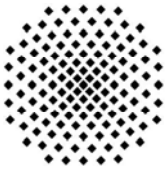

Universität Stuttgart

Fakultät Informatik, Elektrotechnik und Informationstechnik

WS-BPEL Extension for Compliance Fragments (BPEL4CFrags), Version 1.0

Katharina Görlach, Oliver Kopp, Frank Leymann, David Schumm, Steve Strauch

Report 2011/01
January 12, 2011

Institut für Architektur von Anwendungssystemen

Universitätsstr. 38
70569 Stuttgart
Germany

CR: D.2.12, D.2.13, D.3.3, H.4.1

WS-BPEL Extension for Compliance Fragments (BPEL4CFrags), Version 1.0

January 2011

Editor

Steve Strauch

Authors (in alphabetical order)

Katharina Görlach
Oliver Kopp
Frank Leymann
David Schumm
Steve Strauch

Copyright Notice

Licence

Abstract

The Web Services Business Process Execution Language, version 2.0 (WS-BPEL 2.0 or BPEL for brevity) introduces a model for business processes based on Web services. A BPEL process orchestrates interactions among different Web services. The language comprises features required to describe complex control flows, including error handling and compensation behaviour.

BPEL for Compliance Fragments (BPEL4CFrags) enables the specification of compliance fragments providing a reusable solution for implementing and meeting compliance requirements. The compliance fragments are not necessarily executable. Providing process modeling support at design time requires the completion of not directly executable compliance fragments, because compliance fragments contain several degrees of freedom, into already existing executable BPEL processes. Moreover, assistance for creation of executable BPEL processes from scratch during IT refinement has to be provided.

Status

BPEL4CFrags is provided as-is and for review and evaluation only. The authors make no warrantees or representations regarding the specifications in any manner whatsoever.

Table of Contents

1 Introduction

This specification introduces an extension to BPEL [WS-BPEL 2.0] to enable specifying reusable compliance fragments for implementing compliance requirements. The compliance requirements originate from the Mobile Virtual Network Operators scenario (WatchMe) [BDL+10] and the THALES ICT Security scenario [STK+10] of the FP7 COMPAS project (www.compas-ict.eu, contract no. FP7-215175).

Consequently, the current version of this specification allows for and eases the realization and creation of reusable solutions for compliance requirements from domains internal policy, licensing and security at design time. The specification is, however, not limited to these domains.

2 Language Design

The BPEL4CFrags extension is layered on top of BPEL. Its elements can be composed with BPEL elements for specifying compliance fragments.

Due to the fact that compliance fragments are not necessarily executable they have to be completed and integrated into an already existing BPEL process model or a new executable BPEL process has to be modelled from scratch before deployment. Therefore the language elements provided by BPEL4CFrags have to be mapped to standard BPEL elements. The transformation steps are presented in Section 5.

All elements and attributes introduced in this extension are made available to both BPEL executable processes and abstract processes.

2.1 Dependencies on Other Specifications

WS-BPEL Extension for Compliance Fragments utilizes the WS-BPEL 2.0 specification. BPEL4CFrags extends the WS-BPEL 2.0 process model and uses existing WS-BPEL 2.0 capabilities.

2.2 Notational Conventions

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC 2119].

2.3 Namespaces

This specification uses a number of namespace prefixes throughout; they are listed in Table 1. Note that the choice of any namespace prefix is arbitrary and not semantically significant (see [XML Namespaces]).

Prefix	Namespace
bpel	http://docs.oasis-open.org/wsbpel/2.0/process/abstract
b4c	http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags
ext	http://www.iaas.uni-stuttgart.de/ext/id
xsd	http://www.w3.org/2001/XMLSchema

Table 1 Prefixes and namespaces used in this specification

All information items defined by BPEL4CFrags are identified by the XML namespace URIs [XML Namespaces] `http://www.iaas.uni-stuttgart.de/bpel4cfrags`.

2.4 Language Extensibility

The BPEL4CFrags specification extends the reach of the standard BPEL extensibility mechanism to BPEL4CFrags elements. This allows:

- Attributes from other namespaces to appear on any BPEL4CFrags element
- Elements from other namespaces to appear within BPEL4CFrags elements

Extension attributes and extension elements MUST NOT contradict the semantics of any attribute or element from the BPEL4CFrags namespace.

The standard BPEL element <extension> must be used to declare mandatory and optional extensions of BPEL4CFrags.

3 Compliance Fragment Definition and Concept

Based on our research on compliance fragments in the COMPAS project we introduce a general definition of a process fragment which we apply to the field of business process compliance meaning that a process fragment is used for realization of one or more compliance concern [SLM+10]:

“A *process fragment* in our work is defined as a *connected graph*, however with significantly relaxed completeness and consistency criteria compared to an *executable process graph*. A process fragment is made up of *activities*, *activity placeholders* (so-called regions) and *control edges* that define control dependency among them. A process fragment may define a *context* (e.g., variables) and it may contain a *process start* or *process end node*, but it is not required to do so. It may contain multiple entering and leaving *control edges* for integration into a process or with other process fragments. A process fragment has to consist of at least one activity and there must be a way to complete it to an executable process graph.”

A *process graph* describes a model for a process, where *nodes* represent activities of a process and *edges* represent control dependencies between them.

This definition of the term *process fragment* allows creating a fragment from scratch and so there is no need for a process containing the fragment. For avoiding the specification of fragments which make no sense we have added the requirement that there must be way to complete it to an executable process graph. Thus a process fragment is not necessarily directly executable and it may be partially undefined.

Due to the fact that essential information for execution are missing within compliance fragments, e.g., the partner links have to be set and the transformation as described in Section 5 has to be done, we use the namespace for abstract BPEL when specifying compliance fragments. The namespace for abstract BPEL (http://docs.oasis-open.org/wsbpel/2.0/process/abstract) is changed to the namespace of executable BPEL (http://docs.oasis-open.org/wsbpel/2.0/process/executable) when applying the XSL stylesheet during the transformation, cf. Listing 13.

The introduced definition is not specific for BPEL and can basically be applied to other process languages that can be represented by a process graph, as well. For instance, it can also be applied to the Business Process Modeling Notation (BPMN, [BPMN 2.0]).

Based on the solution concepts discussed in the previous paragraphs, we have proposed particular additions to our definition. For usage of process fragments in the field of compliance we propose in [SLM+10] the following additional characteristics:

i. A process fragment may be *parametrizable* in order to mark points of variability. The parts (variables etc.) of a process fragment which may be changed can be explicitly declared. This way it can be better ensured that, after integration into a process, a process fragment still implements the compliance requirement that it has been designed for. For some cases it may also be conceivable to limit the range of valid values for the parameters.

ii. The placeholders contained in a process fragment (i.e. *regions*) may be constrainable. By constraining the regions it can be stated how those placeholders may be filled. Just like the concept of parameterization this concept allows us to ensure more stability of the fragment. However, an important difference to parameterization is that constrained regions can be used to specify how a fragment may be composed. This influences flexibility on the one hand and ensures on the other hand that the design intention of the fragment does not get broken.

iii. Optional and mandatory *entries* and *exits* of a process fragment should be distinguishable. We believe it is of fundamental importance to declare how the fragment has to be wired within a process for ensuring compliant behaviour. Mandatory entries and exits have to be wired for making the fragment function properly, optional ones can be neglected.

4 Specifying a BPEL4CFrag Compliance Fragment

In this section, we introduce the elements of the extension BPEL4CFrags and the overall language structure. We have taken care to adhere to the language extension

formalities provided in [WS-BPEL 2.0] and not to change the semantics of standard BPEL constructs in order to create a standard conform BPEL extension.

All extensions we propose within BPEL4CFrags are design time language extensions, without exception. When integrating a compliance fragment into a process, the fragment has to be tailored to the specific needs of the context in which it is used. For integration we propose the gluing approach as described in [SLM+10] and Section 5. Gluing denotes that compliance fragments are physically copied and integrated into the process. For traceability and for maintenance reasons we can use unique identifiers which allow distinguishing between the original parts of the process and the integrated compliance parts.

Mandatory entries and exits have to be wired with the process or composed with other fragments for achieving an overall complete and executable process model, and thereby get removed. During the task of integration the placeholders are replaced with real functionality, i.e. the parameters are replaced with specific values, and regions are filled with activities or other fragments. The verification that constraints on regions and parameters are satisfied depends on the language that is used for specifying constraints, e.g., Linear Temporal Logic (LTL). This, however, does not require an extension to BPEL, because annotations stating the constrains are made refering from outside the compliance fragment to uniquely identifiable elements within the compliance fragment. This is a simple and reusable approach for annotion while avoiding pollution of the fragment itself. For referencing concrete elements within a compliance fragment from the outside, e.g., for adding constraints we propose to use XML Linking Language (XLink [XLink 1.0]) in combination with the XPointer xpointer() Scheme [XPointer]. The details of the annotation of compliance fragments from outside the corresponding compliance fragments are out of scope of this specification.

This section provides a brief summary of BPEL4CFrags extension elements, described in the following subsections.

For the identifier extension we have chosen a different namespace (`http://www.iaas.uni-stuttgart.de/ext/id`) than for the other extensions, as this extension can be ignored by an execution engine, i.e. `mustUnderstand="no"` (cf. Listing 7). For the XML Schema of the unique identifier extension see Listing 14 in Appendix A.

The other extensions (regions etc.) have to be replaced before execution. By using `mustUnderstand="yes"` for the other namespace `http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags` an engine will reject running a process where not all elements originating from the BPEL4CFrags extension with this namespace are replaced by standard BPEL constructs. For the XML Schema for the WS-BPEL extension for compliance fragments see Listing 15 in Appendix B.

4.1 Unique Identifier (ext:id)

Annotations to a compliance fragment can be either made inline or by referencing from outside. In order to avoid the pollution of compliance fragment we propose the latter. A potential annotation from the outside using, e.g. XLink and XPointer (cf. Section 3) requires that all elements within a compliance fragment have a unique identifier.

In order to keep the fragment clean from many constraints and parameters we propose the extension of all BPEL elements with an identifier attribute for reference from

the outside. An annotation mechanism that references this identifier allows the specification of constraints on regions and parameters on BPEL constructs from the outside.

Keeping the constraint and parameter information outside the compliance fragment enables a looser coupling between the fragment and the constraints as well as parameters, and thereby increases reusability. Listing 1 shows the syntax of the mandatory attribute for a unique identifier.

```
xmlns:ext="http://www.iaas.uni-stuttgart.de/ext/id"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
ext:id="xsd:string"
```

Listing 1 Syntax of the mandatory attribute for Unique Identifier

4.2 Fragment Scope (b4c:fragmentScope)

A compliance fragment may define a context (e.g., variables). In BPEL, the scope activity is used for this purpose. However, we have to distinguish between the scope of a fragment and a BPEL scope for avoiding confusion on the one hand, and for providing clear semantics on the other hand. A *fragmentScope* can be used as container for context constructs, such as *variables, partnerLinks, faultHandlers*, etc. However, the context of a fragment has to be merged with the context of the process in which it is used during integration or with the context of several fragments during composition. Therefore a *fragmentScope* has the same characteristics as a normal BPEL scope in terms of XML schema, but not the same semantics.

The *fragmentScope* is a context container to be used for definition of, e.g., local variables, etc. The following listing introduces the syntax of a fragment scope. Please note that the placeholders *standard-attributes, standard-elements* and *standard-activity* used for sake of simplicity represent the corresponding constructs according to the BPEL specification [WS-BPEL 2.0].

```
xmlns:b4c=" http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"
xmlns:ext="http://www.iaas.uni-stuttgart.de/ext/id">
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
<b4c:fragmentScope ext:id="xsd:string" bpel:isolated="yes|no"?
   bpel:exitOnStandardFault="yes|no"?
   standard-attributes>
   standard-elements
   <bpel:variables ext:id="xsd:string">?
      ...
   </bpel:variables>
   <bpel:partnerLinks ext:id="xsd:string">?
      ...
   </bpel:partnerLinks>
```

```
   <bpel:messageExchanges ext:id="xsd:string">?
      ...
   </bpel:messageExchanges>
   <bpel:correlationSets ext:id="xsd:string">?
      ...
   </bpel:correlationSets>
   <bpel:eventHandlers ext:id="xsd:string">?
      ...
   </bpel:eventHandlers>
   <bpel:faultHandlers ext:id="xsd:string">?
      ...
   </bpel:faultHandlers>
   <bpel:compensationHandler ext:id="xsd:string">?
      ...
   </bpel:compensationHandler>
   <bpel:terminationHandler ext:id="xsd:string">?
      ...
   </bpel:terminationHandler>
   standard-activity
</b4c:fragmentScope>
```

Listing 2 Syntax of a Fragment Scope

4.3 Fragment Entry (b4c:fragmentEntry)

A *fragmentEntry* is one important integration point for integrating a compliance fragment into an already existing process or for composing different fragments. Thus, this placeholder is removed during integration or composition.

