# RDFHomepage
# or
# "Finally, a use for your FOAF file"

Gunnar AAstrand Grimnes, Sven Schwarz, and Leo Sauermann

Knowledge Management
DFKI GmbH
Kaiserslautern, Germany
http://www.dfki.uni-kl.de/∼{grimnes,schwarz,sauermann}
{grimnes,schwarz,sauermann}@dfki.uni-kl.de

**Abstract.** This paper presents the RDFHomepage project, a framework for using a person's structured data sources to auto-generate an HTML homepage. RDFHomepage uses RDF files as input, and currently supports several well-known RDF schemas, such as FOAF. In addition to these we have RDF converters for other structured file-formats, like Bibtex. RDFHomepage produces valid HTML 4.01 Transitional pages, and makes it easy to roll-out functional homepages for a group of people. The generated HTML code is very general, allowing quick and easy page-redesigning using CSS. RDFHomepage is written in PHP and uses our system for generating PHP classes based on RDF class definitions, enabling quick and easy development of RDF handling PHP code.

## 1 Introduction

RDFHomepage is a tool for automatic generation of HTML homepages based on RDF files and other structured information sources which a user might already create and maintain. Figure 1 shows an overview screenshot of the default homepage created based on a user's RDF data. This page contains all things one expects on a typical homepage: the top shows the person's name, email, telephone number etc., this is taken from his FOAF profile; further down there is a short bibliography, this taken from a file using the *homepage-schema* we created for this task; the next section lists the projects this user is involved in, taken from the DFKI Organisational Repository (OrgRep); finally the page has a list of people he knows, based on the friends he specified in his FOAF profile. RDFHomepage has the following attractive features:

- Generated pages are valid HTML 4.01 Transitional.
- Generates well structured HTML code that can easily be styled with Cascading Stylesheets (CSS).
- Enables complete separation of content and appearance.
- An RDF Template engine for generating PHP classes for each RDF Schema is used to make the PHP code easy to write and maintain.

- Enables easy rollout of many identical websites across an organisation.

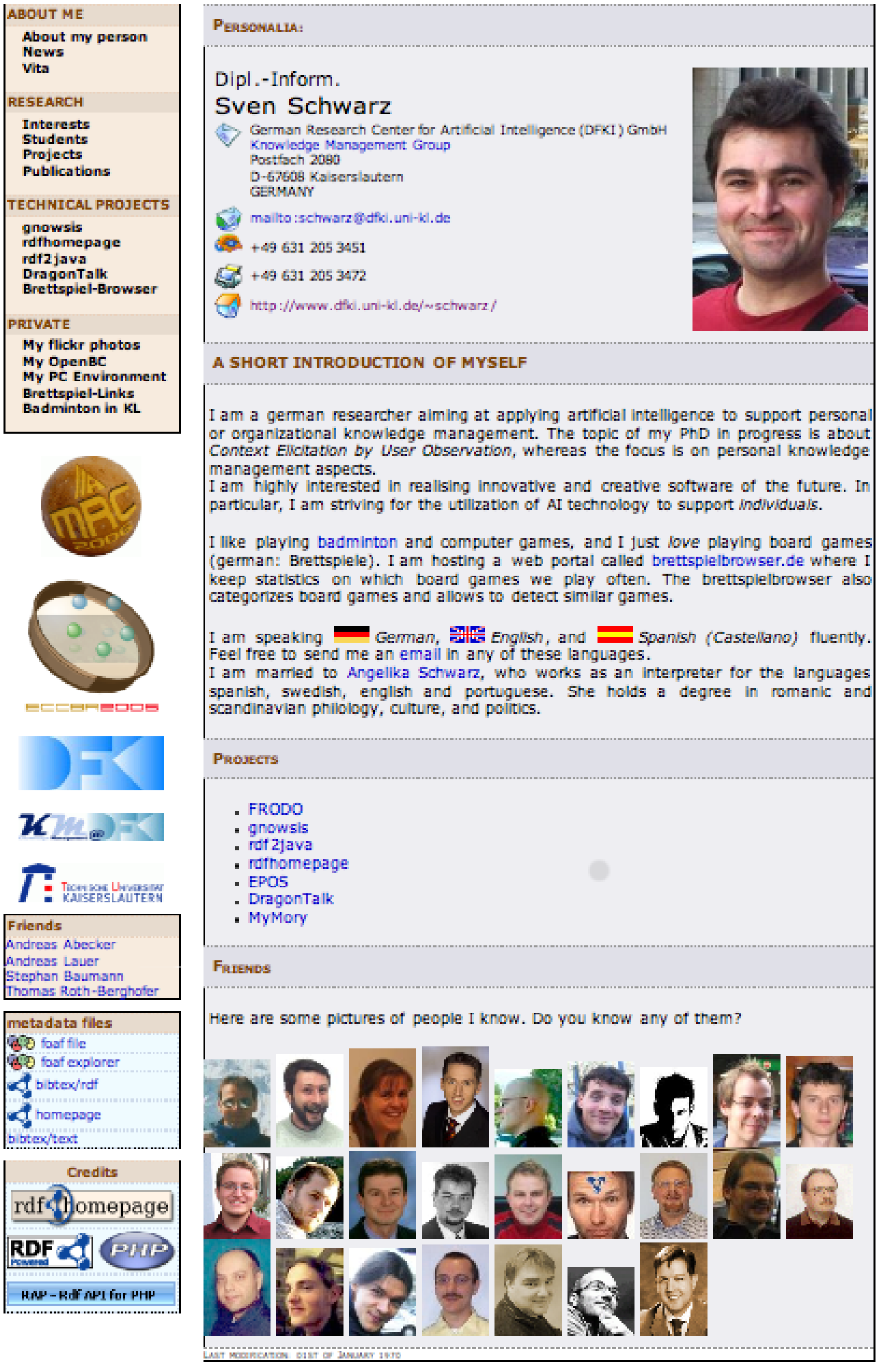
Section 2 outlines the architecture of RDFHomepage. Section 3 discussed the different datasources used by RDFHomepage and Section 4 presents our template engine for generating PHP Classes based on RDF Schemas. Section 5 outlines some future plans and makes concluding remarks.

