Service Choreography Meets the Web of Data Via Micro-Data

Xi Bai and Dave Robertson

Centre For Intelligent Systems and their Applications, School of Informatics, University of Edinburgh

xi.bai@ed.ac.uk

Abstract

Several solutions exist for semantically describing Web Services (WSs) from the perspective of orchestration but little is known about how semantics benefit WS choreography. The most extreme example of a choreography problem occurs in peer-to-peer systems where shared semantics of data may need to be established via services interactions. We present a solution to this problem by sharing micro-data via interaction models. No pre-unified ontology is required in our approach so peers can make use of existing heterogeneous resources having been described in the RDF data model flexibly and compatibly. The experimental results indicate that our approach semantically enhances WS choreography in a lightweight way which complies with principles of Linked Data and republished Interaction Models (IMs) can further facilitate the progress of the Web of data as well as the formation of peer communities generated through peers' interactions.

Choreography Description in LCC

In a peer-to-peer network, peers are autonomous and each of them has both server and client capabilities. From the perspective of choreography, peers collaborate through interactions and we use LCC to describe the choreographies inside the peer-to-peer network. Although our paper employs LCC, the specific choice of the process language is not essential to the core arguments of this paper. An example in which a client purchases a product referenced by a product code from a shop using his or her credit card is depicted in LCC as follows:

```
    r(client, initial, 1, 1)
    r(shop, necessary, 1)

    a(client(PC, CC), C)::
           buy(PC, CC) ⇒ a(shop, S) ← payby(CC)∧ lookup(S) then
           receipt(R) ⇐ a(shop, S)

    a(shop, S)::
           buy(PC, CC) ⇐ a(client(_),C) then
           receipt(R) ⇒ a(client(_), C) ← enough_credit(CC, PC)∧
                                            complete_order(PC, CC, R)
```

However, since there is no explanation about any elements (e.g., roles, messages and constraints) in this IM, it is difficult if not impossible for peers who want to purchase or sell a product to automatically recognize whether this IM is exactly the one they really need. The following XHTML snippet gives the excerpt of proportional source codes of a Web page on which the trade IM described above has been semantically enhanced.

```
<html
xmlns=‘‘http://www.w3.org/1999/xhtml’’
xmlns:openk=‘‘http://homepages.inf.ed.ac.uk/s0896253/openk.owl#’’
xmlns:dbpedia=‘‘http://dbpedia.org/resource/’’
>
...
<div typeof=‘‘openk:InteractionModel’’>
 <span property=‘‘openk:has_declaration’’>r(client, initial, 1, 1)</span><br/>
 <span property=‘‘openk:has_declaration’’>r(shop, necessary, 1)</span>
 <div rel=‘‘openk:has_role’’>
  <div typeof=‘‘openk:Role’’ property=‘‘openk:has_roletype’’ content=‘‘initial’’
  >a(<span property= ‘‘openk:has_name’’>client</span>(
  <span rel=‘‘openk:has_arg’’ typeof=‘‘openk:Argument dbpedia:Universal_Pro
  duct_Code’’>
  <span property=‘‘openk:has_name’’>PC</span></span>), C)::<br/>
  <span rel=‘‘openk:sendout’’>
   <span typeof=‘‘openk:Message’’>
    <span property= ‘‘openk:has_name’’>buy</span>(
    <span rel=‘‘openk:has_arg’’>
     <span typeof=‘‘openk:Argument dbpedia:Universal_Product_Code’’>
      <span property=‘‘openk:has_name’’>PC</span></span>,
     <span typeof=‘‘openk:Argument dbpedia:Credit_card’’>
...
```

