Software Service Adaptation based on Interface Localisation

Luke Collins, Claus Pahl
Dublin City University
Dublin 9
Ireland

Abstract.
The aim of Web services is the provision of software services to a range of different users in different locations. Service localisation in this context can facilitate the internationalisation and localisation of services by allowing their adaption to different locales. We investigate three dimensions: (i) lingual localisation by providing service-level language translation techniques to adopt services to different languages, (ii) regulatory localisation by providing standards-based mappings to achieve regulatory compliance with regionally varying laws, standards and regulations, and (iii) social localisation by taking into account preferences and customs for individuals and the groups or communities in which they participate. The objective is to support and implement an explicit modelling of aspects that are relevant to localisation and runtime support consisting of tools and middleware services to automating the deployment based on models of locales, driven by the two localisation dimensions. We focus here on an ontology-based conceptual information model that integrates locale specification into service architectures in a coherent way.

Keywords.
Web services; Service localisation; Service internationalisation; Language, governance, social localisation.

INTRODUCTION

Web-based software services, particularly in the internet of services or cloud computing context, can support users on a global scale (Armbrust et al., 2009; Buyya, Broberg, & Goscinski, 2011). In economically tightly integrated regions like Europe, where a multitude of languages are spoken, services are often only deployed to support a single language or region (EU Commission, 2010). Often, smaller organisations do not have the capacity or capability to do multi-lingual and multi-regional development. Localisation is the process of adapting digital resources like services and associated data and content to a locale, i.e. the lingual, regulatory and social environment (restrictions, rules, settings) of a location or region. With the emergence of Web and cloud services and a trend towards end-to-end personalisation of service offerings (451 Group, 2010), the need to address this wider understanding of locale and localisation in a dynamic service context is evident.

The objectives of service localisation are, firstly, to introduce service-based localisation techniques that localise software and the interaction at the service interface level, and, secondly, provide localisation techniques at runtime for dynamic service localisation and end-to-end personalisation as an adaptation technique. As there is very few related work, we focus here on defining a conceptual information model as the backbone of a wider

localisation solution. Based on this, we outline further challenges and possible architectural solutions for this context. A significant part here is dedicated to illustrating service localisation as a new concept.

In the literature, localisation often refers to either languages or physical locations only. Three different locale dimensions are the focus of our investigation that embrace these and widen the concepts of localisation and locale for services:

- *lingual localisation* by enabling service-level language translation techniques to adopt services (including API, description, models) to different languages,
- *regulatory localisation* by realising standards-based mappings to achieve regulatory compliance with laws and regulations that might vary regionally (business rules, standardised name/value mappings, currencies and units, and legal governance/compliance rules in relation to different locations or regions),
- *social localisation* by considering preferences and customs for individuals and groups or communities in which they participate (preferred media, forms of interaction and communication).

Progress beyond the state-of-the-art with respect to the following aspects is aimed at:

- localisation at the service interface (API) level - classical concepts of software localisation will be repurposed to address internationalisation at the interface level. Model-driven development including model-based mapping and translation are the techniques to develop a coherent and integrated solution across the locale dimensions here. The challenge is to define a semantic model integrating heterogeneous translation, mapping and adaptation needs within one dynamically processable format.
- adaptation and integration - software adaptation at service-level can be considerably improved by integrating linguistic and regulatory dimensions. This is a significant new direction as the current adaptation concern is on functional and software quality aspects, which are software-technical in nature.
- semantics-enhanced brokering and mediation – the matching of services is equally improved to encompass linguistic, regulatory and social aspects included in a locale-centric negotiation process and infrastructure, which requires respective coordination processes.

We motivate service localisation in the next section. The following section introduces a conceptual information model and a supporting architecture. Then, we define the localisation techniques, before looking at their implementation in the form of locale-specific process adaptation. We evaluate our solution and discuss future challenges. Finally, we end with a discussion of related work and some conclusions.

LOCALISATION-BASED SOFTWARE SERVICE ADAPTATION

The Internet of Services and Cloud Computing are two current technology trends expected to greatly impact the use of software online (Fingar, 2009). The focus here is a platform for software service localisation that makes a step from one-size-fits-all services towards end-to-end personalised service offerings based on different locales. Current service computing for international settings has limitations for localisation and adaptability for multiple but

different users (Mietzner et al., 2009; Wang, Bandara & Pahl, 2010; Metzger et al., 2012), which can, however, be overcome by multi-lingual and multi-regional localisation.

Some application scenarios can highlight the benefits of a localisation solution:

- Media store&play services. Some cloud media players are currently available in only one language. This form of end-user oriented service interaction would benefit from multilingual access. It would involve the localisation (i.e., translation) of service description values, interaction text, and other auxiliary text, which can take place statically (prepared localisation) or dynamically (ad-hoc localisation).
- Business-targeted application services. An analysis and storage service for data as a business-oriented offering could be adapted (localised) to be compliant with different standards and regulations for different regions or legal environments.
- End-user composed service processes. Take public sector applications where governance and regulations are as important as lingual aspects as an example that can be composed/configured based on individual (possibly cloud-based) services, which are adapted (possibly even dynamically) by individual users to their specific needs.
- Mediated e-commerce offerings. Telecoms-supported consumer access services are examples, where online shops use service-based support infrastructure to manage their business online.

The second scenario shall now be detailed. A data archiving service in the cloud might require business-oriented service localisation. Sales data from local subsidiaries shall be stored centrally (e.g. in a private cloud) or alternatively the use of a low-cost storage provider abroad (in a public cloud) can be considered. Focusing on regulatory (REG) and lingual (LING) localisation, but also considering social (SOC) localisation, the following is needed:

- LING: to translate service data between different languages, e.g. from English into German - "Quote" to "Angebot" - based on business standards such as EANCOM or document attributes based on e.g. the GS1 standard for documents. We can refer to the EDIFACT (United Nations rules for Electronic Data Interchange for Administration, Commerce and Transport, see http://www.unece.org/trade/untdid/welcome.html) or GS1 standards (supply and demand chains globally and across multiple sectors, see http://www.gs1.org/) for illustration.
- REG: to transform data between standards or respective variants, e.g. "Quote" translates to "FullQuotation" based on a transformation between EDIFACT variants (subsets like EANCOM, EDIKEY, or EDIFICE). Other examples are currency conversions or the transformation of rules and procedures, such as access rights to enable regulatory compliance by enabling legally required recording of activities through service adaptation.
- SOC: to select and switch between content presentations, like changing between a text-based and graphics-based presentation of data.

These data validation and archiving services might be composed by a broker that mediates this localisation based on integrated semantically enhanced locale policies. The requestor provides an abstract process that the broker implements - for example to offer locale-specific services abroad for roaming network customers (if compatible) using predefined mappings that are then deployed dynamically. End-user composition can be used where the

end-user configures a process of different services in different languages. The potential user searches for best provider internationally for both services, but not necessarily as a package. Sales records where data is validated for sector categories can serve as an example, e.g. GS1 for sector categorisations across different languages.