## 2 Architecture

For RDFHomepage we chose to use the web-scripting language PHP. We chose PHP because it is free and open-source, and is a powerful and feature complete language, with good support for RDF through the RDF API for PHP (RAP) [1]. PHP is also very often provided by cheap hosting providers, unlike more heavyweight solutions such as Java, making the potential userbase of RDFHomepage much larger. In addition, PHP was also initially known as the *Personal Home Page tools*, (although later renamed to *PHP Hypertext Preprocessor*) and we feel RDFHomepage now brings PHP back to its roots. Alternative techniques that were evaluated include: Treehugger by Damian Steer[2] and Masahide Kanzaki's various XSLT tools[3]. These XSLT tools were not chosen because of the steep learning curve, the (often) silent failures which makes debugging very hard in comparison to PHP scripts. RDFHomepage uses RDF data from several standard sources, detailed in the next section, and in addition to these we created a homepage schema, providing the semantic glue between the other sources, and allowing the user to specify additional personal details in a structured form, for example his interests or personal views on projects. Having explicit representations of these items, rather than HTML pages, enables machine processing of the content, such as automatic aggregation of colleagues with similar interest.

The RDFHomepage distribution comes with a standard set of pages, including "About me", "Projects", "Interests" and "Publications". The side-bar menu can be easily customised to include other static or dynamic pages, enabling RDFHomepage to integrate well with exisiting homepage components. There is also a selection of side-bar boxes available, showing elements like links to the raw RDF sources of the page, or links to companies, projects, etc.

Generation of each page compromising the RDFHomepage can take several seconds, depending on the size of the user's data and of course on the web-server being used. A caching mechanism is therefore a part of RDFHomepage, which will only regenerate pages if the underlying RDF data has changed. Unfortunately normal HTTP/1.1 caching mechanisms could not be used for this, as a single page might be dependent on a number of RDF files, both local and remote, so the caching layer was implemented as an additional PHP layer around the page-generating code.



