European Component Oriented Architecture (ECOA®) Collaboration Programme: Guidance Document: Insertion policies

Date: 29/08/2017

Prepared by
Dassault Aviation

Contents

Figures

Tables

0 Executive Summary

This document defines guidance for specifying ECOA insertion policies regarding ASC implementations.

At design time, the aim of insertion policies is for the system designer to express ASCs' requirements, so as to be able to integrate them. For instance, this deals with resource allocations.

The aim of insertion policies is also to facilitate the characterisation and the reuse of existing ASCs onto different platforms, and more generally, in different host environments.

From this perspective, insertion policies provide useful information about the conditions (target environments and usage domains) in which ASCs have been previously integrated. Insertion policies can also be used by an ASC supplier to indicate minimum target environment requirements, regardless of any previous actual integration. These minimum target environments may help the System Integrator to early verify the deployment.

1 Scope

This document is intended to provide guidance on container level checking and time synchronization.

The document is structured as follows:

Section 2 gives a brief introduction to the topic.

Section 3 expands abbreviations used in this report.

Section 4 provides definitions for the key terms used in this report.

Section 5 lists key documents referenced by this report.

Section 6 discusses the guidance to insertion policies.

2 Introduction

This document defines guidance for specifying ECOA insertion policies regarding ASC implementations.

In ECOA, insertion policies relate to the guidance area, meaning that the ECOA standard does not impose any specific formal description of insertion policies.

However, it is believed that insertion policies would be a good practice for facilitating and promoting ASCs reuse, typically as part of an ASC detailed profile card in a library of reusable ASCs. Insertion policies relate to the expected quality of service (QoS) of ASCs, and indicate conditions for achieving these expected QoS (either theoretically or already proven conditions on a real programme).

This purpose of this document is to propose a list of insertion policies deemed to be potentially useful both for ASC suppliers and System Integrators, at design time and reuse time. The document also proposes a formal metamodel for capturing these insertion policies.

This document does not cover technical information regarding ECOA compatible computing platforms. This is covered by the concept of logical system (see [Ref. AS]).

The contents of the document is the following:

- Section 6 provides an abstract view of the technical insertion policy and how this item is related to other component-related items.
- Appendix A provides the concrete XSD metamodel to describe the technical insertion policy of a component.
- Appendix B provides the concrete XSD metamodel to describe the temporal behavior of module entry points.

3 Abbreviations

API	Application Programming Interface
ASC	Application Software Component
DSTL	Defence Science and Technology Laboratory
ECOA	European Component Oriented Architecture
ELI	ECOA Logical Interface
FR	French
IAWG	Industrial Avionics Working Group
I/O	Inputs-Outputs
OS	Operating System
PF	Platform
QoS	Quality of Service
RR	Request-Response
STD	Standard
TR	Technical Report
TRL	Technology Readiness Level
UDP	User Datagram Protocol
UK	United Kingdom
XML	eXtensible Markup Language

4 Definitions

For the purpose of this document, the definitions given in the ECOA Architecture Specification *(ref. [AS])* Part 2 and those given below apply.

Term	***Definition***
(currently none)	

5 References

AS	European Component Oriented Architecture (ECOA) Collaboration Programme: Architecture Specification (Parts 1 to 11) "ECOA" is a registered mark.

6 Guidance to Technical Insertion Policy

For developing, assessing (coarse grain scheduling analysis, safety, security, etc.) and technically using (deployment, etc.) component implementations, several kinds of information are required.

They are defined and refined during the component development lifecycle. To explain this process, the following main stages are considered: specification, development, deployment, qualification.

In particular, they are captured by the following artefacts:

- The component specification, described in section 6.1,
- The component implementation, i.e. the logical architecture of the component (comp.impl.xml), described in [Ref: AS],
- The Technical Insertion Policy, i.e. the resource requirements, described in section 6.2,
- Information about assurances, described in [Ref. SSS],
- The Qualification Data, described in section 6.3.

This information is exchanged at least between the system designer, the component supplier and the system integrator. The following figure displays a possible flow between them.

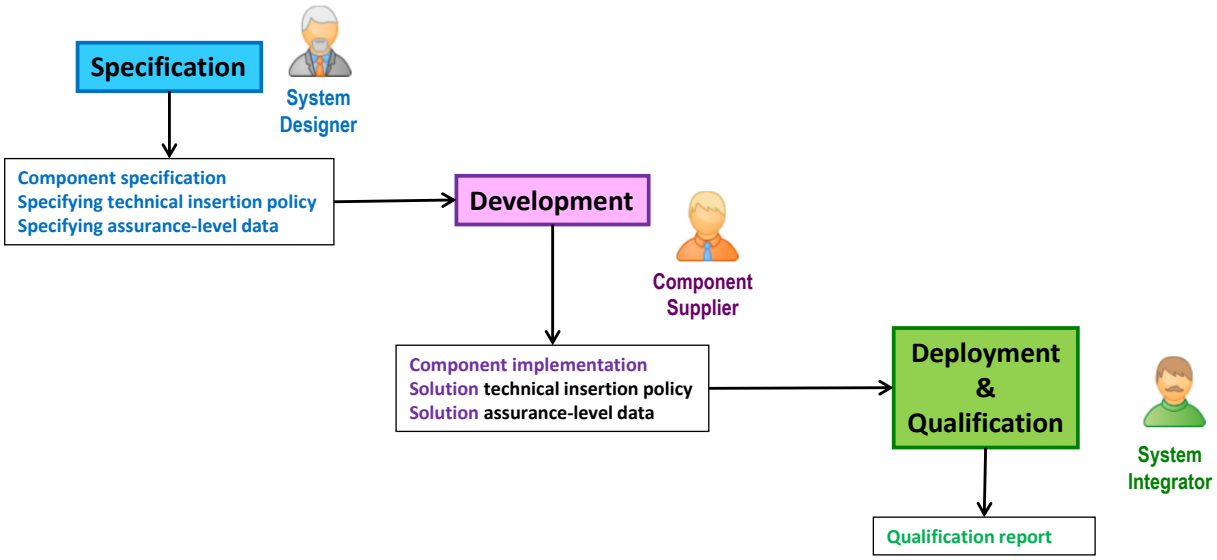

Figure 1 – Raw Flow of Information between component development steps

Additional stakeholders may also request access to the information but they are not considered in this report.