A *fragmentEntry* must have one or more outgoing control links, and it must not have any incoming control links. An attribute (type="mandatory | optional") specifies whether the entry has to be wired or if it (and its outgoing control links) can be removed. If a *fragmentEntry* has multiple outgoing control links, then all of those control links must have their source in one and the same activity in the final composition. Listing 3 shows the syntax of a *fragmentEntry*.

```
xmlns:b4c=" http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"
xmlns:ext="http://www.iaas.uni-stuttgart.de/ext/id">
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
<bpel:extensionActivity>
   <b4c:fragmentEntry b4c:name="xsd:string"
      b4c:type="mandatory|optional"
      ext:id="xsd:string">
      <bpel:sources ext:id="xsd:string">?
```

```
         <bpel:source bpel:linkName="NCName" ext:id="xsd:string">+
            <bpel:transitionCondition
               bpel:expressionLanguage="anyUri">?
         </bpel:source>
      </bpel:sources>
   </b4c:fragmentEntry>
</bpel:extensionActivity>
```

Listing 3 Syntax of a Fragment Entry

4.4 Fragment Exit (b4c:fragmentExit)

Similar to a *fragmentEntry*, a *fragmentExit* is a point of integration or composition. It must have at least one incoming control link, and it must not have any outgoing control links. A fragment exit has the same attribute (`type="mandatory | optional"`) as a *fragmentEntry* with analogous semantics. If a fragment exit has multiple incoming control links, then all of those control links must have their target in one and the same activity in the final composition. Listing 4 presents the syntax of a *fragmentExit.*

```
xmlns:b4c=" http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"
xmlns:ext="http://www.iaas.uni-stuttgart.de/ext/id">
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
<bpel:extensionActivity>
   <b4c:fragmentExit b4c:name="xsd:string"
      b4c:type="mandatory|optional"
      ext:id="xsd:string">
      <bpel:targets ext:id="xsd:string">?
         <bpel:joinCondition
            expressionLanguage="anyUri"
            ext:id="xsd:string"/>?
         <bpel:target linkName="NCName" ext:id="xsd:string"/>+
      </bpel:targets>
   </b4c:fragmentExit>
</bpel:extensionActivity>
```

Listing 4 Syntax of a Fragment Exit

4.5 Fragment Region (b4c:fragmentRegion)

A *fragmentRegion* is a construct that needs to be replaced with (connected) standard BPEL activities and/or structured activities. This means a *fragmentRegion* can also be used for composition of several compliance fragments and thus enables recursive specification of compliance fragments.
The constraints that are imposed on a *fragmentRegion* are out of scope of the BPEL4CFrags extension and thus we do not define which languages may be used for specifying constraints. The syntax of a *fragmentRegion* is shown in Listing 5.

```
xmlns:b4c=" http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"
xmlns:ext="http://www.iaas.uni-stuttgart.de/ext/id">
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
<bpel:extensionActivity>
   <b4c:fragmentRegion b4c:name="xsd:string"
      ext:id="xsd:string">
      <bpel:targets ext:id="xsd:string">?
         <bpel:joinCondition
            expressionLanguage="anyUri" ext:id="xsd:string"/>?
         <bpel:target linkName="NCName" ext:id="xsd:string"/>+
      </bpel:targets>
      <bpel:sources ext:id="xsd:string">?
         <bpel:source linkName="NCName" ext:id="xsd:string"/>+
            <bpel:transitionCondition
               expressionLanguage="anyUri" ext:id="xsd:string">?
         </bpel:source>
      </bpel:sources>
   </b4c:fragmentRegion>
</bpel:extensionActivity>
```

Listing 5 Syntax of a Fragment Region

4.6 Fragment Flow (b4c:fragmentFlow)

BPEL is a hybrid language, both block-structured and graph-based [KMW+09]. For supporting the concept of possible multiple entries and exits of a fragment we have the main focus on the graph-based part, i.e. we have taken the BPEL flow construct as basis. A *fragmentFlow* does not have the same semantic as a standard BPEL flow. When integrating the compliance fragment into a process or composing several compliance fragments, the control links nested in the *fragmentFlow* have to be merged with the control links nested in the parent BPEL flow, and the *fragmentFlow* has to be removed. Please note that the placeholders *standard-attributes, standard-elements* and *standard-activity* used for sake of simplicity represent the corresponding constructs according to the BPEL specification [WS-BPEL 2.0]. Additionally the placeholder *b4c-standard-element* used within the following listing represents all elements within the namespace http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags described in this document.

```
xmlns:b4c="http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"
xmlns:ext="http://www.iaas.uni-stuttgart.de/ext/id">
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
<bpel:extensionActivity>
   <b4c:fragmentFlow ext:id="xsd:string" standard-attributes>
```

```
      standard-elements
      <bpel:links ext:id="xsd:string">?
         <bpel:link bpel:name="NCName" ext:id="xsd:string">+
      </bpel:links>
      (standard-activity|b4c-standard-elements)+
   </b4c:fragmentFlow>
</bpel:extensionActivity>
```

Listing 6 Syntax of a Fragment Flow

4.7 Overall Language Structure

Listing 7 shows the language structure of the BPEL extension for compliance using a notation similar to the Extended Backus–Naur Form (EBNF) [ISO/IEC 14977]. Please note that the placeholders *standard-attributes,* standard-elements *and standard-activity* used for sake of simplicity in Listing 7 represent the following constructs contained in the BPEL specification [WS-BPEL 2.0]:

- *standard-attributes:* represent the standard-attributes allowed for the corresponding construct from the BPEL specification. Due to its definition (see 4.2) a fragmentScope has the same characteristics as a normal BPEL scope and so the standard-attributes of a BPEL scope might be used.
- *standard-elements:* represent standard elements from the BPEL specification allowed at the appropriate place due to the syntax provided within the BPEL specification, e.g., fault handler.
- *standard-activities:* represent standard activities from the BPEL specification allowed at the appropriate place due to the syntax provided within the BPEL specification, e.g., flow, receive, pick, etc.

```
<bpel:process ...
   ...
   xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
   xmlns:b4c=" http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"
   xmlns:ext="http://www.iaas.uni-stuttgart.de/ext/id">
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   ...
   <bpel:extensions>
      <bpel:extension namespace="http://www.iaas.uni-stuttgart.de/
         ext/bpel4cfrags"
         mustUnderstand="yes"/>
      <bpel:extension namespace="http://www.iaas.uni-stuttgart.de/
         ext/id"
         mustUnderstand="no"/>
   </bpel:extensions>
   ...
   <bpel:extensionActivity>
```

```
<b4c:fragmentScope ext:id="xsd:string"
   standard-attributes>
   (standard-elements
   |
   <bpel:extensionActivity>
      <b4c:fragmentEntry b4c:name="xsd:string"
         b4c:type="mandatory|optional"
         ext:id="xsd:string">
         <bpel:sources>?
            <bpel:source linkName="NCName">+
               <bpel:transitionCondition
                  expressionLanguage="anyUri">?
            </bpel:source>
         </bpel:sources>
      </b4c:fragmentEntry>
   </bpel:extensionActivity>
   |
   <bpel:extensionActivity>
      <b4c:fragmentRegion b4c:name="xsd:string"
         ext:id="xsd:string">
         <bpel:targets>?
            <bpel:joinCondition expressionLanguage="anyUri"/>?
            <bpel:target linkName="NCName"/>+
         </bpel:targets>
         <bpel:sources>?
            <bpel:source linkName="NCName">+
               <bpel:transitionCondition
                  expressionLanguage="anyUri">?
            </bpel:source>
         </bpel:sources>
      </b4c:fragmentRegion>
   </bpel:extensionActivity>
   |
   <bpel:extensionActivity>
      <b4c:fragmentFlow ext:id="xsd:string" standard-attributes>
         standard-elements
         <bpel:links>?
            <bpel:link name="NCName">+
         </bpel:links>
         (standard-activity+
         |
```

```
            <bpel:extensionActivity>
               <b4c:fragmentEntry b4c:name="xsd:string"
                  b4c:type="mandatory|optional"
                  ext:id="xsd:string">
                  <bpel:sources>
                     <bpel:source linkName="NCName">+
                        <bpel:transitionCondition
                           expressionLanguage="anyUri">?
                     </bpel:source>
                  </bpel:sources>
               </b4c:fragmentEntry>
            </bpel:extensionActivity>
            |
            <bpel:extensionActivity>
               <b4c:fragmentRegion b4c:name="xsd:string"
                  ext:id="xsd:string">
                  <bpel:targets>?
                     <bpel:joinCondition
                        expressionLanguage="anyUri"/>?
                     <bpel:target linkName="NCName"/>+
                  </bpel:targets>
                  <bpel:sources>?
                     <bpel:source linkName="NCName">+
                        <bpel:transitionCondition
                           expressionLanguage="anyUri">?
                     </bpel:source>
                  </bpel:sources>
               </b4c:fragmentRegion>
            </bpel:extensionActivity>
            |
            <bpel:extensionActivity>
               <b4c:fragmentExit b4c:name="xsd:string"
                  b4c:type="mandatory|optional"
                  ext:id="xsd:string">
                  <bpel:targets>
                     <bpel:joinCondition
                        expressionLanguage="anyUri"/>?
                     <bpel:target linkName="NCName"/>+
                  </bpel:targets>
               </b4c:fragmentExit>
            </bpel:extensionActivity>
            )+
         </b4c:fragmentFlow>
```

```
        </bpel:extensionActivity>
        |
        <bpel:extensionActivity>
           <b4c:fragmentExit b4c:name="xsd:string"
              b4c:type="mandatory|optional"
              ext:id="xsd:string">
              <bpel:targets>?
                 <bpel:joinCondition
                    expressionLanguage="anyUri"/>?
                 <bpel:target linkName="NCName"/>+
              </bpel:targets>
           </b4c:fragmentExit>
        </bpel:extensionActivity>
        )+
     </b4c:fragmentScope>
  <bpel:extensionActivity>
</bpel:process>
```

Listing 7 Language structure of BPEL extension for compliance fragments

5 Transformation of BPEL4CFrags to WS-BPEL

Process design tools (such as the View-based Modelling Framework, cf. [COMPAS D1.3]) have to be extended accordingly for being able to handle the BPEL extensions for compliance fragments.

On the one hand being able to use or at least to properly display a compliance fragment also in standard BPEL design tools (i.e., tools which do not support the extensions we propose) and on the other hand to support the modeller during IT refinement phase, BPEL compliance fragments have to be transformed to standard BPEL code.

In addition, transformation functionality could increase the overall adoption of the concept, because it provides basic interoperability between tools which implement the BPEL extensions and those which do not.

The transformation is done by removing the extension declarations, removing the extension identifiers, replacing each *fragmentRegion*, *fragmentEntry* and *fragmentExit* by an empty activity, and the *fragmentScope* by a regular BPEL scope and the *fragmentFlow* by a regular BPEL flow activity. The names of these constructs have to be manually adjusted, respectively. Please note that this approach is not an automated approach to transform potentially non-executable BPEL compliance fragments for compliance into executable BPEL processes, but rather a part of a semi-automatic approach supporting the process modeller during IT refinement and enabling display of BPEL compliance fragments in standard BPEL modelling tools. Especially the completion of the result of the transformation to an executable BPEL process might require manual editing and adaption, e.g., when integrating a compliance fragment into an existing BPEL process or composing several ones to create a compliant BPEL process from scratch. In this context it must be pointed out, that the semantic might change during the replacement of BPEL4CFrags activities by BPEL empty activities.

We have implemented the transformation using XSL Transformation (XSLT, [XSLT 1.0]) stylesheets. We use these stylesheets in combination with an existing XSL transformer [NXSLT3] in order to transform the compliance fragments into standard BPEL code. Listing 13 in Appendix A – XSL Style Sheet for Transformation shows the concrete XSLT stylesheet that is used for the transformation.

In the following subsections we will provide the results when transforming the particular BPEL4CFrags extension elements to executable BPEL.

5.1 Transformation of Unique Identifier (ext:id)

Due to the fact that the element for the unique identifier has been introduced to enable annotation of compliance fragments from outside, these attributes are not taken into consideration during the semi-automatic transformation. The constraints that might have been annotated to the original compliance fragment have to be resolved and fulfilled by the process modeller during integration phase.

5.2 Transformation of Fragment Scope (b4c:fragmentScope)

Listing 8 shows the result of the transformation of a *fragmentScope* to executable BPEL. The *fragmentScope* is transformed to a BPEL scope and the name of the *fragmentScope* is retained. Please note that the BPEL namespace has changed during the transformation from the abstract one used in the compliance fragments to the executable one, cf. Section 3. Moreover please take into consideration, that the *fragmentScope* within a compliance fragment might contain additional BPEL standard elements as well as additional BPEL4CFrags extension elements. In such a case the BPEL standard elements are simply taken from the compliance fragments and added to the transformation result at the according place. Additional BPEL4CFrags extension elements are transformed as described in the Sections 5.1 to 5.6.

```
xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
<bpel:scope name="The name of the Fragment Scope transformed">
   standard-elements
</bpel:scope>
```

Listing 8 Result of transforming a Fragment Scope into executable BPEL

5.3 Transformation of Fragment Entry (b4c:fragmentEntry)

The result of the transformation of a *fragmentEntry* into executable BPEL is presented in Listing 9. Please note that the BPEL namespace has changed during the transformation from the abstract one used in the compliance fragments to the executable one, cf. Section 3. Besides the name of the *fragmentEntry* transformed as well as its type are retained in the name attribute of the BPEL empty element and the additional BPEL documentation element added inside the BPEL empty element.

```
xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
<bpel:empty name="The name of the Fragment Entry transformed!">
   <bpel:documentation>
      Fragment Entry, type = the value of the attribute type (mandatory
```

```
      or optional) of the Fragment Entry transformed
   </bpel:documentation>
</bpel:empty>
```

Listing 9 Result of transforming a Fragment Entry into executable BPEL

5.4 Transformation of Fragment Exit (b4c:fragmentExit)

Listing 10 shows the result of transforming a *fragmentExit* into executable BPEL. Please note that the BPEL namespace has changed during the transformation from the abstract one used in the compliance fragments to the executable one, cf. Section 3. Besides the name of the *fragmentExit* transformed as well as its type are retained in the name attribute of the BPEL empty element and the additional BPEL documentation element added inside the BPEL empty element.

```
xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
<bpel:empty name="The name of the Fragment Exit transformed!">
   <bpel:documentation>
      Fragment Exit, type = the value of the attribute type (mandatory
      or optional) of the Fragment Exit transformed
   </bpel:documentation>
</bpel:empty>
```

Listing 10 Result of transforming a Fragment Exit into executable BPEL

5.5 Transformation of Fragment Region (b4c:fragmentRegion)

The result of transforming a *fragmentRegion* into executable BPEL by applying the XSL stylesheet provided in Listing 13 is presented in Listing 11. Please note that the BPEL namespace has changed during the transformation from the abstract one used in the compliance fragments to the executable one, cf. Section 3. Besides the name of the *fragmentRegion* transformed as well as the information, that is has been a *fragmentRegion* are retained in the name attribute of the BPEL empty element and the additional BPEL documentation element added inside the BPEL empty element.

```
xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
<bpel:empty name="The name of the Fragment Region transformed!">
   <bpel:documentation>
      Fragment Region
   </bpel:documentation>
</bpel:empty>
```

Listing 11 Result of transforming a Fragment Region into executable BPEL

5.6 Transformation of Fragment Flow (b4c:fragmentFlow)

Listing 12 shows the result of the transformation of a *fragmentFlow* to executable BPEL. The *fragmentFlow* is transformed to a BPEL flow and the name of the *fragmentFlow* is retained. Please note that the BPEL namespace has changed during the

transformation from the abstract one used in the compliance fragments to the executable one, cf. Section 3. Moreover please take into consideration, that the *fragmentFlow* within a compliance fragment might contain additional BPEL standard elements as well as additional BPEL4CFrags extension elements. In such a case the BPEL standard elements are simply taken from the compliance fragment and added to the transformation result at the according place. Additional BPEL4CFrags extension elements are transformed as described in the Sections 5.1 to 5.6.

```
xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
<bpel:flow name="The name of the Fragment Flow transformed!"
   standard-attributes>
   standard-elements
   <bpel:links>?
      <bpel:link name="NCName">+
   </bpel:links>
   activity+
</bpel:flow>
```

Listing 12 Result of transforming a Fragment Flow into executable BPEL

6 Examples

The examples described in this section originate from the Mobile Virtual Network Operators scenario (WatchMe) [BDL+10] of the FP7 COMPAS project (www.compas-ict.eu, contract no. FP7-215175).

