

Error Model Meta Model and Plug-in

Peter Feiler
phf@sei.cmu.edu
May 28, 2007

The error model plug-in implements the Error Model Annex language extension. It provides all the front-end components, i.e., a parser, semantic checker, persistent save as XML file, and unparser (XML->text).

In addition, there is a library of methods to retrieve error model annex properties for given component instances, feature instances, and connection instances taking into account the applies to declarations. The library also has methods to get the error model for declarative model elements – in this case any applies to that associates an error model property with a subcomponent down the system hierarchy is ignored, since we do not have context information about the use of a given component implementation.

The Error Model Meta Model

The Error Model Meta Model defines the representation of Error Model library specifications and Error Model subclauses in the declarative AADL model. This is done through a set of related class specifications using the EMF Ecore notation. The Error Model Annex Meta Model is shown graphically in the figure at the end of this document.

The Error Model Meta Model does currently not define a representation of the error model in an AADL instance model. Instead a set of methods have been made available that allow you to access error model information in the context of instance model objects. This was done since the error model instance information is mostly identical to the error model information in the declarative AADL model – the primary exception is the need to resolve a name reference to out error propagations through incoming features in guard Boolean expression in the context of the instance model, which is supported through a lookup method (see below).

In the future we may include an instance model representation for error models as well in order to simplify processing of instance models with error model information independent of the model processing methods provided through OSATE – although those methods can be used stand-alone.

All classes in the Error Model Meta Model are subclasses of the class AObject, the root class of the AADL Meta Model. Named objects in the error model are subclasses of the PropertyHolder class or the property in the AADL Meta Model. This allows them to have a defining name and allows AADL properties to be associated with them. Examples

is such classes are ErrorState, ErrorEvent, ErrorPropagation. Abstract classes in the Meta model are tagged with an (A).

The Meta model defines attributes on classes that are used temporarily during parsing as well as attributes and reference associations that reflect the resolution of names to object references. The attributes used for parsing only are not stored persistently in XML. The following attributes are not persistent:

- ComponentErrorModelProperty [packageIdentifier, errorModelTypeIdentifier, errorModelImplementationIdentifier]
- ErrorTransition [originName, destinationName, transitionActionName, transitionTriggerName]
- OccurrenceProperty[eventOrPropagationName]
- ReportErrorProperty[stateAndPropagationList]
- ErrorStateMappingRule[componentErrorStateIdentifier]
- ErrorModelProperty[appliesToName]
- ErrorGuard[appliesToFeatureName]
- ModeTransitionGuard[appliesToFeatureContextName]
- ErrorSourceName[componentNameOrFeatureName]

Defining identifiers for objects are stored persistently, e.g., the defining name for error states, error events, and error propagations.

The “errorStateOrPropagationName” of ErrorSourceName is not resolved into a reference since the error model whose features are to be referenced is not identifiable in the declarative model. They can be identified in the instance model by retrieving the error model of the connection instance or connected component instance (see methods below) and finding the state or propagation (findErrorState(name) and findErrorPropagation(name)) in the error model (see find methods in ErrorModelClassifier).

For ErrorModelImplementation the name stored in the class is <typename>.<implname>. This is done the same way as for the ComponentImpl classes in the core language. Methods to retrieve the typename and the implementationname are available.

The ErrorSourceName class records the fact that there is a self reference by not recording a reference in subcomponentOrFeature. In other words, if this reference is null, the component itself is referenced.

Methods for Accessing Error Model Properties

The classes defined in the Error Model Meta Model come with a number of methods that set and retrieve attributes and reference associations. In addition, we have defined a number of methods that support the retrieval of error model properties in declarative AADL models and in AADL instance models.

The retrieval methods for declarative AADL models are more limited as they do not have the use context of the component types and implementations for which error models are associated and guards defined. One reason is that the applies to clause allows error model properties to be specified in a component error model annex for its subcomponents or their subcomponents. Another reason is that error propagation processing requires knowledge of the connection instance or component instance that is connected to an incoming port in order to resolve the name of an out error propagation or error state named in the square brackets of a guard Boolean expression. For example, `Guard_In => inProp1 when (inPort1[outProp1]` has `outProp1` refer to the out error propagation of the component (or connection if the connection has an error model) that is the originator of the connection coming in through `inPort1`.

All the described methods can be found in the package `package edu.erau.aadl.errorannex.util.EAXUtil`

Error Model Access in the AADL Instance Model

The following are methods that provide access to error model information in the AADL instance model. These are methods that retrieve the error model subclauses and libraries, the MODEL property, i.e., the error model associated with a component or connection, and methods for retrieving other error model properties that are to be retrieved relative to the MODEL value associated with a component or connections. Examples of the latter kind are Guards and Occurrence property values on error events or error propagations.

Note that some error model properties refer to element in an error model. The name resolver has resolved those references in the context of the error model that is associated with a component or connection through the MODEL property. Furthermore, error model properties can be declared for subcomponents in the system hierarchy. In those cases their references into an error model must be resolved relative to the error model associated with that component.

