# Facilitating Casual Users in Interacting with Linked Data through Domain Expertise

Cormac Hampson[1] and Owen Conlan[1],

[1] Knowledge & Data Engineering Group,
Trinity College Dublin, Ireland
{Cormac.Hampson, Owen.Conlan}@tcd.ie

**Abstract.** Linked Data use has expanded rapidly in recent years; however there is still a lack of support for casual users to create complex queries over this Web of Data. Until this occurs, the real benefits of having such rich metadata available will not be realised by the general public. This paper introduces an approach to supporting casual users discover relevant information across multiple Linked Data repositories, by enabling them to leverage and tailor semantic attributes. Semantic attributes are semantically meaningful terms that encapsulate expert rules encoded in multiple formats, including SPARQL. Semantic attributes are created in SABer (Semantic Attribute Builder), which is usable by non-technical domain experts. This opens the approach to almost any domain. A detailed evaluation of SABer is described within this paper, as is a case study that shows how casual users can use semantic attributes to explore multiple structured data sources in the music domain.

**Keywords:** Domain Experts, Casual Users, Semantic Attributes, Linked Data, Data Exploration

## 1 Introduction

In recent years, casual computer users have found themselves accessing diverse structured and semi-structured data on an increasingly regular basis. A casual computer user can be seen as one with Internet browsing ability, but without programming skills and data modeling expertise [1]. With the advent of the Linking Open Data community project[1] and the proliferation of web services and mash-ups, this trend of casual users interacting more with distributed data sources is likely to continue. However, while one may intuitively expect the additional structure in the data to have been exploited to provide sophisticated query capabilities, this has largely not proven to be the case [2]. Many applications using structured data provide access to their underlying data store via query languages; however these are suitable primarily for developers with a knowledge of the language rather than regular end-users wishing to ask very specific questions through a usable human interface [2].

Imagine a casual computer user trying to locate "all nostalgia music artists currently touring the USA". The information necessary to answer this quite subjective

[1] http://esw.w3.org/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

query is likely to be stored over multiple sources, thus techniques should support casual users to find it. One such way to help casual users would be to give them access to Subject Matter Expertise (SME) to support complex query formation. Such human domain expertise[2] is widely used in expert systems, which are developed to help users find reliable information in narrow areas such as medicine or accounting [3]. There are many examples of how casual users could benefit from being supported by such experts to explore Linked Data collections. For instance the increasing availability of government health, transport and education data can open up the opportunity to discover things like bicycle accident black spots and correlations between certain environmental factors and high levels of a disease [4]. Thus, supporting casual users with SME can help them find useful information.

Though SME can be an excellent way of supporting casual users to locate useful information, the encoding of this expertise into an Expert System's knowledge base and the creation of the corresponding rules in its inference engine is not trivial. Hence, domain experts who do not have skills in these areas are likely to require the additional support of a knowledge engineer. Furthermore, even when this expertise is encoded successfully, it often resides in bespoke standalone systems, which leaves little scope for the expertise to be reused in different applications. This is especially true when the user interface, domain expertise, and the knowledge base are tightly coupled together. Thus, a generic platform where non-technical experts could encode their experience and knowledge of a domain in a reusable model would be a welcome solution. Such an approach may not capture as much detail as a bespoke system supported by a knowledge engineer, but the simplicity of its design and its generic applicability would make widespread implementation much easier.

This paper addresses the compelling need for casual users to be supported in exploring information domains encoded as Linked Data. It introduces an approach to accomplish this that allows these users to leverage domain expertise to create complex queries across a domain. By appealing to domain experts with limited computer skills, it makes the adoption of this approach in different domains much more likely. As the dependence on digital data escalates, and the increasing use of such repositories by casual users occurs, the need for such user-friendly approaches will be greatly increased. This paper addresses this need directly and is structured as follows: section two discusses some related work; section three gives a brief overview of SARA (Semantic Attribute Reconciliation Architecture) and section four introduces SABer (Semantic Attribute Builder); section five discusses a case study in the music domain; section six details an evaluation of SABer; and section seven summarises the paper.

## 2 Related Work

A number of generic Linked Data browsers have been developed, such as Tabulator [5] and Disco[3], in order to support casual users browse RDF data on the web. These

[2] In terms of this paper an "expert" is defined as anyone with an opinion on a specific domain and the ability to express it; however developers should be free to limit their choice of experts to a specific individual or select group.

[3] http://www4.wiwiss.fu-berlin.de/bizer/ng4j/disco/

systems display all the information that they can find about specific dereferenceable URIs, and enable the user to browse from resource to resource. However, though aimed at supporting casual users, Linked Data browsers have yet to gain wide usage outside the Semantic Web community. Sparallax[4] was developed in order to provide a set-based faceted browsing interface for SPARQL endpoints. Facets and collections are automatically generated as the user browses the data sets, however users of Sparallax are limited to searching one data source at a time, and it is not possible to reconcile information from different repositories. Unfortunately there is no end-user evaluation of the system published, so it is unclear yet if casual users fully accept the set based paradigm and can see its benefits.

