

Ontology-based Translation of Business Process Models

Barry Norton Liliana Cabral
KMi, Open University, UK
{b.j.norton, l.s.cabral}@open.ac.uk

Jörg Nitzsche
IAAS, University of Stuttgart, Germany
joerg.nitzsche@iaas.uni-stuttgart.de

Abstract

Semantic Business Process Management is a recent and promising research area devoted to extending the results from Semantic Web Services — i.e., the application of ontology-based modelling and reasoning to Web Services — to Business Process Management, which is these days often realised using Service-Oriented Architectures. One important task for Semantic Business Process Management is the addition of corresponding semantic annotations to the business-oriented view on processes. Since, however, the Semantic Web approach, realised using ontologies, is not simply to represent conceptualisations in the style of a data model, but to embody relationships and inferences over such conceptualisations, an equally important task to realise Semantic Business Process Management is hence to relate business-oriented and IT/service-oriented views of processes ontologically.

Previous work has shown how an ontology language rich enough to encode rules can provide a relationship between an abstract ontology for business processes and a low-level ontological representation of process behaviour providing behavioural semantics for these business processes. In this paper we show how business-oriented and execution-oriented ontological representations of business processes can similarly be related.

1. Introduction

Business Process Management (BPM) is a well-established discipline whereby company processes are modelled, monitored, managed and adapted according to business experts' viewpoint, well-separated from the IT concerns associated with their realisation. At the same time the approach of Service-Oriented Architecture (SOA) has made strides towards supporting the requirements of agile cross-organisational business processes at this lower level. Maintaining the separation of viewpoints, while seamlessly preserving and exploiting links between them, clearly therefore becomes an important goal.

The SUPER project¹ was one of the first to propose that exactly this separation of viewpoints, alongside the provision of automation via machine reasoning over implicit connections, is foremost among the strengths of the Semantic Web's ontology-based approach, already exploited at the SOA level in Semantic Web Services (SWS) research. The Web Services Modeling Ontology (WSMO) [8] is designed to promote the *ontological separation of concerns* between a goal model and a service model; the former representing a service consumer's perspective of their own concerns for service consumption, the latter service providers' various descriptions of offers of service provision. The matches between these are formed not by forcing one party to adopt the viewpoint of the other but by ontology-based mediation of these separate descriptions. In the same manner it is the SUPER approach, known as *Semantic BPM*, to describe a business expert's overview of entire business processes and implicitly relate these to the service-oriented IT viewpoint, already described by SWS models such as WSMO, while enforcing the principal of separation of concerns.

In order to represent the business viewpoint, SUPER has developed the Business Process Modelling Ontology (BPMP), described in Section 2.1. This is now a proposal for standardisation by the Semantic Technologies Institute International². For the execution of business processes thus defined over semantic web services, SUPER has extended the Business Process Execution Language (BPEL) for Web Services (according to the latest version WS-BPEL 2.0 [5]) along with one of the authors of the original BPEL4WS specification to form BPEL4SWS, described in Section 2.2. This has an ontological representation as Semantic BPEL (sBPEL). For the remainder of the paper Section 3 describes how ontology-based rules, contained in novel ontologies we call BPMP2sBPEL and BPEL4BPMP, allow automated translation between related representations of business processes according to these different viewpoints and their common attributes. Section 4 then offers conclusions and discusses future work, including standardisation.

¹<http://www.ip-super.org/>

²<http://sti2.at>

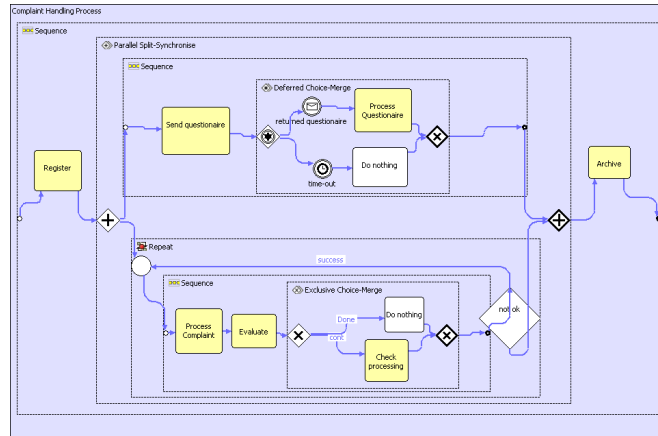


Figure 1. Complaint Handling Process

2 Background

In this section we describe the two models by which business processes are represented in the SUPER project, BPMO and Semantic BPEL, focussed on business experts' modelling viewpoint and IT experts' execution viewpoint respectively.

