DIFFERENT APPROACHES OF WEB SERVICES DISCOVERY

Ajinkya Kolhe

Jinesh Dhruv

Nisha Bhanushali

Introduction

ABOUT WEB SERVICE DISCOVERY

- PROCESS OF RETRIEVING WEB SERVICE DESCRIPTION FROM THE WEB SERVICE REGISTRY AND LOCATING THE DESCRIPTION PUBLISHERS.

IMPORTANCE OF WEB SERVICE DISCOVERY

- DISCOVERY APPROACHES HELPS THE USERS TO QUICKLY GET ACCESS TO THE WEB SERVICES.

CHALLENGES IN WEB SERVICE DISCOVERY

- TO FIND AN EFFICIENT AND SYSTEMATIC APPROACH TO FIND THE BEST POSSIBLE METHODS AND ALGORITHMS TO FIND A WEB SERVICE THAT PERFECTLY CATERS TO CONSUMERS NEED.

Categories of Research Papers

1. Web Services related to Big Data Analytics Domain
2. Web Services using Social Network Principles
3. Semantic Approaches
4. Discovery using Mining & Multi-Level index

1. WEB SERVICE DISCOVERY IN BIG DATA ANALYTICS DOMAIN

Problem Statement:

- To build a automation service composition in the Big Data Analytics Domain
- To perform service discovery to perform each step of the big data analytics.

APPROACHES:

- Domain Ontology based Service discovery

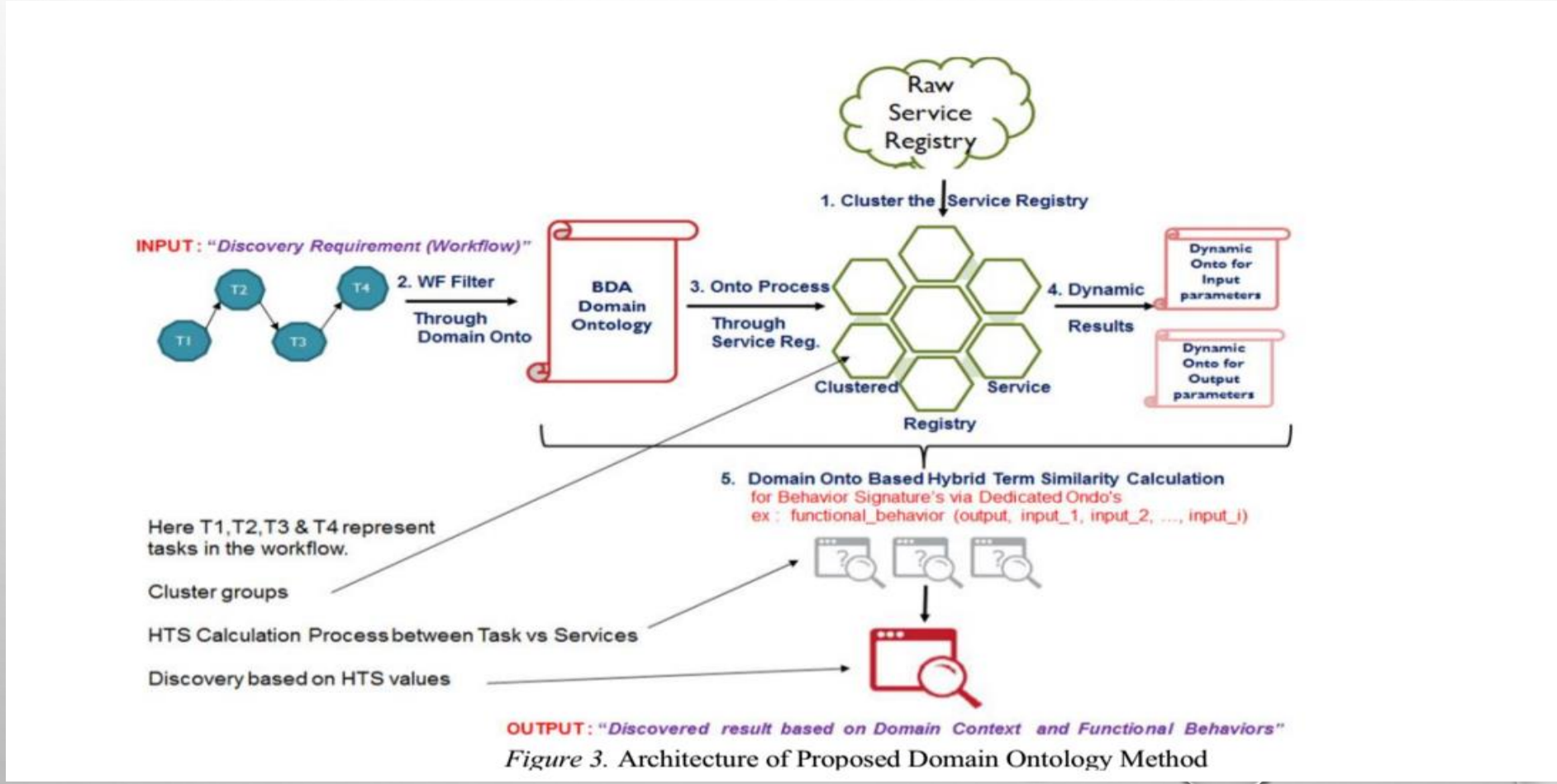


Figure 3. Architecture of Proposed Domain Ontology Method

- LSSN Based Discovery

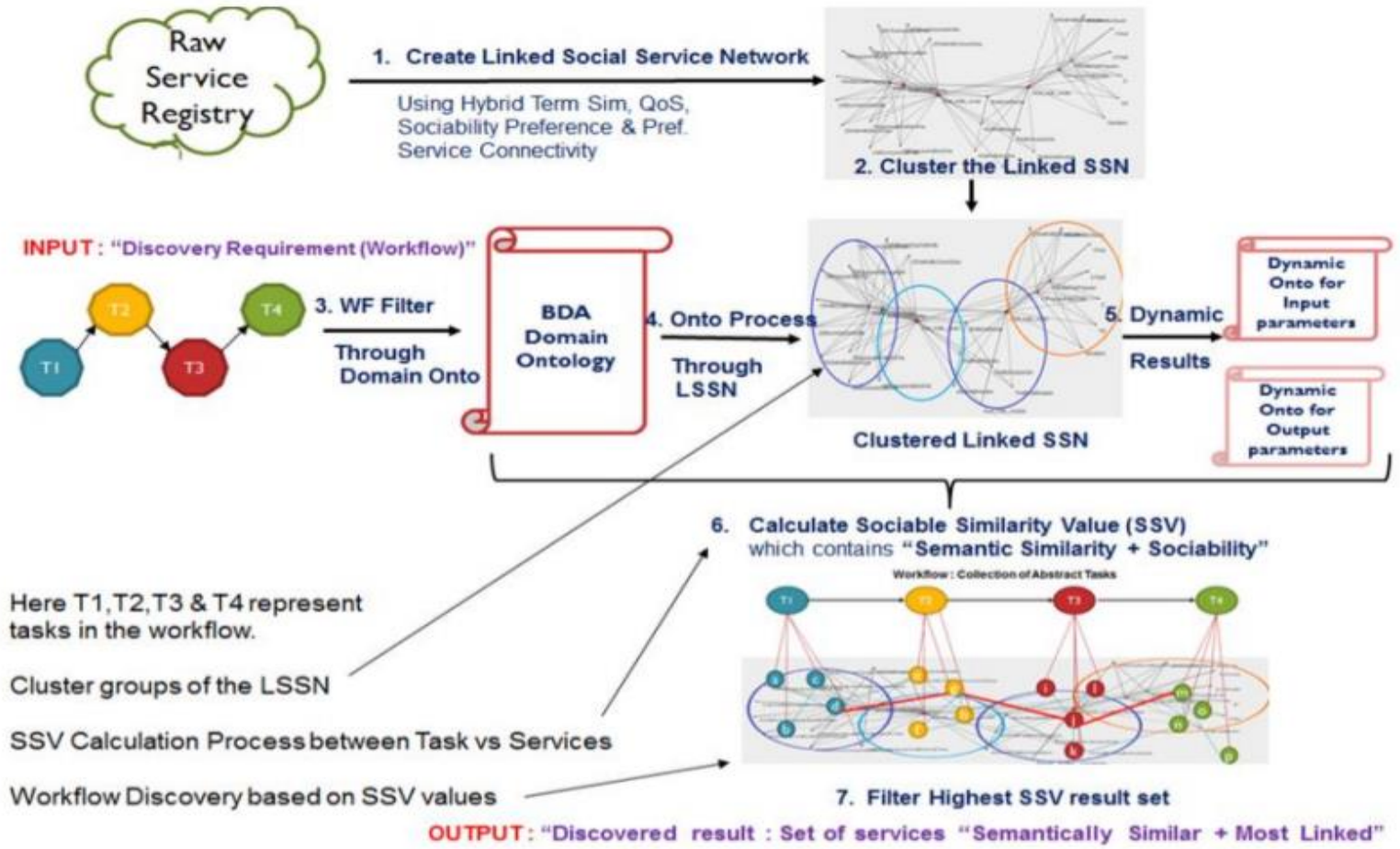


Figure 6. Architecture of Proposed LSSN based Discovery

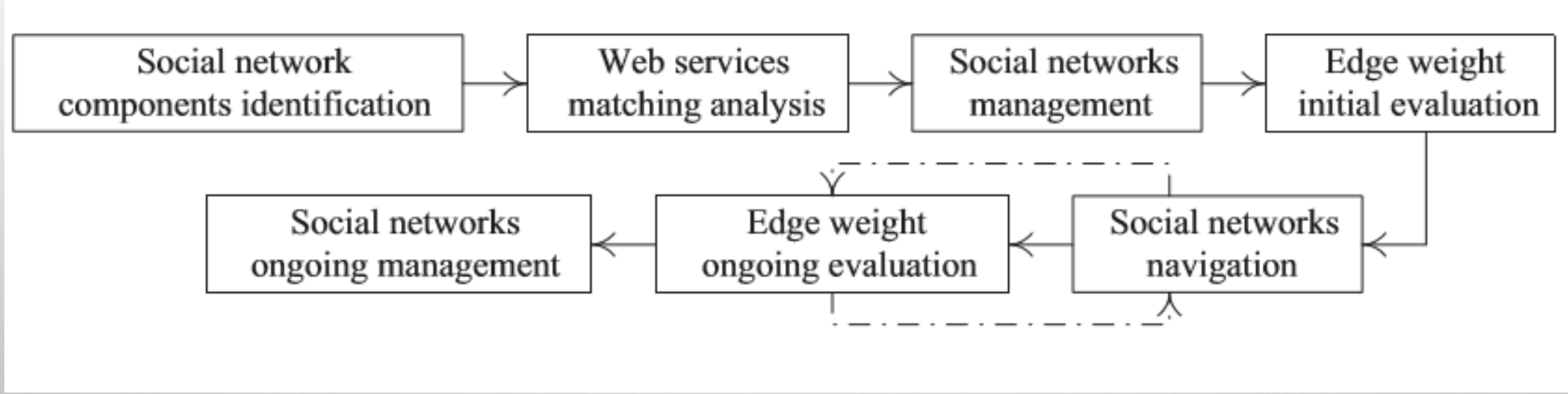
Pros :

- Reducing the work of performing heavy tasks of data analysis
- Improved discovery performance by linking services based on sociability.

Cons:

- Do not perform the task of selecting the web service
- Do not complete the automation service composition of the BDA process.

2. Weaving Social Networks Principles into Web Services Discovery



Pros:

- Use of Web Service social network component & navigation for quick Web Service discovery
- Dynamic assignment of weights on social edges for searching relevant web services

Cons:

- The technique is not tested and compared with other registry based methods
- This techniques only works if it correctly captures the Web Service relationships like substitution, competition and collaboration which is quite hard to achieve.