Explorator [6] is a domain independent tool for exploring the Semantic Web, which aims at enabling users with minimal knowledge of RDF models to explore an RDF database, without a priori knowledge of the data domain. The developers of Explorator feel that current tools which allow the user to manipulate raw RDF data do not provide a user friendly way to ask questions, and that the user only has a limited way to rearrange, group or filter the data, and process it further. Similar to Sparallax, Explorator uses a set-based paradigm, where after an initial key word search (to match resources in the RDF) the resulting set of data can be further manipulated to either remove uninteresting elements or to add additional elements of interest. Initial small scale experiments have been conducted [7] with users who knew some basic concepts of the Semantic Web and RDF. However, even using these participants who are more experienced than casual web users, the participants struggled to perform tasks using Explorator. The developers plan additional larger-scale experiments to compare different user interface alternatives and interaction paradigms, in order to better support both novice and expert users in exploring the Semantic Web. This is important as the developers admit that Explorator is currently better suited to advanced users who have solid knowledge about RDF rather than casual users.

A common way of supporting data exploration by casual end users is by enabling them to leverage Subject Matter Expertise (SME) encoded by domain experts. This practice is widespread in domain specific applications, because generic search technology starts to degrade when addressing the needs of users in particular fields such as healthcare and finance [3]. Unfortunately the complexity of encoding SME into such a system (be it rules for an inference engine or comprehensive development of an ontology), is such that a non-technical domain expert is likely to require the support of a knowledge engineer. This in turn increases the time and costs involved in such an exercise and the SME encoded often resides in a standalone system with little scope for expertise to be reused in different applications. There is a tradeoff between the benefits of offering manually generated expertise to users and the efficiency of providing automatically generated supports (e.g. dynamically created facets). Hence, the development of domain independent tools which are quick and easy to use, but do not overly compromise in terms of the features they deliver (e.g. expressiveness of queries supported) should be greatly encouraged.

Konduit VQB [8] is a visual query builder that aims to assist users in building queries and running them over RDF data. Specifically the tool is aimed for users with little or no knowledge about SPARQL, as well as users more familiar with Semantic

[4] http://sparallax.deri.ie

Web technologies. Though Konduit VQB is not strictly an SME encoding tool; it could be used by non-technical experts to encode SME in SPARQL, with this SME then leveraged by users wishing to explore an RDF repository. The developers feel that the schema-based SELECT query builder in the tool is the most user-friendly approach to building SPARQL queries, and recommend it for those users having the least knowledge of semantic technologies. According to the developers, it is simple, intuitive and satisfies the large number of occasions when the user wants to search for something based on certain properties. This specific approach does not support the full range of expressivity that SPARQL offers. However the developers feel that the extra complexity involved in forming more complex queries (e.g. using the CONSTRUCT query builder they offer) is likely to be inaccessible to users with no knowledge of RDF and SPARQL.

SPARQLViz [9] contains a Graphical Query Composer that allows users to generate syntactically correct SPARQL queries through a wizard interface. It is particularly useful for novice users who have little or no experience with SPARQL, as the user is able to compose a valid query simply by using familiar user interface widgets in a wizard-like manner [9]. According to the developer the two best features of the approach offered by SPARQLViz are that generated queries are always valid, saving a lot of debugging time, and an in-depth knowledge of the SPARQL query language is no longer needed to be able to generate them. Unfortunately there were no experiments conducted regarding the second assertion, so it is not possible to determine what level of SPARQL expertise (if any) SPARQLViz requires of its users in order to generate queries.

In terms of using a visual query builder to encode SME, a form or wizard based approach appears to offer the most potential for non-technical domain experts due to their relative simplicity. Specifically, the more intuitive schema-based approach for generating statements was mentioned as the most suited to non-technical users. An example of a commercial application that uses such a rule building approach is iTunes[5], which has a *smart playlist*[6] builder. The *smart playlist* builder is aimed at casual end users and allows them to generate XQuerys in a schema-based approach. These XQuerys generate specific music playlists that allows users to better manage and sort their music collection. At smartplaylists.com[7], users can share the rules they have encoded in iTunes, so that others can benefit from their expertise. This supports the notion that SME encoded in this way can be a valuable way of supporting other users to explore and manage their day to day data. The smart playlist feature in iTunes has even been used to organise pdfs about radiology [10] which further supports the contention that a schema-based approach to rule generation, coupled with a form/wizard interface is accessible to a non-technical user of computers. There are other approaches to visually building queries such as the use of graphs which are employed by NITELIGHT [11] and iSPARQL[8]. However, in order to use these tools the user must have a full comprehension of the underlying RDF schema and the query language syntax, which implies a high cognitive load for newcomers and less