ABOUT ME
About my person
News
Vita

RESEARCH
Interests
Students
Projects
Publications

TECHNICAL PROJECTS
gnowsis
rdfhomepage
rdf2java
DragonTalk
Brettspiel-Browser

PRIVATE
My flickr photos
My OpenBC
My PC Environment
Brettspiel-Links
Badminton in KL

Friends
Andreas Abecker
Andreas Lauer
Stephan Baumann
Thomas Roth-Berghofer

metadata files
foaf file
foaf explorer
bibtex/rdf
homepage
bibtex/text

Credits

RAP - Rdf API for PHP

PERSONALIA:

Dipl.-Inform.
Sven Schwarz

German Research Center for Artificial Intelligence (DFKI) GmbH
Knowledge Management Group
Postfach 2080
D-67608 Kaiserslautern
GERMANY

mailto:schwarz@dfki.uni-kl.de

+49 631 205 3451

+49 631 205 3472

http://www.dfki.uni-kl.de/~schwarz/

A SHORT INTRODUCTION OF MYSELF

I am a german researcher aiming at applying artificial intelligence to support personal or organizational knowledge management. The topic of my PhD in progress is about *Context Elicitation by User Observation*, whereas the focus is on personal knowledge management aspects.
I am highly interested in realising innovative and creative software of the future. In particular, I am striving for the utilization of AI technology to support *individuals*.

I like playing badminton and computer games, and I just *love* playing board games (german: Brettspiele). I am hosting a web portal called brettspielbrowser.de where I keep statistics on which board games we play often. The brettspielbrowser also categorizes board games and allows to detect similar games.

I am speaking *German*, *English*, and *Spanish (Castellano)* fluently.
Feel free to send me an email in any of these languages.
I am married to Angelika Schwarz, who works as an interpreter for the languages spanish, swedish, english and portuguese. She holds a degree in romanic and scandinavian philology, culture, and politics.

PROJECTS

- FRODO
- gnowsis
- rdf2java
- rdfhomepage
- EPOS
- DragonTalk
- MyMory

FRIENDS

Here are some pictures of people I know. Do you know any of them?

LAST MODIFICATION: 01ST OF JANUARY 1970

**Fig. 1. Overview Screenshot of an RDFHomepage Installation**

# 3 RDF Data

## 3.1 FOAF

The Friend-of-a-Friend ontology [4] was the main point of inspiration for RDFHomepage. A huge number of people in the semantic web community have created their own FOAF profile and published it[1], and there are millions more generated by LiveJournal[2], Ecademy[3] and other social sites producing FOAF. However, there are very few consumers of FOAF files, there have been a range of visualisers and explorers[5], and some proof-of-concepts on using FOAF for distributed authentication[4] and trust [6], but no application that really makes it worth your while to create a FOAF profile. In RDFHomepage the FOAF data is used to create the personalia "About me" page, as well as links to people known by the user, and links to the co-authors of papers on the "Publications" page. Extending and enriching your FOAF file now makes sense: A link to a colleague is missing, just add him to your *foaf.rdf* and all parts of your page are automatically updated.

## 3.2 Bibtex

BibTeX is a format for managing citations when using TeX or LaTeX. BibTex defines different classes of publications, such as articles, books, theses, etc., and associated optional and required properties of these. Most computer scientists will keep a BibTeX file of their own publications up to date, for use when self-citing or when publishing their papers on their website. There are many tools for helping this process (for example JabRef[5] and BibDesk[6]). Since people already maintain this information in a structured format it makes sense to reuse this information, and to this end we used BibTeX2RDF, written by Wolf Siberski[7]. A sample BibTeX entry converted to RDF is shown as N3 in Figure 2. The output uses largely standard schemas, such as Dublin Core and VCARD, but does also declare its own namespace for BibTeX specific items. (BibTeX2RDF is configurable to output RDF confirming to any ontology, and unfortunately the actual schema for our version not available). As can be seen from the example the authors of a paper are represented as RDF instances of a Person class. Each person gets a URI generated from their name and the ID of a paper they authored, this ensures there are no collisions. BibTeX2RDF will also merge people with exactly the same name (i.e. *smushing*), so there will only be one node for each person. RDFHomepage uses the BibTeX information to create the "Publications" page, merging the BibTeX person nodes with FOAF people, and uses the info from the FOAF file to generate links to co-author's webpages.

