Semantic Evaluation at Large Scale Tutorial

Evaluating Semantic Search Tools

Stuart N. Wrigley
Fabio Ciravegna

Organisations, Information and Knowledge (OAK) Group
Department of Computer Science
University of Sheffield, UK

{n.surname}@dcs.shef.ac.uk
http://oak.dcs.shef.ac.uk/

The University Of Sheffield.

SEVENTH FRAMEWORK PROGRAMME

e-infrastructure

Organisations, Information and Knowledge

Outline

- Evaluation design
  - Criteria
  - Two phase approach
- Preparing a tool
  - API
  - Results format
- Running an evaluation
- Data
- Results and Analyses
- Conclusions
- Links to resources

SEARCH EVALUATION DESIGN

What do we want to do?

- Evaluate / benchmark semantic search tools
  - with respect to their semantic peers.

- What type of tools?
  - tools which load, or have access to, one or more data sources
  - tools that allow human users to answer questions (ie, have a GUI)
  - allow as wide a range of interface styles as possible

- How?
  - assess tools on basis of a number of criteria including precision, recall, usability, etc
  - automate (part) of it

Evaluation criteria

Search methodologies will be evaluated according to the following criteria:

- Query expressiveness
  - Is the style of interface suited to the type of query?
  - How complex can the queries be?

Evaluation criteria

Search methodologies will be evaluated according to the following criteria:

- Query expressiveness
- Usability (effectiveness, efficiency, satisfaction)

  - How easy is the tool to use?
  - How easy is it to formulate the queries?
  - How easy is it to work with the answers?

Evaluation criteria

Search methodologies will be evaluated according to the following criteria:

- Query expressiveness
- Usability (effectiveness, efficiency, satisfaction)
- Scalability

  - Ability to cope with a large ontology
  - Ability to query a large repository in a reasonable time
  - Ability to cope with a large amount of results returned

Evaluation criteria

Search methodologies will be evaluated according to the following criteria:

- Query expressiveness
- Usability (effectiveness, efficiency, satisfaction)
- Scalability
- Performance

Resource consumption:
- execution time (speed)
- CPU load
- memory required

Two phase approach

- Semantic search tools evaluation demands a user-in-the-loop phase
  - usability criterion

- Two phases:
  - User-in-the-loop
  - Automated
  - Not compulsory to participate in both

Evaluation criteria

Each phase will address a different subset of criteria.

- **Automated evaluation**: query expressiveness, scalability, performance, quality of documentation
- **User-in-the-loop**: usability, query expressiveness

PREPARING A TOOL

API

- A range of information needs to be acquired from the tool in both phases
- In automated phase, the tool has to be executed and interrogated with **no** human assistance.
- Interface between the SEALS platform and the tool must be formalised

API – common

- **Load ontology**

  ```
  boolean loadOntology(URI ontology, String ontologyName,
  String ontologyNamespace)
  ```

  - success / failure informs the interoperability
- **Results ready?**

  ```
  boolean isResultSetReady()
  ```

  - used to determine execution time
- **Get results**

  ```
  String getResults()
  ```

  - list of URIs (number of results to be determined by developer)
- **Show GUI**

  ```
  void showGUI(boolean show)
  ```

  - switches the graphical user interface on or off

API – user in the loop

- **User query input complete?**
  `boolean isUserInputComplete()`
  - used to determine input time
- **Get user query**
  `String getUserQuery()`
  - String representation of user's query
  - if NL interface, same as text inputted
- **Get internal query**
  `String getInternalQuery()`
  - String representation of the internal query
  - for use with…

API – automated

- **Execute query**
  `boolean executeQuery(String query)`
  - format agnostic – it's just a `String`
  - mustn't constrain tool type to particular format
  - tool provider given questions shortly before evaluation is executed
  - tool provider converts those questions into some form of 'internal representation' which can be serialised as a `String`
  - serialised internal representation passed to this method

Results format

- SPARQL Query Results XML Format (W3C Recommendation 15 January 2008)

