



NoTube

*Networks and ontologies for the transformation and unification of broadcasting
and the Internet*

FP7 – 231761

D5.2 Semantic Annotations of TV Services

Coordinator: S. Dietze (OU)

With contributions from:

A. Conconi, F. Cattaneo (TXT), N. Benn (OU), H. Qing Yu (OU),
Pavel Mihaylov (OT)

Quality Assessor: Dan Brickley

Quality Controller: Lora Aroyo

Document Identifier:	NoTube/2010/D5.1b
Version:	1.0
Date:	10/03/2011
State:	Final
Distribution:	PU

EXECUTIVE SUMMARY

This deliverable describes the annotation and modelling approach followed to describe and broker NoTube services. Our proposed annotation approach and infrastructure is based on the requirements described in D5.1 and derived from the use cases (WP7) and the architectural requirements posed by WP6. In addition, this deliverable presents an overview of the currently used services integration infrastructure which is fundamentally based on state of the art Linked Data (LD) principles.

DOCUMENT INFORMATION

IST Project Number	FP7 - 231761	Acronym	NoTube
Full Title	Networks and ontologies for the transformation and unification of broadcasting and the Internet		
Project URL	http://www.notube.eu/		
Document URL			
EU Project Officer	Leonhard MAQUA		

Deliverable	Number	5.2	Title	Semantic TV Service Annotations
Work Package	Number	5	Title	Semantic TV Resource Broker

Date of Delivery	Contractual	M 23	Actual	
Status	version 1.0		final <input checked="" type="checkbox"/>	
Nature	prototype <input type="checkbox"/> report <input checked="" type="checkbox"/> dissemination <input type="checkbox"/>			
Dissemination level	public <input checked="" type="checkbox"/> consortium <input type="checkbox"/>			

Authors (Partner)	TXT, OU, OT			
Responsible Author	Name	Stefan Dietze	E-mail	s.dietze@open.ac.uk
	Partner	TXT, OU, OT	Phone	+44 (0)1908-858217

Abstract (for dissemination)	This deliverable describes the annotation and modelling approach followed to describe and broker NoTube services. Our proposed annotation approach and infrastructure is based on the requirements described in D5.1 and derived from the use cases (WP7) and the architectural requirements posed by WP6. In addition, this deliverable presents an overview of the currently used services integration infrastructure which is fundamentally based on state of the art Linked Data (LD) principles.
Keywords	Linked Services, Service Broker, Semantic Web, SWS, Services, Service Annotations

Version Log			
Issue Date	Rev. No.	Author	Change
12/09/10	0.1	S. Dietze	Preliminary ToC
25/10/10	0.2	N. Benn	Draft content for Sections 4 and 6
14/12/10	0.3	S. Dietze, F. Cattaneo	Evaluation plan
27/12/10	0.4	S. Dietze	Overall revision and restructuring
05/01/11	0.5	S. Dietze	Preliminary evaluation results
22/01/11	0.6	S. Dietze	Edit of eval plan and results
22/02/11	0.7	S. Dietze	Overall refinement
09/03/11	0.8	P. Mihaylov	Section 4.3 (IdRF) added
10/03/11	1.0	S. Dietze	Consolidation

PROJECT CONSORTIUM INFORMATION












Participants		Contact
Vrije Universiteit Amsterdam		Guus Schreiber Phone: +31 20 598 7739/7718 Email: schreiber@cs.vu.nl
British Broadcasting Corporation		Libby Miller Phone: +44 787 65 65 561 Email: Libby.Miller@bbc.co.uk
Pro-netics S.p.A.		Marco Bruni Phone: +39 0645472503 Email: marco.bruni@pro-netics.com
Engin Medya Hizmetleri A.S.		Ron van der Heiden Phone: +31 6 2003 2006 Email: ron@engin.tv
Institut fuer Rundfunktechnik GmbH		Christoph Dosch Phone: +49 89 32399 349 Email: dosch@irt.de
Ontotext AD		Atanas Kiryakov Phone: +35 928 091 565 Email: naso@sirma.bg
Open University		Stefan Dietze Phone: +44 1908 858 217 Email: s.dietze@open.ac.uk
RAI Radiotelevisione Italiana SPA		Alberto Morello Phone: +39 011 810 31 07 Email: a.morello@rai.it
Semantic Technology Institute International		Lyndon Nixon Phone: +43 1 23 64 002 Email: lyndon.nixon@sti2.org
Stoneroods B.V.		Annelies Kaptein Phone: +31 35 628 47 22 Email: annelies.kaptein@stoneroods
Thomson Video Networks		Raoul Monnier Phone: +33 2 99 27 30 57 Email: raoul.monnier@thomson.nett
TXT e-Solutions		Sergio Gusmeroli Phone: +39 02 2577 1310 Email: sergio.gusmeroliqtxtgroup.com
KT Corporation		Myoung-Wan Koo Phone: +82 2 526 6347 Email: mskim@kt.co.kr

TABLE OF CONTENTS

LIST OF FIGURES	8
LIST OF TABLES	9
LIST OF ACRONYMS.....	10
1 INTRODUCTION	11
2 SEMANTIC WEB SERVICES (SWS) OVERVIEW.....	12
3 TWO-FOLD SERVICES ANNOTATION IN NOTUBE: CHALLENGES AND OVERVIEW	13
4 SEMANTIC WEB SERVICES ANNOTATIONS: INFRASTRUCTURAL SUPPORT	14
4.1 SMARTLINK.....	14
<i>Lightweight services annotation: the Linked Services approach</i>	<i>14</i>
<i>Service annotation and integration via SmartLink.....</i>	<i>15</i>
4.2 IRS-III SWS EXECUTION ENVIRONMENT.....	16
4.2.1 <i>Extending REST-based grounding with support for JSON parsing</i>	<i>17</i>
4.2.2 <i>Extending IRS-III grounding with support for XMPP.....</i>	<i>17</i>
4.3 SERVICE SIMILARITY MATCHING	17
4.3.1 <i>Introduction to IdRF</i>	<i>17</i>
4.3.2 <i>IdRF architecture overview</i>	<i>18</i>
4.3.3 <i>Using IdRF with SmartLink</i>	<i>18</i>
5 NOTUBE SERVICE ANNOTATIONS.....	20
5.1 IMPLEMENTED GOALS AND SERVICE ORCHESTRATIONS	20
5.1.1 <i>Video metadata retrieval.....</i>	<i>20</i>
5.1.2 <i>EPG retrieval functionality.....</i>	<i>21</i>
5.1.3 <i>Enrichment functionality.....</i>	<i>23</i>
5.1.4 <i>Beancounter functionality.....</i>	<i>24</i>
5.2 LIGHTWEIGHT RDF SERVICE & GOAL ANNOTATIONS	25
6 INTEGRATION INTO NOTUBE USE CASE SCENARIOS.....	27
6.1 WP7A SCENARIO – PERSONALISED SEMANTIC NEWS (PSN)	27
6.1.1 <i>Retrieving Enriched EPG data</i>	<i>27</i>
6.1.2 <i>NIC transformation and enrichment</i>	<i>27</i>
6.1.3 <i>News recommendation.....</i>	<i>27</i>
6.1.4 <i>Video reframing</i>	<i>28</i>
6.2 WP7B SCENARIO – PERSONALISED TV GUIDE WITH ADAPTIVE ADVERTISING	29
6.2.1 <i>Retrieving enriched EPG data</i>	<i>29</i>
6.2.2 <i>Profile-based TV programme recommendation.....</i>	<i>29</i>
6.2.3 <i>Ad insertion.....</i>	<i>30</i>
6.3 WP7C SCENARIO – INTERNET TV IN THE SOCIAL WEB	30
6.3.1 <i>Buttons protocol brokering using XMPP.....</i>	<i>30</i>
7 EVALUATION PLAN	32
7.1 EVALUATION CRITERIA	32
7.1.1 <i>Functional evaluation criteria</i>	<i>32</i>
7.1.1.2 <i>Additional functional evaluation opportunities.....</i>	<i>33</i>
7.1.2 <i>Non-functional evaluation criteria.....</i>	<i>34</i>
7.2 EVALUATION PROCEDURE.....	35
7.2.1 <i>Formal setting: parameterised EPG retrieval</i>	<i>35</i>
7.2.2 <i>Evaluation stakeholders.....</i>	<i>36</i>
7.2.3 <i>Evaluation activities.....</i>	<i>37</i>
7.2.3.1 <i>Functional Evaluation Criteria</i>	<i>37</i>
7.2.3.2 <i>Non-functional evaluation criteria</i>	<i>37</i>
8 CONCLUSION	39



9 REFERENCES40

List of Figures

Fig. 1. iServe conceptual model for services – The Minimal Service Model and WSMO-Lite.	15
Fig. 2. SmartLink – conceptual architecture.....	15
Fig. 3. IdRF architecture.....	18
Fig. 4. RDF excerpt of LUPEDIA service description based on MSM.	26
Fig. 5. Workflow for transforming Prestospace to TVA-NIC.	27
Fig. 6. The steps of the workflow for building a personalised news playlist using enriched TVA-NICs.....	28
Fig. 7. Mapping Buttons protocol commands to manufacturer-specific functions via the Broker.....	31

List of Tables

Table 1 - Summarised, enumerated list of requirements for the Semantic Service Integration.	32
Table 2 - Summarised, enumerated list of evaluation criteria (EC) for the Semantic Service Integration.	33
Table 3. Evaluation target groups.	36

List of Acronyms

Acronym	Description
AMI	Ambient Intelligence
EPG	Electronic Program Guide
hRESTs	HTML for RESTful Services
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	RDF Schema
RIF	Rule Interchange Format
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
STB	Set-Top Box
SWS	Semantic Web Services
UDDI	Universal Description, Discovery and Integration
WADL	Web Application Description Language
WSDL	Web Services Description Language
WSMO	Web Service Modelling Ontology

1 Introduction

This deliverable describes the annotation and modelling approach followed to describe and broker NoTube services. Our proposed annotation approach and infrastructure is based on the requirements described in D5.1 and derived from the use cases (WP7) and the architectural requirements posed by WP6. In addition, this deliverable presents an overview of the currently used services integration infrastructure which is fundamentally based on state of the art Linked Data (LD) principles.

The following section proposes a generic introduction to the state of the art in Semantic Web Services (SWS) while Section 3 introduces our two-fold approach which combines two different SWS environments to comply with all NoTube specific services management requirements. Section 4 introduces our SWS infrastructure which is used to produce, store and process services annotations which are described in Section 5. The integration of these environments into the NoTube use case prototypes is discussed in Section 6 while our evaluation plans are discussed in Section 7.

2 Semantic Web Services (SWS) overview

Semantic Web Services (SWS) are ontological descriptions of Web services in terms of their capabilities, interfaces and non-functional properties. Tasks such as Web service discovery, composition, and invocation can be automated to a great extent by applying semantic technologies. For example, semantics allow programs to access services through a machine-processable description of the service's capabilities rather than as a direct service endpoint. The use of semantics thus forms a scalable access layer over Web service data and processes. In deliverable D6.1, a review of Semantic Web and Web Service technologies as well as standards are presented, which will be commonly referred to in this deliverable. In this Section, we present a number of SWS technologies, and in particular, the IRS-III SWS execution environment, which implements the role of the Broker.

Semantic Web Services technologies enable the automatic discovery, selection and composition of distributed services for a particularly expressed user request. Note that the term service here refers to software functionalities which are exposed to and accessible through the Web (i.e. based on HTTP). In that, a Web service might utilise standard Web service technology such as SOAP [30], UDDI [31] and WSDL [32] but also more light-weight approaches such as REST or XML-RPC. Semantic Web Services are being deployed to facilitate interoperability and to increase the degree of automation in a wide range of applications from different domains, such as eLearning [5] or business process management [14].