[1] http://rdfweb.org/topic/FOAFBulletinBoard
[2] http://www.livejournal.com
[3] http://www.ecademy.com/
[4] http://rdflib.net/doc/user_managenemt/
[5] http://jabref.sourceforge.net/
[6] http://bibdesk.sourceforge.net/

This means users have an interest in keeping the names aligned, and may edit their BibTeX accordingly. We note however, that this is not really a technically satisfying solution. In addition to the author links, RDFHomepage will also link to the PDFs of any papers available for download, as well as generating links to the homepages of the venues where papers are published. The PDF downloads links are taken from BibTeX directly, URL being one of the optional fields of all BibTeX types, and most BibTeX tools make is easy to attach PDFs to citations this way. These links are retrieved from an internal DFKI wiki page listing conference abbreviations and corresponding web-links. To generate links to the corresponding conferences, a social approach was taken. This gathering of relevant conferences and their URLs was already being done in our research-group and the existing Wiki page is now simply parsed looking for HTML links tags with abbreviations as link texts. The conference names and URLs are extracted and matched to the names user in the BibTeX data. This motivated people to keep the Wiki page updated, having a similar effect as seen with the FOAF files.

```
@prefix : <http://www.w3.org/2001/vcard-rdf/3.0#> .
@prefix bibtex: <http://www.edutella.org/bibtex#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<aberer2003chatty:ACM_Press> a bibtex:Organization;
  :ADR [
    :Country “USA”;
    :Locality “New York” ];
  :FN “ACM Press” .

<aberer2003chatty:Aberer_Karl> a bibtex:Person;
  :FN “Karl Aberer”;
  :N [
    :Family “Aberer”;
    :Given “Karl” ] .

(...)

<aberer2003chatty> a bibtex:InProceedings;
  dc:creator [
    a rdf:Seq;
    rdf:_1 <aberer2003chatty:Aberer_Karl>;
    rdf:_2 <aberer2003chatty:Cudre-Mauroux_Philippe>;
    rdf:_3 <aberer2003chatty:Hauswirth_Manfred> ];
  dc:date “2003-05”;
  dc:identifier “http://www2003.org/cdrom/papers/refereed/p471/471-aberer.html”;
  dc:publisher <aberer2003chatty:ACM_Press>;
  dc:title “The Chatty Web: Emergent Semantics Through Gossiping”;
  dcterms:isPartOf [
    a bibtex:Proceedings;
    dc:date “2003-05”;
    dc:publisher <aberer2003chatty:ACM_Press>;
    dc:title “Proceedings of the 12th Intl. WWW Conference” ;
    :ADR [
      :Country “Hungary”;
      :Locality “Budapest” ] ];
  bibtex:pages “197-206” .
```

**Fig. 2.** A BibTeX entry as converted to RDF.

### 3.3 Projects

At the DFKI information about all projects and the people working in them is centrally maintained in a Organisational Repository (OrgRep)[7] [8]. OrgRep was originally created for use in the FRODO project, but has been maintained and used in many DFKI projects since, for example EPOS, SmartFlow, and MyMory. The organisation ontology is used in EPOS to infer relevant contextual information, i.e. given a person as input the system can look up relevant projects, and vice versa. An example project entry from OrgRep is shown in Figure 3. In addition to the fields shown here OrgRep also contains a longer general description of a project as HTML data, this is used on the "Project" page, unless the user choses to override this with his own personal view on the project. Again the heavy reuse of the data in several contexts increased the motivation to keep your data up-to-date.

```
@prefix : <http://km.dfki.de/model/org#> .
@prefix a: <http://dfki.rdf.util.rdf2java/default#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

:rdfhomepage a :Project;
  :belongsTo :OrganisationalModel_00095;
  :containsMember a:id_20040921_120247_156f920,
      :GuentherNoack,
      :MalteKiesel,
      :OrganisationalModel_00077,
      :OrganisationalModel_00102;
  :homepage "http://rdfhomepage.opendfki.de/";
  :logo "http://www.dfki.uni-kl.de/~schwarz/rdfhomepage/images/rdfhomepage06-180.png";
  :managedBy :OrganisationalModel_00077;
  :name "rdfhomepage";
  rdfs:label "rdfhomepage" .
```

**Fig. 3.** Example OrgRep Project Entry.

## 4 RDF Class Templates

Working with RDF on the triple level can be very difficult and time consuming for developers and it introduces lots of scope for errors. To facilitate this process we created an RDF Template Generator for PHP, this takes one or more RDF Schema documents and produces a PHP class definition based on a selected RDF Class. This means that programmers no longer have to deal directly with the RDF API, but can continue work on the level of PHP objects which they are more used to. An example PHP Class outline generated from the OrgRep ontology is shown in Figure 4. Note how the properties of the OrgRep have