[5] http://www.itunes.com
[6] http://support.apple.com/kb/HT1801
[7] http://www.smartplaylists.com/
[8] http://demo.openlinksw.com/isparql/

experienced users [6]. Indeed the developers of NITELIGHT admit that the close correspondence between the graphical notations and query language constructs makes the tool largely unsuitable for users who have no previous experience with SPARQL. Hence, the form/wizard interface appears to be more suited to non-technical domain experts, and is the approach employed by SABer. Section 6 of this paper describes a user experiment with non-technical domain experts, which validates this approach.

## 3 SARA (Semantic Attribute Reconciliation Architecture)

This paper describes an approach for exploring Linked Data by enabling users to form queries from Subject Matter Expertise (SME). This approach is realised in a middleware system called SARA (Semantic Attribute Reconciliation Architecture) [12,13]. SARA is a domain independent framework that that supports casual users (using an application connecting to its API) leverage SME in order to query different information sources (including Linked Data) in a consolidated fashion. SARA considers three groups of users: end-users (who access SARA through custom third-party apps that talk to the API); domain experts (who are not necessarily technical and can use SABer to create semantic attributes for use in SARA); knowledge engineers (who register a data source for use in SARA/SABer by creating a Source Model).

In SARA, superclasses are key entities from the domain chosen by an expert (the term is used here in a sense unrelated to OWL or RDFS superclasses). In essence, any queries made by client applications to the SARA are looking to return instances of one of these superclasses. For instance in the music domain you could select superclasses such as *Artist*, *Song*, *Album*, *Venue* etc. There is no limit to the number of superclasses you select, and there is no need to define any relationships or properties for them which can be an arduous task when creating domain ontologies. Furthermore, new superclasses can be added to SARA at any time so you are not limited to your initial selection.

SARA currently supports data sources in XML, RDF or those accessible through Web APIs. In order to add a Linked Data source to SARA a reusable Source Model must be created in XML for each SPARQL endpoint or Linked Data Repository. Figure 1 shows an example of a Source Model for an RDF data source. It contains the name of the data source, the address of the RDF database or SPARQL endpoint, any namespace prefixes that the predicates use, and any superclasses that this source contains. The SPARQL code corresponding to each domain superclass is the code that will return instances of this superclass within the data source. In its simplest form this code is just a single SPARQL triple in the form of *?result ?predicate_name ?id*, with *?predicate_name* the only code changing from one superclass to another. Thus in Figure 1 *?result foaf:name ?id* (line 7) returns instances of the *Music_Artist* superclass and *?result mysp:country ?id* (line 11) returns instances of the *Country* superclass.

As can be seen in Figure 1, the predicate *mysp:country* has the alias *Country that MySpace artist is from* (line 22) so that it is clearer to domain experts what this predicate actually represents. This predicate has the subject *Music_Artist* as this is the domain superclass that has *mysp:country* as a property in this particular data source.

Likewise, the predicate *mysp:country* has a corresponding object of a *Country* superclass (line 24), as these are the type of instances that this predicate returns from this data source. This process of associating predicates with superclasses allows multiple data sources with different schemas to co-exist in SARA, without having to go through the time consuming and problematic process of being homogenised to a canonical model. In the case of the *mysp:totalfriends* predicate, its subject is also *Music_Artist* (line 16), with its object being a specific value (the number of friends an artist has on the MySpace website) rather than another domain superclass. Thus "Value" is inputted instead of a superclass name (line 17). The final part of the model shown in Figure 1 describes the transform information necessary to convert instances of one superclass to another. In this instance it depicts a single SPARQL triple that transforms *Music_Artist* instances into *Countries* (line 30). In other cases a transform may contain several triples.

```
1.  <Name>MySpace SPARQL Endpoint</Name>
2.  <Location>http://virtuoso.dbtune.org/sparql</Location>
3.  <Graph>&lt;http://dbtune.org/myspace/&gt;</Graph>
4.  <Prefix>foaf:&lt;http://xmlns.com/foaf/0.1/&gt;</Prefix>
5.  <Superclass>
6.     <Name>Music_Artist</Name>
7.     <Code>?result foaf:name ?id.</Code>
8.  </Superclass>
9.  <Superclass>
10.    <Name>Country</Name>
11.    <Code>?result mysp:country ?id.</Code>
12. </Superclass>
13. <Predicate>
14.    <Name>mysp:totalFriends</Name>
15.    <Alias>Total friends on MySpace is</Alias>
16.    <Subject>Music_Artist<Subject>
17.    <Object>Value</Object>
18.    <Units>N/A</Units>
19. </Predicate>
20. <Predicate>
21.    <Name>mysp:country</Name>
22.    <Alias>Country that MySpace artist is from</Alias>
23.    <Subject>Music_Artist<Subject>
24.    <Object>Country</Object>
25.    <Units>N/A</Units>
26. </Predicate>
27. <Transform>
28.   <Subject>Music_Artist</Subject>
29.    <Object>Country</Object>
30.    <Join>?Music_Artist mysp:country ?id.</Join>
31. </Transform>
```