6.1 Example Scenario

In order to better contextualize the examples given in Section 6.2, we will refer to the Mobile Virtual Network Operators (MVNO) scenario (WatchMe) [BDL+10] of the FP7 COMPAS project. In the following we briefly introduce the scenario, highlighting the specific compliance requirements.

The WatchMe scenario focuses on advanced telecom services offered by a MVNO, see Figure 1.

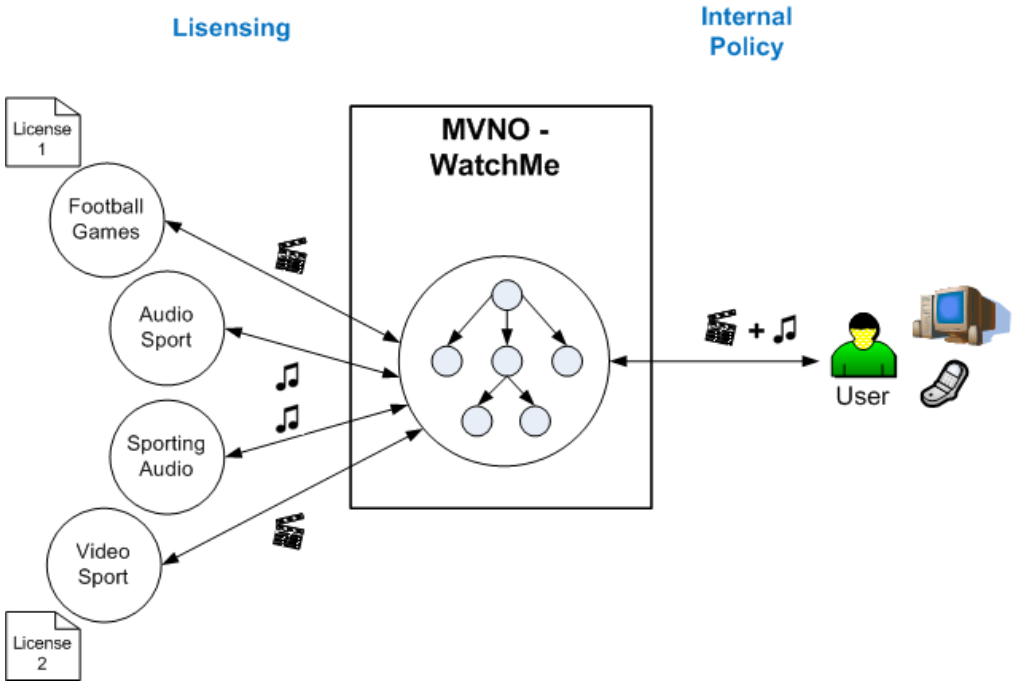

Figure 1 Mobile Virtual Network Operators scenario (WatchMe)

WatchMe is a company offering videos in different languages to customers who have to login or register before to be able to search for available videos in various languages and watch them online afterwards via video streaming.

For providing videos in several languages the MVNO cooperates with video providers named FootballGames and VideoSport and with audio providers named SportingAudio and AudioSport.

The internal policy compliance requirement refers to the communication between the customer and the MVNO WatchMe. The licensing compliance requirements however apply to the interactions between the MVNO WatchMe and the providers, see Figure 1. All compliance requirements impact the business process running within the WatchMe company. Thus, the WatchMe business process has to be compliant to

these compliance requirements. Compliance fragments specified using the BPEL4CFrags extension provide reusable solutions for realizing the concrete compliance requirements.

Table 2 summarizes a selection of the list of compliance requirements used in the WatchMe scenario regarding both internal policy and licensing issues and informally introduces the controls. Due to the terminology used within the COMPAS project a control is a "a statement that describes the restraining or directing influence to check, verify, or enforce rules to satisfy one or more compliance requirement at the business level" [COMPAS Website].

	Compliance Requirements	**Description of Compliance Requirements**	**Control**
Internal Policy	Access to WatchMe service is protected.	The usage of WatchMe service is only allowed for registered users.	A user has to identify himself when interacting with the WatchMe service.
Licensing	Time-based plan	When the WatchMe company subscribes for the Time-based plan it acquires *any* number of times *any* possible streams in a certain period, based on *the amount paid* to the media supplier.	When WatchMe company subscribes for the time-based plan it has to pay **89.90 euro first** and then receive *an unlimited* number of times *any* available stream from the media supplier **in a 30 days** period starting from the contract start date.
	Composition permission	**Only pre-defined combinations** of video and audio providers are allowed due to the licenses specified by the video provider.	**FootballGames** can only have audios streams from **AudioSport** or **SportingAudio**. **VideoSport** can only have audio streams from **AudioSport**.

Table 2 Compliance requirements of the WatchMe scenario

6.2 Example BPEL4CFrags

This section introduces three concrete examples of BPEL4CFrags implementing the compliance requirements from Table 2.

In order to ease understanding of the interested reader we provide (non-normative) graphical representations of the examples exported from the BPEL Eclipse Modeler. Due to the fact, that there is currently no graphical plugin available for any tool sup-

porting the BPEL4CFrags extension the Figure 2, Figure 3 and Figure 4 provide the result of the mapping the corresponding BPEL4CFrags examples to executable BPEL. So the BPEL4CFrags elements contained in the compliance fragments before the transformations are visible as BPEL empty activities in the figures. Additionally the names of the original BPEL4CFrags elements are retained as names of the corresponding BPEL empty activities. Furthermore the former *fragmentEntries* are stressed with blue rectangles; the former *fragmentRegions* are marked with red rectangles and finally the former *fragmentExits* are accented using green rectangles within the (non-normative) graphical representations.

Figure 2 shows the (non-normative) graphical representation of the BPEL4CFrag for implementation of the compliance requirement from domain internal policy. It contains the implemented functionality for user login or user registration as well as using BPEL correlation for asynchronous communication when receiving the media request and the assembled media URL request afterwards. The degrees of freedom in this compliance fragment are represented by the BPEL empty activities stressed using red rectangles, cf. Figure 2. So the business logic for handling the media request and the assembled media URL request as well as the error handling has to be manually implemented by the process modeller during integration phase. Moreover the BPEL empty activity, marked using a green rectangle is the integration point for integration into an already existing BPEL process or for creating one from scratch.

Listing 16 shows the XML representation of the corresponding compliance fragment.

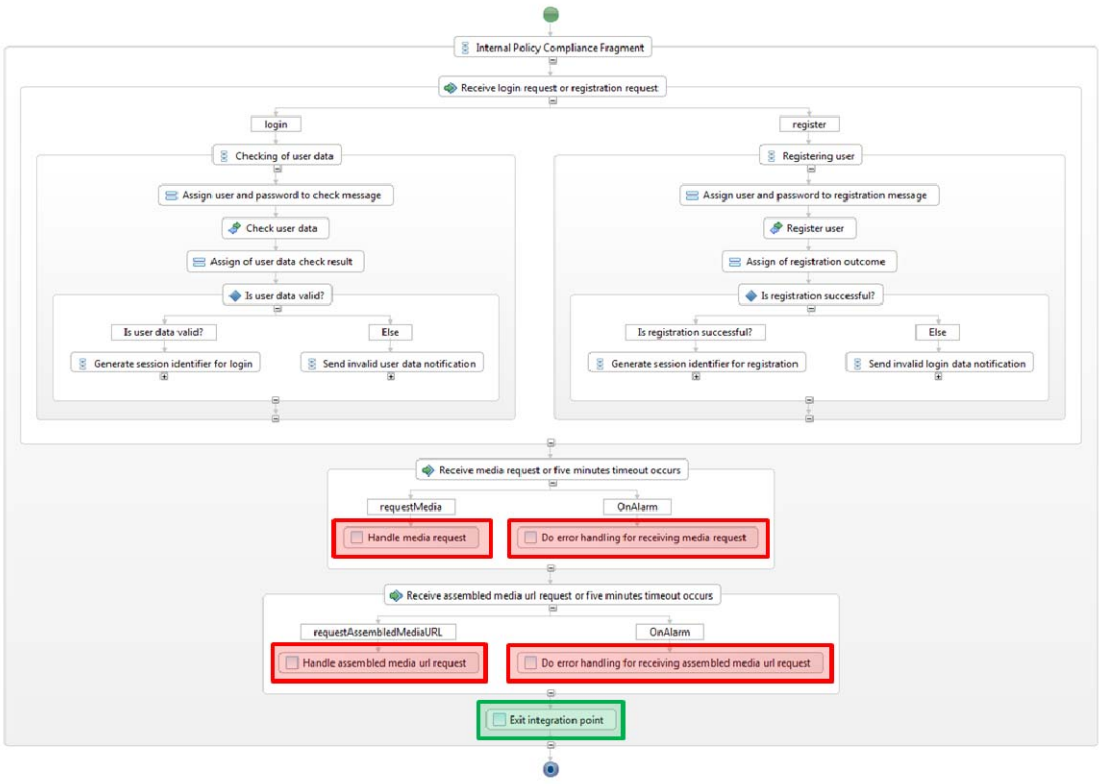

Figure 2 Non-normative Graphical representation of the BPEL4CFrags example for implementing compliance requirement "access to WatchMe service is protected"

The (non-normative) graphical representation of the BPEL4CFrag implementing the compliance requirement time-based plan from domain licensing is provided in Figure 3. For the sake of simplicity we only provide the example focusing on video provider VideoSport. It includes the realization of the functionality for payment check as well as making the payment if time period valid since last payment has expired. Moreover the retrieval of video stream URL for the video stream matching the criteria specified in the assembled media URL request message is implemented. The former *fragmentEntry* is representd by the BPEL empty activity marked using a blue rectangle and named ”Entry integration point”. The degrees of freedom (former *fragmentRegions*) are represented by BPEL empty elements accented with red rectangles, see Figure 3. Thus the logic for handling other video providers as well as the error handling has to be manually implemented by the process modeller during integration phase. The former *fragmentExit* is representd by the BPEL empty activity marked using a green rectangle.

The XML representation of the compliance fragment for implementing time-based plan for video provider VideoSport is provided in Listing 17.

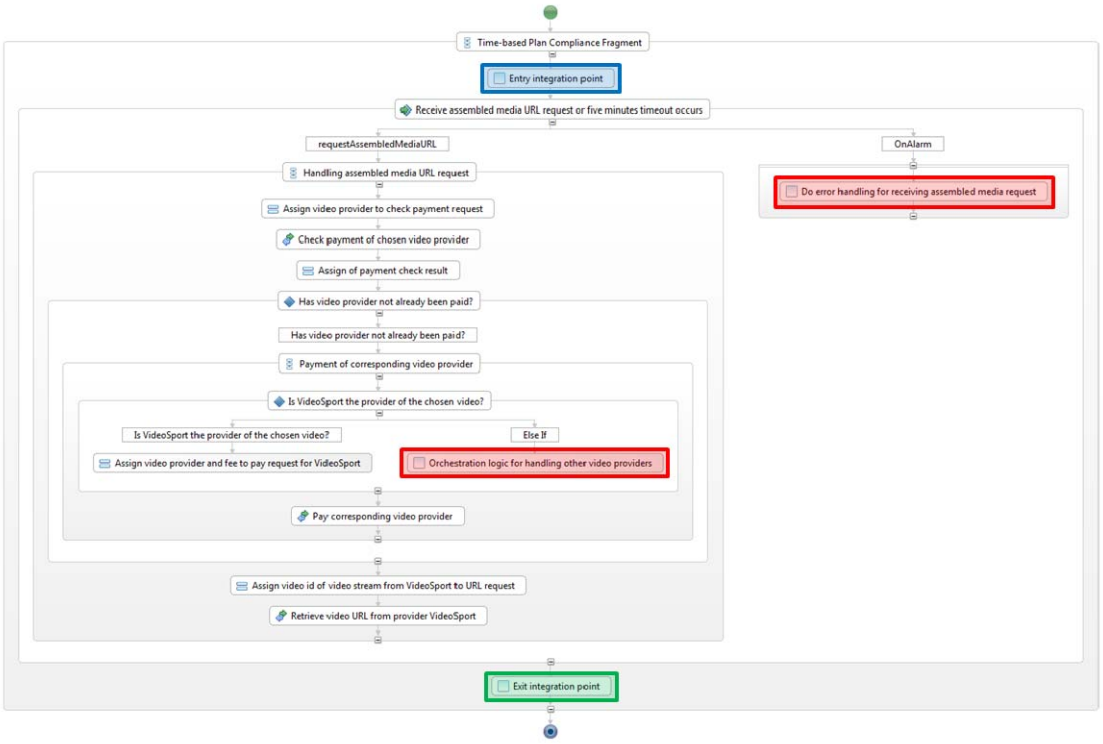

Figure 3 Non-normative graphical representation of the BPEL4CFrags example for implementing compliance requirement “time-based plan for video provider VideoSport”

Figure 4 shows the (non-normative) graphical representation of the BPEL4CFrags example for implementing compliance requirement composition permission for video

provider VideoSport. For the sake of simplicity and limited space we only provide the example focusing on video provider VideoSport.

