# User Interests: Definition, Vocabulary, and Utilization in Unifying Search and Reasoning

Yi Zeng[1], Yan Wang[1], Zhisheng Huang[2],
Danica Damljanovic[3], Ning Zhong[1,4], Cong Wang[1]

[1] International WIC Institute, Beijing University of Technology, China
yzeng@emails.bjut.edu.cn
[2] Vrije University Amsterdam, The Netherlands
huang@cs.vu.nl
[3] University of Sheffield, United Kingdom
d.damljanovic@dcs.shef.ac.uk
[4] Maebashi Institute of Technology, Japan
zhong@maebashi-it.ac.jp

**Abstract.** Consistent description and representation method of user interests are required for personalized Web applications. In this paper, we provide a formal definition and the "e-foaf:interest" vocabulary for describing user interests based on RDF/OWL and the FOAF vocabulary. As an application, under the framework of unifying search and reasoning (ReaSearch), we proposed interests-based unification of search and reasoning (I-ReaSearch) to solve the personalization and scalability requirements for Web-scale data processing. We illustrate how user interests can be used to refine literature search on the Web. Evaluation from the scalability point of view shows that the proposed method provides a practical way to Web-scale problem solving.

**Keywords**. user interests, interest vocabulary, Web search refinement, unifying search and reasoning.

## 1 Introduction

User interests are of vital importance and have impact on various real world applications, especially on the Web. Based on the idea of Linked data [1], it would be very useful if user interests data can be interoperable across various applications. However, interoperability in this context requires consistent description and representation of user interests. In this paper, we address this problem by providing a formal definition of user interests – "e-foaf:interest" vocabulary based on RDF/OWL and the Friend of a Friend (FOAF).

Further on, we show how we apply this vocabulary in the context of ReaSearch – the framework of Unifying Search and Reasoning [2] aimed at removing the scalability barriers of Web-scale reasoning. ReaSearch emphasizes searching the most relevant sub-dataset before the reasoning process. User interests can be considered as contextual constraints that may help to find what the users really want when the original query is vague or there are too many query results that the user has to wade through to find the most relevant ones [3]. Hence, we

propose a concrete method to implement the "ReaSearch" framework, namely, interests-based unification of search and reasoning (I-ReaSearch). As an application domain of user interests and "I-ReaSearch", we investigate on how they can be used in literature search on the Web. We also make a comparative study on the scalability of the proposed method.

## 2 The Definition and Vocabulary of User Interests

In this section, firstly, we give a formal definition of the user interest. Secondly, we propose an RDF/OWL based vocabulary so that various data sources and applications can interoperate with each other based on the proposed vocabulary.

### 2.1 A Definition of User Interests

In Cambridge Advanced Learner's Dictionary, interest is defined as "the activities that you enjoy doing and the subjects that you like to spend time learning about" [4]. From our point of view, a user interest is not only about a topic or a subject, but it is also about the time (for example, when the user interest appeared? or, when the user lost the interest?), and the value of it. Hence, here we give a formal definition of user interest.

A user interest is the subject that an agent (the agent can be a specific user, a group of users, or other types of intelligent agent) wants to get to know, learn about, or be involved in. It can be described as a five tuple:

$$< Interest\_URI, AgentURI, Property(i), Value(i), Time(i) >,$$

where $Interest\_URI$ denotes the $URI$ address that is used to represent the interest, $AgentURI$ denotes the agent that has the specified interest. $Property(i)$ is used to describe the name of the $i$th property of the specified interest (here we assume that there are $n$ properties that is used to describe the interest from different perspectives, and $i \in [1, n]$). $Vaule(i)$ denotes the value of $Property(i)$. $Time(i)$ is the time that $Vaule(i)$ is acquired for the $Property(i)$.

In real world applications, knowledge representation languages are needed to describe user interests based on the above definition. In order to have better integration with the Web of linked data [1], we propose to describe user interests based on RDF/OWL.

