**YAML - XML converter**

**Development Project 14/15 – CloudCycle 2.0**

**Specification 01**

**6 May 2015**

**Specification URIs:  
This Version**

<https://github.com/CloudCycle2/YAML_Transformer/blob/master/Documentation/YAML_XML_Converter_Documentation.pdf>

**Editors:**

Michael Steffl  
Jaasiel Walter

**Abstract:**

This Document describes the supported YAML elements from the converter and the resulting elements on XML side after conversion.

Table of content

[3 Overview 2](#_Toc417943231)

[4 Node Type 3](#_Toc417943232)

[4.1 Property definition 4](#_Toc417943233)

[4.2 Capability definition 5](#_Toc417943234)

[4.3 Requirement definition 6](#_Toc417943235)

[4.4 Interface definition 7](#_Toc417943236)

[4.5 Artifact definition 8](#_Toc417943237)

[5 Node Templates 9](#_Toc417943238)

[5.1 Property definition 11](#_Toc417943239)

[5.2 Requirement definition 11](#_Toc417943240)

[5.2.1 First notation – XML requirement 12](#_Toc417943241)

[5.2.2 Second notation – XML relationship 13](#_Toc417943242)

[5.3 Capability definition 14](#_Toc417943243)

[6 Capability Type 15](#_Toc417943244)

[6.1 Property definition 16](#_Toc417943245)

[7 Artifact Type 17](#_Toc417943246)

[7.1 Property definition 18](#_Toc417943247)

[8 Relationship Type 19](#_Toc417943248)

[8.1 Property definition 20](#_Toc417943249)

[8.2 Interface definition 21](#_Toc417943250)

# Overview

These are the supported elements from the converter. At the moment imports, inputs, outputs and groups are not supported.



# Node Type



|  |
| --- |
| **YAML** |
| **<node\_type\_name>:**  derived\_from: **<parent\_node\_type\_name>**  description: **<node\_type\_description>**  properties:  <property\_definitions>  requirements:  <requirement\_definitions>  capabilities:  <capability\_definitions>  interfaces:  <interface\_definitions>  artifacts:  <artifact\_definitions> |
| **XML** |
| <NodeType name="xs**:node\_type\_name** " targetNamespace="*NamespaceURL*”>  <documentation> **node\_type\_description** </documentation>  <DerivedFrom typeRef="xs:**parent\_node\_type\_name** "/>  <PropertiesDefinition type="xs: **node\_type\_name**Properties"/>  <CapabilityDefinitions>  <CapabilityDefintion …/>  </CapabilityDefinitions>  <RequirementDefinitions>  <RequirementDefinition…/>  </RequirementDefinitions>  <Interfaces>  <Interface…/>  </Interfaces>  </NodeType>  <ArtifactTemplate> |

## Property definition



|  |
| --- |
| **YAML** |
| properties:  **<property\_name>:**  type: <**property\_type**>  description: <property\_description>  default: <default\_value> |
| **XML** |
| <PropertiesDefinition type="xs:**[*CorrespondingNodeType*]**Properties"/> |
| **XSD** |
| <xs:complexType name="t**[*CorrespondingNodeType*]**Properties">  <xs:sequence>  <xs:element name="**property\_name**" type="xs: **property\_type**" />    </xs:sequence  </xs:complexType>  <xs:element name="**[*CorrespondingNodeType*]**Properties "   type="t**[*CorrespondingNodeType*]**Properties Properties" /> |

**Notes**

**[***CorrespondingNodeType***]:** Name of the NodeType where the property corresponds to

## Capability definition



|  |
| --- |
| **YAML** |
| capabilities:  **<capability\_name>:**  type: <**capability\_type**> |
| **XML** |
| <CapabilityDefinitions>  <CapabilityDefinition name="xs: **capability\_name** " capabilityType="xs: **capability\_type** "/>  </CapabilityDefinitions> |

## Requirement definition

There are two different notations in YAML for requirements. Node types only support one notation which will create a requirement definition and the corresponding requirement type on XML side. In YAML there is no possibility to create requirement types. So on XML side they will be created automatically when creating a requirement. The other notation which will create a relationship is only supported in node templates.



|  |
| --- |
| **YAML** |
| requirements:  - **<requirement\_name>**: <**capability\_type\_name**> |
| **XML** |
| <RequirementDefinitions>  <RequirementDefinition name="xs:**requirement\_name** " requirementType=  "xs**:capability\_type\_name [*Capability replaced by Requirement*]** "/>  </RequirementDefinitions>  …  <RequirementType requiredCapabilityType="xs: **capability\_type\_name** " name=  "xs**:capability\_type\_name [*Capability replaced by Requirement*]** "/> |

**Notes**

*[Capability replaced by Requirement]:* “Capability” in capability\_type\_name will be replaced by “Requirement”