Note that part of the component supplier inputs may be refined during the development of the component. For instance, the component implementation may provide a better QoS for the provided services. By convention, in this document, for the specification elements that can be refined, input

elements to the development are the specifying ones while the output elements of the development are the solution ones.

6.1 Component Specification

The component specification needs to address all usual concerns of a software item: the interfaces, the functional behaviour, the non-functional requirements and the insertion constraints. These concerns can then be concretely stored in many ways.

From an ECOA concepts point of view, the static aspect (signature) of the interfaces is described in XML with the help of Type Definitions, Service Definitions and the Component Definition. The dynamic aspect (behaviour, QoS) of the interfaces is described in XML with the help of the service QoS (abstract and concrete models in §8.1 and §8.2 of the Architecture Specification, [Ref: AS]), the service behaviour and the component behaviour.

Behaviours can be described with tools like state charts, sequence diagrams, etc.

Behavioural requirements and non-behavioural requirements (properties of the system and the system lifecycle including development) can be described in several ways: informal text, textual requirements, models such as UML, formal notations such as Z or B, etc. Formalisms are left open and best practices can be used at this level. For the sake of this report, this part is called the “SRS” (Software Requirement Specification).

In addition to the component specification, insertion constraints may be provided through a partial technical insertion policy called the **specifying technical insertion policy** depending on system design requirements. By example, the system designer may request a given scheduling policy or may limit the amount of memory used by the component. In the same manner, the system designer may request a given set of assurance-level data.

6.2 Technical Insertion Policy Detailed Definition

The Technical Insertion Policy contains information or links to external information to constrain a development (specifying technical insertion policy) or to allow the technical insertion on top of an ECOA platform (solution technical insertion policy).

A given Technical Insertion Policy is valid for one given component implementation solution (a given version of a component implementation for one given platform). So, for a given component implementation, multiple Technical Insertion Policies may exist depending on the number of targeted platforms.

The table below lists all elements that may be included in a solution technical insertion policy; a specifying technical insertion policy may only be a subset depending on system design requirements. Some elements are relevant at component-level while other only concern modules. The exact content and the actual location of the information are described in Section 7.

Item	SubItem	SubSubItem	Level of Scope	*	Values	Comments / Actual location
target			component	1	Text	
processorType			component	1	Text	component **bin-desc** file
registerSize			component	1	32|64 bits	
memoryUsage						By comparing memory usage it may be possible to provide an early indication of whether there are sufficient memory resources on the target.
	userContextSize		module	1	XXX Bytes	component **bin-desc** file
	warmStartContextSize		module	1	XXX Bytes	component **bin-desc** file
	dynamicMemoryAllocation		module	1	yes|no (default = no)	
	maxDynamicMemorySize		module	1	XXX Bytes	if dynamicMemoryAllocation == yes
	stackSize		module	1	XXX Bytes	component **bin-desc** file

	heapSize		module	1	XXX Bytes	component **bin-desc** file
realtimeCharacteristics						
	schedulingPolicy		component	1	RT-FIFO|ROUND-ROBIN|ARINC-653	Scheduling policy against which the component has been validated
	In case of module-level scheduling policy					
		moduleActivationType	module	1	reactive|periodic|other	Useful for module-level scheduling
		deadline	module	1	number of computing steps	Information required for scheduling.
		rate	module	1	Highest activation rate of an aperiodic module	Information required for scheduling.
		period	module	1	number of computing steps between two activations of a periodic module	Information required for scheduling
		wcet	module	1	number of computing steps	Information required for scheduling. They can be computed by exploring the associated **module.behaviour.xml** – see proposal in section 8.
		relativePriority	module	1	Number The most important module is the one with the lower number for its priority.	If two modules of the same component share the realtime characteristics, the relativePriority provides a hint about what to schedule first. This may be used to ensure execution order of modules is as required, for example if one module provides inputs to another it may be

					necessary to execute it first.
timeAccuracy		component	1	number of seconds	Maximum offset between time returned through timestamps and time API and a reference clock.
transportProtocol		component	1		To define expectations about transport protocol reliability
	quality	component	1	lossless|none	If the development is not robust against loss of data between modules,a lossless transport protocol is required.
	integrity	component	1	high|none	If the development, is not robust against loss of data integrityfor data transmitted between modules, a high integrity transport protocol is required.
	maximumLatency	component	1	Time in seconds	Maximum latency required to exchange events/requests/responses between any pair of the component modules
	minimumBandwitdh	component	1	Kilobytes / second	Minimum bandwidth required to exchange data between any pair of the component modules
deploymentConstraints		component	1		
	uniqueProtectionDomain		1	yes|no	Requirement that indicates that modules of the ASC must be deployed into the same protection domain, to allow minimizing exchange latencies within the same protection domain. It can be useful to know this information

					for unit testing on the target
	deploymentExample		1	Link to a deployment file	The provided deployment file may be used as it is by the System Integrator. It should have been used during component testing
languageRuntime		module	1	C|C++|Ada	moduleImplementation in **comp.impl.xml**
languageLibraries		module			List of libraries required by the ASC to compile and execute. This allows knowing dependencies. For instance, an ASC containing matlab/Simulink generated source code shall declare dependencies to any needed Matlab/Simunlink RT libraries.
	For each library				
		libraryName	module	* Text	
		libraryType	module	* math|graphics|geodesic|other	
		libraryVersion	module	* Text	
osAPIAccess		module	1	yes|no (default=no)	Indicate if the module code calls directly the underlying OS API (i.e, the ASC is a Driver Component)
osAPIType		module	1	POSIX|A653|ASAAC|FACE|other	if osAPIAccess == yes
compiler		component	1	Text	The identifying name of the compiler
genericCompilerOptions		component	1	text. E.g. "-fpic"	Applicable to all modules
specificCompilerOptions		module	1	text. E.g. "-x"	Specific to one module implementation binary

extraConcerns		component	1	ad-hoc document	Elements that cannot be covered under a generic form For example, it may explain for which reasons/causes the component raises fatal errors towards to the fault handler. This information may be used by the System Integrator to adapt the behavior of the Fault Handler if needed.
defaultPINFO					List of public and private PINFO used for the validation of the component
	publicPINFO	module	*	list of filenames	
	privatePINFO	module	*	list of filenames	
defaultProperties					
	property	component	*	Text	Default values of component-level properties used for the validation of the component and the system integrator may reuse without any question

Table 1 –Technical Insertion Policy

The component supplier should mention whether his solution insertion policy has been proven on a programme (i.e. actually measured in a given known application), or if it indicates theoretical requirements (i.e. extimations/predictions/expections). The element **target** allows to register such mention.

