WSCOLAB
Structured Collaborative Tagging for Web Service Matchmaking

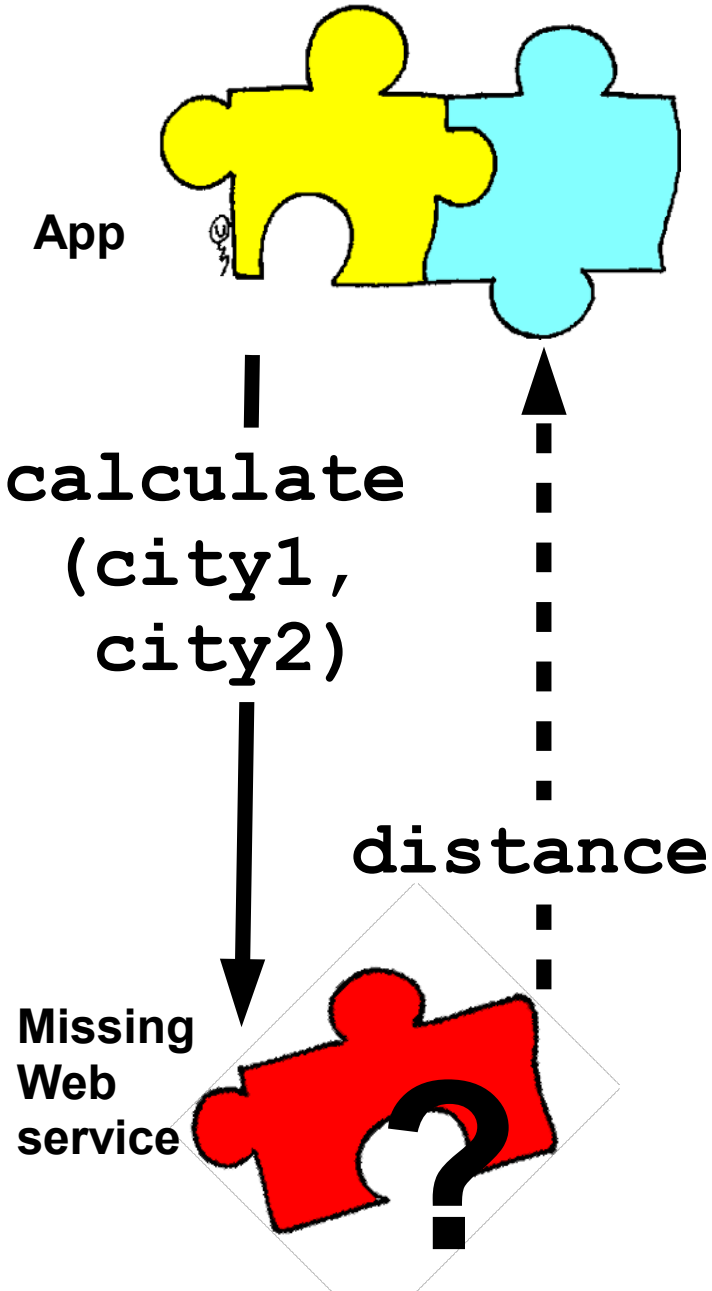
Maciej Gawinecki, Giacomo Cabri

University of Modena and Reggio-Emilia, Italy

Marcin Paprzycki, Maria Ganzha

Polish Academy of Sciences, Poland

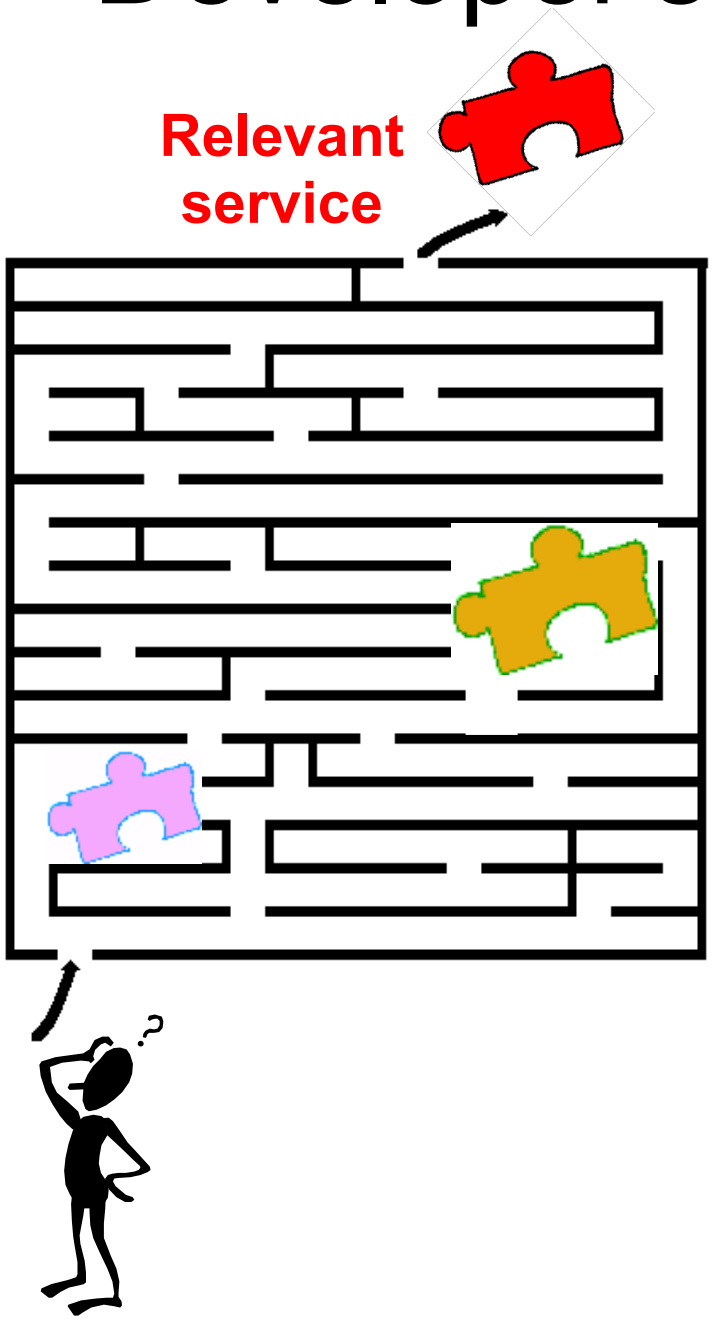
Developer's perspective



"A developer is looking for a service that computes the driving distance in miles between two cities worldwide.

The input should include names of the cities, and optionally their states and countries."

Developer's search for service



Search for service is a part of software requirements specification

Starts from exploration of domain often to a developer

Many candidates need to be examined if they are not described enough

Service broker's perspective

How to describe Web services so developers can find them?

What information to put in?

Where to take it from?

How to express it?

How to do it *efficiently* for large number of Web services?

Repository

Service descriptions need to be relevant for developer's search criteria

equivalent functionality

"A developer is looking for a service that computes the driving distance in miles between two cities worldwide.

equivalent functionality scope

compatible interface

The input should include names of the cities, and optionally their states and countries."

Extracting information from WSDLs (descriptions from service providers)

```
<service>
  <documentation>Returns an estimated distance between
    two given locations. Works worldwide. </documentation>

  <input><parameter name="Location1">
    Location of type geographic point: Latitude and
      longitude of the first location. </parameter>
    ...
  </input>

  <output><parameter name="distance">
     The estimated distance between the given locations in
        miles, km and feet.</parameter>
  </output>
</service>
```

Extracting information from WSDLs (descriptions from service providers)

```
<service>
  <documentation>Returns an estimated distance between
    two given locations. Works worldwide. </documentation>

  <input><parameter name="Location1">
    Location of type geographic point Latitude and
    longitude of the first location. </parameter>
    ...
  </input>

  <output><parameter name="distance">
    The estimated distance between the given locations in
    miles, km and feet. </parameter>
  </output>
</service>
```

Tag cloud: a formalism to describe aspects relevant for a user

- Tags are for Web service categorization
- A service may belong to more than one category
- The bigger the tag, the more relevant it is for the service
- *Is Web service computing a distance for cities or returning cities for some distance?*

geocoding
geographical
information metropolis
distance_miles
geographic geography
driving_direction global

state location
geographic_point sity
city_name

distance coordinates
miles **length** map

Structured tag clouds: seperation of behaviour from interface

- Useful for effective querying

geocoding
geographical
information metropolis
distance_miles
geographic geography
driving_direction global