The following examples illustrate localisable service artefacts with some specifications for locales US (USA) and DE (Germany):

- API: *validate(dataset):result* in US maps to *Validiere(Datensatz):Resultat* in DE
- Semantic Description: *compliant(GS1,dataset) -> validated* in US maps to *einhaltend(GS1,Datensatz) -> validiert* in DE
- Contract: *per_activation = 1 USD (incl. 18% VAT)* and *availability = 98% in US* maps to *pro_Aufruf = 1.35 EUR (inkl. 21% MwSt)* and *Verfügbarkeit = 98%* in DE
- Documentation: *service validates dataset for standards compliance* in US maps to *Service validiert Datensätze und Einhaltung von Standards* in DE

Here inter-artefact relationships exist. For instance validate/validated/validates and complaint/compliance are two sets of terms that are internally cross-referenced. Some progress has been made for Web service internationalisation (Phillips, 2005; W3C, 2005). This has resulted in a broader framework, particularly addressing data formatting and conversion aspects. More complex, rule-based translations are, however, not included but will be addressed by us.

A SERVICE LOCALISATION ONTOLOGY

Semantic technologies can be applied for our localisation solution. Multi-locale modelling and service adaptation needs to supported by a new approach to semantic service description through locale-specific domain ontologies. Locale-specific domain models are needed. Ontologies provide our formal framework. In order to define these, we need to consider a data layer and a services layer, each with respective semantic annotations. A *multi-lingual, multi-ontology framework* with corresponding facts, rules and mappings is needed (Pahl, 2012). We propose adopting the *linked data* approach, which has also been followed by other service modelling approaches like USDL - see http://www.linked-usdl.org/. The XLIFF translation mapping standard, a widely used standard in the translation and localisation industry, is XML-based and can be converted into RDF. We aim to support a multi-local ontology model for data and services and annotation and discovery support.

The localisation of brokered/mediated services is the context in which service localisation takes place – which necessitates a coherent integrated information model for multi-dimensional localisation and a supporting cloud infrastructure to deal with localisation statically and dynamically. The conceptual framework we present here is consists of an information architecture, i.e. an ontology, a systems architecture and an abstract process that would be enacted on the architecture (Pahl & Zhu, 2005). An ontology is a representation of concepts (or classes) defined in terms of their properties, i.e. relationships to other concepts. These concepts can be instantiated into concrete values. Properties can be object or data valued, depending on whether they refer to other complex objects (concept – e.g. the relationship of a service to another service through invocation or inheritance) or data (e.g. the performance of a service as a numeric value). Concepts can be

arranged into a hierarchy using the sub/superconcept relationship, the subsumption ⊑. This relationship forms the taxonomy of the ontology. Concepts can also be organised in a part-of relation, the composition ▷. Concept descriptions are logically combined using conjunction ⊓ and disjunction ⊔.

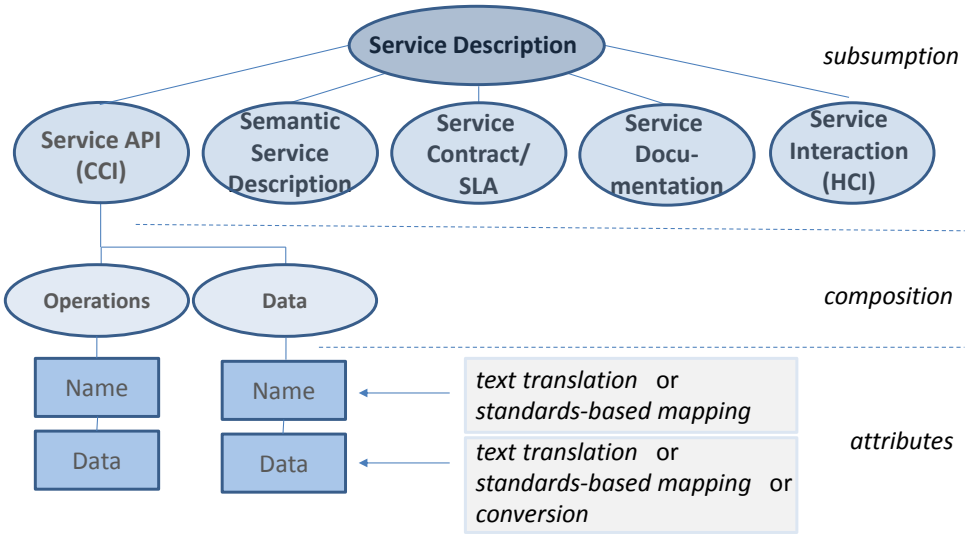

Fig. 1. Localisation Artefact Ontology with Taxonomy and Composition - Excerpt with Focus on Service API.

Localisable artefacts are service specifications and models, but also service contracts and other documentation addressing e.g. interaction in the form of messages and also licensing, usage and access rights, see Fig. 1 for an overview.

The ontology expression

$$ServiceDesc \sqsubseteq ServiceAPI \sqcap SemanticServiceDesc \sqcap ServiceSLA \sqcap ServiceDoc \sqcap ServiceHCI$$

defines a hierarchy of concepts that can be further detailed as a composition

$$ServiceAPI \triangleright Operations \sqcap Data \qquad \text{and} \qquad Data \triangleright Names \sqcup Values$$

We use a rich ontology notation based on description logics with subsumption (subclasses) and composition (part-of relationship) introduced in (Pahl, Giesecke & Hasselbring, 2009). This rich language allows a layered organisation of different functional aspects. The upper subsumption part defines a taxonomy, which acts as a framework to define structural composition.

As part of localisation, translation between controlled vocabularies is necessary. Underlying data schemas are based on standards and mappings that need to be formalised. If services are not locale-independent, localisation is needed. A *locale model* is based on locale-affected attributes like time/data format (the localisable artefact is data), language (the localisable artefact is interaction like text and/or dialogues), and collation/ordering (the localisable artefact is generally data). User-specific locale models (realised as user locale

policies) are applied at the service endpoints after a locale negotiation process with the provider. Services can be functionally described in terms of their interfaces (in WSDL): data types, data, operations. Other, more comprehensive models like USDL support functionality, quality, locale (language, currency, units, legal/tax), context (location, device, platform), and documentation (manual, help). Thus, the localisable artefacts are (as in Fig. 1):

- service interface (API): data-level internationalisation and transformation for the different locale dimensions that apply in particular to data (translation based on e.g. EAN Identification or EANCOM/EDIFACT at schema or data-level).
- semantic service description and contract (SLA): aspects like licensing, usage and access rights, and other metadata aspects are subject to translation between controlled vocabularies (which is data-level translation).

The ontology layers shall be further explained:

- *Taxonomy level (subsumption).* Service descriptions are linked to the generic service description concept through subsumption.
- *Composition level*. Each of the description types is then decomposed (using the is_part_of composition) into individual elements, which can then be localised. Composition is a necessary as subsumption (subtype) does not apply between an artefact and its parts. For the service API, operations and data (in the form of input and output parameters or messages) are the individual, localisable elements.
- *Functionality level (attributes)*. Each localisable element has attributes that characterise the localisation technique to be applied. The localisation type can be text (translated using usual MT techniques), standards-based data (the translation is defined in a predefined glossary) or conversions (e.g. for statically defined measurements or dynamically defined currency conversions). For the given ontology excerpt above, the translation type attribute is given.