**Fig. 1.** Sample Source Model for an RDF source

Once a Source Model is created for a specific repository it can be reused in any other SARA installation. This means that collections of different Linked Data repositories can be assembled very quickly in SARA if their Source Models have already been generated. Any predicates registered in a Source Model (which capture

details about the technical access to a source) can be used to generate semantic attributes (which capture domain knowledge and may be created by non-technical domain experts). Semantic attributes are defined as discrete units of domain expertise that can be combined together and tailored to support user exploration of an information domain. Within SARA, semantic attributes encapsulate query fragments that can be combined together to form complex queries. They typically act as abstractions and simplifications from the raw data, which are intended to make it more accessible for the ordinary, non-expert user. For instance, semantic attributes can encompass subjective characteristics such as *nearness*, *popularity* and *expensiveness*, as well there more objective values such as *distance in miles*, *number of records sold* and *price*.

Tailoring a semantic attribute enables users to specify, if they wish to, what their interpretation is of a *high quality* audio file or a *popular* song etc. This is achieved by allowing variables in a semantic attribute's query fragment to be populated by user inputted parameters. Each semantic attribute can also include default values defined by the domain expert that allow informed queries to be run quickly without tailoring. A semantic attribute may contain just a single predicate or else combine multiple predicates into a single semantic attribute, e.g. combining the predicates *bitrate*, *sample rate* and *file type* into a single semantic attribute *audio file quality*. Furthermore, all semantic attributes can also be sub-categorised into a number of separate ranges or parameters e.g. the semantic attribute *Price* could be divided into {Expensive - Average – Cheap}, and *Weight* into {Under Weight - Normal Weight - Over Weight – Obese}. This categorisation allows non-experts to access information without detailed knowledge of the domain. All semantic attributes within SARA are represented in XML as Semantic Attribute Models.

SARA has already been successfully applied to a number of domains including music, films, digital humanities and publications. This has been helped by the fact that SARA supports the reusability of semantic attributes and source models in different installations. However, in order to make SARA's widespread deployment more likely, it was necessary to support non-technical domain experts to encode SME as semantic attributes. This led to the development of the SABer (Semantic Attribute Builder) authoring tool, which can be used by non-technical experts without the support of a knowledge engineer.

## 4 SABer (Semantic Attribute Builder)

The Semantic Attribute Builder (SABer) was developed in Adobe Flex to work in tandem with SARA, and focuses on allowing non-technical users to encode their expertise in SPARQL, XQuery or as native API calls, and encapsulating this SME as semantic attributes. It achieves this by automating as many processes as possible, ensuring that rules generated are syntactically correct, and by not requiring the domain expert to understand the underlying query languages. This paper will limit discussion to the creation of SPARQL based Semantic Attributes that are compatible with Linked Data repositories. It must be stressed that the novelty of SABer is not specifically in its GUI itself, but rather that in conjunction with SARA it enables non-technical domain experts to quickly generate SME in a wide range of domains.

Creating a semantic attribute using SABer is a two step process with each step having a dedicated page in the application. The first process in step one is to name the semantic attribute being created. It is important to choose a descriptive name that conveys its meaning clearly, as this is the name that end users will see in the client application. The next task to complete on this page is to select the predicates you want to create your semantic attribute rules from. Once a domain expert is satisfied with the predicates they have chosen they must select from a drop down menu the type of semantic attribute they want to create. They have a choice of three; expert, template or hybrid. An expert semantic attribute only contains the expert's default rule(s) which can't be tailored, a template semantic attribute contains no expert default rule(s) and must be tailored by the end user, and a hybrid semantic attribute contains expert default rules as well as corresponding template rules which can be tailored. When the user is satisfied with his choices he can click to move onto the next stage.

Depending on whether the domain expert has selected an expert, template or hybrid rule the next page displayed will vary slightly. However, regardless of the type of semantic attribute being created, it is at this stage that the domain expert creates the rule or rules for their semantic attribute. The first thing a domain expert must do to generate their SPARQL query is to select the domain superclass that they want to return. To choose a superclass, the domain expert must simply select it from a dropdown menu. This generates the first part of each rule and is printed onscreen as "Return any <Superclasses> where" as depicted in Figure 2, where the user has selected the superclass *MusicArtist*.

