A Domain-Specific Model for Data Quality Constraints in Service Process Adaptations

Claus Pahl[1], Neel Mani[1], and Ming-Xue Wang[2]

[1] CNGL, School of Computing, Dublin City University
Dublin 9, Ireland
[2] Network Management Lab, Ericsson Ireland
Ericsson Software Campus, Athlone, Ireland

Abstract. Service processes are often enacted across different boundaries such as organisations, countries or even languages. Specifically, looking at the quality and governance of data or content processed by services in this context is important to control different constraints in this cross-boundary processing. In order to provide a context-aware solution that takes into account data and data processing requirements, a rule-based constraints specification and adapation of processes shall be proposed. A domain ontology shall capture the key data/content data types, activities and constraints, which forms the basis of a rule-based policy monitoring solution. A provenance model is at the core of this ontology solution. The key contribution is a domain-specific model and specification template for constraint policy definition, which can be applied to adapt service processes to domain-specific needs.

Keywords: Service Process; Process Adaptation; Content Services; Constraint Monitoring; Quality and Governance; Domain-Specific Model; Provenance.

1 Introduction

Digital content and data is increasingly processing in distributed settings by different agents - human and/or software. As a consequence, maintaining quality across a boundary-crossing service process is a challenge. The focus of this paper is content and data quality in domain-adapted content processes. While work on quality in service processes has been covered widely, our focus is on domain-specific processes and here specifically those centering on content and data processing. We take on board approaches for constraints specification through rule and policy languages

We aim to, firstly, enable domain-specific service processes for content manipulation and change based on a formalised content model, which requires a layered content model. This layered model consists of the bottom layer with core content (in a formal representation), the states and stages of processing on top of that, and a provenance layer linking content and processing to their origins and dates as the third layer. The provenance model [18] will turn out the solution to the need to link content/data into the process. An activities and operations framework that defines the processing and manipulation activities on content in the context of provenance data. The W3C provenance model [18] plays again a pivotal role here for logging process activities, but also as a metadata framework for constraints and rules.

The second aim is to translate this into a dynamic environment. We aim to define a content and data-centric quality assurance and adaptation framework that allows quality requirements to be defined as constraints to be monitored and managed dynamically. This results in the definition of an inclusive framework for the definition, adaptation, monitoring and handling of quality concerns as dynamic constraints. Particular problems are, firstly, the domain-specific categorisation of constraints into policies and, secondly, a rule-based policy definition and process adaptation based on constraints. While constraints monitoring in service processes has been widely covered [3, 13, 20, 24], our solution provides novel contributions in the form of an ontology-driven policy constraints configuration framework.

Our contribution is a domain-specific model for content modelling, covering content, operators and constraints. Our exploration of quality management for content processes, i.e., to define, monitor and analyse, focuses on model aspects here, with the aim of addressing integration and interoperability problems at description level. In a wider sense, this is a governance concern. Our solution specifically extends process adaptation and customisation techniques [1, 7], e.g., generic policy adaptation for service processes [21, 20], by a domain-specific configuration solution.

We first provide some background on text content processing in service processes in Section 2 and outline challenges and analyse a use case in detail in order to elicit specific requirements. Section 3 defines the domain-specific model for content quality constraints. The rule language we used for quality constraints is then introduced and explained in terms of its utilisation here in Section 4. We describe our implementation in Section 5 where we show how this domain constraints definition approach can be implemented using an existing, generic service policy customisation solution. We end with a discussion of related work in Section 6 and some conclusions in Section 7.

2 Scenario - Service-based Content Processing

2.1 Scenario Introduction

Service-base content processing is a distributed problem. Content is created, searched, manipulated (translated, localised, adapted and personalised) and integrated across different processing agents, exposed as services. We refer to this as intelligent content (IC), if the quality is automatically maintained. A process model for this content path is modelled in Fig. 1. This iterative process consists of a number of content processing activities, such as creation, search, translation or adaptation. This process is specific to text-based content and data as an