behaviour

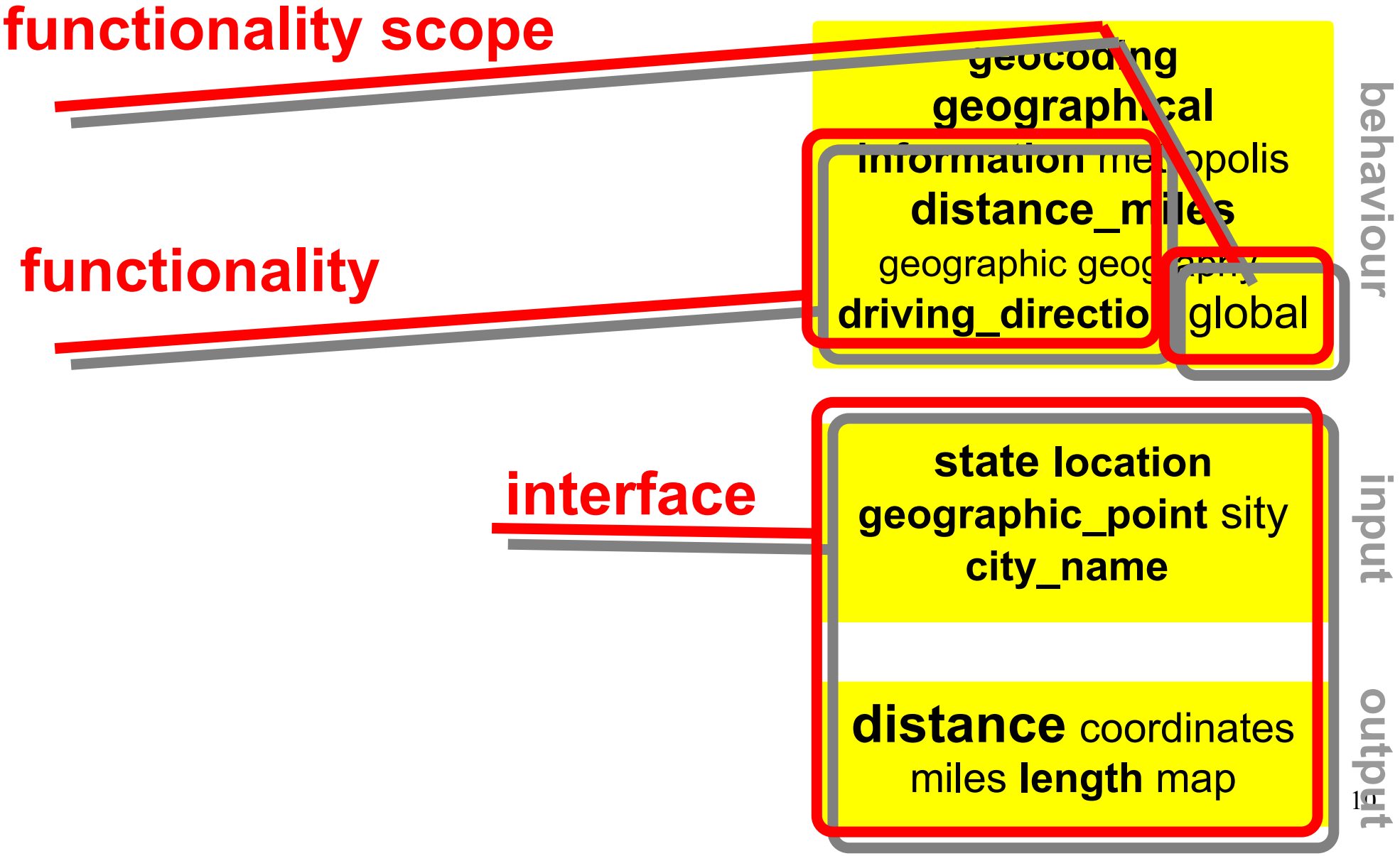
state location
geographic_point sity
city_name

input

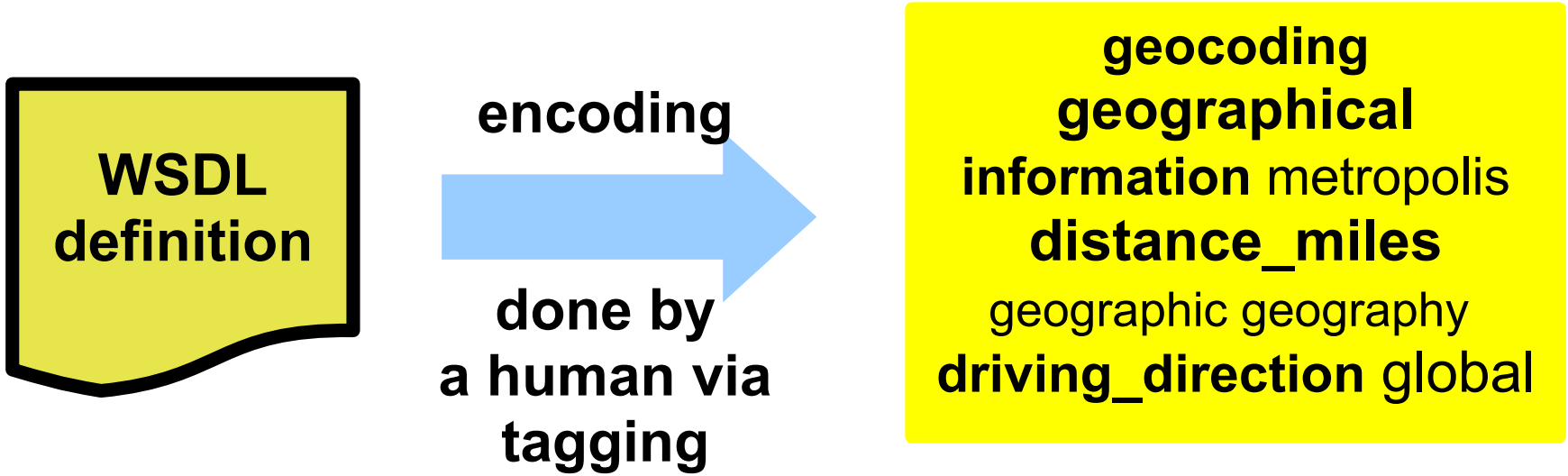
distance coordinates
miles **length** map