Underneath this line the expert is presented with three dropdown menus, a text field and a button all in a row. The first dropdown menu contains all the superclasses that the RDF source has associated with. Depending on what superclass the expert chooses, the predicates (or more precisely the alias of the predicates) in the adjacent dropdown menu will change accordingly. This second dropdown menu contains all the predicates that the domain expert has chosen in the first step, but restricted to those predicates that are associated with the superclass chosen in the first dropdown menu. Thus Figure 1 shows that when the domain expert chooses the superclass *MusicArtist* from the first dropdown menu, they are presented in the second dropdown menu with the elements *Country from is*, *Total Friends on MySpace is*, and *Total page views on MySpace*. However if the domain expert had chosen *Song* as the superclass in the first dropdown menu of that line, then the second dropdown menu would have been populated with *Track Duration*, *Composer*, *Genre* etc. If there are any units associated with the predicate chosen then they are displayed at the end of the line to make clear what range of values is appropriate to input


**Fig. 2.** Sample two Lines of Expert Rule for RDF Based Semantic Attribute

The domain expert would then select the predicate they were interested in and then move on to the third dropdown menu. This dropdown menu contains the available list of operators that the end user can select from. Currently these include; *Greater than*, *Less than*, *Equals*, *Not Equals to*, *Greater than or Equals to*, and *Less than or Equals to*. The domain expert simply selects which operator they want from the drop down menu. The operators other than *Equals* all result in a FILTER statement being added to the SPARQL rule that is in the process of generation.

All the domain expert has to do to finish this line of the rule is to input a value into the adjacent textbox. Thus in Figure 3, on the second line the expert chose the metadata *Total Friends on My Space*, the operator <, and inputted the value *50,000*. This essentially equates to the WHERE part of a SELECT SPARQL statement with the rest of the query automatically generated from the information defined in the Source Model. If the domain expert wants to add more lines to this rule all they have to do is click on the "+" button at the end of the line. This adds another identical line underneath the first, except that it has an additional *and/or* dropdown menu at the start of the line and an additional "-" button at the end.

The *and/or* dropdown menu allows the user to specify if the *MusicArtist* should satisfy both of the rule lines or either of them. If the user selects *or* from this dropdown menu, it results in an OPTIONAL statement being added to the SPARQL rule that is being generated. In Figure 2 the expert has used *and* so only wants *Music Artists* that satisfy both rule lines e.g. *Music Artists whose Total Friends on MySpace is less than 50,000 AND greater than 20,000*. The additional "-" button at the end of the line allows for a rule line to be deleted easily. Each parameter can contain as many rule lines as the domain expert wants, with Figure 2 showing the completed four line rule for *Averagely Popular Irish Artists on MySpace*. At any time in the process the domain experts can select the "Get Results" button to see what instances are currently in the data source that satisfy their rule.


**Fig. 3.** Sample four Lines of Expert Rule for RDF Based Semantic Attribute

The process for creating a template semantic attribute based on RDF data is almost identical to the process just described for creating an expert semantic attribute. The only difference is highlighted in Figure 4. Instead of having a blank text field in which domain experts can input a specific value, they instead are presented with another dropdown menu with two options "*Some Text*" and "*Some Number*". This allows domain experts to create rules such as *Return all Artists where Country From = "Some Text"* or *Return all Songs where chart position < "Some Number"*. By generating these kinds of rules it enables end users to tailor a rule more specifically to what they want.


**Fig. 4.** Sample four Lines of Template Rule for RDF Based Semantic Attribute

The process for a domain expert creating hybrid semantic attributes for RDF sources is identical to creating an expert semantic attribute. The only difference is that when a hybrid semantic attribute is submitted, SABer automatically generates an associated template rule for each of the expert rules to support tailoring. Once any semantic attribute gets submitted, the values and rules inputted into SABer get concatenated with a template to form an XML Semantic Attribute Model. This model then gets saved to SARA and made available to client applications interested in that domain.

## 5 Case Study

This section describes a case study of a SARA installation that connected to five separate music data sources in three different formats. The aim of this case study was to show how SARA technically supported queries to reconcile information from these separate sources. The data sources used in this case study were:

1. An XML iTunes library with over 30,000 songs stored in an eXist database
2. The US Singles charts from 1950-2008 stored as XML in an eXist database
3. The freebase.com music SPARQL endpoint[9]
4. The MySpace.com SPARQL endpoint[10]
5. Last.fm web services[11]

The eXist databases and remote SPARQL endpoints could be directly accessed by queries encapsulated in the semantic attributes. However in the case of web services with a native API, such as the Last.fm service used in this case study, a Java wrapper was needed to proxy queries and results. .Each of these five sources had Source Models registered to SARA which in turn got visualised in SABer. The domain superclasses chosen were *Artist*, *Song*, *Album* and *Country*.