### 2.2 The e-foaf:interest Vocabulary

We title the vocabulary as the "e-foaf:interest Vocabulary", as it is aimed at extending the FOAF vocabulary on user interests evaluated from different perspectives. It focuses on extending "foaf:interest" by providing more details with regards to user interests.

The e-foaf:interest vocabulary has 3 versions, namely: "e-foaf:interest Basic", "e-foaf:interest Complement", and "e-foaf:interest Complete". They are composed of a set of class vocabularies and a set of property vocabularies.

Table 1: The "e-foaf:interest" vocabulary list

| Vocabulary Branch | Vocabulary | Type |
|---|---|---|
| e-foaf:interest Basic | e-foaf:interest | Class |
| | e-foaf:interest_value | Property |
| | e-foaf:interest_value_updatetime | Property |
| | e-foaf:interest_appeared_in | Property |
| | e-foaf:interest_appeare_time | Property |
| | e-foaf:interest_has_synonym | Property |
| | e-foaf:interest_co-occur_with | Property |
| e-foaf:interest Complement | e-foaf:cumulative_interest_value | Property |
| | e-foaf:retained_interest_value | Property |
| | e-foaf:interest_longest_duration | Property |
| | e-foaf:interest_cumulative_duration | Property |

The "e-foaf:interest Complete" is the union of the set of vocabularies from "e-foaf:interest Basic" and "e-foaf:interest Complement". Here we give some details on the definition of each vocabulary (The namespaces are omitted for brevity).

– **e-foaf:interest** (Class)
Definition: e-foaf:interest is a class that is used to represent the agent's interest.

```
<owl:Class rdf:ID="e-foaf:interest">
  <rdfs:subClassOf rdf:resource="&owl;Thing"/>
</owl:Class>
```

– **e-foaf:interest_value** (Property)
Definition: "e-foaf:interest_value" represents the value of an interest. The value of a specified interest is an arbitrary real number. The number represents the degree of interests that a user has in a specific topic. If the agent is interested in an interest, the interest value is greater than zero (namely a positive number). If the agent is not interested in a topic, the interest value of the topic is smaller than zero (namely a negative number).

```
<owl:DatatypeProperty rdf:ID="e-foaf:interest_value">
  <rdfs:domain rdf:resource="#e-foaf:interest" />
  <rdfs:range  rdf:resource="&xsd;number" />
</owl:DatatypeProperty>
```

Note: This property is supposed to be oriented to interest values from any perspective. Some possible perspectives are defined in "e-foaf:interest Complement", namely, the perspective of "cumulative interest value", "retained interest value", "interest lasting time", "interest appear time", etc.). It can also be a user defined value.

– **e-foaf:interest_value_updatetime** (Property)
Definition: "e-foaf:interest_value_updatetime" represents the update time of the interest value. It may be the time when the user specifies the interest value, or the time when an algorithm updates the value of the interest.

```
<owl:DatatypeProperty rdf:ID="e-foaf:interest_value_updatetime">
  <rdfs:domain rdf:resource="#e-foaf:interest" />
  <rdfs:range  rdf:resource="&xsd;dateTime" />
</owl:DatatypeProperty>
```

– **e-foaf:interest_appeared_in** (Property)
Definition: "e-foaf:interest_appeared_in" represents where the interest appeared in.

```
<owl:DatatypeProperty rdf:ID="e-foaf:interest_appeared_in">
  <rdfs:domain rdf:resource="#e-foaf:interest" />
  <rdfs:range  rdf:resource="foaf:Document" />
</owl:DatatypeProperty>
```

Note: This property is used to preserve the original resources where the interests came from, so that these can be reused for calculation of interest value when needed. The design of this property is inspired by "from" in Attention Profiling Markup Language (APML 2009) which is based on XML.