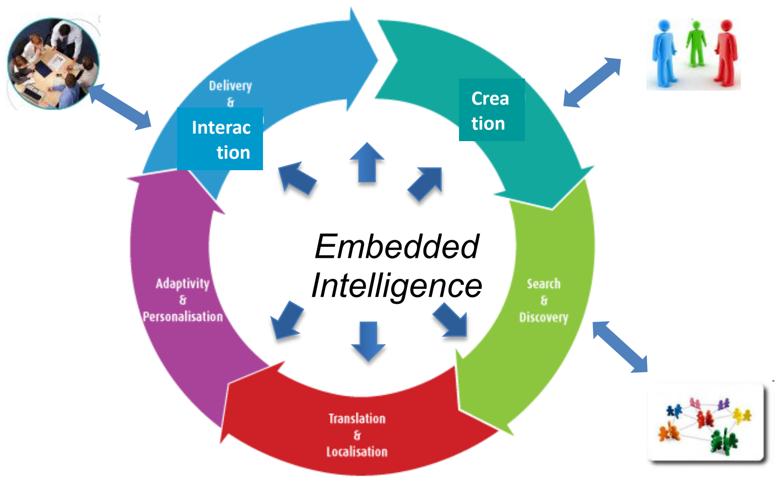

Fig. 1. Application Domain: Intelligent Content Processing for Text-based Content.

application domain. Management and quality assurance concerns are specific to this context. For instance, translatable text is capture in specific formats (e.g. XLIFF) and quality concerns are subject specific.

This challenges quality assurance across the lifecycle of content in distributed service processes. The starting point for the implementation of content quality assurance is an integrated content service process enabled by a content service bus, into which the different processing, integration and management applications are plugged into, see Fig. 2. This scenario defines our wider objective and context beyond this paper. Our aim here is to configure the quality component of this bus by a domain-specific constraints policy language [19]. The provenance model PROV [18] forms the abstract constraints layer. PCPL, the Process Customisation Policy Language [21], is part of the process platform and controls the process adaptation through its policy engine.

In order to facilitate an interoperable content and constraints notation, we assume a core RDF content metadata model (basis of a domain ontology). The data manipulation services and notification trigger functions for constrained processing activities and governance can be defined based on SPARQL query templates over the content and meta-data entities. A possible implementation through synchronous, functional granularity patterns of WSDL/BPEL needs to take this into account, i.e., a mapping to query and data model profiles would need to be considered rather than solely mappings to operations and parameters.

2.2 Challenges and Scenario Analysis

This context description allows us to extract the following research challenges:

– Process model: firstly, to define standard activities and process composition constructs; secondly, to select a host process language to realise the con-

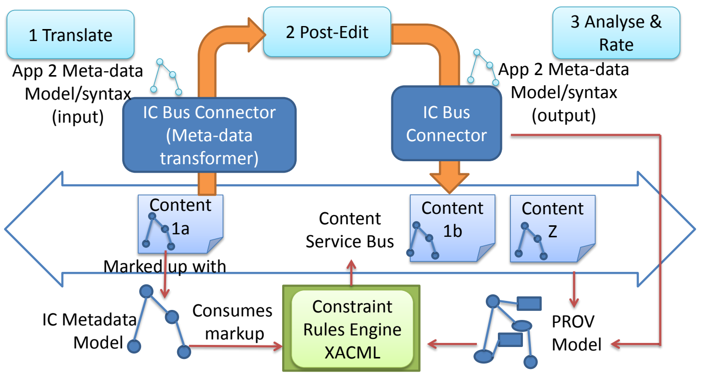

Fig. 2. Content Process Interoperability Infrastructure - Service Bus Architecture.

tent quality constraints description and monitoring; and, thirdly, to define integration of constraints into processes through a weaving technique [21].

- Content model: the definition of a content model consisting of content structure (in terms of standard format such as XML or RDF, but also more specific formats such as XLIFF [15] for content subject to translation) and its link to provenance data [18] that enables tracking and analysis.
- Quality constraints model: a rule language that allows individual constraints (conditions and processing) to be combined into policies and enforced on processes, thus requiring an adaptation and extension of normally service-centric policy languages to deal with quality concerns in a process context.

Obviously, the three individual elements are interlinked. The first step is an empirical determination of detailed requirements for quality management DSL.

A use case shall allow the elicitation of detailed requirements in order to further define the research solution. This documented requirements elicitation process is part of the DSL definition process. This elicitation results in a domain ontology, which will then form the basis of the constraints policy definition.