[7] The OrgRep ontology can be downloaded from http://www.dfki.uni-kl.de/~grimnes/2006/03/orgrep/orgrep.rdfs

been mapped to getter-methods of the form *get_namespace_localname*, where the namespace is abbreviated. Each getter method takes a boolean argument specifying whether an array or a single value should be returned. Arrays are useful when a property is either multi-valued or the value is an RDF list or collection, if multiple values are present but only a single value requested a random one is returned. The values returned from these getters are either instances of other generated template classes, depending on the value being a typed resource and the appropriate class existing, or the raw RDF nodes otherwise. Note also that since RDFHomepage only does processing of RDF data, there are no setter-methods generated, however, these could easily be added following the same pattern. The template generator allows loading of multiple schemas before creating the classes, as is illustrated by the *rdfhome_htmltext* property shown in the example, which originates from the homepage schema, but still has an *rdf:domain* of an *OrgRep project*. The template generator also supports inference, so for example: in the case of FOAF, one can generate a *Person* class which includes the properties from *Agent* (a super class of *Person* in the FOAF schema). The generates templates build on top of RAP and can seamlessly fit into a project already using the RAP API.

The approach of mapping RDF objects to a the Object Orientation framework of a programming language is not new and is similar to many other projects, for example: Tramp [9] by Aaron Swartz (the original as far as we are aware) did this for Python; Sparta [10], a more up-to-date Python solution; ActiveRDF [11], a Ruby version; and several Java variants, for example RDFReactor [12] and rdf2java [13]. Outside the RDF community this approach was also used for mapping relational databases to objects, as done with for example Enterprise Java Beans, or the SQLObject in Python[8]. There also exists a library that provides this functionality for PHP, called RDF World [14], but it is different from our approach in that all the RDF processing is done at runtime, whereas we pregenerate the class-templates. This means our solution is much faster when running, and also removes the need to have the triples of the RDF schemas in memory during runtime.

### 4.1 Evaluation

As shown by the large number of projects offering different solutions for mapping RDF data to classes in object-orientation frameworks, this is a very natural idea. As we see it there are two main reasons to use this approach:

1. Hiding the complexities of RDF – most programmers are comfortable with thinking in terms of objects and classes, whereas working with RDF graphs can be very complicated.
2. Enabling code-autocompletion – modern integrated development environments can be a tremendous help when writing object oriented code, and removes the need to know the ins and outs of the APIs one is using. With

[8] http://www.sqlobject.org/

```
<?
class org_Project {
  function org_Project($uri,$model)

  function get_rdfhomepage_htmlText($a=false)
  function get_org_eMail($a=false)
  function get_org_homepage($a=false)
  function get_org_projectFlyer($a=false)
  function get_org_startYear($a=false)
  function get_org_endYear($a=false)
  function get_org_logo($a=false)
  function get_org_containsMember($a=false)
  function get_org_hasProfile($a=false)
  function get_org_informalDescription($a=false)
  function get_org_managedBy($a=false)
  function get_org_name($a=false)

  function getProperty($s,$p,$m=False)
}
?>
```

**Fig. 4.** Example of Generated PHP Class Outline.

pre-generated code for RDF classes this convenience can also be used with RDF.

3. Adding custom code to wrapper – by creating classes that inherit from the generated classes one can embed functionality specific to particular classes in the most natural place. For example, *foaf:Person* could have an added function for automatically generating sha1 checksums of the email address.

With regard to the first point our RDF Templates were only partly successful, the users still have to understand how multiple values for a single property work in RDF, and still have to understand the difference between resources, literals and anonymous nodes. With regard to the second point, full code-completion is possible with our solution, and was used by several of the coders during devlopment. Code-completion being possible is another positive effect of pre-generating the classes, a downside of this pre-generation is that we are tied to a static schema, and cannot directly deal with properties that do not conform. For instance, if an instance of *foaf:Person* has a *middleName* property (which doesn't appear in the FOAF specification), it cannot be accessed using a get method. However, a general *getProperty* method is generated to work around this.