– **e-foaf:interest_appear_time** (Property)
Definition: "e-foaf:interest_appear_time" is the time when the interest appears in a certain kind of scenario.

```
<owl:DatatypeProperty rdf:ID="e-foaf:interest_appear_time">
  <rdfs:domain rdf:resource="foaf:Document" />
  <rdfs:range  rdf:resource="&xsd;dateTime" />
</owl:DatatypeProperty>
```

– **e-foaf:interest_has_synonym** (Property)
Definition: "e-foaf:interest_has_synonym" represents that the subject and the object of this property are synonyms. Such as "search" and "retrieval".

```
<owl:ObjectProperty rdf:ID="e-foaf:interest_has_synonym">
  <rdfs:domain rdf:resource="#e-foaf:interest" />
  <rdfs:range  rdf:resource="#e-foaf:interest" />
</owl:ObjectProperty>
```

Note: In some use cases, synonyms need to be merged together or marked as semantically very related; this property is very useful in such scenarios.

– **e-foaf:interest_co-occur_with** (Property)
Definition: "e-foaf:interest_co-occur_with" represents that the subject and the object of this predicate co-occur with each other in some cases.

```
<owl:ObjectProperty rdf:ID="e-foaf:interest_co-occur_with">
  <rdfs:domain rdf:resource="#e-foaf:interest" />
  <rdfs:range  rdf:resource="#e-foaf:interest" />
</owl:ObjectProperty>
```

– **e-foaf:cumulative_interest_value** (Property)
Definition: "e-foaf:cumulative_interest_value" is a sub property of "e-foaf:interest_value" representing the cumulative value of the number of times an interest appears in a certain kind of scenario.

```
<owl:DatatypeProperty rdf:ID="e-foaf:cumulative_interest_value">
  <rdfs:subPropertyOf rdf:resource="#e-foaf:interest_value" />
  <rdfs:domain rdf:resource="#e-foaf:interest" />
  <rdfs:range  rdf:resource="&xsd;number" />
</owl:DatatypeProperty>
```

– **e-foaf:retained_interest_value** (Property)
Definition: "e-foaf:retained_interest_value" represents the retained interest value of an interest in a specific time.

```
  <owl:DatatypeProperty rdf:ID="e-foaf:retained_interest_value">
    <rdfs:subPropertyOf rdf:resource="#e-foaf:interest_value" />
    <rdfs:domain rdf:resource="#e-foaf:interest" />
    <rdfs:range  rdf:resource="&xsd;number" />
  </owl:DatatypeProperty>
```

Note: The retained interest value can be calculated based on the interest retention function such as the one proposed in [5].

– **e-foaf:interest_longest_duration** (Property)
Definition: "e-foaf:interest_longest_duration" is used to represent, until a specified time, the longest duration of the interest (between it appears and disappears).
Note: For example, if the interest appears in the following years: 1990, 1991, 1995, 1996, 1997, 1998, 2001, the longest duration is 4 years.

```
  <owl:DatatypeProperty rdf:ID="e-foaf:interest_longest_duration">
    <rdfs:domain rdf:resource="#e-foaf:interest" />
    <rdfs:range  rdf:resource="&xsd;duration" />
  </owl:DatatypeProperty>
```

– **e-foaf:interest_cumulative_duration** (Property)
Definition: "e-foaf:interest_cumulative_duration" is used to represent the cumulative duration of an interest – the duration from the moment when it first appeared.
Note: For example, if the interest appears in the following years: 1990, 1991, 1995, 1996, 1997, 1998, 2001, then its cumulative duration is 7 years.

```
<owl:DatatypeProperty rdf:ID="e-foaf:interest_cumulative_duration">
  <rdfs:domain rdf:resource="#e-foaf:interest" />
  <rdfs:range  rdf:resource="&xsd;duration" />
</owl:DatatypeProperty>
```

We should emphasize that each corresponding interest value is calculated based on a specific function, and the update time might not be consistent. Hence, each interest value (including cumulative interest and retained interest) has a specific update time. An illustrative example using the e-foaf:interest Vocabulary and some working SPARQL queries can be acquired from the "e-foaf:interest" specification web site[5].

