Actas de las
VI Jornadas Científico-Técnicas en Servicios Web y SOA (JSWEB2010)

EDITORES
Juan Pavón
Vicente Pelechano
Santi Ristol
Reservados los derechos para todos los países de lengua española. De conformidad con lo dispuesto en el artículo 270 y siguientes del código penal vigente, podrán ser castigados con penas de multa y privación de libertad quienes reprodujeran o plagien, en todo o en parte, una obra literaria, artística o científica fijada en cualquier tipo de soporte sin la preceptiva autorización. Ninguna parte de esta publicación, incluido el diseño de la cubierta, puede ser reproducida, almacenada o transmitida de ninguna forma, ni por ningún medio, sea éste electrónico, químico, mecánico, el electro-
óptico, grabación, fotocopia o cualquier otro, sin la previa autorización escrita por parte de la editorial.

Diríjase a CEDRO (Centro Español de Derechos Reprográficos), www.cedro.org, si necesita fotocopiación o escanear algún fragmento de esta obra.

COPYRIGHT © 2010 IBERGARCETA PUBLICACIONES, S.L.
info@garceta.es

Actas de las VI Jornadas Científico-Técnicas en Servicios Web y SOA (JSWEB2010)
Derechos reservados ©2010 respecto a la primera edición en español, por LOS AUTORES
Derechos reservados ©2010 respecto a la primera edición en español, por IBERGARCETA PUBLICACIONES, S.L.

1ª Edición, 1ª Impresión
Depósito legal: M-

Maquetación: Los Editores
Coordinación del proyecto: @LIBRÒTEX
Portada: Estudio Dixi
Impresión y encuadernación:
OI: 22/2010
PRINT HOUSE, S.A.

IMPRESO EN ESPAÑA - PRINTED IN SPAIN

Nota sobre enlaces a páginas web ajenas: Este libro puede incluir referencias a sitios web gestionados por terceros y ajenos a IBERGARCETA PUBLICACIONES, S.L., que se incluyen sólo con finalidad informativa. IBERGARCETA PUBLICACIONES, S.L., no asume ningún tipo de responsabilidad por los daños y perjuicios derivados del uso de los datos personales que pueda hacer un tercero encargado del mantenimiento de las páginas web ajenas a IBERGARCETA PUBLICACIONES, S.L., y del funcionamiento, accesibilidad y mantenimiento de los sitios web no gestionados por IBERGARCETA PUBLICACIONES, S.L., directamente. Las referencias se proporcionan en el estado en que se encuentran en el momento de publicación sin garantías expresas o implícitas, sobre la información que se proporcione en ellas.
Model-Driven Aspect-Oriented Quality of Service for Web Services

Guadalupe Ortiz, Behzad Bordbar

### Demostraciones

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Herramienta de Administración Avanzada de la Plataforma de la Interoperabilidad de la Junta de Andalucía (PLATINA)</td>
<td>129</td>
</tr>
<tr>
<td>Manuel Toscano Ferrera, Juan Sebastian Ojeda Pérez</td>
<td></td>
</tr>
<tr>
<td>kAnalytics (Knowledge Analytics): Visualizing Analysis Results Over Linked Data Services.</td>
<td>131</td>
</tr>
<tr>
<td>Guillermo Álvaro, Ricardo Melero, Iván Martínez, José Manuel Gómez</td>
<td></td>
</tr>
<tr>
<td>A demo with SPICES (Semantic Platform for the Interaction and Consumption of Enriched Services)</td>
<td>133</td>
</tr>
<tr>
<td>Guillermo Álvaro, Iván Martínez, Matteo Villa, José Manuel Gómez, Freddy Lecue, Carlos Pedrinaci, Giovanni Di Matteo</td>
<td></td>
</tr>
<tr>
<td>ADA: Agreement Documents Analyser</td>
<td>135</td>
</tr>
<tr>
<td>Carlos Müller, Manuel Resinas, Antonio Ruiz-Cortés</td>
<td></td>
</tr>
<tr>
<td>iServe: a Linked Services Publishing Platform.</td>
<td>139</td>
</tr>
<tr>
<td>Carlos Pedrinaci, Dong Liu, Maria Maleshkova, David Lambert, Jacek Kopecký, John Domingue</td>
<td></td>
</tr>
</tbody>
</table>
ADA: Agreement Documents Analyser

Carlos Müller  Manuel Resinas  Antonio Ruiz-Cortés
ETS. Ingeniería Informática  ETS. Ingeniería Informática  ETS. Ingeniería Informática
Univ. de Sevilla  Univ. de Sevilla  Univ. de Sevilla
cmiller@us.es  resinas@us.es  aruiz@us.es

Abstract

Service Level Agreements (SLAs) are one of the cornerstones of service-oriented architectures. SLAs consist of a set of terms including any relevant information to regulate the trading of services. Many recommendations to specify SLAs have been proposed, among them WS-Agreement recommendation [1] defines an XML-based language and a template-based protocol for creating SLAs by means of agreement offers based on published templates. These agreement documents should be redacted carefully because a mistake to specify their terms could get a conflictive SLA, making impossible to reach an agreement between parties. However, depending on the complexity of the agreement, finding manually these conflicts may become a challenging task. Therefore, in this work we propose the Agreement Documents Analyser (ADA) which is an easy-to-extend framework which supports automated analysis operations to check the consistency within terms of an unique agreement document, and/or the compliance between an agreement offer and its related template specified with WS-Agreement. Furthermore, ADA reports explanations for inconsistency or non-compliance scenarios. In addition, ADA has many extension points by its decoupled structure allowing a dynamic multi-paradigm analysis and it supports WSAgreement, our own succinct plain-text language for WS-

