Mashing up user data in the Grapple User Modeling Framework

Fabian Abel[1], **Dominikus Heckmann**[2], **Eelco Herder**[1], **Jan Hidders**[3], **Geert-Jan Houben**[3], **Daniel Krause**[1], **Erwin Leonardi**[3], **Kees van der Slujis**[4]
[1] L3S Research Center, Hannover, Germany, {abel,herder,krause}@L3S.de
[2] DFKI GmbH, Saarbrücken, Germany, heckmann@dfki.de
[3] WIS, TU Delft, The Netherlands{a.j.h.hidders,g.j.p.m.houben,e.leonardi}@tudelft.nl
[4] CS Department, Eindhoven University of Technology, The Netherlands, k.a.m.sluijs@tue.nl

Abstract

In this paper we demonstrate the Grapple User Modeling Framework (GUMF), which exploits Semantic Web technologies and Web 2.0 paradigms to model users across different applications and domains. It introduces novel features such as *dataspaces*, which logically bundle user data, and *user pipes*, which allow to mash up user data from different sources.

1 Introduction

Web systems such as Amazon or YouTube brought personalization to the general public. While those popular systems base recommendations on a large amount of data - by means of collaborative filtering and social network analysis - [Frias-Martinez *et al.*, 2005], the majority of Web applications cannot build upon a big user population and users might not interact regularly with these systems. A promising approach to compensate for this lack of data is *cross-application user modeling* [Korth and Plumbaum, 2007]. In contrast to the centralized approach of *generic user modeling servers* [Abel *et al.*, 2008], a *conversion-based* approach allows for flexible mappings. These mappings can be created from one system to another, or by making use of generalized representations, such as the General User Model Ontology (GUMO) [Heckmann *et al.*, 2005] and UserRDF [Abel *et al.*, 2008].

In this paper we present the architecture and implementation of the Grapple User Modeling Framework (GUMF) [Abel *et al.*, 2009a], which re-uses, refines and enhances previous work in the area of cross-application and generic user modeling systems. GUMF[1] organizes user profile data in *dataspaces*, which constitute *views* on a specific set of data. Dataspaces are extensible with plug-ins for mapping and integrating data from external data sources; these plug-ins can also be used to reason with existing to deduce further knowledge about the user. In addition to traditional rule-based approaches, GUMF provides so-called *user pipes* [Abel *et al.*, 2009a] that mash up different (user profile) data streams, formatted in RDF or RSS, making use of Semantic Web Pipes[2] or Yahoo Pipes[3].

[1]Currently available at: `http://semweb.kbs.uni-hannover.de:8082/grapple-umf/`
[2]http://pipes.deri.org
[3]http://pipes.yahoo.com

2 GUMF: Grapple User Modeling Framework

Figure 1 shows the architecture of GUMF. The elements at the top provide the essential, generic functionality of the framework; elements part at the bottom right provide generic as well as domain-specific *reasoning logic*.

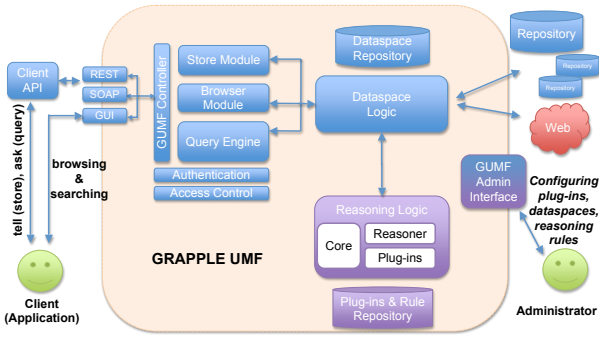

Figure 1: Architecture of the Grapple User Modeling Framework.

Client applications can access GUMF either via a RESTful or SOAP-based API. Further, there is a *Java Client API* that facilitates development of GUMF client applications. Client applications mainly approach GUMF to store user information (handled by the *Store Module*) or to query for information (handled by *Query Engine*). User profile information is modeled by Grapple statements [Abel *et al.*, 2009b], which are basically reified RDF statements about a user, enriched with provenance metadata. GUMF currently supports SPARQL and SeRQL queries as well as a pattern-based query language that exploits the Grapple statement structure to specify what kind of statements should be returned by GUMF. Authorized client requests are answered by GUMF's *Dataspace Logic*. Dataspaces are equipped with data storage repositories that either reside at the GUMF server or are distributed across the Web (possibly maintained by the client application itself), and with (reasoning) plug-ins that further enrich the data that is available in the repositories.

The *Administrator* of a GUMF client application can configure dataspaces and plug-ins via the *GUMF Admin Interface*. Activating or deactivating plug-ins directly influences the behavior of dataspaces. Further, administrators can adjust the plug-ins and reasoning rules to their needs. For example, we developed a plug-in that gathers user profile information from Facebook and maps—with support of

Silk[4]—the profile to a format preferred by the client application administrator (e.g., FOAF[5] or OpenSocial[6]).

Inspired by Web 2.0 practices, a key principle of GUMF is that dataspaces can be shared across different client applications. Therefore, clients can *subscribe* to other dataspaces, given that they are granted approval by the administrator of the dataspace. When subscribed to a dataspace, the client is allowed to query it. However, it might still not be allowed to access all statements that are made available via the dataspace, as fine-grained access control functionality can be embedded in the dataspaces as well.

3 Demonstration Overview

In our demonstration we primarily show how client applications can benefit from the Grapple User Modeling Framework.