A sample text localisation process describes the translation of text content through a sequence of services, Fig. 3, where the corresponding provenance model gives context. The provenance model accompanies a localisation process model, consisting of the following steps:

1. Translate(SRV-TR): Text being machine translated (node 15601)
2. PostEdit(CS-PE): The machine-translated text now being posted-edited (crowd-sourced) resulting in a revised string (node 15709)
3. QA-Rate(CS-ANT): Further crowd-sourced effort is then utilised the annotate the translated string with a translation rating (node 15771)
4. Translate(EXP-TR): Given that the crowd-sourced post-editing of the machine translation produced poor results, it is decided to opt for a professional human translation (node 16723)
5. TextAnalytics(SRV-ANL): A text analytics service is then used to compare the style of the translation to a corpora in the desired style (node 16727)

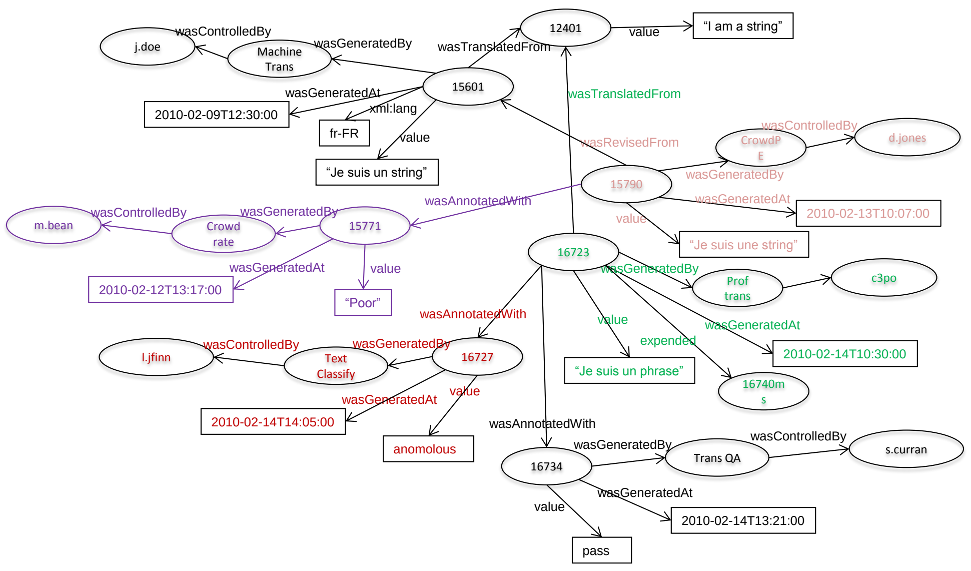

Fig. 3. Provenance Model Localisation/Translation of Text Content.

6. QA-Rate(EXP-ANL): Due to poor ratings, execute human QA (node 16734).

In an abstracted form, this is content life-cycle change or evolution that is enabled through the service process and recorded using PROV [11]. In the process description, we have added two domain-specific categorisations. Activities are the first – we distinguish standard activities in a specific domain, here text translation. Translation, post-editing, or analysis are sample core activities. Roles are the second – we distinguish three service roles in this example: software services (SRV), individual human experts (EXP) and crowds (CS).

The node references in the process refer to the provenance model, Fig. 3. This RDF graph may be build up using the provenance model as the text passes through the process to create an abstract process activity log. Some core activities have been singled out, like translate or post-edit. The process can be formalised by identifying a range of standard content processing operations: extraction, segmentation, curation, text analysis, terminology extraction, translation, post-editing, translation QA and reassembly. In addition to the operation, we can distinguish a core set of actors like EXP - human expert, CS - crowds for crowd-sourced activities and SRV - automated services. The roles can be annotated by activities, e.g. SRV-TR, if translation is the concern, or SRV-ANL for automated text analytics. These annotations are part of the domain model.

A remaining question concerns the quality aspects. We can identify the following concerns for process and content quality and process governance. The quality aspects shall be distinguished into four high-level constraint categories, which we try to motivate here through specific concerns:

– Authorisation/access control:

    - Restricting access to content, following classical access control specifications (subject, access operation, object)
    - Managing resource assignment as a mapping between content and agents
    - Location-dependent storage and content access in distributed processing
- Accountability/audit/tracing:
    - Storage/Backup/Secrecy: decide and control where are data is kept
- Workflow governance:
    - Workflow status (untranslated, postedited, etc.) ensuring that required stages of the content process are reached
    - Containment as a subprocess, e.g. audit tracking to be included
- Quality for content/process:
    - Rating of content quality (poor, sufficient)
    - Performance as a rating of process quality (slow, satisfactory, etc.)
    - Responsibility assignment and tracking as an accountability concern

The constraint format needs to take into account the PROV structure: (i) single element type, e.g., PROV Timestamps (start/end or interval constraints) or Rating (liveness constraints: should always be 'satisfactory' or better; safety constraints: should never be 'poor') and (ii) multi-element type, e.g., access control in terms of PROV (agent, activity, entity), status (entity, 'generatedBy', activity) or governance (entity, 'controlledBy', agent).