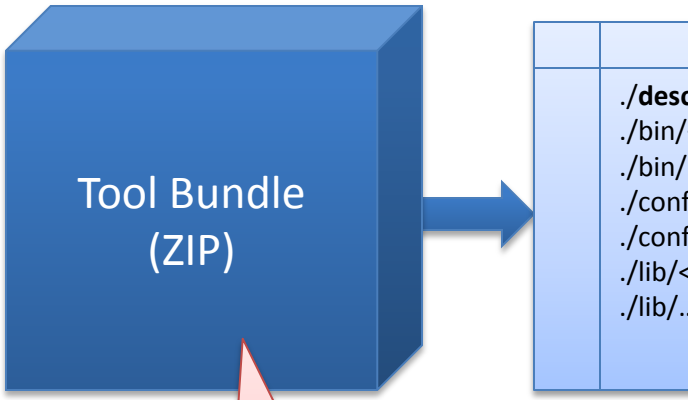
```
<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
    <variable name="questionAnswer"/>
  </head>
  <results>
    <result>
      <binding name="questionAnswer">
        <uri>http://www.ifi.uzh.ch/ddis/evoont/2008/11/som/parsed/org.eclipse.compare_v20020205#getTitle.</uri>
      </binding>
    </result>
    <result>
      <binding name="questionAnswer">
<uri>http://www.ifi.uzh.ch/ddis/evoont/2008/11/som/parsed/org.eclipse.compare_v20020205#getString.Ljava.lang.String._2</uri>
      </binding>
    </result>
  </results>
</sparql>
```

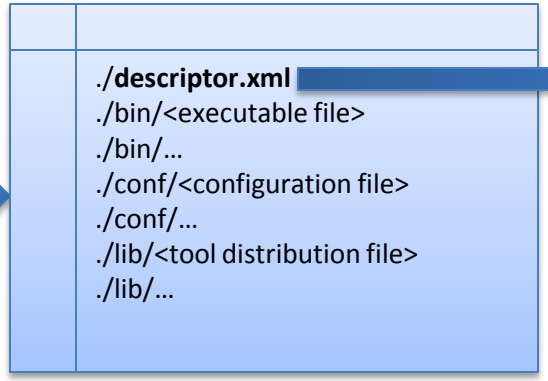
Connecting a search tool

- Implementation of a Java plugin with:
  - Tool Management API
    - Deployment
    - Undeployment
    - Start (optional)
    - Stop (only if start)

    } Java apps or shell scripts
  - Tool invocation API
    - `loadOntology`
    - `executeQuery`
    - *etc*

Packaging my tool





Maven archetype for tool bundle creation

Tool bundle description	
Description	• Id • Version • License • ...
Wrapper	• Tool management • Tool invocation • Dependencies
Requirements	• Modules • Configuration files