2. Web Service as a Global Social Service Network

- Linked social service-specific principles were introduced as a recipe for interlinking of services which is covered in the paper.
- The Global Social Service Network is social network with a structure of a directed graph:
- **G = <V, E>** where V is each node being a linked social service and E is the directed social link for these nodes.
- The Quality of these social links is decided by factors like Dependency Satisfaction Rate, QoS Preference, Sociability Preference and Preferential Service Connectivity as parameters. The proposed solution is known as Link-As-You-Go.
- It allows users to browse in one service and navigate further into social links with related services.
- By observing the shortest path between two given services, we can decide on the efficiency by counting the smallest number of social links that a user has to navigate before settling for a service.

Problems with current solutions:

- The isolation of web services and also the lack of a global linking of these services with a lack of social relationships among services with common connection are the reason for the slow uptake in the use of web services.
- The authors decided to provide a Linked Social Services network as a Service.

Pros:

- Commonly used services today like WSDL, Web APIs and OWL-S largely treat web services as isolated islands with no inter-connection and no knowledge of related set of services.
- Though functionality is widely covered by services today, sociability is the issue addressed in this paper.
- Though the common aspect of building social network is used, it suggests the idea of building a social network of related web services whereas our paper builds a social network of users and check their web services.

Cons:

- Emphasizes the need for factoring in social influence and feedback system which they plan for a future implementation.

3. Integrating multiple conceptual relationships for web service discovery

- The primary metric for determining the efficiency of algorithm is use of Precision, Recall and F-measure which helps evaluating the performance of web service discovery methods.
- The F-measure further integrates Precision and Recall in a single equation using composite harmonic mean.

$$Precision = \frac{Relevant \cap Retrieved}{Retrieved}$$

$$Recall = \frac{Relevant \cap Retrieved}{Relevant}$$

$$F - measure = \frac{2*Precision*Recall}{Precision + Recall}$$

A parameter epsilon represents the similarity of services above a particular value.

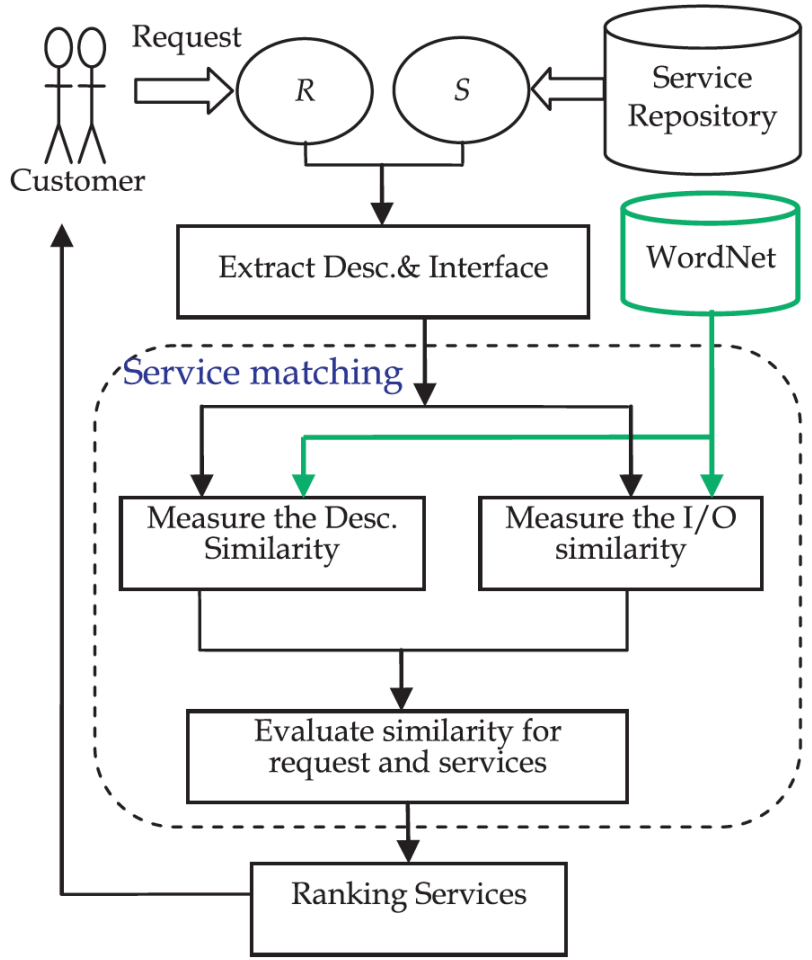


Fig. 3. Processes of Web services discovery approach.