The interest profile can be considered as contextual information when the specific user queries a scientific literature system (e.g. CiteSeerX, DBLP) or uses other types of applications. More refined results can be acquired when adding user interests as implicit constraints to the original (vague) query [3]. In the following section, we investigate how the user interests can be involved in unifying search and reasoning and provide a corresponding logical framework.

## 3 Unifying Search and Reasoning with User Interests

"ReaSearch" proposed in [2] is aimed at solving the problem of scalability for Web-scale reasoning. Its core philosophy is to select an appropriate subset of semantic data required for reasoning. ReaSearch is trying to solve the scalability issue by incomplete reasoning as the dataset acquired from the Web itself is very likely to be incomplete anyway. In [6], the author argues that for the Web search based on large scale data, more relevant results can be found by adding logic. This effort can also be considered as unifying search and reasoning. Nevertheless, more concrete strategies should be developed. In this section, we first introduce a concrete approach for "ReaSearch" based on user interests (I-ReaSearch), followed by the proposal of the two concrete strategies to implement I-ReaSearch.

### 3.1 The I-ReaSearch Framework

When users are trying to find useful knowledge on the Web, bridging the query topic with user background knowledge can help to understand the query results and is convenient for human to learn [7]. User interests can be considered as a special type of background knowledge from users, hence it can be considered as a context for literature search on the Web. In this paper, we propose to unify search and reasoning based on user interests.

Following the notion in [2], we title the efforts as "I-ReaSearch", which means unifying reasoning and search with Interests. The process of I-ReaSearch can be described as the following rule:

[5] The "e-foaf:interest" specification is available from: http://wiki.larkc.eu/e-foaf:interest. The vocabulary specification is an ongoing effort in the EU FP-7 framework project LarKC

$$hasInterests(U, I), hasQuery(U, Q), executesOver(Q, D), \neg contains(Q, I) \rightarrow IReaSearch(I, Q, D),$$

Where $hasInterests(U, I)$ represents that the user "$U$" has a list of interests "I" , $hasQuery(U, Q)$ represents that there is a query input "$Q$" by the user "U", $executesOver(Q, D)$ denotes that the query "$Q$" is executed over the dataset "$D$", $\neg contains(Q, I)$ represents that the query "$Q$" does not contain the list of interests "I", $IReaSearch(I, Q, D)$ represents that by utilizing the interests list "$I$" and the query "$Q$", the process of unifying selection and reasoning is applied to the dataset "$D$".

Currently, there are two strategies under "I-ReaSearch". Both utilize user interests as the context, but their processing mechanisms are different.

### 3.2 Interests-Based Query Refinement

For the strategy of user interests based Query Refinement, the idea is to add more constraints to the user's input query based on the user interests extracted from some historical sources (such as previous publication, visiting logs, etc.). The process can be described by the following rule:

$$hasInterests(U, I), hasQuery(U, Q), executesOver(Q, D), \neg contains(Q, I) \rightarrow refinedAs(Q, Q'), contains(Q', I), executesOver(Q', D).$$

In this rule, $refinedAs(Q, Q')$ represents that the original query "Q" is refined by using the list of Interests as "$Q'$". $contains(Q', I)$ denotes that "$Q'$" contains the list of Interests "$I$". $executesOver(Q', D)$ represents that the refined query "$Q'$" executes over the dataset "D". Namely, in interests-based query refinement, "$refinedAs(Q, Q'), contains(Q', I), executesOver(Q', D)$" implements $IReaSearch(I, Q, D)$ in the I-ReaSearch general framework.

Based on this rule, we emphasize that this approach utilizes the user context to provide a rewritten query so that more relevant results can be acquired.