2.1 BPMO

Currently, business analysts use process modelling notations such as BPMN [6] and EPCs [9] to draw and analyse business process diagrams as part of tool suites for BPM. These notations are useful at the business level but they only provide graphical elements with limited textual information and little by the way of formal semantics. In addition, such tool suites can inform and share very little about the process organisational context and related services.

The Business Process Modelling Ontology (BPMO) enables the semantic annotation of high-level business process models. BPMO includes concepts to describe process behaviour (workflow), activities and related organisational data.

The Business Process Modelling Ontology (BPMO) provides support for various BPM activities, regardless of specific notations, in a way which crosses domains and organisational boundaries. BPMO harnesses a number of Semantic Web technologies so that business process workflows can: a) be exposed through semantic descriptions; b) refer to semantically annotated data and services through URIs; c) incorporate heterogeneous data through semantic mappings; and d) be queried using a reasoner or inference engine. We argue therefore, that BPMO enables the seamless reuse, translation, interoperation, mediation and querying of business processes.

The BPMO model captures control-flow features of notations such as BPMN, process interaction features from BPEL, service description and invocation features from SWS, in particular WSMO, plus domain-independent organisational aspects. Whereas the existing notations and languages, in particular EPCs, BPMN (an OMG standard) and BPEL (an OASIS standard), all suffer from a lack of (official) formal semantics, arbitrary differences and ill-defined (ambiguous and under-defined informal) semantics, BPMO concentrates on the shared and compatible features with well-defined semantics.

In particular in dealing with graph-oriented features, BPMO builds on the formalization of a BPD (Business Process Diagram) as presented in [7]; BPMO is oriented towards the production of well-formed BPDs where graphs decompose unambiguously into subgraphs that start and end with compatible constructs, avoiding undefined semantics where this is not the case.

Within the BPMO approach, a business analyst can draw a business process diagram with a tool that automatically generates BPMO instances³, as shown in our example in Figure 1, or he can use translators that will transform other notations from and to BPMO. Currently SUPER has defined translators from BPMN, EPC, XPD and BPEL. BPMO can thus be viewed as a bridging ontology, allowing for the annotation and automated translation between an open-ended set of existing notations and languages.

In this paper we concentrate on translation with respect to BPEL-based models since these form both a source of translation into BPMO, for legacy IT processes which we want to reverse-engineer into a business-oriented representation, and the standard destination of translation from BPMO, where BPEL4SWS becomes a basis for execution of Semantic Business Processes.

³<http://www.wsmstudio.org>

2.2 Semantic BPEL

BPEL is the de facto standard for specifying business processes in a Web Service world. It is XML based and enables both the composition of Web Services and rendering the composition itself in Web Service form. BPEL is a control-flow based approach where the control flow can be specified either in a block-based manner by nesting structured activities like `<sequence>` (for sequential control flow), `<flow>` (for parallel control flow) and `<if>` (for conditional branches in the control flow) activities, or graph-based by defining `<links>` (i.e., directed edges) between activities in a `<flow>` activity; both styles can be used intermixed. The interaction with partner services is specified using WSDL, i.e., interaction activities (`<receive>`, `<reply>`, `<invoke>`, `<pick>`) are defined in terms of WSDL port types and operations.

BPEL4SWS [3] is an extension of BPEL 2.0. It provides for a communication model that is based on plain message exchange [4], independently of WSDL. It introduces an `<interactionActivity>` that can be configured such that it behaves like any of the existing interaction activities using BPEL's extensibility. Multiple `<interactionActivity>`s are grouped via a `<conversation>` element to enable modelling long running conversational interaction amongst partners. Activity implementation in BPEL4SWS can be either described using WSDL or using any other IDL including Semantic Web Service frameworks such as OWL-S and WSMO. A conversation via which the process offers functionality to a client can be described using an OWL-S service or a WSMO Web Service description. A conversation that consumes functionality another service provides can be described either by an OWL-S template or by a WSMO goal that can be submitted during runtime to a semantic execution environment (SEE). In WSMO the conversation is reflected in the choreography interface of a Web service or goal; in OWL-S services have an implicit atomic conversation with their client. SWS frameworks use ontologies as data model whereas BPEL uses XML. BPEL4SWS uses SAWSDL for annotating data types, i.e., for giving a meaning to used data by referring to ontological concepts. It defines lifting and lowering schemas which enables seamless mapping of XML and ontological representation of data.