output

Structured tag clouds: seperation of behaviour from interface

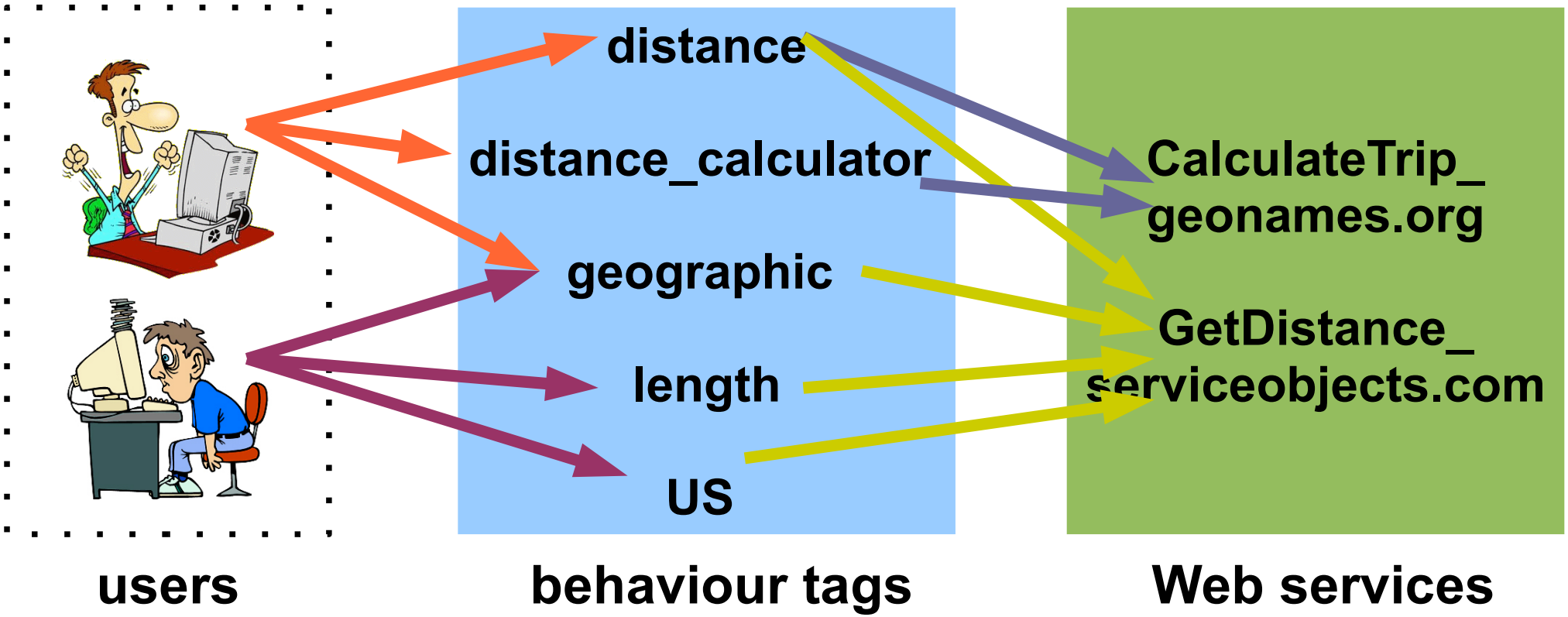


Reasons for using a human not machine to extract information



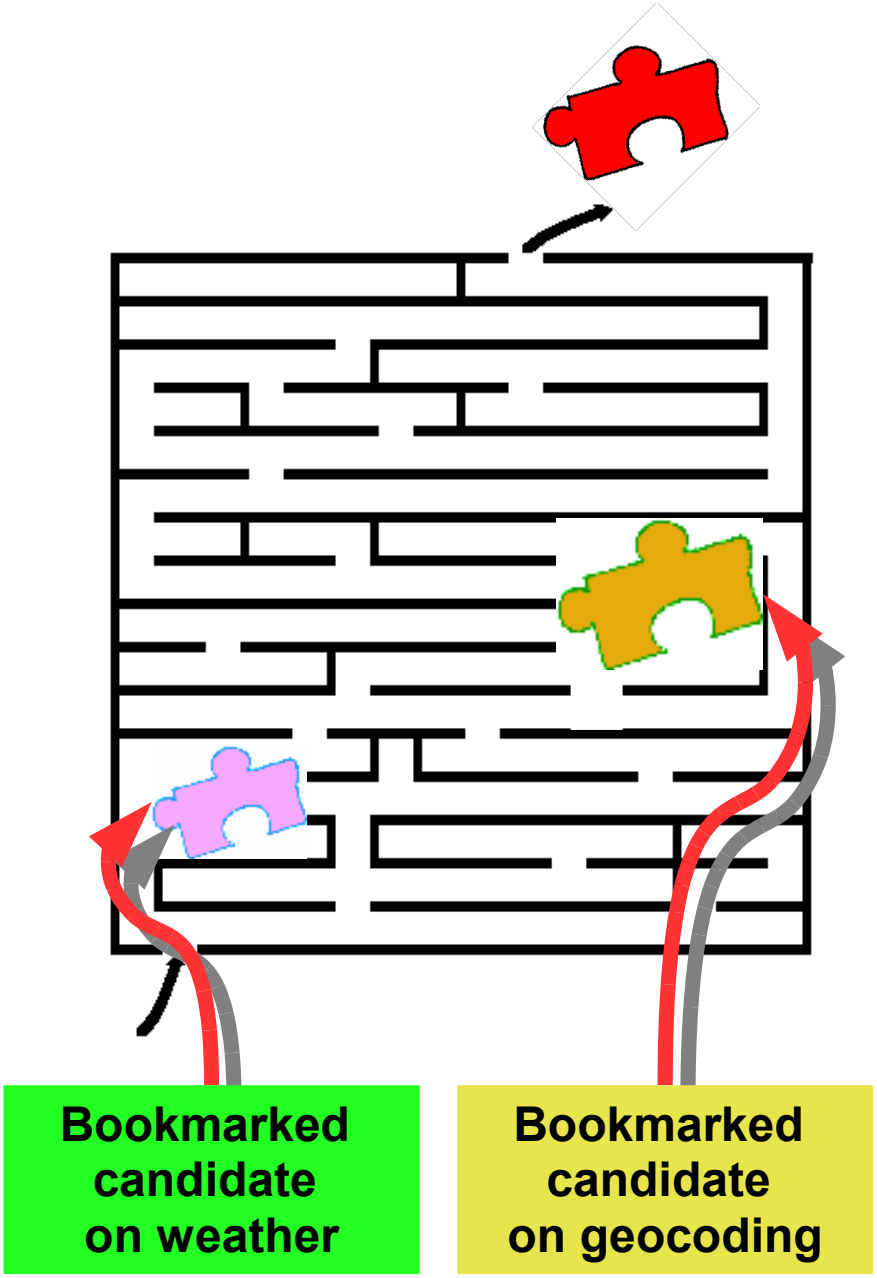
- Classification requires intelligence, e.g. *geocoding*
- No vocabulary problem:
  - a human authority tags an object with such words that can be reused to recall it