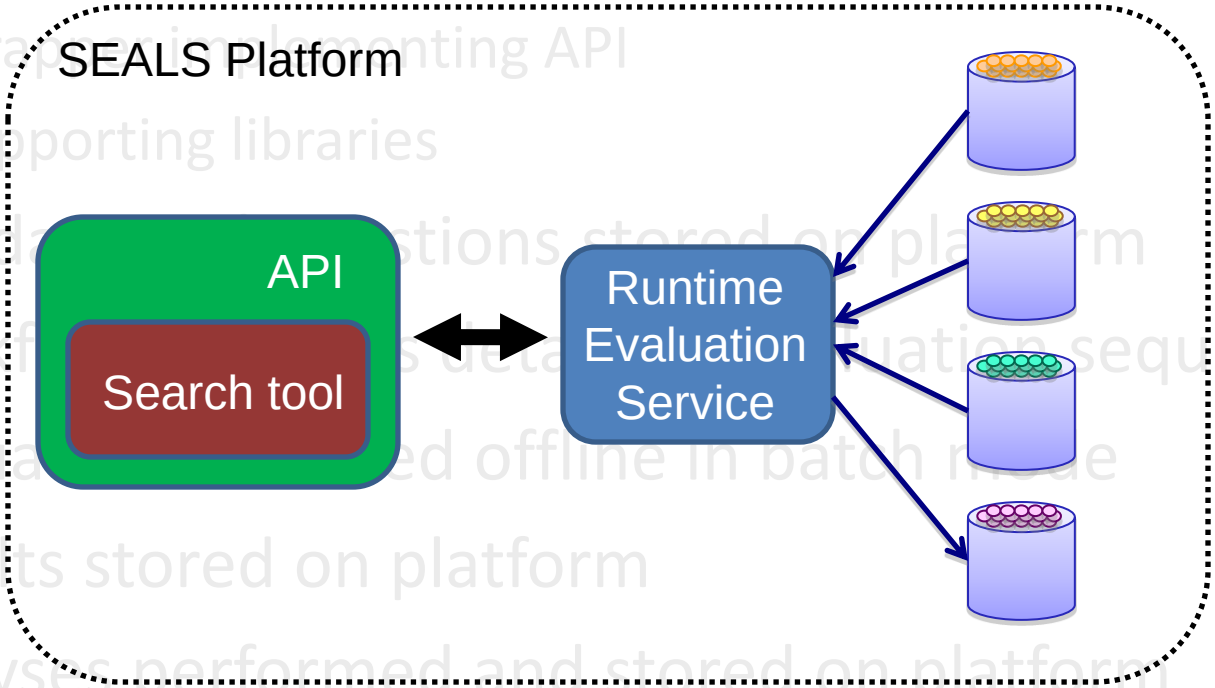
RUNNING THE EVALUATION

Automated evaluation

- Tools uploaded to platform. Includes:
  - wrapper implementing API
  - supporting libraries
- Test data and questions stored on platform
- Workflow specifies details of evaluation sequence
- Evaluation executed offline in batch mode
- Results stored on platform
- Analyses performed and stored on platform

Automated evaluation

- Tools uploaded to platform. Includes:
  - wrapper implementing API
  - supporting libraries
- Test data and configurations stored on platform
- Workflow specifies data and evaluation sequence
- Evaluations executed offline in batch mode
- Results stored on platform
- Analyses performed and stored on platform