Composition and attributes are intra-artefact properties. In addition, there are inter-artefact properties in the ontology. Two examples are is_defined_in and is_explained_in for cross-artefact properties, e.g. a contract refers to a definition of an operation in an API or data explained in documentation.

A *locale policy language* for locale description (for both provider and user) needs to be supported. The suggestion is a (locale) policy language that captures required localisations. This language needs to consider interoperability (e.g. with respect to WS-Policy or XACML). A further requirement is to dynamically instrument service processes with policy-based localisation functions. The PCPL policy language supports this (Wang, Bandara, & Pahl, 2010). The localisation ontology is the foundation, on which then rules (defining localisation policies and actions to be executed for localisation) are introduced.

SERVICE LOCALISATION ARCHITECTURE AND PROCESS

Two infrastructure and mediation techniques are needed to realise a service localisation architecture and processes enabled by this architecture:

- Coordination: a localisation coordination strategy and respective protocol to implement the locale negotiation and coordination processes for different locales - essentially a rule-based solution with conflict resolution.

- Service Mediation: a mediation architecture where an intermediary/broker mediates between clients and providers at different locations. Solution components are based on hybrid techniques for translation and model mappings (across languages, regulations and social contexts) and process adaptation techniques.

We now discuss the principles of a service brokering solution for the adaptation and mediation of services, see Fig. 2. Multi-dimensional user and service models and mappings require techniques for adaption and personalisation. A *user model* developed around a notion of locale achieves cross-lingual, cross-regional and cross-social coordination. To match user models and *service profiles*, we need to formalise and automate mappings and mediation between the models. On the regulatory localisation side, conversions between formats and regulations need to be defined. On the lingual localisation side, meta data translations and translation of values and text need to be facilitated, in both guided and unguided forms. Equally, social mappings based on user profiles is required.

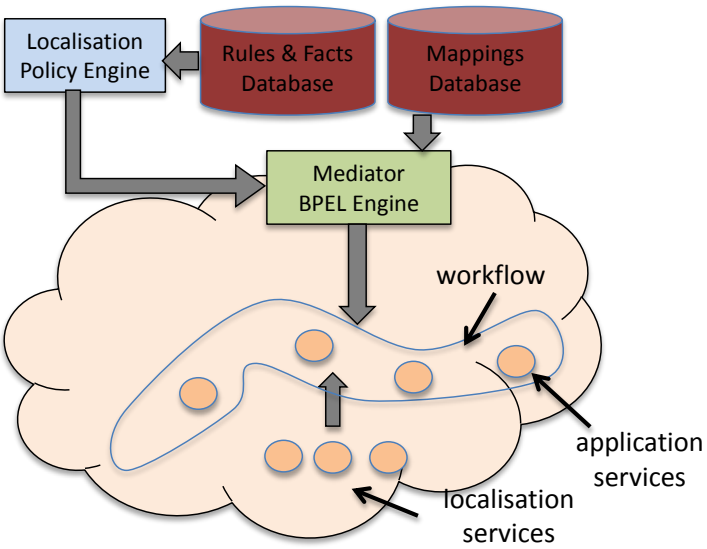

Fig. 2. Localisation Systems Architecture.

A mediation architecture is a process framework for locale mediation based on negotiation and localisation implementation activities (Pahl, Giesecke & Hasselbring, 2009; Barrett et al., 2006). We propose a *dynamic localisation engine* made up of services providing discovery, locale negotiation (static and dynamic, to generate SLAs and BPEL code, respectively), localisation (lingual, regulatory social, prepared and dynamically). This engine enacts a *localisation process* to govern individual activities (Karastoyanova & Leymann, 2009). A repository is needed to ensure the efficient and effective operation of the engine, which maintains pre-translated content and data/format mappings as translation memories. A locale can be seen as a specific type of context. *Context- or locale-driven service provision* is our aim. From a cloud perspective, this can also be seen as a *multi-tenancy* problem, where a cloud tenant is defined by its locale. These aspects are further discussed later on.

SERVICE LOCALISATION TECHNIQUES

Policy Language and Localisation Model

The core of the proposed localisation techniques is a rule language (Kharbili & Keil, 2010; Rosenberg & Dustdar, 2005; Weigand, van den Heuvel & Hiel, 2008), which is an interoperable locale policy language that is needed here (in addition to languages like WSDL that only capture the functionality side of the service API) to define locale descriptions for provider and user. This policy language is based on rules with underlying ontology support (OWL) for the conceptual aspects. A locale policy defines the individual rules and instrumentations that characterise a locale. In Fig. 3, we define a layered localisation model that connects ontology-based locales (from Fig. 1) and rule-based locale mappings:

1. An ontology-based base layer captures the different translation types. The ontology defines the localisable artefacts and their relationships (see Fig. 1). Each unit is characterised in terms of a number of attributes:
    - translation unit type: API (data, operation), semantic description, SLAs, or documentation are examples.
    - translation type: standards-based, ad-hoc text translation, conversion based on distinct repositories, called translation memories (TMs) here.

   An example for a unit definition in terms of the artefact ontology is

   Currency hasType API.Data or *SDescr hasType Text*

   This adds functionality (processing) to the taxonomy/composition ontology (Fig. 1).
2. Basic rules are provided, including *Locale* or *hasCurr* based on which specific locales can be defined. An example is for a *Currency c*

   $UKLocale(?l) \leftarrow Locale(?l) \wedge hasCurr(?l,?c) \wedge ?c=GPB$

   Higher level rules allow a locale to be inferred from partial knowledge, e.g.

   $?c=GBP \rightarrow ?l=UKLocale$

   to detect inconsistencies.
3. Depending on provider and consumer locales, translations might need to be performed. Translations are guided by the locale definitions and utilise mappings defined in the translations layer (either as pairwise translation units (e.g. text) or as executable conversions (e.g. for measurements or currencies). The overall translation is dynamically assembled from the translation base (translation memory). The conversion rule for currencies can be dynamically created:

   $UKLocale2DELocale(?l1,?l2) \leftarrow$
   $hasCurr(?l1,?c1) \wedge hasCurr(?l2,?c2) \wedge ?c2 = convertCurr(GBP,EUR,?c1)$

   Mappings appear primarily as translations between locales. Iin some cases, consequential actions need to be added. For example, dynamic currency conversions can be added or logging of activities for compliance reasons is added. Actions consequently comprise another type of activity in addition to translation. Localisation might entail additional activities like adding logs for activities.