This may be specified in the ASC profile card.

6.3 Qualification Data

The Qualification Data summarise results of acceptance tests run by the System Integrator and possibly the System Designer.

There is one set of Qualification Data per version of component implementation solution.

The exact content of Qualification Data is not prescribed by the ECOA Architecture Specifications.

6.4 Overview of Insertion policies with regard to Component Development Steps

Figure 2 refines Figure 1 and shows, from a component supplier point of view, the main flow of information between development steps. It defines for each step what information is created or refined. In particular, the system designer may constraint the development by providing as input a specifying technical insertion policy. The component supplier will provide the result of his refinement as the solution technical insertion policy. Figure 2 does not preclude any other information flow schema and information names used in the figure are examples for the sake of the explanation.

In addition to the information described above, additional data are made available to ease the acceptance and the deployment of the component. A test harness may also be provided by the component supplier to the system integrator to help the unit acceptance of the component on the target platform.

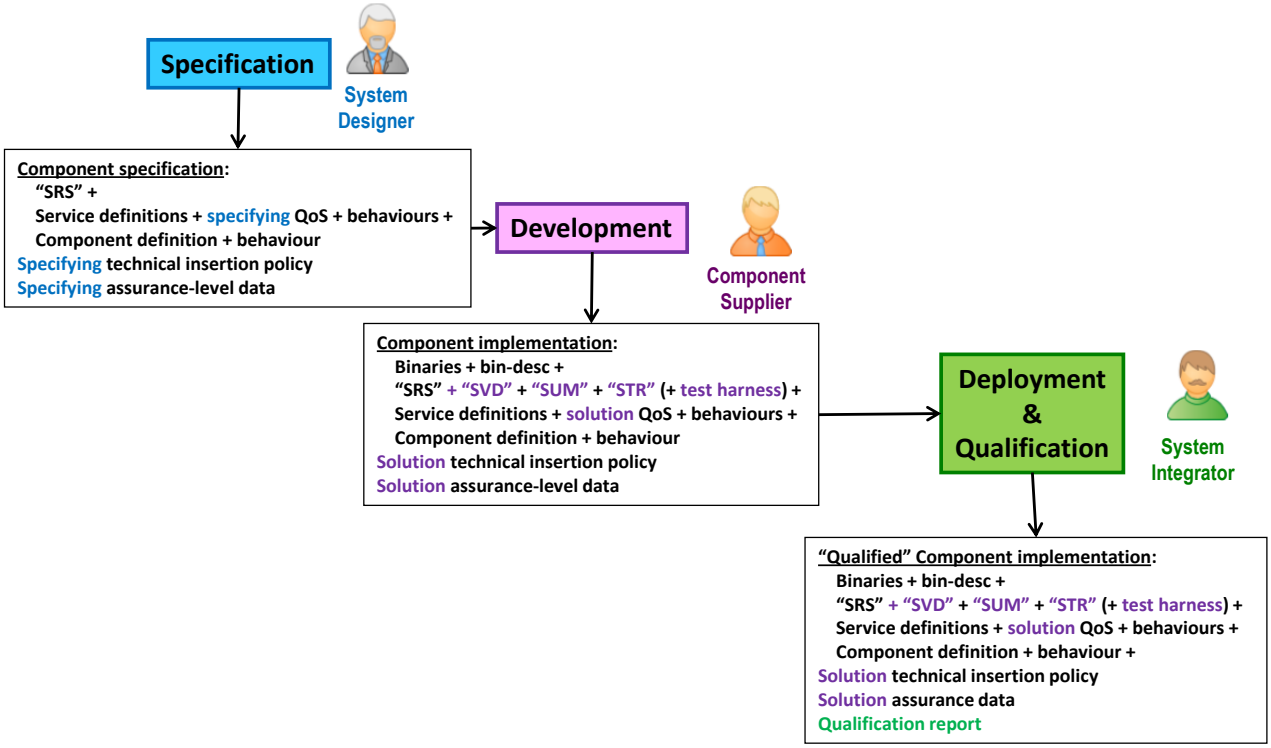

Figure 2 – Flow of Information between component development steps

In the figure above, "SRS", "SUM", "SVD" and "STR" stand respectively for:

- Software Requirement Specification – it describes behavioural and non-behavioural requirements,
- Software User Manual – it describes how to use/instantiate the component within a composite. In particular, it describes property values and PINFO contents required to the good execution of the component,
- Software Version Description – it recaps all items describing a component implementation

- and Software Test Report – it synthetizes results of tests carried on the component.

The exact content of these documents is not prescribed by the ECOA Architecture Specifications.

7 Appendix A – Technical Insertion Policy XML

7.1 Technical Insertion Policy XSD

This section provides an XSD implementing table 1. For information already defined in other XML files such as bin-desc or comp.impl.xml, it is proposed to define an empty XML element with the insertion policy XML but characterized with an xlink+xpointer value.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<xsd:schema xmlns="http://www.ecoa.technology/insertion-policy-2.0"
  xmlns:tns="http://www.ecoa.technology/insertion-policy-2.0"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
targetNamespace="http://www.ecoa.technology/insertion-policy-2.0">

  <!-- Insertion policies for an ECOA binary component -->

  <xsd:include schemaLocation="../ecoa-common-2.0.xsd"/>

  <xsd:element name="insertionPolicies" type="InsertionPolicyList">
    <xsd:annotation>
      <xsd:documentation>Defines the list of insertion policies for
        which the
        component was validated
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>

  <xsd:complexType name="InsertionPolicyList">
    <xsd:sequence>
      <xsd:element maxOccurs="unbounded" minOccurs="1" name="insertionPolicy"
        type="InsertionPolicy">
        <xsd:annotation>
          <xsd:documentation>Insertion policy entry
          </xsd:documentation>
        </xsd:annotation>
```

```
    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="componentImplementation" type="NameId"
    use="required">
    <xsd:annotation>
      <xsd:documentation>Name of the component implementation
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
</xsd:complexType>