Inviting community to annotatote: ***Structured*** collaborative tagging



- Community can describe and classify large number of services more efficiently than a single authority
- Annotations from diverse members of the community emerge into descriptions of Web services

Motivation: social bookmarking



A developer...

Bookmarks a service candidate to keep a *reference* to it.

Annotates a collection of her bookmarked services to *organize* it better.

Shares her bookmarks with everyone else hoping other do the same to *access* larger collection.

Tool
Web Service Tagging Portal

WScolab BEHAVIOUR: INPUT: OUTPUT: Search for service! Hi Maciej! | Logout | Services to classify

Tag and classify Web service

You classified it as:

GeoNames_FindNearbyWikipedia2 RELEVANT | Tag/classify again!

Provider's description

How the provider describes this service:

BEHAVIOUR

Find Wikipedia articles localized close to the given location (identified by a country and a postal code).

INPUT PARAMETERS

- country code -- The ISO code of the country that the returned articles should be localized to.
- lang -- The language returned articles should be in. Optional, default=en.
- maxRows -- Determines the maximun number of places to be returned (optional, default = 5).

OUTPUT PARAMETERS

- distance -- The distance of the localization of the article to the input location
- lat -- The latitude of the geographic position the wikipedia article is localized to.
- lng -- The longitude of the geographic position the wikipedia article is localized to.
- Wikipedia articles -- Wikipedia articles including title, language, summary, link to full

My tags

RULE #1: Try to be specific when describing a Web service. When there are 2 or more different services about weather forecast and only one of them provides weather for whole globe or differs at some points from other similar services, try to underline that fact, e.g. by tags: *worldwide, global*.

RULE #2: Tags should be **space separated**.Use underscore _ to separate words in multi-word tags, e.g. *zip_code*.

BEHAVIOUR TAGS

test

What keywords would you use when quering for the service like this?

Tags recommended by community: Wikipedia article location postal_code country articles

All my behaviour tags: test

INPUT TAGS

What information do you need to have to use this service?

Tags recommended by community: country postal_code country_code radius language zip_code

Developer fomulates service request as *structured keyword query*

calculate_distance distance_miles miles miles_distance driving_direction driving worldwide global

city city_name state location_one location_two country

distance length miles distance_in_miles

"A developer is looking for a service that computes the driving distance in miles between two cities worldwide.

The input should include names of the cities, and optionally their states and countries."

Matchmaking: finding services with matching behaviour and interface

calculate_distance distance_miles miles miles_distance driving_direction driving worldwide global	geocoding geographical information metropolis distance_miles geographic geography driving_direction global	behaviour
city city_name state location_one location_two country	state location geographic_point sity city_name	input
distance length miles distance_in_miles	distance coordinates miles length map	output

#1 Service with matching behaviour, but *incompatible* interface

calculate_distance distance_miles miles miles_distance **driving_direction** driving worldwide **global**	geocoding **geographical** information metropolis **distance_km** geographic geography **driving_direction** global	behaviour
city city_name state location_one location_two country	location zip zip_code post_code	input
distance **length** miles distance_in_miles	**distance** coordinates km **length** map	output

#2 Service with matching interface, but *non-equivalent* behaviour

calculate_distance
distance_miles miles
miles_distance
driving_direction
driving worldwide
global

geocoding
geographical
calculate_distance
linear_distance
spherical_distance
global

