore::TaggedValue tagged_value,
1685         in UmlCore::TaggedValue new_tagged_value)
1686     raises (Reflective::StructuralError,
1687         Reflective::SemanticError,
1688         Reflective::NotFound);
1689 void remove (in UmlCore::ModelElement model_element,
1690         in UmlCore::TaggedValue tagged_value)
1691     raises (Reflective::StructuralError,
1692         Reflective::SemanticError,
1693         Reflective::NotFound);
1694 };
1695
1696 struct ConstraintConstrainsModelElementLink {
1697     ModelElement constrained_element;
1698     UmlCore::Constraint constraint;
1699 };
1700 typedef sequence <ConstraintConstrainsModelElementLink>
1701     ConstraintConstrainsModelElementLinkSet;
1702
1703 interface ConstraintConstrainsModelElement : Reflective::RefAssociation {
1704     readonly attribute UmlCorePackage enclosing_package_ref;
1705     ConstraintConstrainsModelElementLinkSet
1706         all_constraint_constrains_model_element_links();
1707     boolean exists (in ModelElement constrained_element,
1708         in UmlCore::Constraint constraint);
1709     ModelElementUList with_constraint (in UmlCore::Constraint constraint);
1710     UmlCore::ConstraintSet with_constrained_element (
1711         in ModelElement constrained_element);
1712     void add (in ModelElement constrained_element,

```

```

1713         in UmlCore::Constraint constraint)
1714     raises (Reflective::StructuralError, Reflective::SemanticError);
1715 void add_before_constrained_element (in ModelElement constrained_element,
1716                                     in UmlCore::Constraint constraint,
1717                                     in ModelElement before)
1718     raises (Reflective::StructuralError,
1719            Reflective::SemanticError,
1720            Reflective::NotFound);
1721 void modify_constrained_element (in ModelElement constrained_element,
1722                                 in UmlCore::Constraint constraint,
1723                                 in ModelElement new_constrained_element)
1724     raises (Reflective::StructuralError,
1725            Reflective::SemanticError,
1726            Reflective::NotFound);
1727 void modify_constraint (in ModelElement constrained_element,
1728                        in UmlCore::Constraint constraint,
1729                        in UmlCore::Constraint new_constraint)
1730     raises (Reflective::StructuralError,
1731            Reflective::SemanticError,
1732            Reflective::NotFound);
1733 void remove (in ModelElement constrained_element,
1734             in UmlCore::Constraint constraint)
1735     raises (Reflective::StructuralError,
1736            Reflective::SemanticError,
1737            Reflective::NotFound);
1738 };
1739
1740 struct ModelElementIsClientInDependencyLink {
1741     ModelElement client;
1742     Dependency requirement;
1743 };
1744 typedef sequence <ModelElementIsClientInDependencyLink>
1745     ModelElementIsClientInDependencyLinkSet;
1746
1747 interface ModelElementIsClientInDependency : Reflective::RefAssociation {
1748     readonly attribute UmlCorePackage enclosing_package_ref;
1749     ModelElementIsClientInDependencyLinkSet
1750     all_model_element_is_client_in_dependency_links();
1751     boolean exists (in ModelElement client, in Dependency requirement);
1752     ModelElementSet with_requirement (in Dependency requirement);
1753     DependencySet with_client (in ModelElement client);

```

```

1754 void add (in ModelElement client, in Dependency requirement)
1755     raises (Reflective::StructuralError, Reflective::SemanticError);
1756 void modify_client (in ModelElement client,
1757     in Dependency requirement,
1758     in ModelElement new_client)
1759     raises (Reflective::StructuralError,
1760     Reflective::SemanticError,
1761     Reflective::NotFound);
1762 void modify_requirement (in ModelElement client,
1763     in Dependency requirement,
1764     in Dependency new_requirement)
1765     raises (Reflective::StructuralError,
1766     Reflective::SemanticError,
1767     Reflective::NotFound);
1768 void remove (in ModelElement client, in Dependency requirement)
1769     raises (Reflective::StructuralError,
1770     Reflective::SemanticError,
1771     Reflective::NotFound);
1772 };
1773
1774 struct EnumerationOwnsEnumerationLiteralLink {
1775     UmlCore::Enumeration enumeration;
1776     EnumerationLiteral literal;
1777 };
1778 typedef sequence <EnumerationOwnsEnumerationLiteralLink>
1779     EnumerationOwnsEnumerationLiteralLinkSet;
1780
1781 interface EnumerationOwnsEnumerationLiteral : Reflective::RefAssociation {
1782     readonly attribute UmlCorePackage enclosing_package_ref;
1783     EnumerationOwnsEnumerationLiteralLinkSet
1784     all_enumeration_owns_enumeration_literal_links();
1785     boolean exists (in UmlCore::Enumeration enumeration,
1786         in EnumerationLiteral literal);
1787     UmlCore::Enumeration with_literal (in EnumerationLiteral literal);
1788     EnumerationLiteralUList with_enumeration (
1789         in UmlCore::Enumeration enumeration);
1790     void add (in UmlCore::Enumeration enumeration,
1791         in EnumerationLiteral literal)
1792         raises (Reflective::StructuralError, Reflective::SemanticError);
1793     void add_before_literal (in UmlCore::Enumeration enumeration,
1794         in EnumerationLiteral literal,

```

```
1795         in EnumerationLiteral before)
1796     raises (Reflective::StructuralError,
1797           Reflective::SemanticError,
1798           Reflective::NotFound);
1799 void modify_enumeration (in UmlCore::Enumeration enumeration,
1800                          in EnumerationLiteral literal,
1801                          in UmlCore::Enumeration new_enumeration)
1802     raises (Reflective::StructuralError,
1803           Reflective::SemanticError,
1804           Reflective::NotFound);
1805 void modify_literal (in UmlCore::Enumeration enumeration,
1806                    in EnumerationLiteral literal,
1807                    in EnumerationLiteral new_literal)
1808     raises (Reflective::StructuralError,
1809           Reflective::SemanticError,
1810           Reflective::NotFound);
1811 void remove (in UmlCore::Enumeration enumeration,
1812            in EnumerationLiteral literal)
1813     raises (Reflective::StructuralError,
1814           Reflective::SemanticError,
1815           Reflective::NotFound);
1816 };
1817
1818 struct StereotypeOwnsRequiredTaggedValueLink {
1819     TaggedValue required_tag;
1820     UmlCore::Stereotype stereotype;
1821 };
1822 typedef sequence <StereotypeOwnsRequiredTaggedValueLink>
1823     StereotypeOwnsRequiredTaggedValueLinkSet;
1824
1825 interface StereotypeOwnsRequiredTaggedValue : Reflective::RefAssociation {
1826     readonly attribute UmlCorePackage enclosing_package_ref;
1827     StereotypeOwnsRequiredTaggedValueLinkSet
1828     all_stereotype_owns_required_tagged_value_links();
1829     boolean exists (in TaggedValue required_tag,
1830                  in UmlCore::Stereotype stereotype);
1831     TaggedValueSet with_stereotype (in UmlCore::Stereotype stereotype);
1832     UmlCore::Stereotype with_required_tag (in TaggedValue required_tag);
1833     void add (in TaggedValue required_tag, in UmlCore::Stereotype stereotype)
1834         raises (Reflective::StructuralError, Reflective::SemanticError);
1835     void modify_required_tag (in TaggedValue required_tag,
```

```

1836             in UmlCore::Stereotype stereotype,
1837             in TaggedValue new_required_tag)
1838     raises (Reflective::StructuralError,
1839           Reflective::SemanticError,
1840           Reflective::NotFound);
1841 void modify_stereotype (in TaggedValue required_tag,
1842                       in UmlCore::Stereotype stereotype,
1843                       in UmlCore::Stereotype new_stereotype)
1844     raises (Reflective::StructuralError,
1845           Reflective::SemanticError,
1846           Reflective::NotFound);
1847 void remove (in TaggedValue required_tag,
1848            in UmlCore::Stereotype stereotype)
1849     raises (Reflective::StructuralError,
1850           Reflective::SemanticError,
1851           Reflective::NotFound);
1852 };
1853
1854 struct StereotypeExtendsModelElementLink {
1855     UmlCore::Stereotype stereotype;
1856     ModelElement extended_element;
1857 };
1858 typedef sequence <StereotypeExtendsModelElementLink>
1859     StereotypeExtendsModelElementLinkSet;
1860
1861 interface StereotypeExtendsModelElement : Reflective::RefAssociation {
1862     readonly attribute UmlCorePackage enclosing_package_ref;
1863     StereotypeExtendsModelElementLinkSet
1864     all_stereotype_extends_model_element_links();
1865     boolean exists (in UmlCore::Stereotype stereotype,
1866                  in ModelElement extended_element);
1867     UmlCore::StereotypeSet with_extended_element (
1868         in ModelElement extended_element);
1869     ModelElementSet with_stereotype (in UmlCore::Stereotype stereotype);
1870 void add (in UmlCore::Stereotype stereotype,
1871         in ModelElement extended_element)
1872     raises (Reflective::StructuralError, Reflective::SemanticError);
1873 void modify_stereotype (in UmlCore::Stereotype stereotype,
1874                       in ModelElement extended_element,
1875                       in UmlCore::Stereotype new_stereotype)
1876     raises (Reflective::StructuralError,

```

```

1877         Reflective::SemanticError,
1878         Reflective::NotFound);
1879 void modify_extended_element (in UmlCore::Stereotype stereotype,
1880                             in ModelElement extended_element,
1881                             in ModelElement new_extended_element)
1882     raises (Reflective::StructuralError,
1883           Reflective::SemanticError,
1884           Reflective::NotFound);
1885 void remove (in UmlCore::Stereotype stereotype,
1886            in ModelElement extended_element)
1887     raises (Reflective::StructuralError,
1888           Reflective::SemanticError,
1889           Reflective::NotFound);
1890 };
1891
1892 struct ModelElementParameterizedByModelElementLink {
1893     ModelElement template;
1894     ModelElement template_parameter;
1895 };
1896 typedef sequence <ModelElementParameterizedByModelElementLink>
1897     ModelElementParameterizedByModelElementLinkSet;
1898
1899 interface ModelElementParameterizedByModelElement :
1900     Reflective::RefAssociation {
1901     readonly attribute UmlCorePackage enclosing_package_ref;
1902     ModelElementParameterizedByModelElementLinkSet
1903     all_model_element_parameterized_by_model_element_links();
1904     boolean exists (in ModelElement template,
1905                  in ModelElement template_parameter);
1906     ModelElement with_template_parameter (in ModelElement template_parameter);
1907     ModelElementUList with_template (in ModelElement template);
1908     void add (in ModelElement template, in ModelElement template_parameter)
1909         raises (Reflective::StructuralError, Reflective::SemanticError);
1910     void add_before_template_parameter (in ModelElement template,
1911                                       in ModelElement template_parameter,
1912                                       in ModelElement before)
1913         raises (Reflective::StructuralError,
1914               Reflective::SemanticError,
1915               Reflective::NotFound);
1916     void modify_template (in ModelElement template,
1917                          in ModelElement template_parameter,

```



```

1918         in ModelElement new_template)
1919     raises (Reflective::StructuralError,
1920           Reflective::SemanticError,
1921           Reflective::NotFound);
1922 void modify_template_parameter (in ModelElement template,
1923                               in ModelElement template_parameter,
1924                               in ModelElement new_template_parameter)
1925     raises (Reflective::StructuralError,
1926           Reflective::SemanticError,
1927           Reflective::NotFound);
1928 void remove (in ModelElement template, in ModelElement template_parameter)
1929     raises (Reflective::StructuralError,
1930           Reflective::SemanticError,
1931           Reflective::NotFound);
1932 };
1933
1934 struct NamespaceOwnsElementOwnershipLink {
1935     UmlCore::Namespace namespace;
1936     UmlCore::ElementOwnership element_ownership;
1937 };
1938 typedef sequence <NamespaceOwnsElementOwnershipLink>
1939     NamespaceOwnsElementOwnershipLinkSet;
1940
1941 interface NamespaceOwnsElementOwnership : Reflective::RefAssociation {
1942     readonly attribute UmlCorePackage enclosing_package_ref;
1943     NamespaceOwnsElementOwnershipLinkSet
1944     all_namespace_owns_element_ownership_links();
1945     boolean exists (in UmlCore::Namespace namespace,
1946                  in UmlCore::ElementOwnership element_ownership);
1947     UmlCore::Namespace with_element_ownership (
1948         in UmlCore::ElementOwnership element_ownership);
1949     UmlCore::ElementOwnershipSet with_namespace (
1950         in UmlCore::Namespace namespace);
1951     void add (in UmlCore::Namespace namespace,
1952             in UmlCore::ElementOwnership element_ownership)
1953         raises (Reflective::StructuralError, Reflective::SemanticError);
1954     void modify_namespace (in UmlCore::Namespace namespace,
1955                          in UmlCore::ElementOwnership element_ownership,
1956                          in UmlCore::Namespace new_namespace)
1957         raises (Reflective::StructuralError,
1958               Reflective::SemanticError,

```

```
1959         Reflective::NotFound);
1960 void modify_element_ownership (
1961         in UmlCore::Namespace namespace,
1962         in UmlCore::ElementOwnership element_ownership,
1963         in UmlCore::ElementOwnership new_element_ownership)
1964     raises (Reflective::StructuralError,
1965           Reflective::SemanticError,
1966           Reflective::NotFound);
1967 void remove (in UmlCore::Namespace namespace,
1968             in UmlCore::ElementOwnership element_ownership)
1969     raises (Reflective::StructuralError,
1970           Reflective::SemanticError,
1971           Reflective::NotFound);
1972 };
1973
1974 struct ModelElementIsOwnedViaElementOwnershipLink {
1975     ModelElement owned_element;
1976     ElementOwnership namespace1;
1977 };
1978 typedef sequence <ModelElementIsOwnedViaElementOwnershipLink>
1979     ModelElementIsOwnedViaElementOwnershipLinkSet;
1980
1981 interface ModelElementIsOwnedViaElementOwnership :
1982     Reflective::RefAssociation {
1983     readonly attribute UmlCorePackage enclosing_package_ref;
1984     ModelElementIsOwnedViaElementOwnershipLinkSet
1985     all_model_element_is_owned_via_element_ownership_links();
1986     boolean exists (in ModelElement owned_element,
1987                   in ElementOwnership namespace1);
1988     ModelElement with_namespace1 (in ElementOwnership namespace1);
1989     ElementOwnership with_owned_element (in ModelElement owned_element);
1990     void add (in ModelElement owned_element, in ElementOwnership namespace1)
1991         raises (Reflective::StructuralError, Reflective::SemanticError);
1992     void modify_owned_element (in ModelElement owned_element,
1993                               in ElementOwnership namespace1,
1994                               in ModelElement new_owned_element)
1995         raises (Reflective::StructuralError,
1996               Reflective::SemanticError,
1997               Reflective::NotFound);
1998     void modify_namespace1 (in ModelElement owned_element,
1999                             in ElementOwnership namespace1,
```

```

2000         in ElementOwnership new_namespace1)
2001     raises (Reflective::StructuralError,
2002             Reflective::SemanticError,
2003             Reflective::NotFound);
2004 void remove (in ModelElement owned_element, in ElementOwnership namespace1)
2005     raises (Reflective::StructuralError,
2006             Reflective::SemanticError,
2007             Reflective::NotFound);
2008 };
2009
2010 struct StereotypeIsConstrainedByConstraintLink {
2011     Constraint stereotype_constraint;
2012     Stereotype constrained_stereotype;
2013 };
2014 typedef sequence <StereotypeIsConstrainedByConstraintLink>
2015     StereotypeIsConstrainedByConstraintLinkSet;
2016
2017 interface StereotypeIsConstrainedByConstraint : Reflective::RefAssociation {
2018     readonly attribute UmlCorePackage enclosing_package_ref;
2019     StereotypeIsConstrainedByConstraintLinkSet
2020     all_stereotype_is_constrained_by_constraint_links();
2021     boolean exists (in Constraint stereotype_constraint,
2022                   in Stereotype constrained_stereotype);
2023     ConstraintSet with_constrained_stereotype (
2024         in Stereotype constrained_stereotype);
2025     Stereotype with_stereotype_constraint (
2026         in Constraint stereotype_constraint);
2027     void add (in Constraint stereotype_constraint,
2028              in Stereotype constrained_stereotype)
2029         raises (Reflective::StructuralError, Reflective::SemanticError);
2030     void modify_stereotype_constraint (in Constraint stereotype_constraint,
2031                                       in Stereotype constrained_stereotype,
2032                                       in Constraint new_stereotype_constraint)
2033         raises (Reflective::StructuralError,
2034               Reflective::SemanticError,
2035               Reflective::NotFound);
2036     void modify_constrained_stereotype (
2037         in Constraint stereotype_constraint,
2038         in Stereotype constrained_stereotype,
2039         in Stereotype new_constrained_stereotype)
2040         raises (Reflective::StructuralError,

```

```
2041         Reflective::SemanticError,
2042         Reflective::NotFound);
2043     void remove (in Constraint stereotype_constraint,
2044                 in Stereotype constrained_stereotype)
2045         raises (Reflective::StructuralError,
2046               Reflective::SemanticError,
2047               Reflective::NotFound);
2048 };
2049
2050 interface UmlCorePackageFactory {
2051     UmlCorePackage create_uml_core_package ()
2052         raises (Reflective::SemanticError);
2053 };
2054
2055 interface UmlCorePackage : Reflective::RefPackage {
2056     readonly attribute ElementClass element_class_ref;
2057     readonly attribute TaggedValueClass tagged_value_class_ref;
2058     readonly attribute EnumerationLiteralClass enumeration_literal_class_ref;
2059     readonly attribute ModelElementClass model_element_class_ref;
2060     readonly attribute FeatureClass feature_class_ref;
2061     readonly attribute GeneralizationClass generalization_class_ref;
2062     readonly attribute NamespaceClass namespace_class_ref;
2063     readonly attribute ParameterClass parameter_class_ref;
2064     readonly attribute ConstraintClass constraint_class_ref;
2065     readonly attribute DependencyClass dependency_class_ref;
2066     readonly attribute RequestClass request_class_ref;
2067     readonly attribute GeneralizableElementClass
2068         generalizable_element_class_ref;
2069     readonly attribute BehavioralFeatureClass behavioral_feature_class_ref;
2070     readonly attribute ClassifierClass classifier_class_ref;
2071     readonly attribute OperationClass operation_class_ref;
2072     readonly attribute StereotypeClass stereotype_class_ref;
2073     readonly attribute StructuralFeatureClass structural_feature_class_ref;
2074     readonly attribute DataTypeClass data_type_class_ref;
2075     readonly attribute UmlInterfaceClass uml_interface_class_ref;
2076     readonly attribute UmlAttributeClass uml_attribute_class_ref;
2077     readonly attribute AssociationEndClass association_end_class_ref;
2078     readonly attribute AssociationClass association_class_ref;
2079     readonly attribute MethodClass method_class_ref;
2080     readonly attribute EnumerationClass enumeration_class_ref;
2081     readonly attribute ClassClass class_class_ref;
```

2082    readonly attribute PrimitiveClass primitive\_class\_ref;  
2083    readonly attribute StructureClass structure\_class\_ref;  
2084    readonly attribute UmlAssociationClassClass  
2085       uml\_association\_class\_class\_ref;  
2086    readonly attribute ElementOwnershipClass element\_ownership\_class\_ref;  
2087  
2088    readonly attribute AssociationOwnsAssociationEnd  
2089       association\_owns\_association\_end\_ref;  
2090    readonly attribute ClassifierOwnsFeature classifier\_owns\_feature\_ref;  
2091    readonly attribute MethodIsSpecifiedByOperation  
2092       method\_is\_specified\_by\_operation\_ref;  
2093    readonly attribute StructuralFeatureIsOfTypeClassifier  
2094       structural\_feature\_is\_of\_type\_classifier\_ref;  
2095    readonly attribute NamespaceOwnsModelElement  
2096       namespace\_owns\_model\_element\_ref;  
2097    readonly attribute BehavioralFeatureOwnsParameter  
2098       behavioral\_feature\_owns\_parameter\_ref;  
2099    readonly attribute ParameterIsOfTypeClassifier  
2100       parameter\_is\_of\_type\_classifier\_ref;  
2101    readonly attribute GeneralizableElementIsSubtypeInGeneralization  
2102       generalizable\_element\_is\_subtype\_in\_generalization\_ref;  
2103    readonly attribute GeneralizableElementIsSupertypeInGeneralization  
2104       generalizable\_element\_is\_supertype\_in\_generalization\_ref;  
2105    readonly attribute AssociationEndOwnsQualifierAttribute  
2106       association\_end\_owns\_qualifier\_attribute\_ref;  
2107    readonly attribute AssociationEndIsOfTypeClassifier  
2108       association\_end\_is\_of\_type\_classifier\_ref;  
2109    readonly attribute ClassifierIsRealizedByClassifier  
2110       classifier\_is\_realized\_by\_classifier\_ref;  
2111    readonly attribute AssociationEndIsSpecifiedByClassifier  
2112       association\_end\_is\_specified\_by\_classifier\_ref;  
2113    readonly attribute ModelElementIsSupplierInDependency  
2114       model\_element\_is\_supplier\_in\_dependency\_ref;  
2115    readonly attribute ModelElementOwnsTaggedValue  
2116       model\_element\_owns\_tagged\_value\_ref;  
2117    readonly attribute ConstraintConstrainsModelElement  
2118       constraint\_constrains\_model\_element\_ref;  
2119    readonly attribute ModelElementIsClientInDependency  
2120       model\_element\_is\_client\_in\_dependency\_ref;  
2121    readonly attribute EnumerationOwnsEnumerationLiteral  
2122       enumeration\_owns\_enumeration\_literal\_ref;

```

2123  readonly attribute StereotypeOwnsRequiredTaggedValue
2124      stereotype_owns_required_tagged_value_ref;
2125  readonly attribute StereotypeExtendsModelElement
2126      stereotype_extends_model_element_ref;
2127  readonly attribute ModelElementParameterizedByModelElement
2128      model_element_parameterized_by_model_element_ref;
2129  readonly attribute NamespaceOwnsElementOwnership
2130      namespace_owns_element_ownership_ref;
2131  readonly attribute ModelElementIsOwnedViaElementOwnership
2132      model_element_is_owned_via_element_ownership_ref;
2133  readonly attribute StereotypeIsConstrainedByConstraint
2134      stereotype_is_constrained_by_constraint_ref;
2135  };
2136  };

```

### 5.4.2 UMLModelManagement

```

1 #include "UmlCore.idl"
2
3 module UmlModelManagement {
4  interface UmlModelManagementPackage;
5  interface ElementReference;
6  interface ElementReferenceClass;
7  typedef sequence<ElementReference> ElementReferenceUList;
8  typedef sequence<ElementReference> ElementReferenceSet;
9  interface Model;
10 interface ModelClass;
11 typedef sequence<Model> ModelUList;
12 interface Package;
13 interface PackageClass;
14 typedef sequence<Package> PackageUList;
15 typedef sequence<Package> PackageSet;
16 interface Subsystem;
17 interface SubsystemClass;
18 typedef sequence<Subsystem> SubsystemUList;
19
20 interface PackageClass : ::UmlCore::GeneralizableElementClass {

```

```

21  readonly attribute PackageUList all_of_kind_package;
22  readonly attribute PackageUList all_of_type_package;
23  Package create_package (in ::UmlCore::Name name,
24                          in boolean is_root,
25                          in boolean is_leaf,
26                          in boolean is_abstract)
27    raises (Reflective::SemanticError);
28 };
29
30 interface Package : PackageClass, ::UmlCore::GeneralizableElement {
31   ElementReferenceSet supplier_element_reference ()
32     raises (Reflective::NotSet, Reflective::SemanticError);
33   void add_supplier_element_reference (in ElementReferenceSet new_value)
34     raises (Reflective::StructuralError, Reflective::SemanticError);
35   void remove_supplier_element_reference ()
36     raises (Reflective::SemanticError);
37   ::UmlCore::ModelElementSet referenced_element ()
38     raises (Reflective::SemanticError);
39   void add_referenced_element (in ::UmlCore::ModelElementSet new_value)
40     raises (Reflective::StructuralError, Reflective::SemanticError);
41   void remove_referenced_element ()
42     raises (Reflective::SemanticError);
43 };
44
45 interface ModelClass : PackageClass {
46   readonly attribute ModelUList all_of_kind_model;
47   readonly attribute ModelUList all_of_type_model;
48   Model create_model (in ::UmlCore::Name name,
49                      in boolean is_root,
50                      in boolean is_leaf,
51                      in boolean is_abstract)
52     raises (Reflective::SemanticError);
53 };
54
55 interface Model : ModelClass, Package { };
56
57 interface SubsystemClass : PackageClass, ::UmlCore::ClassifierClass {
58   readonly attribute SubsystemUList all_of_kind_subsystem;
59   readonly attribute SubsystemUList all_of_type_subsystem;
60   Subsystem create_subsystem (in ::UmlCore::Name name,
61                               in boolean is_root,

```

```
62             in boolean is_leaf,
63             in boolean is_abstract,
64             in boolean is_instantiable)
65     raises (Reflective::SemanticError);
66 };
67
68 interface Subsystem : SubsystemClass, Package, ::UmlCore::Classifier {
69     boolean is_instantiable ()
70     raises (Reflective::SemanticError);
71     void set_is_instantiable (in boolean new_value)
72     raises (Reflective::SemanticError);
73 };
74
75 interface ElementReferenceClass : ::UmlCore::ElementClass {
76     readonly attribute ElementReferenceUList all_of_kind_element_reference;
77     readonly attribute ElementReferenceUList all_of_type_element_reference;
78     ElementReference create_element_reference (
79         in ::UmlCore::VisibilityKind visibility,
80         in ::UmlCore::Name alias)
81     raises (Reflective::SemanticError);
82 };
83
84 interface ElementReference : ElementReferenceClass, ::UmlCore::Element {
85     ::UmlCore::VisibilityKind visibility ()
86     raises (Reflective::SemanticError);
87     void set_visibility (in ::UmlCore::VisibilityKind new_value)
88     raises (Reflective::SemanticError);
89     ::UmlCore::Name alias ()
90     raises (Reflective::SemanticError);
91     void set_alias (in ::UmlCore::Name new_value)
92     raises (Reflective::SemanticError);
93     Package referencing_package ()
94     raises (Reflective::SemanticError);
95     void set_referencing_package (in Package new_value)
96     raises (Reflective::SemanticError);
97     ::UmlCore::ModelElementSet referenced_element ()
98     raises (Reflective::SemanticError);
99     void add_referenced_element (in ::UmlCore::ModelElementSet new_value)
100     raises (Reflective::StructuralError, Reflective::SemanticError);
101     void remove_referenced_element ()
102     raises (Reflective::SemanticError);
```