## 5 Conclusion & Future Work

RDFHomepage provides a quick and easy to way to create attractive and informative homepages, with all the content typically found on a research active user's homepage. The pages are generated from RDF files allowing complete separation of content and appearance, as well as making the homepage machine processable by semantic web agents. Editing of a few simple configuration files makes customising the generated pages trivial, and if the customisation offered is not sufficient the use of auto-generated PHP classes makes writing additional PHP code to handle RDF very easy. The scenario of "creating a homepage" proved to be a good playground for Semantic Web technologies and as the commercial success of facebook.com and myspace.com show, it is a very active area. Deploying RDFHomepage at our department triggered users to keep their FOAF files, BibTeX files, the organisational repository and internal Wiki pages updated, and this in turn leads to more accurate date in other projects. We have several plans for further developments of RDFHomepage, primarily we continue to add sources of structured information that user's may want to add to their homepage. Things we planning to add very soon include:

- Flickr[9] photo streams – for example, a side-bar box showing thumbnails of the user's last photos.
- RSS/Atom support – listing the last entries in the user's blog.
- Calendar support – where is the user today? What are his plans this week? This could for instance make use of the RDF Calendaring efforts[10], which include converters between iCal and RDF.

There is also scope for improving our template system: to make it a more general solution for RDF based software engineering we need to generate "setters" as well as "getters" for the properties. Another important addition is to add support for data-types that exist in PHP, and automatically convert fields that represent for example dates to a standard format.

RDFHomepage is open source, release under a GNU Public License (GPL) and can be downloaded from http://rdfhomepage.opendfki.de. To get a better impression of what RDFHomepage can do, consider visiting any of the author's homepages, they are all automatically-generated![11]

## References

1. Radoslaw Oldakowski, Christian Bizer, D.W.: RAP: RDF API for PHP. In: Workshop on Scripting for the Semantic Web (SFSW 2005) at 2nd European Semantic Web Conference (ESWC 2005). (2005)
2. Steer, D.: TreeHugger. (http://rdfweb.org/people/damian/treehugger/)
3. Kanzaki, M.: XSLT Tools. (http://www.kanzaki.com/info/who.html.en)
4. Brickley, D., Miller, L.: FOAF Vocabulary Specification (2006) http://xmlns.com/foaf/0.1/.
5. Fredriksen, M.: FOAF Explorer. (http://xml.mfd-consult.dk/foaf/explorer/)
6. Golbeck, J., Parsia, B., Hendler, J.: Trust Networks on the Semantic Web. In: Seventh International Workshop on Cooperative Information Agents (CIA-03), Helsinki, Finland (2003) 238–249
7. Siberski, W.: BibTeX2RDF. (http://www.l3s.de/∼siberski/bibtex2rdf/)

[9] http://flickr.com

[10] http://www.w3.org/2002/12/cal/

[11] See also http://rdfhomepage.opendfki.de/cgi-bin/trac.cgi/wiki/WorkingInstallations

8. van Elst, L., Abecker, A., Bernardi, A., Lauer, A., Maus, H., Schwarz, S.: An Agent-based Framework for Distributed Organizational Memories. In Bichler, M., Holtmann, C., Kirn, S., Mller, J.P., Weinhardt, C., eds.: Coordination and Agent Technology in Value Networks, Multikonferenz Wirtschaftsinformatik (MKWI-2004), 9.-11.3.2004, Essen, GITO-Verlag, Berlin (2004) 181–196
9. Swartz, A.: (TRAMP: Makes RDF look like Python data structures. ) http://www.aaronsw.com/2002/tramp.
10. Nottingham, M.: (Sparta: a Simple API for RDF) http://www.mnot.net/sw/sparta/.
11. Oren, E.: (ActiveRDF - putting the semantic web on rails) http://activerdf.m3pe.org/.
12. Völkel, M., Sure, Y.: RDFReactor – From Ontologies to Programmatic Data Access. In: Poster Proceedings of the Fourth International Semantic Web Conference. (2005) http://rdfreactor.ontoware.org/.
13. Sven Schwarz, M.K., Sintek, M.: (RDF2Java) http://rdf2java.opendfki.de.
14. Snyder, C.: (Rdfworld.php) http://chxo.com/rdfworld/index.htm.