## Interface definition



|  |
| --- |
| **YAML** |
| interfaces:  <**interface\_name**>:  <**operation\_name**>:  implementation: <**implementation\_artifact\_name**> |
| **XML** |
| <Interfaces>  <Interface name=" **interface\_name** ">  <Operation name="**operation\_name** "/>  </Interface>  </Interfaces>  …  <NodeTypeImplementation name=" xs**:[*CorrespondingNodeType*]**Implementation"  nodeType="xs**:[*CorrespondingNodeType*]**">  <ImplementationArtifacts>  <ImplementationArtifact interfaceName="xs: **interface\_name** " operationName=  "xs:**operation\_name** " artifactType="xs:implementation\_artifact\_type" artifactRef="xs:  **implementation\_artifact\_name** "/>  </ImplementationArtifacts>  </NodeTypeImplementation> |

**Notes**

*[CorrespondingNodeType]: Name of the NodeType where the interface corresponds to*Implementation\_artifact\_name: Name of the artifact definitionImplementation\_artifact\_type: Type of the artfifact

## Artifact definition



|  |
| --- |
| **YAML** |
| **<artifact\_name>**: **<artifact\_file\_URI>**  type: **<artifact\_type\_name>**  description: <artifact\_description>  mime\_type: <artifact\_mime\_type\_name> |
| **XML** |
| <ArtifactTemplate name="xs:**artifact\_name**" id=" xs:**artifact\_name** " type=" xs:**artifact\_type\_name** ">  <Properties>  <**artifact\_type\_name**Properties types="http://www.example.org/tosca/yamlgen/types" xmlns="http://www.example.org/tosca/yamlgen/types"/>  </Properties>  <ArtifactReferences>  <ArtifactReference reference=" xs:**artifact\_file\_URI [*Folder*]**">  <Include pattern="xs: **artifact\_file\_URI [*File*]**">  <ArtifactReference>  <ArtifactReferences>  <ArtifactTemplate> |

**Notes**

[*Folder*]: Path of folder where the artifact file is stored.  
[*File*]: Name of artifact file.

# Node Templates



|  |
| --- |
| **YAML** |
| **<node\_template\_name>:**  type: **<node\_type\_name>**  description: **<node\_template\_description>**  properties:  <property\_definitions>  requirements:  <requirement\_definitions>  capabilities:  <capability\_definitions>  interfaces:  <interface\_definitions>  artifacts:  <artifact\_definitions> |
| **XML** |
| <NodeTemplate name="xs**:node\_template\_name**" id=" xs**:node\_template\_name** " type=" xs**:node\_type\_name** ">  <Properties>  <Property…/>  </Properties>  <Capabilities>  <Capability…/>  </Capabilities>  <Requirements>  <Requirement…/>  </Requirements>  </NodeTemplate> |

## Property definition



|  |
| --- |
| **YAML** |
| properties:  <**property\_name**>: <**property\_value**> |
| **XML** |
| <xs:**[*CorrespondingNodeType*]**Properties>  < **property\_name** > **property\_value** </ **property\_name** >  </xs:**[*CorrespondingNodeType*]**Properties> |

**Notes**

[*CorrespondingNodeType*]: Name of the assigned NodeType from the NodeTemplate the property corresponds to

## Requirement definition

There are two different notations in YAML for requirements. Node Templates support both notations. First notation creates requirements and corresponding requirement types. The second notation creates relationships.

### First notation – XML requirement



|  |
| --- |
| **YAML** |
| requirements:  - **<requirement\_name>**: <**capability\_type\_name**> |
| **XML** |
| <Requirements>  <Requirement name="xs: **requirement\_name** " requirementType=  "xs**:capability\_type\_name [*Capability replaced by Requirement*]** "/>  </Requirements>  …  <RequirementType requiredCapabilityType="xs: **capability\_type\_name** " name=  "xs**:capability\_type\_name [*Capability replaced by Requirement*]** "/> |

**Notes**

*[Capability replaced by Requirement]:* “Capability” in capability\_type\_name will be replaced by “Requirement”

### Second notation – XML relationship



|  |
| --- |
| **YAML** |
| requirements:  - **<requirement\_name>**: **<node\_name >**  relationship\_type: <**relationship\_name**> |
| **XML** |
| <RelationshipTemplate id=" **requirement\_name** " type="xs: **relationship\_name** ">  <SourceElement ref="***[CorrespondingNodeTemplate]***"/>  <TargetElement ref=" **node\_name** "/>  </RelationshipTemplate> |

**Notes**

*[CorrespondingNodeTemplate]:* Name of the NodeTemplate the requirement corresponds to. There is only the possibility to relate NodeTemplates.