```
103 };
104
105 struct PackageElementReferenceLink {
106     Package referencing_package;
107     ElementReference supplier_element_reference;
108 };
109 typedef sequence <PackageElementReferenceLink> PackageElementReferenceLinkSet;
110
111 interface PackageElementReference : Reflective::RefAssociation {
112     readonly attribute UmlModelManagementPackage enclosing_package_ref;
113     PackageElementReferenceLinkSet all_package_element_reference_links();
114     boolean exists (in Package referencing_package,
115                   in ElementReference supplier_element_reference);
116     Package with_supplier_element_reference (
117         in ElementReference supplier_element_reference);
118     ElementReferenceSet with_referencing_package (
119         in Package referencing_package);
120     void add (in Package referencing_package,
121             in ElementReference supplier_element_reference)
122         raises (Reflective::StructuralError, Reflective::SemanticError);
123     void modify_referencing_package (
124         in Package referencing_package,
125         in ElementReference supplier_element_reference,
126         in Package new_referencing_package)
127         raises (Reflective::StructuralError,
128             Reflective::SemanticError,
129             Reflective::NotFound);
130     void modify_supplier_element_reference (
131         in Package referencing_package,
132         in ElementReference supplier_element_reference,
133         in ElementReference new_supplier_element_reference)
134         raises (Reflective::StructuralError,
135             Reflective::SemanticError,
136             Reflective::NotFound);
137     void remove (in Package referencing_package,
138                in ElementReference supplier_element_reference)
139         raises (Reflective::StructuralError,
140             Reflective::SemanticError,
141             Reflective::NotFound);
142 };
143
```

```
144 struct ElementReferenceReferencesModelElementLink {
145     ElementReference client_element_reference;
146     ::UmlCore::ModelElement referenced_element;
147 };
148 typedef sequence <ElementReferenceReferencesModelElementLink>
149     ElementReferenceReferencesModelElementLinkSet;
150
151 interface ElementReferenceReferencesModelElement :
152     Reflective::RefAssociation {
153     readonly attribute UmlModelManagementPackage enclosing_package_ref;
154     ElementReferenceReferencesModelElementLinkSet
155     all_element_reference_references_model_element_links();
156     boolean exists (in ElementReference client_element_reference,
157         in ::UmlCore::ModelElement referenced_element);
158     ElementReferenceSet with_referenced_element (
159         in ::UmlCore::ModelElement referenced_element);
160     ::UmlCore::ModelElementSet with_client_element_reference (
161         in ElementReference client_element_reference);
162     void add (in ElementReference client_element_reference,
163         in ::UmlCore::ModelElement referenced_element)
164         raises (Reflective::StructuralError, Reflective::SemanticError);
165     void modify_client_element_reference (
166         in ElementReference client_element_reference,
167         in ::UmlCore::ModelElement referenced_element,
168         in ElementReference new_client_element_reference)
169         raises (Reflective::StructuralError,
170             Reflective::SemanticError,
171             Reflective::NotFound);
172     void modify_referenced_element (
173         in ElementReference client_element_reference,
174         in ::UmlCore::ModelElement referenced_element,
175         in ::UmlCore::ModelElement new_referenced_element)
176         raises (Reflective::StructuralError,
177             Reflective::SemanticError,
178             Reflective::NotFound);
179     void remove (in ElementReference client_element_reference,
180         in ::UmlCore::ModelElement referenced_element)
181         raises (Reflective::StructuralError,
182             Reflective::SemanticError,
183             Reflective::NotFound);
184 };
```

```

185
186 struct PackageReferencesModelElementLink {
187     ::UmlCore::ModelElement referenced_element;
188     Package referencing_package;
189 };
190 typedef sequence <PackageReferencesModelElementLink>
191     PackageReferencesModelElementLinkSet;
192
193 interface PackageReferencesModelElement : Reflective::RefAssociation {
194     readonly attribute UmlModelManagementPackage enclosing_package_ref;
195     PackageReferencesModelElementLinkSet
196     all_package_references_model_element_links();
197     boolean exists (in ::UmlCore::ModelElement referenced_element,
198                   in Package referencing_package);
199     ::UmlCore::ModelElementSet with_referencing_package (
200         in Package referencing_package);
201     PackageSet with_referenced_element (
202         in ::UmlCore::ModelElement referenced_element);
203     void add (in ::UmlCore::ModelElement referenced_element,
204             in Package referencing_package)
205         raises (Reflective::StructuralError, Reflective::SemanticError);
206     void modify_referenced_element (
207         in ::UmlCore::ModelElement referenced_element,
208         in Package referencing_package,
209         in ::UmlCore::ModelElement new_referenced_element)
210         raises (Reflective::StructuralError,
211               Reflective::SemanticError,
212               Reflective::NotFound);
213     void modify_referencing_package (
214         in ::UmlCore::ModelElement referenced_element,
215         in Package referencing_package,
216         in Package new_referencing_package)
217         raises (Reflective::StructuralError,
218               Reflective::SemanticError,
219               Reflective::NotFound);
220     void remove (in ::UmlCore::ModelElement referenced_element,
221                in Package referencing_package)
222         raises (Reflective::StructuralError,
223               Reflective::SemanticError,
224               Reflective::NotFound);
225 };

```

```

226
227 interface UmlModelManagementPackageFactory {
228     UmlModelManagementPackage create_uml_model_management_package ()
229     raises (Reflective::SemanticError);
230 };
231
232 interface UmlModelManagementPackage : Reflective::RefPackage {
233     readonly attribute PackageClass package_class_ref;
234     readonly attribute ModelClass model_class_ref;
235     readonly attribute SubsystemClass subsystem_class_ref;
236     readonly attribute ElementReferenceClass element_reference_class_ref;
237
238     readonly attribute PackageElementReference package_element_reference_ref;
239     readonly attribute ElementReferenceReferencesModelElement
240     element_reference_references_model_element_ref;
241     readonly attribute PackageReferencesModelElement
242     package_references_model_element_ref;
243 };
244 };

```

### 5.4.3 UML Auxiliary Elements

```

1 #include "UmlCore.idl"
2
3 module UmlAuxiliaryElements {
4     interface UmlAuxiliaryElementsPackage;
5     interface Usage;
6     interface UsageClass;
7     typedef sequence<Usage> UsageUList;
8     interface Component;
9     interface ComponentClass;
10    typedef sequence<Component> ComponentUList;
11    typedef sequence<Component> ComponentSet;
12    interface Presentation;
13    interface PresentationClass;
14    typedef sequence<Presentation> PresentationUList;
15    typedef sequence<Presentation> PresentationSet;
16    interface ViewElement;
17    interface ViewElementClass;
18    typedef sequence<ViewElement> ViewElementUList;
19    typedef sequence<ViewElement> ViewElementSet;

```

```

20 interface Binding;
21 interface BindingClass;
22 typedef sequence<Binding> BindingUList;
23 interface Comment;
24 interface CommentClass;
25 typedef sequence<Comment> CommentUList;
26 interface Trace;
27 interface TraceClass;
28 typedef sequence<Trace> TraceUList;
29 interface Node;
30 interface NodeClass;
31 typedef sequence<Node> NodeUList;
32 typedef sequence<Node> NodeSet;
33 interface Refinement;
34 interface RefinementClass;
35 typedef sequence<Refinement> RefinementUList;
36
37 interface ComponentClass : ::UmlCore::ClassifierClass {
38     readonly attribute ComponentUList all_of_kind_component;
39     readonly attribute ComponentUList all_of_type_component;
40     Component create_component (in ::UmlCore::Name name,
41                                 in boolean is_root,
42                                 in boolean is_leaf,
43                                 in boolean is_abstract)
44     raises (Reflective::SemanticError);
45 };
46
47 interface Component : ComponentClass, ::UmlCore::Classifier {
48     ::UmlCore::ModelElementSet model_element ()
49     raises (Reflective::NotSet, Reflective::SemanticError);
50     void add_model_element (in ::UmlCore::ModelElementSet new_value)
51     raises (Reflective::StructuralError, Reflective::SemanticError);
52     void remove_model_element ()
53     raises (Reflective::SemanticError);
54     NodeSet deployment ()
55     raises (Reflective::NotSet, Reflective::SemanticError);
56     void add_deployment (in NodeSet new_value)
57     raises (Reflective::StructuralError, Reflective::SemanticError);
58     void remove_deployment ()
59     raises (Reflective::SemanticError);
60 };

```

```
61
62 interface NodeClass : ::UmlCore::ClassifierClass {
63     readonly attribute NodeUList all_of_kind_node;
64     readonly attribute NodeUList all_of_type_node;
65     Node create_node (in ::UmlCore::Name name,
66                     in boolean is_root,
67                     in boolean is_leaf,
68                     in boolean is_abstract)
69     raises (Reflective::SemanticError);
70 };
71
72 interface Node : NodeClass, ::UmlCore::Classifier {
73     UmlAuxiliaryElements::ComponentSet component ()
74     raises (Reflective::NotSet, Reflective::SemanticError);
75     void add_component (in UmlAuxiliaryElements::ComponentSet new_value)
76     raises (Reflective::StructuralError, Reflective::SemanticError);
77     void remove_component ()
78     raises (Reflective::SemanticError);
79 };
80
81 interface PresentationClass : ::UmlCore::ElementClass {
82     readonly attribute PresentationUList all_of_kind_presentation;
83     readonly attribute PresentationUList all_of_type_presentation;
84     Presentation create_presentation (in ::UmlCore::Geometry geometry,
85                                     in ::UmlCore::GraphicMarker style)
86     raises (Reflective::SemanticError);
87 };
88
89 interface Presentation : PresentationClass, ::UmlCore::Element {
90     ::UmlCore::Geometry geometry ()
91     raises (Reflective::SemanticError);
92     void set_geometry (in ::UmlCore::Geometry new_value)
93     raises (Reflective::SemanticError);
94     ::UmlCore::GraphicMarker style ()
95     raises (Reflective::SemanticError);
96     void set_style (in ::UmlCore::GraphicMarker new_value)
97     raises (Reflective::SemanticError);
98     ::UmlCore::ModelElement model ()
99     raises (Reflective::SemanticError);
100    void set_model (in ::UmlCore::ModelElement new_value)
101    raises (Reflective::SemanticError);
```

```

102  ViewElement view ()
103      raises (Reflective::SemanticError);
104  void set_view (in ViewElement new_value)
105      raises (Reflective::SemanticError);
106  };
107
108  interface ViewElementClass : ::UmlCore::ElementClass {
109      readonly attribute ViewElementUList all_of_kind_view_element;
110  };
111
112  interface ViewElement : ViewElementClass, ::UmlCore::Element {
113      UmlAuxiliaryElements::PresentationSet presentation ()
114          raises (Reflective::NotSet, Reflective::SemanticError);
115      void add_presentation (in UmlAuxiliaryElements::PresentationSet new_value)
116          raises (Reflective::StructuralError, Reflective::SemanticError);
117      void remove_presentation ()
118          raises (Reflective::SemanticError);
119      ::UmlCore::ModelElementSet model ()
120          raises (Reflective::NotSet, Reflective::SemanticError);
121      void add_model (in ::UmlCore::ModelElementSet new_value)
122          raises (Reflective::StructuralError, Reflective::SemanticError);
123      void remove_model ()
124          raises (Reflective::SemanticError);
125  };
126
127  interface BindingClass : ::UmlCore::DependencyClass {
128      readonly attribute BindingUList all_of_kind_binding;
129      readonly attribute BindingUList all_of_type_binding;
130      Binding create_binding (in ::UmlCore::Name name,
131          in string description)
132          raises (Reflective::SemanticError);
133  };
134
135  interface Binding : BindingClass, ::UmlCore::Dependency {
136      ::UmlCore::ModelElementSet argument ()
137          raises (Reflective::SemanticError);
138      void add_argument (in ::UmlCore::ModelElementSet new_value)
139          raises (Reflective::StructuralError, Reflective::SemanticError);
140      void remove_argument ()
141          raises (Reflective::SemanticError);
142  };

```

```
143
144 interface CommentClass : ::UmlCore::ModelElementClass {
145     readonly attribute CommentUList all_of_kind_comment;
146     readonly attribute CommentUList all_of_type_comment;
147     Comment create_comment (in ::UmlCore::Name name)
148         raises (Reflective::SemanticError);
149 };
150
151 interface Comment : CommentClass, ::UmlCore::ModelElement { };
152
153 interface RefinementClass : ::UmlCore::DependencyClass {
154     readonly attribute RefinementUList all_of_kind_refinement;
155     readonly attribute RefinementUList all_of_type_refinement;
156     Refinement create_refinement (in ::UmlCore::Name name,
157         in string description,
158         in ::UmlCore::Mapping mapping)
159         raises (Reflective::SemanticError);
160 };
161
162 interface Refinement : RefinementClass, ::UmlCore::Dependency {
163     ::UmlCore::Mapping mapping ()
164         raises (Reflective::SemanticError);
165     void set_mapping (in ::UmlCore::Mapping new_value)
166         raises (Reflective::SemanticError);
167 };
168
169 interface TraceClass : ::UmlCore::DependencyClass {
170     readonly attribute TraceUList all_of_kind_trace;
171     readonly attribute TraceUList all_of_type_trace;
172     Trace create_trace (in ::UmlCore::Name name,
173         in string description)
174         raises (Reflective::SemanticError);
175 };
176
177 interface Trace : TraceClass, ::UmlCore::Dependency { };
178
179 interface UsageClass : ::UmlCore::DependencyClass {
180     readonly attribute UsageUList all_of_kind_usage;
181     readonly attribute UsageUList all_of_type_usage;
182     Usage create_usage (in ::UmlCore::Name name,
183         in string description)
```



```

184     raises (Reflective::SemanticError);
185 };
186
187 interface Usage : UsageClass, ::UmlCore::Dependency { };
188
189 struct ComponentImplementsModelElementLink {
190     Component implementation;
191     ::UmlCore::ModelElement model_element;
192 };
193 typedef sequence <ComponentImplementsModelElementLink>
194     ComponentImplementsModelElementLinkSet;
195
196 interface ComponentImplementsModelElement : Reflective::RefAssociation {
197     readonly attribute UmlAuxiliaryElementsPackage enclosing_package_ref;
198     ComponentImplementsModelElementLinkSet
199     all_component_implements_model_element_links();
200     boolean exists (in Component implementation,
201                   in ::UmlCore::ModelElement model_element);
202     ComponentSet with_model_element (in ::UmlCore::ModelElement model_element);
203     ::UmlCore::ModelElementSet with_implementation (
204         in Component implementation);
205     void add (in Component implementation,
206             in ::UmlCore::ModelElement model_element)
207         raises (Reflective::StructuralError, Reflective::SemanticError);
208     void modify_implementation (in Component implementation,
209                                in ::UmlCore::ModelElement model_element,
210                                in Component new_implementation)
211         raises (Reflective::StructuralError,
212               Reflective::SemanticError,
213               Reflective::NotFound);
214     void modify_model_element (in Component implementation,
215                               in ::UmlCore::ModelElement model_element,
216                               in ::UmlCore::ModelElement new_model_element)
217         raises (Reflective::StructuralError,
218               Reflective::SemanticError,
219               Reflective::NotFound);
220     void remove (in Component implementation,
221                in ::UmlCore::ModelElement model_element)
222         raises (Reflective::StructuralError,
223               Reflective::SemanticError,
224               Reflective::NotFound);

```

```
225 };
226
227 struct NodeDeploysComponentLink {
228     Node deployment;
229     UmlAuxiliaryElements::Component component;
230 };
231 typedef sequence <NodeDeploysComponentLink> NodeDeploysComponentLinkSet;
232
233 interface NodeDeploysComponent : Reflective::RefAssociation {
234     readonly attribute UmlAuxiliaryElementsPackage enclosing_package_ref;
235     NodeDeploysComponentLinkSet all_node_deploys_component_links();
236     boolean exists (in Node deployment,
237                    in UmlAuxiliaryElements::Component component);
238     NodeSet with_component (in UmlAuxiliaryElements::Component component);
239     UmlAuxiliaryElements::ComponentSet with_deployment (in Node deployment);
240     void add (in Node deployment, in UmlAuxiliaryElements::Component component)
241             raises (Reflective::StructuralError, Reflective::SemanticError);
242     void modify_deployment (in Node deployment,
243                             in UmlAuxiliaryElements::Component component,
244                             in Node new_deployment)
245             raises (Reflective::StructuralError,
246                   Reflective::SemanticError,
247                   Reflective::NotFound);
248     void modify_component (in Node deployment,
249                             in UmlAuxiliaryElements::Component component,
250                             in UmlAuxiliaryElements::Component new_component)
251             raises (Reflective::StructuralError,
252                   Reflective::SemanticError,
253                   Reflective::NotFound);
254     void remove (in Node deployment,
255                 in UmlAuxiliaryElements::Component component)
256             raises (Reflective::StructuralError,
257                   Reflective::SemanticError,
258                   Reflective::NotFound);
259 };
260
261 struct PresentationPresentsModelElementLink {
262     ::UmlCore::ModelElement model;
263     UmlAuxiliaryElements::Presentation presentation;
264 };
265 typedef sequence <PresentationPresentsModelElementLink>
```

```

266  PresentationPresentsModelElementLinkSet;
267
268  interface PresentationPresentsModelElement : Reflective::RefAssociation {
269      readonly attribute UmlAuxiliaryElementsPackage enclosing_package_ref;
270      PresentationPresentsModelElementLinkSet
271      all_presentation_presents_model_element_links();
272      boolean exists (in ::UmlCore::ModelElement model,
273                    in UmlAuxiliaryElements::Presentation presentation);
274      ::UmlCore::ModelElement with_presentation (
275          in UmlAuxiliaryElements::Presentation presentation);
276      UmlAuxiliaryElements::PresentationSet with_model (
277          in ::UmlCore::ModelElement model);
278      void add (in ::UmlCore::ModelElement model,
279              in UmlAuxiliaryElements::Presentation presentation)
280          raises (Reflective::StructuralError, Reflective::SemanticError);
281      void modify_model (in ::UmlCore::ModelElement model,
282                       in UmlAuxiliaryElements::Presentation presentation,
283                       in ::UmlCore::ModelElement new_model)
284          raises (Reflective::StructuralError,
285                Reflective::SemanticError,
286                Reflective::NotFound);
287      void modify_presentation (
288          in ::UmlCore::ModelElement model,
289          in UmlAuxiliaryElements::Presentation presentation,
290          in UmlAuxiliaryElements::Presentation new_presentation)
291          raises (Reflective::StructuralError,
292                Reflective::SemanticError,
293                Reflective::NotFound);
294      void remove (in ::UmlCore::ModelElement model,
295                 in UmlAuxiliaryElements::Presentation presentation)
296          raises (Reflective::StructuralError,
297                Reflective::SemanticError,
298                Reflective::NotFound);
299  };
300
301  struct PresentationPresentsViewElementLink {
302      ViewElement view;
303      UmlAuxiliaryElements::Presentation presentation;
304  };
305  typedef sequence <PresentationPresentsViewElementLink>
306      PresentationPresentsViewElementLinkSet;

```

```
307
308 interface PresentationPresentsViewElement : Reflective::RefAssociation {
309     readonly attribute UmlAuxiliaryElementsPackage enclosing_package_ref;
310     PresentationPresentsViewElementLinkSet
311     all_presentation_presents_view_element_links();
312     boolean exists (in ViewElement view,
313         in UmlAuxiliaryElements::Presentation presentation);
314     ViewElement with_presentation (
315         in UmlAuxiliaryElements::Presentation presentation);
316     UmlAuxiliaryElements::PresentationSet with_view (in ViewElement view);
317     void add (in ViewElement view,
318         in UmlAuxiliaryElements::Presentation presentation)
319     raises (Reflective::StructuralError, Reflective::SemanticError);
320     void modify_view (in ViewElement view,
321         in UmlAuxiliaryElements::Presentation presentation,
322         in ViewElement new_view)
323     raises (Reflective::StructuralError,
324         Reflective::SemanticError,
325         Reflective::NotFound);
326     void modify_presentation (
327         in ViewElement view,
328         in UmlAuxiliaryElements::Presentation presentation,
329         in UmlAuxiliaryElements::Presentation new_presentation)
330     raises (Reflective::StructuralError,
331         Reflective::SemanticError,
332         Reflective::NotFound);
333     void remove (in ViewElement view,
334         in UmlAuxiliaryElements::Presentation presentation)
335     raises (Reflective::StructuralError,
336         Reflective::SemanticError,
337         Reflective::NotFound);
338 };
339
340 struct BindingOwnsArgumentModelElementLink {
341     UmlAuxiliaryElements::Binding binding;
342     ::UmlCore::ModelElement argument;
343 };
344 typedef sequence <BindingOwnsArgumentModelElementLink>
345     BindingOwnsArgumentModelElementLinkSet;
346
347 interface BindingOwnsArgumentModelElement : Reflective::RefAssociation {
```

```

348     readonly attribute UmlAuxiliaryElementsPackage enclosing_package_ref;
349     BindingOwnsArgumentModelElementLinkSet
350     all_binding_owns_argument_model_element_links();
351     boolean exists (in UmlAuxiliaryElements::Binding binding,
352                   in ::UmlCore::ModelElement argument);
353     UmlAuxiliaryElements::Binding with_argument (
354         in ::UmlCore::ModelElement argument);
355     ::UmlCore::ModelElementSet with_binding (
356         in UmlAuxiliaryElements::Binding binding);
357     void add (in UmlAuxiliaryElements::Binding binding,
358             in ::UmlCore::ModelElement argument)
359         raises (Reflective::StructuralError, Reflective::SemanticError);
360     void modify_binding (in UmlAuxiliaryElements::Binding binding,
361                        in ::UmlCore::ModelElement argument,
362                        in UmlAuxiliaryElements::Binding new_binding)
363         raises (Reflective::StructuralError,
364               Reflective::SemanticError,
365               Reflective::NotFound);
366     void modify_argument (in UmlAuxiliaryElements::Binding binding,
367                          in ::UmlCore::ModelElement argument,
368                          in ::UmlCore::ModelElement new_argument)
369         raises (Reflective::StructuralError,
370               Reflective::SemanticError,
371               Reflective::NotFound);
372     void remove (in UmlAuxiliaryElements::Binding binding,
373                in ::UmlCore::ModelElement argument)
374         raises (Reflective::StructuralError,
375               Reflective::SemanticError,
376               Reflective::NotFound);
377 };
378
379 struct ViewElementProvidesViewOfModelElementLink {
380     ViewElement view;
381     ::UmlCore::ModelElement model;
382 };
383 typedef sequence <ViewElementProvidesViewOfModelElementLink>
384     ViewElementProvidesViewOfModelElementLinkSet;
385
386 interface ViewElementProvidesViewOfModelElement :
387     Reflective::RefAssociation {
388     readonly attribute UmlAuxiliaryElementsPackage enclosing_package_ref;

```

```
389 ViewElementProvidesViewOfModelElementLinkSet
390   all_view_element_provides_view_of_model_element_links();
391 boolean exists (in ViewElement view, in ::UmlCore::ModelElement model);
392 ViewElementSet with_model (in ::UmlCore::ModelElement model);
393 ::UmlCore::ModelElementSet with_view (in ViewElement view);
394 void add (in ViewElement view, in ::UmlCore::ModelElement model)
395   raises (Reflective::StructuralError, Reflective::SemanticError);
396 void modify_view (in ViewElement view,
397   in ::UmlCore::ModelElement model,
398   in ViewElement new_view)
399   raises (Reflective::StructuralError,
400     Reflective::SemanticError,
401     Reflective::NotFound);
402 void modify_model (in ViewElement view,
403   in ::UmlCore::ModelElement model,
404   in ::UmlCore::ModelElement new_model)
405   raises (Reflective::StructuralError,
406     Reflective::SemanticError,
407     Reflective::NotFound);
408 void remove (in ViewElement view, in ::UmlCore::ModelElement model)
409   raises (Reflective::StructuralError,
410     Reflective::SemanticError,
411     Reflective::NotFound);
412 };
413
414 interface UmlAuxiliaryElementsPackageFactory {
415   UmlAuxiliaryElementsPackage create_uml_auxiliary_elements_package ()
416   raises (Reflective::SemanticError);
417 };
418
419 interface UmlAuxiliaryElementsPackage : Reflective::RefPackage {
420   readonly attribute ComponentClass component_class_ref;
421   readonly attribute NodeClass node_class_ref;
422   readonly attribute PresentationClass presentation_class_ref;
423   readonly attribute ViewElementClass view_element_class_ref;
424   readonly attribute BindingClass binding_class_ref;
425   readonly attribute CommentClass comment_class_ref;
426   readonly attribute RefinementClass refinement_class_ref;
427   readonly attribute TraceClass trace_class_ref;
428   readonly attribute UsageClass usage_class_ref;
429
```

```

430  readonly attribute ComponentImplementsModelElement
431      component_implements_model_element_ref;
432  readonly attribute NodeDeploysComponent node_deploys_component_ref;
433  readonly attribute PresentationPresentsModelElement
434      presentation_presents_model_element_ref;
435  readonly attribute PresentationPresentsViewElement
436      presentation_presents_view_element_ref;
437  readonly attribute BindingOwnsArgumentModelElement
438      binding_owns_argument_model_element_ref;
439  readonly attribute ViewElementProvidesViewOfModelElement
440      view_element_provides_view_of_model_element_ref;
441  };
442 };

```

#### 5.4.4 UML Collaborations

```

1 #include "UmlCommonBehavior.idl"
2
3 module UmlCollaborations {
4  interface UmlCollaborationsPackage;
5  interface Message;
6  interface MessageClass;
7  typedef sequence<Message> MessageUList;
8  typedef sequence<Message> MessageSet;
9  interface ClassifierRole;
10 interface ClassifierRoleClass;
11 typedef sequence<ClassifierRole> ClassifierRoleUList;
12 typedef sequence<ClassifierRole> ClassifierRoleSet;
13 interface AssociationRole;
14 interface AssociationRoleClass;
15 typedef sequence<AssociationRole> AssociationRoleUList;
16 typedef sequence<AssociationRole> AssociationRoleSet;
17 interface Interaction;
18 interface InteractionClass;
19 typedef sequence<Interaction> InteractionUList;
20 interface AssociationEndRole;
21 interface AssociationEndRoleClass;
22 typedef sequence<AssociationEndRole> AssociationEndRoleUList;
23 typedef sequence<AssociationEndRole> AssociationEndRoleSet;
24 interface Collaboration;
25 interface CollaborationClass;

```

```
26 typedef sequence<Collaboration> CollaborationUList;
27 typedef sequence<Collaboration> CollaborationSet;
28
29 interface AssociationEndRoleClass : ::UmlCore::AssociationEndClass {
30     readonly attribute AssociationEndRoleUList
31         all_of_kind_association_end_role;
32     readonly attribute AssociationEndRoleUList
33         all_of_type_association_end_role;
34     AssociationEndRole create_association_end_role (
35         in ::UmlCore::Name name,
36         in boolean is_navigable,
37         in boolean is_ordered,
38         in ::UmlCore::AggregationKind aggregation,
39         in ::UmlCore::ScopeKind target_scope,
40         in ::UmlCore::Multiplicity multiplicity,
41         in ::UmlCore::ChangeableKind changeable)
42     raises (Reflective::SemanticError);
43 };
44
45 interface AssociationEndRole : AssociationEndRoleClass,
46     ::UmlCore::AssociationEnd {
47     ::UmlCore::AssociationEnd base ()
48     raises (Reflective::SemanticError);
49     void set_base (in ::UmlCore::AssociationEnd new_value)
50     raises (Reflective::SemanticError);
51 };
52
53 interface ClassifierRoleClass : ::UmlCore::ClassifierClass {
54     readonly attribute ClassifierRoleUList all_of_kind_classifier_role;
55     readonly attribute ClassifierRoleUList all_of_type_classifier_role;
56     ClassifierRole create_classifier_role (
57         in ::UmlCore::Name name,
58         in boolean is_root,
59         in boolean is_leaf,
60         in boolean is_abstract,
61         in ::UmlCore::Multiplicity multiplicity)
62     raises (Reflective::SemanticError);
63 };
64
65 interface ClassifierRole : ClassifierRoleClass, ::UmlCore::Classifier {
66     ::UmlCore::Multiplicity multiplicity ()
```