3 Domain-Specific Model for Quality Constraints

The different formats involved, based on the research concerns, are:

- Content: RDF as the canonical meta-format, which facilitates controlled access to RDF stores as the targeted storage infrastructure and modelling of different content types, such as terminology, translation memory, text (to be translated), including XLIFF where required.
- Process: For process modelling, BPEL could be assumed as a textual notation or a graphical format such as BPMN, which if complemented by jBPM and Java process engines for execution, could also be considered). Providing a runtime process execution environment is essential here.
- Constraints: PCPL [21], a process customisation policy language adapted from [14], provides a generic policy notion, extended to a process framework (similar to XACML policy language extension for service processes [20, 21]). Here, an integration with BPMN shall be implemented, following similar work on BPMN constraint extensions [2, 22].

What is needed is a domain-specific model that can be captured as a domain-specific constraints ontology with the following main concepts. Content is of specific types, based on RDF/XML, but often specifically XLIFF for translatable material. Processing activities are content processing oriented. The categorisation of constraints is specific to the different types of quality and governance constraints for content processing.

3.1 Content

The content notations involved are XLIFF to capture text and its translation with associated meta-data [15] and PROV to capture objects with origins (actors) and operations (creation and manipulation) [18]. RDF is the core format in which all data is stored and processed. Content formats are assumed to be given for this research.

A layered domain model based on content, process and provenance ontology data to support constraints shall be proposed. Some questions in relation to this model organisation have to be considered, in particular since the solution serves as the basis of a wider analytics framework for a content processing implementation. The objective here should be modularity and separation of concerns.

3.2 Provenance

A provenance model can be maintained with the processing of content. In the provenance model (RDF linked data), the following is reflected (Fig. 3).

Firstly, change operators are activities, such as GeneratedBy, TranslatedFrom, AnnotatedWith. These can be aligned to the standard content processing operations defined earlier.

Secondly, actors/participants are agent, such as m.bean, j.doe. These are named service providers that can be classified by our role categorisation scheme, e.g., the next expression j.doe:EXP→Translate links a service to an agent in charge of its execution.

Thirdly, objects are entities, such as text being translated (in XLIFF in this case, as a reflection of a specific content type).

3.3 Process

The provenance model can be presented as a process of change operations [11]. This results in a 3-layered architecture (Fig. 4). The upper layer (based on W3C PROV) is made up by the provenance model (extended state-by-state for changes). The middle layer (based on BPMN/BPEL service process descritpions) is a process model based on PROV activities (the changes themselves). Finally, the bottom layer (based on formats such as XLIFF or RDF captures the content aspect. (processed by change operations). BPMN is used here for the modelling of business processes. This can include production processes such as the content process across different participants.

Our aim is to allow a process to be adapted to domain-specific constraints. Two principle solutions to deal with policy constraints can be distinguished. A minimal invasive one weaves quality constraints into a process, where all constraints are monitored and managed by external services. An explicit extension of BPMN models constraints within the language itself and to map quality constraints into this BPMN extension [22, 2]. Regarding the second option, BPMN constraints have been proposed as a BPMN extension. Three types of constraints

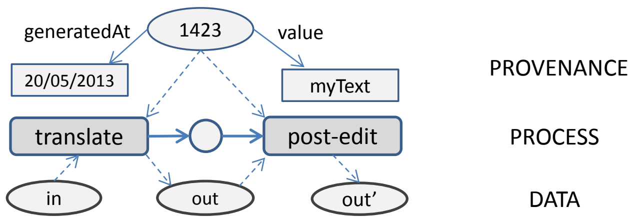

Fig. 4. Sample 3-Layered Service Process Model.

are distinguished: containment, authorisation and resource assignment. Containment means, for instance, that the activity of managing a shopping cart is a subprocess which contains an activity of removing products from the cart. While we adopt their constraints classification to some extent, our implementation will favour the less invasive solution [21] in order to achieve interoperability.

4 Constraints and Rules

Quality constraints and their formulation as policy rules are at the core.