The compliance fragment includes the realization of the functionality for querying the URLs for the corresponding video and audio matching the criteria in the assembled media URL request. Afterwards the corresponding video is retrieved from provider VideoSport and the corresponding audio is retrieved from AudioSport. Finally, video and audio is assembled into one stream and the URL to this assembled stream is send back to the customer. The former *fragmentEntry* is representd by the BPEL empty activity marked using a blue rectangle. The degrees of freedom (former *fragmentRegions*) are represented by BPEL empty elements accented with red rectangles, see Figure 4. So the logic for handling the payment check, cf. Figure 3, the handling of retrieval of video from provider FootballGames and audios from providers AudioSport or SportingAudio as well as the error handling has to be manually added. The former *fragmentExit* is representd by the BPEL empty activity marked using a green rectangle. The empty activities accented with blue and green rectangles are especially important for the modeller during integration, because here the wiring with an already existing or to be modelled from scratch business process has to be done manually.

The XML representation of the compliance fragment for implementing time-based plan for video provider VideoSport is provided in Listing 18.

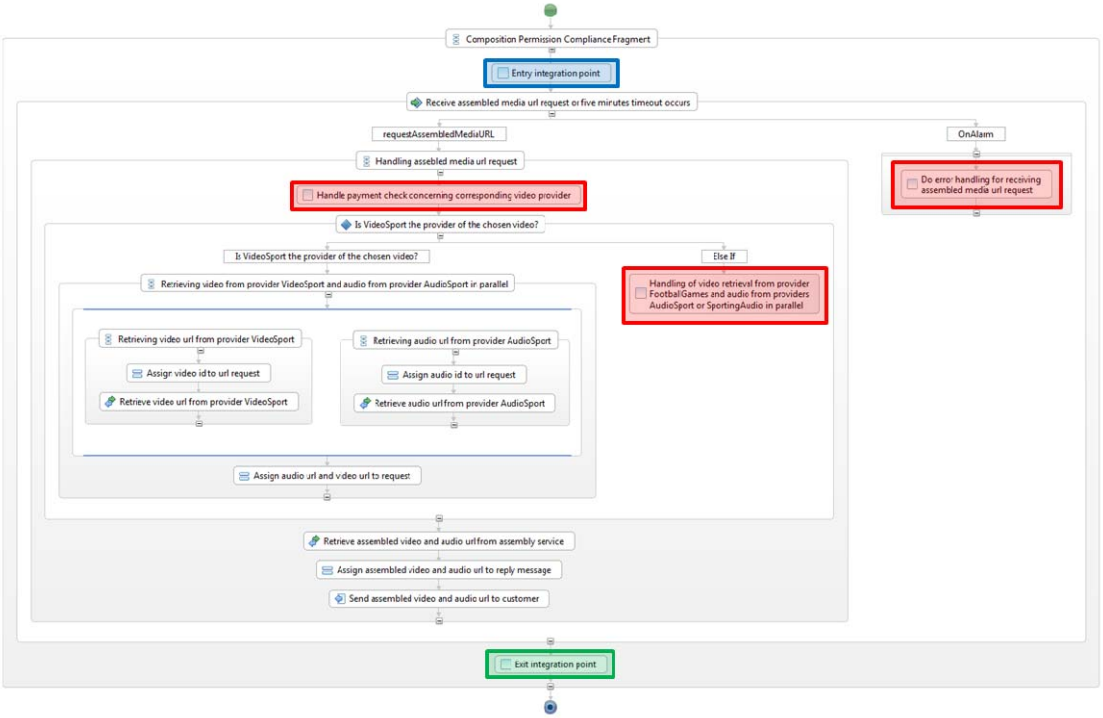

Figure 4 Non-normative graphical representation of the BPEL4CFrags example for implementing compliance requirement "composition permission for video provider VideoSport"

7 Acknowledgements

The work published in this specification was partially funded by the FP7 COMPAS project (www.compas-ict.eu, contract no. FP7-215175).

8 References

[BPMN 2.0]

Business Process Model and Notation Version 2.0 Beta 2, OMG Available Specification, May 2010, Object Management Group, available via http://www.omg.org/spec/BPMN/2.0/Beta2

[ISO/IEC 14977]

Information technology—Syntactic metalanguage—Extended BNF, ISO/IEC 14977 : 1996(E), International Standard, ISO/IEC, 1996, available via http://www.dataip.co.uk/Reference/iso-14977.pdf

[RFC 2119]

Key words for use in RFCs to Indicate Requirement Levels, RFC 2119, available via http://www.ietf.org/rfc/rfc2119.txt

[WS-BPEL 2.0]

Web Service Business Process Execution Language Version 2.0, OASIS Standard, April 2007, OASIS Technical Committee, available via http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[XLink 1.0]

XML Linking Language (XLink) Version 1.0. W3C Recommendation, June 2001, World Wide Web Consortium (W3C), available via http://www.w3.org/TR/xlink/

[XML Namespaces]

Namespaces in XML 1.0 (Second Edition), W3C Recommendation, available via http://www.w3.org/TR/REC-xml-names/

[XPointer]

XPointer xpointer() Scheme, W3C Working Draft, December 2002, World Wide Web Consortium (W3C), available via http://www.w3.org/TR/xptr-xpointer/

[XSLT 1.0]

XSL Transformations (XSLT) Version 1.0, W3C Recommendation, November 1999, OASIS Technical Committee, available via http://www.w3.org/TR/1999/REC-xslt-19991116

9 Non-normative References

[BDL+10]

Birukou, Aliaksandr; D'Andrea, Vincenzo; Leymann, Frank; Serafinski, Jacek; Silveira, Patrícia; Strauch, Steve and Tluczek, Marek: An Integrated Solution for Runtime Compliance Governance in SOA. In: Proceedings of the 8th International Conference on Service-Oriented Computing (ICSOC'10), San Francisco, California, USA, December 7-10, 2010.

[COMPAS D1.3]

MDSD software framework for business compliance, Version 1.0, COMPAS Deliverable, December 2009, COMPAS Consortium, available via http://www.compas-ict.eu/compas_results/deliverables/m23/D1.3_MDSD-software-framework-for-business-compliance.pdf.

[COMPAS D4.2]

BPEL Extensions for Compliant Services, Version 1.0, COMPAS Deliverable, December 2009, COMPAS Consortium, available via http://www.compas-ict.eu/compas_results/deliverables/m23/D4.2_BPEL-extensions-for-compliant-services.pdf.

[COMPAS Website]

COMPAS Terminology, September 2009, COMPAS Consortium, availabel via http://www.compas-ict.eu/terminology.php

[KMW+09]

Kopp, Oliver; Martin, Daniel; Wutke, Daniel and Leymann, Frank: The Difference Between Graph-Based and Block-Structured Business Process Modelling Languages. Enterprise Modelling and Information Systems. Vol. 4(1), pp.3-13, 2009.

[NXSLT3]

Tkachenko, Oleg: NXSLT XSL Transformer, V3.0, 2005, available via http://www.xmllab.net/downloads/nxslt/nxslt-3.0-bin.zip

[SLM+10]

Schumm, David; Leymann, Frank; Ma, Zhilei; Scheibler, Thorsten and Strauch, Steve: Integrating Compliance into Business Processes: Process Fragments as Reusable Compliance Controls. In: Proceedings of the Multikonferenz Wirtschaftsinformatik (MKWI'10), 2010.

[STK+10]

Schumm, David; Turetken, Oktay; Kokash, Natallia; Elgammal, Amal; Leymann, Frank and van den Heuvel, Willem-Jan: Business Process Compliance through Reusable Units of Compliant Processes. In: Proceedings of the 1st Workshop on Engineering SOA and the Web (ESW'10), 2010.

All hyperlinks in this document have been last checked on October 31th 2010.

10 Appendix A – XSL Style Sheet for Transformation

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
   xmlns:b4c=" http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"

   <xsl:output method="xml" version="1.0" encoding="UTF-8"
      indent="yes" />
   <xsl:template match="b4c:fragmentEntry">
      <bpel:empty>
         <xsl:attribute name="name">
            <xsl:value-of select="@name" />
         </xsl:attribute>
         <bpel:documentation>
            Fragment Entry, type =
            <xsl:value-of select="@type" />
         </bpel:documentation>
         <xsl:apply-templates select="*" />
      </bpel:empty>
   </xsl:template>
   <xsl:template match="b4c:fragmentExit">
      <bpel:empty>
         <xsl:attribute name="name">
            <xsl:value-of select="@name" />
         </xsl:attribute>
         <bpel:documentation>
            Fragment Exit, type =
            <xsl:value-of select="@type" />
         </bpel:documentation>
         <xsl:apply-templates select="*" />
      </bpel:empty>
   </xsl:template>
   <xsl:template match="b4c:fragmentRegion">
      <bpel:empty>
         <xsl:attribute name="name">
            <xsl:value-of select="@name" />
         </xsl:attribute>
```

```
         <bpel:documentation>
            Fragment Region
         </bpel:documentation>
         <xsl:apply-templates select="*" />
      </bpel:empty>
   </xsl:template>
   <xsl:template match="frg:fragmentScope">
      <bpel:scope>
         <xsl:attribute name="name">
            <xsl:value-of select="@name" />
         </xsl:attribute>
         <xsl:apply-templates select="*" />
      </bpel:scope>
   </xsl:template>
   <xsl:template match="frg:fragmentFlow">
      <bpel:flow>
         <xsl:attribute name="name">
            <xsl:value-of select="@name" />
         </xsl:attribute>
         <xsl:apply-templates select="*" />
      </bpel:flow>
   </xsl:template>
   <xsl:template match="bpel:extensionActivity">
      <xsl:apply-templates select="*" />
   </xsl:template>
   <xsl:template match="node()">
      <xsl:copy>
         <xsl:copy-of select="@*" />
         <xsl:apply-templates select="*" />
      </xsl:copy>
   </xsl:template>
   <xsl:template match="*">
      <xsl:element name="{local-name()}" namespace="http://docs.oasis-
         open.org/wsbpel/2.0/process/executable">
            <xsl:for-each select="@*|text()">
               <xsl:copy/>
            </xsl:for-each>
            <xsl:apply-templates select="*"/>
      </xsl:element>
   </xsl:template>
</xsl:stylesheet>
```

Listing 13 XSL style sheet for transformation of BPEL4CFragments into standard BPEL code

11 Appendix B – Unique Identifiers XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.iaas.uni-stuttgart.de/ext/id"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://www.iaas.uni-stuttgart.de/ext/id">
   <xsd:attribute name="id" type="xsd:string"/>
</xsd:schema>
```

Listing 14 XML Schema for unique identifiers

12 Appendix C – BPEL4CFrags XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema
   xmlns="http://www.iaas.uni-stuttgart.de/ext/fragment"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
   targetNamespace="http://www.iaas.uni-stuttgart.de/ext/fragment">
   <xsd:import
      namespace=" http://docs.oasis-
         open.org/wsbpel/2.0/process/abstract "
      schemaLocation="bpel.xsd"/>
   <xsd:element name="fragmentExit">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="bpel:targets"/>
         </xsd:sequence>
         <xsd:attribute name="name" type="xsd:string" use="required"/>
         <xsd:attribute name="type" type="tWiring" use="required"/>
         <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="fragmentEntry">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="bpel:sources"/>
         </xsd:sequence>
         <xsd:attribute name="name" type="xsd:string" use="required"/>
         <xsd:attribute name="type" type="tWiring" use="required"/>
         <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="fragmentRegion">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="bpel:targets"/>
            <xsd:element ref="bpel:sources"/>
         </xsd:sequence>
         <xsd:attribute name="name" type="xsd:string" use="required"/>
         <xsd:anyAttribute namespace="##other" processContents="lax"/>
```

```
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="fragmentFlow">
      <xsd:complexType>
         <xsd:complexContent>
            <xsd:extension base="bpel:tFlow"/>
         </xsd:complexContent>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="fragmentScope">
      <xsd:complexType>
         <xsd:complexContent>
            <xsd:extension base="bpel:tScope"/>
         </xsd:complexContent>
      </xsd:complexType>
   </xsd:element>
   <xsd:simpleType name="tWiring">
      <xsd:restriction base="xsd:string">
         <xsd:enumeration value="mandatory"/>
         <xsd:enumeration value="optional"/>
      </xsd:restriction>
   </xsd:simpleType>
</xsd:schema>
```

Listing 15 XML Schema for the WS-BPEL extension for compliance fragments

13 Appendix D – XML Representations of BPEL4CFrags Examples

```
<bpel:process name="WatchMeProcess"
   targetNamespace="http://example.org/bpel4cfrags/watchme"
   xmlns:tns="http://example.org/bpel4cfrags/watchme"
   xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:b4c=" http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"
   xmlns:ext="http://www.iaas.uni-stuttgart.de/ext/id">

   <bpel:documentation xml:lang="EN">
      A simple example of a BPEL4CFrag for implementing the compliance
      requirement access to WatchMe service is protected.
   </bpel:documentation>

   <bpel:extensions>
      <bpel:extension namespace="http://www.iaas.uni-stuttgart.de/
         ext/bpel4cfrags"
         mustUnderstand="yes"/>
      <bpel:extension namespace="http://www.iaas.uni-stuttgart.de/
         ext/id"
         mustUnderstand="no"/>
   </bpel:extensions>