```

67     raises (Reflective::SemanticError);
68 void set_multiplicity (in ::UmlCore::Multiplicity new_value)
69     raises (Reflective::SemanticError);
70 ::UmlCore::Classifier base ()
71     raises (Reflective::SemanticError);
72 void set_base (in ::UmlCore::Classifier new_value)
73     raises (Reflective::SemanticError);
74 ::UmlCore::FeatureSet available_feature ()
75     raises (Reflective::NotSet, Reflective::SemanticError);
76 void add_available_feature (in ::UmlCore::FeatureSet new_value)
77     raises (Reflective::StructuralError, Reflective::SemanticError);
78 void remove_available_feature ()
79     raises (Reflective::SemanticError);
80 UmlCollaborations::MessageSet message ()
81     raises (Reflective::NotSet, Reflective::SemanticError);
82 void add_message (in UmlCollaborations::MessageSet new_value)
83     raises (Reflective::StructuralError, Reflective::SemanticError);
84 void remove_message ()
85     raises (Reflective::SemanticError);
86 UmlCollaborations::MessageSet received_message ()
87     raises (Reflective::NotSet, Reflective::SemanticError);
88 void add_received_message (in UmlCollaborations::MessageSet new_value)
89     raises (Reflective::StructuralError, Reflective::SemanticError);
90 void remove_received_message ()
91     raises (Reflective::SemanticError);
92 };
93
94 interface MessageClass : ::UmlCore::ModelElementClass {
95     readonly attribute MessageUList all_of_kind_message;
96     readonly attribute MessageUList all_of_type_message;
97     Message create_message (in ::UmlCore::Name name)
98     raises (Reflective::SemanticError);
99 };
100
101 interface Message : MessageClass, ::UmlCore::ModelElement {
102     UmlCollaborations::InteractionUList interaction ()
103     raises (Reflective::NotSet, Reflective::SemanticError);
104     void add_interaction (in UmlCollaborations::InteractionUList new_value)
105     raises (Reflective::StructuralError, Reflective::SemanticError);
106     void add_interaction_before (in UmlCollaborations::Interaction new_value,
107                                 in UmlCollaborations::Interaction before)

```

```
108     raises (Reflective::StructuralError,
109           Reflective::NotFound,
110           Reflective::SemanticError);
111 void remove_interaction ()
112     raises (Reflective::SemanticError);
113 Message activator ()
114     raises (Reflective::NotSet, Reflective::SemanticError);
115 void set_activator (in Message new_value)
116     raises (Reflective::SemanticError);
117 void unset_activator ()
118     raises (Reflective::SemanticError);
119 MessageUList activated_message ()
120     raises (Reflective::NotSet, Reflective::SemanticError);
121 void add_activated_message (in MessageUList new_value)
122     raises (Reflective::StructuralError, Reflective::SemanticError);
123 void add_activated_message_before (in Message new_value,
124                                   in Message before)
125     raises (Reflective::StructuralError,
126           Reflective::NotFound,
127           Reflective::SemanticError);
128 void remove_activated_message ()
129     raises (Reflective::SemanticError);
130 ::UmlCommonBehavior::Action action ()
131     raises (Reflective::SemanticError);
132 void set_action (in ::UmlCommonBehavior::Action new_value)
133     raises (Reflective::SemanticError);
134 ClassifierRole sender ()
135     raises (Reflective::SemanticError);
136 void set_sender (in ClassifierRole new_value)
137     raises (Reflective::SemanticError);
138 ClassifierRole receiver ()
139     raises (Reflective::SemanticError);
140 void set_receiver (in ClassifierRole new_value)
141     raises (Reflective::SemanticError);
142 MessageSet predecessor ()
143     raises (Reflective::NotSet, Reflective::SemanticError);
144 void add_predecessor (in MessageSet new_value)
145     raises (Reflective::StructuralError, Reflective::SemanticError);
146 void remove_predecessor ()
147     raises (Reflective::SemanticError);
148 MessageSet successor ()
```

```

149     raises (Reflective::NotSet, Reflective::SemanticError);
150 void add_successor (in MessageSet new_value)
151     raises (Reflective::StructuralError, Reflective::SemanticError);
152 void remove_successor ()
153     raises (Reflective::SemanticError);
154 };
155
156 interface InteractionClass : ::UmlCore::ModelElementClass {
157     readonly attribute InteractionUList all_of_kind_interaction;
158     readonly attribute InteractionUList all_of_type_interaction;
159     Interaction create_interaction (in ::UmlCore::Name name)
160     raises (Reflective::SemanticError);
161 };
162
163 interface Interaction : InteractionClass, ::UmlCore::ModelElement {
164     UmlCollaborations::MessageSet message ()
165     raises (Reflective::SemanticError);
166     void add_message (in UmlCollaborations::MessageSet new_value)
167     raises (Reflective::StructuralError, Reflective::SemanticError);
168     void remove_message ()
169     raises (Reflective::SemanticError);
170     Collaboration uml_context ()
171     raises (Reflective::NotSet, Reflective::SemanticError);
172     void set_uml_context (in Collaboration new_value)
173     raises (Reflective::SemanticError);
174     void unset_uml_context ()
175     raises (Reflective::SemanticError);
176 };
177
178 interface AssociationRoleClass : ::UmlCore::AssociationClass {
179     readonly attribute AssociationRoleUList all_of_kind_association_role;
180     readonly attribute AssociationRoleUList all_of_type_association_role;
181     AssociationRole create_association_role (
182         in ::UmlCore::Name name,
183         in boolean is_root,
184         in boolean is_leaf,
185         in boolean is_abstract,
186         in ::UmlCore::Multiplicity multiplicity)
187     raises (Reflective::SemanticError);
188 };
189

```

```
190 interface AssociationRole : AssociationRoleClass, ::UmlCore::Association {
191     ::UmlCore::Multiplicity multiplicity ()
192     raises (Reflective::SemanticError);
193     void set_multiplicity (in ::UmlCore::Multiplicity new_value)
194     raises (Reflective::SemanticError);
195     ::UmlCore::Association base ()
196     raises (Reflective::SemanticError);
197     void set_base (in ::UmlCore::Association new_value)
198     raises (Reflective::SemanticError);
199 };
200
201 interface CollaborationClass : ::UmlCore::NamespaceClass {
202     readonly attribute CollaborationUList all_of_kind_collaboration;
203     readonly attribute CollaborationUList all_of_type_collaboration;
204     Collaboration create_collaboration (in ::UmlCore::Name name)
205     raises (Reflective::SemanticError);
206 };
207
208 interface Collaboration : CollaborationClass, ::UmlCore::Namespace {
209     UmlCollaborations::InteractionUList interaction ()
210     raises (Reflective::NotSet, Reflective::SemanticError);
211     void add_interaction (in UmlCollaborations::InteractionUList new_value)
212     raises (Reflective::StructuralError, Reflective::SemanticError);
213     void add_interaction_before (in UmlCollaborations::Interaction new_value,
214         in UmlCollaborations::Interaction before)
215     raises (Reflective::StructuralError,
216         Reflective::NotFound,
217         Reflective::SemanticError);
218     void remove_interaction ()
219     raises (Reflective::SemanticError);
220     ::UmlCore::ModelElementSet constraining_element ()
221     raises (Reflective::NotSet, Reflective::SemanticError);
222     void add_constraining_element (in ::UmlCore::ModelElementSet new_value)
223     raises (Reflective::StructuralError, Reflective::SemanticError);
224     void remove_constraining_element ()
225     raises (Reflective::SemanticError);
226     ::UmlCore::Classifier represented_classifier ()
227     raises (Reflective::NotSet, Reflective::SemanticError);
228     void set_represented_classifier (in ::UmlCore::Classifier new_value)
229     raises (Reflective::SemanticError);
230     void unset_represented_classifier ()
```

```

231     raises (Reflective::SemanticError);
232     ::UmlCore::Operation represented_operation ()
233     raises (Reflective::NotSet, Reflective::SemanticError);
234     void set_represented_operation (in ::UmlCore::Operation new_value)
235     raises (Reflective::SemanticError);
236     void unset_represented_operation ()
237     raises (Reflective::SemanticError);
238 };
239
240 struct InteractionContainsMessageLink {
241     UmlCollaborations::Interaction interaction;
242     UmlCollaborations::Message message;
243 };
244 typedef sequence <InteractionContainsMessageLink>
245     InteractionContainsMessageLinkSet;
246
247 interface InteractionContainsMessage : Reflective::RefAssociation {
248     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
249     InteractionContainsMessageLinkSet
250     all_interaction_contains_message_links();
251     boolean exists (in UmlCollaborations::Interaction interaction,
252                   in UmlCollaborations::Message message);
253     UmlCollaborations::InteractionUList
254     with_message (in UmlCollaborations::Message message);
255     UmlCollaborations::MessageSet with_interaction (
256         in UmlCollaborations::Interaction interaction);
257     void add (in UmlCollaborations::Interaction interaction,
258             in UmlCollaborations::Message message)
259     raises (Reflective::StructuralError, Reflective::SemanticError);
260     void add_before_interaction (in UmlCollaborations::Interaction interaction,
261                                in UmlCollaborations::Message message,
262                                in UmlCollaborations::Interaction before)
263     raises (Reflective::StructuralError,
264           Reflective::SemanticError,
265           Reflective::NotFound);
266     void modify_interaction (in UmlCollaborations::Interaction interaction,
267                             in UmlCollaborations::Message message,
268                             in UmlCollaborations::Interaction new_interaction)
269     raises (Reflective::StructuralError,
270           Reflective::SemanticError,
271           Reflective::NotFound);

```

```
272 void modify_message (in UmlCollaborations::Interaction interaction,
273                      in UmlCollaborations::Message message,
274                      in UmlCollaborations::Message new_message)
275     raises (Reflective::StructuralError,
276            Reflective::SemanticError,
277            Reflective::NotFound);
278 void remove (in UmlCollaborations::Interaction interaction,
279             in UmlCollaborations::Message message)
280     raises (Reflective::StructuralError,
281            Reflective::SemanticError,
282            Reflective::NotFound);
283 };
284
285 struct CollaborationOwnsInteractionLink {
286     Collaboration uml_context;
287     UmlCollaborations::Interaction interaction;
288 };
289 typedef sequence <CollaborationOwnsInteractionLink>
290     CollaborationOwnsInteractionLinkSet;
291
292 interface CollaborationOwnsInteraction : Reflective::RefAssociation {
293     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
294     CollaborationOwnsInteractionLinkSet
295         all_collaboration_owns_interaction_links();
296     boolean exists (in Collaboration uml_context,
297                   in UmlCollaborations::Interaction interaction);
298     Collaboration with_interaction (
299         in UmlCollaborations::Interaction interaction);
300     UmlCollaborations::InteractionUList with_uml_context (
301         in Collaboration uml_context);
302     void add (in Collaboration uml_context,
303             in UmlCollaborations::Interaction interaction)
304         raises (Reflective::StructuralError, Reflective::SemanticError);
305     void add_before_interaction (in Collaboration uml_context,
306                                in UmlCollaborations::Interaction interaction,
307                                in UmlCollaborations::Interaction before)
308         raises (Reflective::StructuralError,
309                Reflective::SemanticError,
310                Reflective::NotFound);
311     void modify_uml_context (in Collaboration uml_context,
312                             in UmlCollaborations::Interaction interaction,
```

```

313         in Collaboration new_uml_context)
314     raises (Reflective::StructuralError,
315           Reflective::SemanticError,
316           Reflective::NotFound);
317 void modify_interaction (in Collaboration uml_context,
318                         in UmlCollaborations::Interaction interaction,
319                         in UmlCollaborations::Interaction new_interaction)
320     raises (Reflective::StructuralError,
321           Reflective::SemanticError,
322           Reflective::NotFound);
323 void remove (in Collaboration uml_context,
324             in UmlCollaborations::Interaction interaction)
325     raises (Reflective::StructuralError,
326           Reflective::SemanticError,
327           Reflective::NotFound);
328 };
329
330 struct ClassifierRoleHasBaseOfClassifierLink {
331     UmlCollaborations::ClassifierRole classifier_role;
332     ::UmlCore::Classifier base;
333 };
334 typedef sequence <ClassifierRoleHasBaseOfClassifierLink>
335     ClassifierRoleHasBaseOfClassifierLinkSet;
336
337 interface ClassifierRoleHasBaseOfClassifier : Reflective::RefAssociation {
338     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
339     ClassifierRoleHasBaseOfClassifierLinkSet
340     all_classifier_role_has_base_of_classifier_links();
341     boolean exists (in UmlCollaborations::ClassifierRole classifier_role,
342                  in ::UmlCore::Classifier base);
343     UmlCollaborations::ClassifierRoleSet with_base (
344         in ::UmlCore::Classifier base);
345     ::UmlCore::Classifier with_classifier_role (
346         in UmlCollaborations::ClassifierRole classifier_role);
347     void add (in UmlCollaborations::ClassifierRole classifier_role,
348             in ::UmlCore::Classifier base)
349         raises (Reflective::StructuralError, Reflective::SemanticError);
350     void modify_classifier_role (
351         in UmlCollaborations::ClassifierRole classifier_role,
352         in ::UmlCore::Classifier base,
353         in UmlCollaborations::ClassifierRole new_classifier_role)

```

```

354     raises (Reflective::StructuralError,
355           Reflective::SemanticError,
356           Reflective::NotFound);
357 void modify_base (in UmlCollaborations::ClassifierRole classifier_role,
358                 in ::UmlCore::Classifier base,
359                 in ::UmlCore::Classifier new_base)
360     raises (Reflective::StructuralError,
361           Reflective::SemanticError,
362           Reflective::NotFound);
363 void remove (in UmlCollaborations::ClassifierRole classifier_role,
364             in ::UmlCore::Classifier base)
365     raises (Reflective::StructuralError,
366           Reflective::SemanticError,
367           Reflective::NotFound);
368 };
369
370 struct AssociationEndRoleHasBaseOfAssociationEndLink {
371     ::UmlCore::AssociationEnd base;
372     UmlCollaborations::AssociationEndRole association_end_role;
373 };
374 typedef sequence <AssociationEndRoleHasBaseOfAssociationEndLink>
375     AssociationEndRoleHasBaseOfAssociationEndLinkSet;
376
377 interface AssociationEndRoleHasBaseOfAssociationEnd :
378     Reflective::RefAssociation {
379     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
380     AssociationEndRoleHasBaseOfAssociationEndLinkSet
381     all_association_end_role_has_base_of_association_end_links();
382     boolean
383     exists (in ::UmlCore::AssociationEnd base,
384           in UmlCollaborations::AssociationEndRole association_end_role);
385     ::UmlCore::AssociationEnd with_association_end_role (
386     in UmlCollaborations::AssociationEndRole association_end_role);
387     UmlCollaborations::AssociationEndRoleSet with_base (
388     in ::UmlCore::AssociationEnd base);
389     void add (in ::UmlCore::AssociationEnd base,
390             in UmlCollaborations::AssociationEndRole association_end_role)
391     raises (Reflective::StructuralError, Reflective::SemanticError);
392     void modify_base (
393     in ::UmlCore::AssociationEnd base,
394     in UmlCollaborations::AssociationEndRole association_end_role,

```



```

395         in ::UmlCore::AssociationEnd new_base)
396     raises (Reflective::StructuralError,
397           Reflective::SemanticError,
398           Reflective::NotFound);
399 void modify_association_end_role (
400     in ::UmlCore::AssociationEnd base,
401     in UmlCollaborations::AssociationEndRole association_end_role,
402     in UmlCollaborations::AssociationEndRole new_association_end_role)
403     raises (Reflective::StructuralError,
404           Reflective::SemanticError,
405           Reflective::NotFound);
406 void remove (in ::UmlCore::AssociationEnd base,
407             in UmlCollaborations::AssociationEndRole association_end_role)
408     raises (Reflective::StructuralError,
409           Reflective::SemanticError,
410           Reflective::NotFound);
411 };
412
413 struct AssociationRoleHasBaseOfAssociationLink {
414     ::UmlCore::Association base;
415     UmlCollaborations::AssociationRole association_role;
416 };
417 typedef sequence <AssociationRoleHasBaseOfAssociationLink>
418     AssociationRoleHasBaseOfAssociationLinkSet;
419
420 interface AssociationRoleHasBaseOfAssociation : Reflective::RefAssociation {
421     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
422     AssociationRoleHasBaseOfAssociationLinkSet
423     all_association_role_has_base_of_association_links();
424     boolean exists (in ::UmlCore::Association base,
425                   in UmlCollaborations::AssociationRole association_role);
426     ::UmlCore::Association with_association_role (
427         in UmlCollaborations::AssociationRole association_role);
428     UmlCollaborations::AssociationRoleSet with_base (
429         in ::UmlCore::Association base);
430     void add (in ::UmlCore::Association base,
431             in UmlCollaborations::AssociationRole association_role)
432         raises (Reflective::StructuralError, Reflective::SemanticError);
433     void modify_base (in ::UmlCore::Association base,
434                     in UmlCollaborations::AssociationRole association_role,
435                     in ::UmlCore::Association new_base)

```

```
436     raises (Reflective::StructuralError,
437           Reflective::SemanticError,
438           Reflective::NotFound);
439 void modify_association_role (
440     in ::UmlCore::Association base,
441     in UmlCollaborations::AssociationRole association_role,
442     in UmlCollaborations::AssociationRole new_association_role)
443     raises (Reflective::StructuralError,
444           Reflective::SemanticError,
445           Reflective::NotFound);
446 void remove (in ::UmlCore::Association base,
447             in UmlCollaborations::AssociationRole association_role)
448     raises (Reflective::StructuralError,
449           Reflective::SemanticError,
450           Reflective::NotFound);
451 };
452
453 struct ClassifierRoleProvidesAvailableFeaturesLink {
454     UmlCollaborations::ClassifierRole classifier_role;
455     ::UmlCore::Feature available_feature;
456 };
457 typedef sequence <ClassifierRoleProvidesAvailableFeaturesLink>
458 ClassifierRoleProvidesAvailableFeaturesLinkSet;
459
460 interface ClassifierRoleProvidesAvailableFeatures :
461     Reflective::RefAssociation {
462     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
463     ClassifierRoleProvidesAvailableFeaturesLinkSet
464     all_classifier_role_provides_available_features_links();
465     boolean exists (in UmlCollaborations::ClassifierRole classifier_role,
466                  in ::UmlCore::Feature available_feature);
467     UmlCollaborations::ClassifierRoleSet with_available_feature (
468         in ::UmlCore::Feature available_feature);
469     ::UmlCore::FeatureSet with_classifier_role (
470         in UmlCollaborations::ClassifierRole classifier_role);
471     void add (in UmlCollaborations::ClassifierRole classifier_role,
472            in ::UmlCore::Feature available_feature)
473         raises (Reflective::StructuralError, Reflective::SemanticError);
474     void modify_classifier_role (
475         in UmlCollaborations::ClassifierRole classifier_role,
476         in ::UmlCore::Feature available_feature,
```

```

477         in UmlCollaborations::ClassifierRole new_classifier_role)
478     raises (Reflective::StructuralError,
479           Reflective::SemanticError,
480           Reflective::NotFound);
481 void modify_available_feature (
482     in UmlCollaborations::ClassifierRole classifier_role,
483     in ::UmlCore::Feature available_feature,
484     in ::UmlCore::Feature new_available_feature)
485     raises (Reflective::StructuralError,
486           Reflective::SemanticError,
487           Reflective::NotFound);
488 void remove (in UmlCollaborations::ClassifierRole classifier_role,
489             in ::UmlCore::Feature available_feature)
490     raises (Reflective::StructuralError,
491           Reflective::SemanticError,
492           Reflective::NotFound);
493 };
494
495 struct CollaborationHasConstrainingModelElementLink {
496     Collaboration constrained_collab;
497     ::UmlCore::ModelElement constraining_element;
498 };
499 typedef sequence <CollaborationHasConstrainingModelElementLink>
500 CollaborationHasConstrainingModelElementLinkSet;
501
502 interface CollaborationHasConstrainingModelElement :
503     Reflective::RefAssociation {
504     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
505     CollaborationHasConstrainingModelElementLinkSet
506     all_collaboration_has_constraining_model_element_links();
507     boolean exists (in Collaboration constrained_collab,
508                  in ::UmlCore::ModelElement constraining_element);
509     CollaborationSet with_constraining_element (
510         in ::UmlCore::ModelElement constraining_element);
511     ::UmlCore::ModelElementSet with_constrained_collab (
512         in Collaboration constrained_collab);
513     void add (in Collaboration constrained_collab,
514             in ::UmlCore::ModelElement constraining_element)
515         raises (Reflective::StructuralError, Reflective::SemanticError);
516     void modify_constrained_collab (
517         in Collaboration constrained_collab,

```

```

518         in ::UmlCore::ModelElement constraining_element,
519         in Collaboration new_constrained_collab)
520     raises (Reflective::StructuralError,
521           Reflective::SemanticError,
522           Reflective::NotFound);
523 void modify_constraining_element (
524     in Collaboration constrained_collab,
525     in ::UmlCore::ModelElement constraining_element,
526     in ::UmlCore::ModelElement new_constraining_element)
527     raises (Reflective::StructuralError,
528           Reflective::SemanticError,
529           Reflective::NotFound);
530 void remove (in Collaboration constrained_collab,
531             in ::UmlCore::ModelElement constraining_element)
532     raises (Reflective::StructuralError,
533           Reflective::SemanticError,
534           Reflective::NotFound);
535 };
536
537 struct MessageActivatesMessageLink {
538     Message activator;
539     Message activated_message;
540 };
541 typedef sequence <MessageActivatesMessageLink> MessageActivatesMessageLinkSet;
542
543 interface MessageActivatesMessage : Reflective::RefAssociation {
544     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
545     MessageActivatesMessageLinkSet all_message_activates_message_links();
546     boolean exists (in Message activator, in Message activated_message);
547     Message with_activated_message (in Message activated_message);
548     MessageUList with_activator (in Message activator);
549     void add (in Message activator, in Message activated_message)
550         raises (Reflective::StructuralError, Reflective::SemanticError);
551     void add_before_activated_message (in Message activator,
552                                       in Message activated_message,
553                                       in Message before)
554         raises (Reflective::StructuralError,
555               Reflective::SemanticError,
556               Reflective::NotFound);
557     void modify_activator (in Message activator,
558                           in Message activated_message,

```

```

559         in Message new_activator)
560     raises (Reflective::StructuralError,
561           Reflective::SemanticError,
562           Reflective::NotFound);
563 void modify_activated_message (in Message activator,
564                               in Message activated_message,
565                               in Message new_activated_message)
566     raises (Reflective::StructuralError,
567           Reflective::SemanticError,
568           Reflective::NotFound);
569 void remove (in Message activator, in Message activated_message)
570     raises (Reflective::StructuralError,
571           Reflective::SemanticError,
572           Reflective::NotFound);
573 };
574
575 struct CollaborationRepresentsClassifierLink {
576     Collaboration representing_collaboration;
577     ::UmlCore::Classifier represented_classifier;
578 };
579 typedef sequence <CollaborationRepresentsClassifierLink>
580     CollaborationRepresentsClassifierLinkSet;
581
582 interface CollaborationRepresentsClassifier : Reflective::RefAssociation {
583     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
584     CollaborationRepresentsClassifierLinkSet
585         all_collaboration_represents_classifier_links();
586     boolean exists (in Collaboration representing_collaboration,
587                   in ::UmlCore::Classifier represented_classifier);
588     CollaborationSet with_represented_classifier (
589         in ::UmlCore::Classifier represented_classifier);
590     ::UmlCore::Classifier with_representing_collaboration (
591         in Collaboration representing_collaboration);
592     void add (in Collaboration representing_collaboration,
593             in ::UmlCore::Classifier represented_classifier)
594         raises (Reflective::StructuralError, Reflective::SemanticError);
595     void modify_representing_collaboration (
596         in Collaboration representing_collaboration,
597         in ::UmlCore::Classifier represented_classifier,
598         in Collaboration new_representing_collaboration)
599         raises (Reflective::StructuralError,

```

```

600         Reflective::SemanticError,
601         Reflective::NotFound);
602     void modify_represented_classifier (
603         in Collaboration representing_collaboration,
604         in ::UmlCore::Classifier represented_classifier,
605         in ::UmlCore::Classifier new_represented_classifier)
606     raises (Reflective::StructuralError,
607           Reflective::SemanticError,
608           Reflective::NotFound);
609     void remove (in Collaboration representing_collaboration,
610                in ::UmlCore::Classifier represented_classifier)
611     raises (Reflective::StructuralError,
612           Reflective::SemanticError,
613           Reflective::NotFound);
614 };
615
616 struct CollaborationRepresentsOperationLink {
617     Collaboration representing_collaboration;
618     ::UmlCore::Operation represented_operation;
619 };
620 typedef sequence <CollaborationRepresentsOperationLink>
621 CollaborationRepresentsOperationLinkSet;
622
623 interface CollaborationRepresentsOperation : Reflective::RefAssociation {
624     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
625     CollaborationRepresentsOperationLinkSet
626     all_collaboration_represents_operation_links();
627     boolean exists (in Collaboration representing_collaboration,
628                   in ::UmlCore::Operation represented_operation);
629     CollaborationSet with_represented_operation (
630         in ::UmlCore::Operation represented_operation);
631     ::UmlCore::Operation with_representing_collaboration (
632         in Collaboration representing_collaboration);
633     void add (in Collaboration representing_collaboration,
634              in ::UmlCore::Operation represented_operation)
635     raises (Reflective::StructuralError, Reflective::SemanticError);
636     void modify_representing_collaboration (
637         in Collaboration representing_collaboration,
638         in ::UmlCore::Operation represented_operation,
639         in Collaboration new_representing_collaboration)
640     raises (Reflective::StructuralError,

```

```

641         Reflective::SemanticError,
642         Reflective::NotFound);
643 void modify_represented_operation (
644     in Collaboration representing_collaboration,
645     in ::UmlCore::Operation represented_operation,
646     in ::UmlCore::Operation new_represented_operation)
647     raises (Reflective::StructuralError,
648           Reflective::SemanticError,
649           Reflective::NotFound);
650 void remove (in Collaboration representing_collaboration,
651             in ::UmlCore::Operation represented_operation)
652     raises (Reflective::StructuralError,
653           Reflective::SemanticError,
654           Reflective::NotFound);
655 };
656
657 struct MessageIsSentByActionLink {
658     Message message0;
659     ::UmlCommonBehavior::Action action;
660 };
661 typedef sequence <MessageIsSentByActionLink> MessageIsSentByActionLinkSet;
662
663 interface MessageIsSentByAction : Reflective::RefAssociation {
664     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
665     MessageIsSentByActionLinkSet all_message_is_sent_by_action_links();
666     boolean exists (in Message message0,
667                   in ::UmlCommonBehavior::Action action);
668     MessageSet with_action (in ::UmlCommonBehavior::Action action);
669     ::UmlCommonBehavior::Action with_message0 (in Message message0);
670     void add (in Message message0,
671             in ::UmlCommonBehavior::Action action)
672         raises (Reflective::StructuralError, Reflective::SemanticError);
673     void modify_message0 (in Message message0,
674                          in ::UmlCommonBehavior::Action action,
675                          in Message new_message0)
676         raises (Reflective::StructuralError,
677               Reflective::SemanticError,
678               Reflective::NotFound);
679     void modify_action (in Message message0,
680                       in ::UmlCommonBehavior::Action action,
681                       in ::UmlCommonBehavior::Action new_action)

```

