Import of Complex XSD Type Deriving from a Simple Type

XML schemas allow to specify a complex type by extending a simple one. The schema below shows an example defining the complex type **MyDerivedElement** by extending the type **MyType**. **MyType** is an enumeration of string values. A XML instance complying to this schema is depicted below as well. When mapping **MyDerivedElement** to an UML class, we follow the same rules as for other complex types except that we add an additional attribute **XMLSimpleContent** that holds that actual value of the XML element. In the example below, **XMLSimpleContent** would evaluate to the string **a**.

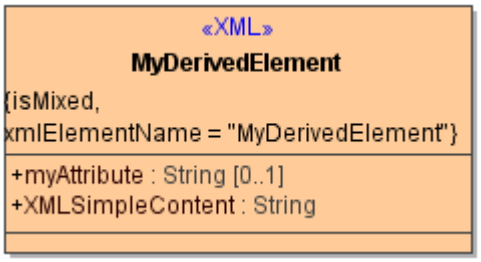
Schema

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:simpleType name="MyType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="a"
/>
            <xs:enumeration value="b"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:element name="MyDerivedElement">
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="MyType">
                    <xs:attribute name="myAttribute" type="
xs:string"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

XML Instance

```
<MyDerivedElement myAttribute="xx">a<
/MyDerivedElement>
```

UML Class



Mapping Rule

Each complex XSD type that is derived from a simple type is mapped to an UML class having a special attribute **XMLSimpleContent** that holds the actual value of the element. We do not map the XSD extension element to an UML generalization in this case because this would be inconsistent to our internal meta model where each type deriving from a simple type is a simple type as well.