   <bpel:import location="UserDataCheck_withPartnerLinkType.wsdl"
      namespace="http://www.compas-ict.eu/watchme"
      importType="http://schemas.xmlsoap.org/wsdl/" />
   <bpel:import location="SessionIDGeneration_withPartnerLinkType.wsdl"
      namespace="http://www.compas-ict.eu/watchme"
      importType="http://schemas.xmlsoap.org/wsdl/" />
   <bpel:import location="WatchMeProcessArtifacts.wsdl"
      namespace="http://www.compas-ict.eu/watchme"
      importType="http://schemas.xmlsoap.org/wsdl/" />
   <bpel:import namespace="http://www.iaas.uni-
      stuttgart.de/ext/bpel4cfrags"
      location="fragment.xsd"
      importType="http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"/>
   <bpel:import namespace="http://www.iaas.uni-stuttgart.de/ext/id"
      location="id.xsd"
      importType="http://www.iaas.uni-stuttgart.de/ext/id"/>
```

```
<bpel:extensionActivity>
   <b4c:fragmentScope name="fragmentContainer" ext:id="001">
      <bpel:partnerLinks ext:id="002">
         <bpel:partnerLink name="WatchMeProcessPL"
            partnerLinkType="tns:WatchMeProcessPLT"
            myRole="WatchMeProcessProvider" ext:id="003" />
         <bpel:partnerLink name="SessionIDGenerationPL"
            partnerLinkType="tns:SessionIDGenerationPLT"
            partnerRole="SessionIDGenerationProvider"
            initializePartnerRole="yes" ext:id="004" />
         <bpel:partnerLink name="UserDataCheckPL"
            partnerLinkType="tns:UserDataCheckPLT"
            partnerRole="UserDataCheckProvider"
            initializePartnerRole="yes" ext:id="005" />
      </bpel:partnerLinks>

      <bpel:variables ext:id="006">
         <bpel:variable name="input"
            messageType="tns:WatchMeProcessRequestMessage"
            ext:id="007" />
         <bpel:variable name="output"
            messageType="tns:WatchMeProcessResponseMessage"
            ext:id="008" />
         <bpel:variable name="Registrationinput"
            messageType="tns:WatchMeRegisterRequestMessage"
            ext:id="009" />
         <bpel:variable name="Registrationoutput"
            messageType="tns:WatchMeRegisterResponseMessage"
            ext:id="010" />
         <bpel:variable name="sessionIDin"
            messageType="tns:generateSessionIDRequest"
            ext:id="011" />
         <bpel:variable name="sessionIDout"
            messageType="tns:generateSessionIDResponse"
            ext:id="012" />
         <bpel:variable name="UserDataCheckPLResponse"
            messageType="tns:checkUserDataResponse" ext:id="013" />
         <bpel:variable name="UserDataCheckPLRequest"
            messageType="tns:checkUserDataRequest" ext:id="014" />
         <bpel:variable name="RegisterUserRequest"
            messageType="tns:registerUserRequest" ext:id="015" />
```

```
            <bpel:variable name="RegisterUserResponse"
               messageType="tns:registerUserResponse" ext:id="016" />
            <bpel:variable name="UserDataValid" type="xsd:boolean"
               ext:id="017" />
            <bpel:variable name="IsRegistrationSuccessful"
               type="xsd:boolean" ext:id="018" />
            <bpel:variable name="MediaRequestInput"
               messageType="tns:WatchMeMediaRequestMessage"
               ext:id="019" />
            <bpel:variable name="AssembledMediaURLInput"
               messageType="tns:WatchMeAssembledMediaURLRequestMessage"
               ext:id="020" />
         </bpel:variables>

         <bpel:correlationSets ext:id="021">
            <bpel:correlationSet name="CustomerIdentification"
               properties="tns:correlationID" ext:id="022" />
         </bpel:correlationSets>

         <bpel:sequence name="Internal Policy Compliance Fragment"
            ext:id="023">
            <bpel:pick name="Receive login request or registration
               request" createInstance="yes" ext:id="024">
               <bpel:onMessage partnerLink="WatchMeProcessPL"
                  portType="tns:WatchMeProcess" operation="login"
                  variable="input" ext:id="025" >
                  <bpel:sequence name="Checking of user data"
                     ext:id="026">
                     <bpel:assign validate="no" name="Assign user and
                        password to check message" ext:id="027">
                        <bpel:copy ext:id="028">
                           <bpel:from ext:id="029">
                              <bpel:literal xml:space="preserve">
                                 <impl:checkUserData
                                    xmlns:impl="http://www.compas-
                                       ict.eu/watchme"
                                    xmlns:xsi="http://www.w3.org/2001/
                                       XMLSchema-instance">
                                    <impl:user></impl:user>
                                    <impl:passwd></impl:passwd>
                                 </impl:checkUserData>
                              </bpel:literal>
```

```
                        </bpel:from>
                        <bpel:to part="parameters"
                           variable="UserDataCheckPLRequest"
                           ext:id="030">
                        </bpel:to>
                     </bpel:copy>
                     <bpel:copy ext:id="031">
                        <bpel:from part="payload" variable="input"
                           ext:id="032">
                           <bpel:query queryLanguage="urn:oasis:
                              names:tc:wsbpel:2.0:sublang:xpath1.0">
                              <![CDATA[tns:user]]>
                           </bpel:query>
                        </bpel:from>
                     </bpel:copy>
                     <bpel:copy ext:id="034">
                        <bpel:from part="payload" variable="input"
                           ext:id="035">
                           <bpel:query queryLanguage="urn:oasis:
                              names:tc:wsbpel:2.0:sublang:xpath1.0">
                              <![CDATA[tns:passwd]]>
                           </bpel:query>
                        </bpel:from>
                        <bpel:to part="parameters"
                           variable="UserDataCheckPLRequest"
                           ext:id="036">
                           <bpel:query queryLanguage="urn:oasis:
                              names:tc:wsbpel:2.0:sublang:xpath1.0">
                              <![CDATA[tns:passwd]]>
                           </bpel:query>
                        </bpel:to>
                     </bpel:copy>
                  </bpel:assign>
                  <bpel:invoke
                     partnerLink="UserDataCheckPL"
                     operation="checkUserData"
                     portType="tns:UserDataCheck"
                     inputVariable="UserDataCheckPLRequest"
                     outputVariable="UserDataCheckPLResponse"
                     name="Check user data" ext:id="037"/>
                  <bpel:assign validate="no" name="Assign of user
                     data check result" ext:id="038">
                     <bpel:copy ext:id="039">
```

```
                        <bpel:from part="parameters"
                           variable="UserDataCheckPLResponse"
                           ext:id="040">
                           <bpel:query queryLanguage="urn:oasis:
                              names:tc:wsbpel:2.0:sublang:xpath1.0">
                              <![CDATA[tns:checkUserDataReturn]]>
                           </bpel:query>
                        </bpel:from>
                        <bpel:to variable="UserDataValid"
                           ext:id="041"></bpel:to>
                     </bpel:copy>
                  </bpel:assign>
                  <bpel:if name="Is user data valid?" ext:id="042">
                     <bpel:condition ext:id="043">
                        <![CDATA[$UserDataValid="true"]]>
                     </bpel:condition>
                     <bpel:sequence name="Generate session identifier
                        for login" ext:id="044">
                        <bpel:assign validate="yes" name="Initialize
                           variable for session id generation request
                           during login" ext:id="045">
                           <bpel:copy ext:id="046">
                              <bpel:from ext:id="047">
                                 <bpel:literal xml:space="preserve">
                                    <impl:generateSessionID
                                       xmlns:impl="http://www.compas-
                                          ict.eu/watchme"
                                       xmlns:xsi="http://www.w3.org/
                                          2001/XMLSchema-instance">
                                    </impl:generateSessionID>
                                 </bpel:literal>
                              </bpel:from>
                              <bpel:to variable="sessionIDin"
                                 part="parameters" ext:id="048"/>
                           </bpel:copy>
                           <bpel:copy ext:id="049">
                              <bpel:from ext:id="050">
                                 <bpel:literal xml:space="preserve">
                                    <impl:generateSessionIDResponse
                                       xmlns:impl="http://www.compas-
                                          ict.eu/watchme"
                                       xmlns:xsi="http://www.w3.org/
                                          2001/XMLSchema-instance">
```

```
                                    <impl:generateSessionIDReturn>
                                    </impl:generateSessionIDReturn
                                       >
                                 </impl:generateSessionIDResponse>
                              </bpel:literal>
                           </bpel:from>
                           <bpel:to variable="sessionIDout"
                              part="parameters"
                              ext:id="051">
                           </bpel:to>
                        </bpel:copy>
                        <bpel:copy ext:id="052">
                           <bpel:from ext:id="053">
                              <bpel:literal xml:space="preserve">
                                 <tns:WatchMeProcessResponse
                                    xmlns:tns="http://www.compas-
                                       ict.eu/watchme"
                                    xmlns:xsi="http://www.w3.org/
                                       2001/XMLSchema-instance">
                                       <tns:result></tns:result>
                                 </tns:WatchMeProcessResponse>
                              </bpel:literal>
                           </bpel:from>
                           <bpel:to variable="output"
                              part="payload"
                              ext:id="054">
                           </bpel:to>
                        </bpel:copy>
                     </bpel:assign>
                     <bpel:invoke name="Generate unique login
                        session identifier"
                        partnerLink="SessionIDGenerationPL"
                        operation="generateSessionID"
                        portType="tns:SessionIDGeneration"
                        inputVariable="sessionIDin"
                        outputVariable="sessionIDout"
                        ext:id="055">
                     </bpel:invoke>
                     <bpel:assign validate="no" name="Copy unique
                        session identifier to login output
                        variable" ext:id="056">
                        <bpel:copy ext:id="057">
```

```
                        <bpel:from ext:id="058">
                           <bpel:literal xml:space="preserve">
                              <tns:WatchMeProcessResponse
                                 xmlns:tns="http://www.compas-
                                    ict.eu/watchme"
                                 xmlns:xsi="http://www.w3.org/
                                    2001/XMLSchema-instance">
                                 <tns:result></tns:result>
                              </tns:WatchMeProcessResponse>
                           </bpel:literal>
                        </bpel:from>
                        <bpel:to variable="output"
                           part="payload" ext:id="059"/>
                     </bpel:copy>
                     <bpel:copy ext:id="060">
                        <bpel:from part="parameters"
                           variable="sessionIDout"
                           ext:id="061">
                           <bpel:query queryLanguage="urn:
                              oasis:names:tc:wsbpel:2.0:
                              sublang:xpath1.0">
                              <![CDATA[
                                 tns:generateSessionIDReturn]]>
                           </bpel:query>
                        </bpel:from>
                        <bpel:to part="payload"
                           variable="output" ext:id="062">
                           <bpel:query queryLanguage="urn:
                              oasis:names:tc:wsbpel:2.0:
                              sublang:xpath1.0">
                              <![CDATA[tns:result]]>
                           </bpel:query>
                        </bpel:to>
                     </bpel:copy>
                  </bpel:assign>
                  <bpel:reply name="Send session id to customer
                     after login"
                     partnerLink="WatchMeProcessPL"
                     operation="login"
                     portType="tns:WatchMeProcess"
                     variable="output" ext:id="063">
                     <bpel:correlations ext:id="064">
                        <bpel:correlation
```

```
                        set="CustomerIdentification"
                        initiate="yes" ext:id="065"/>
                     </bpel:correlations>
                  </bpel:reply>
               </bpel:sequence>
            <bpel:else name="User data invalid" ext:id="066">
               <bpel:sequence name="Send invalid user data
                  notification" ext:id="067">
                  <bpel:assign validate="yes" name="Assign
                     failed login info to notification message"
                     ext:id="068">
                     <bpel:copy ext:id="069">
                        <bpel:from ext:id="070">
                           <bpel:literal xml:space="preserve">
                              <tns:WatchMeProcessResponse
                                 xmlns:tns="http://www.compas-
                                    ict.eu/watchme"
                                 xmlns:xsi="http://www.w3.org/
                                   2001/XMLSchema-instance">
                                 <tns:result>Invalid user
                                    data</tns:result>
                              </tns:WatchMeProcessResponse>
                           </bpel:literal>
                        </bpel:from>
                        <bpel:to variable="output"
                           part="payload"
                           ext:id="071">
                        </bpel:to>
                     </bpel:copy>
                  </bpel:assign>
                  <bpel:reply name="Send notification regarding
                     failed login to customer"
                     partnerLink="WatchMeProcessPL"
                     operation="login"
                     portType="tns:WatchMeProcess"
                     variable="output" ext:id="072"/>
                  <bpel:exit name="End process after sending
                     invalid login data notification"
                     ext:id="073">
                  </bpel:exit>
               </bpel:sequence>
            </bpel:else>
            </bpel:if>
```

```
            </bpel:sequence>
         </bpel:onMessage>
         <bpel:onMessage
            partnerLink="WatchMeProcessPL"
            portType="tns:WatchMeProcess"
            operation="register"
            variable="Registrationinput"
            ext:id="074" >
            <bpel:sequence name="Registering user" ext:id="075">
               <bpel:assign validate="no" name="Assign user and
                  password to registration message" ext:id="076">
                  <bpel:copy ext:id="077">
                     <bpel:from ext:id="078">
                        <bpel:literal xml:space="preserve">
                           <impl:registerUser
                              xmlns:impl="http://www.compas-
                                 ict.eu/watchme"
                              xmlns:xsi="http://www.w3.org/2001/
                                 XMLSchema-instance">
                              <impl:user></impl:user>
                              <impl:passwd></impl:passwd>
                           </impl:registerUser>
                        </bpel:literal>
                     </bpel:from>
                     <bpel:to variable="RegisterUserRequest"
                       part="parameters" ext:id="079">
                     </bpel:to>
                  </bpel:copy>
                  <bpel:copy ext:id="080">
                     <bpel:from part="payload"
                        variable="Registrationinput"
                        ext:id="081">
                        <bpel:query queryLanguage="urn:oasis:
                           names:tc:wsbpel:2.0:sublang:
                           xpath1.0">
                           <![CDATA[tns:user]]>
                        </bpel:query>
                     </bpel:from>
                     <bpel:to part="parameters"
                        variable="RegisterUserRequest"
                        ext:id="082">
                        <bpel:query queryLanguage="urn:oasis:
                           names:tc:wsbpel:2.0:sublang:
```