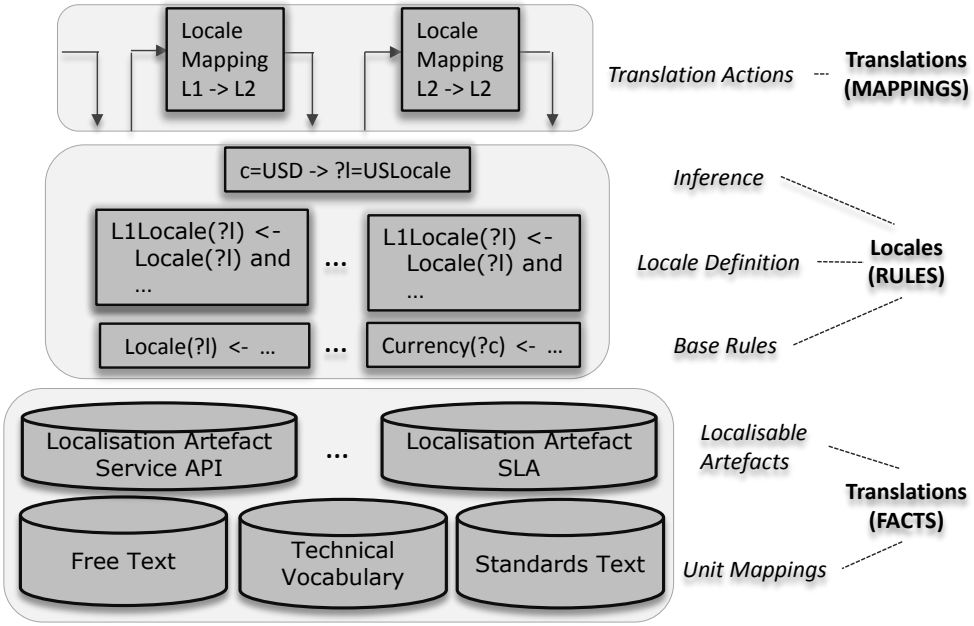

Fig. 3. Conceptual Localisation Model – Rules and Mappings over Ontology.

Examples of locale configuration are the metric system of units or a tax system. A locale mapping is compliant if it does not violate any locale configuration. While a coherent conceptual model is presented in Fig. 3, some support is needed for the localisation dimensions:

- Linguistic localisation and machine translation: the challenge is localisation of service artefacts with little textual context - at design and run-time.
- Regulatory localisation and governance: integrated and coherent adaptation to different regulatory standards and procedures based on semantic, rule-based techniques using multi-locale modelling and discovery using ontologies; and in addition multi-dimensional user and service models and mappings for adaptation.
- Social localisation: preference identification using rule-based techniques; conflict identification.

Consistency is a key requirement behind the rule-based approach. Locale mappings need to be consistent. A translation (top layer Mappings) is *consistent*, if two (or more) rules apply to different translations or mappings to the same unit. An example is currency conversion. Inference rules in the rule layer provide the mechanism to detect inconsistencies and enforce correct mappings. An example are countries that use a coherent set of measurements, e.g. the metric system. Also, currencies are linked to fiscal systems defined in a specific language. Since locale mappings are often dynamically assembled from translations, their composition needs to be checked for consistency.

Lingual Localisation and Translation

Localisation of artefacts with limited textual context – in both prepared and dynamic settings - is the focus. Translation of technical content based on reduced-context machine translation techniques is an avenue (van Genabith and Crouch, 1999). Furthermore, tailored techniques are needed to facilitate reliable translations between technical content

representations, e.g. formalised, technical content such as ontologies, service API and service models (van Genabith, 2001). Multi-lingual ontologies form the centrepiece of the solution. They provide a mapping and glossary (as core translation memory). For enable effective processing, we need to consider merging of prepared translation material with other translations generated dynamically. The key challenge is the translation accuracy and quality, which needs to be trustworthy in automated environments. Ontology translations of technical and business application domains and translation of controlled technical context with little contextual information are the solution (van Genabith and Crouch, 1999; Graham & van Genabith, 2009), consisting of translation techniques and supporting translation memories using predefined units (words/phrases) for standardised multilingual glossaries and variable units (words/phrases) involving statistical machine translation techniques based on ontological relatedness to guide the translation. The XLIFF format (XML Localization Interchange File Format) is an OASIS standard for localisation exchange - http://www.oasis-open.org/committees/tc$_$home.php?wg$_$abbrev=xliff – used to capture translations.

Translations units (pairs here) are kept in translation memories, formalised as

hasTransl(?t1.?t2)

where *t1* and *t2* are the XLIFF source and target translation units. This is an ontology-level representation of a localisation activity. Rules that guide the application of localisation activities are defined in terms of conversion and translation rules, similar to those for currency conversion. A deeper investigation of computational linguistics techniques is, however, beyond the scope here.

Regulatory Localisation and Governance

Regulatory localisation and governance through the adaptation of artefacts to different regulatory standards and procedures is based on localising regulatory concerns, which are often captured in (legal) terms and conditions. Regulations apply for the identification and description of business entities or the way service processes are handled (Leusse, Dimitrakos & Brossard, 2009; Wang, Bandara & Pahl, 2009). Furthermore, regulatory locale aspects like tax, currencies or units need to be considered (Bandara, Wang & Pahl, 2009; Phillips, 2005).

The objective here are specifically multi-facetted rules, which are modular and composable and that enable interference checking. For instance for data object localisation, we a multilingual, multiregional schema mapping and integration technique is needed to adapt information to regulations in different regions, in possibly different languages (Fu, Brennan, O'Sullivan, 2009). For SLA translation as another concern, enabling mappings of a SLA into a different locale that needs to consider the respective standards and procedures is a solution. A translation technique needs to implement quality assurance and the accountability of the translation. We can distinguish physical units like length, size, weight, and also colour and financial aspects like pricing, payment, currency or tax. Consequently, we require regulatory rule-based information translation techniques based on predefined mappings with fixed conversions between lengths and other units and consequential actions

(instrumentations) that are needed when dynamic conversions (an example is currency) or auxiliary actions (logging for accountability) are entailed. Individual definitions are not the core problem – which is rather keeping consistency.

Social Localisation

Groups of individual differ, depending on common characteristics of their members. These groups might form social groups of friends, e.g. young adults at school or university, or groups of team members in organisations, e.g. which could be sports clubs as well as companies. An example of a social group that consumes content differently from others are young, media-competent adults, which might prefer video-based content over text.

$$YoungAdultsLocale(?l) \leftarrow Locale(?l) \wedge hasMediaPref(?l,?m) \wedge ?m=video$$

Specifically, (social) media services would exploit this type of adaptation setting.

The technical problems are preference identification using rule-based techniques and conflict identification. A range of preference could result in conflicts if they were to be met for a specific application service.

LOCALISATION PROCESS AND ADAPTATION ARCHITECTURE

Localisation Process

A service localisation framework is based on guided translations (following ontology structures), which may be pre-translated. Static value mappings (cross-language or cross-regulation) and also dynamic value mappings need to be implemented. A static setting means that localisations are prepared. The solution architecture needs to enable providers to prepare material for a range of locales in advance and to check their quality. An intermediary (like a mediator) deploys and executes these techniques depending on user profile and a negotiation process, from which then SLAs are formed. These are used to govern the invocation of services. In a dynamic setting, a mediator selects the required services (involving query translation), implements the negotiation using mappings (based on closest profiles) by using localisation and quality assurance services. The overall quality will, of course, vary between in-advance and fully automated scenarios.