P2P network Topology

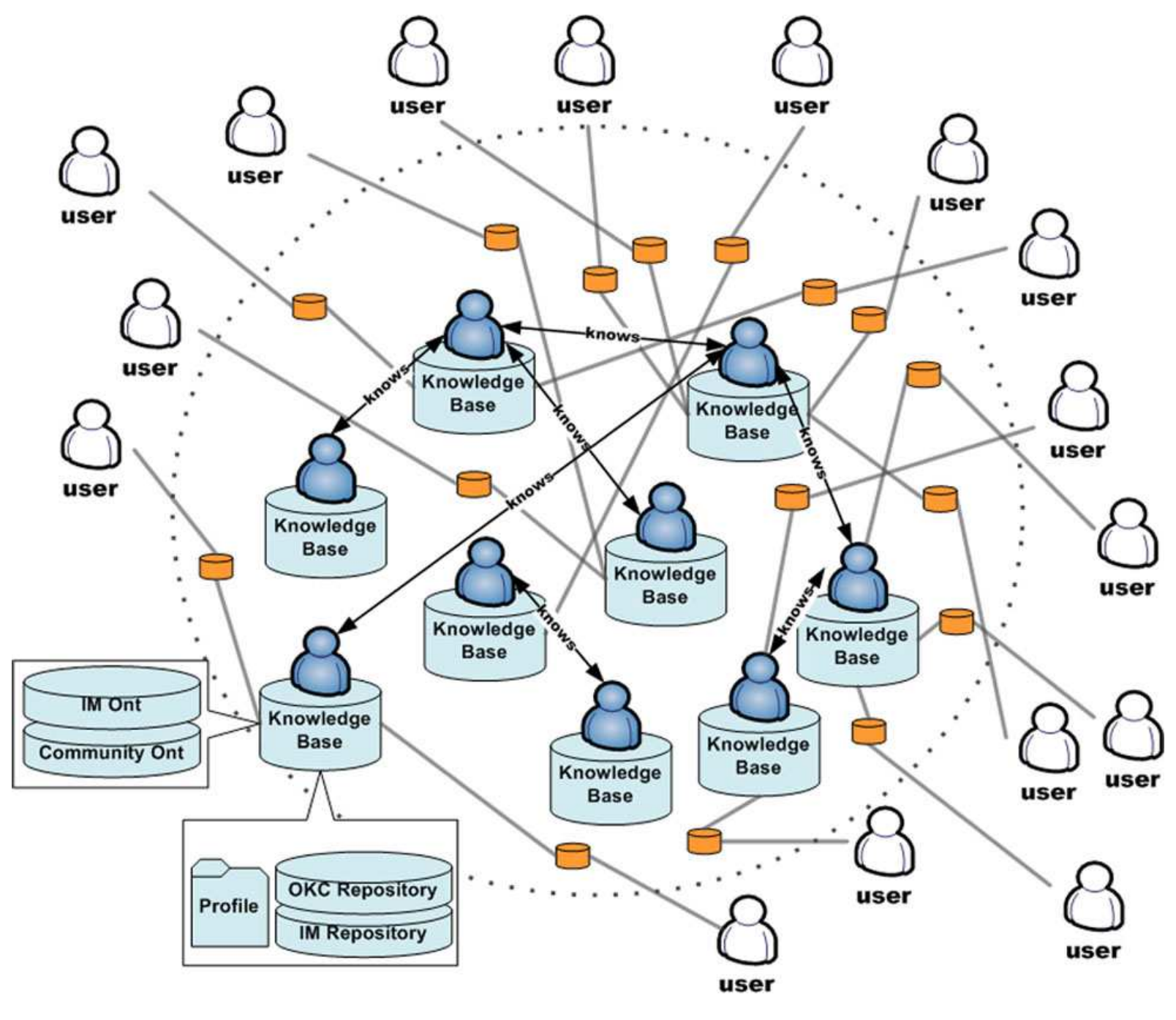


FIGURE 1: Peer-to-peer network Topology

Linking Choreography to the Web of Data

Micro-Data Injection

Currently, our republishing tool provides following functionalities: LCC editing, peer profile browsing, IM annotating, annotation revising and annotation issuing. These functionalities are achieved by designing and implementing the following modules:

a. **Preprocessing Module** is used for assisting a peer in importing the local profile the IM it expects to be published or republished.
b. **Profile Browsing Module** is dedicated to display auxiliary information derived from peers' profiles.
c. **Annotating Module** provides a drag&drop way for publishers who want to attach IM elements with semantic annotations (see Figure 2).
d. **Revising Module** helps IM publishers delete, modify and replace existing annotations before IMs are finally republished.
e. **Issuing Module** has two functionalities, one of which is harvesting embedded RDFa of the IM as new knowledge using RDFa parsers [1] and adding it to the peer's local profile. The other functionality is generating Web page and publishing it on the Distributed Discovery Service (DDS).

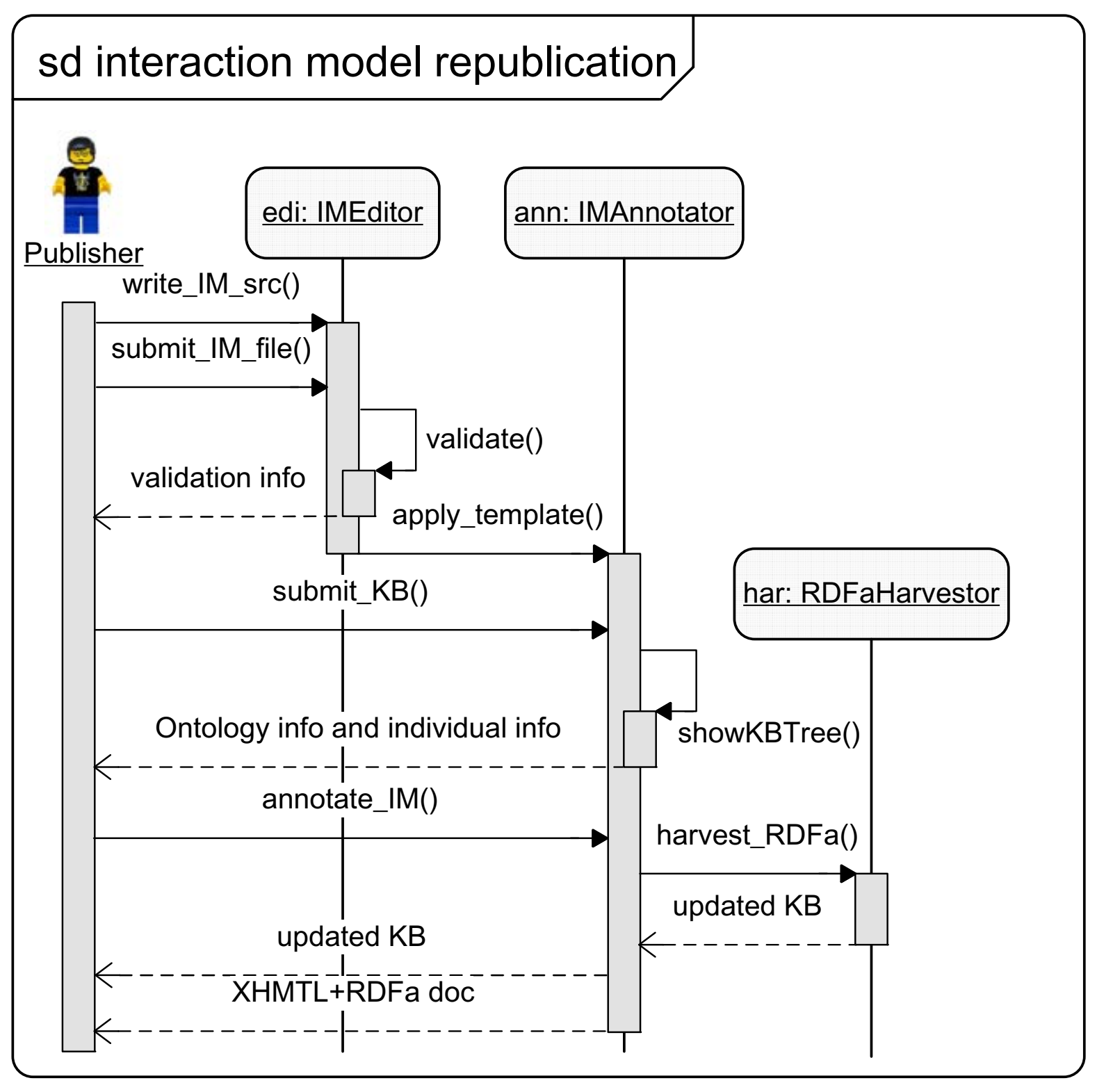


FIGURE 2: Sequence diagram for IM republication

Micro-Data Consumption

Republished IMs can be consumed in various ways and one of them is to assist the DDS in discovering desired IMs. The DDS is one of most important components for the OpenKnowledge system. IMs are located on different peers in a distributed way and the DDS is in charge of discovering IMs that meet the user's requirement and other peers that can collaborate with in terms of his or her input query. Here, "query" means key words or URIs. Since URIs are hard to remember and some URIs are heterogeneous, users can use URIs discovered by RDF search engines like Sindice or URIs through the DBpedia Lookup service [2]. Republished IMs have been semantically enhanced and by harvesting embedded RDFa, the DDS can provide a more precise and more extensible query processing thanks to dereferenceable URIs on the Web of Linked Data.

Another way of consuming embedded RDFa happens when an IM is executed after all required roles are filled by peers. Within the process of running an IM, users normally need to query the data embedded in Web pages. However, this query is different from the aforementioned query that triggers discoveries of appropriate IMs and collaborative peers. Here, the query will be more specific and more targeted such as a tourist asking an airline service for a cheap flight from Edinburgh to San Francisco. In these cases, the OKCs at the side of the service provider will take charge of parsing the Web pages and by matching the harvested RDF data against with the user's query, the coordinator will find out the information that meets the user's requirement.