Current results of SWS research are available in terms of reference ontologies, such as *OWL-S* [10], *WSMO*¹ [23], and *SAWSDL*² as well as comprehensive frameworks such as those produced by the DIP project³). These reference ontologies and frameworks are intended to enable fully automated service matchmaking based on comprehensive semantic specifications of service capabilities.

Recent derivations of WSMO, enable representation of rather light-weight service descriptions based on RDF and the hREST microformat:

- *WSMO-Lite*⁴: Lightweight Descriptions of Services on the Web – a lightweight set of semantic service descriptions in RDFS that can be used for annotations of various WSDL elements using the SAWSDL annotation mechanism. WSMO-Lite exploits the standard languages of W3C including RDF and RDFS as well as various extensions of those languages such as OWL, WSML and RIF for semantic service descriptions.
- *MicroWSMO*⁵: Semantic Annotations for RESTful Services – a semantic annotation mechanism for RESTful Web services based on the hRESTs microformat.
- *hRESTs*⁶ (HTML for RESTful Services) - microformat for machine-understandable descriptions of Web APIs, backed by a simple service model. The hRESTs microformat describes main aspects of services, such as operations, inputs and outputs. Also available are two extensions of hRESTs: SA-REST, which captures the facets of public APIs important for mashup developers, and MicroWSMO, which provides support for semantic automation.

While the above mentioned lightweight approaches are less costly to apply, they envisage a much lower degree of automation and do not facilitate comprehensive matchmaking scenarios as foreseen by more complex frameworks such as WSMO.

¹ <http://cms-wg.sti2.org/TR/d1/v1.0/>

² <http://www.w3.org/2002/ws/sawSDL/>

³ DIP Project: <http://dip.semanticweb.org>

⁴ http://cms-wg.sti2.org/TR/d11/v0.2/20090310/d11v02_20090310.pdf

⁵ http://cms-wg.sti2.org/TR/d12/v0.1/20090310/d12v01_20090310.pdf

⁶ <http://knoesis.wright.edu/research/srl/projects/hRESTs>

3 Two-Fold services annotation in NoTube: challenges and overview

The past decade has seen a wide range of research efforts in the area of Semantic Web Services (SWS), mainly aiming at the automation of Web service-related tasks such as discovery, orchestration or mediation via broker-based approaches. Building on formal service semantics, several conceptual models, such as OWL-S [20] and WSMO [13] and also standards such as SAWSDL [25] have been proposed which aim at formalizing semantic service descriptions usually covering aspects such as service capabilities, interfaces and non-functional properties. Besides, a considerable research community evolved around these SWS frameworks, providing, for instance, related annotation and execution tools.

While semantics are used to mark up a wide variety of data-centric resources on the Web, that does not apply to online services in significant numbers. The reasons for this are two-fold. Firstly, SWS research has for the most part targeted WSDL or SOAP-based Web services, which are not prevalent on the Web. Secondly, due to the inherent complexity required to fully capture computational functionality, creating SWS descriptions has represented an important knowledge acquisition bottleneck and has required the use of rich knowledge representation languages and complex reasoners. There exists an inherent conflict between the need to capture comprehensive and meaningful service semantics – to allow reasoning-based automation of any sort – and the requirement to keep the costs for providing services descriptions low in order to simplify the modelling process and to ensure that efficient and scalable solutions can be implemented [23]. Hence, despite considerable amount of research dedicated to the SWS vision and the existence of a range of SWS-related projects, tools and specifications, so far there has been little take up of SWS technology within non-academic environments.

The prevalent lack of impact of SWS technology is particularly concerning since Web services – nowadays including a range of often more light-weight technologies beyond the WSDL/SOAP approach, such as RESTful services, HTTP GET-style requests or XML-feeds – are in widespread use throughout the Web where applications use distributed requests to combine and mash-up data from a variety of open data sources. Hence, the challenges SWS attempted to tackle are of even more crucial importance for today’s highly distributed Web applications. These issues have led to the emergence of more simplified SWS approaches to which we shall refer here as “lightweight”, such as WSMO-Lite [26] or the Micro-WSMO/hRESTs [15] approach which replace “heavyweight” service semantics with less comprehensive and less costly to produce service models that are represented in RDF(S), and hence, comply with the infrastructure of the growing *Semantic Web*[2]. Analogous to the *Linked (Open) Data (LOD)* term [3], this approach was recently dubbed as the *Linked Service* approach [18]. Due to the fact that such service annotations are much easier to produce and can be populated with references to widely established LOD vocabularies, they address a much wider audience and allow even non-SWS experts to describe and annotate services. However, while those models are easier to produce [23], they merely aim at enabling structured, semantics-enabled search by humans or automated service clustering. More expressive solutions are required to achieve greater levels of automation, for instance, dealing with matching service requests with extensive capability representations of available services, or with handling of data-level mismatches when executing a set of services in an orchestrated manner.

Therefore, here, we aim to combine the strengths of both distinctive SWS approaches – lightweight Linked Services and more heavyweight broker-based SWS automation – into a coherent SWS framework. By integrating collaborative and user-driven Web-scale service annotations with comprehensive SWS specifications, we benefit from both low cost for providing annotations and a high level of automation. This also has the benefit of enabling a range of matchmaking scenarios (from user-driven keyword matching to automated capability matchmaking).

4 Semantic Web Services annotations: infrastructural support

In the context of our two-fold approach, this section describes the latest infrastructure development in WP5. The infrastructure described here strives to support the two major fundamental requirements within NoTube, namely (a) supporting automated services brokerage and (b) aiding developers in documenting and tracking services (see previous deliverable D5.1 for further details). In this context, this infrastructure development consists, on the one hand, of tools to support our lightweight, linked services approach, (Section 4.1) and on the other hand, extending our existing brokering environment that displays a more comprehensive, heavyweight approach to semantic web services (Section 4.2).

4.1 SmartLink

Lightweight services annotation: the Linked Services approach

In order to support annotation of a variety of services, the OU has developed iServe⁷ a novel and open platform for publishing semantic annotations of services based on a direct application of linked data principles (<http://linkeddata.org/>). iServe supports publishing service annotations as linked data—Linked Services—expressed in terms of a simple conceptual model that is suitable for both human and machine consumption and abstracts from existing heterogeneity around service kinds and annotation formalisms. In particular iServe provides:

- Import of service annotations in a range of formalisms (e.g., SAWSDL, WSMO-Lite, MicroWSMO, OWL-S) covering both WSDL services and Web APIs;
- Means for publishing semantic annotations of services which are automatically assigned a resolvable HTTP URI;
- Support for content negotiation so that service annotations can be returned in plain HTML or in RDF for direct machine consumption;
- SPARQL endpoint allowing querying over the services annotations;
- REST API to allow remote applications to consume and provide annotations.
- Support for linking service annotations to existing vocabularies on the Web.

In order to cater for interoperability, iServe uses what can be considered the maximum common denominator between existing SWS formalisms which we refer to as the Minimal Service Model (MSM). The MSM is a simple RDF(S) ontology able to capture (part of) the semantics of both Web services and Web APIs in a common model. MSM is extensible to benefit from the added expressivity of other formalisms. The MSM, first introduced together with WSMO-Lite and hRESTS [25], is thus a simple RDF(S) ontology able to capture (part of) the semantics of both Web services and Web APIs in a common model. MSM is extensible to benefit from the added expressivity of other formalisms. The MSM, denoted with the 'msm' namespace in Fig. 1, defines *Services* as having a number of *Operations* each of which have an *Input*, *Output MessageContent*, and *Faults*. In turn, a *MessageContent* may be composed of *MessageParts* which may be *mandatory* or *optional*. iServe additionally uses the SAWSDL, WSMO-Lite and hRESTS vocabularies. The SAWSDL vocabulary captures in RDF the three main kinds of annotations over WSDL and XML Schema, including *modelReference*, *liftingSchemaMapping* and *loweringSchemaMapping* that SAWSDL supports. WSMO-Lite builds upon SAWSDL by extending it with a model specifying the semantics of the particular service annotations. It provides a simple RDFS ontology together with a methodology for expressing functional and non-functional semantics, and an information model for WSDL services based on SAWSDL's *modelReference* hooks. The hRESTS vocabulary extends the MSM with specific attributes for operations so as to allow modelling additional details necessary for Web APIs.

⁷ <http://iserve.kmi.open.ac.uk>

SmartLink builds on existing technologies and standards to enable wide reach of its annotations. Users can annotate arbitrary services - whether REST-ful or WSDL/SOAP-based - via a simple Web form. Annotations are stored in RDF following established service schemas, namely the Minimal Service Model (MSM, http://iserve.kmi.open.ac.uk/wiki/index.php/Simple_vocabulary) which follow a light-weight approach to Semantic Web Services. Storage of annotations is spread across two public RDF-stores: iServe (<http://iserve.kmi.open.ac.uk>) handles all functional properties defined in the MSM schema while an additional and SmartLink-specific SESAME repository hosts further non-functional service properties. These non-functional properties are, for instance, *contact person*, *developer name*, *Quality of Service (QoS)*, *development status*, *service license*, and *WSMO goal reference*. The latter property directly contributes to facilitate our vision of allowing MSM models to refer to existing WSMO goals which utilize the same service entity; that is, it facilitates our model referencing vision between MSM and WSMO models. In addition, by allowing developers to directly annotate existing REST-ful services and APIs, SmartLink directly provides another contribution to enable our service model integration vision based on allowing the annotation of WSMO goal requests – which in fact are REST-ful services themselves – as MSM service instances.

SmartLink currently provides mechanisms that enable the export of particular (MSM) service instances as RDF or human-readable HTML. In order to facilitate service model transformation and augmentation between MSM and WSMO, current research deals with the establishment of an export mechanism of MSM service models as WSMO instances. While current implementation work is concerned with adding corresponding export facilities to SmartLink, model transformation is just enabled on a manual basis at the moment.

In summary, the main SmartLink features include:

- A simple Web annotation form
- An RDF schema defined by OU and VU based on existing service model standards
- The ability to define references from a service annotation to any existing Linked Data vocabulary (either NoTube or external vocabularies)
- Storage in iServe and an additional OWLIM repository
- Export of individual service descriptions into RDF and human-readable HTML
- Service browsing and search facilities
- Classification of services based on the NoTube services taxonomy (ongoing work with TXT, WP6) and other established, general-purpose service classification schemes
- A SPARQL endpoint
- OpenID authentication

4.2 IRS-III SWS execution environment

As introduced in deliverable D5.1a/b, IRS-III⁹ [4] is a SWS execution environment which acts as a service broker – mediating between the goals of a client and relevant services that are deployed on the Web – striving for a high level of service automation. IRS-III adopts the WSMO conceptual model of services. The ultimate aim of the WSMO model of Web services is to be able to provide a well-defined semantics, which can then be interpreted by a reasoner to enable automatic discovery, selection, composition, mediation, execution, and monitoring of services [10]. As opposed to MSM, IRS-III directly covers execution-related aspects.

The WSMO conceptual model of services consists of the following core elements: *goal*, *mediator*, and *Web service*. These are described in a formal representation language, for instance, OCML [21] in the case of IRS-III. The functionality offered by a Web Service is captured by its *capability* description, which defines necessary *pre-* and *postconditions* as well as underlying *assumptions* and *effects* of the service. These are usually formalized as logical expressions. The means to interact with the Web service is captured by its *interface* definition.