## Capability definition



|  |
| --- |
| **YAML** |
| capabilities:  <**capability\_name**>:  type: <**capability\_type**> |
| **XML** |
| <Capabilities>  <Capability name="xs:**capability\_name** " id="xs:**capability\_name** " type="xs:**capability\_type** "/>  </Capabilities> |

# Capability Type



|  |
| --- |
| **YAML** |
| capability\_types:  <**capability\_type\_name**>:  derived\_from: <**parent\_capability\_type\_name**>  description: <**capability\_description**>  properties:  <property\_definitions> |
| **XML** |
| <CapabilityType name="xs: **capability\_type\_name** ">  <documentation> **capability\_description** </documentation>  <DerivedFrom typeRef="xs: **parent\_capability\_type\_name** "/>  <PropertiesDefinition type="xs:**capability\_type\_name** Properties"/>  </CapabilityType> |

## Property definition



|  |
| --- |
| **YAML** |
| properties:  **<property\_name>:**  type: <**property\_type**>  description: <property\_description>  default: <default\_value> |
| **XML** |
| <PropertiesDefinition type="xs:**[*CorrespondingCapabilityType*]**Properties"/> |
| **XSD** |
| <xs:complexType name="t**[*CorrespondingCapabilityType*]**Properties">  <xs:sequence>  <xs:element name="**property\_name**" type="xs: **property\_type**" />    </xs:sequence  </xs:complexType>  <xs:element name="**[*CorrespondingCapabilityType*]**Properties "   type="t**[*CorrespondingCapabilityType*]**Properties Properties" /> |

**Notes**

*[CorrespondingCapabilityType]:* Name of the CapabilityType the property corresponds to

# Artifact Type



|  |
| --- |
| **YAML** |
| artifact\_types:  <**artifact\_type\_name**>:  derived\_from: <parent\_artifact\_type\_name>  description: <artifact\_description>  mime\_type: <mime\_type\_string>  file\_ext: [<file\_extension\_1>, ..., <file\_extension\_n>]  properties:  <property\_definitions> |
| **XML** |
| <ArtifactType name="**artifact\_type\_name**" targetNamespace="*NamespaceURL*">  <PropertiesDefinition type="xs: **artifact\_type\_name**Properties"/>  </ArtifactType> |

## Property definition



|  |
| --- |
| **YAML** |
| properties:  **<property\_name>:**  type: <**property\_type**>  description: <property\_description>  default: <default\_value> |
| **XML** |
| <PropertiesDefinition type="xs:**[*CorrespondingArtifactType*]**Properties"/> |
| **XSD** |
| <xs:complexType name="t**[*CorrespondingArtifactType*]**Properties">  <xs:sequence>  <xs:element name="**property\_name**" type="xs: **property\_type**" />    </xs:sequence  </xs:complexType>  <xs:element name="**[*CorrespondingArtifactType*]**Properties "   type="t**[*CorrespondingArtifactType*]**Properties Properties" /> |

**Notes**

*[CorrespondingArtifactType]:* Name of the ArtifactType the property corresponds to

# Relationship Type



|  |
| --- |
| **YAML** |
| relationship\_types:  <**relationship\_type\_name**>:  derived\_from: <**parent\_relationship\_type\_name**>  description: <**relationship\_description**>  properties:  <property\_definitions>  interfaces:  <interface\_definitions>  valid\_targets: [ <**entity\_name\_or\_type**>] |
| **XML** |
| <RelationshipType name="xs: **relationship\_type\_name** ">  <documentation> **relationship\_description** </documentation>  <DerivedFrom typeRef="xs: **parent\_relationship\_type\_name** "/>  <PropertiesDefinition type="xs:**[*CorrespondingRelationshipType*]**Properties"/>  <TargetInterfaces>  <Interface…/>  </TargetInterfaces>  <ValidTarget typeRef="xs: **entity\_name\_or\_type** "/>  </RelationshipType> |

## Property definition



|  |
| --- |
| **YAML** |
| properties:  **<property\_name>:**  type: <**property\_type**>  description: <property\_description>  default: <default\_value> |
| **XML** |
| <PropertiesDefinition type="xs:**[*CorrespondingRelationshipType*]**Properties"/> |
| **XSD** |
| <xs:complexType name="t**[*CorrespondingRelationshipType*]**Properties">  <xs:sequence>  <xs:element name="**property\_name**" type="xs: **property\_type**" />    </xs:sequence  </xs:complexType>  <xs:element name="**[*CorrespondingRelationshipType*]**Properties "   type="t**[*CorrespondingRelationshipType*]**Properties Properties" /> |

**Notes**

*[CorrespondingRelationshipType]:* Name of the RelationshipType the property corresponds to

## Interface definition



|  |
| --- |
| **YAML** |
| interfaces:  <**interface\_name**>:  <**operation\_name**>:  implementation: <**implementation\_artifact\_name**> |
| **XML** |
| <TargetInterfaces>  <Interface name=" **interface\_name** ">  <Operation name="**operation\_name** "/>  </Interface>  </ TargetInterfaces > |