Experiment

As mentioned above, one way of harnessing republished IMs is to assist the DDS in discovering appropriate IMs that meet the peers' requirements. Suppose a peer P sends a query (both URI-based queries and phrase-based queries are supported) to a DDS peer denoted by D. By matching the query against with micro-data embedded in published IMs, D can select most relevant IMs which are likely to meet the requirement of a user who has logged on to P and created that query. By querying these triples harvested by an RDFa parser, P can check if it can play a specific role in this IM. Moreover, P will also have a chance to query the above harvested triples to find out if this IM has belonged to some communities and then P can apply for a membership and join these communities if it is happy with that. Suppose the URI for this IM is denoted by im_uri; the named graph containing P's profile is denoted by $IM_{triples}$; the URI for the graph containing harvested triples is denoted by IM. Then following SPARQL query will help the peer figure out which role it can play and which potential community it may join:

```
SELECT ?role ?community
FROM ⟨IM⟩
FROM NAMED ⟨IM_triples⟩
WHERE {
 GRAPH ⟨IM_triples⟩ {?P a openk:Peer. ?role a openk:
  Role. ?P openk:can_play ?role.}
 ?community a openk:P2PCommunity.
 ⟨im_uri⟩ openk:has_role ?role.
 ⟨im_uri⟩ openk:belongs_to ?community.
}
```

Another way of making use of republished IMs occurs in their execution processes. In order to test this usage, we created and published an IM for the job vacancy service on UK Civil Service Website [3]. Using the Atom feed provided by this Website, we finally retrieved 338 pages corresponding to 338 jobs. Suppose all pages are attached with a unique ID (0 to 337), the time cost of page retrievals is described in Figure 3. Job vacancy pages from this site are based on a unified form that is actually a template by which the site manager can easily create and update job information. Therefore, the time cost interval for most pages is [$171ms$, $188ms$]. Then the RDFa data are harvested from these page and the corresponding time cost for each harvest is described in Figure 4.