Retrieval of Error Model Subclauses and Libraries

Methods for retrieving the error model subclause from component types and component implementations, or the error model library from packages. These are convenience methods.

```
/**
 * get the ComponentErrorModelProperty associated with a component
 * instance
 * @param ci ComponentInstance
 * @return ComponentErrorModelProperty
 */
public ComponentErrorModelProperty
getComponentErrorModelProperty(ComponentInstance ci)

/**
 * get the ComponentErrorModelProperty associated with a component
```

```

* instance
* @param ci ComponentInstance
* @return ComponentErrorModelProperty
*/
public ComponentErrorModelProperty
getComponentErrorModelProperty(ConnectionInstance conni)

```

Retrieval of the MODEL property

The following are methods for retrieving the MODEL property and its value, namely the reference to an error model, that is associated with a component instance or connection instance. These methods take into account the applies to clauses that may associate an error model property with a component down the system hierarchy.

The first set of methods retrieve the MODEL property object – the class being called ComponentErrorModelProperty.

```

/**
* get the ComponentErrorModelProperty associated with a component
* instance.
* @param ci Component Instance
* @return ComponentErrorModelProperty
*/

/**
* get the ComponentErrorModelProperty associated with a Connection
* Instance
* @param conni Connection Instance
* @return ComponentErrorModelProperty
*/
public ComponentErrorModelProperty
getComponentErrorModelProperty(ConnectionInstance conni)

```

The next set of methods retrieves the actual reference value of the MODEL property, i.e., the reference to an error model type or error model implementation (class ErrorModelClassifier).

```

public ErrorModelClassifier getComponentErrorModelPropertyValue(
ComponentInstance ci )

public ErrorModelClassifier
getComponentErrorModelPropertyValue(ConnectionInstance conni)

```

Retrieval of Other Error Model Properties

These retrieval methods are sensitive to retrieving the property resolved to the correct error model. Each method exist for retrieval of the error model property associated with a component instance and with a connection instance. The following methods take either ComponentInstance or ConnectionInstance as parameter.

- `getModelHierarchyProperty`: retrieval of the `ModelHierarchyProperty` object,
- `getReportErrorProperty`: retrieval of the `ReportErrorProperty` object,
- `getDerivedErrorStateMappingGuard`: retrieval of the `DerivedErrorStateMappingGuard` object.

The next set of methods take a component instance or connection instance as first parameter and take a second parameter to provide additional context information.

- `getInErrorPropagationGuard(ComponentInstance ci, Feature f)`: retrieval of the `InErrorPropagationGuard` object that is associated with the feature (port, access) `f` of the specified component instance `ci`,
- `getInErrorPropagationGuard(FeatureInstance fi)`: retrieval of the `InErrorPropagationGuard` object that is associated with the feature instance `fi`,
- `getOutErrorPropagationGuard(ComponentInstance ci, Feature f)`: retrieval of the `OutErrorPropagationGuard` object that is associated with the feature (port, access) `f` of the specified component instance `ci`,
- `getOutErrorPropagationGuard(FeatureInstance fi)`: retrieval of the `OutErrorPropagationGuard` object that is associated with the feature instance `fi`,
- `getPortEventGuard(ComponentInstance ci, EventPort ep)`: retrieval of the `PortEventGuard` object that is associated with the `EventPort` `ep` of the specified component instance `ci`,
- `getPortEventGuard(FeatureInstance fi, EventPort ep)`: retrieval of the `PortEventGuard` object that is associated with the `EventPort` `ep` of the specified component instance `ci`,
- `getModeTransitionGuard(ComponentInstance ci, EventPort feature, PropertyHolder featureContext)`: retrieval of the `ModeTransitionGuard` object in the component instance `ci` by the `Feature` `f` with context `featureContext` (`Subcomponent` or `ComponentImpl`) for which the guard is defined,
- `getModeTransitionGuard(ComponentInstance ci, FeatureInstance fi)`: retrieval of the `ModeTransitionGuard` object in the component instance `ci` by the `FeatureInstance` `fi` for which the guard is defined,
- `getOccurrenceProperty(ComponentInstance ci, ErrorEventOrPropagation ep)`: retrieval of `OccurrenceProperty` object for the specified error event or error propagation associated with component instance `ci`.

Retrieval of `ErrorPropagationRule` or `ErrorStateMappingRule`

The following methods allow the user to retrieve a specific guard rule for an error propagation or an error state from a given guard object.