SABer was then used to create semantic attributes which were stored in SARA's Semantic Attribute Library. As will be shown in the section 6, SABer can support non-technical domain experts to generate such semantic attributes. For this case study twenty-five semantic attributes for the domain were created including:

- Artists currently touring specific countries
- Top MySpace artists from specific countries
- Popular Jazz artists in the US Charts in the 1980s
- Similar artists to a specified artist

Once the semantic attributes were made available in SARA it was possible for queries to be sent to it from a client application via its API. For instance, queries

[9] http://lod.openlinksw.com/sparql
[10] http://virtuoso.dbtune.org/sparql
[11] http://www.last.fm/api

combining multiple semantic attributes that reference different sources could be sent to SARA such as:

- Return all Artists from the iTunes collection (iTunes XML database), that have Concerts Scheduled in the USA (Last.fm web service), despite their most recent top 10 Album in the USA being more than ten years ago (US charts database).
- Return all Countries (MySpace SPARQL endpoint) that had popular Artists in the USA during the 1990s (US charts database)
- Return all Songs by The Beatles (freebase SPARQL endpoint) that are in top 10 popular Beatles songs on Last.fm (Last.fm web service) despite not charting in the top 10 in Americas (US charts database).

Many of these queries allowed specifics to be tailored by the end user, so that they could easily specify different bands other than *The Beatles*, tailor the definition of *popular*, or change the range of time. Once the results from the individual data sources were sent back to SARA they were reconciled into a final result set. This was made possible as the different sources contained instance level identifiers to help disambiguation (dereferenceable URIs used in Linked Data are an example of such instance level identifiers). The final result set was then sent as XML for rendering in the client application.

This case study has shown how SARA supports semantic attributes created in SABer to be utilised by a client application, gives consolidated access to multiple sources of different types, and enables instance level integration of results from different data sources. Furthermore, extra superclasses, data sources and semantic attributes could be appended to the system seamlessly if required, and the models generated for this case study could be plugged into different installations that required access to music information.

## 6 SABer Evaluation

For SARA to be applicable to many domains it had to be shown that non-technical domain experts could successfully generate semantic attributes in SABer. As stated previously, for the purposes of this paper 'non-technical' people refers to computer literate participants with basic skills such as operating Internet browsers, but with no computer programming experience. SABer has previously been used by domain experts with computer programming experience to create useful semantic attributes that were deployed within applications. Hence, this experiment was devised to measure the usability of SABer and its effectiveness in supporting non-technical domain experts to generate semantic attributes. Furthermore, it helped explore whether SABer (and by extension the Semantic Attribute Model) can sufficiently abstract the user away from the differences in the various semantic attributes and their underlying data types.

Two groups were assembled of twelve participants each (one group technical the other non-technical), with each person engaging in the experiment separately and in isolation from other participants. An entire session including the demonstration, performance of tasks and filling in of questionnaires typically took around forty

minutes to complete. The first step of the experiment was for each participant to have the creation of three different semantic attributes in the music domain demonstrated to them. This demonstration was done by the evaluator and consisted of him inputting the semantic attribute details from a task sheet into SABer. Demonstrating these three tasks involved an identical process to what the participants would be undertaking in the experiment. SABer supports semantic attributes of three different types (expert, template and hybrid) and of three data types (XML, RDF and data accessible through a Web API). Each of the three tasks demonstrated to the participants involved a combination of a different semantic attribute type with one of the different data types.

Once these three tasks were demonstrated, the participants were given nine different semantic attributes of varying complexity to create in SABer. These tasks were presented to the participants in a random order as users tend to get quicker with later tasks when they are more familiar with the application interface. By presenting the tasks in a random order it meant that the average time taken to create semantic attributes would not be longer due to appearing near the start of the list, and likewise not be shorter due to being near the end of the list. The semantic attributes users were asked to create are listed as follows:

1. Quality Of Audio Files in my iTunes Collection
2. Top Singles in US in 1990s
3. Artists of a Genre who had US single that Reached a Specific Position
4. Countries Paul McCartney has Concerts Scheduled in
5. Artists Scheduled to Play Iceland
6. Popular Beatles Songs According to Last.fm
7. Songs On A Specific Album By Specific Artist
8. Countries with Very Popular Artists on MySpace
9. Irish Artists Popularity on MySpace

These semantic attributes spanned the same five sources outlined in the case study section.