An orchestrated service process in WS-BPEL integrates application services and localisation support services. These techniques can be facilitated for service localisation through an adaptation and instrumentation of an application service process (Wang, Bandara & Pahl, 2009). The mediator links clients (at different locations) and providers (at different locations) by enacting this core *orchestrated process with localisation* adaptations (Pahl, 2010).

Different localisation orchestration *patterns* emerge. For a *single-provider setting*, one provider supports n > 0 locales; for a *market-place setting*, a client uses a composition of different provided services n > 1. Both actually enable a localisation process with locale

negotiation (brokering) and localisation (mediation) implementing a localisation coordination protocol (e.g. in the form of a BPEL process).

- First: Dynamic Generation - for regulatory localisation, an aspect-based instrumentation can be generated on-the-fly for each defined locale, allowing the user the control over locale definition
- Second: Configuration Management and Generation - different endpoints and their respective bindings for different locales are generated for a localised API. Then localisations (translations and mappings) are applied. Which of these to use can be dynamically decided.
- Third: Negotiation and Coordination – an exchange and agreement on locale policies through SOAP headers for coordination based on (Wang, Bandara, & Pahl, 2010).
- Fourth: Architecture – the data integration layer based on ontologies; the service localisation layer is based on adaptation (user model) and regulatory, lingual and social localisation; and management through a locale negotiation process.

The orchestration process that governs the activities is a basic sequence with *Negotiation*, followed by *Mediation & Localisation* and *Service Execution*. Abstract services and locale policies are the basis of negotiation. From these service endpoints give access to the application services or instrumented processes. These in turn need a selection of core services for translation LING, adaptation REG, and personalisation SOC and also the necessary instrumentation, i.e. localisation as a service implemented as a BPEL process. Take a sample application service

provider-Service

that is adapted into a service orchestration

if negotiate(locale(user-US) locale(provider-DE))
then localised-Interface := localise(US-locale, provider-Service);
result := invoke(localised-Interface);
de-localise(result)

of application service *provider-Service* and the respective localisation support services.

Architecture Implementation

The architecture for locale adaptation is based on the localisation process, see Fig. 4. The main components of the architecture are

- a rule definition editor to define the translation and conversion rules
- a process engine (jBPM and Tomcat) to execute application and localisation services,
- a monitoring system to determine current settings (based on PostgreSQL) and
- a rule engine to execute the rules.

Figure 4 illustrates the interaction between rule engine and process. The rule engine is decomposed into localisation policy definition, monitoring support and policy validation. The implementation is assumed to be a BPEL engine, which needs to be combined with policy weaving to allow the localisation policies to be automatically added (i.e. adapted) to an application process. We have used Java as the implementation language.

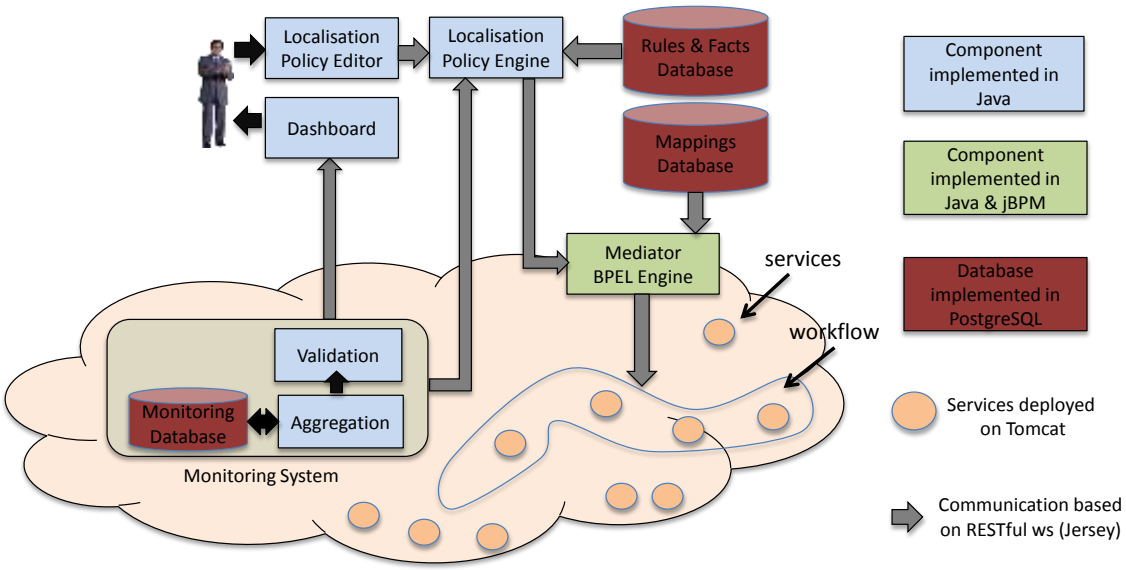

Fig. 4. Localisation Policy Orchestration Architecture.

Enhanced adaptivity is addressed here. The architecture allows localisation policies to be defined at the client side. Then the process is adapted to client domain needs and enacted by a central process engine. This allows easy adaptation to specific locale settings. Details about the architecture without the localisation domain extension are provided elsewhere (Wang, Bandara, & Pahl, 2010). There are two major components. Firstly, a policy language to define localisation policies and, secondly, a coordination framework based on WS-COORDINATION allowing client-side specified policies to be communicated and woven into the server-side process. In our implementation of the service process customisation and adaptation prototype illustrated in Fig. 4, we used Jersey, Tomcat, PostgreSQL, jBPM and Eclipse. This prototype is a generic process adaptation infrastructure, described in (Wang, Bandara, & Pahl, 2010). This architecture is implemented around the generic PCPL policy language that we developed for constraints definition. This generic platform for service processes was adopted to configure user-specific localisation constraints following the localisation model. Our solution is an extension of the general-purpose policy management solution for domain-specific customisation. Consequently, here in this investigation, our focus has been conceptual rather than on infrastructure aspects. Future work in the implementation context will concentrate on domain-specific implementations. A focus is on translation, where content is marked up in the XLIFF format and respective processing and quality constraints are implemented.

VALIDATION AND EVALUATION

A use case based evaluation shall help us tp validate the utility and efficacy of the localisation techniques. This shows that the proposed techniques are indeed necessary and beneficial. We refer to a use case scenario, which extends the earlier justification of the need for a service localisation solution as a domain-specific adaptation technique. We also address technical concerns such as performance overhead at runtime as dynamicity was a concern.