Data flow in BPEL(4SWS) is implicit. It is realized via (globally) shared data. Variables can be defined either in the process itself or within a restricted scope. Different activities can use these variables as input or output containers and read or write data respectively. Data can be copied from one variable to another using the assign activity, based either on syntactic manipulation, as in WS-BPEL, or semantic mediation, such as in the kind of WSML-Flight rules described in this paper [2].

3 Translation

Inter-translation between business-oriented representations of business processes in BPMO and execution-oriented BPEL-like representations are effected by two ontologies named BPMO2sBPEL and BPEL2BPMO. Each import the representations of these models in WSML, the Web Services Modeling Language [1], and contain F-logic-like rules within WSML-Flight axioms that infer instances in the target ontology from those in the source.

For example, the BPEL20 ontology, which is imported for extension by sBPEL, contains a process concept roughly as follows⁴:

```
concept bpeL#Process subConceptOf upo#BusinessProcessModel
nonFunctionalProperties
  xmlns hasValue "http://docs.oasis-open.org/wsBPEL/2.0/
  Process/executable"
  dc:description hasValue "Concept of being a <process> -
  Element of an executable BPEL Process"
endNonFunctionalProperties
hasPartnerLink ofType bpeL#PartnerLink
hasMessageExchange ofType MessageExchange
hasVariable ofType Variable
hasActivity ofType (1) Activity
```

Listing 1. Basic Process Concept in BPEL Ontology

Conversely in BPMO the process concept is defined as follows:

```
concept bpmo#Process subConceptOf { bpmo#BusinessActivity, upo#
  BusinessProcessModel }
nonFunctionalProperties
  dc:description hasValue "Business process as an identifiable
  set of tasks, decision points and their associated workflow."
endNonFunctionalProperties
hasWorkflow ofType (0 1) Workflow

concept bpmo#BusinessActivity subConceptOf upo#BusinessActivity
nonFunctionalProperties
  dc:description hasValue "A Process or Task, typically the result
  of a business process design or business process
  engineering activity."
endNonFunctionalProperties
hasName ofType (0 1) .string
hasDescription ofType (0 1) .string
hasNonFunctionalProperties ofType (0 1)
  BusinessActivityNonFunctionalProperties
hasBusinessDomain ofType upo#BusinessDomain
hasBusinessFunction ofType upo#BusinessFunction
hasBusinessStrategy ofType upo#BusinessStrategy
hasBusinessPolicy ofType upo#BusinessPolicy
hasBusinessProcessMetrics ofType upo#BusinessProcessMetrics
hasBusinessProcessGoal ofType upo#BusinessProcessGoal
hasBusinessResource ofType upo#Resource
```

Listing 2. Basic Process Concept in BPMO

⁴Throughout we omit details for brevity, but the complete ontologies are available online at the SUPER web site.

While we do not concentrate in this paper on UPO⁵, the SUPER upper ontology for business processes, it is clear that although these concepts share one superconcept that defines their attributes as processes, the BPMO process allows annotation with much more information about the organisational context.

As a container for a workflow description, though, the basis of the rule for translating an instance from the BPEL ontology into BPMO is as follows:

```

relation bpel2bpmo#processTranslation( impliesType bpel#Process,
impliesType bpmo#Process)

axiom aProcessWithACommonActivityIsTranslated
  nfp
    dc#description hasValue "Implies BPMO representation of BPEL
    processes"
  endnfp
  definedBy
    ?p[bpel#hasActivity hasValue ?a] memberOf bpel#Process and
    activityCommonTranslation(?a, ?e) implies
    sem(?p)[bpmo#hasWorkflow hasValue wf(?p)] memberOf
    bpmo#Process and
    wf(?p)[bpmo#hasHomeProcess hasValue sem(?p),
    bpmo#hasFirstWorkflowElement hasValue ?e]
    memberOf bpmo#Workflow and
    processTranslation(?p, sem(?p)).
  