⁹ <http://technologies.kmi.open.ac.uk/irs> - IRS: Internet Reasoning Service

Given that IRS-III directly aims at automating service execution related aspects, the interface covers *choreography* and *orchestration* descriptions. Choreography addresses the communication between the IRS-III broker and a Web service, and is described as so-called *grounding*. The IRS-III grounding mechanism supports REST-based, SOAP-based, and XML-RPC based services [15]. Grounding involves two processes referred to as *lifting* and *lowering*. Lowering involves transforming input parameters at the semantic level to data input to the service at the syntactic level. Lifting involves the opposite transformation, i.e. transforming the data output from the service at the syntactic level into an ontological object at the semantic level.

Orchestration addresses the problem of how to model functionality that is composed of several Web services. At the semantic level the orchestration is represented by a workflow model expressed in OCML, that describes the flow of control between the Web services. The IRS-III orchestration model supports the main control-flow primitives of sequence, selection, and repetition.

At runtime, IRS-III automatically discovers and invokes Web services suitable for a given client request, formulated as a goal instance, by selecting suitable services and executing these whilst adhering to any data, control flow and Web service invocation constraints. In principle, selection is based on comparing the capability descriptions of the request with the ones of the relevant SWS. Such matchmaking is currently supported, for instance, via (a) comparison and evaluation of logical expressions used in the capability descriptions, or (b) a hybrid approach [9] which combines similarity-computation via vector representations of SWS instances with (a). The IRS-III functionalities are exposed through a Java API¹⁰, and an HTTP-based REST API, which applications use to interact with IRS-III.

4.2.1 Extending REST-based grounding with support for JSON parsing

Following the principles set out in [16], we have extended the REST-based grounding component in the IRS-III so that it can handle services that use the JSON format. As explained in [16] and [17], current semantic web services frameworks, and implementations of these frameworks, assume a homogeneous environment of SOAP services and XML serialisations, whereas many services actually deployed on the Web use REST-ful interfaces and non-XML serialisations like JSON. JSON is a simple data format derived from JavaScript, and is increasingly popular as a lightweight alternative to XML, particularly in RESTful services. Our lift and lower mechanism within IRS-III could directly manipulate the string representations of JSON, but this becomes cumbersome, prone to error, and fails to model in any meaningful way the transformations. Instead, we introduce a simple ontologisation of JSON. This ontologisation captures the two main structures in JSON: objects and arrays, where an object is an unordered set of key/value pairs, and an array is a sequence of values.

4.2.2 Extending IRS-III grounding with support for XMPP

Finally, we have recently been working on getting the Broker to 'talk' XMPP ---not as easy as it sounds since, as a Web services broker, it has always been designed with HTTP in mind. The invocation model of the Broker has been based on the one-shot, request and response interaction of HTTP. Our new research is to explore how to adapt this model for other modes of interaction, like publish-subscribe, which are possible with XMPP.

4.3 Service similarity matching

SmartLink and iServe provide the means for keeping track of the growing number of services developed for the project. However, keeping track of what the services can do is a different matter. For example, there is no easy way to find out if two services are similar or compatible in some way. This is where Ontotext's Identity Resolution Framework (IdRF) will help. It can be used to compare service descriptions to other service descriptions and find potentially similar services.

4.3.1 Introduction to IdRF

IdRF implements a general solution to identifying (partly) identical objects. It can fit in different applications and work with respect to their particular domain. The novel idea of this solution is the

¹⁰ <http://technologies.kmi.open.ac.uk/irs/irs3docs/api/index.html>

use of a rich semantic knowledge representation that allows for flexible and unified interpretation during the resolution process. The use of an ontology extends the possibilities of the identity criteria to operate with already discovered entities or the context of their appearance. Another important feature of the framework is that identity criteria are customisable. The importance and respectively the weights of these criteria can also be set for a particular task, meeting the specificity of the objects, the context they appear in, and the system's requirements. Thus, the support for customisable criteria and tunable weights helps IdRF to resolve identity for a wide variety of object types and using information integrated from different sources.

4.3.2 IdRF architecture overview

Figure 3 provides an overview of the IdRF architecture. The IdRF architecture consists of three main data processing components: (i) *pre-filtering*, (ii) *evidence collection* and (iii) *decision maker*.

The process starts when a new entity is passed to the system. Then the pre-filtering component will retrieve all entity candidates from the target dataset and for each of them build a new-coming/candidate pair. All the pairs will then be passed to the evidence collection module, which will calculate the similarity between the two objects in the pair based on pre-configured identity criteria. The results will be finally passed to the decision maker component which fuses the data and activates the entity storing procedure. As the framework's native data model is based on ontologies, all the input data should be provided as entity descriptions.

The backbone of the framework is the Semantic Description Compatibility Engine (SDCE), which is responsible for the mediation between the data processing component and the repository. It translates the pre-filtering restrictions into semantic queries and similarity rules into executable methods. SDCE plays the most significant role in the framework implementation and contains domain specific rules for identification.

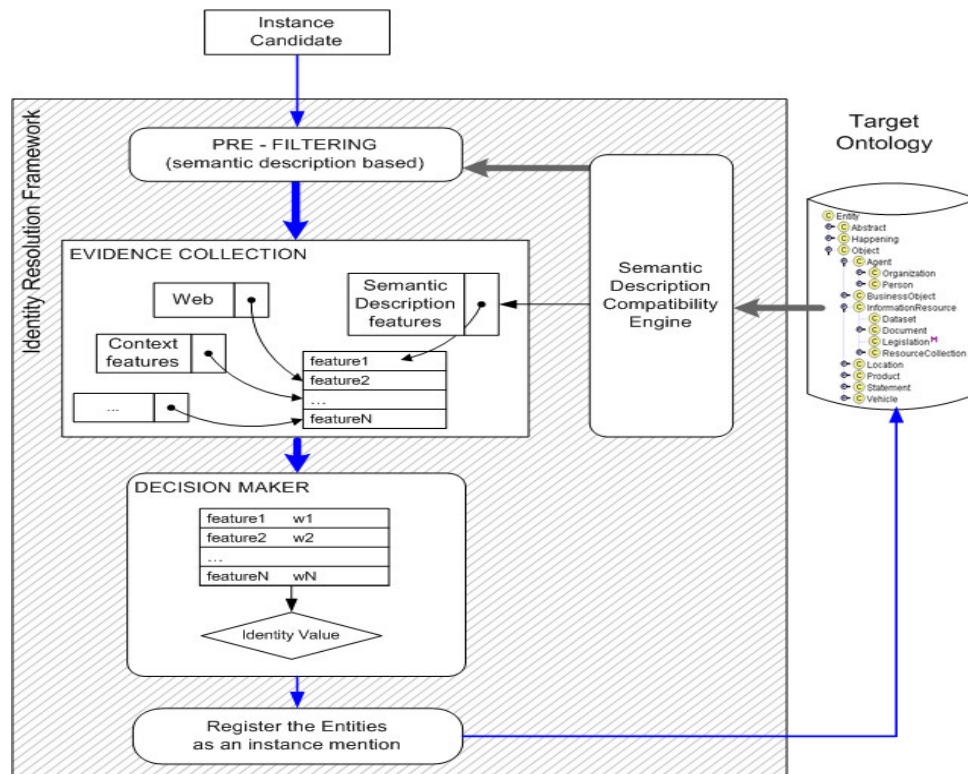


Fig. 3. IdRF architecture.

4.3.3 Using IdRF with SmartLink

Since we are interested only in measuring similarity between two service descriptions and not merging existing service descriptions, we only need to go through the pre-filtering and the evidence

collection stage. This will result in computing a *similarity score* for the input entity (service description) and each of the other entities (other service descriptions) we compare to. As SmartLink is based on iServe, which in turn uses MSM, the input data is already an ontology and as such can be passed directly to the framework. Ontotext has provided the framework, as well as documented example code for comparing service descriptions. The work needs to continue in the direction of developing strategies for service descriptions comparison and using IdRF to perform the comparison.

5 NoTube service annotations

This section provides an overview of the different semantic annotations currently provided as part of NoTube WP5.

5.1 *Implemented goals and service orchestrations*

The implemented goals can be placed into five 'functionality' categories, which represent the main classes of services being developed within NoTube: Video metadata retrieval functionality, EPG retrieval functionality, Enrichment functionality, Beancounter functionality, and Recommender functionality.

5.1.1 Video metadata retrieval

GET-VIDEO-METADATA-GOAL: One key benefit of invoking goals rather than directly executing services is that the Broker can select from among a set of services the most appropriate service for a particular context (e.g. a particular user language or environment). For example, the following shows an 'achieve-goal' request where the goal selects from among 5 metadata search services, each querying and filtering metadata from a different repository (i.e. BBC programmes RDF store @ TALIS, <http://www.open-video.org>, BBC Backstage news feeds, Open University's YouTube channel, YouTube mobile suitable content).

Consider the following achieve-goal request which is associated with the 5 metadata query services listed above:

[http://notube.open.ac.uk:8080/cs-invocation/achieve-cs-goal?ontology=mmgo-goals&goal=get-video-metadata-Goal&has-method=getvideosbykeywords&has-keywords=physics&assumption=\(%22education-entertainment-purpose%22\)](http://notube.open.ac.uk:8080/cs-invocation/achieve-cs-goal?ontology=mmgo-goals&goal=get-video-metadata-Goal&has-method=getvideosbykeywords&has-keywords=physics&assumption=(%22education-entertainment-purpose%22))

The Broker selects and then invokes the most suitable one depending on the assumption parameter. The assumption parameter is a string that refers to instance-IDs in an ontology within the Broker. For this achieve-goal, the assumption parameter defines the nature of the requested content (i.e., whether the content is for an educational or entertainment "purpose"). Note that the Broker accepts a range of assumption parameters defining for instance the user "language" or the technical "environment" (e.g. resolution and bandwidth), all of which define the users current context. Consider the following achieve-goal request which is associated with the same 5 metadata query services listed above:

<http://notube.open.ac.uk:8080/cs-invocation/achieve-cs-goal-metrics?ontology=mmgo-goals&goal=get-video-metadata-Goal&has-method=getvideosbykeywords&has-keywords=the&assumption=purpose-type;80;0;0;environment-type;80;100>

The request above achieves a goal and selects the most suitable one depending on the numbers which are part of the assumption parameter. Here, the assumption parameter is represented through measurements (for instance defining a particular location) which are matched to predefined instances (and their measurements) as part of the semantic descriptions within the Broker.

GET-YOUTUBE-TRAILER-GOAL: This goal provides functionality to retrieve metadata describing any YouTube video clip that matches a given set of keywords. Consider the following achieve-goal URL. The keywords that need to be matched are "open" and "university". These keywords are given as the value of the HAS-KEYWORDS parameter of the achieve-goal URL (the keywords are actually separated by two-semi-colons in a single string):

<http://notube.open.ac.uk:8080/api-rest/achieve-goal?ontology=MMPO-GOALS&goal=GET-YOUTUBE-TRAILER-GOAL&HAS-KEYWORDS=%22open;;university%22>

The result is the following metadata:

```
<metadata>
  <item>
    <title>Open University blooper reel- A Bit of Fry and Laurie- BBC
Comedy</title>
    <description>Stephen Fry introduces his favourite Open University blooper
reel in his own outtake tv sketch. Hilarious video from comedy superstars Stephen
Fry and Hugh Laurie. Video taken from BBC show 'A Bit of Fry and
Laurie'.</description>
    <url>http://www.youtube.com/watch?v=2un9rO2ZF4g&feature=youtube_gdata</url>
    <pubDate>2008-10-15T15:42:27.000Z</pubDate>
  </item>
  <item>
    <title>The Two Ronnies - Open University Lecture</title>
    <description>Ronnie Barker gives a lecture on archaeology in this hilarious
clip. This clip belongs to the BBC.</description>
    <url>http://www.youtube.com/watch?v=qFLOHu8Ozm8&feature=youtube_gdata</url>
    <pubDate>2008-09-22T15:35:41.000Z</pubDate>
  </item>
  ...
</metadata>
```

5.1.2 EPG retrieval functionality

GET-EPG-BY-KEYWORD-AND-DATE-GOAL: Similar to the video-metadata-retrieval goal, another goal is being provided here which allows the user to query EPG metadata from a wide variety of EPG sources. The current goal provides a single entry point to query the different EPG sources. Therefore we have annotated services provided by VU (<http://www.notube.tv/wiki/index.php/Services/iZapperdatawarehouse>) and have provided additional services on top of the metadata feeds provided by Engyn Media (http://www.notube.tv/wiki/index.php/LInk_for_EPG_I_fanzy). Each service allows to query EPG of a specific channel, while query parameters are the time period (start- and enddate) and a set of keywords which is used to filter the metadata. Note, the keywords are optional and could be left blank. While querying each channel/service individually is time consuming and requires some mechanism of selecting the most appropriate service/channel for a given context or user, our goal-based approach provides a single entry point to submit a query while the broker automatically selects the most suitable EPG services for a given context and orchestrates and executes these.