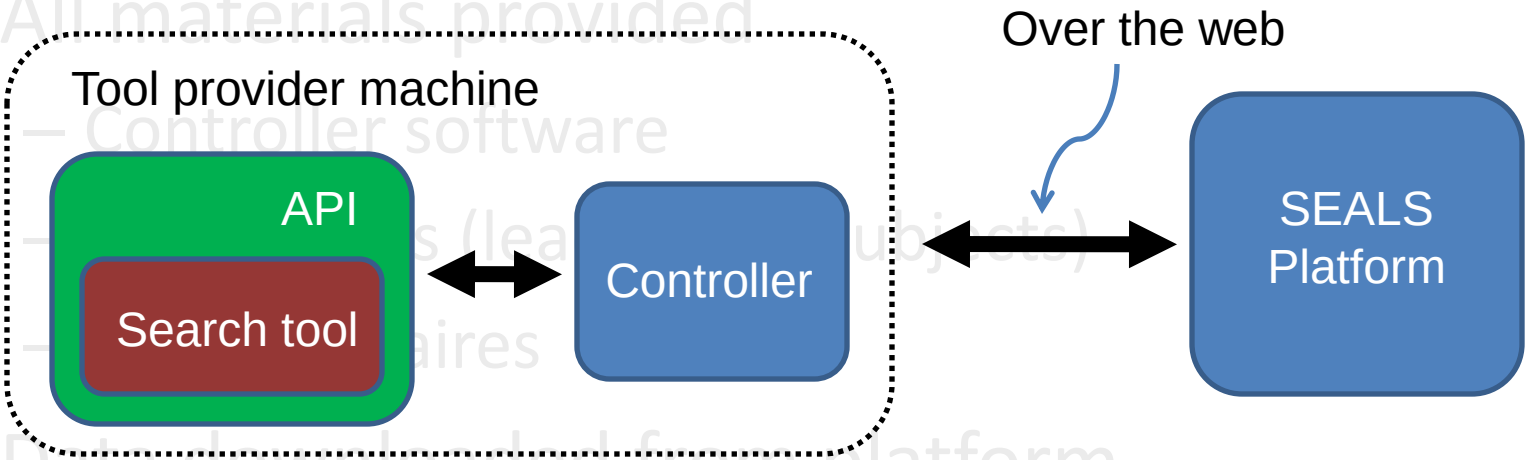


User in the loop evaluation

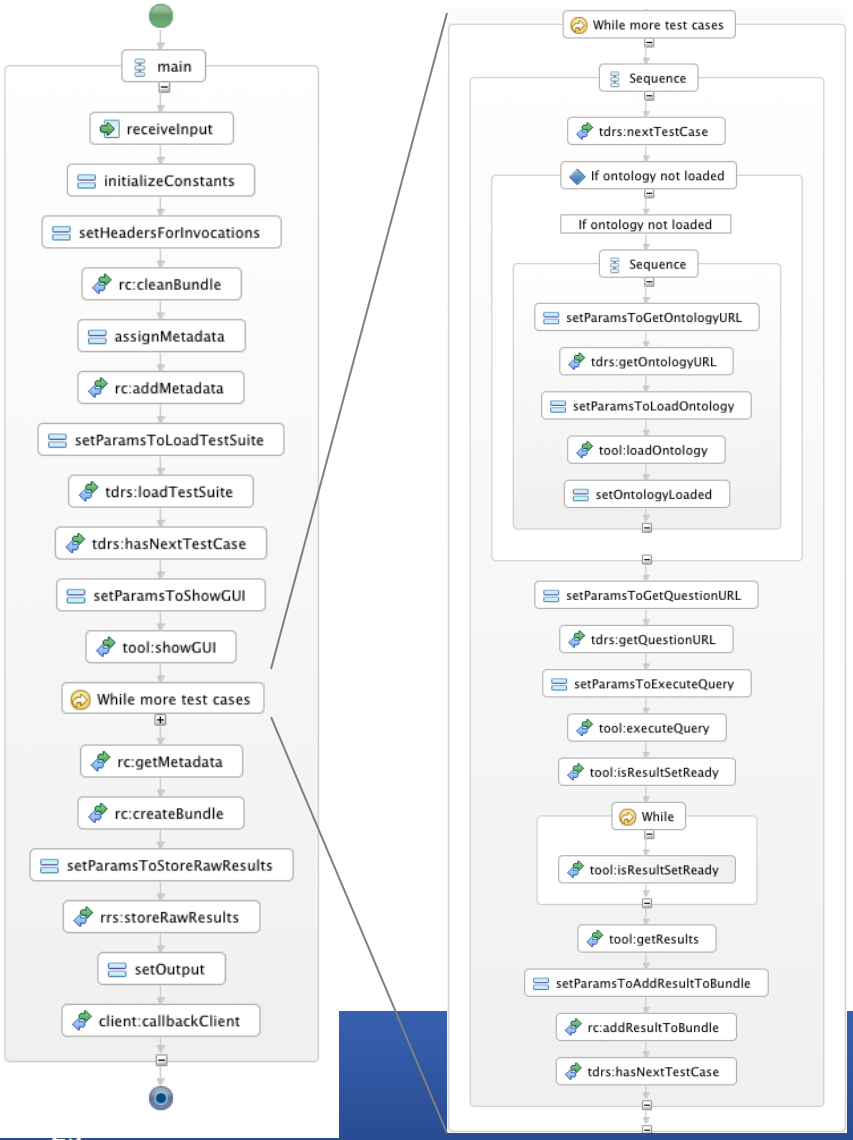
- Performed at tool provider site
- All materials provided
  - Controller software
  - Instructions (leader and subjects)
  - Questionnaires
- Data downloaded from platform
- Results uploaded to platform

User in the loop evaluation

- Performed at tool provider site
- All materials provided
  - Controller software
  - [illegible]s (lea[illegible]ubjects)
  - [illegible]aires
- Data downloaded from platform
- Results uploaded to platform



Evaluation workflow (automated)



- BPEL workflow using
  - Platform services:
    - Test Data Repository
    - Result Repository
    - Result bundling
  - External services
    - Tool invocation
    - Interpretation
    - Any other custom service

DATA

Data set – user in the loop

- Mooney Natural Language Learning Data
  - used by previous semantic search evaluation
  - simple and well-known domain
  - using geography subset
    - 9 classes
    - 11 datatype properties
    - 17 object properties and
    - 697 instances
  - 877 questions already available

Data set – automated I

- EvoOnt
  - set of object-oriented software source code ontologies
  - easy to create different ABox sizes given a TBox
  - 5 data set sizes: 1k, 10k, 100k, 1M, 10M triples
  - questions generated by software engineers

Data set – automated II

- QALD
  - Used in *Question Answering Over Linked Data* 2011 challenge
  - two RDF datasets: DBpedia 3.6 and MusicBrainz
  - 50 training questions per dataset
    - NL and SPARQL
  - 50 test questions per dataset
    - NL