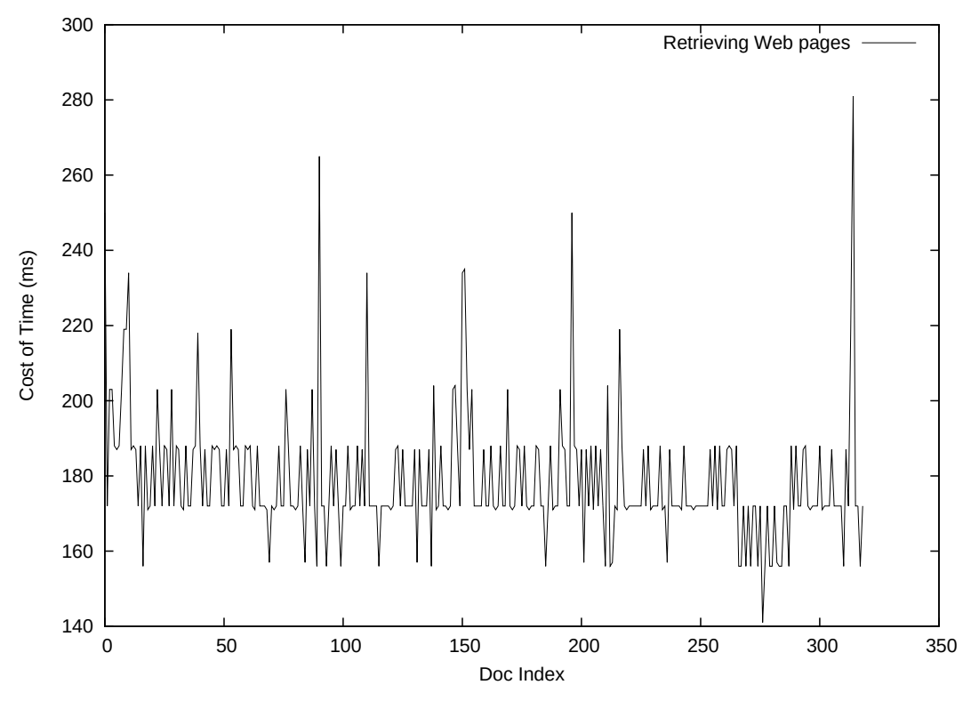


FIGURE 3: Time cost of retrieving pages