The following example URI sends a request to achieve the get-epg-metadata goal:

[http://notube.open.ac.uk:8080/cs-invocation/achieve-cs-goal?ontology=mmmpo-goals&goal=get-EPG-BY-KEYWORD-AND-DATE-GOAL&has-method=getvideosbykeywords&has-keywords=%22week%22&has-start-date=%2219-11-2009%2000:00%22&has-end-date=%2225-11-2009%2023:59%22&assumption=\(%22english-language%22\)](http://notube.open.ac.uk:8080/cs-invocation/achieve-cs-goal?ontology=mmmpo-goals&goal=get-EPG-BY-KEYWORD-AND-DATE-GOAL&has-method=getvideosbykeywords&has-keywords=%22week%22&has-start-date=%2219-11-2009%2000:00%22&has-end-date=%2225-11-2009%2023:59%22&assumption=(%22english-language%22))

The request above sends a request to the Broker for EPG metadata within the specified period ("has-start-date", "has-end-date") and which match the keyword "week" (specified via the "has-keywords" parameter). While these parameters are used as inputs for the actually selected services, the "assumption" parameter refers to instances within the Broker which are used for selecting the most appropriate EPG source. Within the query above, the user asks for "english-language" leading to the Broker selecting only services which provide English language EPG data. Other values would be, for instance, "german-language", "dutch-language", "turkish-language" which would all lead to the selection of different EPG sources.

The result is a consolidated set of XML metadata of EPG programmes which match the criteria described above. Since different services respond distinct metadata following different XML schemas, the IRS-III automatically applies a common schema and returns the respective results.

For instance the following would be part of the response to the query above:

```
<metadata>
  <item>
    <title>The National Lottery: Midweek Draws</title>
    <description></description>
    <url>b00p1dxt</url>
    <pubDate>2009-11-18 22:35:00</pubDate>
    <channel>BBC 1</channel>
  </item>
  <item>
    <title>This Week</title>
    <description>A weekly showcase for reports from the BBC's network of over 250
global correspondents.</description>
    <url></url>
    <pubDate>2009-11-22 02:30:00</pubDate>
    <endDate>2009-11-22 03:00:00</endDate>
    <channel>BBC World</channel>
  </item>
</metadata>
```

GET-ENRICHED-EPG-BY-KEYWORD-AND-DATE-GOAL: This goal is similar to the GET-EPG-BY-KEYWORD-AND-DATE goal described above, in that it takes the same input (keywords, start date, end date, and language). The difference is that this goal returns **enriched** EPG metadata. That is, in addition to the regular EPG data (e.g. title and description), the goal returns metadata enriched with data and links from a number of sources including IMDB, Freebase, and DBPedia.

For example, the following URL sends a request to the broker for ENRICHED EPG data within the period "19-04-2010 12:00" and "19-04-2010 13:00", for "english-language" channels, and with no keyword filtering (i.e. has-keyword parameter is set to ""):

[http://notube.open.ac.uk:8080/cs-invocation/achieve-cs-goal?ontology=mmppo-goals&goal=get-ENRICHED-EPG-BY-KEYWORD-AND-DATE-GOAL&has-method=getvideosbykeywords&has-keywords=%22%22&has-start-date=%2219-04-2010%2012:00%22&has-end-date=%2219-04-2010%2013:00%22&assumption=\(%22english-language%22\)](http://notube.open.ac.uk:8080/cs-invocation/achieve-cs-goal?ontology=mmppo-goals&goal=get-ENRICHED-EPG-BY-KEYWORD-AND-DATE-GOAL&has-method=getvideosbykeywords&has-keywords=%22%22&has-start-date=%2219-04-2010%2012:00%22&has-end-date=%2219-04-2010%2013:00%22&assumption=(%22english-language%22))

The result is the following XML:

```
<item>
  <title>Working Lunch</title>
  <description></description>
  <url>http://purl.org/identifiers/epg/broadcast/b00s3yx8</url>
  <pubDate>2010-04-19T12:30:00Z</pubDate>
  <endDate>2010-04-19T13:00:00Z</endDate>
  <channel>BBC 2</channel>
  <enrichments>
    <enrichment>
      <eName>brand</eName>
      <eValue>Working Lunch</eValue>
      <eUrl></eUrl>
      <eSource>BBC</eSource>
    </enrichment>
    <enrichment>
      <eName>channeluri</eName>
      <eValue>BBC TWO</eValue>
      <eUrl>http://www.bbc.co.uk/programmes/bbctwo#service</eUrl>
      <eSource>BBC</eSource>
    </enrichment>
    ...
  </enrichments>
</item>
```

Note that the enrichment makes use of the program enrichment services in the Datawarehouse, which in turn also makes use of the WP4 LUPedia enrichment service.

GET-EPG-BY-KEYWORD-PERIOD-AND-LANGUAGE: This goal is meant as a more general approach to the task of retrieving EPG data based on user language. The goal orchestrates two separate bits of functionality:

- retrieving a list of channels that match a particular language (e.g. Italian, English, ...) and then
- retrieving EPG data for the list of channels retrieved in the first step. This second step is performed using the **GET-ENRICHED-EPG-BY-KEYWORD-DATE-AND-CHANNELS** goal described above.

Consider the following achieve-goal URL. This takes as input a language (in this case Italian), a particular period for the EPG data (in this case "19-04-2010 00:00" until "19-04-2010 23:59"), and a particular enrichment source (in this case "imdb"). Note there are no keywords supplied so the EPGs are not in this case filtered by keyword.

<http://notube.open.ac.uk:8080/api-rest/achieve-goal?ontology=MMPO-GOALS&goal=GET-EPG-BY-KEYWORD-PERIOD-AND-LANGUAGE-GOAL&HAS-LANGUAGE=Italian&HAS-KEYWORDS=%22%22&HAS-START-DATE=%2219-04-2010%2000:00%22&HAS-END-DATE=%2219-04-2010%2023:59%22&HAS-ENRICHMENT-SOURCE=%22imdb%22>

The result is the following EPG data for RAI1, RAI2, and RAI3, the channels we currently have annotated as "Italian":

```
<metadata>
  <item>
    <title>Telegiornale</title>
    <description></description>

<url>http://purl.org/identifiers/epg/broadcast/10027_2010_04_19_01_00_00</url>
  <pubDate>2010-04-19T01:00:00Z</pubDate>
  <endDate>2010-04-19T01:30:00Z</endDate>
  <channel>Rai Uno</channel>
  <enrichments>
    <enrichment>
      <eName>Actor</eName>
      <eValue>Jos&#xE9; Rodrigues Dos Santos</eValue>
      <eUrl>http://www.imdb.com/name/nm0234335</eUrl>
      <eSource>IMDB</eSource>
    </enrichment>
    <enrichment>
      <eName>Actor</eName>
      <eValue>Jos&#xE9; Alberto Carvalho</eValue>
      <eUrl>http://www.imdb.com/name/nm1563271</eUrl>
      <eSource>IMDB</eSource>
    </enrichment>
    ...
  </enrichments>
</item>
  ...
</metadata>
```

5.1.3 Enrichment functionality

GET-LUPEDIA-ENRICHMENT-GOAL: This goal exposes the WP4 LUPedia functionality -- i.e. the DBPedia Look-up service. The service takes as input any text string. The text will be analysed to find named entities, and these named entities will be looked up in the DBPedia knowledge base and enriched descriptions will be returned.

Consider the following achieve-goal URL. The input parameter for the text is HAS-TEXT:

<http://notube.open.ac.uk:8080/api-rest/achieve-goal?ontology=MMPO-GOALS&goal=GET-LUPEDIA-ENRICHMENT-GOAL&HAS-TEXT=%22Elvis%20Presley%22>

The result is the following metadata:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <lookupResult xmlns="http://lookupws.ontotext.com">
    <lookups>
      <startOffset>1</startOffset>
      <endOffset>14</endOffset>
      <instanceUri>http://dbpedia.org/resource/Elvis_Presley</instanceUri>
      <instanceClass>http://dbpedia.org/ontology/Person</instanceClass>
    </lookups>
  </lookupResult>
```

5.1.4 Beancounter functionality

GET-BEANCOUNTER-USER-PROFILE-GOAL: This goal exposes the Beancounter functionality of getting a user profile. The Beancounter service is documented in the ["Get user profile"](#) section of the Getting User Data wiki page.

Consider the following achieve-goal URL. Note that the input to the goal is the name of a Beancounter user (in this case michele_minno) given as the value of the HAS-USER parameter:

http://notube.open.ac.uk:8080/api-rest/achieve-goal?ontology=MMPO-GOALS&goal=GET-BEANCOUNTER-USER-PROFILE-GOAL&HAS-USER=michele_minno

The following JSON string is returned:

```
{ "interests": [
  { "identifier": "http://dbpedia.org/resource/Category:Actors_from_New_York", "weight":
    0.020408162847161293, "scale": "0..92" },
  { "identifier": "http://dbpedia.org/resource/Category:Actors_who_attempted_suicide", "
    weight": 0.05000000074505806, "scale": "0..92" },
  { "identifier": "http://dbpedia.org/resource/Category:Irish_Americans", "weight": 0.044
    44444552063942, "scale": "0..92" },
  { "identifier": "http://dbpedia.org/resource/Category:Actors_from_Texas", "weight": 0.0
    5000000074505806, "scale": "0..92" },
  { "identifier": "http://dbpedia.org/resource/Category:2000s_comedy_films", "weight": 0.
    02222222276031971, "scale": "0..92" },
  ...
  { "identifier": "http://dbpedia.org/resource/Category:American_screenwriters", "weight
    ": 0.020408162847161293, "scale": "0..92" },
  { "identifier": "http://dbpedia.org/resource/Category:American_Jews", "weight": 0.02222
    22276031971, "scale": "0..92" }
]
}
```

GET-BEANCOUNTER-USER-ACTIVITIES-GOAL: This goal exposes the Beancounter functionality of retrieving a list of user activities. The documentation for this Beancounter service can be found in the ["Get user activities"](#) section of the Getting User Data wiki page.

Consider the following achieve-goal URL. Note that the input to the goal is the name of a Beancounter user (in this case michele_minno) given as the value of the HAS-USER parameter:

http://notube.open.ac.uk:8080/api-rest/achieve-goal?ontology=MMPO-GOALS&goal=GET-BEANCOUNTER-USER-ACTIVITIES-GOAL&HAS-USER=michele_minno

The result is the following XML describing the user's activities:

```
<?xml version="1.0"?>
<activities>
  <activity>
    <verb>Looked</verb>
    <object>the pillow factory</object>
    <source>glue</source>
  </activity>
  <activity>
    <verb>Looked</verb>
    <object>Eiffel Tower</object>
    <source>glue</source>
  </activity>
  <activity>
    <verb>Looked</verb>
    <object>Owen Wilson</object>
    <source>glue</source>
  </activity>
  <activity>
    <verb>Looked</verb>
    <object>Ben Stiller</object>
    <source>glue</source>
  </activity>
  <activity>
    <verb>Looked</verb>
    <object>Zoolander</object>
    <source>glue</source>
  </activity>
  ...
</activities>
```

GET-BEANCOUNTER-USER-USED-SERVICES-GOAL: This goal exposes the Beancounter functionality of retrieving the social network services (e.g. facebook, glue, lastfm) that are used by a particular user. The documentation for this service can be found in the ["Get user used services"](#) section of the [Getting user data](#) wiki page.

Consider the following achieve-goal URL. Note that the input to the goal is the name of a Beancounter user (in this case michele_minno) given as the value of the HAS-USER parameter:

http://notube.open.ac.uk:8080/api-rest/achieve-goal?ontology=MMPO-GOALS&goal=GET-BEANCOUNTER-USER-USED-SERVICES-GOAL&HAS-USER=michele_minno

The result is the following JSON string:

```
{"glue":85,"facebook":7}
```

5.2 Lightweight RDF service & goal annotations

This section describes the work done in supporting the task of semantically describing and annotating the services in the NoTube platform. One of the requirements for the services work in the project (done under both WP5 and WP6) is the semantic annotation of TV data and service sources (see in particular Task 5.2 as set out in the Description of Work and as anticipated in D5.1a/b). Thus, the work here is motivated by a need to document, in both human-readable and machine-understandable form, the services throughout NoTube, and to provide a basis for formal service descriptions that automate the discovery of services.

The aim of SmartLink is to be the main environment to document, annotate and search for services within NoTube. As such, it replaces the existing service documentation efforts (i.e. the services catalogue and the Word forms) with a single tool that allows not only documentation for humans (i.e., us developers) but also structured, machine-understandable annotations that are accessible for the broker or third-party applications. Furthermore, SmartLink provides a SPARQL endpoint, allowing

us to share data about all our NoTube services publicly in a structured way. While SmartLink currently mainly aims at providing structured, machine-processable annotations and query facilities – without supporting service execution - current work aims at an API layer which allows to discover and execute in a more automated manner. Finally, the long-term plan is that the SmartLink/iServe environment will provide capabilities to fully replace the current brokering environment that is based on IRS-III. The new tools will allow the OU to provide its services via a single coherent environment based on the emerging Linked Services approach.

Listing 1 depicts an extract of the RDF description of a particular NoTube service (LUPEDIA¹¹) which performs a lookup of free text in DBPedia in order to allow enrichment of EPG metadata with additional DBPedia entities. Besides the utilisation of model references to external vocabularies – please note the highlighted reference (<sa:modelReference>) at the bottom – the listing also highlights some of the integrative elements which had been utilized within NoTube. For instance, the <so:hasGoal>-property refers to a particular WSMO goal instance within IRS-III to cater for our model referencing approach.

```

<rdf:RDF xmlns:so="http://www.purl.org/vocabularies/service-ontology#"
xmlns:msm="http://cms-wg.sti2.org/ns/minimal-service-model"
xmlns:saw="http://www.w3.org/ns/sawSDL#"...>

