



Demos and Posters REWERSE Review Meeting Year 4 23-24 January 2008

1	Overview Demos REWERSE Review Meeting Year 4.....	2
1.1	WG I1 – Rule Markup Languages	2
1.1.1	The New Eclipse-Based Version of the REWERSE Rule Modeling Tool Strelka.....	2
1.1.2	Using the REWERSE Rule Modeling, Markup and Translation Technologies in a Financial Services Use Case.....	2
1.1.3	ERDF Tool Demo	2
1.2	WG I2 – Policy enforcement composition and conformance	4
1.2.1	Advanced Policy Reasoning, Negotiations and Explanations on the Web.....	4
1.2.2	APE – Attempto Parsing Engine.....	4
1.2.3	AceWiki	4
1.2.4	ACE View	4
1.3	WG I3 – Composition and Typing.....	5
1.3.1	Reuseware Composition Framework-Introduction	5
1.3.2	Reuseware Application 1: Modules for Xcerpt (and Datalog).....	5
1.3.3	Reuseware Application 2: Role modeling with Ontologies.....	5
1.3.4	SemViz – A prototypical visualization environment for ontologies.....	5
1.4	WG I4 – Reasoning-Aware Querying.....	6
1.4.1	Reasoning-aware Querying by Example—An Overview of I4's Work through Demonstrations and Prototypes	6
1.4.2	Xcerpt—Language Evolution towards Versatility.....	6
1.4.2.1	GRDDL Use Case	6
1.4.2.2	Through a Looking Glass: Xcerpt's Future and the MusicBrainz Use Case (Poster)	7
1.4.2.3	RDFLog—Filling in the Blanks in RDF Querying (Poster)	7
1.4.3	Many Languages, one Evaluation: The CIQLog Calculus	7
1.4.4	Scaling beyond Trees: The CIQCAG Algebra (Poster).....	8
1.5	WG I5 – Evolution and Reactivity	9
1.5.1	r3-based PubMed Reactive Classifier	9
1.6	WG A1 – Web-based Decision Support for Event, Temporal and Geographical Data	10
1.6.1	Visualisation of the Local Data Stream Management System.....	10
1.6.2	Semantic Overview and Navigation on a Document Repository—an Application of the EFGT Net	10
1.7	WG A2 – Towards a Bioinformatics Semantic Web	11
1.7.1	GoPubMed and beyond.....	11
1.8	WP A3 – Personalised Information Systems	12
1.8.1	AC4RDF - Access control for RDF databases.....	12
1.8.2	UMService - An RDF based user modeling service	12

1 Overview Demos REVERSE Review Meeting Year 4

1.1 WG I1 – Rule Markup Languages

1.1.1 The New Eclipse-Based Version of the REVERSE Rule Modeling Tool Strelka

The REVERSE I1 rule modeling tool [Strelka](#), is now available on the [Eclipse](#) platform. Strelka allows to express rules visually within a UML class diagram that defines the underlying vocabulary. It also provides the option to externalize the rule model in the form of R2ML markup. The new version of the tool has a number of improvements, including the use of a fast OCL parser for processing URML filter expressions, the new Eclipse-based interface and the integration with [I1 rule interchange services](#).

Presented by Adrian Giurca and Gerd Wagner.

1.1.2 Using the REVERSE Rule Modeling, Markup and Translation Technologies in a Financial Services Use Case

This advanced (rule-based) Web application implements the [UServ Product Derby Case Study](#) with the help of REVERSE I1 technologies.

To obtain the rules in the executable languages *JenaRules* and *JBoss Rules*, starting from the natural language representation, the following steps are performed :

1. Create an object-oriented model of the use case vocabulary in the form of a UML class diagram.
2. Express each rule on top of the vocabulary model in the form of a visual [URML](#) rule model with arrows and suitable annotations, using the Eclipse-based rule modeling tool [Strelka](#).
3. Using Strelka, generate the [R2ML](#) markup for the rules (or, in other words, externalize the rules in an XML format).
4. Use the [R2ML to Jena](#) and the [R2ML to JBossRules](#) translators for obtaining the executable rule code for JenaRules and JBoss Rules from the R2ML markup.