behaviour

city **city_name** **state**
location_one
location_two country

state location
geographic_point sity
city_name

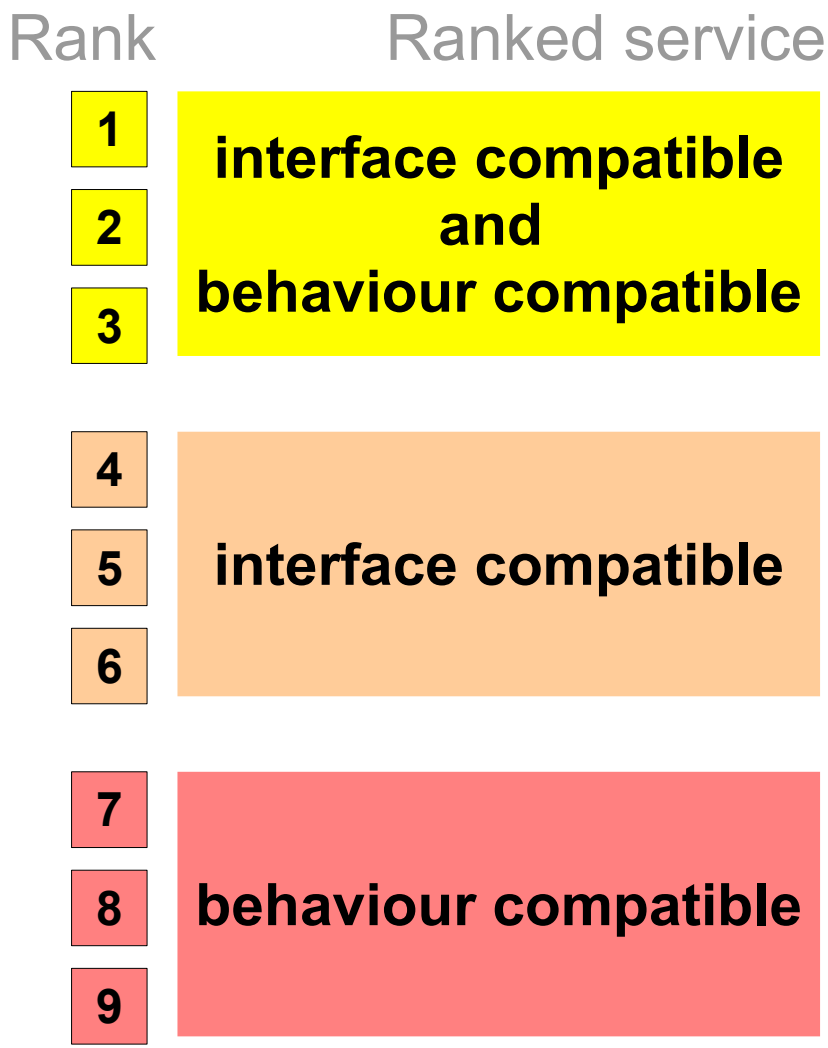
input

distance **length** **miles**
distance_in_miles

distance coordinates
miles **length** map

output

Matchmaking mechanism: ranking selected services

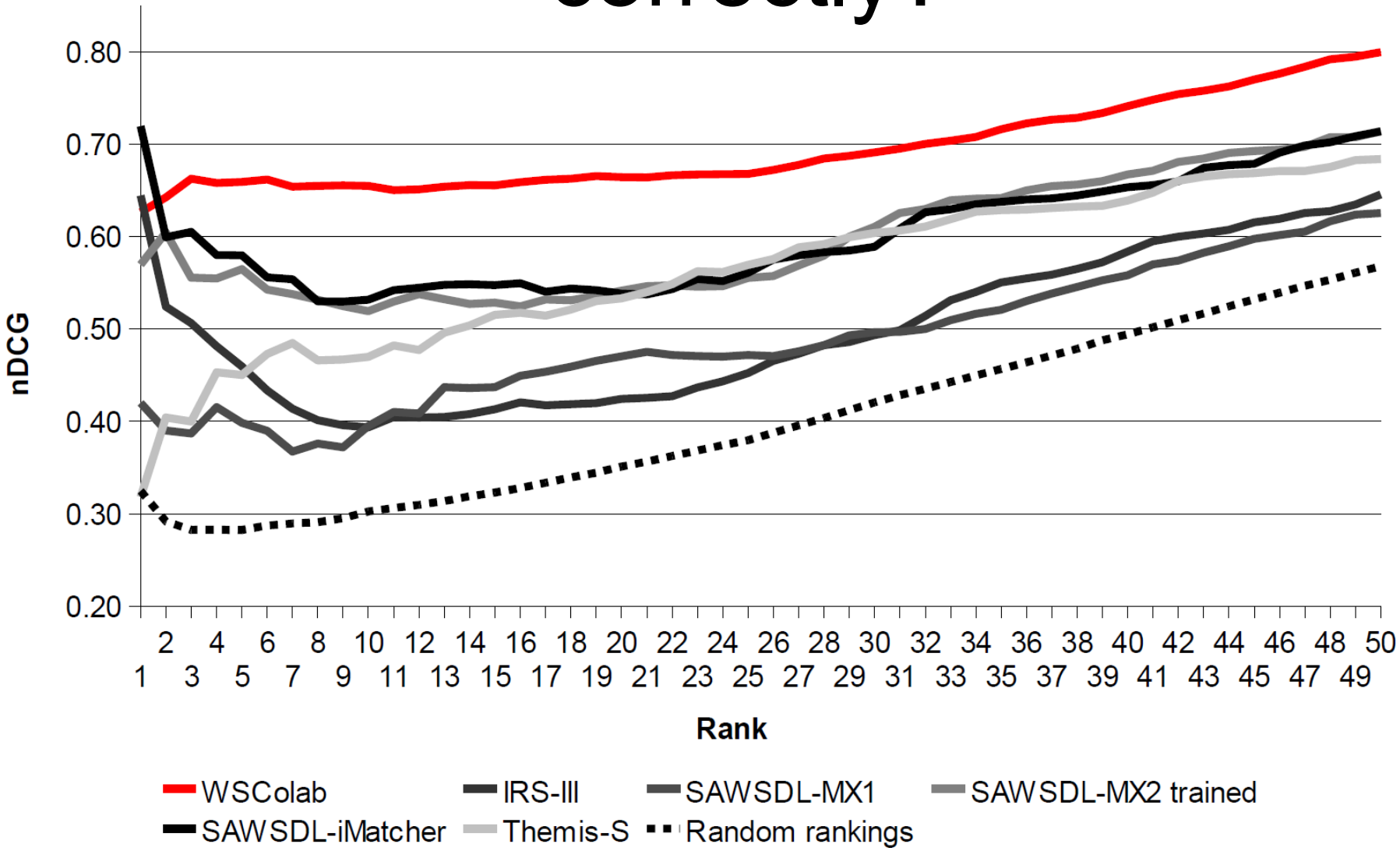


- Service selection:
  - interface compatible
    - matching input/output tag
  - behavior compatible
    - matching behaviour tag
- Service rank: combination of ranks for each facet (*input*, *output*, *behaviour*)
  - idea: more tags overlaping with query → higher rank
    - TF-IDF weighting schema

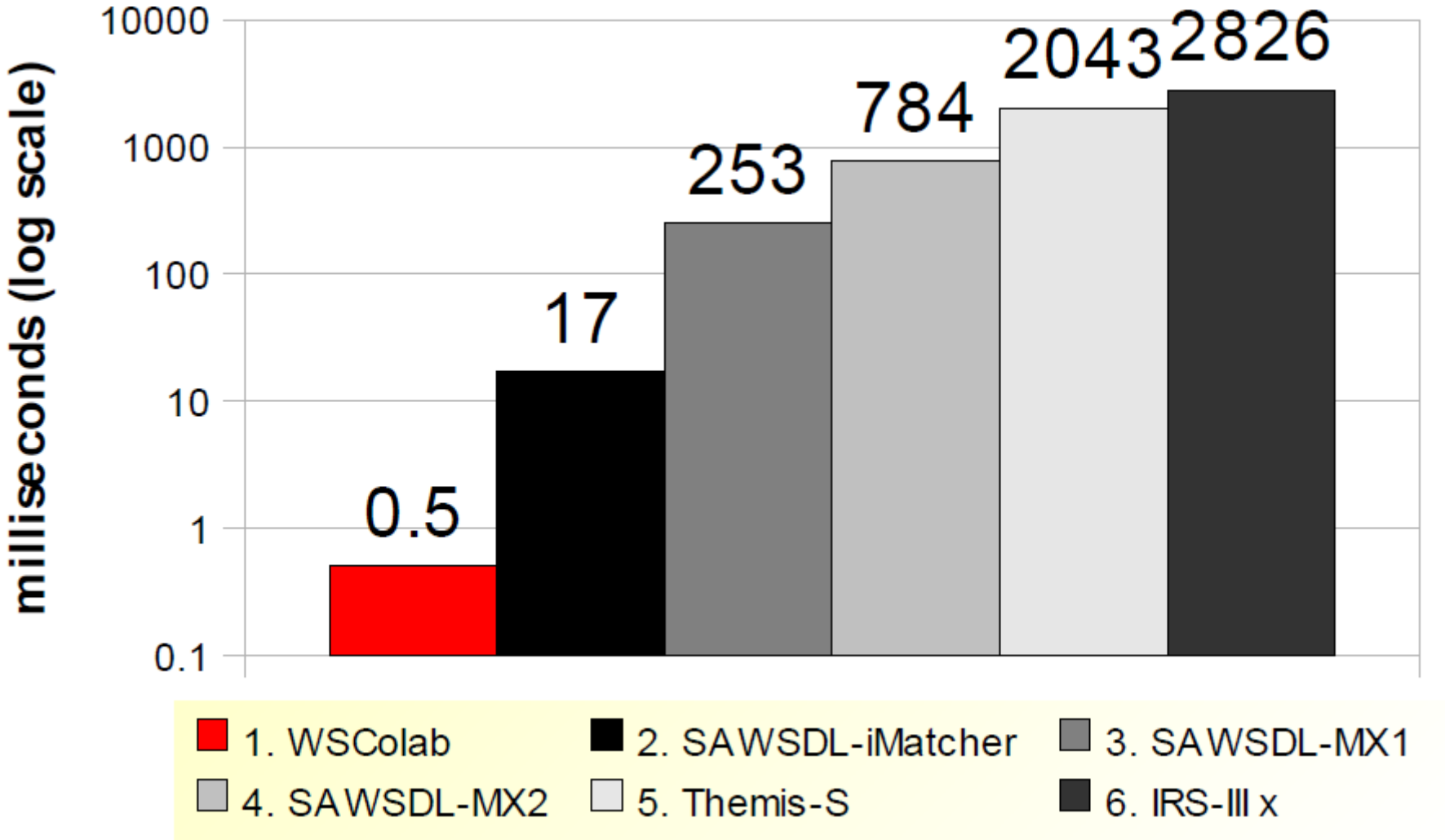
How our approach compares to other annotation formalisms?