4.1 Requirements and Examples

The first problem to be addressed is the identification of all relevant constraint types. We have already provided a classification of several quality and governance concerns: authorisation, accountability, workflow governance and quality, which takes the BPMN constraints extension into account. The objectives of rule-based process quality constraints for domain-specific process adaptation and monitoring are twofold: firstly, optimisation, i.e., to improve quality of content and the process (by looking at ratings or performance measures) and, secondly, governance, i.e., to enforce access control and privacy rules (user defined policies or legal requirements).

Constraints are technically conditions on concerns. A rule associates an action related to a condition in the Event-Condition-Action (ECA) format, that checks on an event the correctness of a condition and triggers the execution of an action, if required by the condition. Thus, based on the four constraint types, we define four rules types to link conditions and actions (illustrated by some examples):

– Authorisation/access control – example: to restrict content in data stores
– Accountability/audit/tracing – example: where are records/copies kept
– Workflow governance – example: $status = untranslated \rightarrow translate(..)$ or $status = translated \rightarrow crowdsource\text{-}PostEdit(..)$
– Quality for content/process – example content rating: $automatedQA = poor \rightarrow humanQA()$; or for process performance: $time(translation) > t \rightarrow alert$

4.2 Formalisation

A number of rule and policy and rule languages exist that would allow constraints to be specified. Examples are XACML, which allows security policies to be defined, or rule languages such as RuleML or SWRL. While these generic language are in prinicpal suitable, we need a platform, not only a language. This platform needs to allow remote constraints definition, coordination and weaving betweem service clients and providers. We follow [21] and use the PCPL policy language and its supporting platform for process customisation [19, 14] to implement PROV-based constraint policies based on individual rules. PCPL serves as a policy engine for PROV constraints. The generic PCPL is utilised here for a specific context. It controls content process adaptation. Process constrains are defined in the process adaption policy. Provenance constrains can be integrated in the process constrains as parts of conditions (XPath or SPARQL queries). PCPL policies consist of the following notational elements:

– Objects: here content defined in terms of XLIFF and XML text, processed by activities like translate or post-edit
– Activity states: capturing processing state and quality assurance state based on the domain activities
– Conditions covering the content context (owner, format etc.), the activity context (service price, failure rate etc.) or provenance/log data (authorisation, state etc.):
  - Performance/Time for processing, includes manual effort (asynchronous) and execution time of service (synchronous)
  - Authorisation: who can process/access content including the location of objects (e.g. no externalisation/outsourcing allowed as a condition)
  - Existence of entity/object in a state: e.g. translated (in XLIFF) as a workflow stage
– Actions: process adaptation decisions, which cover the constraint violation handing strategies
– Fault handlers: adaption policy execution fault handler
– Algorithms: configurations of the policy execution behaviours, such as policy conflicts

The policy model is designed for a generic process. The PCPL example below illustrates this policy definition: a document must be post-edited before sent for QA-Rating:

```
<p1:Policy policyId="QA-Rate-policy1" priority="0">
  <p1:Objects>
    <p1:ObjectsAnyOf>
      <p1:ObjectsAllOf>
        <p1:Activity>
          <Name>QA-Rate crowd-sourced</Name>
        </p1:Activity>
      </p1:ObjectsAllOf>
    </p1:ObjectsAnyOf>
  </p1:Objects>
  <p1:ActivityStates>
    <p1:ActivityState>Validating-Pre</p1:ActivityState>
  </p1:ActivityStates>

  <p1:Rule priority="0" ruleId="constraintRule-QA-Rate">
    <p1:Conditions>
      <p1:ConditionExpression type="Provenance-Context">
            <p1:Para>//Document/ID</p1:Para>
            <p1:Expr>constraintRule-QA-Rate_query.sparql</p1:Expr>
      </p1:ConditionExpression>
    </p1:Conditions>

    <p1:Actions>
      <p1:Pa-Violate>
        <p1:Violation>
          <Type>Functional:Protocol</Type>
        </p1:Violation>
      </p1:Pa-Violate>
    </p1:Actions>

    <p1:FaultHandler>
      <p1:Ca-Log level="5"> </p1:Ca-Log>
    </p1:FaultHandler>
  </p1:Rule>

  <p1:ConstraintCombiningAlgorithm type="Pa-Validate-Unless-Pa-Violate-TA"/>
  <p1:RemedyCombiningAlgorithm type="Pa-Cancel-Unless-Defined-Sequence-TA">
    <DefinedSequenceElement>Pa-Cancel</DefinedSequenceElement>
    <!-- more DefinedSequenceElement ... -->
  </p1:RemedyCombiningAlgorithm>
  <p1:SequencingAlgorithm type="Ordered"/>
</p1:Policy>
```