<rdf:Description
rdf:about="http://lupedia.ontotext.com/lookup#text2rdfa">
  <so:hasContactPerson>Stefan Dietze</so:hasContactPerson>
  <so:hasGoal>GET-LUPEDIA-ENTITIES-GOAL</so:hasGoal>
  <msm:hasInput
rdf:resource="http://lupedia.ontotext.com/lookup/input#lookupText"/>
  <msm:hasOutput
rdf:resource="http://lupedia.ontotext.com/lookup/output#lookupResult"/>
  ...
  <so:hasOneLiner>Lookup of free text in DBPedia based on entity recognition and
DBPedia lookup.</so:hasOneLiner>
  <msm:hasOperation
rdf:resource="http://lupedia.ontotext.com/lookup/#text2rdfa"/>
  <sa:modelReference rdf:resource="http://www.service-
finder.eu/ontologies/ServiceCategories#Multimedia"/>
  <sa:modelReference rdf:resource="http://www.service-
finder.eu/ontologies/ServiceCategories#Content"/>
  ...
</rdf:Description>
...
<rdf:Description
rdf:about="http://lupedia.ontotext.com/lookup/output#lookupResult">
  <sa:modelReference rdf:resource="http://dbpedia.org/data/Entity"/>
</rdf:Description>
...
</rdf:RDF>

```

Fig. 4. RDF excerpt of LUPEDIA service description based on MSM.

¹¹ <http://lupedia.ontotext.com/>

6 Integration into NoTube use case scenarios

In this section, we describe the integration of the WP5 service brokering technology into the NoTube use cases.

6.1 WP7a scenario – personalised semantic news (PSN)

6.1.1 Retrieving Enriched EPG data

While the EPG retrieval scenario is central to the WP7a scenario, it is fully described as part of the evaluation scenario description in Section 7.

6.1.2 NIC transformation and enrichment

We have been working with IRT, RAI, and TXT to finalise a workflow for creating enriched News Item Containers (NICs) that are structured according to the TV-Anytime (TVA) specification¹². These enriched TVA-NICs are created in a process that includes three main phases of *a*) retrieving the Prestospace output of RAI’s internal ANTS server, *b*) transforming this Prestospace output into TVA output (using IRT’s metadata transformation service), and *c*) enriching the resulting TVA output (using OT’s LUPedia service). Those steps in the process that involve using services from the NoTube service providers (i.e. steps ‘b’ and ‘c’ aforementioned) are executed via the Broker. Fig. 5. shows the steps of this workflow. The steps are shown divided into three categories: steps A1–A3, B1–B7, and C1–C5. Steps A1–A3 describe how the Prestospace Metadata Repository is populated; B1–B7 describe how Prestospace metadata is transformed into TVA-NICs; and C1–C5 describe how the TVA-NICs are enriched. Note that steps C1–C5 do not have to flow synchronously after B1–B7 and they are repeated as many times as the RAI Personalised Semantic News (PSN) application needs to enrich a TVA-NIC.

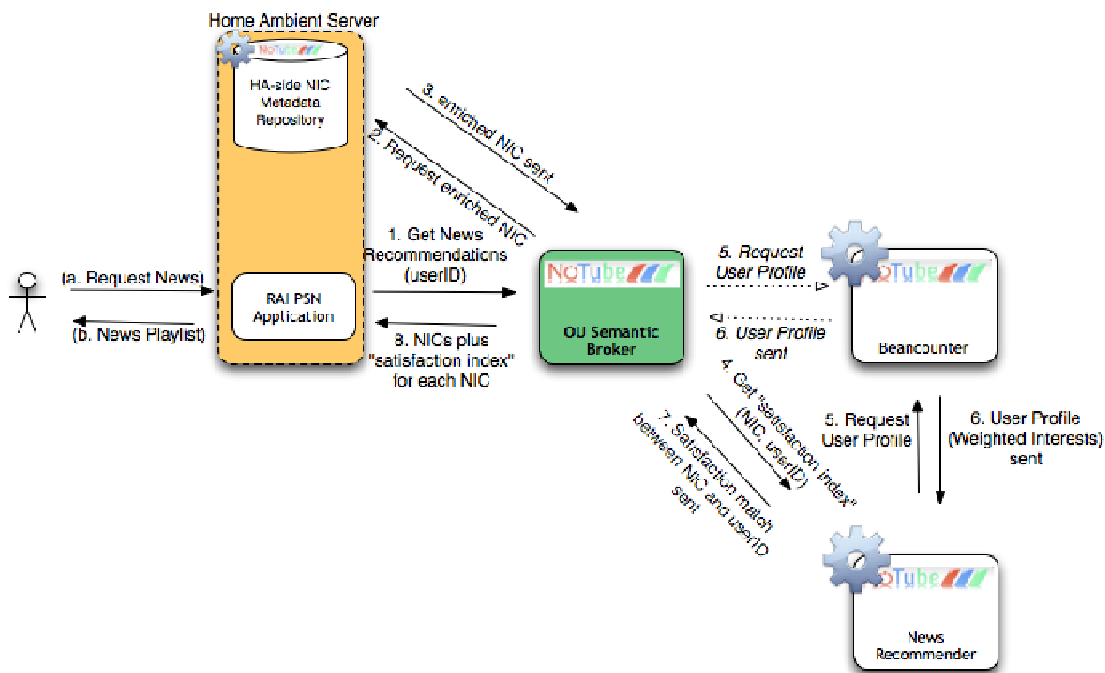


Fig. 5. Workflow for transforming Prestospace to TVA-NIC.

6.1.3 News recommendation

¹² Minutes of the meetings to discuss this workflow can be found on the wiki: <http://www.notube.tv/wiki/index.php/29-Jul-10-WP2-telecon-minutes>, <http://www.notube.tv/wiki/index.php/25-Aug-10-WP2-telecon-minutes>.

We have also worked on describing the workflow, on the Home Ambient (i.e. User) side, of building a personalised news playlist for the user. This workflow uses the enriched TVA-NICs that are created during the NIC transformation and enrichment workflow on the Service Provider side. This workflow is shown in **Error! Reference source not found.** It uses services provided by two NoTube components: the News Recommender and the Beancounter. At present, the workflow indicates that the News Recommender retrieves a user profile directly from the Beancounter. However, from WP5’s perspective, we should consider executing this step via the Broker as well.

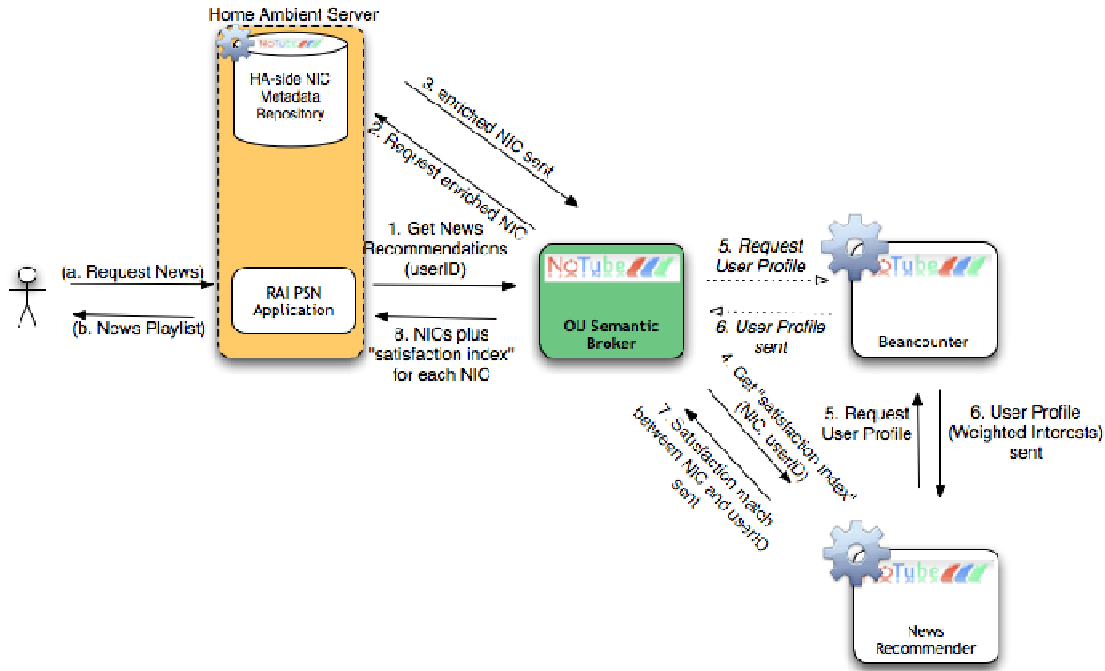


Fig. 6. The steps of the workflow for building a personalised news playlist using enriched TVA-NICs.

6.1.4 Video reframing

We have also begun to consider the issues with integrating what can be considered processor-intensive (because of the large media files that are involved) asynchronous services such as Loudness Harmonisation (from IRT, to be used in WP7a), and Video Reframing and Ad Insertion (from TGV, to be used in WP7b).

Such services present new technical challenges for integrating via a middleware-Broker (which until now has mainly been focusing on handling requests and responses involving textual data and metadata). Addressing such challenges presents new research opportunities for WP5.

In relation to this, IRT have taken a lead to sketch a possible strategy for meeting the technical challenge of integrating processor-intensive, asynchronous services. On the wiki¹³, IRT have sketched possible ways of a partial integration with the Broker, where the strategy would be to use the Broker to exchange preliminary details to coordinate the integration, but then the client (in this case the use-case partner) establishes a direct connection to the service provider for completing the actual task (in this case the Loudness Harmonisation provided by IRT).

From a WP5 perspective, if we can implement such a strategy for brokering asynchronous services, it would give us a nice research story for the review (most likely Y3 review rather than Y2).

In the meantime, together with WP6, WP4 (IRT and TGV), WP7a, and WP7b, we need to determine the workflow for integrating these services in the relevant use cases (similar to what we have described in previous sections).

¹³ http://www.notube.tv/wiki/index.php/Loudness_Harmonization

We have investigated ways of including the Broker in a workflow with the processing-intensive, asynchronous services being created within WP4. In this example, the RAI application needs to crop the video of a news item so that the video can fit the smaller screen of a mobile device. This functionality is performed by the Video Reframing service developed within WP4 by TGV. Currently, we have proposed a workflow that consists of the following steps:

- The RAI PSN application invokes the (hypothetical) goal in the Broker, CROP-VIDEO-GOAL, and provides the Broker with the TVA-NIC ID.
- The Broker uses the TVA-NIC ID to retrieve the TVA-NIC from the metadata repository.
- The Broker then analyses the NIC to extract the appropriate physical location (in this case, the appropriate physical location could be the location of the high-quality version of the content in the file format required by the TGV Reframing service).
- The Broker then calls the TGV Reframing service, passing the service the physical location of the video content.
- The TGV Reframing service retrieves the content from the physical location and temporarily stores the video content on the local TGV platform.
- The service then crops the video content and stores the newly cropped video back in the content repository at a new physical location.
- When this happens, the Broker updates the original TVA-NIC by adding the new physical location as another physical location for the same news item.
- The Broker then stores the updated TVA-NIC in the metadata repository.

6.2 WP7b scenario – personalised TV guide with adaptive advertising

6.2.1 Retrieving enriched EPG data

For the first year WP7b prototype, the focus was on integrating services that retrieved electronic programme guide (EPG) data. At its most basic this involved exposing, via the Broker, a service which can search the different available EPG feeds for programme information that falls within a given time period. However, this basic functionality was incrementally extended to include EPG selection based on desired broadcast language and on user-specified keywords. Furthermore, the raw EPG data was enriched with DBPedia, IMDB, and SKOS entities that provided additional information about each programme in the EPG feed. The iFanzly client was able to invoke this functionality via the Broker (specifically a goal called GET-EPG-BY-KEYWORD-PERIOD-AND-LANGUAGE-GOAL) and display it in the interface.

6.2.2 Profile-based TV programme recommendation

One of the main aims of the iFanzly client is to display TV program recommendations based on user's activities and personal interests as stored in a user profile. Thus the core services that underpin this scenario are user profiling and recommendation services provided by WP3, and these services have been semantically described in the Broker and exposed as goals. For example, two key functionalities from the Beancounter that are exposed via the Broker are the functionality to retrieve a user profile and the functionality to retrieve a list of user activities. With respect to the first of these functionalities, we have provided a goal called GET-BEANCOUNTER-USER-PROFILE-GOAL¹⁴. The service exposed via the Broker takes a known user and returns the profile for that user. So for example, there is a user in the Beancounter called michele_minno, thus to retrieve this user's profile, the following URL is invoked:

http://notube.open.ac.uk:8080/api-rest/achieve-goal?ontology=MMPO-GOALS&goal=GET-BEANCOUNTER-USER-PROFILE-GOAL&HAS-USER=michele_minno

¹⁴ Goal documented here: <http://www.notube.tv/wiki/index.php/Wp5-goals-documentation#GET-BEANCOUNTER-USER-PROFILE-GOAL>

With respect to the second of these Beancounter functionalities, we have provided a goal called GET-BEANCOUNTER-USER-ACTIVITIES-GOAL¹⁵. The service exposed via the Broker takes a known user and returns the list of activities for that user. So for example, using the same michele_minno user as above, to retrieve this user's list of activities the following URL is invoked:

http://notube.open.ac.uk:8080/api-rest/achieve-goal?ontology=MMPO-GOALS&goal=GET-BEANCOUNTER-USER-ACTIVITIES-GOAL&HAS-USER=michele_minno

6.2.3 Ad insertion

As well as in WP7a, in WP7b we have also been investigating ways of including the Broker in a workflow with the asynchronous services being created within WP4. In the WP7b example, the iFanz application needs to insert ads in a movie and store two separate versions of the movie: one version with ads and one version without ads (shown to paying customers). This functionality is performed by the Ad Insertion service developed within WP4 by TGV. In addition, this workflow uses the Loudness Harmonisation service. The reason for running the Loudness Analyser on the new movie with the ads inserted is that the loudness levels of ads are usually higher than the movie so the user may want the different levels to be harmonised. However, we have to bear in mind the potential commercial conflict if the content provider is reducing volume of ads which advertisers have paid to be inserted into the content. Currently we have proposed a workflow that consists of the following steps:

- The Stoneroos iFanz application invokes the (hypothetical) goal in the Broker, INSERT-AD-GOAL.
- [...] (The steps to obtain physical location have not been decided)
- The TGV Ad Insertion service retrieves the movie from the physical location and temporarily stores the movie on the local TGV platform.
- The service inserts the ad into the movie and stores the new version of the movie back into the repository at a new physical location.
- When this happens, the Broker calls the IRT Loudness Analyser (LA) service and passes the new physical location to the IRT service.
- The IRT LA service retrieves the movie, with ads inserted, and temporarily stores the movie on the local IRT platform.
- The LA service analyses the loudness levels of the movie audio and returns a dB value (?)
- [...] (The steps after the audio has been analysed have not been decided)

6.3 WP7c scenario – Internet TV in the social Web

6.3.1 Buttons protocol brokering using XMPP

The scenario we have in mind is something that Dan alluded to in Istanbul for WP7c:

A user has the Buttons remote controller software installed on his iPod and wants to be able to control a range of set-top-box environments (MythTV, AppleTV, Freevo, etc.). What the Buttons software on the iPod needs to know is which of the functions are available on which STB device.

The idea that we want to work on in the next months is to use the Broker as the middleware between the iPod Buttons software and individual set-top-box environments. The Broker would implement the Buttons protocol and also some discovery mechanism (possibly incorporating the XEP-0030¹⁶ specification) for determining the capabilities of each set-top-box environment. The iPod Buttons

¹⁵ Goal documented here: <http://www.notube.tv/wiki/index.php/Wp5-goals-documentation#GET-BEANCOUNTER-USER-ACTIVITIES-GOAL>

¹⁶ <http://xmpp.org/extensions/xep-0030.html>

software would communicate with the Broker using the Buttons protocol and the Broker would handle the mappings to the remote control functions supported by each set-top-box. This is depicted in Fig. 7. .

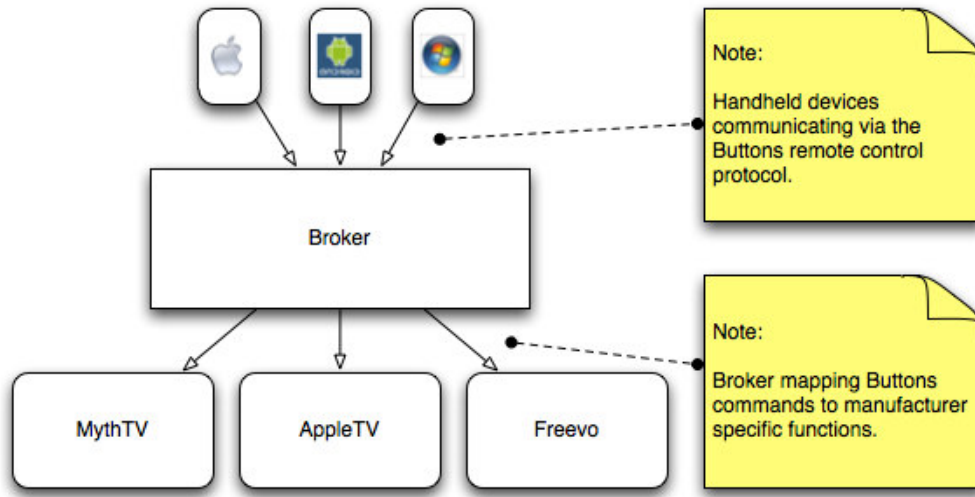


Fig. 7. Mapping Buttons protocol commands to manufacturer-specific functions via the Broker.

If we could manage to implement this idea then it could be an interesting story to tell at the next NoTube review. And for us it contains some interesting research about how for some scenarios we have to change our thinking about brokering using request-response interactions (as in HTTP) to brokering using more message-oriented interactions (as in XMPP).

In the meantime, we are making some progress, slowly getting our heads wrapped around the XMPP specification (RFC3920¹⁷). The next steps are to finish implementing the core XMPP concepts from RFC3920 in the Broker and then to incorporate the specific XMPP requirements of the Buttons protocol (e.g. the XMPP-stanza format for the different Buttons commands).

¹⁷ <http://tools.ietf.org/html/rfc3920>

7 Evaluation plan

7.1 Evaluation criteria

This section describes the criteria for evaluating the Semantic Broker. The section is divided into two parts: Functional evaluation criteria and Non-functional evaluation criteria. Functional evaluation is about measuring the impact of the WP5 Semantic Broker in its role as a middleware capable of annotating and orchestrating different services to achieve a specific goal. Non-functional evaluation is about assessing certain attributes of the Semantic Broker as a piece of software.

7.1.1 Functional evaluation criteria