The application uses the [Jena Java API](#) and the [Drools Java API](#) for applying the rules to the initial facts, which are collected from the user with the help of a HTML form. A set of default facts can be loaded by using the 'Auto Complete' push button.

The [JenaRules version](#) and the [JBoss Rules version](#) of the application are available online for tests. A [use case browser](#) is also available online.

Presented by Adrian Giurca and Gerd Wagner.

1.1.3 ERDF Tool Demo

The REVERSE I1 [ERDF Tool](#) allows executing [Jena](#) and [ERDF](#) rules. The atoms in the condition and in the conclusion of a rule are either *triple patterns* or they are formed with built-in predicates. In both cases they may contain variables.

ERDF rules extend the syntax of Jena rules by adding two new operators for expressing the two types of negation, namely `strong negation` and `negation-as-failure`. Strong nega-

tion is expressed by the minus symbol in front of the property, i.e. `(?x -rdf:type ex:MediumStudent)`. Weak negation is expressed in the form of a built-in with three arguments forming a triple pattern, e.g. `naf(?x rdf:type ex:MediumStudent)`. We can also express the weak negation of a negative triple, i.e. `naf(?x -rdf:type ex:MediumStudent)`.

In order to reason with ERDF facts and rules, the Jena API was extended in a suitable way. A new inference engine, the `ERDFReasoner`, which is based on the Jena RDFS reasoner, has been developed in order to deal with these new kinds of facts and rules.

Presented by Adrian Giurca and Gerd Wagner.

1.2 WG I2 – Policy enforcement composition and conformance

1.2.1 Advanced Policy Reasoning, Negotiations and Explanations on the Web

Trust negotiation protocols have emerged as a solution for open environments such as the Web, in which parties may make connections and interact without being previously known to each other. This demo will present a policy driven access control framework for the Web that not only provides advanced policy reasoning mechanisms and explanations for static resources but also policy-based personalised generation of content. This approach separates web presentation from the security and application logic, integrates a flexible and expressive policy language, enables (possibly automated) interactions with human and software agents, and boosts user awareness and cooperative enforcement of such policies.

Presented by Daniel Olmedilla.

1.2.2 APE – Attempto Parsing Engine

Latest version of the ACE 6 parser demonstrating new language features of ACE 6.

1.2.3 AceWiki

Semantic Wiki using a predictive editor for ACE 6.

1.2.4 ACE View

ACE 6 as input and output language of the ontology editor Protégé.

All ACE demos presented by Norbert E. Fuchs.

1.3 WG I3 – Composition and Typing

1.3.1 Reuseware Composition Framework-Introduction

The objective of the Reuseware Composition Framework is to extend languages lacking component-oriented constructs with such possibilities. Many domain-specific languages (DSLs) are usually initially designed without such constructs, which also holds true for many of the DSLs developed for use on the Web and for the Semantic Web. While the obvious and most straight-forward solution to the problem is to include such capabilities for each language, when they are designed, this is not always possible, not least due to cost reasons. Furthermore, a language-egoistic approach to language extensions does not enable reuse of such extensions across different languages. For these reasons, Reuseware supports a language-inclusive approach where any language can be extending according to the methodology encapsulated in the framework.

Presented by Jakob Henriksson.

1.3.2 Reuseware Application 1: Modules for Xcerpt (and Datalog)

We demonstrate how 'modules' can be integrated into Xcerpt such that programmers are able to design and program applications in a modular fashion. We present updated Modular Xcerpt syntax and examples. Furthermore, we show how the same notion of 'modules' can be integrated into Datalog in a very similar manner (by reusing the module-extension from Xcerpt).

Presented by Jakob Henriksson.