The policy has one constrain rule and a fault rule (the fault rule is skipped in the code). The policy targets the "QA-Rate crowd-sourced" activity before its be exceuted. The constraint rule has a condition on the provenance context or the document history. A parameterized SPARQL query checks if the current document (using the document ID as parameter) has NOT been post-edited. If the condition is true, the rule results in a functional:Protocol violation, see [21, 20] where protocol violation refers to faults related to the consistent exchange of messages between services involved in a service composition to achieve their goals. A fault rule can be defined for handling the violation. The policy will cancel the current process based the defined RemedyCombiningAlgorithm, if no remedy action was found in the fault rule for violation handling.

At a pre post-editing stage (i.e., before post-editing starts for a document translation), a request must be made for post-editing to take place, in this case through crowd-sourcing (CS). A quality rating condition could be violated, resulting in different handling actions to take place (Cancel and Skip, in Sequence). We assume respective handling algorithms to be defined.

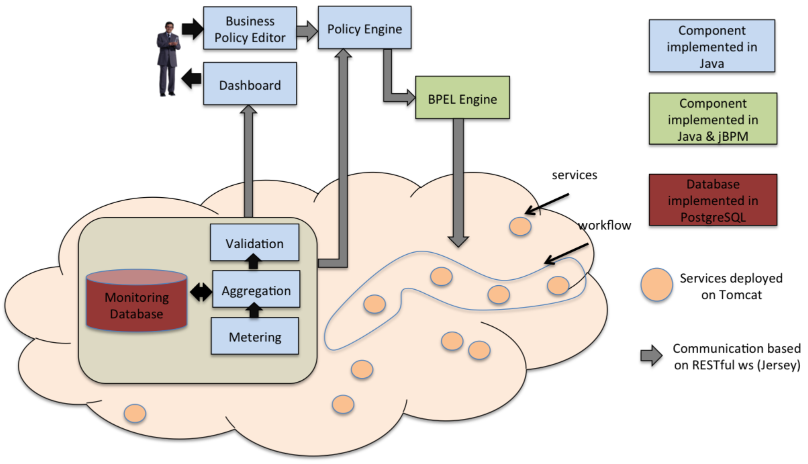
Business Policy Editor
Policy Engine
Dashboard
BPEL Engine
Validation
Aggregation
Metering
Monitoring Database
services
workflow
Component implemented in Java
Component implemented in Java & jBPM
Database implemented in PostgreSQL
Services deployed on Tomcat
Communication based on RESTful ws (Jersey)

Fig. 5. Constraints Implementation architecture.

5 Implementation – Policy Definition and Adaptation

In this section, we outline a suitable architecture for constraints definition and process adaptation, see Fig. 5. The main components are a policy definition editor, a process engine, a monitoring system and a policy engine. The diagram details the interaction between the rule engine and the process from Fig. 2. The rule engine from Fig. 2 is here decomposed into policy definition, monitoring and policy validation engine. The implementation platform is here assumed to be a BPEL engine. This engine needs to be combined with constraint weaving to allow the quality constraints to be automatically added to a business or technical process as a adaptation.

Enhanced, flexible adaptivity is a key concern, which is addressed by the architecture. This architecture allows the policies to be defined locally at the client side and then the process adapted to client domain needs and enacted by a central process engine. Thus, it allows easy adaptation to specific domains and user needs. More details about the generic architecture without the domain extension are provided in [21], which presents the two major components. This is firstly the policy language to define the constraints and secondly a coordination framework based on WS-COORDINATION, which allows the client-side specified constraints to be communicated and woven into the server-side process.

The following components for service processing describe the currently implemented service process customisation and adaptation prototype illustrated in Fig. 5: Jersey, Tomcat, PostgreSQL, jBPM and Eclipse. This prototype is a generic processes adaptation infrastructure, described in detail in [21]. It support the generic PCPL language.

Here, we utilise this generic infrastructure for service processes to configure user-specific domain constraints following the domain model approach above. Thus, the solution here is an extension of the generic policy management infrastructure for domain-specific customisation. Consequently, in this paper, the focus has been on notational rather than infrastructure aspects. Future work in the implementation context will concentrate on domain-specific implementations. For instance, a focus will be on the translation activity, where content is marked up in the XML-based XLIFF format and respective processing and quality constraints (such as *isPostEdited*) are implemented.