At the heart of the functional evaluation in WP5 is the evaluation of the WP5 technology with respect to the functional requirements pre-scribed in D5.1a and D5.1b. Therefore, the following subsection (7.1.1.1) summarises the overall requirements and derives evaluation criteria. In addition, subsection 7.1.1.2 describes additional evaluation opportunities which might be exploited during the evaluation activities to further measure the benefits and drawbacks.

7.1.1.1 Overall functional requirements

The core functional evaluation has to reflect on the implemented functionalities currently supported by the WP5 environments with respect to the functional requirements proposed in the previous deliverables D5.1a and D5.1b. Deliverable D5.1b defined a number of service integration requirements which are listed in the table below.

Table 1 - Summarised, enumerated list of requirements for the Semantic Service Integration.

Requirement Number	Requirement description
R1	To automatically discover repositories and access services/content.
R2	To automatically discover metadata enrichment services
R3	To automatically discover external Web services/content
R4	To orchestrate appropriate services in order to enable the application layer to query for context filtered items
R5	To mediate between distinct service I/O messages
R6	To abstract from service implementations and communication standards
R7	To automatically discover metadata conversion services
R8	To automatically discover context services
R9	To automatically discover user-profiling services
R10	To automatically discover recommendation services
R11	To discover services for classifying TV-related contents (programmes, broadcast events, brands) and user categories
R12	To store structured documentation of services used within NoTube
R13	To provide project-wide access to service documentation in order to facilitate development
R14	To provide search facilities to allow NoTube developers to find and reuse available services

While the above requirements are already taking into the account specific kind of services – i.e. the usage context of the brokering functionalities – we abstract the following higher-level functional criteria for the WP5 technology.

Table 2 - Summarised, enumerated list of evaluation criteria (EC) for the Semantic Service Integration.

Evaluation criteria	Requirement description	Derived from
EC1	Abstracting from services implementations based on semantics	R6, R12
EC2	Service annotation and tracking facilities for developers	R14
EC3	Automated services discovery.	R1-R3, R7-R11
EC4	Automated services orchestration	R4
EC5	Automated services I/O mediation	R5

While the above criteria are the core functional criteria to be evaluated, the following subsection describes additional, optional criteria which might be taken into account during the evaluation phase.

7.1.1.2 Additional functional evaluation opportunities

Derived from the above functional requirements, another functional evaluation criterion to be evaluated is labelled **Development-time Scalability**, which encapsulates a measure of the effort that a developer has to spend integrating client applications with NoTube services. The Development-time Scalability criteria can be further divided into two components: Service Integration Costs and Service Mediation Costs, which will both be described in greater detail below.

The rationale for this additional development-focused criterion for evaluating the Broker is that the working hypothesis for our Broker research (and which motivates the use of the Broker in NoTube) is that using the Semantic Broker as a middleware component allows the client applications to be loosely coupled with the various available NoTube services. Such loose coupling reduces the amount of development effort (for which we introduce a new unit of measure called *the coding unit*) that client application developers have to spend integrating with the NoTube services.

We now describe the two components of Development-time Scalability, i.e. Service Integration Costs and Service Mediation Costs, in greater detail:

(A) Development-time Scalability: Service Integration Costs

Here the focus is on measuring the effort required to integrate NoTube services within client applications. In terms of service integration, one major benefit that we claim of our work is that integration effort on the part of the client application developer is reduced. This is revealed most prominently in cases where service interfaces are constantly changing. Because the semantic service descriptions stored in the Semantic Broker are declarative in nature, it is easier to change such descriptions when a service interface changes than it would be to change the code of client applications in those cases where the client application directly integrates the services.

The reduction in integration effort on the part of application developers is also evidenced by the fact that the interfaces exposed to client applications via the Broker can be more abstract (and therefore easier to integrate) than the interfaces provided directly by the services. Let's take the simple example of integrating the LUPEDIA service to illustrate the reduced integration costs using a Broker-based approach. The raw LUPEDIA service takes as input a piece of text to be enriched as well as a set of 8 filter parameters:

http://lupedia.ontotext.com/lookup/text2xml?lookupText=The+Football+League+Show&skip_sh3=false&skip_stp=false&case_sensitive=false&keep_fnl=false&skip_ldata=false&single_match=false&keep_highest=true&lang=it

However, the use case application developers consider that it is ideal for them to give the Broker just text as input. This would mean that the application would not have to manage LUPEDIA filtering; instead the Broker can handle passing the different filter combinations to the LUPEDIA service and can expose just a simplified, more abstract interface to the client application:

<http://localhost:8080/api-rest/achieve-goal?ontology=MMPO-GOALS&goal=GET-LUPEDIA-ENRICHMENT-GOAL&HAS-TEXT=The+Football+League+Show>

In the evaluation scenario (described in Section 7.2) the key variables to be measured with respect to measuring Service Integration Costs are:

- (1) Number of services, S – these are the distributed NoTube services that are currently available for the 2nd year prototypes and that need to be integrated into the prototypes.
- (2) Number of client applications, A – these are the distinct client applications that need to be integrated with the NoTube services. These client applications are primarily developed within the use-case WPs, but our overall aim is to be able to support any external client application.
- (3) Integration coding effort, E – this is the effort required to integrate the services in (1) into the applications in (2). We programmatically measure this effort in terms of an abstract *coding unit*. One coding unit in this case is equated to the effort required to manually code one service request from one application, including the processing of I/O.

(B) Development-time Scalability: Service Mediation Costs

Here the focus is on measuring the effort to deal with multiple output schemas from the NoTube services. We argue that one of the major benefits of the Semantic Brokering approach is that the Broker can lift the outputs from a variety of output schemas and produce a single output schema, in agreement with the client application, that abstracts from the individual schemas, thereby supporting interoperability between services that would not have otherwise been possible.

In the evaluation scenario (described in Section 7.2), the key variables to be monitored with respect to measuring Service Interoperability Costs are:

- (1) Number of service formats, F – these are the individual output formats from the NoTube services. These formats include XML, RDF, and JSON.
- (2) Number of client applications, A – as above, these are the distinct client applications that need to be integrated with the NoTube services.
- (3) Integration coding effort, E – as above, this is the effort required to integrate the services in (1) into the applications in (2). This, as above, is also measured in coding units.

7.1.2 Non-functional evaluation criteria

There are three non-function evaluation criteria to be considered – System Performance, Runtime scalability, and Usability.

- *System Performance* – Generic Performance metrics include execution time and throughput.. Compare the performance of directly integrated services and the same services orchestrated by the broker. This will give an indication of the overhead introduced by the tool with respect to the benefits
- *Runtime scalability* – Scalability of SWS tools are associated with the ability to perform an activity (e.g. discovery) involving an increasing amount of data, services, and service descriptions. This can be measured together with performance (above), however, this is more related to the scalability of repositories than to the tools themselves.” [Ref: SEALS D14.1]
- *Usability* – This encompasses both usability of the annotation environment as well as usability of the services query interface.

7.2 Evaluation procedure

The previous section focused on what particularly will be evaluated in WP5. This section describes how we will perform the evaluation. It starts by describing the formal setting for the evaluation and then goes on to describe the actual evaluation activities to be carried out.

7.2.1 Formal setting: parameterised EPG retrieval

It is planned, to focus on a particular scenario when carrying out the actual evaluation. Due to its well-definedness, we have opted for the *parameterised EPG retrieval* scenario which is of general importance to NoTube and covers most of the evaluation criteria to be investigated. Here we provide details of the evaluation setting for this scenario. That is, in the context of the EPG retrieval scenario, we enumerate all of the available services, all of the service schemas, each of the client applications that make use of the EPG retrieval functionality, and the components of the Broker goal (please see also Section 5.1) used to integrate the services with the client applications.

(A) Available EPG services

EPG services are available from two main NoTube partners: VUA (who maintain what is called a Datawarehouse that exposes most of the EPG services) and Engin Medya. The available services are listed below:

- Datawarehouse EPG endpoint (<http://services.notube.tv/epg/datawarehouse.php>)
 - o Getprogrammestoday
 - o Getprogrammestoday&channelid
 - o Getprogrammestodayrdf&channelid
 - o Getprogrammesperiod&channel_id&start&stop
 - o Programmesperiodenriched&channeluri&start&stop
 - o Programmesskosperiodenriched&channeluri&start&stop
 - o Dbpediaenrichedperiod&channeluri&start&stop
 - o Imdbenrichedperiod&channeluri&start&stop
 - o Getprogrammesperiodrdf
- Engin endpoint (<http://tumdata.triplinq.com/phpscripts/xmlfeed4allifanzy.php>)

Note that for each of the EPG services listed there is a large number of channels (approx. 81 from Engin and 1815 from DWH, although the main channels number about 25) that can be retrieved.

(B) Used EPG data formats

Not only are there many different EPG services and different channels for each service, there is also some variation in the output format used for the EPG services. These are listed below:

- JSON (the output format of most of the DWH services)
- RDF (the output format of some of the DWH services)
- XML, specifically using the TV-Anytime schema (the output format of the Engin services)

(C) Client applications

Below is just a brief description of the use of the above services within the individual client applications. For a more detailed description of the applications, please refer to Section 6 of this deliverable as well as the respective WP7 deliverables.

The 7a Personalised Semantic News (PSN) application (cf. Section 6.1) consists of an EPG Access module that makes a single HTTP GET request to the Broker instead of attempting to integrate with all the individual EPG services.

The 7b application has a requirement for building a Personalised Electronic Programme Guide (cf. Section 6.2). It contains a component called the Semantic-Broker-iFanzzy-Adapter-Interface, which makes a single HTTP GET request to the Broker. The alternative would be to construct multiple interfaces to connect to all the individual services.

In addition, the *SugarTube*¹⁸ application was developed to allow users to search previously annotated videos in a repository, and explore related materials through Linked Data approaches. SugarTube (which stands for Semantics Used to Get Annotated video Recording), allows the user to query and navigate the video search results using the semantic data (e.g. related knowledge, concepts, videos, and learning materials). Just like the 7a PSN application, SugarTube makes use of a single HTTP GET request to the Broker to retrieve EPG data from multiple sources (the EPG retrieval is performed simultaneously with the search for annotated videos). In addition to this, the SugarTube application makes use of Broker goal that exposes functionality for retrieving YouTube trailers that match a set of keywords [17].

(D) Service integration: brokered goals and service orchestration

The final component of the evaluation setting is the service integration (i.e. the Broker goals). In the context of EPG retrieval, we have created the following main goal:

- GET-EPG-BY-KEYWORD-PERIOD-AND-LANGUAGE. This goal (cf. Section 5.1.2 for further details) is meant as a highly generic approach to the task of retrieving EPG data, based on user language. The goal orchestrates two separate pieces of functionality, where each is accomplished by a number of service calls:
 - o Retrieving a list of channels that match a particular language (e.g. Italian, English, German, etc.). This relies on a knowledge model that contains semantic descriptions of the available channels. These semantic descriptions include many attributes of a channel, including the broadcast language of the channel. The Broker reasons over the semantic descriptions to be able to map broadcast language to channel IDs are channel URIs that are recognised by the individual EPG services available from Engin and DWH.