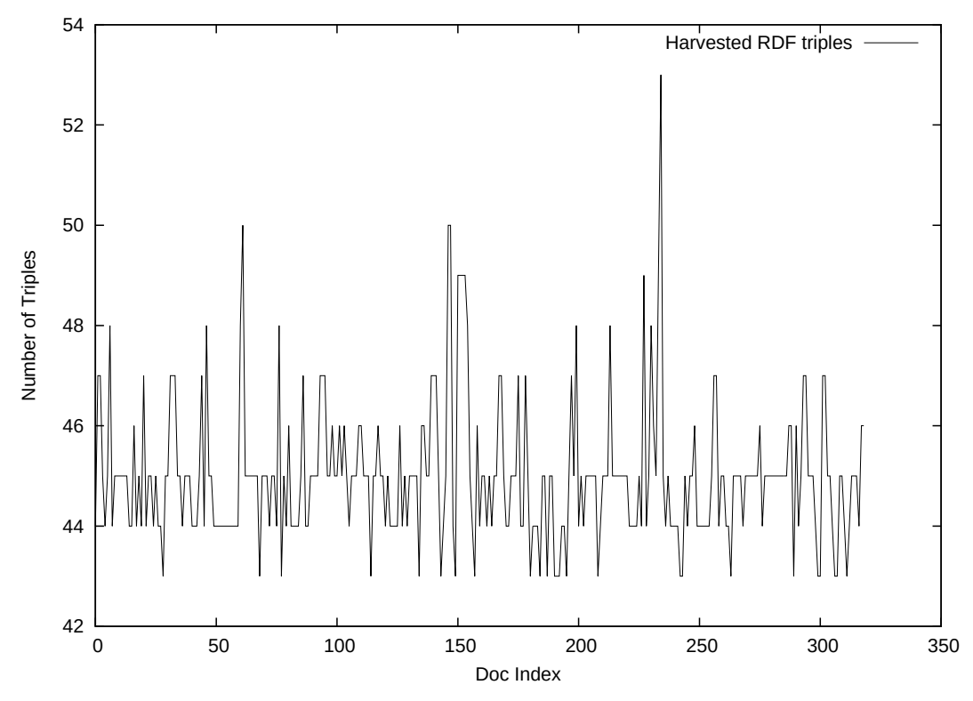
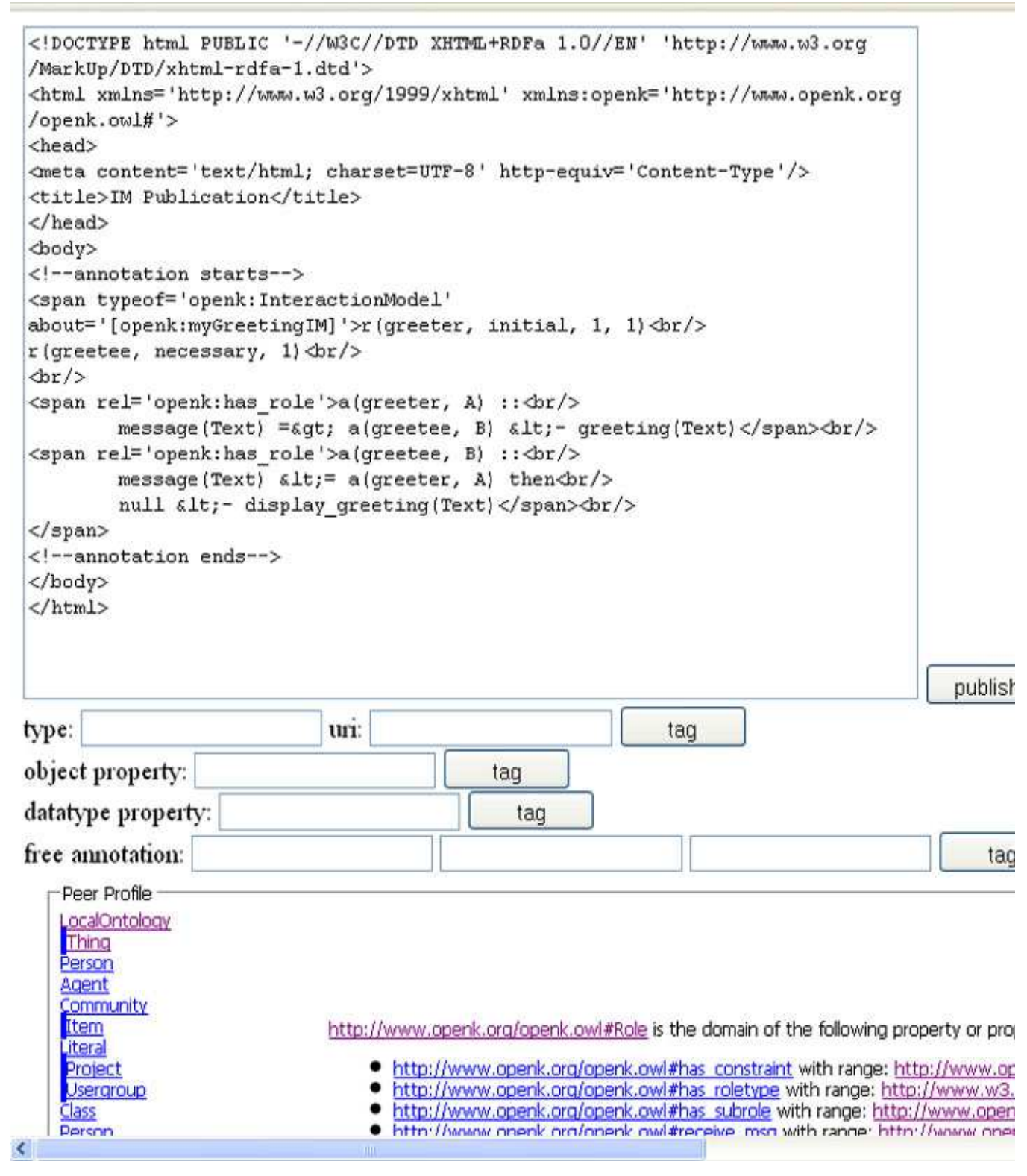