<xsd:complexType name="InsertionPolicy">
  <xsd:annotation>
    <xsd:documentation>A consistent set of policies for which the
      component was
      validated, which means that its functional
      behaviour is correct and it
      fulfills its expected QoS
    </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="target" type="Target"
      minOccurs="0" maxOccurs="1">
      <xsd:annotation>
        <xsd:documentation>
          Targeted system on which the insertion policy applies to
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="processorTarget" minOccurs="1" maxOccurs="1">
      <xsd:annotation>
        <xsd:documentation>Processor Target
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="registerSize" type="RegisterSize"
      minOccurs="1" maxOccurs="1">
```

```
    <xsd:annotation>
      <xsd:documentation>
        Size of the processor register
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:element name="memoryUsage" type="MemoryUsage" minOccurs="1"
    maxOccurs="1">
    <xsd:annotation>
      <xsd:documentation>
        Information on how the component uses memory
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:element name="realtimeCharacteristics" type="RealtimeCharacteristics"
    minOccurs="1" maxOccurs="1">
    <xsd:annotation>
      <xsd:documentation>
        Information on scheduling, module activation
        profiles, etc.
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:element name="timeAccuracy" type="TimeAccuracy"
    minOccurs="0" maxOccurs="1">
    <xsd:annotation>
      <xsd:documentation>
        Time Accuracy
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:element name="transportProtocol" type="TransportProtocol"
    minOccurs="0" maxOccurs="1">
    <xsd:annotation>
      <xsd:documentation>
        Transport Protocol Requirements
      </xsd:documentation>
```

```
  </xsd:annotation>
</xsd:element>
<xsd:element name="deploymentConstraints" type="DeploymentConstraints"
  minOccurs="0" maxOccurs="1">
  <xsd:annotation>
    <xsd:documentation>
      Deployment Constraints
    </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="ECOAProfile" type="ECOAProfile" minOccurs="1"
  maxOccurs="1">
  <xsd:annotation>
    <xsd:documentation>
      ECOA Profile used for developing the component
    </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="compiler" type="Compiler" minOccurs="1"
  maxOccurs="1">
  <xsd:annotation>
    <xsd:documentation>
      Compiler used to generate the binaries
    </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="genericCompilationOptions" type="GenericCompilationOptions"
  minOccurs="0" maxOccurs="1">
  <xsd:annotation>
    <xsd:documentation>
      Compilation/Linkage options global to the component
    </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="modulesDependencies" type="ModulesDependencies"
  minOccurs="0" maxOccurs="1">
  <xsd:annotation>
```

```
        <xsd:documentation>
          Compilation/Linkage options specific to a module
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="extraConcerns" type="ExtraConcerns"
      minOccurs="0" maxOccurs="1">
      <xsd:annotation>
        <xsd:documentation>
          Any other concerns stored in a separate file
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="defaultPINFO" type="DefaultPINFO"
      minOccurs="0" maxOccurs="1">
      <xsd:annotation>
        <xsd:documentation>
          PINFO which were used to validate the component
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="defaultProperties" type="DefaultProperties"
      minOccurs="0" maxOccurs="1">
      <xsd:annotation>
        <xsd:documentation>
          Default property values against which the component
          has been validated
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

<!-- Register Size -->
<xsd:complexType name="RegisterSize">
  <xsd:attribute name="size" use="required">
    <xsd:annotation>
```

```
      <xsd:documentation>
        Register size in bits
      </xsd:documentation>
    </xsd:annotation>
    <xsd:simpleType>
      <xsd:restriction base="xsd:positiveInteger">
        <xsd:enumeration value="16"/>
        <xsd:enumeration value="32"/>
        <xsd:enumeration value="64"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>

<!-- Memory Usage -->
<xsd:complexType name="MemoryUsage">
  <xsd:sequence>
    <xsd:element name="moduleMemoryUsage" type="ModuleMemoryUsage"
      maxOccurs="unbounded">
      <xsd:annotation>
        <xsd:documentation>Module memory usage entry
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ModuleMemoryUsage">
  <xsd:annotation>
    <xsd:documentation>Information on module memory usage
    </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="userContextSize">
      <xsd:annotation>
        <xsd:documentation>User context size in bytes
        </xsd:documentation>
```

```
  </xsd:annotation>
</xsd:element>
<xsd:element name="warmStartContextSize">
  <xsd:annotation>
    <xsd:documentation>Warm start context size in bytes
    </xsd:documentation>
  </xsd:annotation>
</xsd:element>

<xsd:element name="dynamicMemory">
  <xsd:annotation>
    <xsd:documentation>Information about dynamic memory allocation
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:attribute name="allocation" use="required">
      <xsd:annotation>
        <xsd:documentation>Does the module allocate dynamically ?
        </xsd:documentation>
      </xsd:annotation>
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="yes"/>
          <xsd:enumeration value="no"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="maxSize" type="xsd:nonNegativeInteger"
      use="required">
      <xsd:annotation>
        <xsd:documentation>Max size of the dynamic memory in bytes
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
  </xsd:complexType>
</xsd:element>
<xsd:element name="stackSize">
```

```
      <xsd:annotation>
        <xsd:documentation>Max stack size in bytes
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="heapSize">
      <xsd:annotation>
        <xsd:documentation>Max heap size in bytes
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="instanceName" type="NameId" use="required">
    <xsd:annotation>
      <xsd:documentation>
        Reference to a module instance name
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
</xsd:complexType>


<!-- Realtime Characteristics -->
<xsd:complexType name="RealtimeCharacteristics">
  <xsd:sequence>
    <xsd:element name="schedulingPolicy">
      <xsd:annotation>
        <xsd:documentation>A scheduling policy for which the component
          was
          validated
        </xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
        <xsd:attribute name="policy" type="SchedulingPolicy"
          use="required">
          <xsd:annotation>
            <xsd:documentation>OS-level scheduling policy used to schedule
```

```
                modules
              </xsd:documentation>
            </xsd:annotation>
          </xsd:attribute>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="moduleActivationProfiles" type="ModuleActivationProfiles">
        <xsd:annotation>
          <xsd:documentation>Set of module activation profiles
          </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="ModuleActivationProfiles">
    <xsd:sequence>
      <xsd:element name="activationProfile" type="ModuleActivationProfile"
        minOccurs="1" maxOccurs="unbounded">
        <xsd:annotation>
          <xsd:documentation>Module activation profile entry
          </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="ModuleActivationProfile">
    <xsd:choice minOccurs="1" maxOccurs="1">
      <xsd:element name="periodic" type="PeriodicActivationProfile">
        <xsd:annotation>
          <xsd:documentation>To describe a periodic activation model
          </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="reactive" type="AperiodicActivationProfile">
        <xsd:annotation>
```