Because the participants were given the rules to encode into semantic attributes (e.g. highly Irish popular artists on MySpace are those with *Total friends on MySpace > 50,000 and Country from is = Ireland*) it was not necessary for them to be "expert" in the domain per se. This was justified as the usefulness of the semantic attributes being created was not being evaluated, but rather the ease in which coherent rules could be constructed by non-technical users. During the course of the experiment the length of time it took each semantic attribute to be created was recorded, the accuracy of the semantic attribute noted (whether it exactly matched the semantic attribute given to them on paper) and any questions or problems that they asked recorded. Users were also given the opportunity to create semantic attributes of their own after creating the nine semantic attributes set for them.

Once finished using SABer, each user completed a SUS (Standard Usability Scale) test [14] to measure the systems usability and also filled in a short post questionnaire. The SUS test provided an indicator as to the usability of SABer and the post questionnaire gave space for participants to elaborate on any usability issues or functionality they would like to see. The post questionnaire asked participants to specify if they found inputting rules for any of expert, hybrid or template semantic attributes a considerably more difficult challenge, and likewise if they found inputting rules for any specific query type significantly more challenging. This would allow it

to be accessed if part of the SABer application needed to be adjusted to make inputting semantic attributes of certain types more intuitive. Moreover it would help determine if the Semantic Attribute Model was sufficiently generic and abstract to allow users to ignore the underlying idiosyncrasies of querying different data types.

Once the experiment was completed the results from the twelve technical users were used to provide a baseline performance in terms of speed and accuracy in creating the semantic attributes. Technical domain experts have already used SABer to generate useful and deployable semantic attributes, thus by comparing the time it takes for non-technical users to create accurate semantic attributes it would give a good indicator as to the usability and utility of the tool. Hence, if the group of non-technical users performed, on average, at a level of accuracy and efficiency near their more technical counterparts, it could be reasonably concluded that SABer was suitable for non-technical domain experts to use. Moreover, by comparing the average SUS score for technical, non-technical and overall groups, a good indication of the tool's usability would be garnered.

With regards to the speed in creating semantic attributes, on average the non-technical users were only 8.4% slower than their technical counterparts. In terms of the accuracy of the semantic attributes created, the difference in performance was even smaller between groups than in the speed comparison. Technical users on average got 8.5 out of 9 accurate with a Standard Deviation (SD) of 0.67,, with non-technical users getting 8.2 out of 9 accurate (SD of 0.84). It must also be noted that all inaccuracies by participants in both groups can be classified as slips where wrong figures or spellings were inputted by the users. Slips are defined by attentional failures where the action was unintended [15]. These kinds of errors can be easily corrected, and there were no cases of a user fundamentally not being able to create a semantic attribute or giving up half way through. Furthermore, all users were able to create their own semantic attributes after completing the nine semantic attributes that were set for them. Many of these semantic attributes were of comparable complexity to those set for them during the experiment. In terms of usability, on average the technical users gave SABer a SUS score of 83.3% (SD of 9.4), and the non-technical users 74.4% (SD of 10.7) The average SUS score for all 24 users was 78.85% (SD of 10.05). Systems that score above 72.5% on the SUS scale can be classified as having good usability [16], so it can be concluded that SABer is considered a usable tool by both non-technical and technical users.

The post questionnaire that participants filled in asked them to specify if they found inputting rules for any of expert, hybrid or template semantic attributes a considerably more difficult challenge. 22 of the 24 users found no significant extra difficulty in creating semantic attributes of different types (hybrid, template or expert). None of the participants felt that creating template semantic attributes was more difficult than any of the others. This meant that all participants were comfortable with selecting *"some text"* or *"some number"* while creating rules instead of inputting specific values. This was important to validate as non-technical users would typically not be as familiar with the concept of a variable as technical users would, and this concept had to be presented to them in an intuitive fashion. The post questionnaire also asked if participants found inputting rules in a specific query language significantly more challenging. 22 out of the 24 users found no significant difference in difficulty in creating semantic attributes using XQuery, SPARQL or API

calls. This was important as it showed that users were largely indifferent to the underlying technologies they were working with, and meant that the semantic attribute sufficiently abstracted them away from the underlying technical complexity of each query language.

Usability is defined by the International Organization for Standardization (ISO) as the effectiveness, efficiency and satisfaction with which a specified set of users can achieve a specified set of tasks in a particular environment [17] . Participants in both the technical and non-technical groups were shown to perform the tasks effectively and efficiently. Moreover, there was no significant difference in speed, accuracy or perceived usability of SABer between the two groups. Users also found no significant difference in creating semantic attributes of different types or with different queries thus showing that SABer sufficiently abstracted users away from the underlying data sources. It can thus be concluded from this section that SABer is a tool with good usability and that domain experts without a computer science background could use it to create semantic attributes with a minimal amount of training. The significance of this is that it should be possible for experts in metadata rich domains to capture SME as semantic attributes if given a sufficient choice of elements. Hence, the tool is very applicable to Linked Data sources.