```
682     raises (Reflective::StructuralError,
683           Reflective::SemanticError,
684           Reflective::NotFound);
685 void remove (in Message message0,
686             in ::UmlCommonBehavior::Action action)
687     raises (Reflective::StructuralError,
688           Reflective::SemanticError,
689           Reflective::NotFound);
690 };
691
692 struct MessageIsSentByClassifierRoleLink {
693     UmlCollaborations::Message message;
694     ClassifierRole sender;
695 };
696 typedef sequence <MessageIsSentByClassifierRoleLink>
697     MessageIsSentByClassifierRoleLinkSet;
698
699 interface MessageIsSentByClassifierRole : Reflective::RefAssociation {
700     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
701     MessageIsSentByClassifierRoleLinkSet
702     all_message_is_sent_by_classifier_role_links();
703     boolean exists (in UmlCollaborations::Message message,
704                  in ClassifierRole sender);
705     UmlCollaborations::MessageSet with_sender (in ClassifierRole sender);
706     ClassifierRole with_message (in UmlCollaborations::Message message);
707     void add (in UmlCollaborations::Message message, in ClassifierRole sender)
708     raises (Reflective::StructuralError, Reflective::SemanticError);
709     void modify_message (in UmlCollaborations::Message message,
710                        in ClassifierRole sender,
711                        in UmlCollaborations::Message new_message)
712     raises (Reflective::StructuralError,
713           Reflective::SemanticError,
714           Reflective::NotFound);
715     void modify_sender (in UmlCollaborations::Message message,
716                       in ClassifierRole sender,
717                       in ClassifierRole new_sender)
718     raises (Reflective::StructuralError,
719           Reflective::SemanticError,
720           Reflective::NotFound);
721     void remove (in UmlCollaborations::Message message,
722                in ClassifierRole sender)
```



```

723     raises (Reflective::StructuralError,
724           Reflective::SemanticError,
725           Reflective::NotFound);
726 };
727
728 struct MessageIsReceivedByClassifierRoleLink {
729     ClassifierRole receiver;
730     Message received_message;
731 };
732 typedef sequence <MessageIsReceivedByClassifierRoleLink>
733     MessageIsReceivedByClassifierRoleLinkSet;
734
735 interface MessageIsReceivedByClassifierRole : Reflective::RefAssociation {
736     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
737     MessageIsReceivedByClassifierRoleLinkSet
738     all_message_is_received_by_classifier_role_links();
739     boolean exists (in ClassifierRole receiver, in Message received_message);
740     ClassifierRole with_received_message (in Message received_message);
741     MessageSet with_receiver (in ClassifierRole receiver);
742     void add (in ClassifierRole receiver, in Message received_message)
743         raises (Reflective::StructuralError, Reflective::SemanticError);
744     void modify_receiver (in ClassifierRole receiver,
745                          in Message received_message,
746                          in ClassifierRole new_receiver)
747         raises (Reflective::StructuralError,
748               Reflective::SemanticError,
749               Reflective::NotFound);
750     void modify_received_message (in ClassifierRole receiver,
751                                  in Message received_message,
752                                  in Message new_received_message)
753         raises (Reflective::StructuralError,
754               Reflective::SemanticError,
755               Reflective::NotFound);
756     void remove (in ClassifierRole receiver, in Message received_message)
757         raises (Reflective::StructuralError,
758               Reflective::SemanticError,
759               Reflective::NotFound);
760 };
761
762 struct MessageHasPredecessorMessageLink {
763     Message predecessor;

```

```
764 Message successor;
765 };
766 typedef sequence <MessageHasPredecessorMessageLink>
767 MessageHasPredecessorMessageLinkSet;
768
769 interface MessageHasPredecessorMessage : Reflective::RefAssociation {
770     readonly attribute UmlCollaborationsPackage enclosing_package_ref;
771     MessageHasPredecessorMessageLinkSet
772     all_message_has_predecessor_message_links();
773     boolean exists (in Message predecessor, in Message successor);
774     MessageSet with_successor (in Message successor);
775     MessageSet with_predecessor (in Message predecessor);
776     void add (in Message predecessor, in Message successor)
777         raises (Reflective::StructuralError, Reflective::SemanticError);
778     void modify_predecessor (in Message predecessor,
779                             in Message successor,
780                             in Message new_predecessor)
781         raises (Reflective::StructuralError,
782               Reflective::SemanticError,
783               Reflective::NotFound);
784     void modify_successor (in Message predecessor,
785                           in Message successor,
786                           in Message new_successor)
787         raises (Reflective::StructuralError,
788               Reflective::SemanticError,
789               Reflective::NotFound);
790     void remove (in Message predecessor, in Message successor)
791         raises (Reflective::StructuralError,
792               Reflective::SemanticError,
793               Reflective::NotFound);
794 };
795
796 interface UmlCollaborationsPackageFactory {
797     UmlCollaborationsPackage create_uml_collaborations_package ()
798         raises (Reflective::SemanticError);
799 };
800
801 interface UmlCollaborationsPackage : Reflective::RefPackage {
802     readonly attribute AssociationEndRoleClass association_end_role_class_ref;
803     readonly attribute ClassifierRoleClass classifier_role_class_ref;
804     readonly attribute MessageClass message_class_ref;
```

```
805     readonly attribute InteractionClass interaction_class_ref;
806     readonly attribute AssociationRoleClass association_role_class_ref;
807     readonly attribute CollaborationClass collaboration_class_ref;
808
809     readonly attribute InteractionContainsMessage
810         interaction_contains_message_ref;
811     readonly attribute CollaborationOwnsInteraction
812         collaboration_owns_interaction_ref;
813     readonly attribute ClassifierRoleHasBaseOfClassifier
814         classifier_role_has_base_of_classifier_ref;
815     readonly attribute AssociationEndRoleHasBaseOfAssociationEnd
816         association_end_role_has_base_of_association_end_ref;
817     readonly attribute AssociationRoleHasBaseOfAssociation
818         association_role_has_base_of_association_ref;
819     readonly attribute ClassifierRoleProvidesAvailableFeatures
820         classifier_role_provides_available_features_ref;
821     readonly attribute CollaborationHasConstrainingModelElement
822         collaboration_has_constraining_model_element_ref;
823     readonly attribute MessageActivatesMessage message_activates_message_ref;
824     readonly attribute CollaborationRepresentsClassifier
825         collaboration_represents_classifier_ref;
826     readonly attribute CollaborationRepresentsOperation
827         collaboration_represents_operation_ref;
828     readonly attribute MessageIsSentByAction message_is_sent_by_action_ref;
829     readonly attribute MessageIsSentByClassifierRole
830         message_is_sent_by_classifier_role_ref;
831     readonly attribute MessageIsReceivedByClassifierRole
832         message_is_received_by_classifier_role_ref;
833     readonly attribute MessageHasPredecessorMessage
834         message_has_predecessor_message_ref;
835 };
836 };
```

### 5.4.5 *UMLCommonBehavior*

```
1 #include "UmlCore.idl"
2
3 module UmlCommonBehavior {
4   interface UmlCommonBehaviorPackage;
5   interface LinkEnd;
6   interface LinkEndClass;
7   typedef sequence<LinkEnd> LinkEndUList;
8   typedef sequence<LinkEnd> LinkEndSet;
9   interface Action;
10  interface ActionClass;
11  typedef sequence<Action> ActionUList;
12  typedef sequence<Action> ActionSet;
13  interface UmlObject;
14  interface UmlObjectClass;
15  typedef sequence<UmlObject> UmlObjectUList;
16  interface DataValue;
17  interface DataValueClass;
18  typedef sequence<DataValue> DataValueUList;
19  interface UmlInstance;
20  interface UmlInstanceClass;
21  typedef sequence<UmlInstance> UmlInstanceUList;
22  typedef sequence<UmlInstance> UmlInstanceSet;
23  interface ReturnAction;
24  interface ReturnActionClass;
25  typedef sequence<ReturnAction> ReturnActionUList;
26  interface ActionSequence;
27  interface ActionSequenceClass;
28  typedef sequence<ActionSequence> ActionSequenceUList;
29  interface LocalInvocation;
30  interface LocalInvocationClass;
31  typedef sequence<LocalInvocation> LocalInvocationUList;
32  interface Reception;
33  interface ReceptionClass;
34  typedef sequence<Reception> ReceptionUList;
35  typedef sequence<Reception> ReceptionSet;
36  interface Signal;
```

```
37 interface SignalClass;
38 typedef sequence<Signal> SignalUList;
39 interface TerminateAction;
40 interface TerminateActionClass;
41 typedef sequence<TerminateAction> TerminateActionUList;
42 interface MessageInstance;
43 interface MessageInstanceClass;
44 typedef sequence<MessageInstance> MessageInstanceUList;
45 typedef sequence<MessageInstance> MessageInstanceSet;
46 interface Argument;
47 interface ArgumentClass;
48 typedef sequence<Argument> ArgumentUList;
49 interface CallAction;
50 interface CallActionClass;
51 typedef sequence<CallAction> CallActionUList;
52 interface CreateAction;
53 interface CreateActionClass;
54 typedef sequence<CreateAction> CreateActionUList;
55 typedef sequence<CreateAction> CreateActionSet;
56 interface UninterpretedAction;
57 interface UninterpretedActionClass;
58 typedef sequence<UninterpretedAction> UninterpretedActionUList;
59 interface Call;
60 interface CallClass;
61 typedef sequence<Call> CallUList;
62 interface Link;
63 interface LinkClass;
64 typedef sequence<Link> LinkUList;
65 typedef sequence<Link> LinkSet;
66 interface SendAction;
67 interface SendActionClass;
68 typedef sequence<SendAction> SendActionUList;
69 interface AttributeLink;
70 interface AttributeLinkClass;
71 typedef sequence<AttributeLink> AttributeLinkUList;
72 typedef sequence<AttributeLink> AttributeLinkSet;
73 interface UmlException;
74 interface UmlExceptionClass;
75 typedef sequence<UmlException> UmlExceptionUList;
76 typedef sequence<UmlException> UmlExceptionSet;
77 interface DestroyAction;
```

```
78 interface DestroyActionClass;
79 typedef sequence<DestroyAction> DestroyActionUList;
80 interface LinkObject;
81 interface LinkObjectClass;
82 typedef sequence<LinkObject> LinkObjectUList;
83
84 interface CallClass : ::UmlCore::ModelElementClass {
85     readonly attribute CallUList all_of_kind_call;
86     readonly attribute CallUList all_of_type_call;
87     Call create_call (in ::UmlCore::Name name)
88         raises (Reflective::SemanticError);
89 };
90
91 interface Call : CallClass, ::UmlCore::ModelElement { };
92
93 interface LinkEndClass : ::UmlCore::ModelElementClass {
94     readonly attribute LinkEndUList all_of_kind_link_end;
95     readonly attribute LinkEndUList all_of_type_link_end;
96     LinkEnd create_link_end (in ::UmlCore::Name name)
97         raises (Reflective::SemanticError);
98 };
99
100 interface LinkEnd : LinkEndClass, ::UmlCore::ModelElement {
101     UmlCommonBehavior::UmlInstance uml_instance ()
102         raises (Reflective::SemanticError);
103     void set_uml_instance (in UmlCommonBehavior::UmlInstance new_value)
104         raises (Reflective::SemanticError);
105     Link owning_link ()
106         raises (Reflective::SemanticError);
107     void set_owning_link (in Link new_value)
108         raises (Reflective::SemanticError);
109     void add_owning_link_before (in Link new_value, in Link before)
110         raises (Reflective::StructuralError,
111             Reflective::NotFound,
112             Reflective::SemanticError);
113     ::UmlCore::AssociationEnd association_end ()
114         raises (Reflective::SemanticError);
115     void set_association_end (in ::UmlCore::AssociationEnd new_value)
116         raises (Reflective::SemanticError);
117 };
118
```

```

119 interface LinkClass : ::UmlCore::ModelElementClass {
120     readonly attribute LinkUList all_of_kind_link;
121     readonly attribute LinkUList all_of_type_link;
122     Link create_link (in ::UmlCore::Name name)
123         raises (Reflective::SemanticError);
124 };
125
126 interface Link : LinkClass, ::UmlCore::ModelElement {
127     ::UmlCore::Association association ()
128         raises (Reflective::SemanticError);
129     void set_association (in ::UmlCore::Association new_value)
130         raises (Reflective::SemanticError);
131     LinkEndSet link_role ()
132         raises (Reflective::SemanticError);
133     void add_link_role (in LinkEndSet new_value)
134         raises (Reflective::StructuralError, Reflective::SemanticError);
135     void modify_link_role (in LinkEnd old_value, in LinkEnd new_value)
136         raises (Reflective::StructuralError,
137             Reflective::NotFound,
138             Reflective::SemanticError);
139     void remove_link_role ()
140         raises (Reflective::StructuralError, Reflective::SemanticError);
141 };
142
143 interface AttributeLinkClass : ::UmlCore::ModelElementClass {
144     readonly attribute AttributeLinkUList all_of_kind_attribute_link;
145     readonly attribute AttributeLinkUList all_of_type_attribute_link;
146     AttributeLink create_attribute_link (in ::UmlCore::Name name)
147         raises (Reflective::SemanticError);
148 };
149
150 interface AttributeLink : AttributeLinkClass, ::UmlCore::ModelElement {
151     ::UmlCore::UmlAttribute uml_attribute ()
152         raises (Reflective::SemanticError);
153     void set_uml_attribute (in ::UmlCore::UmlAttribute new_value)
154         raises (Reflective::SemanticError);
155     UmlInstance uml_value ()
156         raises (Reflective::SemanticError);
157     void set_uml_value (in UmlInstance new_value)
158         raises (Reflective::SemanticError);
159     UmlInstance owning_instance ()

```

```
160     raises (Reflective::SemanticError);
161 void set_owning_instance (in UmlInstance new_value)
162     raises (Reflective::SemanticError);
163 void add_owning_instance_before (in UmlInstance new_value,
164     in UmlInstance before)
165     raises (Reflective::StructuralError,
166     Reflective::NotFound,
167     Reflective::SemanticError);
168 };
169
170 interface UmlInstanceClass : ::UmlCore::ModelElementClass {
171     readonly attribute UmlInstanceUList all_of_kind_uuml_instance;
172     readonly attribute UmlInstanceUList all_of_type_uuml_instance;
173     UmlInstance create_uuml_instance (in ::UmlCore::Name name)
174     raises (Reflective::SemanticError);
175 };
176
177 interface UmlInstance : UmlInstanceClass, ::UmlCore::ModelElement {
178     ::UmlCore::ClassifierSet classifier ()
179     raises (Reflective::SemanticError);
180     void add_classifier (in ::UmlCore::ClassifierSet new_value)
181     raises (Reflective::StructuralError, Reflective::SemanticError);
182     void remove_classifier ()
183     raises (Reflective::SemanticError);
184     AttributeLinkSet slot ()
185     raises (Reflective::NotSet, Reflective::SemanticError);
186     void add_slot (in AttributeLinkSet new_value)
187     raises (Reflective::StructuralError, Reflective::SemanticError);
188     void remove_slot ()
189     raises (Reflective::SemanticError);
190     UmlCommonBehavior::LinkEndSet link_end ()
191     raises (Reflective::NotSet, Reflective::SemanticError);
192     void add_link_end (in UmlCommonBehavior::LinkEndSet new_value)
193     raises (Reflective::StructuralError, Reflective::SemanticError);
194     void remove_link_end ()
195     raises (Reflective::SemanticError);
196     MessageInstanceSet received_message_instance ()
197     raises (Reflective::NotSet, Reflective::SemanticError);
198     void add_received_message_instance (in MessageInstanceSet new_value)
199     raises (Reflective::StructuralError, Reflective::SemanticError);
200     void remove_received_message_instance ()
```



```

201     raises (Reflective::SemanticError);
202 AttributeLinkSet owned_attribute_link ()
203     raises (Reflective::NotSet, Reflective::SemanticError);
204 void add_owned_attribute_link (in AttributeLinkSet new_value)
205     raises (Reflective::StructuralError, Reflective::SemanticError);
206 void remove_owned_attribute_link ()
207     raises (Reflective::SemanticError);
208 MessageInstanceSet argument_owner ()
209     raises (Reflective::NotSet, Reflective::SemanticError);
210 void add_argument_owner (in MessageInstanceSet new_value)
211     raises (Reflective::StructuralError, Reflective::SemanticError);
212 void remove_argument_owner ()
213     raises (Reflective::SemanticError);
214 MessageInstanceSet sent_message_instance ()
215     raises (Reflective::NotSet, Reflective::SemanticError);
216 void add_sent_message_instance (in MessageInstanceSet new_value)
217     raises (Reflective::StructuralError, Reflective::SemanticError);
218 void remove_sent_message_instance ()
219     raises (Reflective::SemanticError);
220 };
221
222 interface UmlObjectClass : UmlInstanceClass {
223     readonly attribute UmlObjectUList all_of_kind_uml_object;
224     readonly attribute UmlObjectUList all_of_type_uml_object;
225     UmlObject create_uml_object (in ::UmlCore::Name name)
226     raises (Reflective::SemanticError);
227 };
228
229 interface UmlObject : UmlObjectClass, UmlInstance { };
230
231 interface MessageInstanceClass : ::UmlCore::ModelElementClass {
232     readonly attribute MessageInstanceUList all_of_kind_message_instance;
233     readonly attribute MessageInstanceUList all_of_type_message_instance;
234     MessageInstance create_message_instance (in ::UmlCore::Name name)
235     raises (Reflective::SemanticError);
236 };
237
238 interface MessageInstance : MessageInstanceClass, ::UmlCore::ModelElement {
239     ::UmlCore::Request specification ()
240     raises (Reflective::SemanticError);
241     void set_specification (in ::UmlCore::Request new_value)

```

```

242     raises (Reflective::SemanticError);
243 UmlInstance receiver ()
244     raises (Reflective::SemanticError);
245 void set_receiver (in UmlInstance new_value)
246     raises (Reflective::SemanticError);
247 UmlInstanceSet argument ()
248     raises (Reflective::NotSet, Reflective::SemanticError);
249 void add_argument (in UmlInstanceSet new_value)
250     raises (Reflective::StructuralError, Reflective::SemanticError);
251 void remove_argument ()
252     raises (Reflective::SemanticError);
253 UmlInstance sender ()
254     raises (Reflective::SemanticError);
255 void set_sender (in UmlInstance new_value)
256     raises (Reflective::SemanticError);
257 };
258
259 interface DataValueClass : UmlInstanceClass {
260     readonly attribute DataValueUList all_of_kind_data_value;
261     readonly attribute DataValueUList all_of_type_data_value;
262     DataValue create_data_value (in ::UmlCore::Name name)
263     raises (Reflective::SemanticError);
264 };
265
266 interface DataValue : DataValueClass, UmlInstance { };
267
268 interface SignalClass : ::UmlCore::GeneralizableElementClass,
269     ::UmlCore::RequestClass {
270     readonly attribute SignalUList all_of_kind_signal;
271     readonly attribute SignalUList all_of_type_signal;
272     Signal create_signal (in ::UmlCore::Name name,
273         in boolean is_root,
274         in boolean is_leaf,
275         in boolean is_abstract)
276     raises (Reflective::SemanticError);
277 };
278
279 interface Signal : SignalClass, ::UmlCore::GeneralizableElement,
280     ::UmlCore::Request {
281     UmlCommonBehavior::ReceptionSet reception ()
282     raises (Reflective::NotSet, Reflective::SemanticError);

```

```

283 void add_reception (in UmlCommonBehavior::ReceptionSet new_value)
284     raises (Reflective::StructuralError, Reflective::SemanticError);
285 void remove_reception ()
286     raises (Reflective::SemanticError);
287 ::UmlCore::ParameterUList parameter ()
288     raises (Reflective::NotSet, Reflective::SemanticError);
289 void add_parameter (in ::UmlCore::ParameterUList new_value)
290     raises (Reflective::StructuralError, Reflective::SemanticError);
291 void add_parameter_before (in ::UmlCore::Parameter new_value,
292     in ::UmlCore::Parameter before)
293     raises (Reflective::StructuralError,
294     Reflective::NotFound,
295     Reflective::SemanticError);
296 void remove_parameter ()
297     raises (Reflective::SemanticError);
298 };
299
300 interface UmlExceptionClass : SignalClass {
301     readonly attribute UmlExceptionUList all_of_kind_uml_exception;
302     readonly attribute UmlExceptionUList all_of_type_uml_exception;
303     UmlException create_uml_exception (in ::UmlCore::Name name,
304     in boolean is_root,
305     in boolean is_leaf,
306     in boolean is_abstract)
307     raises (Reflective::SemanticError);
308 };
309
310 interface UmlException : UmlExceptionClass, Signal {
311     ::UmlCore::BehavioralFeatureSet uml_context ()
312     raises (Reflective::NotSet, Reflective::SemanticError);
313     void add_uml_context (in ::UmlCore::BehavioralFeatureSet new_value)
314     raises (Reflective::StructuralError, Reflective::SemanticError);
315     void remove_uml_context ()
316     raises (Reflective::SemanticError);
317 };
318
319 interface ReceptionClass : ::UmlCore::BehavioralFeatureClass {
320     readonly attribute ReceptionUList all_of_kind_reception;
321     readonly attribute ReceptionUList all_of_type_reception;
322     Reception create_reception (in ::UmlCore::Name name,
323     in ::UmlCore::ScopeKind owner_scope,

```

```
324             in ::UmlCore::VisibilityKind visibility,
325             in boolean is_query,
326             in boolean is_polymorphic,
327             in ::UmlCore::Uninterpreted specification)
328     raises (Reflective::SemanticError);
329 };
330
331 interface Reception : ReceptionClass, ::UmlCore::BehavioralFeature {
332     boolean is_polymorphic ()
333     raises (Reflective::SemanticError);
334     void set_is_polymorphic (in boolean new_value)
335     raises (Reflective::SemanticError);
336     ::UmlCore::Uninterpreted specification ()
337     raises (Reflective::SemanticError);
338     void set_specification (in ::UmlCore::Uninterpreted new_value)
339     raises (Reflective::SemanticError);
340     UmlCommonBehavior::Signal signal ()
341     raises (Reflective::SemanticError);
342     void set_signal (in UmlCommonBehavior::Signal new_value)
343     raises (Reflective::SemanticError);
344 };
345
346 interface ArgumentClass : ::UmlCore::ModelElementClass {
347     readonly attribute ArgumentUList all_of_kind_argument;
348     readonly attribute ArgumentUList all_of_type_argument;
349     Argument create_argument (in ::UmlCore::Name name,
350                             in ::UmlCore::Expression uml_value)
351     raises (Reflective::SemanticError);
352 };
353
354 interface Argument : ArgumentClass, ::UmlCore::ModelElement {
355     ::UmlCore::Expression uml_value ()
356     raises (Reflective::SemanticError);
357     void set_uml_value (in ::UmlCore::Expression new_value)
358     raises (Reflective::SemanticError);
359     Action owning_action ()
360     raises (Reflective::NotSet, Reflective::SemanticError);
361     void set_owning_action (in Action new_value)
362     raises (Reflective::SemanticError);
363     void unset_owning_action ()
364     raises (Reflective::SemanticError);
```

```
365 };
366
367 interface ActionClass : ::UmlCore::ModelElementClass {
368     readonly attribute ActionUList all_of_kind_action;
369     readonly attribute ActionUList all_of_type_action;
370     Action create_action (in ::UmlCore::Name name,
371                          in ::UmlCore::Expression recurrence,
372                          in ::UmlCore::ObjectSetExpression target,
373                          in boolean is_asynchronous,
374                          in string script)
375     raises (Reflective::SemanticError);
376 };
377
378 interface Action : ActionClass, ::UmlCore::ModelElement {
379     ::UmlCore::Expression recurrence ()
380     raises (Reflective::SemanticError);
381     void set_recurrence (in ::UmlCore::Expression new_value)
382     raises (Reflective::SemanticError);
383     ::UmlCore::ObjectSetExpression target ()
384     raises (Reflective::SemanticError);
385     void set_target (in ::UmlCore::ObjectSetExpression new_value)
386     raises (Reflective::SemanticError);
387     boolean is_asynchronous ()
388     raises (Reflective::SemanticError);
389     void set_is_asynchronous (in boolean new_value)
390     raises (Reflective::SemanticError);
391     string script ()
392     raises (Reflective::SemanticError);
393     void set_script (in string new_value)
394     raises (Reflective::SemanticError);
395     ArgumentUList actual_argument ()
396     raises (Reflective::NotSet, Reflective::SemanticError);
397     void add_actual_argument (in ArgumentUList new_value)
398     raises (Reflective::StructuralError, Reflective::SemanticError);
399     void add_actual_argument_before (in Argument new_value,
400                                    in Argument before)
401     raises (Reflective::StructuralError,
402            Reflective::NotFound,
403            Reflective::SemanticError);
404     void remove_actual_argument ()
405     raises (Reflective::SemanticError);
```

```
406   ::UmlCore::Request message ()
407     raises (Reflective::NotSet, Reflective::SemanticError);
408 void set_message (in ::UmlCore::Request new_value)
409     raises (Reflective::SemanticError);
410 void unset_message ()
411     raises (Reflective::SemanticError);
412 UmlCommonBehavior::ActionSequence action_sequence ()
413     raises (Reflective::NotSet, Reflective::SemanticError);
414 void set_action_sequence (in UmlCommonBehavior::ActionSequence new_value)
415     raises (Reflective::SemanticError);
416 void unset_action_sequence ()
417     raises (Reflective::SemanticError);
418 };
419
420 interface CallActionClass : ActionClass {
421   readonly attribute CallActionUList all_of_kind_call_action;
422   readonly attribute CallActionUList all_of_type_call_action;
423   CallAction create_call_action (in ::UmlCore::Name name,
424                                 in ::UmlCore::Expression recurrence,
425                                 in ::UmlCore::ObjectSetExpression target,
426                                 in boolean is_asynchronous,
427                                 in string script,
428                                 in ::UmlCore::SynchronousKind mode)
429     raises (Reflective::SemanticError);
430 };
431
432 interface CallAction : CallActionClass, Action {
433   ::UmlCore::SynchronousKind mode ()
434     raises (Reflective::SemanticError);
435   void set_mode (in ::UmlCore::SynchronousKind new_value)
436     raises (Reflective::SemanticError);
437 };
438
439 interface CreateActionClass : ActionClass {
440   readonly attribute CreateActionUList all_of_kind_create_action;
441   readonly attribute CreateActionUList all_of_type_create_action;
442   CreateAction create_create_action (
443     in ::UmlCore::Name name,
444     in ::UmlCore::Expression recurrence,
445     in ::UmlCore::ObjectSetExpression target,
446     in boolean is_asynchronous,
```