```
        <xsd:documentation>To describe an event-driven activation model
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:choice>
  <xsd:attribute name="instanceName" type="NameId" use="required">
    <xsd:annotation>
      <xsd:documentation>
        Reference to a module instance name
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
  <xsd:attribute name="activationType">
    <xsd:annotation>
      <xsd:documentation>Module activation type (event-driven, time-driven,
        other)
      </xsd:documentation>
    </xsd:annotation>
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="reactive"/>
        <xsd:enumeration value="periodic"/>
        <xsd:enumeration value="other"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute default="1" name="queueDepth" type="xsd:positiveInteger"
    use="optional">
    <xsd:annotation>
      <xsd:documentation>
        Depth of the incoming operations queue.
        This value has
        no impact of the code generation.
        It is only to store intermediate
        analysis result done by the
        component supplier.
      </xsd:documentation>
```

```
    </xsd:annotation>
  </xsd:attribute>
  <xsd:attribute default="1" name="maxNbOfProcessedOpsPerActivation"
    type="xsd:positiveInteger">
    <xsd:annotation>
      <xsd:documentation>
        Max number of processed operations per
        activation
        including the
        activating one.
        This value has no
        impact of the code
        generation.
        It is only to store intermediate
        analysis result done by the
        component supplier.
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
  <xsd:attribute name="behaviour" type="xsd:anyURI" use="optional">
    <xsd:annotation>
      <xsd:documentation>
        Link towards the module temporal behaviour
        xxx.behaviour.xml
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
</xsd:complexType>

<xsd:complexType name="PeriodicActivationProfile">
  <xsd:attribute name="deadline" type="Deadline">
    <xsd:annotation>
      <xsd:documentation>
        Deadline of the module in steps. It shall
        cover the
        processing of all non activating operations and the
        activating one
```

```
        which
        initiates the processing.
        Value to be taken into account by the
        system
        integrator.
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
  <xsd:attribute name="wcet" type="WCET">
    <xsd:annotation>
      <xsd:documentation>
        WCET of the module in steps. It shall cover
        the
        processing of all non activating operations and the
        activating one
        which
        initiates the processing.
        Value to be taken into account by the
        system
        integrator.
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
  <xsd:attribute name="period" type="Period">
    <xsd:annotation>
      <xsd:documentation>
        Period of the module in steps. Expected time
        between
        the
        arrival of two activating operations.
        Value to be taken into account
        by
        the
        system integrator.
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
```

```
    <xsd:attribute name="relativePriority" type="RelativePriority">
      <xsd:annotation>
        <xsd:documentation>
          Relative priority of this module instance to
          others
          module instances
          To help to distinguish several module
          instances sharing
          the same profile
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
  </xsd:complexType>

  <xsd:complexType name="AperiodicActivationProfile">
    <xsd:sequence>
      <xsd:element name="rate" type="HighestActivationRate"
        minOccurs="0" maxOccurs="1">
        <xsd:annotation>
          <xsd:documentation>
            Highest activation rate
          </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="deadline" type="Deadline">
      <xsd:annotation>
        <xsd:documentation>
          Deadline of the module in steps. It shall
          cover the
          processing of all non activating operations and the
          activating one
          which
          initiates the processing.
          Value to be taken into account by the
          system
          integrator.
```

```
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
  <xsd:attribute name="wcet" type="WCET">
    <xsd:annotation>
      <xsd:documentation>
        WCET of the module in steps. It shall cover
        the
        processing of all non activating operations and the
        activating one
        which
        initiates the processing.
        Value to be taken into account by the
        system
        integrator.
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
  <xsd:attribute name="relativePriority" type="RelativePriority">
    <xsd:annotation>
      <xsd:documentation>
        Relative priority of this module instance to
        others
        module instances
        To help to distinguish several module
        instances sharing
        the same profile
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
</xsd:complexType>

<xsd:simpleType name="Deadline">
  <xsd:restriction base="Steps"/>
</xsd:simpleType>

<xsd:simpleType name="WCET">
```

```
  <xsd:restriction base="Steps"/>
</xsd:simpleType>

<xsd:simpleType name="Period">
  <xsd:restriction base="Steps"/>
</xsd:simpleType>

<xsd:simpleType name="RelativePriority">
  <xsd:restriction base="xsd:nonNegativeInteger"/>
</xsd:simpleType>

<xsd:complexType name="HighestActivationRate">
  <xsd:attribute name="numberOfActivations" type="xsd:decimal"
    use="optional" default="1">
    <xsd:annotation>
      <xsd:documentation>Min or max number of activations occurring during a
        specified duration
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
  <xsd:attribute name="timeFrame" type="TimeDuration" use="optional">
    <xsd:annotation>
      <xsd:documentation>Equal to min or max inter-arrival time when
        NumberOfActivations value is 1.
        In other cases, specifies a sizing
        duration for activations
        bursts.
        Unit is second.
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
</xsd:complexType>

<!-- Time accuracy -->
<xsd:complexType name="TimeAccuracy">
  <xsd:attribute name="value" type="TimeDuration">
    <xsd:annotation>
```

```
      <xsd:documentation>
        Time accuracy in seconds
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
</xsd:complexType>

<!-- Transport Protocol Requirements -->
<xsd:complexType name="TransportProtocol">
  <xsd:attribute name="quality" default="lossless">
    <xsd:annotation>
      <xsd:documentation>
        Expected quality of the underlying transport protocol
      </xsd:documentation>
    </xsd:annotation>
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="lossless"/>
        <xsd:enumeration value="none"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="integrity" default="high">
    <xsd:annotation>
      <xsd:documentation>
        Expected integrity of the underlying transport
        protocol
      </xsd:documentation>
    </xsd:annotation>
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="high"/>
        <xsd:enumeration value="none"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="maxLatency" type="TimeDuration">
```