6 Related Work

Current open research concerns for service computing include customisation of governance and quality policies and the non-intrusive adaptation of processes with policies [19, 20, 14, 7, 17, 25]. Service management and monitoring techniques are combinations of rule or policy-based modelling languages that can be enforced at runtime. Today, one-size-fits-all service monitoring techniques exist and provide support for software systems in classical sectors such as finance and telecommunications [3]. However, their inherent structural inflexibility makes constraints difficult to manage, resulting in significant efforts and costs to adapt to individual domains needs.

We discuss related work in the field of constraints and policy definition and adaptive BPEL processes. While we have also refered to BPMN, there is more work on WS-BPEL in our context, which we discuss here. These approaches can be classified into two categories.

– In the first category are BPEL process extensions that are designed to realize platform-independence. [23] and [24] allow BPEL specifications to be extended with fault policies. Exception handling policies are bound into process schemas as a BPEL extension. The SRRF approach [13] generates BPEL processes based on the defined policies. However, binding domain-specific policies into business processes directly are not an option for our objective, as it is difficult to support user/domain-specific adaptation needs adequately.
– In a second category, BPEL engines can also be modified, but the solution is platform-dependent. The limitation of the Dynamo project [3] is that BPEL event handlers must be statically embedded into the process prior to deployment, i.e. the recovery logic is fixed and can only be customised through the event handler itself [3]. This approach does neither support dynamic policies nor a customisation and adaptation environment. The PAWS framework [1] extends ActiveBPEL to provide a flexible process that can change its behaviour dynamically, according to variable execution contexts and constraints.

Furthermore, process-centricity is a concern. Recently, business-processes-as-a-service is discussed. While not addressed here, this perspective needs to be further complemented by an architectural style for its implementation [21].

We now address specific constraints and provenance aspects. We have proposed a classification of several quality and governance concerns: authorisation, accountability, workflow governance and quality. This takes the BPMN constraints extensions [22, 2] that suggest containment, authorisation and resource assignment as categories into account, but realises these in a less intrusive, less invasive process adaptation solution.

Some provenance-enabled workflow systems have been developed [8, 6]. These workflow systems monitor workflow or process executions and record task names, execution durations or parameters as provenance information. Other work has focused on data [10, 9], recording owner or creation and modification time for provenance. Various query mechanisms such as SQL, SPARQL, and proprietary APIs are supported for different provenance data storage solutions. However, for a document or content-centric service process system where the activities of processes are responsible for content manipulations and changes, a domain-specific should be defined in a content-centric way to capture provenance information at process level and, thus, to support provenance-based process adaptation. Work in [12] focuses on using PROV to collect and analyse data in change processes. Our system is a hybrid approach, which supports both data-oriented and process-oriented provenance requirements, such as content and process activity access control. Moreover, the provenance query is integrated into a process customization policy model to enable provenance-based process adaptation.

7 Conclusions

We have proposed a notation for the description of quality and governance constraints for adaptive content processes. This is a domain-specific data/content constraints model, here applied to translatable text content. The content is of specific types, based on RDF/XML, but often specifically XLIFF for translatable material. Processing activities are content processing and translation oriented. The categorisation of constraints is specific to the different types of quality and governance constraints for content processing. A layered, modular information model covering content, processes and constraints facilitates its implementation in a wider interoperable content integration system. Interoperability is a critical driver in the application context. PROV has played a critical role, for the monitoring and recording as well as supporting the adaptivity for domains (here localisation workflow processes). Mappings between solution technologies and interoperable platforms need to be considered. This application serves as a template for domain-specific constraints and policy definition. Together with the user-based customisation architecture, service processes can be adapted to meet domain-specific needs (e.g., for the translation industry).

As part of our future work, an exploration of RDF-based SWRL rules and SPARQL queries as more RDF-interoperable notations shall be conducted that extent the PCPL approach taken so far. Also, PROV can possibly play a more central role as the process analytics model. For the exploration of the concept in this paper, the domain ontology has not been fully formalised. This would need to be done for a comprehensive evaluation. An implementation within PROV/XLIFF-based workflow system can be considered as a more targeted domain system. We have already mentioned the translation focus in the implementation section.

Acknowledgements. This material is based upon works supported by the Science Foundation Ireland under Grant No. 07/CE/I1142 as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University (DCU).