### 3.3 Querying with Interests-based selection

One of the ways to achieve a Web-scale reasoning is to perform a selection step beforehand – this step would identify only those statements which are necessary, enabling the reasoner to finish all tasks in real time [2]. The Strategy of querying with Interests-based selection builds on this idea of selection: the assumption is that user interests might help to find a relevant subset so that the reasoner does not have to process the large amounts of data, but just those parts which are necessary. The process can be described by the following rule:

$$hasInterests(U, I), hasQuery(U, Q), executesOver(Q, D), \neg contains(Q, I) \rightarrow Select(D', D, I), executesOver(Q, D').$$

where "Select(I,D')" represent the selection of a sub dataset "$D'$" from the original dataset "D" based on the interests list "I", and $executesOver(Q, D')$ represents that the query is executed over the selected sub dataset "$D'$". Namely, in querying with interests-based selection, "$Select(D', D, I), executesOver(Q, D')$" implements $IReaSearch(I, Q, D)$ in the I-ReaSearch general framework.

## 4 Interests-Based ReaSearch from Different Perspectives

When the query is vague/incomplete, research interests can serve as constraints that can be used to refine the queries. Research interests can be evaluated from various perspectives and each perspective reflects one unique characteristic. When the user is not satisfied with the specific perspective, the interests list used for interests-based ReaSearch will be changed. This process can be described by the following rules:

$$IReaSearch(I, Q, D), \neg satisfies(U, R) \rightarrow IReaSearch(I', Q, D),$$

where $IReaSearch(I, Q, D)$ denotes that the ReaSearch process is based on the interest list "I" and the query "Q" on the dataset "D". $\neg satisfies(U, R)$ denotes that the user "U" does not satisfy with the query results "R" from $IReaSearch(I, Q, D)$. $IReaSearch(I', Q, D)$ denotes that the interest list is changed from "I" to "$I'$" and the ReaSearch process is based on the new interest list "$I'$".

In this paper, four perspectives on the evaluation of user interests are considered. Namely, the cumulative interest value, the retained interest value, the interest longest duration, and the interest cumulative duration, which are introduced in Section 2.2. Some illustrative examples on how user interests can help to get more refined results are available from [8] and http://wiki.larkc.eu/csri-rdf.

## 5 Evaluation

In [8], the evaluation results from user studies have shown that the user prefers the refined results with user interests in comparison to the unrefined ones. In this section, we evaluate the proposed method from the perspective of scalability. We present a comparative study on the query effectiveness among three different strategies :

1. Query based on the original user input (no refinement).
2. Interests-based query refinement (introduced in Section 3.2).
3. Querying with Interests-based selection (introduced in Section 3.3).

As an illustrative example, we take the SwetoDBLP dataset [9] which is divided into 22 sub-datasets. We evaluate the 3 implemented strategies by using these datasets at different scales. A comparative study is provided in Figure 1. Two users are taken as examples, namely Frank van Harmelen and Ricardo Baeza-Yates. Top 9 retained interests for each of them are acquired based on the retained interest function (introduced in [3]) and used to unify the selection

and reasoning process. The above three different kinds of querying strategies are performed on the gradually growing dataset (each time adding 2 subsets with the same size, around 55M for each, and 1.08G in total).

As shown in Figure 1, strategy 2 considers more constraints compared to the strategy 1, and therefore requires more time for processing – as the size of the dataset grows, the processing time grows very rapidly, which means that this method does not scale well if we just consider the required time. However, the quality of acquired query results is much better than with strategy 1.

Since strategy 3 selects relevant sub-dataset in advance, the required query time is significantly reduced. As the size of the dataset grows, the query time increases but not as fast as is the case for strategy 2. At the same time, the quality of the query results is the same as with strategy 2. Hence, this method scales better.