1 Introduction

Service Level Agreements (SLAs) consist of a set of terms that include information about functional features, non-functional guarantees and any other terms with relevant information to the agreement. There are many proposals to specify SLAs. Among them WS-Agreement specification [1] defines an XML-based language and a protocol for advertising the capabilities and preferences of services providers in templates, and creating agreements based on them. In a typical interaction process using this protocol an agreement initiator would create an agreement offer since the template published by the responder party. In this SLA creation process both agreement documents should be redacted carefully because a mistake to specify their terms could make impossible to reach the agreement between parties. Then, the resulting SLA should be specified in a correct way, avoiding conflicting terms (i.e. they state opposite assertions), because they cannot be fulfilled at the same time. We consider two kind of conflicts: the inconsistency which is a conflict within terms of an unique agreement document and the non-compliance which is a conflict between terms of an agreement offer and its related template. However, depending on the complexity of the agreement, finding manually these conflicts may become a challenging task.

For instance, in a translation service scenario, an SLA could be agreed including non-functional properties such as the cost, the size
of the input text to translate (size), in number of words, and the demanded time (time), in days. Assuming that: size ranges between [100..10000], time ranges between ∈ [1..5] and cost = size/time. If the SLA includes a term obligating to pay a Cost of 10, at first sight the SLA is consistent. However the lowest valid value for cost is 100/5 = 20, so this term cannot be fulfilled. Therefore, a conflict checker that automatically checks the SLA for conflict between its terms would be very appealing from a practical point of view. Moreover, it is of interest not only to obtain an automated way of checking if an agreement document has conflicting terms or not, but also to return an explanation for the document conflict. Thus, in the previous scenario we would obtain as explanation the following conflicting terms: [(Cost = 20), (size ∈ [100..10000]), (time ∈ [1..5]), (cost = size/time)].

As result of previous works [2, 3, 4, 6], we present here the Agreement Documents Analyser (ADA), which is an easy-to-extend framework supporting mentioned automated conflict checking and explaining for WS-Agreement documents. ADA has many extension points by its decoupled structure which allows the use of a dynamic multi-paradigm analysis. Furthermore, to avoid the verbosity of XML-based documents of WS-Agreement, we have developed our own succinct plaintext language for WS-Agreement documents, called WSAg4People. The reader can consult http://www.isi.u.cs.es/ADA for a much more thorough description of ADA. Documentation section of such web site includes a short demonstration video and try it online! section allows a try of ADA operations from our developed front end.

2 ADA Extension Points

ADA has a decouple subsystem-based architecture controlled by a central core, the ADA Core, as Figure 1 depicts. ADA Core controls these subsystems which provide or consume functionality. Currently, ADA has four operations: consistency, compliance, consistency explaining and completeness explaining.

one CSP (Constraint Satisfaction Problem)-based analyser: Choco solver; a default selection criteria, because the optimal selector is not still necessary and it is under development; and two agreement documents formats, WS-Agreement XML and WS-Agreement4People. However, by its architecture, ADA framework can be easily extended in many ways:

• Operations. New operations can be added, as programmatic interfaces.

• Analysers. ADA is ready to be extended with different analysers of diverse paradigms, for instance: CSP-based analysers, DL (Description Logic)-based analysers, etc.

• Selection criteria. When several analysers are available, a selection criteria is needed to choose one of them.

• Agreement definition formats. ADA provides mechanisms to read/write models from/to different file formats, and to transform many formats into others.

3 Using ADA

ADA is developed as a java application and it can be used from other tools as a traditional Application Programming Interface (API). However, we pursue the goal of putting ADA available for more uses. Thus, ADA is deployed into an OSGi (www.osgi.org) container as a set of java components (OSGi bundles). OSGi is a specification of a Java Framework, with multiple components, that provides a SOA architecture. It makes deployment, communication, discovery, and, in general, integration among bundles, easier. Therefore, the ADA integration with other tools or frameworks that use it, e.g., Eclipse, is straightforward. Moreover, there are many OSGi plugins that help us in multiple tasks such as: transform an OSGi service to a WSDL definition. Then we have used such plugin to get our ADA-WS(Web Service) in order to put ADA services available over the
network. Other advantage of being OSGi compliant is to add functionality without restarting systems. Thus, the integration of a new bundle in ADA is straightforward, it is only necessary to install and start the bundle.

We have planned a future use of ADA as a service engine of an Enterprise Service Bus (ESB). This would allow ADA to be used in conjunction with the proposed service engine of the Framework for Automated Service Trading (FAST) [5]. This advanced use would allow the automated analysis of SLAs in the whole service trading process.

A RIA front-end, developed with JavaFX and that uses the ADA-WS api, is available on www.isa.us.es/ADA. It allows to load, save and edit WS-Agreement documents, on XML or WSAg4People formats, and to analyse these documents by means of ADA Framework. This front-end has an usability-based design including colored syntax, skeletons to create a new document, or transformations between XML and WSAg4People formats. The reader can consult also at this web a much more thorough description of supported agreement documents in Documentation->ADA Features section; for a short video demonstration of ADA in Documentation->Demonstration Video section; and for a try of ADA operations used from the front-end in try it online! section.

4 Conclusions

In this work, we propose the use of ADA, an easy-to-extend and easy-to-use framework to edit and analyse agreement documents. ADA satisfies the necessity of a tool that automates the error-prone and tedious task of analyse agreement documents. ADA has many extensions points and we have developed a RIA front-end as proof-of-concept.

References