References

1. Ardagna, D., Comuzzi, M., Mussi, E., Pernici, B. and Plebani, P.: Paws: a framework for executing adaptive web-service processes. IEEE Software 24(6):39-46 (2007)
2. Awad, A., Grosskopf, A., Meyer, A., and Weske, M.: Enabling resource assignment constraints in BPMN. Technical report, Business Process Tech. HPI (2009)
3. Baresi, L. and Guinea, S.: Self-supervising bpel processes. IEEE Transactions on Software Engineering, 37(2):247 – 263 (2011)
4. Baresi, L., Guinea, S. and Plebani, P.: Policies and aspects for the supervision of bpel processes. *Intern. Conf. on Adv Information Systems Engineering* (2007)
5. Barrett, R., Patcas, L.M., Murphy, J. and Pahl, C.: Model Driven Distribution Pattern Design for Dynamic Web Service Compositions. International Conference on Web Engineering ICWE06. Pages 129-136. Palo Alto, US. ACM Press (2006)
6. Davidson, S.B. and Freire, J.: Provenance and Scientific Workflows: Challenges and Opportunities. ACM SIGMOD Intl Conference on Management of data (2008)
7. Erradi. A.: Policy-Driven Framework for Manageable and Adaptive Service-Oriented Processes. PhD thesis. The University of New South Wales (2008)
8. Freire, J., Koop, D., Santos, E. and Silva, C.T.: Provenance for Computational Tasks: A Survey. Computing in Science and Engineering, 10(3): p. 11-21 (2008)
9. Glavic, B. and Dittrich, K.R.: Data Provenance: A Categorization of Existing Approaches. 12th GI Conference on Database Systems in Business (2007)
10. Hartig, O.: Provenance Information in the Web of Data. Workshop on Linked Data on the Web (2009)
11. Javed, M., Abgaz, Y.M., Pahl, C.: A pattern-based framework of change operators for ontology evolution. On the Move to Meaningful Internet Systems: OTM Workshops, LNCS 5872, pp. 544–553. Springer (2009)
12. Javed, M.: Operational Change Management and Change Pattern Identification for Ontology Evolution. Ph.D. Thesis. Dublin City University (2013)
13. Kareliotis, C., Vassilakis, C., and Panayiotis, G.: Enhancing bpel scenarios with dynamic relevance-based exception handling. Intl Conf on Web Services (2007)
14. OASIS: Extensible access control markup language (xacml) 3.0 (2010) `http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.html`.
15. OASIS: XLIFF (XML Localisation Interchange File Format). (2013) docs.oasis-open.org/xliff/xliff-core/xliff-core.html
16. Pahl, C, Giesecke, S. and Hasselbring, W.: An Ontology-based Approach for Modelling Architectural Styles. Europ Conf on Software Architecture ECSA2007 (2007)
17. Riegen, M. von, Husemann, M., Fink, S. and Ritter, N.: Rule-based coordination of distributed web service transactions. IEEE Trans on Serv Comp, 3(1):60–70 (2010)
18. W3C: PROV-O: The PROV Ontology. (2013) http://www.w3.org/TR/prov-o/
19. W3C web services policy 1.2 - framework (ws-policy). `http://www.w3.org/Submission/WS-Policy`.

20. Wang, M.X., Bandara, K.Y. and Pahl, C.: Process as a Service - Distributed Multi-tenant Policy-based Process Runtime Governance. IEEE International Conference on Services Computing SCC 2010. IEEE Press (2010)
21. Wang, M.X.: A Policy-based Governance Framework for Cloud Service Process Architectures. Ph.D. Thesis. Dublin City University (2012)
22. Wolter, C., and Schaad, A.: Modeling of task-based authorization constraints in BPMN. Business Process Management (pp. 64-79). Springer (2007)
23. Wu, Y. and Doshi, P.: Making bpel flexible and adapting in the context of coordination constraints using ws-bpel. Intl Conf on Services Computing (2008)
24. Zeng, L., Lei, H., Jeng, J.J., Chung, J.Y., and Benatallah, B.: Policy-driven exception-management for composite web services. IEEE Intl Conf on E-Commerce Tech (2005)
25. Zhou, Y.C., Liu, X.P., Wang, X.N., Xue, L., Tian, C. and Liang, X.X.: Context model based soa policy framework. IEEE Intern. Conf. on Web Services (2010)