## 7 Summary

This paper described an approach to support casual users discover relevant information across multiple Linked Data repositories, by enabling them to leverage and tailor semantic attributes. Currently the rich metadata exposed through Linked Data is only minimally used by casual users, hence it is important to make this information more accessible to these users. SARA and its authoring tool SABer are designed to support interaction by casual users with Linked Data, and both systems were detailed within this paper. It was also shown how Linked Data repositories can be registered with SARA through a reusable Source Model, which means that once created, the same model can be reused in any installation that wants to access that specific data source.

The paper also detailed how SABer supports non-technical domain experts to encode SME as queries in multiple languages (including SPARQL) that are then encapsulated as semantic attributes. A successful evaluation of SABer was discussed in detail. Any semantic attributes generated in SABer can be reused in any installation of SARA that access those same sources, which makes it easy for different client applications to quickly benefit from the SME encoded by domain experts. This reusability of SME and its non-reliance on knowledge engineers make the approach supported by SARA and SABer suitable for almost any domain with rich metadata. A case study of SARA's use in the music domain was also described, which showed how users could form complex queries over multiple structured and semi-structured data sources (including Linked Data accessed by SPARQL endpoints). This case study showed SARA's potential to be used to help casual users navigate the rapidly expanding Web of Data.

**Acknowledgments.** This research has been supported by The Irish Research Council for Science, Engineering and Technology: funded by the National Development Plan.

## References

1. Huynh, D., Miller, R. and Karger, D., "Potluck: Data mash-up tool for casual users," Web Semantics: Science, Services and Agents on the World Wide Web, vol. 6, Nov. pp. 274-282 (2008)
2. Bizer, C., Heath, T. and Berners-Lee, T., "Linked data-the story so far," International Journal on Semantic Web and Information Systems, vol. 5, p. 1–22 (2009)
3. Vaughan-Nichols, S., "Researchers Make Web Searches More Intelligent," Computer, vol. 39, pp. 16–18 (2006)
4. Hall, W,. "What web science could mean for businesses," http://www.computerweekly.com/Articles/2010/04/12/240862/interview-wendy-hall-on-what-web-science-could-mean-for.htm%20 uterweekly.com, (2010)
5. Berners-Lee, T. et al, "Tabulator: Exploring and analyzing linked data on the semantic web", Proceedings of the 3rd International Semantic Web User Interaction Workshop (2006)
6. De Araújo, S.F.C. and Schwabe, D., "Explorator: a tool for exploring RDF data through direct manipulation", in LDOW 2009: Linked Data on the Web, (2009)
7. De Araújo, S.F.C., Schwabe, D. and Barbosa, S.D.J.,"Experimenting with Explorator: a Direct Manipulation Generic RDF Browser and Querying Tool", in Visual Interfaces to the Social and the Semantic Web, pp. 1-9 (2009).
8. Ambrus, O., Moeller, K. and Handschuh, S.,"Konduit VQB: a Visual Query Builder for SPARQL on the Social Semantic Desktop," in Workshop on Visual Interfaces to the Social and Semantic Web, (2010)
9. Borsje, J., Embregts, H.: Graphical Query Composition and Natural Language Processing in an RDF Visualization Interface. B.Sc. Erasmus University, Rotterdam (2006).
10. Qian, L.J., Zhou, M. and Xu, J.R., "An easy and effective approach to manage radiologic portable document format (PDF) files using iTunes.," AJR. American journal of roentgenology, vol. 191, Jul., pp. 290-1 (2008)
11. A. Russell, P. Smart, and D. Braines, NR, "NITELIGHT: A Graphical Tool for Semantic Query Construction," Semantic Web User, pp. 1-10 (2008)
12. Hampson, C. and Conlan, O., "Supporting Personalized Information Exploration through Subjective Expert-created Semantic Attributes," IEEE International Conference on Semantic Computing. ICSC'09., Berkeley, pp. 384-389 (2009)
13. Hampson, C. and Conlan, O., "Leveraging Domain Expertise to Support Complex, Personalized and Semantically Meaningful Queries Across Separate Data Sources," IEEE International Conference on Semantic Computing. ICSC'10., Pittsburgh, pp. 305-308 (2010)
14. Brooke, J., "SUS-A quick and dirty usability scale," Usability evaluation in industry, pp. 189-194 (1996)
15. Reason, J., "Human error", Cambridge University Press (1990).
16. Bangor, A., Kortum, P. and Miller, P., "Determining what individual SUS scores mean: Adding an adjective rating scale," Journal of Usability Studies, vol. 4, pp. 114-123 (2009)
17. Jokela, T., Iivari, N., Matero, J. and Karukka, M., "The standard of user-centered design and the standard definition of usability: analyzing ISO 13407 against ISO 9241-11," Proceedings of the Latin American conference on Human-computer interaction, pp. 53-60 (2003)