1.3.3 Reuseware Application 2: Role modeling with Ontologies

We have investigated an interesting reuse unit for ontologies: role models. Role modeling has been investigated in data, conceptual and object-oriented modeling, but never been fully investigated for today's ontology languages. We demonstrate how 'role models' can be specified, reused across different ontologies and composed in Reuseware.

Presented by Jakob Henriksson.

1.3.4 SemViz – A prototypical visualization environment for ontologies

SemViz is a model-driven architecture for flexible visualization of domain ontologies. Based on a mapping from the domain ontology to Fresnel, an OWL-based visualization language (stage 1), a second stage transforms the platform-independent visualization into a platform-dependent one (X3D, svg, pdf, stage 2). Because also filtering is integrated, SemViz provides a flexible way to visualize ontologies. The demo shows the visualization of an ontology for technical history of the 20th century.

Self-presenting demo.

Additionally, three posters will be presented.

1.4 WG I4 – Reasoning-Aware Querying

1.4.1 Reasoning-aware Querying by Example—An Overview of I4's Work through Demonstrations and Prototypes

Over the last four years, the I4 working group of REWERE has developed a number of reasoning-aware query languages around its vision of versatile querying: Languages that are able to access data in any of the emerging Web formats such as XML, RDF, Topic Maps, etc. In this demonstration, we illustrate this vision along the three languages proposed by I4: (1) Xcerpt is the first versatile Web query language that allows seamless access to both XML and RDF. (2) dlhex showcases the integration of traditional query languages with external reasoners and data sources such as RDF, ontology, or theory reasoners. (3) RDFLog is a test-bed for investigating expressiveness and complexity of RDF query languages with particular emphasis on blank nodes (or existential information in facts and rule heads). The three demonstrations are all accessible from the I4 demonstrator Website without additional software installation. We provide example queries and data, tutorials, and/or screen casts for each demonstrator to allow for easy experimentation.

Presented by Tim Furche

1.4.2 Xcerpt—Language Evolution towards Versatility

1.4.2.1 GRDDL Use Case

In the last years, the Semantic Web has significantly gained momentum, and the amount of RDF data on the Web has been increasing exponentially ever since the publication of the RDF recommendation. However, a great amount of such semantic data is intermingled with HTML and XML through the help of microformats such as hCalendar, embedded RDF, and RDF/A.

To deal with this situation, the W3C proposes the GRDDL (Gleaning Resource Descriptions From Dialects of Languages) framework for the extraction of Semantic Web information from HTML and XML documents together with a collection of descriptions of use-cases as a motivation for employing the GRDDL method. One of these use-cases is the scheduling of a meeting between friends who publish their calendars either as hcalendar, embedded RDF, RDFa or RSS 1.0 on their homepages. In a first step, an RDF representation of the calendar information is harvested from the respective homepages, and the resulting RDF graphs are combined in a single RDF model. In a second step, the RDF data is queried to find a date for the meeting that fits in everybody's schedule.

Our system implements this use case in two different manners. The first implementation follows the recommendation of the GRDDL standard, employing an XSLT processor and a SPARQL implementation (in our case Jena). The second implementation uses Xcerpt for both processing stages.

The presented system constitutes the first complete implementation of a GRDDL use-case and allows to draw the following conclusions: (1) Extracting RDF information from microformats is a non-trivial task and calls for expressive and user-friendly query languages specifically aimed at querying heterogeneous XML data. (2) For usability as well as efficiency purposes it is desirable to have a language that is both capable of extracting the relevant information and of further semantic processing. (3) Although SPARQL is a very well-specified and expected to become the most widely used RDF query language, it lacks some features---, most notably grouping which limit its use in our examples. In the RDFLog demonstration, it is discussed how SPARQL can be extended with more expressive grouping constructs without increase in

query complexity. Currently, these limitations of SPARQL queries mean that it must be embedded in a more powerful general purpose programming language to solve all the GRDDL use cases. (4) While Xcerpt is still a research prototype, it already shows that versatile, pattern-oriented and rule-based querying has the potential to considerably ease the authoring of data intensive web-applications.