```

Listing 3. Relation and basic rule in BPEL2BPMO

It can be easily seen that this rule essentially infers, via the ‘implies’ keyword, a pairing of any BPEL instance that matches the antecedant with an inferred BPMO process and workflow in the ‘processTranslation’ relation. The names for the inferred instances are based on the name of the BPEL process instance (due to the use of the variable ‘?p’) using the functions ‘sem()’ and ‘wf()’. In WSML if these functions are undefined this compound function symbol itself becomes the name of the instance.

It should also be clear that the antecedant in this rules include a dependency on a further relation over activities ‘activityCommonTranslation’ and in this way an induction is formed over the structure of the process. In fact there are three such relations in BPEL2BPMO; in addition ‘activity-BlockTranslation’ and ‘activityGraphTranslation’. This is due to BPMO’s support for both block and graph-oriented modelling; block-oriented modelling is preferred due to the completeness of forward translation to BPEL4SWS, however since BPEL itself supports both forms of modelling some processes can only be translated into graph form. Due to space constraints we ignore these cases in this paper and concentrate of the workflow elements shown in Listing 4.

⁵For clarity we omit ontology and namespace definitions but assume namespace shortcuts that match the ontology names and include these in the names when defining concepts and relations.

```

concept Workflow subConceptOf upo#ProcessOrchestrationSpecification
  hasHomeProcess ofType (0 1) Process
  hasFirstWorkflowElement ofType (1) WorkflowElement

concept WorkflowElement subConceptOf upo#
  ProcessOrchestrationElement
  hasHomeProcess ofType (0 1) Process

concept GraphPattern subConceptOf WorkflowElement

concept BlockPattern subConceptOf WorkflowElement

concept Sequence subConceptOf BlockPattern
  hasOrderedElement ofType (1 *) OrderedElement
  hasSize ofType (0 1) .integer

concept ParallelSplitSynchronise subConceptOf BlockPattern
  hasBranch ofType (2 *) UnconditionalBranch

concept DeferredChoiceMerge subConceptOf BlockPattern
  hasEventBranch ofType (2 *) EventBranch

concept ExclusiveChoiceMerge subConceptOf BlockPattern
  hasSize ofType (0 1) .integer
  hasConditionalBranch ofType (2 *) ConditionalBranch
  hasDefaultBranch ofType (0 1) UnconditionalBranch

concept Loop subConceptOf BlockPattern
  hasCondition ofType (0 1) Condition
  executes ofType (0 1) WorkflowElement

concept Repeat subConceptOf Loop
  
```

Listing 4. Workflow elements in BPMO

In terms of these concepts we can see how the example in Figure 1 is modelled in BPMO; a topmost sequence decomposes into a message receipt followed by a parallel set of activities followed by sending a message. The two parallel processes demonstrate the different types of choice — an exclusive (internal) choice is decided on data, whereas a deferred choice is decided on the receipt of messages, possibly including a time-out in case messages are not received — and the latter looping.

In BPEL correspondingly sequences, loops and choices are represented by block structured activities; sequences and choices (so long as these are based on mutually exclusive conditions) may also be represented in graph form, but loops are only represented by blocks. Internal choices are represented in if-then-else form with an implicit ordering (since there is no exclusivity constraint), which is optional in BPMO; external choices are called ‘pick’s. The syntactic version of message receipt in BPEL is translated in the common activities relation as shown below:

```

axiom aSyntacticReceive
  nfp
    dc#description hasValue "Implies BPMO for BPEL receives"
  endnfp
  definedBy
    ?r memberOf bpel#Receive implies
    sem(?r) memberOf bpmo#Receive and
    activityCommonTranslation(?r, sem(?r)).
  
```

Listing 5. Translation of BPEL Receive

Conversely when the BPMO version of Receive is translated into the Semantic BPEL ontology the semantic information about the ontological representation (concept) of the message received is present in the model and represented in the execution. As mentioned before, in BPEL4SWS (the XML representation of the execution language) this information will be represented in SAWSDL-style model references.

```

axiom aReceive
  nfp
    dc:description hasValue "Implies sBPEL representation of BPMO receives"
  endnfp
  definedBy
    ?r[bpmo#hasInputDescription hasValue ?cap] memberOf bpmo#Receive and
    ?cap[bpmo#hasSemanticDescription hasValue ?concept] memberOf bpmo#SemanticCapability
  implies
    sem(?r)[bpel#hasActivity hasValue semext(?r)] memberOf bpel#ExtensionActivity and
    semext(?r)[sbpel#hasVariable hasValue semvar(?r)] memberOf sbpel#Receive and
    semvar(?r)[sbpel#hasSemanticType hasValue ?concept] memberOf elementCommonTranslation(?r, sem(?r)).