Use Case Validation

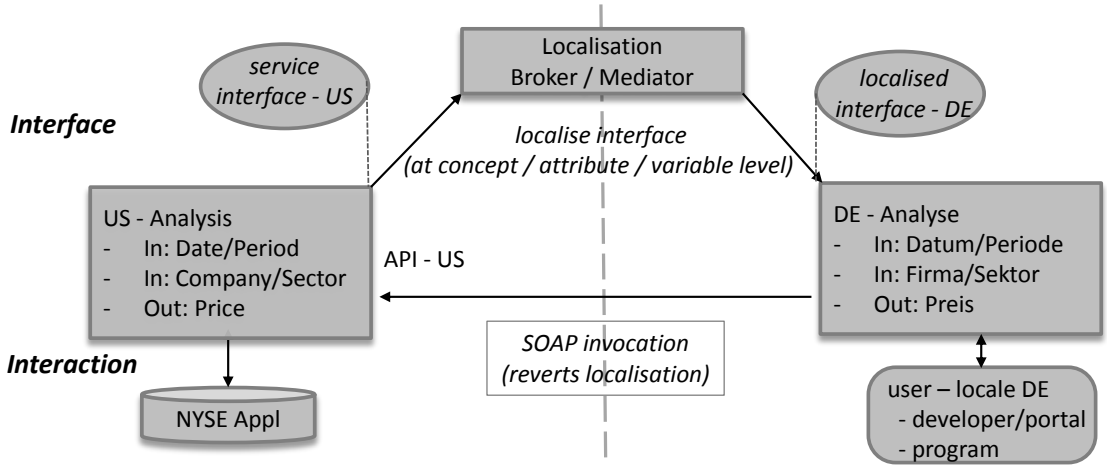

Fig. 5. Localisation of Stock Market Analysis Feature - Focus on Service API.

In addition to the demonstration of the need for localisation, we chose another use case to discuss more detailed requirements. We use an environment that provides service-level access to stock market information and respective analyses. This is based on a case study using real financial stock market information and analysis services from http://xignite.com/ and http://deutsche-boerse.com . A German user might want to access data from the New York stock exchange (NYSE), which is provided in an English format. We present a scenario in which the user can implement a locale-compliant service interface that would allow technical interaction of service interface and description aspects in German. At the application-level, two sample calls of a data analysis service for the two chosen locales (here US-locale with English as language and USD as currency and DE-locale with German as language and EUR as currency) could be:

Analyse(10/30/2011, logistics) -> 3.82 USD and *Analysis(30.10.2011, Logistik) -> 4.23 EUR*

The localisable artefacts in this scenario are the following

- Date: a format change is needed - which also applies to time and collation aspects,
- Sector: data values describing an industry sector are localised based on a translation between standardised terminologies - which also applies to product categories,
- Language: operation names (and possibly other interface and software model elements) are translated between languages,
- Currency: values are converted – in the same way as other measurements and units.

This list can be extended. For instance, different regulatory environments can be based on multilingual and standardised glossaries and dictionaries or calculations and conversions can be based on rules (fixed) or repositories (dynamic); tax rates and customs duties can be added if products are sold; any messages including help and error messages to which text translation would be applied. Typical examples for technical concepts (terms) that require translation in the stock markets (or other financial) context are (*average price - Durchschnittspreis*), (*main trading phase - Haupthandelsphase*), or (*volume weighted*

average - *volumengewichteter Durchschnitt*). These are based on accepted, in some cases standardised vocabularies. Some examples might be defined in terms of classification and categorisation standards: (*logistics* - *Logistik*) for a sector or (*dairy* - *Milchprodukte*) for product categories.

In the example illustrated in Fig. 5, the user-DE can discover services based on a German specification and can invoke them based on a German interface. An analysis service provider can add a DE-locale to its default US-locale policy. This would allow a correct match in a full negotiation process in which a user searches for services that are provided in a locale-specific way. Here, the provider is able to support US-to-DE locale mappings if necessary. In an architecture implementing this, the service instrumentation would yield a process to be enacted, rather than just a single SOAP request (see Fig. 5). This process could for instance comprise service invocation and logging (of locations) for accountability purposes where the location is a parameter. This then indicates where and how records are kept (an important option if required by privacy laws). A coordination protocol governs the exchange of locales and the subsequent SOAP-based application interaction.

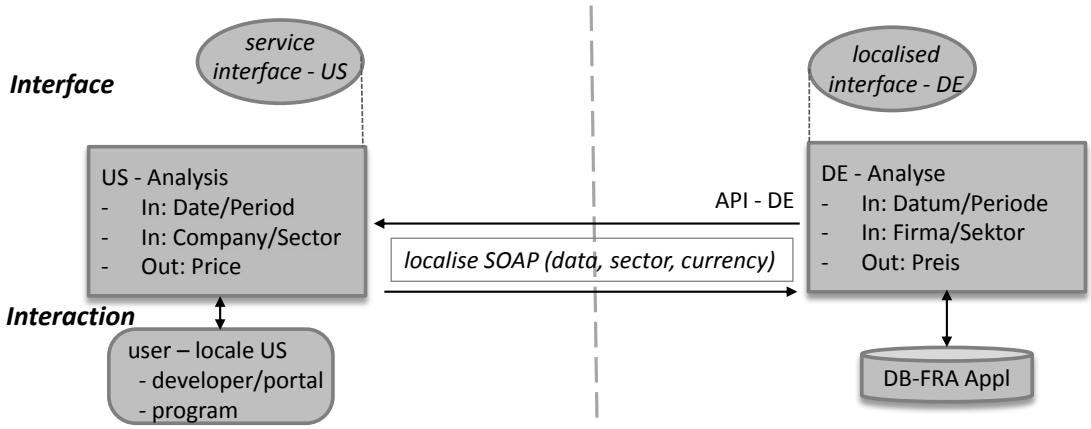

Fig.6. Localisation of Stock Market Analysis Feature - extends NYSE-specific Analysis Service to include DB-FRA.

The scenario could be further utilised to allow an American user (with the locale US) to access a German-language stock market information provider, e.g. for Deutsche Börse, Frankfurt (DB-FRA). Localisation can be provided on a SOAP level, see Fig. 6, for automated service activations and interactions.

The discussion of this real-world scenario demonstrates the utility and efficacy of our localisation solution as a domain-specific adaptation framework, which benefits from a targeted framework of specific models, rules and support services.

Technical Evaluation

Besides utility and efficacy, the central technical concern of particularly dynamic localisation is runtime performance, as only some aspects can be pre-computed and these need to be integrated at runtime with other dynamic concerns. Scalability is another concern to be considered.

The localisation architecture presented in Fig. 4 was fully implemented as a Java solution, which has the objective of studying the feasibility of the conceptual solution. We assess this based on the criteria performance and extensibility. These have different effects on the end-user experience of the product. These criteria are key performance indicators and essentially critical success factors of the localisation platform described.