Presented by Benedikt Linse

1.4.2.2 Through a Looking Glass: Xcerpt's Future and the MusicBrainz Use Case (Poster)

The web-based MusicBrainz database contains information about 300.000 artists, 400.000 albums and 6 million tracks in RDF format. It is commonly used by desktop applications to find information about music files on a local hard drive. We demonstrate an extension of this scenario in which up-to-date news about a song's artists is retrieved from XML news feeds and displayed within an HTML file, while the user is listening to the song. This task can be achieved in a particularly convenient manner using a versatile query language such as Xcerpt.

It also illustrates one of the directions for future research on Xcerpt: increasing ease and efficiency of RDF querying with Xcerpt with particular emphasize on chaining and specificities of RDF such as blank nodes, see 1.4.2.3 RDFLog—Filling in the Blanks in RDF Querying (Poster)

Presented by Benedikt Linse

1.4.2.3 RDFLog—Filling in the Blanks in RDF Querying (Poster)

Since the Resource Description Framework (RDF) has emerged as the basic format of choice for representing Semantic Web data, query languages for RDF such as SPARQL, RQL, and Versa have been developed. Although blank nodes are a prominent feature of RDF, these query languages have, until now, largely ignored the need for sophisticated blank node support.

In this article we analyze the limitations of existing RDF query languages with respect to blank nodes by studying a formal, rule-based query language called RDFLog. We define the operational semantics of RDFLog by a LOGSPACE transformation of RDFLog programs to logic programs and back to RDF graphs, using Skolemisation and a novel form of un-Skolemisation. This yields a generic and closed operational semantics. Moreover, a declarative semantics is provided, and the operational semantics is shown to be sound and complete with respect to the declarative semantics. For several fragments of RDFLog, we study complexity and expressiveness. In addition, we provide complexity results for SPARQL and show how to increase SPARQLs expressiveness without affecting its complexity.

Demo see 1.4.1 Reasoning-aware Querying by Example.

Presented by Benedikt Linse

1.4.3 Many Languages, one Evaluation: The CIQLog Calculus

Unquestionably, versatile Web query languages are convenient from a user's perspective. However, there already exist a large number of query languages for the Web and considerable investment in these languages. Therefore, we have developed a formal foundation that fits all major Web query languages, versatile or format specific, from XPath over XQuery and Xcerpt to SPARQL. This formal foundation, called CIQLog, is a variant of datalog with negation and value invention tailored specifically to Web query languages and their data

models. In this demonstration, we show how to translate XPath, XQuery, Xcerpt, and SPARQL to CIQLog and illustrate that translation with the prototype compiler.

CIQLog together with these translations yield

- (1) a purely logical semantics for XPath, XQuery, Xcerpt, and SPARQL. For XQuery and SPARQL, this is the first purely logical semantics to the best of our knowledge.
- (2) a better understanding of commonalities and differences between these languages. In particular, the step from XPath to (composition-free) XQuery illustrates how only a small number of additional features dramatically affects the semantics and evaluation of XQuery.
- (3) Together with the CIQCAG algebra demonstrated in the last demonstration, (a) a space optimal implementation of navigational XPath with space complexity $O(q \cdot d)$ and time complexity $O(q \cdot n)$ where q is the query, n the data size, and d the depth of the tree data; (b) the first implementation for SPARQL with polynomial-time complexity for tree queries on arbitrary graphs and linear complexity on tree and certain graph data; (c) efficient implementations for Xcerpt and XQuery that scale over different data and query shapes, i.e., that provide on each restricted class time and space complexity rivaling the best known approaches limited to that class.
- (4) a foundation for the integration of queries from several of these languages.