```
      <xsd:annotation>
        <xsd:documentation>
          Maximum latency for exchange of events between two
          modules.
          If the actual maximum latency is higher than the defined
          value, the component may not run correctly.
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="minBandwidth" type="xsd:nonNegativeInteger">
      <xsd:annotation>
        <xsd:documentation>
          Minimum bandwidth in kBytes/s for exchange of data
          between two modules.
          If the actual maximum bandwidth is lesser than the
          defined
          value, the component may not run correctly.
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
  </xsd:complexType>

  <!-- Deployment Constraints -->
  <xsd:complexType name="DeploymentConstraints">
    <xsd:attribute name="uniqueProtectionDomain" type="xsd:boolean">
      <xsd:annotation>
        <xsd:documentation>Do the modules need to be co-located in the same
          protection domain ?
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="deploymentExample" type="xsd:anyURI" use="optional">
      <xsd:annotation>
        <xsd:documentation>
          Link towards an external file showing the deployment
          used to validate the
          component
```

```
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
</xsd:complexType>

<!-- ECOA Profile -->
<xsd:complexType name="ECOAProfile">
  <xsd:attribute name="profile" use="required">
    <xsd:annotation>
      <xsd:documentation>
        ECOA Profile
      </xsd:documentation>
    </xsd:annotation>
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="core"/>
        <xsd:enumeration value="extended"/>
        <xsd:enumeration value="other"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>

<!-- Module Dependencies -->
<xsd:complexType name="ModulesDependencies">
  <xsd:sequence>
    <xsd:element name="moduleDependencies" type="ModuleDependencies"
      minOccurs="1" maxOccurs="unbounded">
      <xsd:annotation>
        <xsd:documentation>Module dependencies entry
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ModuleDependencies">
```

```
<xsd:sequence>
  <xsd:element name="languageRuntime" minOccurs="1" maxOccurs="1">
    <xsd:annotation>
      <xsd:documentation>Programming language used for developing the module
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:element name="languageLibraries" type="LanguageLibraries"
    minOccurs="0" maxOccurs="1">
    <xsd:annotation>
      <xsd:documentation>Libraries
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:element name="directOSAccess" type="OSAPIType" minOccurs="0"
    maxOccurs="unbounded">
    <xsd:annotation>
      <xsd:documentation>Underlying OS Standard
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:element name="moduleCompilationOptions" type="ModuleCompilationOptions"
    minOccurs="0" maxOccurs="1">
    <xsd:annotation>
      <xsd:documentation>
        Module specific compilation options
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
</xsd:sequence>
<xsd:attribute name="instanceName" type="NameId" use="required">
  <xsd:annotation>
    <xsd:documentation>
      Reference to the module instance
    </xsd:documentation>
  </xsd:annotation>
</xsd:attribute>
```

```
</xsd:complexType>


<!-- Language Libraries -->
<xsd:complexType name="LanguageLibraries">
  <xsd:sequence>
    <xsd:element name="Library" type="LanguageLibrary" minOccurs="1"
      maxOccurs="unbounded">
      <xsd:annotation>
        <xsd:documentation>Library entry
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="LanguageLibrary">
  <xsd:attribute name="name" type="xsd:string" use="required">
    <xsd:annotation>
      <xsd:documentation>Name of the library
      </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
  <xsd:attribute name="type" use="optional">
    <xsd:annotation>
      <xsd:documentation>Type of library
      </xsd:documentation>
    </xsd:annotation>
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="math"/>
        <xsd:enumeration value="graphics"/>
        <xsd:enumeration value="geodesic"/>
        <xsd:enumeration value="other"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
```

```
    <xsd:attribute name="version" type="xsd:string" use="optional">
      <xsd:annotation>
        <xsd:documentation>Semantic version number of the library
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
  </xsd:complexType>


  <!-- OS API -->
  <xsd:complexType name="OSAPIType">
    <xsd:attribute name="type" use="required">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="POSIX"/>
          <xsd:enumeration value="ASAAC"/>
          <xsd:enumeration value="A653"/>
          <xsd:enumeration value="FACE"/>
          <xsd:enumeration value="other"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:attribute>
  </xsd:complexType>


  <!-- Compiler -->
  <xsd:complexType name="Compiler">
    <xsd:attribute name="name" type="xsd:string" use="required">
      <xsd:annotation>
        <xsd:documentation>Compiler name
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
  </xsd:complexType>

  <!-- Generic Compilation Options -->
  <xsd:simpleType name="GenericCompilationOptions">
```

```
  <xsd:restriction base="xsd:string"/>
</xsd:simpleType>

<!-- Specific Compilation Options -->
<xsd:complexType name="ModuleCompilationOptions">
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="instanceName" type="NameId" use="required">
        <xsd:annotation>
          <xsd:documentation>
            Reference to the module instance
          </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>

<!-- Extra Concerns -->
<xsd:complexType name="ExtraConcerns">
  <xsd:attribute name="file">
    <xsd:simpleType>
      <xsd:annotation>
        <xsd:documentation>
          URI to an external file
        </xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:anyURI"/>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>

<!-- Default PINFO -->
<xsd:complexType name="DefaultPINFO">
  <xsd:sequence>
    <xsd:element name="modulePINFO" minOccurs="1" maxOccurs="unbounded">
      <xsd:annotation>
```

```
      <xsd:documentation>Set of PINFOs used by the module
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element minOccurs="0" maxOccurs="unbounded" name="publicPINFO"
          type="PINFOValue">
          <xsd:annotation>
            <xsd:documentation>Public PINFO used by the module
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element minOccurs="0" maxOccurs="unbounded" name="privatePINFO"
          type="PINFOValue">
          <xsd:annotation>
            <xsd:documentation>Private PINFO used by the module
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
      <xsd:attribute name="instanceName" type="NameId" use="required">
        <xsd:annotation>
          <xsd:documentation>
            Reference to the module instance
          </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
    </xsd:complexType>
    <xsd:unique name="pinfovaluename">
      <xsd:selector xpath="tns:publicPINFO|tns:privatePINFO"/>
      <xsd:field xpath="@name"/>
    </xsd:unique>
  </xsd:element>
 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="PINFOValue">
```

```
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="name" type="xsd:string" use="required">
        <xsd:annotation>
          <xsd:documentation>
            Name of the PINFO
          </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>

<!-- Default Properties -->
<xsd:complexType name="DefaultProperties">
  <xsd:sequence>
    <xsd:element name="property" type="defaultPropertyValue"
      minOccurs="1" maxOccurs="unbounded">
      <xsd:annotation>
        <xsd:documentation>
          Default Property Value
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="defaultPropertyValue">
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="name" type="xsd:string">
        <xsd:annotation>
          <xsd:documentation>
            Name of the property
          </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
```

```
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>

  <!-- Targeted system on which the insertion policy applies to -->
  <xsd:simpleType name="Target">
    <xsd:restriction base="xsd:string"/>
  </xsd:simpleType>

</xsd:schema>
```

7.2 Example

The XML file below shows an example of insertion policy for a component implemented with two module instances M1 and M2.