```

Listing 6. Translation of BPMO Receive

In general translation into Semantic BPEL requires some loosening of constraints in the version of the ontology published by the SUPER project. For example we note that among the attributes we elided earlier of the BPEL process concept, of which the sBPEL process is a subconcept, are the following:

```

hasName ofType (1) _string
hasTargetNamespace ofType (1) _string /* _iri */

```

Listing 7. Further Attributes of BPEL Process

Variables, as created as a placeholder for the concept in the rule above, are also required to have a syntactic name, since the published ontology matches the BPEL schema one-to-one in this regard. The translator offered by the project, which was encoded as ATL rules⁶ within a Java program, is able to form such names via the generation of random numbers and string concatenation of names, these and literals. Although this is beyond the scope of reasoning in WSML-Flight, our assertion is that the need for syntactic names is a redundancy in an ontology-based model where unique identifiers implicitly exist for every instance. As a result we can weaken the constraints of the Semantic BPEL ontology and use the WSML identifiers — themselves, by nature, URIs — in the XML representation on lowering.

⁶<http://www.eclipse.org/M2M/ATL>

As a final example of the power of WSML-Flight rules to effect this translation we look again at the translation from BPEL to BPMO, where the naming issue is not present even in the unaltered ontologies, and consider the translation of sequences. This requires an auxiliary relation, called ‘list-Translation’ over which to build an induction as follows:

```

relation listTranslation( impliesType bpel#OrderedActivity, impliesType
  bpmo#OrderedElement, impliesType _integer)

axiom aSequence
  nfp
    dc:description hasValue "Implies BPMO representation of BPEL sequences"
  endnfp
  definedBy
    ?s[bpel#hasOrderedActivity hasValue ?oa] memberOf bpel#Sequence and
    listTranslation(?oa, ?oe, 1) implies
      sem(?s)[bpmo#hasOrderedElement hasValue ?oe] memberOf bpmo#Sequence and
      activityCommonTranslation(?s, sem(?s)).

axiom anOrderedActivity1
  nfp
    dc:description hasValue "Implies BPMO representation of initial BPEL ordered activities"
  endnfp
  definedBy
    ?oa1[bpel#hasActivity hasValue ?a] memberOf bpel#OrderedActivity and
    ?s memberOf bpel#Sequence and
    ?s[bpel#hasOrderedActivity hasValue ?oa1] and
    activityCommonTranslation(?a, sem(?a))
  implies
    //note: this does not define the nextElement — see anOrderedActivity2
    sem(?oa1)[bpmo#hasElement hasValue sem(?a), bpmo#hasOrder hasValue 1] memberOf bpmo#OrderedElement and
    listTranslation(?oa1, sem(?oa1), 1).

axiom anOrderedActivity2
  nfp
    dc:description hasValue "Implies BPMO representation of non-initial BPEL ordered activities"
  endnfp
  definedBy
    ?oanext[bpel#hasActivity hasValue ?anext] memberOf bpel#OrderedActivity and
    ?oaprevious memberOf bpel#OrderedActivity and
    ?oaprevious[bpel#hasOrderedActivity hasValue ?oanext] and
    listTranslation(?oaprevious, ?oeprevious, (?nnext - 1)) and
    activityCommonTranslation(?anext, ?enext)
  implies
    sem(?oanext)[bpmo#hasElement hasValue ?enext, bpmo#hasOrder hasValue ?nnext] memberOf bpmo#OrderedElement and
    ?oeprevious[bpmo#hasNextElement hasValue sem(?oanext)] and
    listTranslation(?oanext, sem(?oanext), ?nnext).

```

Listing 8. Translation of BPEL Sequences

We note that both the BPMO and BPEL ontologies adopt a linked-list like representation of members of a sequence, via the concepts ‘OrderedElement’ and ‘OrderedActivity’ respectively, and that the inductive part of the construction is in the second rule and in particular the population of the ‘hasNextElement’ attribute.