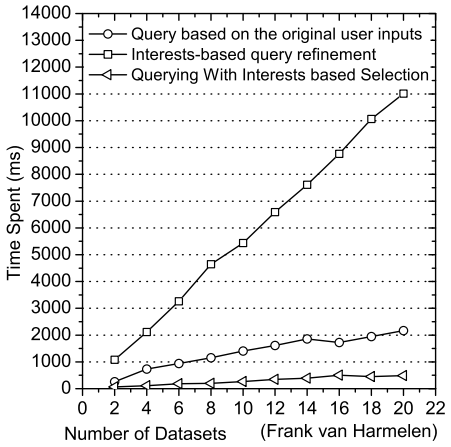


Fig. 1: Scalability on query time for three different strategies

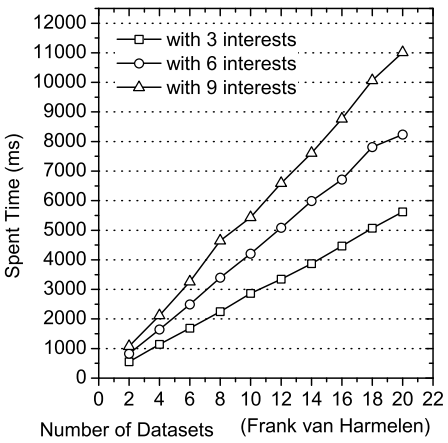


Fig. 2: Interests based refined query time with 3, 6, 9 interests

Further on, for strategy 2 and strategy 3, we examined the impact of the number of constraints in a query so that one can immediately see the necessity of having a balance among query refinement and processing time. Figure 2 shows the query processing time with 3,6,9 interests constraints from the user "Frank van Harmelen". We can conclude that the number of constraints in the query is positively correlated with the query processing time. Hence, even if strategy 2 and 3 yield better results compared to strategy 1, one should be cautious about adding too many constraints to the original query each time.

## 6 Conclusion

In order to support user interests based applications on the linked data Web, in this paper, we give a formal definition of the user interest, and define an extended vocabulary of FOAF focusing on user interests. The aim of this vocabulary is to make the description and representation of user interests in a more consistent way so that various applications can share user interests data.

As an application of user interests, interests-based unification of search and reasoning (I-ReaSearch) is proposed to solve the scalability and diversity problems for Web-scale reasoning. Two types of strategies are introduced, namely, interests-based query refinement, and querying with interests-based selection. From the result quality perspective, they perform equally well. From the scalability perspective, latter scales better than the former. This effort can be considered as a foundation towards user centered knowledge retrieval on the Web [10].

## Acknowledgments

This study is supported by the research grant from the European Union 7th framework project FP7-215535 LarKC [6].

## References

1. Bizer, C.: The emerging web of linked data. IEEE Intelligent Systems **24**(5) (2009) 87–92
2. Fensel, D., van Harmelen, F.: Unifying reasoning and search to web scale. IEEE Internet Computing **11**(2) (2007) 96, 94–95
3. Zeng, Y., Yao, Y., Zhong, N.: Dblp-sse: A dblp search support engine. In: Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence. (2009) 626–630
4. : Cambridge Advanced Learner's Dictionary. 3 edn. Cambridge University Press (2008)
5. Zeng, Y., Yao, Y., Zhong, N.: Dblp-sse: A dblp search support engine. In: Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence. (2009) 626–630
6. Berners-Lee, T., Fischetti, M.: Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor. HarperSanFrancisco (1999)
7. Bransford, J., Brown, A., Cocking, R.: How People Learn: Brain, Mind, Experience, and School. National Academy Press (2000)
8. Zeng, Y., Ren, X., Qin, Y., Zhong, N., Huang, Z., Wang, Y.: Social relation based scalable semantic search refinement. In: The 1st Asian Workshop on Scalable Semantic Data Processing (AS2DP 2009), co-located with the 2009 Asian Semantic Web Conference (ASWC 2009). (2009)
9. Aleman-Meza, B., Hakimpour, F., Arpinar, I., Sheth, A.: Swetodblp ontology of computer science publications. Journal of Web Semantics **5**(3) (2007) 151–155
10. Yao, Y., Zeng, Y., Zhong, N., Huang, X.: Knowledge retrieval (kr). In: Proceedings of the 2007 IEEE/WIC/ACM International Conference on Web Intelligence. (2007) 729–735

[6] http://www.larkc.eu