RESULTS AND ANALYSES

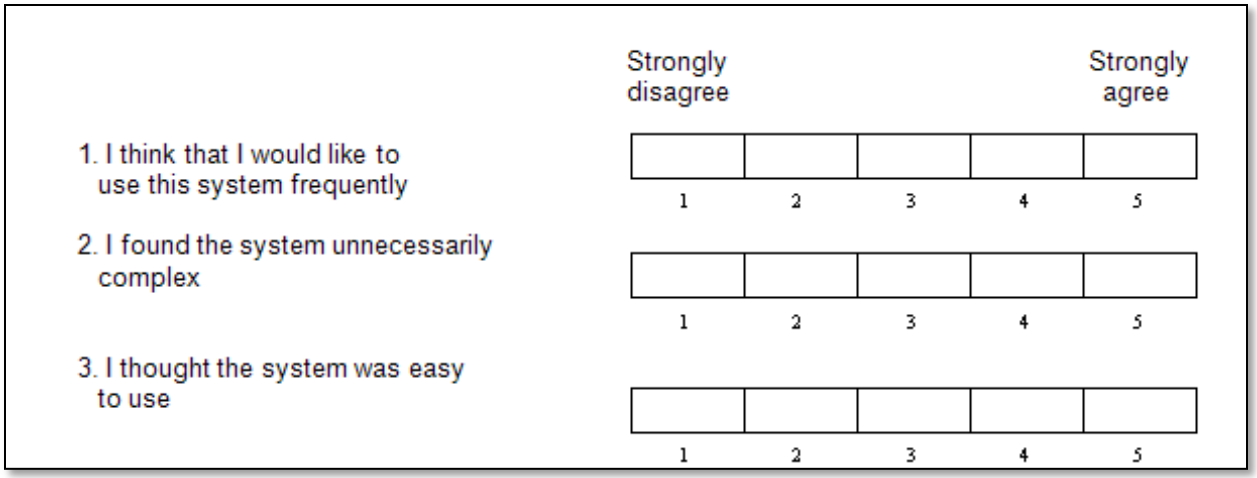
User-in-the-loop

3 questionnaires:

- SUS questionnaire
- Extended questionnaire
  - similar to SUS in terms of type of question but more detailed
- Demographics questionnaire

System Usability Scale (SUS) score

- SUS is a *Likert* scale
- 10-item questionnaire
- Each question has 5 levels (*strongly disagree* to *strongly agree*)
- SUS scores have a range of 0 to 100.
- A score of around **60** and above is generally considered as an indicator of good usability.



Automated

Results

- Execution success (OK / FAIL / PLATFORM ERROR)
- Triples returned
- Time to execute each query
- CPU load, memory usage

Analyses

- Ability to load ontology and query (interoperability)
- Precision, Recall and F-Measure (search accuracy and query expressiveness)
- Tool robustness: ratio of all benchmarks executed to number of failed executions
- Average timings

User in the loop

Results (other than core results similar to automated phase)

- Query captured by the tool
- Underlying query (e.g., SPARQL)
- Is answer in result set? (user may try a number of queries before being successful)
- time required to obtain answer
- number of queries required to answer question

Analyses

- Precision, Recall and F-Measure
- Average timings
- Correlations between results and SUS scores, demographics, etc

Dissemination of outcomes

- Results and interpretations browsable on the SEALS portal

- Split into three areas:
  - performance
  - usability
  - comparison between tools

- Workshop at ESWC 2012

CONCLUSIONS

Conclusions

- Methodology and design of a semantic search tool evaluation campaign
- Exists within the wider context of the SEALS initiative
- Feedback encouraged
  - still the chance to influence the details of the campaign
- Emphasis on the user experience (for search)
  - Two phase approach

Get involved!

- Second Evaluation Campaign in all SEALS technology areas this Summer through to Spring 2012
- Get involved – your input and participation is crucial

- Workshop planned for ESWC 2012 after campaign

- Find out more (and take part!) at:
  http://www.seals-project.eu
  or talk to me, or email me (s.wrigley@dcs.shef.ac.uk)

Links to resources

- **Evaluation campaign**
  - http://www.seals-project.eu/seals-evaluation-campaigns/semantic-search-tools
- **Test suites**
  - Mooney (NL questions)
    - http://seals.sti2.at/tdrs-web/testdata/persistent/Mooney+NL+Testsuite/2010/
  - EvoOnt (SPARQL questions)
    - http://seals.sti2.at/tdrs-web/testdata/persistent/EvoOnt+1K+SPARQL+Testsuite/2010/
    - http://seals.sti2.at/tdrs-web/testdata/persistent/EvoOnt+10K+SPARQL+Testsuite/2010/
    - http://seals.sti2.at/tdrs-web/testdata/persistent/EvoOnt+100K+SPARQL+Testsuite/2010/
    - http://seals.sti2.at/tdrs-web/testdata/persistent/EvoOnt+1000K+SPARQL+Testsuite/2010/
    - http://seals.sti2.at/tdrs-web/testdata/persistent/EvoOnt+10000K+SPARQL+Testsuite/2010/
  - QALD
    - Coming soon!
- **Last year's campaign outcomes**
  - http://www.seals-project.eu/seals-evaluation-campaigns/semantic-search-tools/results-2010