```
<insertionPolicies xmlns="http://www.ecoa.technology/insertion-policy-2.0"
  xmlns:xlink="http://www.w3.org/1999/xlink" componentImplementation="Blake_C1">
  <insertionPolicy>

    <target>ECOA System A</target>

    <processorTarget xlink:type="simple"
      xlink:href="bin-desc.xml#xpointer(/processorTarget)"/>
    <registerSize size="32"/>

    <memoryUsage>
      <moduleMemoryUsage instanceName="M1">
        <userContextSize xlink:type="simple"
          xlink:href="bin-desc.xml#xpointer(/binaryModule[@reference=`M1'])"/>
        <warmStartContextSize xlink:type="simple"
          xlink:href="bin-desc.xml#xpointer(/binaryModule[@reference=`M1'])"/>
        <dynamicMemory allocation="yes" maxSize="1000000"/>
        <stackSize xlink:type="simple"
          xlink:href="bin-desc.xml#xpointer(/binaryModule[@reference=`M1'])"/>
        <heapSize xlink:type="simple"
          xlink:href="bin-desc.xml#xpointer(/binaryModule[@reference=`M1'])"/>
```

```
    </moduleMemoryUsage>
    <moduleMemoryUsage instanceName="M2">
      <userContextSize xlink:type="simple"
        xlink:href="bin-desc.xml#xpointer(/binaryModule[@reference=`M2'])"/>
      <warmStartContextSize xlink:type="simple"
        xlink:href="bin-desc.xml#xpointer(/binaryModule[@reference=`M2'])"/>
      <dynamicMemory allocation="yes" maxSize="1000000"/>
      <stackSize xlink:type="simple"
        xlink:href="bin-desc.xml#xpointer(/binaryModule[@reference=`M2'])"/>
      <heapSize xlink:type="simple"
        xlink:href="bin-desc.xml#xpointer(/binaryModule[@reference=`M2'])"/>
    </moduleMemoryUsage>
  </memoryUsage>

  <realtimeCharacteristics>
    <schedulingPolicy policy="RT-FIFO"/>
    <moduleActivationProfiles>
      <activationProfile instanceName="M1" activationType="periodic"
        queueDepth="3" maxNbOfProcessedOpsPerActivation="2">
        <periodic deadline="30" wcet="45" period="200"
          relativePriority="0"/>
      </activationProfile>
      <activationProfile instanceName="M2" activationType="reactive"
        queueDepth="3" maxNbOfProcessedOpsPerActivation="2">
        <reactive deadline="30" wcet="45" relativePriority="1">
          <rate numberOfActivations="3" timeFrame="4"/>
        </reactive>
      </activationProfile>
    </moduleActivationProfiles>
  </realtimeCharacteristics>

  <timeAccuracy value="0.2"/>
  <transportProtocol quality="lossless" integrity="high"
    maxLatency="0.01" minBandwidth="10"/>
  <deploymentConstraints uniqueProtectionDomain="true"/>

  <ECOAProfile profile="core"/>
```

```
    <compiler name="gcc"/>
    <genericCompilationOptions>-O2</genericCompilationOptions>

    <modulesDependencies>
      <moduleDependencies instanceName="M1">
        <languageRuntime xlink:type="simple"
xlink:href="Blake_C1.impl.xml#xpointer(/moduleImplementation[@name=`M1_Im'])"/>
        <languageLibraries>
          <library name="m" type="math" version="1.0"/>
        </languageLibraries>
        <directOSAccess type="POSIX"/>
        <directOSAccess type="other"/>
      </moduleDependencies>
      <moduleDependencies instanceName="M2">
        <languageRuntime xlink:type="simple"
xlink:href="Blake_C1.impl.xml#xpointer(/moduleImplementation[@name=`M2_Im'])"/>
        <moduleCompilationOptions instanceName="M1">-O3</moduleCompilationOptions>
      </moduleDependencies>
    </modulesDependencies>

    <extraConcerns file="bin-desc.xml"/>

    <defaultPINFO>
      <modulePINFO instanceName="M1">
        <publicPINFO name="mypublicPinfoOne">pubdata_1.txt</publicPINFO>
        <privatePINFO name="myprivatePinfoOne">"privdata_1.txt"</privatePINFO>
        <privatePINFO name="myprivatePinfoTwo">"subfold/privdata_2.txt"</privatePINFO>
      </modulePINFO>
      <modulePINFO instanceName="M2">
        <publicPINFO name="mypublicPinfoOne">pubdata_1.txt</publicPINFO>
        <privatePINFO name="myprivatePinfoOne">"privdata_1.txt"</privatePINFO>
        <privatePINFO name="myprivatePinfoTwo">"subfold/privdata_2.txt"</privatePINFO>
      </modulePINFO>
    </defaultPINFO>
```

```
    <defaultProperties>
      <property name="init_parameter">110</property>
      <property name="init_parameter">120</property>
    </defaultProperties>