```
                        xpath1.0">
                        <![CDATA[tns:user]]>
                     </bpel:query>
                  </bpel:to>
               </bpel:copy>
               <bpel:copy ext:id="083">
                  <bpel:from part="payload"
                     variable="Registrationinput"
                     ext:id="084">
                     <bpel:query queryLanguage="urn:oasis:
                        names:tc:wsbpel:2.0:sublang:
                        xpath1.0">
                        <![CDATA[tns:passwd]]>
                     </bpel:query>
                  </bpel:from>
                  <bpel:to part="parameters"
                     variable="RegisterUserRequest"
                     ext:id="085">
                     <bpel:query queryLanguage="urn:oasis:
                        names:tc:wsbpel:2.0:sublang:
                        xpath1.0">
                        <![CDATA[tns:passwd]]>
                     </bpel:query>
                  </bpel:to>
               </bpel:copy>
            </bpel:assign>
            <bpel:invoke
               partnerLink="UserDataCheckPL"
               operation="registerUser"
               portType="tns:UserDataCheck"
               inputVariable="RegisterUserRequest"
               outputVariable="RegisterUserResponse"
               name="Register user" ext:id="086"/>
            <bpel:assign validate="no" name="Assign of
               registration outcome" ext:id="087">
               <bpel:copy ext:id="088">
                  <bpel:from part="parameters"
                     variable="RegisterUserResponse"
                     ext:id="089">
                     <bpel:query queryLanguage="urn:oasis:
                        names:tc:wsbpel:2.0:sublang:
                        xpath1.0">
```

```
                        <![CDATA[tns:registerUserReturn]]>
                     </bpel:query>
                  </bpel:from>
                  <bpel:to
                     variable="IsRegistrationSuccessful"
                     ext:id="090">
                  </bpel:to>
               </bpel:copy>
            </bpel:assign>
            <bpel:if name="Is registration successful?"
               ext:id="091">
               <bpel:condition ext:id="092">
                  <![CDATA[$IsRegistrationSuccessful=
                     "true"]]>
               </bpel:condition>
               <bpel:sequence name="Generate session
                  identifier for registration" ext:id="093">
                  <bpel:assign validate="yes"
                     name="Initialize variable for session
                        id generation request during
                        registration" ext:id="094">
                     <bpel:copy ext:id="095">
                        <bpel:from ext:id="096">
                           <bpel:literal
                              xml:space="preserve">
                              <impl:generateSessionID
                                 xmlns:impl="http://www.
                                    compas-ict.eu/watchme"
                                 xmlns:xsi="http://www.w3.
                                    org/2001/XMLSchema
                                    instance">
                              </impl:generateSessionID>
                           </bpel:literal>
                        </bpel:from>
                        <bpel:to variable="sessionIDin"
                           part="parameters" ext:id="097"/>
                     </bpel:copy>
                     <bpel:copy ext:id="098">
                        <bpel:from ext:id="099">
                           <bpel:literal
                              xml:space="preserve">
                              <impl:generateSessionIDResponse
                                 xmlns:impl="http://www.
```

```
                                    compas-ict.eu/watchme"
                                 xmlns:xsi="http://www.w3.org/
                                    2001/XMLSchema-instance">
                                    <impl:generateSession
                                       IDReturn>
                                    </impl:generateSession
                                       IDReturn>
                              </impl:generateSessionIDResponse>
                           </bpel:literal>
                        </bpel:from>
                        <bpel:to variable="sessionIDout"
                           part="parameters" ext:id="100">
                        </bpel:to>
                     </bpel:copy>
                     <bpel:copy ext:id="101">
                        <bpel:from ext:id="102">
                           <bpel:literal
                              xml:space="preserve">
                              <tns:WatchMeProcessResponse
                              xmlns:tns="http://www.
                                 compas-ict.eu/watchme"
                              xmlns:xsi="http://www.w3.
                                 org/2001/XMLSchema-instance">
                                 <tns:result>
                                 </tns:result>
                              </tns:WatchMeProcessResponse>
                           </bpel:literal>
                        </bpel:from>
                        <bpel:to variable="output"
                           part="payload" ext:id="103">
                        </bpel:to>
                     </bpel:copy>
                  </bpel:assign>
                  <bpel:invoke name="Generate unique
                     registration session identifier"
                     partnerLink="SessionIDGenerationPL"
                     operation="generateSessionID"
                     portType="tns:SessionIDGeneration"
                     inputVariable="sessionIDin"
                     outputVariable="sessionIDout"
                     ext:id="104">
                  </bpel:invoke>
```

```
                        <bpel:assign validate="no"
                           name="Copy unique session identifier to
                              register output variable" ext:id="105">
                           <bpel:copy ext:id="106">
                              <bpel:from ext:id="107">
                                 <bpel:literal
                                    xml:space="preserve">
                                    <tns:WatchMeRegisterResponse
                                       xmlns:tns="http://www.
                                          compas-ict.eu/watchme"
                                       xmlns:xsi="http://www.w3.
                                          org/2001/XMLSchema
                                          instance">
                                          <tns:result>
                                          </tns:result>
                                    </tns:WatchMeRegisterResponse>
                                 </bpel:literal>
                              </bpel:from>
                              <bpel:to part="payload"
                                 variable="Registrationoutput"
                                 ext:id="108"/>
                           </bpel:copy>
                           <bpel:copy ext:id="109">
                              <bpel:from part="parameters"
                                 variable="sessionIDout"
                                 ext:id="110">
                                 <bpel:query queryLanguage="urn:
                                    oasis:names:tc:wsbpel:2.0:
                                    sublang:xpath1.0">
                                    <![CDATA[tns:generateSession
                                       IDReturn]]>
                                 </bpel:query>
                              </bpel:from>
                              <bpel:to part="payload"
                                 variable="Registrationoutput"
                                 ext:id="111">
                                 <bpel:query queryLanguage="urn:
                                    oasis:names:tc:wsbpel:2.0:
                                    sublang:xpath1.0">
                                    <![CDATA[tns:result]]>
                                 </bpel:query>
                              </bpel:to>
                           </bpel:copy>
```

```
                        </bpel:assign>
                        <bpel:reply name="Send session id to
                           customer after registration"
                           partnerLink="WatchMeProcessPL"
                           operation="register"
                           portType="tns:WatchMeProcess"
                           variable="Registrationoutput"
                           ext:id="112">
                           <bpel:correlations ext:id="113">
                              <bpel:correlation
                                 set="CustomerIdentification"
                                 initiate="yes" ext:id="114"/>
                           </bpel:correlations>
                        </bpel:reply>
                     </bpel:sequence>
                  <bpel:else name="Login data invalid"
                     ext:id="115">
                     <bpel:sequence name="Send invalid login data
                        notification" ext:id="116">
                        <bpel:assign validate="yes" name="Assign
                           failed registration info to
                           notification message" ext:id="117">
                           <bpel:copy ext:id="118">
                              <bpel:from ext:id="119">
                                 <bpel:literal
                                    xml:space="preserve">
                                    <tns:WatchMeRegisterResponse
                                       xmlns:tns="http://www.
                                          compas-ict.eu/watchme"
                                       xmlns:xsi="http://www.w3.
                                          org/2001/XMLSchema
                                          instance">
                                       <tns:result>Invalid user
                                          data or user already
                                          registered
                                       </tns:result>
                                    </tns:WatchMeRegisterResponse>
                                 </bpel:literal>
                              </bpel:from>
                              <bpel:to part="payload"
                                 variable="Registrationoutput"
                                 ext:id="120">
```

```
                        </bpel:to>
                     </bpel:copy>
                  </bpel:assign>
                  <bpel:reply name="Send notification
                     regarding failed registration to
                     customer"
                     partnerLink="WatchMeProcessPL"
                     operation="register"
                     portType="tns:WatchMeProcess"
                     variable="Registrationoutput"
                     ext:id="121"/>
                  <bpel:exit name="End process after sending
                     invalid registration data notification"
                     ext:id="122">
                  </bpel:exit>
               </bpel:sequence>
            </bpel:else>
            </bpel:if>
         </bpel:sequence>
      </bpel:onMessage>
   </bpel:pick>
   <bpel:pick name="Receive media request or five minutes
      timeout occurs" createInstance="no" ext:id="123">
      <bpel:onMessage partnerLink="WatchMeProcessPL"
         portType="tns:WatchMeProcess"
         operation="requestMedia"
         variable="MediaRequestInput"
         ext:id="124" >
         <bpel:correlations ext:id="125">
            <bpel:correlation set="CustomerIdentification"
               initiate="no" ext:id="126"/>
         </bpel:correlations>
         <bpel:extensionActivity ext:id="127">
            <b4c:fragmentRegion  name="Handle media request"
               ext:id="128">
            </b4c:fragmentRegion>
         </bpel:extensionActivity>
      </bpel:onMessage>
      <bpel:onAlarm ext:id="129">
         <bpel:for ext:id="130">
            <![CDATA['PT5M']]>
```

```
                </bpel:for>
                <bpel:extensionActivity ext:id="131">
                   <b4c:fragmentRegion  name="Do error handling for
                      receiving media request" ext:id="132">
                   </b4c:fragmentRegion>
                </bpel:extensionActivity>
             </bpel:onAlarm>
          </bpel:pick>
          <bpel:pick
             name="Receive assembled media URL request or five minutes
                timeout occurs"
             createInstance="no"
             ext:id="133">
             <bpel:onMessage
                partnerLink="WatchMeProcessPL"
                portType="tns:WatchMeProcess"
                operation="requestAssembledMediaURL"
                variable="AssembledMediaURLInput"
                ext:id="134">
                <bpel:correlations ext:id="135">
                   <bpel:correlation set="CustomerIdentification"
                      initiate="no" ext:id="136" />
                </bpel:correlations>
                <bpel:extensionActivity ext:id="137">
                   <b4c:fragmentRegion name="Handle assembled media
                      URL request" ext:id="138">
                   </b4c:fragmentRegion>
                </bpel:extensionActivity>
             </bpel:onMessage>
             <bpel:onAlarm ext:id="139">
                <bpel:for ext:id="140">
                   <![CDATA['PT5M']]>
                </bpel:for>
                <bpel:extensionActivity ext:id="141">
                   <b4c:fragmentRegion  name="Do error handling for
                      receiving assembled media URL request"
                      ext:id="142">
                   </b4c:fragmentRegion>
                </bpel:extensionActivity>
             </bpel:onAlarm>
          </bpel:pick>
          <bpel:extensionActivity ext:id="143">
             <b4c:fragmentExit name="Exit integration point"
```

```
                type="mandatory" ext:id="144">
            </b4c:fragmentExit>
         </bpel:extensionActivity>
      </bpel:sequence>
   </b4c:fragmentScope>
</bpel:extensionActivity>
</bpel:process>
```

Listing 16 XML representation of BPEL4CFrags example for implementing compliance requirement access to WatchMe service is protected

```
<bpel:process name="WatchMeProcess"
   targetNamespace="http://example.org/bpel4cfrags/watchme"
   xmlns:tns="http://example.org/bpel4cfrags/watchme"
   xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:b4c=" http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"
   xmlns:ext="http://www.iaas.uni-stuttgart.de/ext/id">

   <bpel:documentation xml:lang="EN">
      A simple example of a BPEL4CFrag for implementing the compliance
      requirement time-based plan for the concrete video provider
      VideoSport.
   </bpel:documentation>

   <bpel:extensions>
      <bpel:extension namespace="http://www.iaas.uni-stuttgart.de/
         ext/bpel4cfrags"
         mustUnderstand="yes"/>
      <bpel:extension namespace="http://www.iaas.uni-stuttgart.de/
         ext/id"
         mustUnderstand="no"/>
   </bpel:extensions>

   <bpel:import location="WatchMeProcessArtifacts.wsdl"
      namespace="http://www.compas-ict.eu/watchme"
      importType="http://schemas.xmlsoap.org/wsdl/" />
   <bpel:import location="VideoSport_withPartnerLinkType.wsdl"
      namespace="http://www.compas-ict.eu/watchme"
      importType="http://schemas.xmlsoap.org/wsdl/" />
   <bpel:import location="Bank_withPartnerLinkType.wsdl"
      namespace="http://www.compas-ict.eu/watchme"
      importType="http://schemas.xmlsoap.org/wsdl/" />
```

```
<bpel:import namespace="http://www.iaas.unistuttgart.de/
   ext/bpel4cfrags"
   location="fragment.xsd"
   importType="http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"/>
<bpel:import namespace="http://www.iaas.uni-stuttgart.de/ext/id"
   location="id.xsd"
   importType="http://www.iaas.uni-stuttgart.de/ext/id"/>

<bpel:extensionActivity>
   <b4c:fragmentScope name="fragmentContainer" ext:id="001">
      <bpel:partnerLinks ext:id="002">
         <bpel:partnerLink
            name="WatchMeProcessPL"
            partnerLinkType="tns:WatchMeProcessPLT"
            myRole="WatchMeProcessProvider" ext:id="003"/>
         <bpel:partnerLink
            name="VideoSportPL"
            partnerLinkType="tns:VideoSportPLT"
            partnerRole="VideoSportProvider"
            initializePartnerRole="yes"
            ext:id="004"/>
         <bpel:partnerLink
            name="BankPL"
            partnerLinkType="tns:BankPLT"
            partnerRole="BankProvider"
            initializePartnerRole="yes"
            ext:id="005"/>
      </bpel:partnerLinks>

      <bpel:variables ext:id="006">
         <bpel:variable name="VideosFromVideoSportRequest"
            messageType="tns:requestVideosRequest" ext:id="007"/>
         <bpel:variable name="VideosFromVideoSportResponse"
            messageType="tns:requestVideosResponse" ext:id="008"/>
         <bpel:variable name="AssembledMediaURLInput"
            messageType="tns:WatchMeAssembledMediaURLRequest
               Message" ext:id="009"/>
         <bpel:variable name="AssembledMediaURLOutput"
            messageType="tns:WatchMeAssembledMediaURLResponse
               Message" ext:id="010"/>
         <bpel:variable name="CheckPaymentRequest"
            messageType="tns:checkPaymentRequest" ext:id="011"/>
         <bpel:variable name="CheckPaymentResponse"
```