FIGURE 4: Number of harvested triples

Case Studies

The number of harvested RDF triples for each document is depicted in Figure 5, from which we can see that most of pages contain the same numbers of RDF triples (from 43 triples to 53 triples).



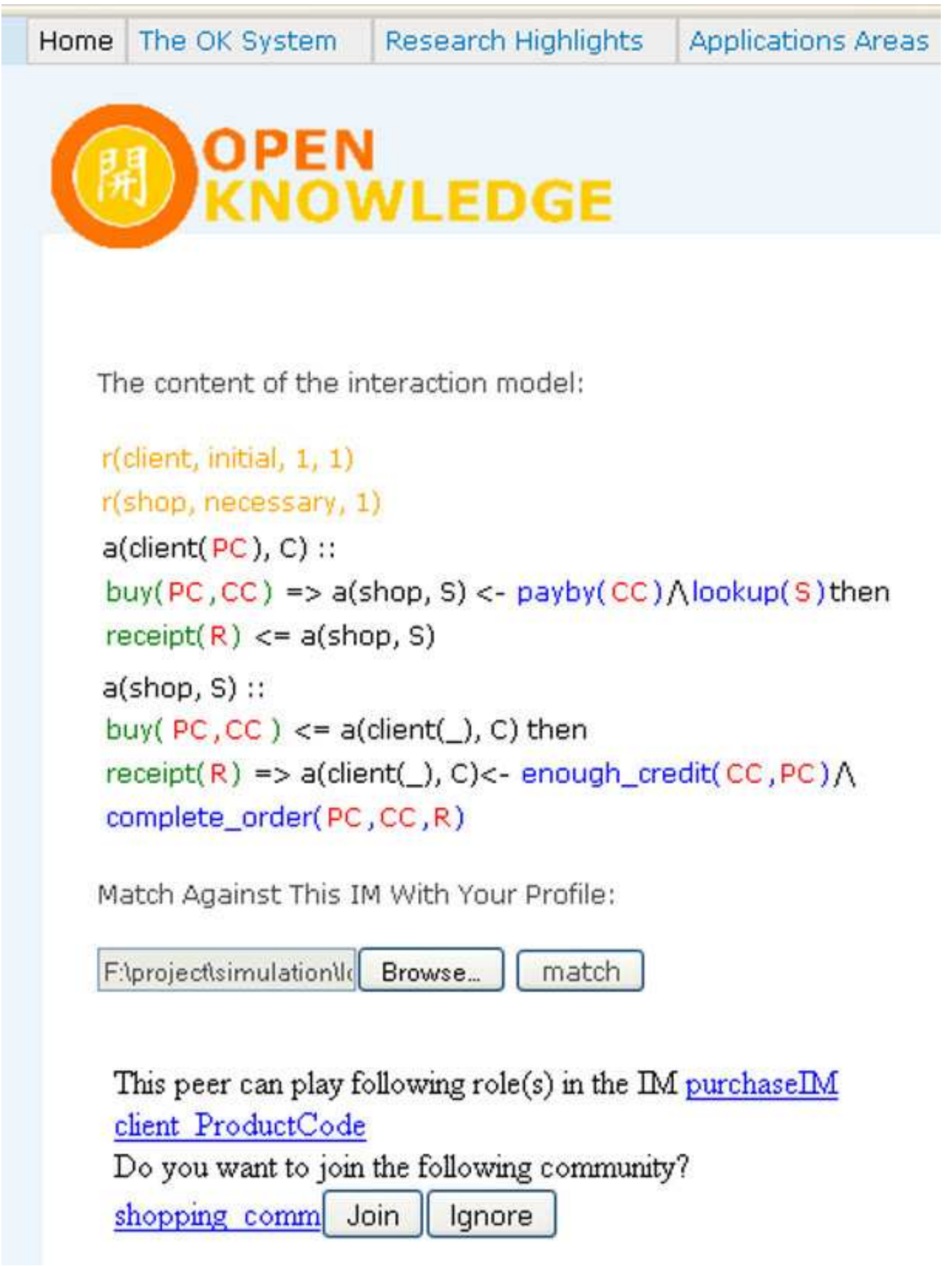
```
<!DOCTYPE html PUBLIC '-//W3C//DTD XHTML+RDFa 1.0//EN' 'http://www.w3.org
/MarkUp/DTD/xhtml-rdfa-1.dtd'>
<html xmlns='http://www.w3.org/1999/xhtml' xmlns:openk='http://www.openk.org
/openk.owl#'>
<head>
<meta content='text/html; charset=UTF-8' http-equiv='Content-Type'/>
<title>IM Publication</title>
</head>
<body>
<!--annotation starts-->
<span typeof='openk:InteractionModel'
about='[openk:myGreetingIM]'>r(greeter, initial, 1, 1)<br/>
r(greetee, necessary, 1)<br/>
<br/>
<span rel='openk:has_role'>a(greeter, A) ::<br/>
        message(Text) =&gt; a(greetee, B) &lt;- greeting(Text)</span><br/>
<span rel='openk:has_role'>a(greetee, B) ::<br/>
        message(Text) &lt;= a(greeter, A) then<br/>
        null &lt;- display_greeting(Text)</span><br/>
</span>
<!--annotation ends-->
</body>
</html>
```