- Evaluation: Semantic Service Selection Contest
  - competition on matchmaking effectiveness and efficiency and annotation effort
    - 6 service matchmakers with different annotation formalisms: SAWSDL, OCML, collaborative tags, eTVSM
  - evaluated over the same test collection
    - 50 service candidates + 9 service requests
- Our annotation:
  - Collaborative tagging portal:
    - 2541 annotations collected from 27 users in 12 days
    - query formulations from 5 different users

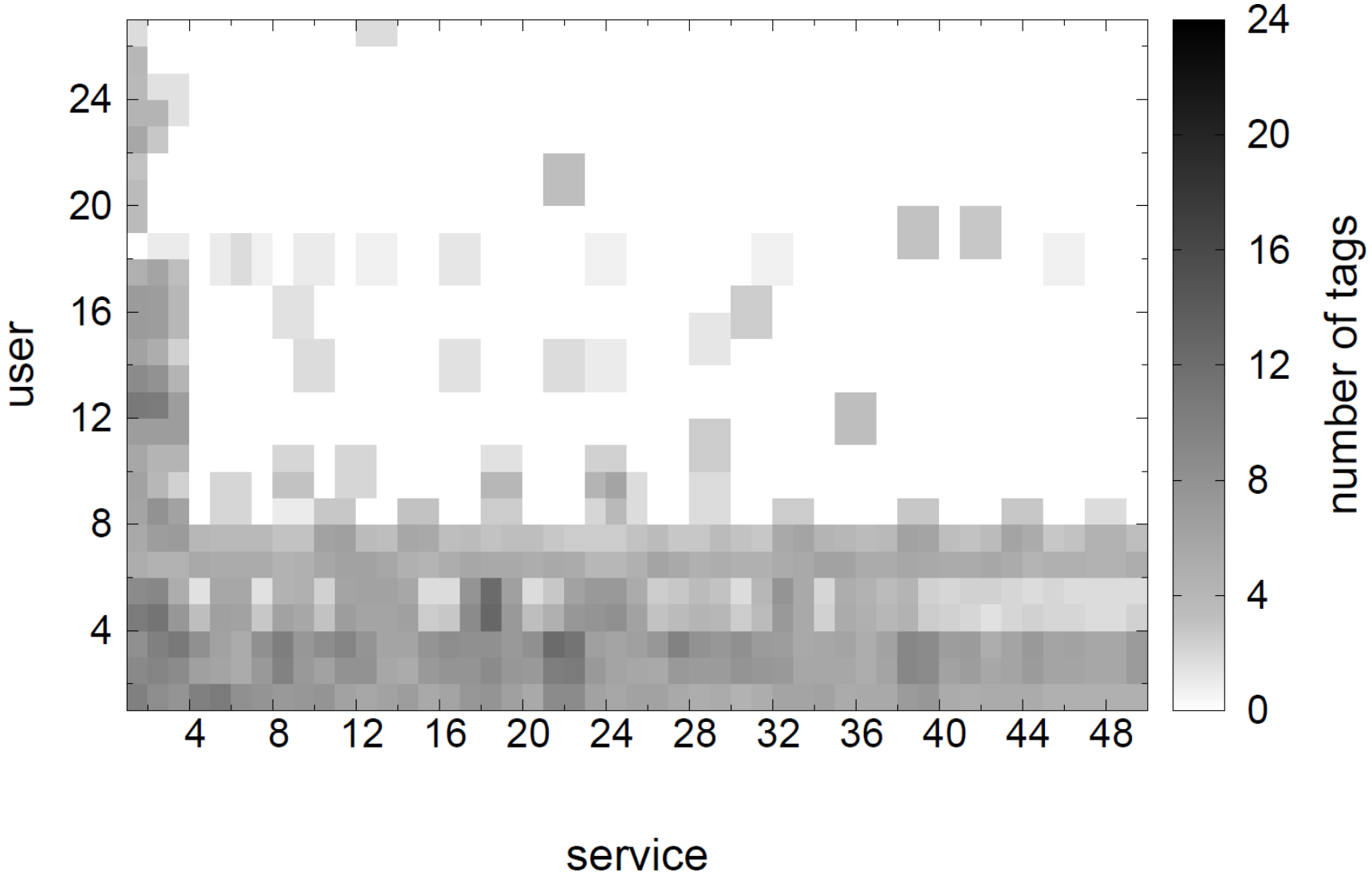
Which matchmaker ranks results correctly?



Which matchmaker returns results in the shortest time?



Does users contribute equally to annotation of the same services?



Conclusions

- **Problem:** Lack of efficient and effective categorizing approach for repositories with large number of Web services
- **Solution:** Structured collaborative tagging for describing Web services + Web service matchmaker using tag clouds for evaluating behaviour and interface compatibility
- **Evaluation:** Good trade-off between annotation complexity and retrieval effectiveness but some services might remain undescribed and thus be difficult to find

Thank you!

- Do you have some questions?

- WSColab collaborative tagging portal:
  - http://mars.ing.unimo.it/wscolab/new.php
- Semantic Service Selection Contest 2009 results (Cross-evaluation Track):
  - **http://fusion.cs.uni-jena.de/professur/jgdeval/jgdeval-at-s3-contest-2009-results**