```
            messageType="tns:checkPaymentResponse" ext:id="012"/>
         <bpel:variable name="HasVideoProviderBeenPaid"
            type="xsd:boolean" ext:id="013"/>
         <bpel:variable name="VideoProvider" type="xsd:string"
            ext:id="014"/>
         <bpel:variable name="PayRequest"
            messageType="tns:payRequest" ext:id="015"/>
         <bpel:variable name="PayResponse"
            messageType="tns:payResponse" ext:id="016"/>
         <bpel:variable name="VideoSportURLRequest"
            messageType="tns:requestVideoURLRequest"
            ext:id="017"/>
         <bpel:variable name="VideoSportURLResponse"
            messageType="tns:requestVideoURLResponse"
            ext:id="018"/>
      </bpel:variables>

      <bpel:correlationSets ext:id="019">
         <bpel:correlationSet name="CustomerIdentification"
            properties="tns:correlationID" ext:id="020"/>
      </bpel:correlationSets>

      <bpel:sequence name="Time-based Plan Compliance Fragment"
         ext:id="021">
         <bpel:extensionActivity ext:id="022">
            <b4c:fragmentEntry name="Entry integration point"
               type="mandatory" ext:id="23">
            </b4c:fragmentEntry>
         </bpel:extensionActivity>
         <bpel:pick name="Receive assembled media URL request or
            five minutes timeout occurs" createInstance="no"
            ext:id="024">
            <bpel:onMessage
               partnerLink="WatchMeProcessPL"
               portType="tns:WatchMeProcess"
               operation="requestAssembledMediaURL"
               variable="AssembledMediaURLInput"
               ext:id="025">
               <bpel:correlations ext:id="026">
                  <bpel:correlation set="CustomerIdentification"
                     initiate="no" ext:id="027"/>
               </bpel:correlations>
```

```
<bpel:sequence name="Handling assembled media URL
   request" ext:id="028">
   <bpel:assign validate="no" name="Assign video
      provider to check payment request"
      ext:id="029">
      <bpel:copy ext:id="030">
         <bpel:from ext:id="031">
            <bpel:literal xml:space="preserve">
               <impl:checkPayment
                  xmlns:impl="http://www.compasict.eu/
                     watchme"
                  xmlns:xsi="http://www.w3.org/2001/
                     XMLSchema-instance">
                  <impl:videoProvider>
                  </impl:videoProvider>
               </impl:checkPayment>
            </bpel:literal>
         </bpel:from>
         <bpel:to part="parameters"
            variable="CheckPaymentRequest"
            ext:id="032">
         </bpel:to>
      </bpel:copy>
      <bpel:copy ext:id="033">
         <bpel:from part="payload"
            variable="AssembledMediaURLInput"
            ext:id="034">
            <bpel:query queryLanguage="urn:oasis:
               names:tc:wsbpel:2.0:sublang:
               xpath1.0">
               <![CDATA[tns:videoProvider]]>
            </bpel:query>
         </bpel:from>
         <bpel:to part="parameters"
            variable="CheckPaymentRequest"
            ext:id="035">
            <bpel:query queryLanguage="urn:oasis:
               names:tc:wsbpel:2.0:sublang:xpath1.0">
               <![CDATA[tns:videoProvider]]>
            </bpel:query>
         </bpel:to>
      </bpel:copy>
   </bpel:assign>
```

```
                    <bpel:invoke
                       partnerLink="BankPL"
                       operation="checkPayment"
                       portType="tns:Bank"
                       inputVariable="CheckPaymentRequest"
                       outputVariable="CheckPaymentResponse"
                       name="Check payment of chosen video provider"
                       ext:id="036"/>
                    <bpel:assign validate="no" name="Assign of
                       payment check result" ext:id="037">
                       <bpel:copy ext:id="038">
                          <bpel:from part="parameters"
                             variable="CheckPaymentResponse"
                             ext:id="039">
                             <bpel:query queryLanguage="urn:oasis:
                                names:tc:wsbpel:2.0:sublang:
                                xpath1.0">
                                <![CDATA[tns:checkPaymentReturn]]>
                             </bpel:query>
                          </bpel:from>
                          <bpel:to
                             variable="HasVideoProviderBeenPaid"
                             ext:id="040">
                          </bpel:to>
                       </bpel:copy>
                       <bpel:copy ext:id="041">
                          <bpel:from part="payload"
                             variable="AssembledMediaURLInput"
                             ext:id="042">
                             <bpel:query queryLanguage="urn:oasis:
                                names:tc:wsbpel:2.0:sublang:
                                xpath1.0">
                                <![CDATA[tns:videoProvider]]>
                             </bpel:query>
                          </bpel:from>
                          <bpel:to variable="VideoProvider"
                             ext:id="043">
                          </bpel:to>
                       </bpel:copy>
                       <bpel:copy ext:id="044">
                          <bpel:from part="payload"
                             variable="AssembledMediaURLInput"
                             ext:id="045">
```

```
                        <bpel:query queryLanguage="urn:oasis:
                           names:tc:wsbpel:2.0:sublang:
                           xpath1.0">
                           <![CDATA[tns:audioProvider]]>
                        </bpel:query>
                     </bpel:from>
                     <bpel:to variable="AudioProvider"
                        ext:id="046">
                     </bpel:to>
                  </bpel:copy>
               </bpel:assign>
               <bpel:if name="Has video provider not already
                  been paid?" ext:id="047">
                  <bpel:condition ext:id="048">
                     <![CDATA[$HasVideoProviderBeenPaid="false"]]>
                  </bpel:condition>
                  <bpel:sequence name="Payment of corresponding
                     video provider" ext:id="049">
                     <bpel:if name="Is VideoSport the provider
                        of the chosen video?" ext:id="050">
                        <bpel:condition ext:id="051">
                           <![CDATA[$VideoProvider="VideoSport"]]>
                        </bpel:condition>
                        <bpel:assign validate="no"
                           name="Assign video provider and fee
                              to pay request for VideoSport"
                           ext:id="052">
                           <bpel:copy ext:id="053">
                              <bpel:from ext:id="054">
                                 <bpel:literal
                                    xml:space="preserve">
                                    <impl:pay
                                       xmlns:impl="http://www.
                                          compas-ict.eu/watchme"
                                       xmlns:xsi="http://www.w3
                                          .org/2001/XMLSchema
                                          instance">
                                       <impl:videoProviderID>
                                       </impl:videoProviderID>
                                       <impl:fee>
                                       </impl:fee>
                                    </impl:pay>
                                 </bpel:literal>
```

```
            </bpel:from>
            <bpel:to part="parameters"
               variable="PayRequest"
               ext:id="055">
            </bpel:to>
         </bpel:copy>
         <bpel:copy ext:id="056">
            <bpel:from part="payload"
               variable="AssembledMediaURLInput"
               ext:id="057">
               <bpel:query
                  queryLanguage="urn:oasis:
                  names:tc:wsbpel:2.0:sublang:
                  xpath1.0">
                  <![CDATA[tns:videoProvider]]>
               </bpel:query>
            </bpel:from>
            <bpel:to part="parameters"
               variable="PayRequest"
               ext:id="058">
               <bpel:query queryLanguage="urn:
                  oasis:names:tc:wsbpel:2.0:
                  sublang:xpath1.0">
                  <![CDATA[tns:video
                     ProviderID]]>
               </bpel:query>
            </bpel:to>
         </bpel:copy>
         <bpel:copy ext:id="059">
            <bpel:from ext:id="060">
               <bpel:literal
                  xml:space="preserve">89.90
               </bpel:literal>
            </bpel:from>
            <bpel:to part="parameters"
               variable="PayRequest"
               ext:id="061">
               <bpel:query
                  queryLanguage="urn:oasis:
                     names:tc:wsbpel:2.0:
                     sublang:xpath1.0">
                  <![CDATA[tns:fee]]>
               </bpel:query>
```

```
                        </bpel:to>
                     </bpel:copy>
                  </bpel:assign>
               <bpel:elseif name="Handling of other
                  video providers" ext:id="062">
                  <bpel:condition ext:id="063">
                     <![CDATA[$VideoProvider=
                        "FootballGames"]]>
                  </bpel:condition>
                  <bpel:extensionActivity ext:id="064">
                     <b4c:fragmentRegion name="Orchestration
                        logic for handling other video
                        providers" ext:id="065">
                     </b4c:fragmentRegion>
                  </bpel:extensionActivity>
               </bpel:elseif>
               </bpel:if>
               <bpel:invoke partnerLink="BankPL"
                  operation="pay"
                  portType="tns:Bank"
                  inputVariable="PayRequest"
                  outputVariable="PayResponse"
                  name="Pay corresponding video provider"
                  ext:id="066"/>
            </bpel:sequence>
         </bpel:if>
         <bpel:assign validate="no"
            name="Assign video id of video stream from
               VideoSport to URL request"
            ext:id="067">
            <bpel:copy ext:id="068">
               <bpel:from ext:id="069">
                  <bpel:literal xml:space="preserve">
                     <impl:requestVideoURL
                        xmlns:impl="http://www.compas-ict.
                           eu/watchme"
                        xmlns:xsi="http://www.w3.org/2001/
                           XMLSchema-instance">
                        <impl:videoID>
                        </impl:videoID>
                     </impl:requestVideoURL>
                  </bpel:literal>
               </bpel:from>
```

```
                        <bpel:to part="parameters"
                           variable="VideoSportURLRequest"
                           ext:id="070">
                        </bpel:to>
                     </bpel:copy>
                     <bpel:copy ext:id="071">
                        <bpel:from part="payload"
                           variable="AssembledMediaURLInput"
                           ext:id="072">
                           <bpel:query queryLanguage="urn:oasis:
                              names:tc:wsbpel:2.0:sublang:
                              xpath1.0">
                              <![CDATA[tns:videoID]]>
                           </bpel:query>
                        </bpel:from>
                        <bpel:to part="parameters"
                           variable="VideoSportURLRequest"
                           ext:id="073">
                           <bpel:query queryLanguage="urn:oasis:
                              names:tc:wsbpel:2.0:sublang:
                              xpath1.0">
                              <![CDATA[tns:videoID]]>
                           </bpel:query>
                        </bpel:to>
                     </bpel:copy>
                  </bpel:assign>
                  <bpel:invoke partnerLink="VideoSportPL"
                     operation="requestVideoURL"
                     portType="tns:VideoSport"
                     inputVariable="VideoSportURLRequest"
                     outputVariable="VideoSportURLResponse"
                     name="Retrieve video URL from provider
                        VideoSport"
                     ext:id="074"/>
               </bpel:sequence>
            </bpel:onMessage>
            <bpel:onAlarm ext:id="075">
               <bpel:for ext:id="076"><![CDATA['PT5M']]>
               </bpel:for>
               <bpel:scope>
                  <bpel:extensionActivity ext:id="077">
                     <b4c:fragmentRegion  name="Do error handling for
                        receiving assembled media request"
```

```
                        ext:id="78">
                    </b4c:fragmentRegion>
                </bpel:extensionActivity>
            </bpel:scope>
         </bpel:onAlarm>
      </bpel:pick>
      <bpel:extensionActivity ext:id="079">
         <b4c:fragmentExit name="Exit integration point"
            type="mandatory" ext:id="80">
         </b4c:fragmentExit>
      </bpel:extensionActivity>
   </bpel:sequence>
</b4c:fragmentScope>
</bpel:extensionActivity>
</bpel:process>
```

Listing 17 XML representation of BPEL4CFrags example for implementing compliance requirement time-based plan for video provider VideoSport

```
<bpel:process name="WatchMeProcess"
   targetNamespace="http://example.org/bpel4cfrags/watchme"
   xmlns:tns="http://example.org/bpel4cfrags/watchme"
   xmlns:bpel="http://docs.oasis-open.org/wsbpel/2.0/process/abstract"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:b4c=" http://www.iaas.uni-stuttgart.de/ext/bpel4cfrags"
   xmlns:ext="http://www.iaas.uni-stuttgart.de/ext/id">

   <bpel:documentation xml:lang="EN">
      A simple example of a BPEL4CFrag for implementing the compliance
      requirement composition permission for the concrete video provider
      VideoSport.
   </bpel:documentation>

   <bpel:extensions>
      <bpel:extension namespace="http://www.iaas.uni-stuttgart.de/
         ext/bpel4cfrags"
         mustUnderstand="yes"/>
         <bpel:extension namespace="http://www.iaas.uni-stuttgart.de/
            ext/id" mustUnderstand="no"/>
   </bpel:extensions>

   <bpel:import location="WatchMeProcessArtifacts.wsdl"
```

```
      namespace="http://www.compas-ict.eu/watchme"
      importType="http://schemas.xmlsoap.org/wsdl/"/>
   <bpel:import location="AudioSport_withPartnerLinkType.wsdl"
      namespace="http://www.compas-ict.eu/watchme"
      importType="http://schemas.xmlsoap.org/wsdl/" />
   <bpel:import location="VideoSport_withPartnerLinkType.wsdl"
      namespace="http://www.compas-ict.eu/watchme"
      importType="http://schemas.xmlsoap.org/wsdl/" />
   <bpel:import location="Assembly_withPartnerLinkType.wsdl"
      namespace="http://www.compas-ict.eu/watchme"
      importType="http://schemas.xmlsoap.org/wsdl/" />