 - o Retrieving enriched EPG data for the list of channels retrieved in the first step. The goal takes as input a language, a particular period for the EPG data (e.g. "19-11-2010 00:00" until "19-11-2010 23:59"), and a particular enrichment source (e.g., "dbpedia" "imdb", or "skos", or "all" of these or "none" of these.). Based on these inputs, this step involves the orchestration of one service call per required channel and, in particular, exploits reasoning on the available SWS to:
 - 1) Select the appropriate Datawarehouse services based on the input parameters (e.g. the *Imdbenrichedperiod* if IMDB enrichments are required)
 - 2) Construct HTTP requests for service invocation (one per channel)
 - 3) Orchestrate all services from (2)
 - 4) Lift results into required output format
- The input roles of the goal are listed below. :
 - o LANGUAGE
 - o KEYWORDS
 - o START
 - o END
 - o ENRICHMENT-SOURCE

(E) Service integration: service annotations via SmartLink

In addition to the service integration via brokered goals, SmartLink (Section 4.1) was used to annotate both the original services as described in (A) and the exposed goals as described in (D), which themselves constitute services as they are invocable via the IRS-III REST API.

7.2.2 Evaluation stakeholders

In this Section, we describe the different target groups and stakeholders involved in the evaluation.

Table 3. Evaluation target groups.

Evaluation Subject	Stakeholder
--------------------	-------------

¹⁸ <http://kmi.open.ac.uk/technologies/name/sugartube> and <http://notube.open.ac.uk/sugartube>

Functional	<ul style="list-style-type: none"> • WP5 Leader <ul style="list-style-type: none"> ○ to identify which services are worth to be brokered and which not ○ to show the seamless replacement of services • Application Scenarios Leaders and Developers <ul style="list-style-type: none"> ○ to ensure that the integration workflow is not influenced by broker internal changes ○ to illustrate the integration of specific WP5 technologies into the applications
Non-functional	<ul style="list-style-type: none"> • WP5 Partners <ul style="list-style-type: none"> ○ to benchmark the semantic engine as a standalone component ○ to benchmark the semantic broker as an integrated component ○ to measure the performance hit within application scenarios ○ to show the behaviour of the system in case of semantic broker's failure • Technology Partners (WP1, WP2, WP3, WP4, WP5, WP6) <ul style="list-style-type: none"> ○ To provide feedback on usability ○ To report the effort impact related to annotating services

7.2.3 Evaluation activities

In this section, the actual evaluation activities are described and structured according to the classification of criteria into (A) functional and (B) non-functional ones.

7.2.3.1 Functional Evaluation Criteria

The functional evaluation criteria will be evaluated based on the evaluation setting described in the previous Section 7.2.1. I.e., based on a single scenario (“EPG Retrieval”), we will assess to what extent each of the functional criteria (Section 7.1.1) has been addressed. In particular, the automation of services tasks in the specific use case will be analysed and it will be assessed and measured, to what extent the implemented functionalities are actually used as part of real-world applications. Here, we will also consider server log files to analyse and evaluate usage statistics.

7.2.3.2 Non-functional evaluation criteria

This section describes some concrete activities to be carried out in order to measure non-functional criteria mentioned in section 7.1.2 namely:

- System performance
- Runtime scalability
- Usability

System Performance

In order to evaluate the broker performance the idea is to run it on top of project-specific orchestration workflows, as commonly requested by WP7.a, b, c by means of goals (setting in Section 7.2.1) , then to benchmark the direct invocation of the same services orchestrated by the Broker, and finally to compare the obtained results.

To achieve that, the following steps have to be performed:

1. Identify one or more service goals for each category of services
2. Identify/Create one testing dataset for each use case.

3. **A)** Develop a Broker client stub (suggestion: Java) for each selected service goal in bullet 1. These modules' aim is to simulate the invocation (thus the integration) of the Broker in a generic NoTube application workflow.
B) According the testing clients developed in the previous bullet, develop another set of client stubs that translate the Semantic Broker orchestration workflow in direct NoTube services invocation.
4. Setup a testing environment in order to perform the performance evaluation
5. Run the client stubs described at bullet 3.A & 3.B one by one. For each stub measure the time elapsed from the triggering of the "run action" to the returned results. This could be easily done via software, within the client stub code
6. Analyse and compare the retrieved performance measurements

Runtime Scalability

The goal of this evaluation is to assess the Semantic Broker performance with different workloads, namely involving an increasing amount of data, services, and therefore service descriptions.

The scalability then could be measured in two ways:

- Data scalability - Running the client stubs described above with differing input data, e.g. querying EPG for large periods of time t and or differing amounts of enrichments e
- Services scalability – Similarly, queries could be carried out involving different numbers of services s .

The same test environment and procedure as for the Performance evaluation be used, but with different invocation parameters in order to measure the impact of the variables described above on the overall performance.

Usability

The usability of the Smartlink environment should be based on users feedback. In particular the users here are the developers adding new services description.

For this reasons the evaluation activities are composed as follows:

1. Setup a questionnaire including:
 - a. Accessibility and usability of the Web tool in terms of Web site navigation
 - b. Understanding of the semantic annotation technology (i.e.: time required to learn it)
 - c. Overhead evaluation in terms of:
 - i. Effort required to annotate a service (man-month)
 - ii. Effort required to add a service in the Smart Link Service repository (man-month)
 - iii. Effort required to update a service in the Smart Link Service repository (man-month)
2. Circulate the questionnaire to WP1, WP2, WP3, WP4, WP5
3. Collect results and derive analytical statistics
4. Enrich with statics about the Web tool usage

8 Conclusion

In this document, we have described the different approaches followed in NoTube to annotate services and APIs for reasoning-based services automation. While currently two different environments and representational approaches are being used – SmartLink/iServe vs IRS-III – our most recent efforts aim at consolidating this infrastructure into a single environment. That will improve maintainability and solves consistency issues which arose due to the maintenance of service models in different representational domains. In addition, while our focus has shifted towards our Linked Services-based approach (iServe/SmartLink) a unified infrastructure will be based entirely on Linked Data principles (RDF, URIs, SPARQL) to be compliant with the current state of the art in Web-scale data sharing, integration and interoperability leading up to many major advantages with respect to exploitation of this work. In order to achieve this vision, most recent and future work is concerned with adding the representational and reasoning support to the iServe/SmartLink tool suite which enable more complex automation (discovery, orchestration, execution) in order to be able to perform brokering tasks.

9 References

- [1] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2008). Dbpedia: A nucleus for a web of open data. In Proceedings of 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC+ASWC 2007), pages 722–735.
- [2] Berners-Lee, T., Hendler, J., Lassila, O (2001). "The Semantic Web". Scientific American Magazine. retrieved March 29, 2009.
- [3] Bizer, C., T. Heath, et al. (2009). "Linked data - The Story So Far." Special Issue on Linked data, International Journal on Semantic Web and Information Systems (IJSWIS).
- [4] Cabral, L., Domingue, J., Galizia, S., Gugliotta, A., Norton, B., Tanasescu, V., Pedrinaci, C. (2006): IRS-III: A Broker for Semantic Web Services based Applications. Proceedings of the 5th International Semantic Web Conference (ISWC), Athens, USA.
- [5] Davies, J., Domingue, J., Pedrinaci, C., Fensel, D., Gonzalez-Cabero, R., Potter, M., Richardson, M., and Stincic, S. (2009). Towards the open service web. BT Technology Journal, 26(2).
- [6] Dietze, S., Benn, N., Domingue, J., Conconi, A., and Cattaneo, F.: "Interoperable Multimedia Metadata through Similarity-based Semantic Web Service Discovery". In Proceedings of 4th International Conference on Semantic and Digital Media Technologies (SAMT '09), 2--4 December 2009, Graz, Austria.
- [7] Dietze, S., Benn, N., Domingue, J., Conconi, A., and Cattaneo, F. (2009) Two-Fold Semantic Web Service Matchmaking – Applying Ontology Mapping for Service Discovery, 4th Asian Semantic Web Conference, Shanghai, China.
- [8] Dietze, S., Gugliotta, A., Domingue, J., (2007) A Semantic Web Service oriented Framework for adaptive Learning Environments. European Semantic Web Conference (ESWC) 2007, Innsbruck, Austria, June 2007.
- [9] Dietze, S., Gugliotta, A., Domingue, J., (2008) Conceptual Situation Spaces for Situation-Driven Processes. 5th European Semantic Web Conference (ESWC), Tenerife, Spain, June 2008.
- [10] Domingue, J., Cabral, L., Galizia, S., Tanasescu, V., Gugliotta, A., Norton, B., Pedrinaci, C.: "IRS-III: A broker-based approach to Semantic Web Services", Journal of Web Semant, pp. 109-132. Elsevier Science Publishers B. V, 2008.
- [11] Dietze, S., Yu, H.Q., Pedrinaci, C., Liu, D. and Domingue, J. (2011) SmartLink: a Web-based editor and search environment for Linked Services, 8th Extended Semantic Web Conference (ESWC), Heraklion, Greece
- [12] Farrell, J., and Lausen, H. 2007. Semantic Annotations for WSDL and XML Schema. <http://www.w3.org/TR/sawsdl/>. W3C Candidate Recommendation 26 January 2007.
- [13] Fensel, D.; Lausen, H.; Polleres, A.; de Bruijn, J.; Stollberg, M.; Roman, D.; and Domingue, J. 2007. Enabling Semantic Web Services: The Web Service Modelling Ontology. Springer.
- [14] Hepp, M., Leymann, F., Domingue, J., Wahler, A. and Fensel, D.: Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management. Proceedings of IEEE Intl. Conf. on e-Business Engineering (ICEBE 2005) pp. 535-540. Beijing, China (2005).
- [15] Kopecky, J.; Vitvar, T.; and Gomadam, K. 2008. MicroWSMO. Deliverable, Conceptual Models for Services Working Group, URL: http://cms-wg.sti2.org/TR/d12/v0.1/20090310/d12v01_20090310.pdf.
- [16] Lambert, D., and Domingue, J. (2008) Grounding semantic web services with rules, Workshop: Semantic Web Applications and Perspectives, Rome, Italy

- [17] Lambert, D. and Yu, H.Q. (2010) Linked Data Based Video Annotation and Browsing for Distance Learning, Workshop: SEMHE '10: THE SECOND INTERNATIONAL WORKSHOP ON SEMANTIC WEB APPLICATIONS IN HIGHER EDUCATION, Southampton, UK
- [18] Maleshkova, M., Pedrinaci, C., and Domingue, J. (2009). Supporting the creation of semantic restful service descriptions. In Workshop: Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2) at 8th International Semantic Web Conference.
- [19] Maleshkova, M., Kopecky, J., and Pedrinaci, C. (2009). Adapting SAWSDL for semantic annotations of restful services. In Workshop: Beyond SAWSDL at OnTheMove Federated Conferences & Workshops.
- [20] Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., and Sycara, K. (2004). OWL-S: Semantic Markup for Web Services. Member submission, W3C. W3C Member Submission 22 November 2004.
- [21] Motta, E., Reusable Components For Knowledge Modelling: Case Studies in Parametric Design Problem Solving. IOS Press, ISBN I 58603 003 5, 1999.
- [22] OWL-S Coalition: OWL-S 1.1 release. (2004). <http://www.daml.org/services/owl-s/1.1/>
- [23] Pedrinaci, C., Domingue, J., and Reto Krummenacher (2010) Services and the Web of Data: An Unexploited Symbiosis, Workshop: Linked AI: AAAI Spring Symposium "Linked data Meets Artificial Intelligence".
- [24] Pedrinaci, C., Liu, D., Maleshkova, M., Lambert, D., Kopecky, J., and Domingue, J. (2010) iServe: a Linked Services Publishing Platform, Workshop: Ontology Repositories and Editors for the Semantic Web at 7th Extended Semantic Web Conference.
- [25] Sheth, A. P., Gomadam, K., and Ranabahu, A. (2008). Semantics enhanced services: Meteor-s, SAWSDL and SA-REST. IEEE Data Eng. Bul 1., 31(3):8–12.
- [26] Vitvar, T.; Kopecky, J.; Viskova, J.; and Fensel, D. 2008. Wsmo-lite annotations for web services. In Hauswirth, M.; Koubarakis, M.; and Bechhofer, S., eds., Proceedings of the 5th European Semantic Web Conference, LNCS. Berlin, Heidelberg: Springer Verlag.
- [27] WSMO Working Group (2004), D2v1.0: Web service Modelling Ontology (WSMO). WSMO Working Draft, (2004). (<http://www.wsmo.org/2004/d2/v1.0/>).
- [28] WSMO Working Group (2004), D2v1.0: Web service Modelling Ontology (WSMO). WSMO Working Draft, (2004). (<http://www.wsmo.org/2004/d2/v1.0/>).
- [29] Wagner, M., Kellerer, W. 2004. Web services selection for distributed composition of multimedia content, Proceedings of the 12th annual ACM international conference on Multimedia, October 10-16, 2004, New York, NY, USA.
- [30] World Wide Web Consortium, W3C: Simple Object Access Protocol, SOAP, Version 1.2 Part 0: Primer, (2003). (<http://www.w3.org/TR/soap12-part0/>).
- [31] World Wide Web Consortium, W3C: Universal Description, Discovery and Integration: UDDI Spec Technical Committee Specification v. 3.0, (2003). <http://uddi.org/pubs/uddi-v3.0.1-20031014.htm>.
- [32] World Wide Web Consortium, W3C: WSDL: Web services Description Language (WSDL) 1.1, (2001). (<http://www.w3.org/TR/2001/NOTE-wsdl-20010315>)