```
447         in string script)
448     raises (Reflective::SemanticError);
449 };
450
451 interface CreateAction : CreateActionClass, Action {
452     ::UmlCore::Classifier instantiation ()
453     raises (Reflective::SemanticError);
454     void set_instantiation (in ::UmlCore::Classifier new_value)
455     raises (Reflective::SemanticError);
456 };
457
458 interface DestroyActionClass : ActionClass {
459     readonly attribute DestroyActionUList all_of_kind_destroy_action;
460     readonly attribute DestroyActionUList all_of_type_destroy_action;
461     DestroyAction create_destroy_action (
462         in ::UmlCore::Name name,
463         in ::UmlCore::Expression recurrence,
464         in ::UmlCore::ObjectSetExpression target,
465         in boolean is_asynchronous,
466         in string script)
467     raises (Reflective::SemanticError);
468 };
469
470 interface DestroyAction : DestroyActionClass, Action { };
471
472 interface LocalInvocationClass : ActionClass {
473     readonly attribute LocalInvocationUList all_of_kind_local_invocation;
474     readonly attribute LocalInvocationUList all_of_type_local_invocation;
475     LocalInvocation create_local_invocation (
476         in ::UmlCore::Name name,
477         in ::UmlCore::Expression recurrence,
478         in ::UmlCore::ObjectSetExpression target,
479         in boolean is_asynchronous,
480         in string script)
481     raises (Reflective::SemanticError);
482 };
483
484 interface LocalInvocation : LocalInvocationClass, Action { };
485
486 interface SendActionClass : ActionClass {
487     readonly attribute SendActionUList all_of_kind_send_action;
```

```
488     readonly attribute SendActionUList all_of_type_send_action;
489     SendAction create_send_action (in ::UmlCore::Name name,
490                                   in ::UmlCore::Expression recurrence,
491                                   in ::UmlCore::ObjectSetExpression target,
492                                   in boolean is_asynchronous,
493                                   in string script)
494     raises (Reflective::SemanticError);
495 };
496
497 interface SendAction : SendActionClass, Action { };
498
499 interface ReturnActionClass : ActionClass {
500     readonly attribute ReturnActionUList all_of_kind_return_action;
501     readonly attribute ReturnActionUList all_of_type_return_action;
502     ReturnAction create_return_action (
503         in ::UmlCore::Name name,
504         in ::UmlCore::Expression recurrence,
505         in ::UmlCore::ObjectSetExpression target,
506         in boolean is_asynchronous,
507         in string script)
508     raises (Reflective::SemanticError);
509 };
510
511 interface ReturnAction : ReturnActionClass, Action { };
512
513 interface TerminateActionClass : ActionClass {
514     readonly attribute TerminateActionUList all_of_kind_terminate_action;
515     readonly attribute TerminateActionUList all_of_type_terminate_action;
516     TerminateAction create_terminate_action (
517         in ::UmlCore::Name name,
518         in ::UmlCore::Expression recurrence,
519         in ::UmlCore::ObjectSetExpression target,
520         in boolean is_asynchronous,
521         in string script)
522     raises (Reflective::SemanticError);
523 };
524
525 interface TerminateAction : TerminateActionClass, Action { };
526
527 interface UninterpretedActionClass : ActionClass {
528     readonly attribute UninterpretedActionUList
```



```

529     all_of_kind_uninterpreted_action;
530     readonly attribute UninterpretedActionUList
531     all_of_type_uninterpreted_action;
532     UninterpretedAction create_uninterpreted_action (
533         in ::UmlCore::Name name,
534         in ::UmlCore::Expression recurrence,
535         in ::UmlCore::ObjectSetExpression target,
536         in boolean is_asynchronous,
537         in string script)
538     raises (Reflective::SemanticError);
539 };
540
541 interface UninterpretedAction : UninterpretedActionClass, Action { };
542
543 interface ActionSequenceClass : ::UmlCore::ModelElementClass {
544     readonly attribute ActionSequenceUList all_of_kind_action_sequence;
545     readonly attribute ActionSequenceUList all_of_type_action_sequence;
546     ActionSequence create_action_sequence (in ::UmlCore::Name name)
547     raises (Reflective::SemanticError);
548 };
549
550 interface ActionSequence : ActionSequenceClass, ::UmlCore::ModelElement {
551     UmlCommonBehavior::ActionSet action ()
552     raises (Reflective::NotSet, Reflective::SemanticError);
553     void add_action (in UmlCommonBehavior::ActionSet new_value)
554     raises (Reflective::StructuralError, Reflective::SemanticError);
555     void remove_action ()
556     raises (Reflective::SemanticError);
557 };
558
559 interface LinkObjectClass : UmlObjectClass, LinkClass {
560     readonly attribute LinkObjectUList all_of_kind_link_object;
561     readonly attribute LinkObjectUList all_of_type_link_object;
562     LinkObject create_link_object (in ::UmlCore::Name name)
563     raises (Reflective::SemanticError);
564 };
565
566 interface LinkObject : LinkObjectClass, UmlObject, Link { };
567
568 struct InstanceInstantiatesClassifierLink {
569     UmlInstance instantiated_instance;

```

```
570     ::UmlCore::Classifier classifier;
571 };
572 typedef sequence <InstanceInstantiatesClassifierLink>
573     InstanceInstantiatesClassifierLinkSet;
574
575 interface InstanceInstantiatesClassifier : Reflective::RefAssociation {
576     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
577     InstanceInstantiatesClassifierLinkSet
578     all_instance_instantiates_classifier_links();
579     boolean exists (in UmlInstance instantiated_instance,
580                   in ::UmlCore::Classifier classifier);
581     UmlInstanceSet with_classifier (in ::UmlCore::Classifier classifier);
582     ::UmlCore::ClassifierSet with_instantiated_instance (
583         in UmlInstance instantiated_instance);
584     void add (in UmlInstance instantiated_instance,
585             in ::UmlCore::Classifier classifier)
586         raises (Reflective::StructuralError, Reflective::SemanticError);
587     void modify_instantiated_instance (
588         in UmlInstance instantiated_instance,
589         in ::UmlCore::Classifier classifier,
590         in UmlInstance new_instantiated_instance)
591         raises (Reflective::StructuralError,
592               Reflective::SemanticError,
593               Reflective::NotFound);
594     void modify_classifier (in UmlInstance instantiated_instance,
595                            in ::UmlCore::Classifier classifier,
596                            in ::UmlCore::Classifier new_classifier)
597         raises (Reflective::StructuralError,
598               Reflective::SemanticError,
599               Reflective::NotFound);
600     void remove (in UmlInstance instantiated_instance,
601                in ::UmlCore::Classifier classifier)
602         raises (Reflective::StructuralError,
603               Reflective::SemanticError,
604               Reflective::NotFound);
605 };
606
607 struct ActionOwnsArgumentLink {
608     Argument actual_argument;
609     Action owning_action;
610 };
```

```

611 typedef sequence <ActionOwnsArgumentLink> ActionOwnsArgumentLinkSet;
612
613 interface ActionOwnsArgument : Reflective::RefAssociation {
614     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
615     ActionOwnsArgumentLinkSet all_action_owns_argument_links();
616     boolean exists (in Argument actual_argument, in Action owning_action);
617     ArgumentUList with_owning_action (in Action owning_action);
618     Action with_actual_argument (in Argument actual_argument);
619     void add (in Argument actual_argument, in Action owning_action)
620         raises (Reflective::StructuralError, Reflective::SemanticError);
621     void add_before_actual_argument (in Argument actual_argument,
622                                     in Action owning_action,
623                                     in Argument before)
624         raises (Reflective::StructuralError,
625               Reflective::SemanticError,
626               Reflective::NotFound);
627     void modify_actual_argument (in Argument actual_argument,
628                                 in Action owning_action,
629                                 in Argument new_actual_argument)
630         raises (Reflective::StructuralError,
631               Reflective::SemanticError,
632               Reflective::NotFound);
633     void modify_owning_action (in Argument actual_argument,
634                               in Action owning_action,
635                               in Action new_owning_action)
636         raises (Reflective::StructuralError,
637               Reflective::SemanticError,
638               Reflective::NotFound);
639     void remove (in Argument actual_argument, in Action owning_action)
640         raises (Reflective::StructuralError,
641               Reflective::SemanticError,
642               Reflective::NotFound);
643 };
644
645 struct CreateActionInstantiatesClassifierLink {
646     UmlCommonBehavior::CreateAction create_action;
647     ::UmlCore::Classifier instantiation;
648 };
649 typedef sequence <CreateActionInstantiatesClassifierLink>
650     CreateActionInstantiatesClassifierLinkSet;
651

```

```

652 interface CreateActionInstantiatesClassifier : Reflective::RefAssociation {
653     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
654     CreateActionInstantiatesClassifierLinkSet
655     all_create_action_instantiates_classifier_links();
656     boolean exists (in UmlCommonBehavior::CreateAction create_action,
657                   in ::UmlCore::Classifier instantiation);
658     UmlCommonBehavior::CreateActionSet with_instantiation (
659         in ::UmlCore::Classifier instantiation);
660     ::UmlCore::Classifier with_create_action (
661         in UmlCommonBehavior::CreateAction create_action);
662     void add (in UmlCommonBehavior::CreateAction create_action,
663             in ::UmlCore::Classifier instantiation)
664         raises (Reflective::StructuralError, Reflective::SemanticError);
665     void modify_create_action (
666         in UmlCommonBehavior::CreateAction create_action,
667         in ::UmlCore::Classifier instantiation,
668         in UmlCommonBehavior::CreateAction new_create_action)
669         raises (Reflective::StructuralError,
670               Reflective::SemanticError,
671               Reflective::NotFound);
672     void modify_instantiation (
673         in UmlCommonBehavior::CreateAction create_action,
674         in ::UmlCore::Classifier instantiation,
675         in ::UmlCore::Classifier new_instantiation)
676         raises (Reflective::StructuralError,
677               Reflective::SemanticError,
678               Reflective::NotFound);
679     void remove (in UmlCommonBehavior::CreateAction create_action,
680                in ::UmlCore::Classifier instantiation)
681         raises (Reflective::StructuralError,
682               Reflective::SemanticError,
683               Reflective::NotFound);
684 };
685
686 struct AttributeLinkIsInstanceOfAttributeLink {
687     AttributeLink instance;
688     ::UmlCore::UmlAttribute uml_attribute;
689 };
690 typedef sequence <AttributeLinkIsInstanceOfAttributeLink>
691     AttributeLinkIsInstanceOfAttributeLinkSet;
692

```

```

693 interface AttributeLinkIsInstanceOfAttribute : Reflective::RefAssociation {
694     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
695     AttributeLinkIsInstanceOfAttributeLinkSet
696     all_attribute_link_is_instance_of_attribute_links();
697     boolean exists (in AttributeLink instance,
698                   in ::UmlCore::UmlAttribute uml_attribute);
699     AttributeLinkSet with_uml_attribute (in ::UmlCore::UmlAttribute
700                                         uml_attribute);
701     ::UmlCore::UmlAttribute with_instance (in AttributeLink instance);
702     void add (in AttributeLink instance,
703             in ::UmlCore::UmlAttribute uml_attribute)
704     raises (Reflective::StructuralError, Reflective::SemanticError);
705     void modify_instance (in AttributeLink instance,
706                          in ::UmlCore::UmlAttribute uml_attribute,
707                          in AttributeLink new_instance)
708     raises (Reflective::StructuralError,
709           Reflective::SemanticError,
710           Reflective::NotFound);
711     void modify_uml_attribute (in AttributeLink instance,
712                               in ::UmlCore::UmlAttribute uml_attribute,
713                               in ::UmlCore::UmlAttribute new_uml_attribute)
714     raises (Reflective::StructuralError,
715           Reflective::SemanticError,
716           Reflective::NotFound);
717     void remove (in AttributeLink instance,
718                in ::UmlCore::UmlAttribute uml_attribute)
719     raises (Reflective::StructuralError,
720           Reflective::SemanticError,
721           Reflective::NotFound);
722 };
723
724 struct AttributeLinkHasValueOfLinkLink {
725     AttributeLink slot;
726     UmlInstance uml_value;
727 };
728 typedef sequence <AttributeLinkHasValueOfLinkLink>
729     AttributeLinkHasValueOfLinkLinkSet;
730
731 interface AttributeLinkHasValueOfLink : Reflective::RefAssociation {
732     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
733     AttributeLinkHasValueOfLinkLinkSet

```

```

734     all_attribute_link_has_value_of_link_links();
735     boolean exists (in AttributeLink slot, in UmlInstance uml_value);
736     AttributeLinkSet with_uml_value (in UmlInstance uml_value);
737     UmlInstance with_slot (in AttributeLink slot);
738     void add (in AttributeLink slot, in UmlInstance uml_value)
739         raises (Reflective::StructuralError, Reflective::SemanticError);
740     void modify_slot (in AttributeLink slot,
741                     in UmlInstance uml_value,
742                     in AttributeLink new_slot)
743         raises (Reflective::StructuralError,
744               Reflective::SemanticError,
745               Reflective::NotFound);
746     void modify_uml_value (in AttributeLink slot,
747                           in UmlInstance uml_value,
748                           in UmlInstance new_uml_value)
749         raises (Reflective::StructuralError,
750               Reflective::SemanticError,
751               Reflective::NotFound);
752     void remove (in AttributeLink slot, in UmlInstance uml_value)
753         raises (Reflective::StructuralError,
754               Reflective::SemanticError,
755               Reflective::NotFound);
756 };
757
758 struct LinkEndIsOfTypeInstanceLink {
759     UmlCommonBehavior::UmlInstance uml_instance;
760     UmlCommonBehavior::LinkEnd link_end;
761 };
762 typedef sequence <LinkEndIsOfTypeInstanceLink> LinkEndIsOfTypeInstanceLinkSet;
763
764 interface LinkEndIsOfTypeInstance : Reflective::RefAssociation {
765     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
766     LinkEndIsOfTypeInstanceLinkSet all_link_end_is_of_type_instance_links();
767     boolean exists (in UmlCommonBehavior::UmlInstance uml_instance,
768                   in UmlCommonBehavior::LinkEnd link_end);
769     UmlCommonBehavior::UmlInstance with_link_end (
770         in UmlCommonBehavior::LinkEnd link_end);
771     UmlCommonBehavior::LinkEndSet with_uml_instance (
772         in UmlCommonBehavior::UmlInstance uml_instance);
773     void add (in UmlCommonBehavior::UmlInstance uml_instance,
774              in UmlCommonBehavior::LinkEnd link_end)

```

```

775     raises (Reflective::StructuralError, Reflective::SemanticError);
776 void modify_uml_instance (
777     in UmlCommonBehavior::UmlInstance uml_instance,
778     in UmlCommonBehavior::LinkEnd link_end,
779     in UmlCommonBehavior::UmlInstance new_uml_instance)
780     raises (Reflective::StructuralError,
781         Reflective::SemanticError,
782         Reflective::NotFound);
783 void modify_link_end (in UmlCommonBehavior::UmlInstance uml_instance,
784     in UmlCommonBehavior::LinkEnd link_end,
785     in UmlCommonBehavior::LinkEnd new_link_end)
786     raises (Reflective::StructuralError,
787         Reflective::SemanticError,
788         Reflective::NotFound);
789 void remove (in UmlCommonBehavior::UmlInstance uml_instance,
790     in UmlCommonBehavior::LinkEnd link_end)
791     raises (Reflective::StructuralError,
792         Reflective::SemanticError,
793         Reflective::NotFound);
794 };
795
796 struct ReceptionFeatureReceivesSignalLink {
797     UmlCommonBehavior::Signal signal;
798     UmlCommonBehavior::Reception reception;
799 };
800 typedef sequence <ReceptionFeatureReceivesSignalLink>
801     ReceptionFeatureReceivesSignalLinkSet;
802
803 interface ReceptionFeatureReceivesSignal : Reflective::RefAssociation {
804     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
805     ReceptionFeatureReceivesSignalLinkSet
806     all_reception_feature_receives_signal_links();
807     boolean exists (in UmlCommonBehavior::Signal signal,
808         in UmlCommonBehavior::Reception reception);
809     UmlCommonBehavior::Signal with_reception (
810         in UmlCommonBehavior::Reception reception);
811     UmlCommonBehavior::ReceptionSet with_signal (
812         in UmlCommonBehavior::Signal signal);
813     void add (in UmlCommonBehavior::Signal signal,
814         in UmlCommonBehavior::Reception reception)
815         raises (Reflective::StructuralError, Reflective::SemanticError);

```

```
816 void modify_signal (in UmlCommonBehavior::Signal signal,
817                     in UmlCommonBehavior::Reception reception,
818                     in UmlCommonBehavior::Signal new_signal)
819     raises (Reflective::StructuralError,
820            Reflective::SemanticError,
821            Reflective::NotFound);
822 void modify_reception (in UmlCommonBehavior::Signal signal,
823                       in UmlCommonBehavior::Reception reception,
824                       in UmlCommonBehavior::Reception new_reception)
825     raises (Reflective::StructuralError,
826            Reflective::SemanticError,
827            Reflective::NotFound);
828 void remove (in UmlCommonBehavior::Signal signal,
829             in UmlCommonBehavior::Reception reception)
830     raises (Reflective::StructuralError,
831            Reflective::SemanticError,
832            Reflective::NotFound);
833 };
834
835 struct SignalOwnsParameterLink {
836     UmlCommonBehavior::Signal signal;
837     ::UmlCore::Parameter parameter;
838 };
839 typedef sequence <SignalOwnsParameterLink> SignalOwnsParameterLinkSet;
840
841 interface SignalOwnsParameter : Reflective::RefAssociation {
842     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
843     SignalOwnsParameterLinkSet all_signal_owns_parameter_links();
844     boolean exists (in UmlCommonBehavior::Signal signal,
845                   in ::UmlCore::Parameter parameter);
846     UmlCommonBehavior::Signal with_parameter (
847         in ::UmlCore::Parameter parameter);
848     ::UmlCore::ParameterUList with_signal (
849         in UmlCommonBehavior::Signal signal);
850     void add (in UmlCommonBehavior::Signal signal,
851             in ::UmlCore::Parameter parameter)
852         raises (Reflective::StructuralError, Reflective::SemanticError);
853     void add_before_parameter (in UmlCommonBehavior::Signal signal,
854                               in ::UmlCore::Parameter parameter,
855                               in ::UmlCore::Parameter before)
856         raises (Reflective::StructuralError,
```



```

857         Reflective::SemanticError,
858         Reflective::NotFound);
859 void modify_signal (in UmlCommonBehavior::Signal signal,
860                   in ::UmlCore::Parameter parameter,
861                   in UmlCommonBehavior::Signal new_signal)
862     raises (Reflective::StructuralError,
863           Reflective::SemanticError,
864           Reflective::NotFound);
865 void modify_parameter (in UmlCommonBehavior::Signal signal,
866                      in ::UmlCore::Parameter parameter,
867                      in ::UmlCore::Parameter new_parameter)
868     raises (Reflective::StructuralError,
869           Reflective::SemanticError,
870           Reflective::NotFound);
871 void remove (in UmlCommonBehavior::Signal signal,
872            in ::UmlCore::Parameter parameter)
873     raises (Reflective::StructuralError,
874           Reflective::SemanticError,
875           Reflective::NotFound);
876 };
877
878 struct ActionIsInitiatedByRequestLink {
879     ::UmlCore::Request message;
880     UmlCommonBehavior::Action action;
881 };
882 typedef sequence <ActionIsInitiatedByRequestLink>
883     ActionIsInitiatedByRequestLinkSet;
884
885 interface ActionIsInitiatedByRequest : Reflective::RefAssociation {
886     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
887     ActionIsInitiatedByRequestLinkSet
888         all_action_is_initiated_by_request_links();
889     boolean exists (in ::UmlCore::Request message,
890                  in UmlCommonBehavior::Action action);
891     ::UmlCore::Request with_action (in UmlCommonBehavior::Action action);
892     UmlCommonBehavior::ActionSet with_message (in ::UmlCore::Request message);
893     void add (in ::UmlCore::Request message,
894             in UmlCommonBehavior::Action action)
895         raises (Reflective::StructuralError, Reflective::SemanticError);
896     void modify_message (in ::UmlCore::Request message,
897                        in UmlCommonBehavior::Action action,

```

```

898         in ::UmlCore::Request new_message)
899     raises (Reflective::StructuralError,
900           Reflective::SemanticError,
901           Reflective::NotFound);
902 void modify_action (in ::UmlCore::Request message,
903                   in UmlCommonBehavior::Action action,
904                   in UmlCommonBehavior::Action new_action)
905     raises (Reflective::StructuralError,
906           Reflective::SemanticError,
907           Reflective::NotFound);
908 void remove (in ::UmlCore::Request message,
909             in UmlCommonBehavior::Action action)
910     raises (Reflective::StructuralError,
911           Reflective::SemanticError,
912           Reflective::NotFound);
913 };
914
915 struct MessageInstanceIsSpecifiedByRequestLink {
916     MessageInstance instance;
917     ::UmlCore::Request specification;
918 };
919 typedef sequence <MessageInstanceIsSpecifiedByRequestLink>
920     MessageInstanceIsSpecifiedByRequestLinkSet;
921
922 interface MessageInstanceIsSpecifiedByRequest : Reflective::RefAssociation {
923     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
924     MessageInstanceIsSpecifiedByRequestLinkSet
925     all_message_instance_is_specified_by_request_links();
926     boolean exists (in MessageInstance instance,
927                   in ::UmlCore::Request specification);
928     MessageInstanceSet with_specification (
929         in ::UmlCore::Request specification);
930     ::UmlCore::Request with_instance (in MessageInstance instance);
931     void add (in MessageInstance instance, in ::UmlCore::Request specification)
932         raises (Reflective::StructuralError, Reflective::SemanticError);
933     void modify_instance (in MessageInstance instance,
934                           in ::UmlCore::Request specification,
935                           in MessageInstance new_instance)
936         raises (Reflective::StructuralError,
937               Reflective::SemanticError,
938               Reflective::NotFound);

```

```

939 void modify_specification (in MessageInstance instance,
940                             in ::UmlCore::Request specification,
941                             in ::UmlCore::Request new_specification)
942     raises (Reflective::StructuralError,
943            Reflective::SemanticError,
944            Reflective::NotFound);
945 void remove (in MessageInstance instance,
946              in ::UmlCore::Request specification)
947     raises (Reflective::StructuralError,
948            Reflective::SemanticError,
949            Reflective::NotFound);
950 };
951
952 struct InstanceReceivesMessageInstanceLink {
953     UmlInstance receiver;
954     MessageInstance received_message_instance;
955 };
956 typedef sequence <InstanceReceivesMessageInstanceLink>
957     InstanceReceivesMessageInstanceLinkSet;
958
959 interface InstanceReceivesMessageInstance : Reflective::RefAssociation {
960     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
961     InstanceReceivesMessageInstanceLinkSet
962         all_instance_receives_message_instance_links();
963     boolean exists (in UmlInstance receiver,
964                   in MessageInstance received_message_instance);
965     UmlInstance with_received_message_instance (
966         in MessageInstance received_message_instance);
967     MessageInstanceSet with_receiver (in UmlInstance receiver);
968     void add (in UmlInstance receiver,
969             in MessageInstance received_message_instance)
970         raises (Reflective::StructuralError, Reflective::SemanticError);
971     void modify_receiver (in UmlInstance receiver,
972                          in MessageInstance received_message_instance,
973                          in UmlInstance new_receiver)
974         raises (Reflective::StructuralError,
975                Reflective::SemanticError,
976                Reflective::NotFound);
977     void modify_received_message_instance (
978         in UmlInstance receiver,
979         in MessageInstance received_message_instance,

```

```
980         in MessageInstance new_received_message_instance)
981     raises (Reflective::StructuralError,
982           Reflective::SemanticError,
983           Reflective::NotFound);
984 void remove (in UmlInstance receiver,
985             in MessageInstance received_message_instance)
986     raises (Reflective::StructuralError,
987           Reflective::SemanticError,
988           Reflective::NotFound);
989 };
990
991 struct InstanceOwnsAttributeLinkLink {
992     AttributeLink owned_attribute_link;
993     UmlInstance owning_instance;
994 };
995 typedef sequence <InstanceOwnsAttributeLinkLink>
996     InstanceOwnsAttributeLinkLinkSet;
997
998 interface InstanceOwnsAttributeLink : Reflective::RefAssociation {
999     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
1000     InstanceOwnsAttributeLinkLinkSet all_instance_owns_attribute_link_links();
1001     boolean exists (in AttributeLink owned_attribute_link,
1002                   in UmlInstance owning_instance);
1003     AttributeLinkSet with_owning_instance (in UmlInstance owning_instance);
1004     UmlInstance with_owned_attribute_link (
1005         in AttributeLink owned_attribute_link);
1006     void add (in AttributeLink owned_attribute_link,
1007             in UmlInstance owning_instance)
1008         raises (Reflective::StructuralError, Reflective::SemanticError);
1009     void add_before_owning_instance (in AttributeLink owned_attribute_link,
1010                                     in UmlInstance owning_instance,
1011                                     in UmlInstance before)
1012         raises (Reflective::StructuralError,
1013               Reflective::SemanticError,
1014               Reflective::NotFound);
1015     void modify_owned_attribute_link (
1016         in AttributeLink owned_attribute_link,
1017         in UmlInstance owning_instance,
1018         in AttributeLink new_owned_attribute_link)
1019         raises (Reflective::StructuralError,
1020               Reflective::SemanticError,
```

```

1021         Reflective::NotFound);
1022 void modify_owning_instance (in AttributeLink owned_attribute_link,
1023                             in UmlInstance owning_instance,
1024                             in UmlInstance new_owning_instance)
1025     raises (Reflective::StructuralError,
1026           Reflective::SemanticError,
1027           Reflective::NotFound);
1028 void remove (in AttributeLink owned_attribute_link,
1029             in UmlInstance owning_instance)
1030     raises (Reflective::StructuralError,
1031           Reflective::SemanticError,
1032           Reflective::NotFound);
1033 };
1034
1035 struct MessageInstanceHasArgumentOfInstanceLink {
1036     UmlInstance argument;
1037     MessageInstance argument_owner;
1038 };
1039 typedef sequence <MessageInstanceHasArgumentOfInstanceLink>
1040     MessageInstanceHasArgumentOfInstanceLinkSet;
1041
1042 interface MessageInstanceHasArgumentOfInstance : Reflective::RefAssociation {
1043     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
1044     MessageInstanceHasArgumentOfInstanceLinkSet
1045     all_message_instance_has_argument_of_instance_links();
1046     boolean exists (in UmlInstance argument,
1047                   in MessageInstance argument_owner);
1048     UmlInstanceSet with_argument_owner (in MessageInstance argument_owner);
1049     MessageInstanceSet with_argument (in UmlInstance argument);
1050     void add (in UmlInstance argument, in MessageInstance argument_owner)
1051         raises (Reflective::StructuralError, Reflective::SemanticError);
1052     void modify_argument (in UmlInstance argument,
1053                           in MessageInstance argument_owner,
1054                           in UmlInstance new_argument)
1055         raises (Reflective::StructuralError,
1056               Reflective::SemanticError,
1057               Reflective::NotFound);
1058     void modify_argument_owner (in UmlInstance argument,
1059                                 in MessageInstance argument_owner,
1060                                 in MessageInstance new_argument_owner)
1061         raises (Reflective::StructuralError,

```