   <bpel:extensionActivity>
      <b4c:fragmentScope name="fragmentContainer" ext:id="001">
         <bpel:partnerLinks ext:id="002">
            <bpel:partnerLink
               name="WatchMeProcessPL"
               partnerLinkType="tns:WatchMeProcessPLT"
               myRole="WatchMeProcessProvider"
               ext:id="003"/>
            <bpel:partnerLink
               name="AudioSportPL"
               partnerLinkType="tns:AudioSportPLT"
               partnerRole="AudiSportProvider"
               initializePartnerRole="yes"
               ext:id="004"/>
            <bpel:partnerLink
               name="VideoSportPL"
               partnerLinkType="tns:VideoSportPLT"
               partnerRole="VideoSportProvider"
               initializePartnerRole="yes"
               ext:id="005"/>
            <bpel:partnerLink
               name="AssemblyPL"
               partnerLinkType="tns:AssemblyPLT"
               partnerRole="AssemblyProvider"
               initializePartnerRole="yes"
               ext:id="006"/>
         </bpel:partnerLinks>

         <bpel:variables ext:id="007">
            <bpel:variable name="VideoProvider" type="xsd:string"
```

```
            ext:id="008"/>
         <bpel:variable name="AudioProvider" type="xsd:string"
            ext:id="009"/>
         <bpel:variable name="PayRequest"
            messageType="tns:payRequest" ext:id="010"/>
         <bpel:variable name="PayResponse"
            messageType="tns:payResponse" ext:id="011"/>
         <bpel:variable name="VideoSportURLRequest"
            messageType="tns:requestVideoURLRequest"
            ext:id="012"/>
         <bpel:variable name="VideoSportURLResponse"
            messageType="tns:requestVideoURLResponse"
            ext:id="013"/>
         <bpel:variable name="AudioSportURLRequest"
            messageType="tns:requestAudioURLRequest"
            ext:id="014"/>
         <bpel:variable name="AudioSportURLResponse"
            messageType="tns:requestAudioURLResponse"
            ext:id="015"/>
         <bpel:variable name="AssembledVideoAndAudioURLRequest"
            messageType="tns:requestAssembledVideoAndAudioURL
            Request" ext:id="016"/>
         <bpel:variable name="AssembledVideoAndAudioURLResponse"
            messageType="tns:requestAssembledVideoAndAudioURL
            Response" ext:id="017"/>
         <bpel:variable name="WrapperVariable"
            messageType="tns:WrapperMessage" ext:id="018"/>
      </bpel:variables>

      <bpel:correlationSets ext:id="019">
         <bpel:correlationSet name="CustomerIdentification"
            properties="tns:correlationID" ext:id="020"/>
      </bpel:correlationSets>

      <bpel:sequence name="Composition Permission Compliance
         Fragment" ext:id="021">
         <bpel:extensionActivity ext:id="022">
            <b4c:fragmentEntry name="Entry integration point"
               type="mandatory" ext:id="23">
            </b4c:fragmentEntry>
         </bpel:extensionActivity>
```

```
<bpel:pick name="Receive assembled media url request or
   five minutes timeout occurs" createInstance="no"
   ext:id="024">
   <bpel:onMessage partnerLink="WatchMeProcessPL"
      portType="tns:WatchMeProcess"
      operation="requestAssembledMediaURL"
      variable="AssembledMediaURLInput" ext:id="025">
      <bpel:correlations ext:id="026">
         <bpel:correlation set="CustomerIdentification"
            initiate="no" ext:id="027"/>
      </bpel:correlations>

      <bpel:sequence name="Handling assebled media url
         request" ext:id="028">
         <bpel:extensionActivity ext:id="029">
            <b4c:fragmentRegion name="Handle payment
               check concerning corresponding video
               provider" ext:id="030">
            </b4c:fragmentRegion>
         </bpel:extensionActivity>
         <bpel:if name="Is VideoSport the provider of the
            chosen video?" ext:id="031">
            <bpel:condition ext:id="032">
               <![CDATA[$VideoProvider="VideoSport"]]>
            </bpel:condition>
            <bpel:sequence name="Retrieving video from
               provider VideoSport and audio from
               provider AudioSport in parallel"
               ext:id="033">
               <bpel:flow name="Parallel retrieving of
                  video url from VideoSport and audio url
                  from AudioSport" ext:id="034">
                  <bpel:sequence name="Retrieving video
                     url from provider VideoSport"
                     ext:id="035">
                     <bpel:assign validate="no"
                        name="Assign video id to url
                        request" ext:id="036">
                        <bpel:copy ext:id="037">
                           <bpel:from ext:id="038">
                              <bpel:literal
                              xml:space="preserve">
```

```
                        <impl:requestVideoURL
                           xmlns:impl="http://
                           www.compas
                           ict.eu/watchme"
                           xmlns:xsi="http://
                           www.w3.org/2001/
                           XMLSchema-instance">
                           <impl:videoID>
                           </impl:videoID>
                        </impl:requestVideoURL>
                     </bpel:literal>
                  </bpel:from>
                  <bpel:to part="parameters"
                     variable="VideoSportURL
                     Request" ext:id="039">
                  </bpel:to>
               </bpel:copy>
               <bpel:copy ext:id="040">
                  <bpel:from part="payload"
                     variable="AssembledMediaURL
                        Input" ext:id="041">
                     <bpel:query
                        queryLanguage="urn:
                        oasis:names:tc:wsbpel:
                        2.0:sublang:xpath1.0">
                        <![CDATA[tns:videoID]]>
                     </bpel:query>
                  </bpel:from>
                  <bpel:to part="parameters"
                     variable="VideoSportURL
                        Request" ext:id="042">
                     <bpel:query queryLanguage=
                        "urn:oasis:names:tc:
                        wsbpel:2.0:sublang:
                        xpath1.0">
                        <![CDATA[tns:videoID]]>
                     </bpel:query>
                  </bpel:to>
               </bpel:copy>
            </bpel:assign>
            <bpel:invoke
               partnerLink="VideoSportPL"
```

```
                operation="requestVideoURL"
                portType="tns:VideoSport"
                inputVariable="VideoSportURL
                   Request"
                outputVariable="VideoSportURL
                Response" name="Retrieve video
                url from provider VideoSport"
                ext:id="043" />
          </bpel:sequence>
          <bpel:sequence name="Retrieving audio
             url from provider AudioSport"
             ext:id="044">
             <bpel:assign validate="no"
                name="Assign audio id to url
                request" ext:id="045">
                <bpel:copy ext:id="046">
                   <bpel:from ext:id="047">
                      <bpel:literal
                         xml:space="preserve">
                         <impl:requestAudioURL
                            xmlns:impl="http://
                            www.compas-ict.eu/
                            watchme"
                            xmlns:xsi="http://
                            www.w3.org/2001/
                            XMLSchema-instance">
                            <impl:audioID>
                            </impl:audioID>
                         </impl:requestAudioURL>
                      </bpel:literal>
                   </bpel:from>
                   <bpel:to part="parameters"
                      variable="AudioSportURL
                         Request" ext:id="048">
                   </bpel:to>
                </bpel:copy>
                <bpel:copy ext:id="049">
                   <bpel:from part="payload"
                      variable="AssembledMedia
                         URLInput" ext:id="050">
                      <bpel:query
                         queryLanguage="urn:
                         oasis:names:tc:wsbpel:
```

```
                                    2.0:sublang:xpath1.0">
                                    <![CDATA[tns:audioID]]>
                                 </bpel:query>
                              </bpel:from>
                              <bpel:to part="parameters"
                                 variable="AudioSportURL
                                 Request" ext:id="051">
                                 <bpel:query
                                    queryLanguage="urn:
                                    oasis:names:tc:wsbpel:
                                    2.0:sublang:xpath1.0">
                                    <![CDATA[tns:audioID]]>
                                 </bpel:query>
                              </bpel:to>
                           </bpel:copy>
                        </bpel:assign>
                        <bpel:invoke
                           partnerLink="AudioSportPL"
                           operation="requestAudioURL"
                           portType="tns:AudioSport"
                           inputVariable="AudioSportURL
                              Request"
                           outputVariable="AudioSportURL
                              Response"
                           name="Retrieve audio url from
                              provider AudioSport"
                           ext:id="052"/>
                     </bpel:sequence>
                  </bpel:flow>
                  <bpel:assign validate="no" name="Assign
                     audio url and video url to request"
                     ext:id="053">
                     <bpel:copy ext:id="054">
                        <bpel:from ext:id="055">
                           <bpel:literal
                              xml:space="preserve">
                              <impl:requestAssembledVideo
                                 AndAudioURL
                                 xmlns:impl="http://www.
                                    compas-ict.eu/watchme"
                                 xmlns:xsi="http://www.
                                    w3.org/2001/XMLSchema
```

```
                        instance">
                    <impl:aVideoURL>
                    </impl:aVideoURL>
                    <impl:anAudioURL>
                    </impl:anAudioURL>
                </impl:requestAssembledVideo
                    AndAudioURL>
            </bpel:literal>
        </bpel:from>
        <bpel:to part="parameters"
            variable="AssembledVideoAnd
            AudioURLRequest" ext:id="056">
        </bpel:to>
    </bpel:copy>
    <bpel:copy ext:id="057">
        <bpel:from part="parameters"
            variable="VideoSportURLResponse"
            ext:id="058">
            <bpel:query queryLanguage="urn:
                oasis:names:tc:wsbpel:2.0:
                sublang:xpath1.0">
                <![CDATA[tns:requestVideo
                    URLReturn]]>
            </bpel:query>
        </bpel:from>
        <bpel:to part="parameters"
            variable="AssembledVideoAndAudio
                URLRequest" ext:id="059">
            <bpel:query queryLanguage="urn:
                oasis:names:tc:wsbpel:2.0:
                sublang:xpath1.0">
                <![CDATA[tns:aVideoURL]]>
            </bpel:query>
        </bpel:to>
    </bpel:copy>
    <bpel:copy ext:id="060">
        <bpel:from part="parameters"
            variable="AudioSportURLResponse"
            ext:id="061">
            <bpel:query queryLanguage="urn:
                oasis:names:tc:wsbpel:2.0:
                sublang:xpath1.0">
```

```
                        <![CDATA[tns:requestAudioURL
                           Return]]>
                     </bpel:query>
                  </bpel:from>
                  <bpel:to part="parameters"
                     variable="AssembledVideoAndAudio
                        URLRequest" ext:id="062">
                     <bpel:query queryLanguage="urn:
                        oasis:names:tc:wsbpel:2.0:
                        sublang:xpath1.0">
                        <![CDATA[tns:anAudioURL]]>
                     </bpel:query>
                  </bpel:to>
               </bpel:copy>
            </bpel:assign>
         </bpel:sequence>
         <bpel:elseif ext:id="063">
            <bpel:condition ext:id="064">
               <![CDATA[$VideoProvider=
                  "FootballGames"]]>
            </bpel:condition>
            <bpel:extensionActivity ext:id="065">
               <b4c:fragmentRegion name="Handling of
                  video retrieval from provider
                  FootballGames and audio from
                  providers AudioSport or
                  SportingAudio in parallel"
                  ext:id="066">
               </b4c:fragmentRegion>
            </bpel:extensionActivity>
         </bpel:elseif>
      </bpel:if>
      <bpel:invoke partnerLink="AssemblyPL"
         operation="requestAssembledVideoAndAudioURL"
         portType="tns:Assembly"
         inputVariable="AssembledVideoAndAudiURLRequest"
         outputVariable="AssembledVideoAndAudio
         URLResponse" name="Retrieve assembled video
         and audio url from assembly service"
         ext:id="067"/>
      <bpel:assign validate="no" name="Assign
         assembled video and audio url to reply
```

```
                    message" ext:id="068">
                    <bpel:copy ext:id="069">
                       <bpel:from ext:id="070">
                          <bpel:literal xml:space="preserve">
                             <tns:WatchMeAssembledMediaURL
                                Response
                                xmlns:tns="http://www.compas
                                   ict.eu/watchme"
                                xmlns:xsi="http://www.
                                   w3.org/2001/XMLSchema
                                   instance">
                                <tns:result></tns:result>
                             </tns:WatchMeAssembledMediaURL
                             Response>
                          </bpel:literal>
                       </bpel:from>
                       <bpel:to variable="AssembledMediaURL
                          Output"
                          part="payload" ext:id="071"/>
                    </bpel:copy>
                    <bpel:copy ext:id="072">
                       <bpel:from part="parameters"
                          variable="AssembledVideoAndAudioURL
                             Response" ext:id="073">
                          <bpel:query queryLanguage="urn:oasis:
                             names:tc:wsbpel:2.0:sublang:
                             xpath1.0">
                             <![CDATA[tns:requestAssembled
                                VideoAndAudioURLReturn]]>
                          </bpel:query>
                       </bpel:from>
                       <bpel:to part="payload"
                          variable="AssembledMediaURLOutput"
                          ext:id="074">
                          <bpel:query queryLanguage="urn:oasis:
                             names:tc:wsbpel:2.0:sublang:
                             xpath1.0">
                             <![CDATA[tns:result]]>
                          </bpel:query>
                       </bpel:to>
                    </bpel:copy>
                 </bpel:assign>
                 <bpel:reply name="Send assembled video and audio
```

```
                        url to customer"
                        partnerLink="WatchMeProcessPL"
                        operation="requestAssembledMediaURL"
                        portType="tns:WatchMeProcess"
                        variable="AssembledMediaURLOutput"
                        ext:id="075">
                     </bpel:reply>
                  </bpel:sequence>
               </bpel:onMessage>
               <bpel:onAlarm ext:id="076">
                  <bpel:for ext:id="077">
                     <![CDATA['PT5M']]>
                  </bpel:for>
                  <bpel:scope ext:id="078">
                    <bpel:extensionActivity ext:id="079">
                        <b4c:fragmentRegion name="Do error handling
                           for receiving assembled media url request"
                           ext:id="080">
                        </b4c:fragmentRegion>
                     </bpel:extensionActivity
                  </bpel:scope>
               </bpel:onAlarm>
            </bpel:pick>
            <bpel:extensionActivity ext:id="081">
               <b4c:fragmentExit name="Exit integration point"
                  type="mandatory" ext:id="82">
               </b4c:fragmentExit>
            </bpel:extensionActivity>
         </bpel:sequence>
      </b4c:fragmentScope>
   </bpel:extensionActivity>
</bpel:process>
```

Listing 18 XML representation of BPEL4CFrags example for implementing compliance requirement composition permission for video provider Video-Sport