Developers of client applications first have to register their application at GUMF. Upon registration, a dataspace is generated that can immediately be used to store user profile information. As an example, a client might store user interests such as *"Peter is interested in Darmstadt"* in GUMF, by using the RESTful API and the Java Client implementation. GUMF models such information as Grapple statement.

```
<gc:Statement rdf:about="&ds10;6357701291243375806816">
  <gc:subject rdf:resource="&guser;peter"/>
  <gc:predicate rdf:resource="&foaf;interest"/>
  <gc:object rdf:resource="&dbpedia;Darmstadt"/>
  <gc:level rdf:datatype="&xsd;double">0.7</gc:level>
  <gc:origin>[peter(Interest: Darmstadt, 0.7]</gc:origin>
  <gc:created rdf:datatype="&xsd;dateTime">
    2009-05-27T00:10:06.817+02:00</gc:created>
  <gc:creator rdf:resource="&gclient;10"/>
</gc:Statement>
```

The core part of the Grapple statement consists of a subject-predicate-object triple, possibly extended with `gc:level` that describes to which degree the statement is true. In addition, the client can store the information in its original format (`gc:origin`). GUMF enriches the statement with metadata such as a globally unique ID, a timestamp (`gc:created`, which is a subproperty of `dc:created`), or the client (or plug-in) that created the statement (`gc:creator`, which is a sub-property of `dc:creator`).

Figure 2 shows the administration interface of GUMF, in particular the configuration of the dataspaces. Administrators can add plug-ins to a dataspace (cf. *"add plug-in"*) and adjust which client applications are allowed to access the dataspace (cf. *"Subscriptions"*). Via GUMF's RESTful API, client applications send advanced SPARQL queries or queries based on simple patterns. As an example, `../ds/13/predicate/interest` would return all Grapple statements in the dataspace `../ds/13` on user interests. The output format of a query can be selected as well. At the moment, GUMF supports RDF/XML, RSS 2.0 and SPARQL Query Results XML format.

In our demonstration at ABIS, we will show how GUMF is applied to mash up and reason with user profile information from different tagging and social networking systems (Flickr, Facebook, TagMe![7], and GroupMe![8]).

[4]http://www4.wiwiss.fu-berlin.de/bizer/silk/

[5]http://xmlns.com/foaf/spec/

[6]http://web-semantics.org/ns/opensocial/

[7]http://tagme.groupme.org

[8]http://groupme.org

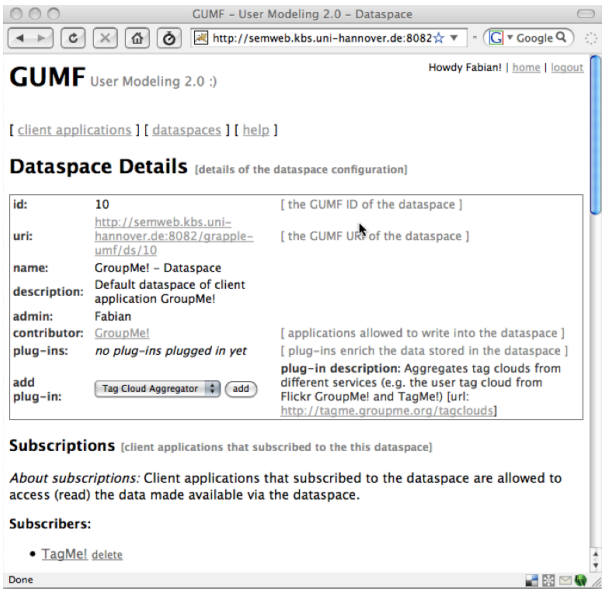

Figure 2: Configuration of Dataspaces and Plug-Ins in the GUMF Web Application.

References

[Abel *et al.*, 2009a] F. Abel, D. Heckmann, E. Herder, J. Hidders, G.-J. Houben, D. Krause, E. Leonardi, and K. van der Slujis. A Framework for Flexible User Profile Mashups. In Proc. of *Int. Workshop on Adaptation and Personalization for Web 2.0 at UMAP '09*, Trento, Italy, 2009.

[Abel *et al.*, 2009b] F. Abel, D. Heckmann, E. Herder, J. Hidders, G.-J. Houben, D. Krause, E. Leonardi, and K. van der Slujis. Definition of an appropriate User profile format. Technical Report D2.1, Grapple project, March 2009.

[Abel *et al.*, 2008] F. Abel, N. Henze, D. Krause, and D. Plappert. User modeling and user profile exchange for Semantic Web applications. In Proc. of *Workshop on Adaptivity and User Modeling in Interactive Systems*, Wuerzburg, Germany, 2008.

[Frias-Martinez *et al.*, 2005] E. Frias-Martinez, G. Magoulas, S. Chen, and R. Macredie. Modeling human behavior in user-adaptive systems: Recent advances using soft computing techniques. *Expert Systems with Applications* **29**:320–229, 2005.

[Heckmann *et al.*, 2005] D. Heckmann, T. Schwartz, B. Brandherm, M. Schmitz, and M. von Wilamowitz-Moellendorff. GUMO - The General User Model Ontology. In Proc. of *Int. Conf. on User Modeling*, Edinburgh, UK, 428–432, 2005.

[Kobsa *et al.*, 2001] A. Kobsa, J. Koenemann, and W. Pohl. Personalized hypermedia presentation techniques for improving customer relationships. *The Knowledge Engineering Review* **16 (2)**:111–155, 2001.

[Korth and Plumbaum, 2007] A. Korth and T. Plumbaum. A framework for ubiquitous user modeling. In Proc. of *Int. Conf. on Information Reuse and Integration*, 2007.