Insufficient performance often affects software significantly as different users consume a service at the same time. The core problem is the overhead created by adding localisation support dynamically to service provisioning. This is impacted on by the policy engine, which we separated from the process engine to allow client-side constraints to be facilitated. The remote instrumentation of processes with these constraints slightly impacts on performance. Our evaluation shows an acceptable overhead of about 10-15% additional execution time for fully localised services (i.e., the localisation involving different localisation aspects). This overhead is still relatively low compared to other performance factors like network latency and the average service execution time anyway (Wang, Bandara, & Pahl, 2010). As the application deals with different users, the latency would also increase as a consequence of extra load put on the platforms services. This clearly defines latency as a KPI. Latency is also an area to be looked at as integrating the localisation solution into the workflow of an existing process has the potential to add lag-time. This lag-time arises due to time required to compute and also initialise the various variables. The propagation latency is presented in Table 1. Please note that the numbers can be affected by environmental changes or the locale we are transforming from and the locale we are transforming into.

Table 1. Latency Table - Localisation of Service

Service	Prior (ms)	Post (ms)	Delta t (ms)
NASDAQ	132	182	50
FTSE	110	152	42

The chosen software architecture has a major influence on the performance. We aimed to improve performance by using pre-translated aspects (using stored mappings). A closely related aspect is scalability. This becomes more important when a service may have large numbers of users. Scalability has not been empirically addressed for this phase of research and will be evaluated in later prototypes.

- Some platform components would need modification to efficiently enable the infrastructure to vertically scale-up or scale-out. Solutions here are stateless programs and data externalisation. Using the rule base and the pre-translation repositories, some beneficial architectural decisions have already been implemented.
- Horizontal scalability - i.e., the addition of more localisation concerns - is easily be supported by the modular mediator architecture (to be addressed further below as part of the extensibility context from an implementation view).

Another approach to address scalability is a tuple space-based coordination (Pahl, 2010), which allows a flexible and elastic assignment of localisation services to multiple requests.

Extensibility becomes an important concern when dealing with complete solutions like a localisation platform. During an initial development of a system, often features need to be

included due to various constraints. For the localisation platform described here, some localisation services where not developed, some of which include a service to handle taxation. However, the platform was designed to be extendable. At a platform level, this is extensible, allowing the addition of further services and the support for more locales.

Discussion – Merits, Limitations and Applicability

We now discuss and summarise the merits, limitations and general applicability of the solution. The benefits have been highlighted throughout, but a limitation is the possible diversity of locale settings. Locale definitions might vary from domain to domain. For business applications, in finances or e-commerce, lingual and various regulatory constraints apply. For social media application, some of the latter (e.g. standards) are likely to be replaced by social constraints. A wider range of different applications from social networking to entertainment and gaming to education might require the definition of more strongly group-oriented than region-oriented locale definitions.

Based on the evaluation carried out in this section, we have demonstrated the overall applicability and efficacy of the solution, using a dynamic, real-world scenario. However, further validations would need to be carried out, e.g. regarding the data model and an additional empirical validation of a social media application (we have focussed in this presentation on business applications). However, a more complete model would need to be discussed - while here the focus has been on the higher-level structure of the ontology model.

RELATED WORK

Several aspects are related to the adaptivity context here, although we approach the problem in a different way compared to other research publications (Fuji & Suda, 2009; Kharbili & Keil, 2010; Mietzner, Leyman & Papazoglou, 2008; Weigand, van den Heuvel & Hiel, 2008). Work covered by for example EU-supported research projects, like SOA4All (www.soa4all.eu), ACSI (www.acsi-project.eu), 4Caast (4caast.morfeo-project.org) or mOSAIC (www.mosaic-cloud.eu) address end-user adaptivity using generic semantic models (SOA4All), software-centric coordination and marketplaces (ACSI, 4Caast), or multi-cloud provisioning (mOSAIC). Our framework is orthogonal to these works on end-to-end offerings. It is different in its multi-facetted character focusing on linguistic and regulatory aspects. More general, our proposal relates to the following aspects:

- Adaptation and Data Integration: software adaptation at service-level and data integration are common techniques. Examples are schema mappings for data integration, where consistency and semantic preservation are key concerns. We followed here ontology-based approaches, using these for semantics preservation, but added a multi-lingual layer using ontology mappings. In a similar approach, service/process instrumentation is used to add enhanced processing abilities, but our that focuses on lingual, regulatory and social concerns within a rule-based framework is novel.
- Service Context-awareness has been covered by a range of contributions such as (Bandara, Wang & Pahl, 2009). The notion of context is similar to that of locales, reflecting properties of the execution or client environment. Context, however,

generally does not include lingual, regulatory and social aspects. We suggest to adopt and extend common adaptation and instrumentation techniques (Baresi & Guinea, 2005; Kapitsaki et al., 2009).
    - Multi-tenancy is a cloud computing concern (Mietzner et al., 2009; Wang, Bandara & Pahl, 2010) that requires in a fully customised framework different solutions for users with different needs, to keep them separate.
    - Semantic Technologies: matching of services and supporting the negotiation process and infrastructure (Doulkeridis, Loutas & Vazirgiannis, 2006; Fuji & Suda, 2009; Pahl & Zhu, 2005). Through ontology mappings, multi-lingual terminologies and multi-regional regulations can be captured and dynamically processed.
- Software Localisation: localisation of software normally addresses the human interaction with software, i.e. messages and dialogues produced by the software. We focus on localisation at the service interface level (i.e. machine-based interaction) through internationalisation and ontology mapping and translation. Current technology on service internationalisation, which is the closest, is supported by the W3C Service Internationalisation activity (Phillips, 2005; W3C, 2005). The focus in that initiative is data localisation, specifically for dates, currencies and units and common approaches to collation. The respective solutions are conversions - statically defined between units or dynamically defined between currencies as examples. Lingual aspects or more complex regulatory or business aspects are not addressed, although in the documentation for instance taxation as a sample concern is mentioned.

We have presented a basic localisation conceptualisation in (Pahl, 2012). Here, we have extended this model by adding a third dimension (SOC) to provide a solution beyond the business application domain. This investigation here also formalises the conceptual model as an ontology. Furthermore, we deepened the process aspect and also implemented and evaluated the solution systematically.

CONCLUSIONS

Service interface localisation is a form of service personalisation and adaptation to specific use needs. Our investigation addresses service engineering methods and tools allowing a service to be adapted to different locales. Services are readied for their integration by providing localisation of services as software in combination with related content and data processed and communicated by these services. We have discussed a range of techniques to enable the extension of service offerings for different markets – something that is of significant economic benefit. Localisation can bring products and here particularly services to markets that are otherwise not accessible.

Multi-locale services can support interoperable services and thus contribute to a market of services by allowing services to be internationalised and localised. This is of importance for regions such as the EU which has 27 members and an even higher numbers of local languages and regulatory systems or even different social context. Service localisation adds to the availability of platforms for simplified development and deployment of value-added services through service front-ends to localise and manage multi-locale services. Localisable services enable users to develop, select, combine and use value-added services and to allow providers to provide new solutions without in-house localisation capabilities.

The objectives of this investigation were two-fold. Firstly, we have introduced a conceptual framework to capture the key concerns of multi-lingual, multi-regulatory and multi-context service localisation as a rule- and ontology-based information framework. Secondly, our investigation aimed at justifying the need for service localisation, e.g. for Web and cloud based services. In this vein, we identified current concerns and directions to be addressed. The focus of our paper was, however, limited. Aspects not included like program-level localisation, end-to-end cloud personalisation and multi-lingual clouds are examples of aspects left out at this stage. It is also evident that more implementation work needs to be done to aid further empirical evaluations. So far, the conceptual solution is only motivated and justified through the discussed case studies and the implementation and only been used for performance evaluation.