```

1062         Reflective::SemanticError,
1063         Reflective::NotFound);
1064 void remove (in UmlInstance argument, in MessageInstance argument_owner)
1065     raises (Reflective::StructuralError,
1066           Reflective::SemanticError,
1067           Reflective::NotFound);
1068 };
1069
1070 struct ExceptionIsRaisedByBehavioralFeatureLink {
1071     ::UmlCore::BehavioralFeature uml_context;
1072     UmlException raised_exception;
1073 };
1074 typedef sequence <ExceptionIsRaisedByBehavioralFeatureLink>
1075     ExceptionIsRaisedByBehavioralFeatureLinkSet;
1076
1077 interface ExceptionIsRaisedByBehavioralFeature : Reflective::RefAssociation {
1078     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
1079     ExceptionIsRaisedByBehavioralFeatureLinkSet
1080     all_exception_is_raised_by_behavioral_feature_links();
1081     boolean exists (in ::UmlCore::BehavioralFeature uml_context,
1082                  in UmlException raised_exception);
1083     ::UmlCore::BehavioralFeatureSet with_raised_exception (
1084         in UmlException raised_exception);
1085     UmlExceptionSet with_uml_context (
1086         in ::UmlCore::BehavioralFeature uml_context);
1087     void add (in ::UmlCore::BehavioralFeature uml_context,
1088             in UmlException raised_exception)
1089         raises (Reflective::StructuralError, Reflective::SemanticError);
1090     void modify_uml_context (in ::UmlCore::BehavioralFeature uml_context,
1091                             in UmlException raised_exception,
1092                             in ::UmlCore::BehavioralFeature new_uml_context)
1093         raises (Reflective::StructuralError,
1094               Reflective::SemanticError,
1095               Reflective::NotFound);
1096     void modify_raised_exception (in ::UmlCore::BehavioralFeature uml_context,
1097                                  in UmlException raised_exception,
1098                                  in UmlException new_raised_exception)
1099         raises (Reflective::StructuralError,
1100               Reflective::SemanticError,
1101               Reflective::NotFound);
1102     void remove (in ::UmlCore::BehavioralFeature uml_context,

```

```

1103         in UmlException raised_exception)
1104     raises (Reflective::StructuralError,
1105           Reflective::SemanticError,
1106           Reflective::NotFound);
1107 };
1108
1109 struct LinkInstantiatesAssociationLink {
1110     ::UmlCore::Association association;
1111     Link instance;
1112 };
1113 typedef sequence <LinkInstantiatesAssociationLink>
1114     LinkInstantiatesAssociationLinkSet;
1115
1116 interface LinkInstantiatesAssociation : Reflective::RefAssociation {
1117     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
1118     LinkInstantiatesAssociationLinkSet
1119     all_link_instantiates_association_links();
1120     boolean exists (in ::UmlCore::Association association, in Link instance);
1121     ::UmlCore::Association with_instance (in Link instance);
1122     LinkSet with_association (in ::UmlCore::Association association);
1123     void add (in ::UmlCore::Association association, in Link instance)
1124     raises (Reflective::StructuralError, Reflective::SemanticError);
1125     void modify_association (in ::UmlCore::Association association,
1126                             in Link instance,
1127                             in ::UmlCore::Association new_association)
1128     raises (Reflective::StructuralError,
1129           Reflective::SemanticError,
1130           Reflective::NotFound);
1131     void modify_instance (in ::UmlCore::Association association,
1132                          in Link instance,
1133                          in Link new_instance)
1134     raises (Reflective::StructuralError,
1135           Reflective::SemanticError,
1136           Reflective::NotFound);
1137     void remove (in ::UmlCore::Association association, in Link instance)
1138     raises (Reflective::StructuralError,
1139           Reflective::SemanticError,
1140           Reflective::NotFound);
1141 };
1142
1143 struct LinkOwnsLinkEndLink {

```

```
1144   Link owning_link;
1145   LinkEnd link_role;
1146 };
1147 typedef sequence <LinkOwnsLinkEndLink> LinkOwnsLinkEndLinkSet;
1148
1149 interface LinkOwnsLinkEnd : Reflective::RefAssociation {
1150   readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
1151   LinkOwnsLinkEndLinkSet all_link_owns_link_end_links();
1152   boolean exists (in Link owning_link, in LinkEnd link_role);
1153   Link with_link_role (in LinkEnd link_role);
1154   LinkEndSet with_owning_link (in Link owning_link);
1155   void add (in Link owning_link, in LinkEnd link_role)
1156     raises (Reflective::StructuralError, Reflective::SemanticError);
1157   void add_before_owning_link (in Link owning_link,
1158                               in LinkEnd link_role,
1159                               in Link before)
1160     raises (Reflective::StructuralError,
1161            Reflective::SemanticError,
1162            Reflective::NotFound);
1163   void modify_owning_link (in Link owning_link,
1164                           in LinkEnd link_role,
1165                           in Link new_owning_link)
1166     raises (Reflective::StructuralError,
1167            Reflective::SemanticError,
1168            Reflective::NotFound);
1169   void modify_link_role (in Link owning_link,
1170                        in LinkEnd link_role,
1171                        in LinkEnd new_link_role)
1172     raises (Reflective::StructuralError,
1173            Reflective::SemanticError,
1174            Reflective::NotFound);
1175   void remove (in Link owning_link, in LinkEnd link_role)
1176     raises (Reflective::StructuralError,
1177            Reflective::SemanticError,
1178            Reflective::NotFound);
1179 };
1180
1181 struct LinkEndInstantiatesAssociationEndLink {
1182   ::UmlCore::AssociationEnd association_end;
1183   LinkEnd instance;
1184 };
```



```

1185 typedef sequence <LinkEndInstantiatesAssociationEndLink>
1186   LinkEndInstantiatesAssociationEndLinkSet;
1187
1188 interface LinkEndInstantiatesAssociationEnd : Reflective::RefAssociation {
1189   readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
1190   LinkEndInstantiatesAssociationEndLinkSet
1191     all_link_end_instantiates_association_end_links();
1192   boolean exists (in ::UmlCore::AssociationEnd association_end,
1193                 in LinkEnd instance);
1194   ::UmlCore::AssociationEnd with_instance (in LinkEnd instance);
1195   LinkEndSet with_association_end (
1196     in ::UmlCore::AssociationEnd association_end);
1197   void add (in ::UmlCore::AssociationEnd association_end,
1198            in LinkEnd instance)
1199     raises (Reflective::StructuralError, Reflective::SemanticError);
1200   void modify_association_end (
1201     in ::UmlCore::AssociationEnd association_end,
1202     in LinkEnd instance,
1203     in ::UmlCore::AssociationEnd new_association_end)
1204     raises (Reflective::StructuralError,
1205            Reflective::SemanticError,
1206            Reflective::NotFound);
1207   void modify_instance (in ::UmlCore::AssociationEnd association_end,
1208                        in LinkEnd instance,
1209                        in LinkEnd new_instance)
1210     raises (Reflective::StructuralError,
1211            Reflective::SemanticError,
1212            Reflective::NotFound);
1213   void remove (in ::UmlCore::AssociationEnd association_end,
1214              in LinkEnd instance)
1215     raises (Reflective::StructuralError,
1216            Reflective::SemanticError,
1217            Reflective::NotFound);
1218 };
1219
1220 struct InstanceSendsMessageInstanceLink {
1221   UmlInstance sender;
1222   MessageInstance sent_message_instance;
1223 };
1224 typedef sequence <InstanceSendsMessageInstanceLink>
1225   InstanceSendsMessageInstanceLinkSet;

```

```
1226
1227 interface InstanceSendsMessageInstance : Reflective::RefAssociation {
1228     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
1229     InstanceSendsMessageInstanceLinkSet
1230     all_instance_sends_message_instance_links();
1231     boolean exists (in UmlInstance sender,
1232                   in MessageInstance sent_message_instance);
1233     UmlInstance with_sent_message_instance (
1234                   in MessageInstance sent_message_instance);
1235     MessageInstanceSet with_sender (in UmlInstance sender);
1236     void add (in UmlInstance sender, in MessageInstance sent_message_instance)
1237     raises (Reflective::StructuralError, Reflective::SemanticError);
1238     void modify_sender (in UmlInstance sender,
1239                       in MessageInstance sent_message_instance,
1240                       in UmlInstance new_sender)
1241     raises (Reflective::StructuralError,
1242           Reflective::SemanticError,
1243           Reflective::NotFound);
1244     void modify_sent_message_instance (
1245                   in UmlInstance sender,
1246                   in MessageInstance sent_message_instance,
1247                   in MessageInstance new_sent_message_instance)
1248     raises (Reflective::StructuralError,
1249           Reflective::SemanticError,
1250           Reflective::NotFound);
1251     void remove (in UmlInstance sender,
1252                in MessageInstance sent_message_instance)
1253     raises (Reflective::StructuralError,
1254           Reflective::SemanticError,
1255           Reflective::NotFound);
1256 };
1257
1258 struct ActionSequenceOwnsActionLink {
1259     UmlCommonBehavior::ActionSequence action_sequence;
1260     UmlCommonBehavior::Action action;
1261 };
1262 typedef sequence <ActionSequenceOwnsActionLink>
1263     ActionSequenceOwnsActionLinkSet;
1264
1265 interface ActionSequenceOwnsAction : Reflective::RefAssociation {
1266     readonly attribute UmlCommonBehaviorPackage enclosing_package_ref;
```

```

1267   ActionSequenceOwnsActionLinkSet all_action_sequence_owns_action_links();
1268   boolean exists (in UmlCommonBehavior::ActionSequence action_sequence,
1269                 in UmlCommonBehavior::Action action);
1270   UmlCommonBehavior::ActionSequence with_action (
1271                 in UmlCommonBehavior::Action action);
1272   UmlCommonBehavior::ActionSet with_action_sequence (
1273                 in UmlCommonBehavior::ActionSequence action_sequence);
1274   void add (in UmlCommonBehavior::ActionSequence action_sequence,
1275            in UmlCommonBehavior::Action action)
1276     raises (Reflective::StructuralError, Reflective::SemanticError);
1277   void modify_action_sequence (
1278                 in UmlCommonBehavior::ActionSequence action_sequence,
1279                 in UmlCommonBehavior::Action action,
1280                 in UmlCommonBehavior::ActionSequence new_action_sequence)
1281     raises (Reflective::StructuralError,
1282            Reflective::SemanticError,
1283            Reflective::NotFound);
1284   void modify_action (in UmlCommonBehavior::ActionSequence action_sequence,
1285                     in UmlCommonBehavior::Action action,
1286                     in UmlCommonBehavior::Action new_action)
1287     raises (Reflective::StructuralError,
1288            Reflective::SemanticError,
1289            Reflective::NotFound);
1290   void remove (in UmlCommonBehavior::ActionSequence action_sequence,
1291               in UmlCommonBehavior::Action action)
1292     raises (Reflective::StructuralError,
1293            Reflective::SemanticError,
1294            Reflective::NotFound);
1295 };
1296
1297 interface UmlCommonBehaviorPackageFactory {
1298   UmlCommonBehaviorPackage create_uml_common_behavior_package ()
1299     raises (Reflective::SemanticError);
1300 };
1301
1302 interface UmlCommonBehaviorPackage : Reflective::RefPackage {
1303   readonly attribute CallClass call_class_ref;
1304   readonly attribute LinkEndClass link_end_class_ref;
1305   readonly attribute LinkClass link_class_ref;
1306   readonly attribute AttributeLinkClass attribute_link_class_ref;
1307   readonly attribute UmlInstanceClass uml_instance_class_ref;

```

---

1308   readonly attribute UmlObjectClass uml\_object\_class\_ref;  
1309   readonly attribute MessageInstanceClass message\_instance\_class\_ref;  
1310   readonly attribute DataValueClass data\_value\_class\_ref;  
1311   readonly attribute SignalClass signal\_class\_ref;  
1312   readonly attribute UmlExceptionClass uml\_exception\_class\_ref;  
1313   readonly attribute ReceptionClass reception\_class\_ref;  
1314   readonly attribute ArgumentClass argument\_class\_ref;  
1315   readonly attribute ActionClass action\_class\_ref;  
1316   readonly attribute CallActionClass call\_action\_class\_ref;  
1317   readonly attribute CreateActionClass create\_action\_class\_ref;  
1318   readonly attribute DestroyActionClass destroy\_action\_class\_ref;  
1319   readonly attribute LocalInvocationClass local\_invocation\_class\_ref;  
1320   readonly attribute SendActionClass send\_action\_class\_ref;  
1321   readonly attribute ReturnActionClass return\_action\_class\_ref;  
1322   readonly attribute TerminateActionClass terminate\_action\_class\_ref;  
1323   readonly attribute UninterpretedActionClass uninterpreted\_action\_class\_ref;  
1324   readonly attribute ActionSequenceClass action\_sequence\_class\_ref;  
1325   readonly attribute LinkObjectClass link\_object\_class\_ref;  
1326  
1327   readonly attribute InstanceInstantiatesClassifier  
1328   instance\_instantiates\_classifier\_ref;  
1329   readonly attribute ActionOwnsArgument action\_owns\_argument\_ref;  
1330   readonly attribute CreateActionInstantiatesClassifier  
1331   create\_action\_instantiates\_classifier\_ref;  
1332   readonly attribute AttributeLinkIsInstanceOfAttribute  
1333   attribute\_link\_is\_instance\_of\_attribute\_ref;  
1334   readonly attribute AttributeLinkHasValueOfLink  
1335   attribute\_link\_has\_value\_of\_link\_ref;  
1336   readonly attribute LinkEndIsOfTypeInstance  
1337   link\_end\_is\_of\_type\_instance\_ref;  
1338   readonly attribute ReceptionFeatureReceivesSignal  
1339   reception\_feature\_receives\_signal\_ref;  
1340   readonly attribute SignalOwnsParameter signal\_owns\_parameter\_ref;  
1341   readonly attribute ActionIsInitiatedByRequest  
1342   action\_is\_initiated\_by\_request\_ref;  
1343   readonly attribute MessageInstanceIsSpecifiedByRequest  
1344   message\_instance\_is\_specified\_by\_request\_ref;  
1345   readonly attribute InstanceReceivesMessageInstance  
1346   instance\_receives\_message\_instance\_ref;  
1347   readonly attribute InstanceOwnsAttributeLink  
1348   instance\_owns\_attribute\_link\_ref;

```

1349  readonly attribute MessageInstanceHasArgumentOfInstance
1350    message_instance_has_argument_of_instance_ref;
1351  readonly attribute ExceptionIsRaisedByBehavioralFeature
1352    exception_is_raised_by_behavioral_feature_ref;
1353  readonly attribute LinkInstantiatesAssociation
1354    link_instantiates_association_ref;
1355  readonly attribute LinkOwnsLinkEnd link_owns_link_end_ref;
1356  readonly attribute LinkEndInstantiatesAssociationEnd
1357    link_end_instantiates_association_end_ref;
1358  readonly attribute InstanceSendsMessageInstance
1359    instance_sends_message_instance_ref;
1360  readonly attribute ActionSequenceOwnsAction
1361    action_sequence_owns_action_ref;
1362  };
1363  };

```

#### 5.4.6 UMLStateMachines

```

1 #include "UmlCommonBehavior.idl"
2
3 module UmlStateMachines {
4  interface UmlStateMachinesPackage;
5  interface ActivityState;
6  interface ActivityStateClass;
7  typedef sequence<ActivityState> ActivityStateUList;
8  interface CompositeState;
9  interface CompositeStateClass;
10 typedef sequence<CompositeState> CompositeStateUList;
11 interface TimeEvent;
12 interface TimeEventClass;
13 typedef sequence<TimeEvent> TimeEventUList;
14 interface ActionState;
15 interface ActionStateClass;
16 typedef sequence<ActionState> ActionStateUList;
17 interface CallEvent;
18 interface CallEventClass;
19 typedef sequence<CallEvent> CallEventUList;
20 typedef sequence<CallEvent> CallEventSet;
21 interface ChangeEvent;
22 interface ChangeEventClass;
23 typedef sequence<ChangeEvent> ChangeEventUList;

```

```
24 interface SignalEvent;
25 interface SignalEventClass;
26 typedef sequence<SignalEvent> SignalEventUList;
27 typedef sequence<SignalEvent> SignalEventSet;
28 interface Pseudostate;
29 interface PseudostateClass;
30 typedef sequence<Pseudostate> PseudostateUList;
31 interface Event;
32 interface EventClass;
33 typedef sequence<Event> EventUList;
34 typedef sequence<Event> EventSet;
35 interface StateVertex;
36 interface StateVertexClass;
37 typedef sequence<StateVertex> StateVertexUList;
38 typedef sequence<StateVertex> StateVertexSet;
39 interface StateMachine;
40 interface StateMachineClass;
41 typedef sequence<StateMachine> StateMachineUList;
42 typedef sequence<StateMachine> StateMachineSet;
43 interface SimpleState;
44 interface SimpleStateClass;
45 typedef sequence<SimpleState> SimpleStateUList;
46 interface ObjectFlowState;
47 interface ObjectFlowStateClass;
48 typedef sequence<ObjectFlowState> ObjectFlowStateUList;
49 typedef sequence<ObjectFlowState> ObjectFlowStateSet;
50 interface SubmachineState;
51 interface SubmachineStateClass;
52 typedef sequence<SubmachineState> SubmachineStateUList;
53 typedef sequence<SubmachineState> SubmachineStateSet;
54 interface ActivityModel;
55 interface ActivityModelClass;
56 typedef sequence<ActivityModel> ActivityModelUList;
57 interface Guard;
58 interface GuardClass;
59 typedef sequence<Guard> GuardUList;
60 interface Transition;
61 interface TransitionClass;
62 typedef sequence<Transition> TransitionUList;
63 typedef sequence<Transition> TransitionSet;
64 interface ClassifierInState;
```

```

65 interface ClassifierInStateClass;
66 typedef sequence<ClassifierInState> ClassifierInStateUList;
67 typedef sequence<ClassifierInState> ClassifierInStateSet;
68 interface Partition;
69 interface PartitionClass;
70 typedef sequence<Partition> PartitionUList;
71 typedef sequence<Partition> PartitionSet;
72 interface State;
73 interface StateClass;
74 typedef sequence<State> StateUList;
75 typedef sequence<State> StateSet;
76
77 interface EventClass : ::UmlCore::ModelElementClass {
78     readonly attribute EventUList all_of_kind_event;
79 };
80
81 interface Event : EventClass, ::UmlCore::ModelElement {
82     UmlStateMachines::StateSet state ()
83     raises (Reflective::NotSet, Reflective::SemanticError);
84     void add_state (in UmlStateMachines::StateSet new_value)
85     raises (Reflective::StructuralError, Reflective::SemanticError);
86     void remove_state ()
87     raises (Reflective::SemanticError);
88     UmlStateMachines::TransitionSet transition ()
89     raises (Reflective::NotSet, Reflective::SemanticError);
90     void add_transition (in UmlStateMachines::TransitionSet new_value)
91     raises (Reflective::StructuralError, Reflective::SemanticError);
92     void remove_transition ()
93     raises (Reflective::SemanticError);
94 };
95
96 interface CallEventClass : EventClass {
97     readonly attribute CallEventUList all_of_kind_call_event;
98     readonly attribute CallEventUList all_of_type_call_event;
99     CallEvent create_call_event (in ::UmlCore::Name name)
100     raises (Reflective::SemanticError);
101 };
102
103 interface CallEvent : CallEventClass, Event {
104     ::UmlCore::Operation operation ()
105     raises (Reflective::SemanticError);

```

```
106 void set_operation (in ::UmlCore::Operation new_value)
107     raises (Reflective::SemanticError);
108 };
109
110 interface ChangeEventClass : EventClass {
111     readonly attribute ChangeEventUList all_of_kind_change_event;
112     readonly attribute ChangeEventUList all_of_type_change_event;
113     ChangeEvent create_change_event (
114         in ::UmlCore::Name name,
115         in ::UmlCore::BooleanExpression change_expression)
116     raises (Reflective::SemanticError);
117 };
118
119 interface ChangeEvent : ChangeEventClass, Event {
120     ::UmlCore::BooleanExpression change_expression ()
121     raises (Reflective::SemanticError);
122     void set_change_expression (in ::UmlCore::BooleanExpression new_value)
123     raises (Reflective::SemanticError);
124 };
125
126 interface SignalEventClass : EventClass {
127     readonly attribute SignalEventUList all_of_kind_signal_event;
128     readonly attribute SignalEventUList all_of_type_signal_event;
129     SignalEvent create_signal_event (in ::UmlCore::Name name)
130     raises (Reflective::SemanticError);
131 };
132
133 interface SignalEvent : SignalEventClass, Event {
134     ::UmlCommonBehavior::Signal signal ()
135     raises (Reflective::SemanticError);
136     void set_signal (in ::UmlCommonBehavior::Signal new_value)
137     raises (Reflective::SemanticError);
138 };
139
140 interface TimeEventClass : EventClass {
141     readonly attribute TimeEventUList all_of_kind_time_event;
142     readonly attribute TimeEventUList all_of_type_time_event;
143     TimeEvent create_time_event (in ::UmlCore::Name name,
144         in ::UmlCore::TimeExpression duration)
145     raises (Reflective::SemanticError);
146 };
```



```
147
148 interface TimeEvent : TimeEventClass, Event {
149     ::UmlCore::TimeExpression duration ()
150     raises (Reflective::SemanticError);
151     void set_duration (in ::UmlCore::TimeExpression new_value)
152     raises (Reflective::SemanticError);
153 };
154
155 interface StateVertexClass : ::UmlCore::ModelElementClass {
156     readonly attribute StateVertexUList all_of_kind_state_vertex;
157 };
158
159 interface StateVertex : StateVertexClass, ::UmlCore::ModelElement {
160     CompositeState parent ()
161     raises (Reflective::NotSet, Reflective::SemanticError);
162     void set_parent (in CompositeState new_value)
163     raises (Reflective::SemanticError);
164     void unset_parent ()
165     raises (Reflective::SemanticError);
166     TransitionSet outgoing ()
167     raises (Reflective::NotSet, Reflective::SemanticError);
168     void add_outgoing (in TransitionSet new_value)
169     raises (Reflective::StructuralError, Reflective::SemanticError);
170     void remove_outgoing ()
171     raises (Reflective::SemanticError);
172     TransitionSet incoming ()
173     raises (Reflective::NotSet, Reflective::SemanticError);
174     void add_incoming (in TransitionSet new_value)
175     raises (Reflective::StructuralError, Reflective::SemanticError);
176     void remove_incoming ()
177     raises (Reflective::SemanticError);
178 };
179
180 interface GuardClass : ::UmlCore::ModelElementClass {
181     readonly attribute GuardUList all_of_kind_guard;
182     readonly attribute GuardUList all_of_type_guard;
183     Guard create_guard (in ::UmlCore::Name name,
184         in ::UmlCore::BooleanExpression expression)
185     raises (Reflective::SemanticError);
186 };
187
```

```
188 interface Guard : GuardClass, ::UmlCore::ModelElement {
189     ::UmlCore::BooleanExpression expression ()
190     raises (Reflective::SemanticError);
191     void set_expression (in ::UmlCore::BooleanExpression new_value)
192     raises (Reflective::SemanticError);
193     UmlStateMachines::Transition transition ()
194     raises (Reflective::SemanticError);
195     void set_transition (in UmlStateMachines::Transition new_value)
196     raises (Reflective::SemanticError);
197 };
198
199 interface TransitionClass : ::UmlCore::ModelElementClass {
200     readonly attribute TransitionUList all_of_kind_transition;
201     readonly attribute TransitionUList all_of_type_transition;
202     Transition create_transition (in ::UmlCore::Name name)
203     raises (Reflective::SemanticError);
204 };
205
206 interface Transition : TransitionClass, ::UmlCore::ModelElement {
207     UmlStateMachines::Guard guard ()
208     raises (Reflective::NotSet, Reflective::SemanticError);
209     void set_guard (in UmlStateMachines::Guard new_value)
210     raises (Reflective::SemanticError);
211     void add_guard_before (in UmlStateMachines::Guard new_value,
212         in UmlStateMachines::Guard before)
213     raises (Reflective::StructuralError,
214         Reflective::NotFound,
215         Reflective::SemanticError);
216     void unset_guard ()
217     raises (Reflective::SemanticError);
218     ::UmlCommonBehavior::ActionSequence effect ()
219     raises (Reflective::NotSet, Reflective::SemanticError);
220     void set_effect (in ::UmlCommonBehavior::ActionSequence new_value)
221     raises (Reflective::SemanticError);
222     void unset_effect ()
223     raises (Reflective::SemanticError);
224     UmlStateMachines::State state ()
225     raises (Reflective::NotSet, Reflective::SemanticError);
226     void set_state (in UmlStateMachines::State new_value)
227     raises (Reflective::SemanticError);
228     void unset_state ()
```

```

229     raises (Reflective::SemanticError);
230 Event trigger ()
231     raises (Reflective::NotSet, Reflective::SemanticError);
232 void set_trigger (in Event new_value)
233     raises (Reflective::SemanticError);
234 void unset_trigger ()
235     raises (Reflective::SemanticError);
236 UmlStateMachines::StateMachine state_machine ()
237     raises (Reflective::NotSet, Reflective::SemanticError);
238 void set_state_machine (in UmlStateMachines::StateMachine new_value)
239     raises (Reflective::SemanticError);
240 void unset_state_machine ()
241     raises (Reflective::SemanticError);
242 StateVertex source ()
243     raises (Reflective::SemanticError);
244 void set_source (in StateVertex new_value)
245     raises (Reflective::SemanticError);
246 StateVertex target ()
247     raises (Reflective::SemanticError);
248 void set_target (in StateVertex new_value)
249     raises (Reflective::SemanticError);
250 };
251
252 interface PseudostateClass : StateVertexClass {
253     readonly attribute PseudostateUList all_of_kind_pseudostate;
254     readonly attribute PseudostateUList all_of_type_pseudostate;
255     Pseudostate create_pseudostate (in ::UmlCore::Name name,
256                                     in ::UmlCore::PseudostateKind kind)
257     raises (Reflective::SemanticError);
258 };
259
260 interface Pseudostate : PseudostateClass, StateVertex {
261     ::UmlCore::PseudostateKind kind ()
262     raises (Reflective::SemanticError);
263     void set_kind (in ::UmlCore::PseudostateKind new_value)
264     raises (Reflective::SemanticError);
265 };
266
267 interface StateClass : StateVertexClass {
268     readonly attribute StateUList all_of_kind_state;
269     readonly attribute StateUList all_of_type_state;

```

```
270   State create_state (in ::UmlCore::Name name)
271     raises (Reflective::SemanticError);
272 };
273
274 interface State : StateClass, StateVertex {
275   ::UmlCommonBehavior::ActionSequence entry ()
276     raises (Reflective::NotSet, Reflective::SemanticError);
277   void set_entry (in ::UmlCommonBehavior::ActionSequence new_value)
278     raises (Reflective::SemanticError);
279   void unset_entry ()
280     raises (Reflective::SemanticError);
281   ::UmlCommonBehavior::ActionSequence exit ()
282     raises (Reflective::NotSet, Reflective::SemanticError);
283   void set_exit (in ::UmlCommonBehavior::ActionSequence new_value)
284     raises (Reflective::SemanticError);
285   void unset_exit ()
286     raises (Reflective::SemanticError);
287   UmlStateMachines::ClassifierInStateSet classifier_in_state ()
288     raises (Reflective::NotSet, Reflective::SemanticError);
289   void add_classifier_in_state (
290     in UmlStateMachines::ClassifierInStateSet new_value)
291     raises (Reflective::StructuralError, Reflective::SemanticError);
292   void remove_classifier_in_state ()
293     raises (Reflective::SemanticError);
294   UmlStateMachines::StateMachine state_machine ()
295     raises (Reflective::NotSet, Reflective::SemanticError);
296   void set_state_machine (in UmlStateMachines::StateMachine new_value)
297     raises (Reflective::SemanticError);
298   void unset_state_machine ()
299     raises (Reflective::SemanticError);
300   EventSet deferred_event ()
301     raises (Reflective::NotSet, Reflective::SemanticError);
302   void add_deferred_event (in EventSet new_value)
303     raises (Reflective::StructuralError, Reflective::SemanticError);
304   void remove_deferred_event ()
305     raises (Reflective::SemanticError);
306   TransitionSet internal_transition ()
307     raises (Reflective::NotSet, Reflective::SemanticError);
308   void add_internal_transition (in TransitionSet new_value)
309     raises (Reflective::StructuralError, Reflective::SemanticError);
310   void remove_internal_transition ()
```