  </insertionPolicy>
</insertionPolicies>
```

8 Appendix B – Module Behaviour XML

8.1 Principles

The module behaviour metamodel provides means to describe the temporal behaviour of each module entry-point through the XML element **EntryPoint**. This information may help to refine the temporal analysis of the whole component behaviour.

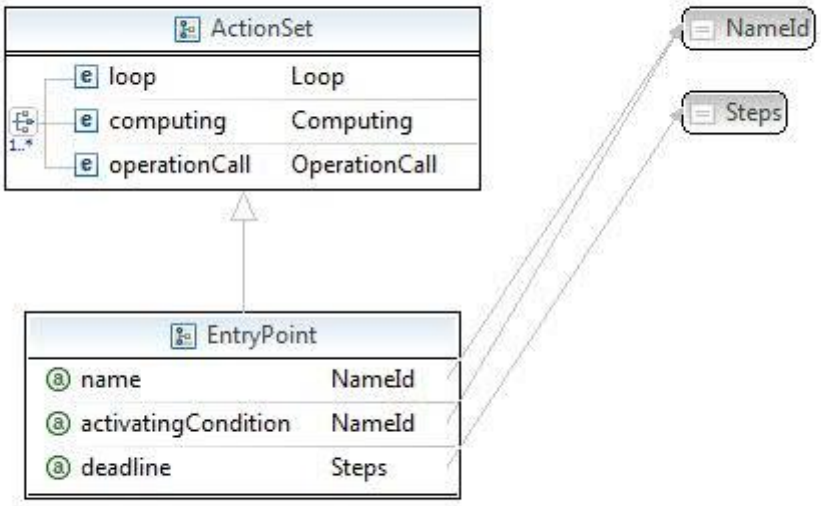

Figure 3: Module Operation Behaviour

The description of the operation behaviour is associated to the operation definition at **moduleType** level within the component implementation XML file through the attribute **name**.

The attribute **deadline** defines the deadline of this operation; this deadline starting once the operation is queued at container level.

The attribute **activatingCondition** defines the operation activating the entry point in the case where the entry point is associated to a non-activating operation.

The module behaviour is described by a set of action durations (element **ActionSet**), an action being a computing phase (element **computing**), a call to an operation of another module (element **operationCall**) or a loop of actions (element **loop**).

Action duration is defined by a range of computing steps limited by **minComputingSteps** and **maxComputingSteps**.

The module operation behaviour covers also the behaviour of notifications such as versioned data update and asynchronous response processing.

8.2 ecoa- module-behaviour-2.0.xsd

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<xsd:schema xmlns="http://www.ecoa.technology/module-behaviour-2.0"
xmlns:tns="http://www.ecoa.technology/module-behaviour-2.0"

  xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
targetNamespace="http://www.ecoa.technology/module-behaviour-2.0">


  <!-- Behaviour of module operations -->

  <!-- This XSD contains data describing a module's behaviour, which are not being
used by ECOA platforms,

  but which may contribute to early validation activities, especially regarding
schedulability analysis of a

  non-periodic system. -->


  <xsd:include schemaLocation="../ecoa-common-2.0.xsd"/>


  <xsd:element name="moduleBehaviour" type="ModuleBehaviour"/>


  <xsd:complexType name="ModuleBehaviour">

    <xsd:sequence>

      <xsd:element maxOccurs="unbounded" minOccurs="1" name="entryPoint"
type="EntryPoint">

          <xsd:annotation>

            <xsd:documentation>Incoming operation (event or request received)

            </xsd:documentation>

          </xsd:annotation>

      </xsd:element>

      <xsd:element maxOccurs="unbounded" minOccurs="0" name="dataNotification"
type="DataNotification">

          <xsd:annotation>
```

```
            <xsd:documentation>Incoming versioned data update
            </xsd:documentation>
          </xsd:annotation>
      </xsd:element>
      <xsd:element maxOccurs="unbounded" minOccurs="0" name="responseNotification"
type="ResponseNotification">
          <xsd:annotation>
            <xsd:documentation>Incoming response</xsd:documentation>
          </xsd:annotation>
      </xsd:element>

    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="ActionSet">
    <xsd:annotation>
      <xsd:documentation>A set of actions to be sequentially executed by the
        module.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:choice maxOccurs="unbounded">
      <xsd:element name="loop" type="Loop">
        <xsd:annotation>
          <xsd:documentation>Loop of actions</xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="computing" type="Computing">
        <xsd:annotation>
          <xsd:documentation>Processing internal to the module
          </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="operationCall" type="OperationCall">
        <xsd:annotation>
          <xsd:documentation>Call to an operation outside of the module
          </xsd:documentation>
        </xsd:annotation>
```

```
    </xsd:element>
  </xsd:choice>
</xsd:complexType>

<xsd:complexType name="EntryPoint">
  <xsd:complexContent>
    <xsd:extension base="ActionSet">
      <xsd:attribute name="name" type="NameId" use="required">
        <xsd:annotation>
          <xsd:documentation>Name of the incoming module operation
            See compl.impl.xml.
          </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="activatingCondition" type="NameId" use="optional">
        <xsd:annotation>
          <xsd:documentation>Reference to the activating operation if the
          operation here described is not activating
          </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="deadline" type="Steps" use="optional">
        <xsd:annotation>
          <xsd:documentation>Deadline of the operation.
            This value has no impact of the code generation.
            It is only to store intermediate analysis result done by the
            component supplier.
          </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="DataNotification">
  <xsd:complexContent>
    <xsd:extension base="EntryPoint"/>
```

```
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ResponseNotification">
  <xsd:complexContent>
    <xsd:extension base="EntryPoint"/>
  </xsd:complexContent>
</xsd:complexType>


<xsd:complexType name="Loop">
  <xsd:complexContent>
    <xsd:extension base="ActionSet">
      <xsd:attribute name="Iterations" type="xsd:positiveInteger" use="required">
        <xsd:annotation>
          <xsd:documentation>Number of iterations</xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="Computing">
  <xsd:attribute name="minComputingSteps" type="Steps" use="required">
    <xsd:annotation>
      <xsd:documentation>Minimum number of computing steps</xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
  <xsd:attribute name="maxComputingSteps" type="Steps" use="required">
    <xsd:annotation>
      <xsd:documentation>Maximum number of computing steps</xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
</xsd:complexType>

<xsd:complexType name="OperationCall">
  <xsd:attribute name="moduleOperationRef" type="NameId" use="required">
    <xsd:annotation>
```

```
        <xsd:documentation>Reference to a required operation</xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute default="0" name="minComputingSteps" type="Steps" use="optional">
      <xsd:annotation>
        <xsd:documentation>Minimum number of computing steps due to the
          operation call itself (not including computing steps to
          execute the operation in the called module)
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute default="0" name="maxComputingSteps" type="Steps" use="optional">
      <xsd:annotation>
        <xsd:documentation>Maximum number of computing steps due to the
          operation call itself (not including computing steps to
          execute the operation in the called module)
        </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
  </xsd:complexType>
</xsd:schema>
```