Method in `ErrorGuard` class

```
/**
 * returns the error propagation rule for the specified in/out error
 * propagation guard
 * @param prop ErrorPropagation
 * @return ErrorPropagationRule
 */
```

```
public ErrorPropagationRule getErrorPropagationRule(ErrorPropagation
prop)
```

Method in DerivedErrorStateMappingGuard class

```
/**
 * returns the mapping rule for the specified error state
 * @param state ErrorState
 * @return ErrorStateMappingRule
 */
public ErrorStateMappingRule getErrorStateMappingRule(ErrorState
state);
```

Resolving Error Propagation or Error State Names

The guard conditions may name an error propagation or an error state in the error model of the connection or connected component. These names can be resolved in the context of the instance model. The following methods allow the user to resolve those name by looking them up in the respective error model (ErrorModelClassifier):

```
/**
 * find ErrorPropagation in the error model. This method works on
 * error model types and implementations
 * @param epname ErrorPropagation name
 * @return ErrorPropagation
 */
public ErrorPropagation findErrorPropagation(String epname);

/**
 * find ErrorState in the error model. This method works on
 * error model types and implementations
 * @param epname ErrorState name
 * @return ErrorState
 */
public ErrorState find ErrorState(String epname);
```

Following Instance Model Connections

In order to process error propagation users are interested in finding components that are potentially affected by a given component. For that purpose the user is interested in finding components that are connected to a particular component. The FeatureInstance class provides a number of methods that return a list of connectioninstances. These methods include getOutgoingPortConnectionInstance for FeatureInstance objects, which returns the set of connections for which the feature instance is the source, getIncomingPortConnectionInstance for FeatureInstance objects, which returns the set of connections for which the feature instance is the destination, getDstAccessConnection for FeatureInstance objects that represent data or bus access objects, which returns the set of access connection instances whose source is a data/bus component instance, getSrcModeTransitionConnection for FeatureInstance objects that represent the event port instance triggering a mode transition, which returns the set of mode transition connection instances whose destination is a mode transition instance.

From such connection instances the user can either retrieve the error model associated with the connection or get hold of the component instance that is at the other end of the connection.

Error Model Access in the Declarative AADL Model

The following are methods that provide access to error model information in the declarative AADL model.

Retrieval of Error Model Subclauses and Libraries

Methods for retrieving the error model subclause from component types and component implementations, or the error model library from packages. These are convenience methods.

```
/**
 * Get error annex subclause for a given component classifier
 * @param root
 * @return ErrorAnnexSubClause
 */
public ErrorAnnexSubClause getErrorAnnexSubclause(ComponentClassifier
root)

/**
 * find error annex library for given package
 * @param aadlpackagename
 * @return ErrorAnnexLibrary
 */
public ErrorAnnexLibrary getErrorAnnexLibrary(String aadlpackagename)

/**
 * find the error annex library in an AadlSpec or AadlPackageSection
 * @param root AObject AadlSpec, AadlPackage or AadlPackageSection
 * @return ErrorAnnexLibrary
 */
public ErrorAnnexLibrary getErrorAnnexLibrary(AObject root){
```

Retrieval of the MODEL property

The following are methods for retrieving the MODEL property and its value, namely the reference to an error model, that is associated with a component type or implementation, or connection. These methods ignore any applies to whose role is to associate error model properties with subcomponents or their subcomponents.

The first set of methods retrieve the MODEL property object – the class being called ComponentErrorModelProperty.

```
/**
 * get the ComponentErrorModelProperty associated with a component
 * classifier. The ComponentErrorModelProperty is assumed to be
 * declared in the component, not via applies to
```

```

* @param cc Component classifier
* @return ComponentErrorModelProperty
*/
public ComponentErrorModelProperty GetComponentErrorModelProperty(
ComponentClassifier cc )

/**
* get the ComponentErrorModelProperty associated with a subcomponent
* The ComponentErrorModelProperty is assumed to be declared in
* the component, not via applies to
* @param sub subcomponent
* @return ComponentErrorModelProperty
*/
public ComponentErrorModelProperty
GetComponentErrorModelProperty(Subcomponent sub)

/**
* get the ComponentErrorModelProperty associated with a Connection
* The ComponentErrorModelProperty is assumed to be declared in
* the component, not via applies to
* @param conn Connection
* @return ComponentErrorModelProperty
*/
public ComponentErrorModelProperty
GetComponentErrorModelProperty(Connection conn)

```

The next set of methods retrieves the actual reference value of the MODEL property, i.e., the reference to an error model type or error model implementation (class `ErrorModelClassifier`).

```

public ErrorModelClassifier GetComponentErrorModelPropertyValue(
ComponentClassifier cc )

public ErrorModelClassifier
GetComponentErrorModelPropertyValue(Subcomponent sub)

public ErrorModelClassifier
GetComponentErrorModelPropertyValue(Connection conn)

```

Retrieval of Other Error Model Properties

We have defined an equivalent set of methods to the ones for the instance model for retrieving other error model properties. Note, however, that these methods may not be useful in the context of the declarative model as context information is missing. They are only safely usable on a declarative model if the applies to clauses to not use subcomponent paths.

May 28, 2007: Added reference from ModeTransitionGuard to a mode transition. This is to support the ability to refer to a mode transition by name. Mode transitions were not named in AADL V1, but this capability has been added to OSATE V1.5.

Previously the reference to the mode transition was implicitly expressed by referring to the event ports named by a mode transition. This did not provide a unique identification of mode transitions under some circumstances.