```
311     raises (Reflective::SemanticError);
312 };
313
314 interface CompositeStateClass : StateClass {
315     readonly attribute CompositeStateUList all_of_kind_composite_state;
316     readonly attribute CompositeStateUList all_of_type_composite_state;
317     CompositeState create_composite_state (in ::UmlCore::Name name,
318         in boolean is_concurrent)
319     raises (Reflective::SemanticError);
320 };
321
322 interface CompositeState : CompositeStateClass, State {
323     boolean is_concurrent ()
324     raises (Reflective::SemanticError);
325     void set_is_concurrent (in boolean new_value)
326     raises (Reflective::SemanticError);
327     StateVertexSet substate ()
328     raises (Reflective::SemanticError);
329     void add_substate (in StateVertexSet new_value)
330     raises (Reflective::StructuralError, Reflective::SemanticError);
331     void remove_substate ()
332     raises (Reflective::SemanticError);
333 };
334
335 interface PartitionClass : ::UmlCore::ModelElementClass {
336     readonly attribute PartitionUList all_of_kind_partition;
337     readonly attribute PartitionUList all_of_type_partition;
338     Partition create_partition (in ::UmlCore::Name name)
339     raises (Reflective::SemanticError);
340 };
341
342 interface Partition : PartitionClass, ::UmlCore::ModelElement {
343     UmlStateMachines::ActivityModel activity_model ()
344     raises (Reflective::SemanticError);
345     void set_activity_model (in UmlStateMachines::ActivityModel new_value)
346     raises (Reflective::SemanticError);
347     ::UmlCore::ModelElementSet contents ()
348     raises (Reflective::NotSet, Reflective::SemanticError);
349     void add_contents (in ::UmlCore::ModelElementSet new_value)
350     raises (Reflective::StructuralError, Reflective::SemanticError);
351     void remove_contents ()
```

```
352     raises (Reflective::SemanticError);
353 };
354
355 interface ClassifierInStateClass : ::UmlCore::ClassifierClass {
356     readonly attribute ClassifierInStateUList all_of_kind_classifier_in_state;
357     readonly attribute ClassifierInStateUList all_of_type_classifier_in_state;
358     ClassifierInState create_classifier_in_state (in ::UmlCore::Name name,
359                                                 in boolean is_root,
360                                                 in boolean is_leaf,
361                                                 in boolean is_abstract)
362     raises (Reflective::SemanticError);
363 };
364
365 interface ClassifierInState : ClassifierInStateClass, ::UmlCore::Classifier {
366     State in_state ()
367     raises (Reflective::SemanticError);
368     void set_in_state (in State new_value)
369     raises (Reflective::SemanticError);
370     UmlStateMachines::ObjectFlowStateSet object_flow_state ()
371     raises (Reflective::NotSet, Reflective::SemanticError);
372     void add_object_flow_state (
373         in UmlStateMachines::ObjectFlowStateSet new_value)
374     raises (Reflective::StructuralError, Reflective::SemanticError);
375     void remove_object_flow_state ()
376     raises (Reflective::SemanticError);
377     ::UmlCore::Classifier type ()
378     raises (Reflective::SemanticError);
379     void set_type (in ::UmlCore::Classifier new_value)
380     raises (Reflective::SemanticError);
381 };
382
383 interface StateMachineClass : ::UmlCore::ModelElementClass {
384     readonly attribute StateMachineUList all_of_kind_state_machine;
385     readonly attribute StateMachineUList all_of_type_state_machine;
386     StateMachine create_state_machine (in ::UmlCore::Name name)
387     raises (Reflective::SemanticError);
388 };
389
390 interface StateMachine : StateMachineClass, ::UmlCore::ModelElement {
391     State top ()
392     raises (Reflective::SemanticError);
```

```

393 void set_top (in State new_value)
394     raises (Reflective::SemanticError);
395 TransitionSet transitions ()
396     raises (Reflective::NotSet, Reflective::SemanticError);
397 void add_transitions (in TransitionSet new_value)
398     raises (Reflective::StructuralError, Reflective::SemanticError);
399 void remove_transitions ()
400     raises (Reflective::SemanticError);
401 UmlStateMachines::SubmachineStateSet submachine_state ()
402     raises (Reflective::NotSet, Reflective::SemanticError);
403 void add_submachine_state (
404     in UmlStateMachines::SubmachineStateSet new_value)
405     raises (Reflective::StructuralError, Reflective::SemanticError);
406 void remove_submachine_state ()
407     raises (Reflective::SemanticError);
408 ::UmlCore::ModelElement uml_context ()
409     raises (Reflective::NotSet, Reflective::SemanticError);
410 void set_uml_context (in ::UmlCore::ModelElement new_value)
411     raises (Reflective::SemanticError);
412 void unset_uml_context ()
413     raises (Reflective::SemanticError);
414 };
415
416 interface ActivityModelClass : StateMachineClass {
417     readonly attribute ActivityModelUList all_of_kind_activity_model;
418     readonly attribute ActivityModelUList all_of_type_activity_model;
419     ActivityModel create_activity_model (in ::UmlCore::Name name)
420     raises (Reflective::SemanticError);
421 };
422
423 interface ActivityModel : ActivityModelClass, StateMachine {
424     PartitionSet owned_partition ()
425     raises (Reflective::NotSet, Reflective::SemanticError);
426     void add_owned_partition (in PartitionSet new_value)
427     raises (Reflective::StructuralError, Reflective::SemanticError);
428     void remove_owned_partition ()
429     raises (Reflective::SemanticError);
430 };
431
432 interface SimpleStateClass : StateClass {
433     readonly attribute SimpleStateUList all_of_kind_simple_state;

```

```
434     readonly attribute SimpleStateUList all_of_type_simple_state;
435     SimpleState create_simple_state (in ::UmlCore::Name name)
436     raises (Reflective::SemanticError);
437 };
438
439 interface SimpleState : SimpleStateClass, State { };
440
441 interface ActivityStateClass : SimpleStateClass {
442     readonly attribute ActivityStateUList all_of_kind_activity_state;
443     readonly attribute ActivityStateUList all_of_type_activity_state;
444     ActivityState create_activity_state (in ::UmlCore::Name name)
445     raises (Reflective::SemanticError);
446 };
447
448 interface ActivityState : ActivityStateClass, SimpleState { };
449
450 interface ObjectFlowStateClass : SimpleStateClass {
451     readonly attribute ObjectFlowStateUList all_of_kind_object_flow_state;
452     readonly attribute ObjectFlowStateUList all_of_type_object_flow_state;
453     ObjectFlowState create_object_flow_state (in ::UmlCore::Name name)
454     raises (Reflective::SemanticError);
455 };
456
457 interface ObjectFlowState : ObjectFlowStateClass, SimpleState {
458     UmlStateMachines::ClassifierInState type_state ()
459     raises (Reflective::SemanticError);
460     void set_type_state (in UmlStateMachines::ClassifierInState new_value)
461     raises (Reflective::SemanticError);
462 };
463
464 interface ActionStateClass : SimpleStateClass {
465     readonly attribute ActionStateUList all_of_kind_action_state;
466     readonly attribute ActionStateUList all_of_type_action_state;
467     ActionState create_action_state (in ::UmlCore::Name name)
468     raises (Reflective::SemanticError);
469 };
470
471 interface ActionState : ActionStateClass, SimpleState { };
472
473 interface SubmachineStateClass : StateClass {
474     readonly attribute SubmachineStateUList all_of_kind_submachine_state;
```



```

475     readonly attribute SubmachineStateUList all_of_type_submachine_state;
476     SubmachineState create_submachine_state (in ::UmlCore::Name name)
477         raises (Reflective::SemanticError);
478 };
479
480 interface SubmachineState : SubmachineStateClass, State {
481     UmlStateMachines::StateMachine submachine ()
482         raises (Reflective::SemanticError);
483     void set_submachine (in UmlStateMachines::StateMachine new_value)
484         raises (Reflective::SemanticError);
485 };
486
487 struct StateOwnsEntryActionSequenceLink {
488     State entry_action_state;
489     ::UmlCommonBehavior::ActionSequence entry;
490 };
491 typedef sequence <StateOwnsEntryActionSequenceLink>
492     StateOwnsEntryActionSequenceLinkSet;
493
494 interface StateOwnsEntryActionSequence : Reflective::RefAssociation {
495     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
496     StateOwnsEntryActionSequenceLinkSet
497         all_state_owns_entry_action_sequence_links();
498     boolean exists (in State entry_action_state,
499                   in ::UmlCommonBehavior::ActionSequence entry);
500     State with_entry (in ::UmlCommonBehavior::ActionSequence entry);
501     ::UmlCommonBehavior::ActionSequence with_entry_action_state (
502         in State entry_action_state);
503     void add (in State entry_action_state,
504              in ::UmlCommonBehavior::ActionSequence entry)
505         raises (Reflective::StructuralError, Reflective::SemanticError);
506     void modify_entry_action_state (
507         in State entry_action_state,
508         in ::UmlCommonBehavior::ActionSequence entry,
509         in State new_entry_action_state)
510         raises (Reflective::StructuralError,
511               Reflective::SemanticError,
512               Reflective::NotFound);
513     void modify_entry (in State entry_action_state,
514                       in ::UmlCommonBehavior::ActionSequence entry,
515                       in ::UmlCommonBehavior::ActionSequence new_entry)

```

```
516     raises (Reflective::StructuralError,
517           Reflective::SemanticError,
518           Reflective::NotFound);
519 void remove (in State entry_action_state,
520             in ::UmlCommonBehavior::ActionSequence entry)
521     raises (Reflective::StructuralError,
522           Reflective::SemanticError,
523           Reflective::NotFound);
524 };
525
526 struct StateOwnsExitActionSequenceLink {
527     State exit_action_state;
528     ::UmlCommonBehavior::ActionSequence exit;
529 };
530 typedef sequence <StateOwnsExitActionSequenceLink>
531 StateOwnsExitActionSequenceLinkSet;
532
533 interface StateOwnsExitActionSequence : Reflective::RefAssociation {
534     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
535     StateOwnsExitActionSequenceLinkSet
536     all_state_owns_exit_action_sequence_links();
537     boolean exists (in State exit_action_state,
538                   in ::UmlCommonBehavior::ActionSequence exit);
539     State with_exit (in ::UmlCommonBehavior::ActionSequence exit);
540     ::UmlCommonBehavior::ActionSequence with_exit_action_state (
541         in State exit_action_state);
542     void add (in State exit_action_state,
543             in ::UmlCommonBehavior::ActionSequence exit)
544         raises (Reflective::StructuralError, Reflective::SemanticError);
545     void modify_exit_action_state (in State exit_action_state,
546                                  in ::UmlCommonBehavior::ActionSequence exit,
547                                  in State new_exit_action_state)
548         raises (Reflective::StructuralError,
549               Reflective::SemanticError,
550               Reflective::NotFound);
551     void modify_exit (in State exit_action_state,
552                     in ::UmlCommonBehavior::ActionSequence exit,
553                     in ::UmlCommonBehavior::ActionSequence new_exit)
554         raises (Reflective::StructuralError,
555               Reflective::SemanticError,
556               Reflective::NotFound);
```

```

557 void remove (in State exit_action_state,
558             in ::UmlCommonBehavior::ActionSequence exit)
559     raises (Reflective::StructuralError,
560           Reflective::SemanticError,
561           Reflective::NotFound);
562 };
563
564 struct TransitionOwnsGuardLink {
565     UmlStateMachines::Guard guard;
566     UmlStateMachines::Transition transition;
567 };
568 typedef sequence <TransitionOwnsGuardLink> TransitionOwnsGuardLinkSet;
569
570 interface TransitionOwnsGuard : Reflective::RefAssociation {
571     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
572     TransitionOwnsGuardLinkSet all_transition_owns_guard_links();
573     boolean exists (in UmlStateMachines::Guard guard,
574                  in UmlStateMachines::Transition transition);
575     UmlStateMachines::Guard with_transition (
576         in UmlStateMachines::Transition transition);
577     UmlStateMachines::Transition with_guard (in UmlStateMachines::Guard guard);
578     void add (in UmlStateMachines::Guard guard,
579             in UmlStateMachines::Transition transition)
580         raises (Reflective::StructuralError, Reflective::SemanticError);
581     void add_before_guard (in UmlStateMachines::Guard guard,
582                          in UmlStateMachines::Transition transition,
583                          in UmlStateMachines::Guard before)
584         raises (Reflective::StructuralError,
585               Reflective::SemanticError,
586               Reflective::NotFound);
587     void modify_guard (in UmlStateMachines::Guard guard,
588                      in UmlStateMachines::Transition transition,
589                      in UmlStateMachines::Guard new_guard)
590         raises (Reflective::StructuralError,
591               Reflective::SemanticError,
592               Reflective::NotFound);
593     void modify_transition (in UmlStateMachines::Guard guard,
594                            in UmlStateMachines::Transition transition,
595                            in UmlStateMachines::Transition new_transition)
596         raises (Reflective::StructuralError,
597               Reflective::SemanticError,

```

```
598         Reflective::NotFound);
599     void remove (in UmlStateMachines::Guard guard,
600                 in UmlStateMachines::Transition transition)
601         raises (Reflective::StructuralError,
602               Reflective::SemanticError,
603               Reflective::NotFound);
604 };
605
606 struct SignalEventIsOccurrenceOfSignalLink {
607     ::UmlCommonBehavior::Signal signal;
608     SignalEvent occurrence;
609 };
610 typedef sequence <SignalEventIsOccurrenceOfSignalLink>
611     SignalEventIsOccurrenceOfSignalLinkSet;
612
613 interface SignalEventIsOccurrenceOfSignal : Reflective::RefAssociation {
614     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
615     SignalEventIsOccurrenceOfSignalLinkSet
616     all_signal_event_is_occurrence_of_signal_links();
617     boolean exists (in ::UmlCommonBehavior::Signal signal,
618                   in SignalEvent occurrence);
619     ::UmlCommonBehavior::Signal with_occurrence (in SignalEvent occurrence);
620     SignalEventSet with_signal (in ::UmlCommonBehavior::Signal signal);
621     void add (in ::UmlCommonBehavior::Signal signal, in SignalEvent occurrence)
622         raises (Reflective::StructuralError, Reflective::SemanticError);
623     void modify_signal (in ::UmlCommonBehavior::Signal signal,
624                        in SignalEvent occurrence,
625                        in ::UmlCommonBehavior::Signal new_signal)
626         raises (Reflective::StructuralError,
627               Reflective::SemanticError,
628               Reflective::NotFound);
629     void modify_occurrence (in ::UmlCommonBehavior::Signal signal,
630                            in SignalEvent occurrence,
631                            in SignalEvent new_occurrence)
632         raises (Reflective::StructuralError,
633               Reflective::SemanticError,
634               Reflective::NotFound);
635     void remove (in ::UmlCommonBehavior::Signal signal,
636                 in SignalEvent occurrence)
637         raises (Reflective::StructuralError,
638               Reflective::SemanticError,
```

```
639         Reflective::NotFound);
640     };
641
642     struct ActivityModelOwnsPartitionLink {
643         UmlStateMachines::ActivityModel activity_model;
644         Partition owned_partition;
645     };
646     typedef sequence <ActivityModelOwnsPartitionLink>
647     ActivityModelOwnsPartitionLinkSet;
648
649     interface ActivityModelOwnsPartition : Reflective::RefAssociation {
650         readonly attribute UmlStateMachinesPackage enclosing_package_ref;
651         ActivityModelOwnsPartitionLinkSet
652             all_activity_model_owns_partition_links();
653         boolean exists (in UmlStateMachines::ActivityModel activity_model,
654             in Partition owned_partition);
655         UmlStateMachines::ActivityModel with_owned_partition (
656             in Partition owned_partition);
657         PartitionSet with_activity_model (
658             in UmlStateMachines::ActivityModel activity_model);
659         void add (in UmlStateMachines::ActivityModel activity_model,
660             in Partition owned_partition)
661             raises (Reflective::StructuralError, Reflective::SemanticError);
662         void modify_activity_model (
663             in UmlStateMachines::ActivityModel activity_model,
664             in Partition owned_partition,
665             in UmlStateMachines::ActivityModel new_activity_model)
666             raises (Reflective::StructuralError,
667                 Reflective::SemanticError,
668                 Reflective::NotFound);
669         void modify_owned_partition (
670             in UmlStateMachines::ActivityModel activity_model,
671             in Partition owned_partition,
672             in Partition new_owned_partition)
673             raises (Reflective::StructuralError,
674                 Reflective::SemanticError,
675                 Reflective::NotFound);
676         void remove (in UmlStateMachines::ActivityModel activity_model,
677             in Partition owned_partition)
678             raises (Reflective::StructuralError,
679                 Reflective::SemanticError,
```

```
680         Reflective::NotFound);
681     };
682
683     struct PartitionHasContextOfModelElementLink {
684         ::UmlCore::ModelElement contents;
685         Partition context_partition;
686     };
687     typedef sequence <PartitionHasContextOfModelElementLink>
688         PartitionHasContextOfModelElementLinkSet;
689
690     interface PartitionHasContextOfModelElement : Reflective::RefAssociation {
691         readonly attribute UmlStateMachinesPackage enclosing_package_ref;
692         PartitionHasContextOfModelElementLinkSet
693             all_partition_has_context_of_model_element_links();
694         boolean exists (in ::UmlCore::ModelElement contents,
695             in Partition context_partition);
696         ::UmlCore::ModelElementSet with_context_partition (
697             in Partition context_partition);
698         PartitionSet with_contents (in ::UmlCore::ModelElement contents);
699         void add (in ::UmlCore::ModelElement contents,
700             in Partition context_partition)
701             raises (Reflective::StructuralError, Reflective::SemanticError);
702         void modify_contents (in ::UmlCore::ModelElement contents,
703             in Partition context_partition,
704             in ::UmlCore::ModelElement new_contents)
705             raises (Reflective::StructuralError,
706                 Reflective::SemanticError,
707                 Reflective::NotFound);
708         void modify_context_partition (in ::UmlCore::ModelElement contents,
709             in Partition context_partition,
710             in Partition new_context_partition)
711             raises (Reflective::StructuralError,
712                 Reflective::SemanticError,
713                 Reflective::NotFound);
714         void remove (in ::UmlCore::ModelElement contents,
715             in Partition context_partition)
716             raises (Reflective::StructuralError,
717                 Reflective::SemanticError,
718                 Reflective::NotFound);
719     };
720
```

```

721 struct ClassifierInStateIsInStateStateLink {
722     UmlStateMachines::ClassifierInState classifier_in_state;
723     State in_state;
724 };
725 typedef sequence <ClassifierInStateIsInStateStateLink>
726     ClassifierInStateIsInStateStateLinkSet;
727
728 interface ClassifierInStateIsInStateState : Reflective::RefAssociation {
729     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
730     ClassifierInStateIsInStateStateLinkSet
731     all_classifier_in_state_is_in_state_state_links();
732     boolean exists (in UmlStateMachines::ClassifierInState classifier_in_state,
733                   in State in_state);
734     UmlStateMachines::ClassifierInStateSet with_in_state (in State in_state);
735     State with_classifier_in_state (
736         in UmlStateMachines::ClassifierInState classifier_in_state);
737     void add (in UmlStateMachines::ClassifierInState classifier_in_state,
738             in State in_state)
739         raises (Reflective::StructuralError, Reflective::SemanticError);
740     void modify_classifier_in_state (
741         in UmlStateMachines::ClassifierInState classifier_in_state,
742         in State in_state,
743         in UmlStateMachines::ClassifierInState new_classifier_in_state)
744         raises (Reflective::StructuralError,
745             Reflective::SemanticError,
746             Reflective::NotFound);
747     void modify_in_state (
748         in UmlStateMachines::ClassifierInState classifier_in_state,
749         in State in_state,
750         in State new_in_state)
751         raises (Reflective::StructuralError,
752             Reflective::SemanticError,
753             Reflective::NotFound);
754     void remove (in UmlStateMachines::ClassifierInState classifier_in_state,
755                in State in_state)
756         raises (Reflective::StructuralError,
757             Reflective::SemanticError,
758             Reflective::NotFound);
759 };
760
761 struct ObjectFlowStateRepresentsClassifierInStateLink {

```

```
762 ClassifierInState type_state;
763 UmlStateMachines::ObjectFlowState object_flow_state;
764 };
765 typedef sequence <ObjectFlowStateRepresentsClassifierInStateLink>
766 ObjectFlowStateRepresentsClassifierInStateLinkSet;
767
768 interface ObjectFlowStateRepresentsClassifierInState :
769     Reflective::RefAssociation {
770     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
771     ObjectFlowStateRepresentsClassifierInStateLinkSet
772     all_object_flow_state_represents_classifier_in_state_links();
773     boolean exists (in ClassifierInState type_state,
774         in UmlStateMachines::ObjectFlowState object_flow_state);
775     ClassifierInState with_object_flow_state (
776         in UmlStateMachines::ObjectFlowState object_flow_state);
777     UmlStateMachines::ObjectFlowStateSet with_type_state (
778         in ClassifierInState type_state);
779     void add (in ClassifierInState type_state,
780         in UmlStateMachines::ObjectFlowState object_flow_state)
781         raises (Reflective::StructuralError, Reflective::SemanticError);
782     void modify_type_state (
783         in ClassifierInState type_state,
784         in UmlStateMachines::ObjectFlowState object_flow_state,
785         in ClassifierInState new_type_state)
786         raises (Reflective::StructuralError,
787             Reflective::SemanticError,
788             Reflective::NotFound);
789     void modify_object_flow_state (
790         in ClassifierInState type_state,
791         in UmlStateMachines::ObjectFlowState object_flow_state,
792         in UmlStateMachines::ObjectFlowState new_object_flow_state)
793         raises (Reflective::StructuralError,
794             Reflective::SemanticError,
795             Reflective::NotFound);
796     void remove (in ClassifierInState type_state,
797         in UmlStateMachines::ObjectFlowState object_flow_state)
798         raises (Reflective::StructuralError,
799             Reflective::SemanticError,
800             Reflective::NotFound);
801 };
802
```



```
803 struct ClassifierInStateIsOfTypeClassifierLink {
804     ::UmlCore::Classifier type;
805     UmlStateMachines::ClassifierInState classifier_in_state;
806 };
807 typedef sequence <ClassifierInStateIsOfTypeClassifierLink>
808     ClassifierInStateIsOfTypeClassifierLinkSet;
809
810 interface ClassifierInStateIsOfTypeClassifier : Reflective::RefAssociation {
811     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
812     ClassifierInStateIsOfTypeClassifierLinkSet
813     all_classifier_in_state_is_of_type_classifier_links();
814     boolean exists (
815         in ::UmlCore::Classifier type,
816         in UmlStateMachines::ClassifierInState classifier_in_state);
817     ::UmlCore::Classifier with_classifier_in_state (
818         in UmlStateMachines::ClassifierInState classifier_in_state);
819     UmlStateMachines::ClassifierInStateSet with_type (
820         in ::UmlCore::Classifier type);
821     void add (in ::UmlCore::Classifier type,
822             in UmlStateMachines::ClassifierInState classifier_in_state)
823         raises (Reflective::StructuralError, Reflective::SemanticError);
824     void modify_type (
825         in ::UmlCore::Classifier type,
826         in UmlStateMachines::ClassifierInState classifier_in_state,
827         in ::UmlCore::Classifier new_type)
828         raises (Reflective::StructuralError,
829             Reflective::SemanticError,
830             Reflective::NotFound);
831     void modify_classifier_in_state (
832         in ::UmlCore::Classifier type,
833         in UmlStateMachines::ClassifierInState classifier_in_state,
834         in UmlStateMachines::ClassifierInState new_classifier_in_state)
835         raises (Reflective::StructuralError,
836             Reflective::SemanticError,
837             Reflective::NotFound);
838     void remove (in ::UmlCore::Classifier type,
839                in UmlStateMachines::ClassifierInState classifier_in_state)
840         raises (Reflective::StructuralError,
841             Reflective::SemanticError,
842             Reflective::NotFound);
843 };
```

```
844
845 struct StateMachineOwnsTopStateLink {
846     State top;
847     UmlStateMachines::StateMachine state_machine;
848 };
849 typedef sequence <StateMachineOwnsTopStateLink>
850     StateMachineOwnsTopStateLinkSet;
851
852 interface StateMachineOwnsTopState : Reflective::RefAssociation {
853     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
854     StateMachineOwnsTopStateLinkSet all_state_machine_owns_top_state_links();
855     boolean exists (in State top,
856                   in UmlStateMachines::StateMachine state_machine);
857     State with_state_machine (in UmlStateMachines::StateMachine state_machine);
858     UmlStateMachines::StateMachine with_top (in State top);
859     void add (in State top, in UmlStateMachines::StateMachine state_machine)
860         raises (Reflective::StructuralError, Reflective::SemanticError);
861     void modify_top (in State top,
862                    in UmlStateMachines::StateMachine state_machine,
863                    in State new_top)
864         raises (Reflective::StructuralError,
865               Reflective::SemanticError,
866               Reflective::NotFound);
867     void modify_state_machine (
868         in State top,
869         in UmlStateMachines::StateMachine state_machine,
870         in UmlStateMachines::StateMachine new_state_machine)
871         raises (Reflective::StructuralError,
872               Reflective::SemanticError,
873               Reflective::NotFound);
874     void remove (in State top, in UmlStateMachines::StateMachine state_machine)
875         raises (Reflective::StructuralError,
876               Reflective::SemanticError,
877               Reflective::NotFound);
878 };
879
880 struct StateDefersEventLink {
881     UmlStateMachines::State state;
882     Event deferred_event;
883 };
884 typedef sequence <StateDefersEventLink> StateDefersEventLinkSet;
```

```

885
886 interface StateDefersEvent : Reflective::RefAssociation {
887     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
888     StateDefersEventLinkSet all_state_defers_event_links();
889     boolean exists (in UmlStateMachines::State state, in Event deferred_event);
890     UmlStateMachines::StateSet with_deferred_event (in Event deferred_event);
891     EventSet with_state (in UmlStateMachines::State state);
892     void add (in UmlStateMachines::State state, in Event deferred_event)
893         raises (Reflective::StructuralError, Reflective::SemanticError);
894     void modify_state (in UmlStateMachines::State state,
895                       in Event deferred_event,
896                       in UmlStateMachines::State new_state)
897         raises (Reflective::StructuralError,
898               Reflective::SemanticError,
899               Reflective::NotFound);
900     void modify_deferred_event (in UmlStateMachines::State state,
901                                in Event deferred_event,
902                                in Event new_deferred_event)
903         raises (Reflective::StructuralError,
904               Reflective::SemanticError,
905               Reflective::NotFound);
906     void remove (in UmlStateMachines::State state, in Event deferred_event)
907         raises (Reflective::StructuralError,
908               Reflective::SemanticError,
909               Reflective::NotFound);
910 };
911
912 struct CallEventIsOccurrenceOfOperationLink {
913     CallEvent occurrence;
914     ::UmlCore::Operation operation;
915 };
916 typedef sequence <CallEventIsOccurrenceOfOperationLink>
917     CallEventIsOccurrenceOfOperationLinkSet;
918
919 interface CallEventIsOccurrenceOfOperation : Reflective::RefAssociation {
920     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
921     CallEventIsOccurrenceOfOperationLinkSet
922         all_call_event_is_occurrence_of_operation_links();
923     boolean exists (in CallEvent occurrence,
924                   in ::UmlCore::Operation operation);
925     CallEventSet with_operation (in ::UmlCore::Operation operation);

```