http://www.openk.org/openk.owl#Role is the domain of the following property or pro

- http://www.openk.org/openk.owl#has_constraint with range: http://www.op
- http://www.openk.org/openk.owl#has_roletype with range: http://www.w3.
- http://www.openk.org/openk.owl#has_subrole with range: http://www.open
- http://www.openk.org/openk.owl#receive_msg with range: http://www.ope

FIGURE 5: Snapshot of the user interface for IM republication

By matching the user's query with the micro-data embedded in republished IMs, the previously ignored IM is discovered as shown in Figure 6. It also depicts a peer-side consumer that analyzes the discovered IM in terms of the local profile and informs the peer of which roles it can play and which communities it may join.



The content of the interaction model:

```
r(client, initial, 1, 1)
r(shop, necessary, 1)
a(client(PC), C) ::
buy(PC,CC) => a(shop, S) <- payby(CC)∧lookup(S)then
receipt(R) <= a(shop, S)
a(shop, S) ::
buy( PC,CC ) <= a(client(_), C) then
receipt(R) => a(client(_), C)<- enough_credit(CC,PC)∧
complete_order(PC,CC,R)
```

Match Against This IM With Your Profile:

F:\project\simulation\l

This peer can play following role(s) in the IM purchaseIM
client_ProductCode
Do you want to join the following community?
shopping_comm

FIGURE 6: Snapshot of the user interface for micro-data consumption

Conclusions

In this paper, we add a semantic layer to the IM by republishing it using the micro-data-embedded Web page. Republished IMs can assist peers in discovering services and collaborating peers meet their requirements more precisely due to unambiguous URIs of resources. Moreover, IM republication complies with the principles of Linked Data and will further contribute to and benefit from the Web of data. On the other hand, the IM republication provides a more secure and controllable way of transferring knowledge in the distributed environment.

[1] http://rdfa.info/rdfa-implementations

[2] http://lookup.dbpedia.org/api/search.asmx

[3] http://www.civilservice.gov.uk/jobs/