4 Conclusions and Further Work

In this paper we have described the motivations of the new paradigm of Semantic Business Process Management and the approach taken to realise this by the SUPER project. In particular we motivated the separate conceptualisation of two distinct perspectives on a business process: the business analyst's view and the IT expert's view.

We have described the design of the SUPER project's Business Process Modelling Ontology (BPMO) as a basis to represent the business analyst's viewpoint of business processes, wherein annotations are made to a notation- and technology-agnostic model of process, including semantic representations not only of data but also organisational and other business aspects.

Similarly we described the extension of the industry *de facto* standard for executable business processes, BPEL, for the execution of such semantic business processes using semantic web service technology. The resulting BPEL4SWS schema then has an ontological representation as Semantic BPEL, of which we have discussed the nature and current limitations.

Finally we discussed the use of ontology-based rules, which the reasoner can effect in a decidable fashion consistent with the Semantic Web approach, which link the common elements of these two representations of a process. These can be used to derive an execution for a process modelled in BPMO or to extract a BPMO model for semantic annotation from a legacy process.

Both BPMO and Semantic BPEL are now candidates for an open standardisation process via STI's Conceptual Models for Services (CMS) Working Group⁷, where this ontology-based link between models will be supported, and to which interested parties are invited to participate.

Future work involves the consideration of translation to BPMO from other process notations. Currently SUPER considers EPCs, BPMN and XPDL and carries out the translation extra-ontologically using Java, although the reasoner is applied by these translators.

The advantage of implicit translation via ontology-based rules is that an evolving relationship is drawn between two models enabling reasoning that spans both perspectives. This is broken by an off-line extra-ontological translation, even if the connection is explicitly represented (for instance the current sBPEL ontology has 'derivedFrom' attributes ranging over BPMO concepts), if the resulting model is then changed.

The real challenge of addressing wholly graph-based representations of processes lies in conversion of graph-oriented to block-oriented features, an acknowledged open problem in BPM which, to the authors' knowledge, has not previously been addressed with ontology-based technology.

⁷<http://cms-wg.sti2.org>

References

- [1] J. de Bruijn, H. Lausen, R. Krummenacher, A. Polleres, L. Predoiu, M. Kifer, and D. Fensel. The web service modeling language WSML v0.3. <http://www.wsmo.org/TR/d16/d16.1/v0.3/>, October 2005.
- [2] J. Nitzsche and B. Norton. Ontology-based data mediation in BPEL (for semantic web services). In *Advances in Semantics for Web Services (semantics4ws'08)*, volume to appear of LNCS, 2008.
- [3] J. Nitzsche, T. van Lessen, D. Karastoyanova, and F. Leymann. BPEL for Semantic Web Services. In *Proceedings of the 3rd International Workshop on Agents and Web Services in Distributed Environments (AWeSOME'07)*, Nov. 2007.
- [4] J. Nitzsche, T. van Lessen, D. Karastoyanova, and F. Leymann. BPEL^{light}. In *5th International Conference on Business Process Management (BPM)*, Sept. 2007. Brisbane, Australia.
- [5] OASIS Web Services Business Process Execution Language (WSBPEL) TC. Web services business process execution language version 2.0. Technical report, OASIS, 2007. <http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf>.
- [6] Object Management Group. Business process modelling notation (BPMN) specification. Technical report, Object Management Group, 2006. <http://www.omg.org/docs/dtc/06-02-01.pdf>.
- [7] C. Ouyang, W. Van der Aalst, M. Dumas, and A. Hofstede. From business process models to process-oriented software systems: The BPMN to BPEL way. Technical report, Queensland University of Technology, 2006. <http://eprints.qut.edu.au/archive/00005266/01/5266.pdf>.
- [8] D. Roman, U. Keller, H. Lausen, J. de Bruijn, R. Lara, M. Stollberg, A. Polleres, C. Feier, C. Bussler, and D. Fensel. Web service modeling ontology. *Applied Ontology*, 1(1), 2005.
- [9] W. van der Aalst, J. Desel, and E. Kindler. On the semantics of EPCs. In *Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (EPK 2002)*, pages 71–79, 2002.

Acknowledgements

The authors wish to thank the whole SUPER consortium for their part in the background work described.

The SUPER project is funded by the European Union under the 6th Framework Programme, within Information Society Technologies (IST) priority