```
926     ::UmlCore::Operation with_occurrence (in CallEvent occurrence);
927     void add (in CallEvent occurrence, in ::UmlCore::Operation operation)
928         raises (Reflective::StructuralError, Reflective::SemanticError);
929     void modify_occurrence (in CallEvent occurrence,
930         in ::UmlCore::Operation operation,
931         in CallEvent new_occurrence)
932         raises (Reflective::StructuralError,
933             Reflective::SemanticError,
934             Reflective::NotFound);
935     void modify_operation (in CallEvent occurrence,
936         in ::UmlCore::Operation operation,
937         in ::UmlCore::Operation new_operation)
938         raises (Reflective::StructuralError,
939             Reflective::SemanticError,
940             Reflective::NotFound);
941     void remove (in CallEvent occurrence, in ::UmlCore::Operation operation)
942         raises (Reflective::StructuralError,
943             Reflective::SemanticError,
944             Reflective::NotFound);
945 };
946
947 struct CompositeStateOwnsSubstateStateVertexLink {
948     CompositeState parent;
949     StateVertex substate;
950 };
951 typedef sequence <CompositeStateOwnsSubstateStateVertexLink>
952     CompositeStateOwnsSubstateStateVertexLinkSet;
953
954 interface CompositeStateOwnsSubstateStateVertex :
955     Reflective::RefAssociation {
956     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
957     CompositeStateOwnsSubstateStateVertexLinkSet
958         all_composite_state_owns_substate_state_vertex_links();
959     boolean exists (in CompositeState parent, in StateVertex substate);
960     CompositeState with_substate (in StateVertex substate);
961     StateVertexSet with_parent (in CompositeState parent);
962     void add (in CompositeState parent, in StateVertex substate)
963         raises (Reflective::StructuralError, Reflective::SemanticError);
964     void modify_parent (in CompositeState parent,
965         in StateVertex substate,
966         in CompositeState new_parent)
```

```

967     raises (Reflective::StructuralError,
968           Reflective::SemanticError,
969           Reflective::NotFound);
970 void modify_substate (in CompositeState parent,
971                     in StateVertex substate,
972                     in StateVertex new_substate)
973     raises (Reflective::StructuralError,
974           Reflective::SemanticError,
975           Reflective::NotFound);
976 void remove (in CompositeState parent, in StateVertex substate)
977     raises (Reflective::StructuralError,
978           Reflective::SemanticError,
979           Reflective::NotFound);
980 };
981
982 struct TransitionOwnsEffectActionSequenceLink {
983     UmlStateMachines::Transition transition;
984     ::UmlCommonBehavior::ActionSequence effect;
985 };
986 typedef sequence <TransitionOwnsEffectActionSequenceLink>
987     TransitionOwnsEffectActionSequenceLinkSet;
988
989 interface TransitionOwnsEffectActionSequence : Reflective::RefAssociation {
990     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
991     TransitionOwnsEffectActionSequenceLinkSet
992         all_transition_owns_effect_action_sequence_links();
993     boolean exists (in UmlStateMachines::Transition transition,
994                  in ::UmlCommonBehavior::ActionSequence effect);
995     UmlStateMachines::Transition with_effect (
996         in ::UmlCommonBehavior::ActionSequence effect);
997     ::UmlCommonBehavior::ActionSequence with_transition (
998         in UmlStateMachines::Transition transition);
999     void add (in UmlStateMachines::Transition transition,
1000             in ::UmlCommonBehavior::ActionSequence effect)
1001         raises (Reflective::StructuralError, Reflective::SemanticError);
1002     void modify_transition (in UmlStateMachines::Transition transition,
1003                          in ::UmlCommonBehavior::ActionSequence effect,
1004                          in UmlStateMachines::Transition new_transition)
1005         raises (Reflective::StructuralError,
1006               Reflective::SemanticError,
1007               Reflective::NotFound);

```

```
1008 void modify_effect (in UmlStateMachines::Transition transition,
1009                     in ::UmlCommonBehavior::ActionSequence effect,
1010                     in ::UmlCommonBehavior::ActionSequence new_effect)
1011     raises (Reflective::StructuralError,
1012           Reflective::SemanticError,
1013           Reflective::NotFound);
1014 void remove (in UmlStateMachines::Transition transition,
1015             in ::UmlCommonBehavior::ActionSequence effect)
1016     raises (Reflective::StructuralError,
1017           Reflective::SemanticError,
1018           Reflective::NotFound);
1019 };
1020
1021 struct StateOwnsInternalTransitionLink {
1022     UmlStateMachines::State state;
1023     Transition internal_transition;
1024 };
1025 typedef sequence <StateOwnsInternalTransitionLink>
1026     StateOwnsInternalTransitionLinkSet;
1027
1028 interface StateOwnsInternalTransition : Reflective::RefAssociation {
1029     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
1030     StateOwnsInternalTransitionLinkSet
1031     all_state_owns_internal_transition_links();
1032     boolean exists (in UmlStateMachines::State state,
1033                   in Transition internal_transition);
1034     UmlStateMachines::State with_internal_transition (
1035         in Transition internal_transition);
1036     TransitionSet with_state (in UmlStateMachines::State state);
1037     void add (in UmlStateMachines::State state,
1038             in Transition internal_transition)
1039         raises (Reflective::StructuralError, Reflective::SemanticError);
1040     void modify_state (in UmlStateMachines::State state,
1041                      in Transition internal_transition,
1042                      in UmlStateMachines::State new_state)
1043         raises (Reflective::StructuralError,
1044               Reflective::SemanticError,
1045               Reflective::NotFound);
1046     void modify_internal_transition (in UmlStateMachines::State state,
1047                                     in Transition internal_transition,
1048                                     in Transition new_internal_transition)
```

```
1049     raises (Reflective::StructuralError,
1050             Reflective::SemanticError,
1051             Reflective::NotFound);
1052 void remove (in UmlStateMachines::State state,
1053             in Transition internal_transition)
1054     raises (Reflective::StructuralError,
1055             Reflective::SemanticError,
1056             Reflective::NotFound);
1057 };
1058
1059 struct TransitionIsTriggeredByEventLink {
1060     UmlStateMachines::Transition transition;
1061     Event trigger;
1062 };
1063 typedef sequence <TransitionIsTriggeredByEventLink>
1064     TransitionIsTriggeredByEventLinkSet;
1065
1066 interface TransitionIsTriggeredByEvent : Reflective::RefAssociation {
1067     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
1068     TransitionIsTriggeredByEventLinkSet
1069     all_transition_is_triggered_by_event_links();
1070     boolean exists (in UmlStateMachines::Transition transition,
1071                   in Event trigger);
1072     UmlStateMachines::TransitionSet with_trigger (in Event trigger);
1073     Event with_transition (in UmlStateMachines::Transition transition);
1074     void add (in UmlStateMachines::Transition transition, in Event trigger)
1075         raises (Reflective::StructuralError, Reflective::SemanticError);
1076     void modify_transition (in UmlStateMachines::Transition transition,
1077                             in Event trigger,
1078                             in UmlStateMachines::Transition new_transition)
1079         raises (Reflective::StructuralError,
1080                 Reflective::SemanticError,
1081                 Reflective::NotFound);
1082     void modify_trigger (in UmlStateMachines::Transition transition,
1083                         in Event trigger,
1084                         in Event new_trigger)
1085         raises (Reflective::StructuralError,
1086                 Reflective::SemanticError,
1087                 Reflective::NotFound);
1088     void remove (in UmlStateMachines::Transition transition, in Event trigger)
1089         raises (Reflective::StructuralError,
```

```
1090         Reflective::SemanticError,
1091         Reflective::NotFound);
1092     };
1093
1094     struct StateMachineOwnsTransitionLink {
1095         UmlStateMachines::StateMachine state_machine;
1096         Transition transitions;
1097     };
1098     typedef sequence <StateMachineOwnsTransitionLink>
1099         StateMachineOwnsTransitionLinkSet;
1100
1101     interface StateMachineOwnsTransition : Reflective::RefAssociation {
1102         readonly attribute UmlStateMachinesPackage enclosing_package_ref;
1103         StateMachineOwnsTransitionLinkSet
1104             all_state_machine_owns_transition_links();
1105         boolean exists (in UmlStateMachines::StateMachine state_machine,
1106             in Transition transitions);
1107         UmlStateMachines::StateMachine with_transitions (
1108             in Transition transitions);
1109         TransitionSet with_state_machine (
1110             in UmlStateMachines::StateMachine state_machine);
1111         void add (in UmlStateMachines::StateMachine state_machine,
1112             in Transition transitions)
1113             raises (Reflective::StructuralError, Reflective::SemanticError);
1114         void modify_state_machine (
1115             in UmlStateMachines::StateMachine state_machine,
1116             in Transition transitions,
1117             in UmlStateMachines::StateMachine new_state_machine)
1118             raises (Reflective::StructuralError,
1119                 Reflective::SemanticError,
1120                 Reflective::NotFound);
1121         void modify_transitions (in UmlStateMachines::StateMachine state_machine,
1122             in Transition transitions,
1123             in Transition new_transitions)
1124             raises (Reflective::StructuralError,
1125                 Reflective::SemanticError,
1126                 Reflective::NotFound);
1127         void remove (in UmlStateMachines::StateMachine state_machine,
1128             in Transition transitions)
1129             raises (Reflective::StructuralError,
1130                 Reflective::SemanticError,
```



```
1131         Reflective::NotFound);
1132     };
1133
1134     struct TransitionHasSourceStateVertexLink {
1135         Transition outgoing;
1136         StateVertex source;
1137     };
1138     typedef sequence <TransitionHasSourceStateVertexLink>
1139         TransitionHasSourceStateVertexLinkSet;
1140
1141     interface TransitionHasSourceStateVertex : Reflective::RefAssociation {
1142         readonly attribute UmlStateMachinesPackage enclosing_package_ref;
1143         TransitionHasSourceStateVertexLinkSet
1144             all_transition_has_source_state_vertex_links();
1145         boolean exists (in Transition outgoing, in StateVertex source);
1146         TransitionSet with_source (in StateVertex source);
1147         StateVertex with_outgoing (in Transition outgoing);
1148         void add (in Transition outgoing, in StateVertex source)
1149             raises (Reflective::StructuralError, Reflective::SemanticError);
1150         void modify_outgoing (in Transition outgoing,
1151             in StateVertex source,
1152             in Transition new_outgoing)
1153             raises (Reflective::StructuralError,
1154                 Reflective::SemanticError,
1155                 Reflective::NotFound);
1156         void modify_source (in Transition outgoing,
1157             in StateVertex source,
1158             in StateVertex new_source)
1159             raises (Reflective::StructuralError,
1160                 Reflective::SemanticError,
1161                 Reflective::NotFound);
1162         void remove (in Transition outgoing, in StateVertex source)
1163             raises (Reflective::StructuralError,
1164                 Reflective::SemanticError,
1165                 Reflective::NotFound);
1166     };
1167
1168     struct TransitionHasTargetStateVertexLink {
1169         Transition incoming;
1170         StateVertex target;
1171     };
```

```
1172 typedef sequence <TransitionHasTargetStateVertexLink>
1173   TransitionHasTargetStateVertexLinkSet;
1174
1175 interface TransitionHasTargetStateVertex : Reflective::RefAssociation {
1176   readonly attribute UmlStateMachinesPackage enclosing_package_ref;
1177   TransitionHasTargetStateVertexLinkSet
1178     all_transition_has_target_state_vertex_links();
1179   boolean exists (in Transition incoming, in StateVertex target);
1180   TransitionSet with_target (in StateVertex target);
1181   StateVertex with_incoming (in Transition incoming);
1182   void add (in Transition incoming, in StateVertex target)
1183     raises (Reflective::StructuralError, Reflective::SemanticError);
1184   void modify_incoming (in Transition incoming,
1185     in StateVertex target,
1186     in Transition new_incoming)
1187     raises (Reflective::StructuralError,
1188     Reflective::SemanticError,
1189     Reflective::NotFound);
1190   void modify_target (in Transition incoming,
1191     in StateVertex target,
1192     in StateVertex new_target)
1193     raises (Reflective::StructuralError,
1194     Reflective::SemanticError,
1195     Reflective::NotFound);
1196   void remove (in Transition incoming, in StateVertex target)
1197     raises (Reflective::StructuralError,
1198     Reflective::SemanticError,
1199     Reflective::NotFound);
1200 };
1201
1202 struct SubmachineStateContainsStateMachineLink {
1203   StateMachine submachine;
1204   UmlStateMachines::SubmachineState submachine_state;
1205 };
1206 typedef sequence <SubmachineStateContainsStateMachineLink>
1207   SubmachineStateContainsStateMachineLinkSet;
1208
1209 interface SubmachineStateContainsStateMachine : Reflective::RefAssociation {
1210   readonly attribute UmlStateMachinesPackage enclosing_package_ref;
1211   SubmachineStateContainsStateMachineLinkSet
1212     all_submachine_state_contains_state_machine_links();
```

```

1213 boolean exists (in StateMachine submachine,
1214                 in UmlStateMachines::SubmachineState submachine_state);
1215 StateMachine with_submachine_state (
1216     in UmlStateMachines::SubmachineState submachine_state);
1217 UmlStateMachines::SubmachineStateSet with_submachine (
1218     in StateMachine submachine);
1219 void add (in StateMachine submachine,
1220          in UmlStateMachines::SubmachineState submachine_state)
1221     raises (Reflective::StructuralError, Reflective::SemanticError);
1222 void modify_submachine (
1223     in StateMachine submachine,
1224     in UmlStateMachines::SubmachineState submachine_state,
1225     in StateMachine new_submachine)
1226     raises (Reflective::StructuralError,
1227            Reflective::SemanticError,
1228            Reflective::NotFound);
1229 void modify_submachine_state (
1230     in StateMachine submachine,
1231     in UmlStateMachines::SubmachineState submachine_state,
1232     in UmlStateMachines::SubmachineState new_submachine_state)
1233     raises (Reflective::StructuralError,
1234            Reflective::SemanticError,
1235            Reflective::NotFound);
1236 void remove (in StateMachine submachine,
1237             in UmlStateMachines::SubmachineState submachine_state)
1238     raises (Reflective::StructuralError,
1239            Reflective::SemanticError,
1240            Reflective::NotFound);
1241 };
1242
1243 struct StateMachineExhibitsBehaviorOfModelElementLink {
1244     ::UmlCore::ModelElement uml_context;
1245     StateMachine behavior;
1246 };
1247 typedef sequence <StateMachineExhibitsBehaviorOfModelElementLink>
1248     StateMachineExhibitsBehaviorOfModelElementLinkSet;
1249
1250 interface StateMachineExhibitsBehaviorOfModelElement :
1251     Reflective::RefAssociation {
1252     readonly attribute UmlStateMachinesPackage enclosing_package_ref;
1253     StateMachineExhibitsBehaviorOfModelElementLinkSet

```

```
1254     all_state_machine_exhibits_behavior_of_model_element_links());
1255     boolean exists (in ::UmlCore::ModelElement uml_context,
1256                   in StateMachine behavior);
1257     ::UmlCore::ModelElement with_behavior (in StateMachine behavior);
1258     StateMachineSet with_uml_context (in ::UmlCore::ModelElement uml_context);
1259     void add (in ::UmlCore::ModelElement uml_context, in StateMachine behavior)
1260             raises (Reflective::StructuralError, Reflective::SemanticError);
1261     void modify_uml_context (in ::UmlCore::ModelElement uml_context,
1262                             in StateMachine behavior,
1263                             in ::UmlCore::ModelElement new_uml_context)
1264             raises (Reflective::StructuralError,
1265                   Reflective::SemanticError,
1266                   Reflective::NotFound);
1267     void modify_behavior (in ::UmlCore::ModelElement uml_context,
1268                          in StateMachine behavior,
1269                          in StateMachine new_behavior)
1270             raises (Reflective::StructuralError,
1271                   Reflective::SemanticError,
1272                   Reflective::NotFound);
1273     void remove (in ::UmlCore::ModelElement uml_context,
1274                in StateMachine behavior)
1275             raises (Reflective::StructuralError,
1276                   Reflective::SemanticError,
1277                   Reflective::NotFound);
1278 };
1279
1280 interface UmlStateMachinesPackageFactory {
1281     UmlStateMachinesPackage create_uml_state_machines_package ()
1282             raises (Reflective::SemanticError);
1283 };
1284
1285 interface UmlStateMachinesPackage : Reflective::RefPackage {
1286     readonly attribute EventClass event_class_ref;
1287     readonly attribute CallEventClass call_event_class_ref;
1288     readonly attribute ChangeEventClass change_event_class_ref;
1289     readonly attribute SignalEventClass signal_event_class_ref;
1290     readonly attribute TimeEventClass time_event_class_ref;
1291     readonly attribute StateVertexClass state_vertex_class_ref;
1292     readonly attribute GuardClass guard_class_ref;
1293     readonly attribute TransitionClass transition_class_ref;
1294     readonly attribute PseudostateClass pseudostate_class_ref;
```

1295   readonly attribute StateClass state\_class\_ref;  
1296   readonly attribute CompositeStateClass composite\_state\_class\_ref;  
1297   readonly attribute PartitionClass partition\_class\_ref;  
1298   readonly attribute ClassifierInStateClass classifier\_in\_state\_class\_ref;  
1299   readonly attribute StateMachineClass state\_machine\_class\_ref;  
1300   readonly attribute ActivityModelClass activity\_model\_class\_ref;  
1301   readonly attribute SimpleStateClass simple\_state\_class\_ref;  
1302   readonly attribute ActivityStateClass activity\_state\_class\_ref;  
1303   readonly attribute ObjectFlowStateClass object\_flow\_state\_class\_ref;  
1304   readonly attribute ActionStateClass action\_state\_class\_ref;  
1305   readonly attribute SubmachineStateClass submachine\_state\_class\_ref;  
1306  
1307   readonly attribute StateOwnsEntryActionSequence  
1308     state\_owns\_entry\_action\_sequence\_ref;  
1309   readonly attribute StateOwnsExitActionSequence  
1310     state\_owns\_exit\_action\_sequence\_ref;  
1311   readonly attribute TransitionOwnsGuard transition\_owns\_guard\_ref;  
1312   readonly attribute SignalEventIsOccurrenceOfSignal  
1313     signal\_event\_is\_occurrence\_of\_signal\_ref;  
1314   readonly attribute ActivityModelOwnsPartition  
1315     activity\_model\_owns\_partition\_ref;  
1316   readonly attribute PartitionHasContextOfModelElement  
1317     partition\_has\_context\_of\_model\_element\_ref;  
1318   readonly attribute ClassifierInStateIsInStateState  
1319     classifier\_in\_state\_is\_in\_state\_state\_ref;  
1320   readonly attribute ObjectFlowStateRepresentsClassifierInState  
1321     object\_flow\_state\_represents\_classifier\_in\_state\_ref;  
1322   readonly attribute ClassifierInStateIsOfTypeClassifier  
1323     classifier\_in\_state\_is\_of\_type\_classifier\_ref;  
1324   readonly attribute StateMachineOwnsTopState  
1325     state\_machine\_owns\_top\_state\_ref;  
1326   readonly attribute StateDefersEvent state\_defers\_event\_ref;  
1327   readonly attribute CallEventIsOccurrenceOfOperation  
1328     call\_event\_is\_occurrence\_of\_operation\_ref;  
1329   readonly attribute CompositeStateOwnsSubstateStateVertex  
1330     composite\_state\_owns\_substate\_state\_vertex\_ref;  
1331   readonly attribute TransitionOwnsEffectActionSequence  
1332     transition\_owns\_effect\_action\_sequence\_ref;  
1333   readonly attribute StateOwnsInternalTransition  
1334     state\_owns\_internal\_transition\_ref;  
1335   readonly attribute TransitionIsTriggeredByEvent

```

1336     transition_is_triggered_by_event_ref;
1337     readonly attribute StateMachineOwnsTransition
1338     state_machine_owns_transition_ref;
1339     readonly attribute TransitionHasSourceStateVertex
1340     transition_has_source_state_vertex_ref;
1341     readonly attribute TransitionHasTargetStateVertex
1342     transition_has_target_state_vertex_ref;
1343     readonly attribute SubmachineStateContainsStateMachine
1344     submachine_state_contains_state_machine_ref;
1345     readonly attribute StateMachineExhibitsBehaviorOfModelElement
1346     state_machine_exhibits_behavior_of_model_element_ref;
1347 };
1348 };

```

#### 5.4.7 UMLUseCases

```

1 #include "UmlCommonBehavior.idl"
2
3 module UmlUseCases {
4     interface UmlUseCasesPackage;
5     interface ExtensionPoint;
6     interface ExtensionPointClass;
7     typedef sequence<ExtensionPoint> ExtensionPointUList;
8     typedef sequence<ExtensionPoint> ExtensionPointSet;
9     interface UseCase;
10    interface UseCaseClass;
11    typedef sequence<UseCase> UseCaseUList;
12    interface Actor;
13    interface ActorClass;
14    typedef sequence<Actor> ActorUList;
15    interface UseCaseInstance;
16    interface UseCaseInstanceClass;
17    typedef sequence<UseCaseInstance> UseCaseInstanceUList;
18
19    interface ActorClass : ::UmlCore::ClassifierClass {
20        readonly attribute ActorUList all_of_kind_actor;
21        readonly attribute ActorUList all_of_type_actor;
22        Actor create_actor (in ::UmlCore::Name name,
23                            in boolean is_root,
24                            in boolean is_leaf,
25                            in boolean is_abstract)

```

```

26     raises (Reflective::SemanticError);
27 };
28
29 interface Actor : ActorClass, ::UmlCore::Classifier { };
30
31 interface UseCaseClass : ::UmlCore::ClassifierClass {
32     readonly attribute UseCaseUList all_of_kind_use_case;
33     readonly attribute UseCaseUList all_of_type_use_case;
34     UseCase create_use_case (in ::UmlCore::Name name,
35                             in boolean is_root,
36                             in boolean is_leaf,
37                             in boolean is_abstract)
38     raises (Reflective::SemanticError);
39 };
40
41 interface UseCase : UseCaseClass, ::UmlCore::Classifier {
42     UmlUseCases::ExtensionPointSet extension_point ()
43     raises (Reflective::NotSet, Reflective::SemanticError);
44     void add_extension_point (in UmlUseCases::ExtensionPointSet new_value)
45     raises (Reflective::StructuralError, Reflective::SemanticError);
46     void remove_extension_point ()
47     raises (Reflective::SemanticError);
48 };
49
50 interface UseCaseInstanceClass : ::UmlCommonBehavior::UmlInstanceClass {
51     readonly attribute UseCaseInstanceUList all_of_kind_use_case_instance;
52     readonly attribute UseCaseInstanceUList all_of_type_use_case_instance;
53     UseCaseInstance create_use_case_instance (in ::UmlCore::Name name)
54     raises (Reflective::SemanticError);
55 };
56
57 interface UseCaseInstance : UseCaseInstanceClass,
58                             ::UmlCommonBehavior::UmlInstance { };
59
60 interface ExtensionPointClass : ::UmlCore::ModelElementClass {
61     readonly attribute ExtensionPointUList all_of_kind_extension_point;
62     readonly attribute ExtensionPointUList all_of_type_extension_point;
63     ExtensionPoint create_extension_point (in ::UmlCore::Name name,
64                                           in ::UmlCore::Expression body)
65     raises (Reflective::SemanticError);
66 };

```

```
67
68 interface ExtensionPoint : ExtensionPointClass, ::UmlCore::ModelElement {
69     ::UmlCore::Expression body ()
70     raises (Reflective::SemanticError);
71     void set_body (in ::UmlCore::Expression new_value)
72     raises (Reflective::SemanticError);
73     UmlUseCases::UseCase use_case ()
74     raises (Reflective::NotSet, Reflective::SemanticError);
75     void set_use_case (in UmlUseCases::UseCase new_value)
76     raises (Reflective::SemanticError);
77     void unset_use_case ()
78     raises (Reflective::SemanticError);
79 };
80
81 struct UseCaseOwnsExtensionPointLink {
82     UmlUseCases::UseCase use_case;
83     UmlUseCases::ExtensionPoint extension_point;
84 };
85 typedef sequence <UseCaseOwnsExtensionPointLink>
86     UseCaseOwnsExtensionPointLinkSet;
87
88 interface UseCaseOwnsExtensionPoint : Reflective::RefAssociation {
89     readonly attribute UmlUseCasesPackage enclosing_package_ref;
90     UseCaseOwnsExtensionPointLinkSet all_use_case_owns_extension_point_links();
91     boolean exists (in UmlUseCases::UseCase use_case,
92         in UmlUseCases::ExtensionPoint extension_point);
93     UmlUseCases::UseCase with_extension_point (
94         in UmlUseCases::ExtensionPoint extension_point);
95     UmlUseCases::ExtensionPointSet with_use_case (
96         in UmlUseCases::UseCase use_case);
97     void add (in UmlUseCases::UseCase use_case,
98         in UmlUseCases::ExtensionPoint extension_point)
99     raises (Reflective::StructuralError, Reflective::SemanticError);
100     void modify_use_case (in UmlUseCases::UseCase use_case,
101         in UmlUseCases::ExtensionPoint extension_point,
102         in UmlUseCases::UseCase new_use_case)
103     raises (Reflective::StructuralError,
104         Reflective::SemanticError,
105         Reflective::NotFound);
106     void modify_extension_point (
107         in UmlUseCases::UseCase use_case,
```



```
108         in UmlUseCases::ExtensionPoint extension_point,
109         in UmlUseCases::ExtensionPoint new_extension_point)
110     raises (Reflective::StructuralError,
111           Reflective::SemanticError,
112           Reflective::NotFound);
113 void remove (in UmlUseCases::UseCase use_case,
114             in UmlUseCases::ExtensionPoint extension_point)
115     raises (Reflective::StructuralError,
116           Reflective::SemanticError,
117           Reflective::NotFound);
118 };
119
120 interface UmlUseCasesPackageFactory {
121     UmlUseCasesPackage create_uml_use_cases_package ()
122     raises (Reflective::SemanticError);
123 };
124
125 interface UmlUseCasesPackage : Reflective::RefPackage {
126     readonly attribute ActorClass actor_class_ref;
127     readonly attribute UseCaseClass use_case_class_ref;
128     readonly attribute UseCaseInstanceClass use_case_instance_class_ref;
129     readonly attribute ExtensionPointClass extension_point_class_ref;
130
131     readonly attribute UseCaseOwnsExtensionPoint
132     use_case_owns_extension_point_ref;
133 };
134 };
```