Presented by Antonius Weinzierl

1.4.4 Scaling beyond Trees: The CIQCAG Algebra (Poster)

Versatile Web query languages such as Xcerpt may, at first glance, be at a disadvantage compared to specialized languages such as XQuery or SPARQL when we consider evaluation complexity and performance. After all, they have to cope with data in different formats. However, in this work we propose a versatile evaluation of all major Web query languages that scales over different shapes of queries and data. More precisely, the CIQCAG algebra, proposed for the versatile evaluation of Web queries formulated in XPath, XQuery, Xcerpt, or SPARQL, provides the best known time and space complexities for tree queries and tree data, but is capable to process general graph queries on graph data. Furthermore, we identify a novel class of graphs, where tree queries can be evaluated as efficiently as on tree data, but that covers many practical data sets, in particular RDF data sets, that are not tree-shaped.

Presented by Tim Furche

1.5 WG I5 – Evolution and Reactivity

1.5.1 r3-based PubMed Reactive Classifier

PubMed, <http://www.pubmed.gov/>, the main biomedical literature database references over 12.000.000 entries. Based on events signalling additions to PubMed, this use-case generates events detecting the relevant additions (filtered) according to a set of (so called) relevant terms. The relevant terms and publications are maintained in a Prova rulebase, based on the occurrence of events stating new publications added to PubMed and new or no longer relevant terms. A publication is deemed relevant if it contains, according to PubMed eSearch utility, any of the supplied relevant terms. For each relevant publication information is retrieved (and mirrored) for inclusion in the generated events. Additionally events are also generated signalling any publication that becomes irrelevant due to the removal of a relevant term.

Presented by José Alferes.

1.6 WG A1 – Web-based Decision Support for Event, Temporal and Geographical Data

1.6.1 Visualisation of the Local Data Stream Management System

L-DSMS is a flexible framework for managing and processing any kind of data streams. This demo gives a brief example on how L-DSMS is used to transform incoming binary data (TMC messages) into KML (an XML format) that can be displayed by Google Earth. Furthermore we will give you an introduction on how to manage a running instance of L-DSMS using the graphical management tool VISU-L-DSMS.

Presented by Christian Hänsel and Edgar Stoffel.

1.6.2 Semantic Overview and Navigation on a Document Repository—an Application of the EFGT Net

The demo shows semantic aided browsing by means of the EFGT Net which contains temporal, geographic and thematic Topics and Entities. A thematic overview is provided over large document repositories, in this case a digitized newspaper archive.

Presented by Levin Brunner.

1.7 WG A2 – Towards a Bioinformatics Semantic Web

1.7.1 GoPubMed and beyond

The demo will comprise the latest release of GoPubMed with a number of new features: Filtering by categories what, who, where, when; novel ranking of top terms; author profiles for millions of authors. The key to GoPubMed are ontologies. We have developed an editor, which supports semi-automated generation of ontology terms. GoPubMed is a flexible, open system and we have developed from it a semantic Wiki called TransWiki. It allows users to add, delete, and edit articles. Editing comprises auto-completion of terms from the ontology and mapping of articles into the ontology afterwards. TransWiki has been adapted to realise a prototypical Job search engine.

Presented by Michael Schroeder

1.8 WP A3 – Personalised Information Systems

1.8.1 AC4RDF - Access control for RDF databases

Semantic Web databases allow efficient storage and access to RDF statements. Applications are able to use expressive query languages in order to retrieve relevant metadata to perform different tasks. However, access to metadata may not be public to just any application or service. Instead, powerful and flexible mechanisms for protecting sets of RDF statements are required for many Semantic Web applications. Unfortunately, current RDF stores do not provide fine-grained protection. This approach fills this gap and presents a mechanism by which complex and expressive policies can be specified in order to protect access to metadata in multi-service environments.

Presented by Daniel Krause

1.8.2 UMService - An RDF based user modeling service

For the usage and maintenance of user profiles in decentralized Web Service based framework, a so-called user modeling service was created. This service provides an RDF storage format to store information and meta-information about users. Furthermore, this service implements a simple query language to extract information easily from a user profile. A web-based user interface enables users to review and modify their user profiles.

Presented by Daniel Krause