REFERENCES

451 Group (2010). Report on Cloud Computing 'As-a-service' market sizing - Report II. 451 Research.

Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., & Zaharia, M. (2009). A view of cloud computing. Comm. ACM 53(4):50-58.

EU Commission (2010). Report on The Future of Cloud Computing - Opportunities for European Cloud Computing Beyond 2010. European Commission.

Bandara, K.Y., Wang, M.X., & Pahl, C. (2009). Context modeling and constraints binding in web service business processes. Proceedings of the Workshop on Context-Aware Software Technology and Applications CASTA. pp. 29-32. ACM Press.

Baresi, L., & Guinea, S. (2005). Towards Dynamic Monitoring of WS-BPEL Processes. Proceedings International Conference on Service-Oriented Computing ICSOC'05. Springer-Verlag, Berlin, Heidelberg, pp. 269-282.

Barrett, R., Patcas, L.M., Pahl, C., & Murphy, J., (2006). Model driven distribution pattern design for dynamic web service compositions. *Proceedings International Conference on Web Engineering* ICWE '06. ACM, New York, NY, USA, pp. 129-136.

Boukadi, K., Ghedira, C., Chaari, S., Vincent, L., & Bataineh, E. (2008). CWSC4EC: how to employ context, web service, and community in enterprise collaboration. In Proceedings of the 8th International Conference on New Technologies in Distributed Systems (NOTERE '08). ACM, New York, NY, USA, Article 6 , 12 pages.

Buyya, R., Broberg, J., & Goscinski, A. (2011). Cloud Computing - Principles and Paradigms. New York, US: Wiley.

Doulkeridis, C., Loutas, N., & Vazirgiannis, M. (2006). A system architecture for context aware service discovery. Electron. Notes Theor. Comput. Sci. 146, 1 (January 2006):101-116.

Fingar, P. (2009). Cloud computing and the promise of on-demand business innovation. Intelligent Enterprise.

Fu B., Brennan R., & O'Sullivan D. (2009). Multilingual Ontology Mapping: Challenges and a Proposed Framework. Proceedings of the 23rd Convention of the Society for the Study of Artifitial Intelligence and Simulation of Behaviour - Adaptive & Emergent Behaviour & Complex Systems, pp 32 - 35.

Fujii, K., & Suda, T. (2009). Semantics-based context-aware dynamic service composition. ACM Trans. Auton. Adapt. Syst., 4(2):1-31.

Genabith, J.v. (2001). Metaphors, Logic and Type Theory. Metaphor and Symbol, 16(1/2):42-57.

Genabith, J.v., & Crouch, R. (1999). Dynamic and Underspecified Semantics for LFG. Semantics and Syntax in Lexical Functional Grammar: The Resource Logic Approach. MIT Press, pp. 209-260.

Graham, Y., & van Genabith, J. (2009). An Open Source Rule Induction Tool for Transfer-Based SMT. The Prague Bulletin of Mathematical Linguistics, Special Issue: Open Source Tools for Machine Translation, 91:37-46.

Kapitsaki, G., Kateros, D., Prezerakos, G., & Venierris, I. (2009). Model-driven development of composite context-aware web applications. Information and Software Technology 51: 1244-1260.

Karastoyanova, D., & Leymann, F. (2009). BPEL'n'Aspects: Adapting service orchestration logic. International Conference on Web Services. pp. 229-229. IEEE Press.

Kharbili, M.E., & Keil, T. (2010). Bringing agility to business process management: Rules deployment in an SOA. Emerging Web Services Technology Volume III. Whitestein Series in Software Agent Technologies and Autonomic Computing. pp. 157-170. Birkhaeuser Verlag.

Leusse, P.D., Dimitrakos, T., & Brossard, D. (2009). A governance model for SOA. IEEE International Conference on Web Services. pp. 1020-1027. IEEE Press.

Lewis, D., Curran, S., Feeney, K., Etzioni, Z., Keeney, J., Way, A., & Schäler, R. (2009). Web service integration for next generation localisation. Workshop on Software Engineering, Testing, and Quality Assurance for Natural Language Processing, pp. 47-55.

Mietzner, R., Unger, T., Titze, R., & Leymann, F. (2009). Combining different multi-tenancy patterns in service-oriented applications. IEEE International Enterprise Distributed Object Computing Conference. pp. 131-140. IEEE Press.

Mietzner, R., Leymann, F., & Papazoglou, M.P. (2008). Defining composite configurable SaaS application packages using SCA, variability descriptors and multi-tenancy patterns. International Conference on Internet and Web Applications and Services. pp. 156-161. IEEE Press.

Metzger, A., Pohl, K., Papazoglou, M., Di Nitto, E., Marconi, A., & Karastoyanova, D. (2012). Research challenges on adaptive software and services in the future internet: towards an S-Cube research roadmap. Workshop on European Software Services and Systems Research-Results and Challenges. pp. 1-7. S-Cube Proceedings.

Pahl, C., & Zhu, Y. (2005). A Semantical Framework for the Orchestration and Choreography of Web Services. International Workshop on Web Languages and Formal Methods WLFM'05. ENTCS 151(2):3-18.

Pahl, C. (2010). Dynamic adaptive service architecture - towards coordinated service composition. European Conference on Software Architecture ECSA'2010. pp. 472-475. Springer-Verlag.

Pahl, C., Giesecke, S., & Hasselbring, W. (2009). Ontology-based Modelling of Architectural Styles. Information and Software Technology, 1(12): 1739-1749.

Pahl, C. (2012). Cloud Service Localisation. European Conference on Service-Oriented and Cloud Computing ESOCC 2012. pp. 138-153. Springer-Verlag.

Phillips, A.P. (2005). Web Services and Internationalization. Whitepaper. Retrieved December, 10, 2013 from http://www.inter-locale.com/whitepaper/multilingual/ml73-ws-20050524.xml

Rosenberg, F., & Dustdar, S. (2005). Business rules integration in BPEL - a service-oriented approach. 7th IEEE Intl Conference on E-Commerce Technology. pp. 476 – 479. IEEE Press.

Wang, M.X., Bandara, K.Y., & Pahl, C. (2010). Process as a Service - Distributed Multi-tenant Policy-based Process Runtime Governance. IEEE International Conference on Services Computing SCC 2010. pp. 578-585. IEEE Press.

Weigand, H., Heuvel, W.-J. v. d., & Hiel, M. (2008). Rule-based service composition and service-oriented business rule management. Interdisciplinary Workshop Regulations Modelling and Deployment. pp. 1-12. Retrieved December, 10, 2013 from http://ceur-ws.org/Vol-342/

W3C (2005). Web Services Internationalization Usage Scenarios. Retrieved December, 10, 2013 from http://www.w3.org/TR/ws-i18n-scenarios