- This is a paper written primarily on probabilistic and semantic approach as encountered in English language for discovery of web services.
- It makes use of ontologies like the is-a and has-a relationships and antonomy differences represented by vector of terms.

Pros:

- The lack of depth that is encountered in UDDI registry due to syntactic keyword matchmaking.
- A semantic based approach instead of the more commonly used syntactic based method, such as Ontology Web Language for Services and Web Service Modeling Ontology is used.
- The advantage of such a technique is the contextual information extraction. For example, in case of financial knowledge, semantic Web services Support Business Intelligence, executive information systems and uses financial concepts like cash flow, sale, profit, etc.
- Conventional searching methods use metadata to match service requests, whereas this paper made use of bipartite graphs for comparison and techniques like TF-IDF

3. Web Service Directory Framework for Environments

- The unique identification of devices is done by the Jabber Identifier (JID) which follows the format like: localID@domainID/resourceID, where domainID is fixed for a local network and localID depicts the type of web service.

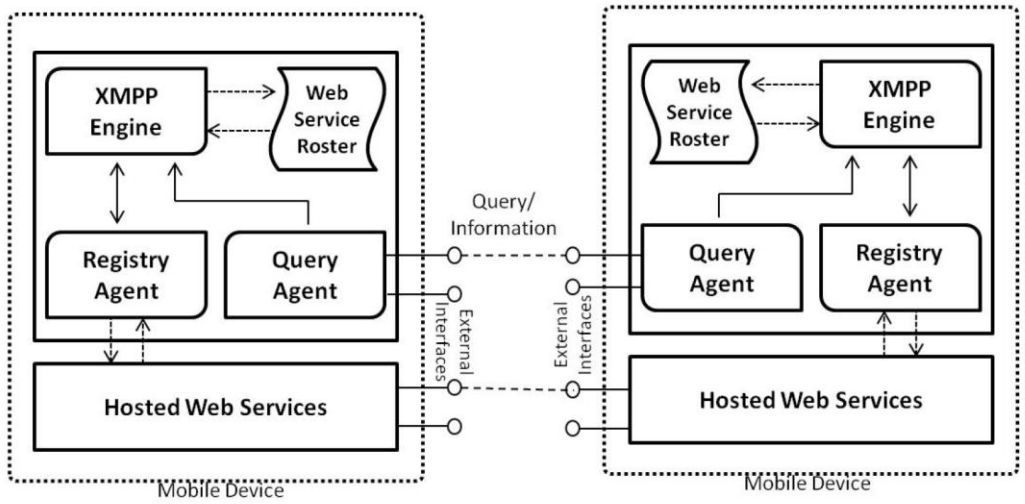


Fig. 2. Service Discovery Architecture

- The primary problems of mobile devices are minimal infrastructure, battery and network constraints, limited computational power of devices and dynamic service registry management which are not efficiently handled by mobile devices.
- The centralized management of web services, need of frequent updating and volatility of such devices makes the use of legacy services inefficient.
- Several service providers do not register with UDDI registries due to the complexity involved.

Pros:

- Makes use of Service oriented architecture (SOA) and deploying service registry over such mobile devices.
- The paper primarily substitutes UDDI with XMPP which makes it more lightweight, loosely coupled and extensible, server-less working for distributed application appropriate for cooperative and personal web services hosted over mobile devices in a local area.

Cons:

- Setting up a web service roster on each device will eat up memory and space in handheld devices.
- The service registries though implemented on a local device still do not have option for customization according to the user which is fundamentally why such an implementation cannot be considered efficient.

4. MULTILEVEL INDEX FOR LARGE SCALE REPOSITORY

Problem Statement:

- Store the web services in an efficient data structure of multilevel index to reduce the time required to search and discover the web services from large scale repository.

- **PROPOSED MODEL :**

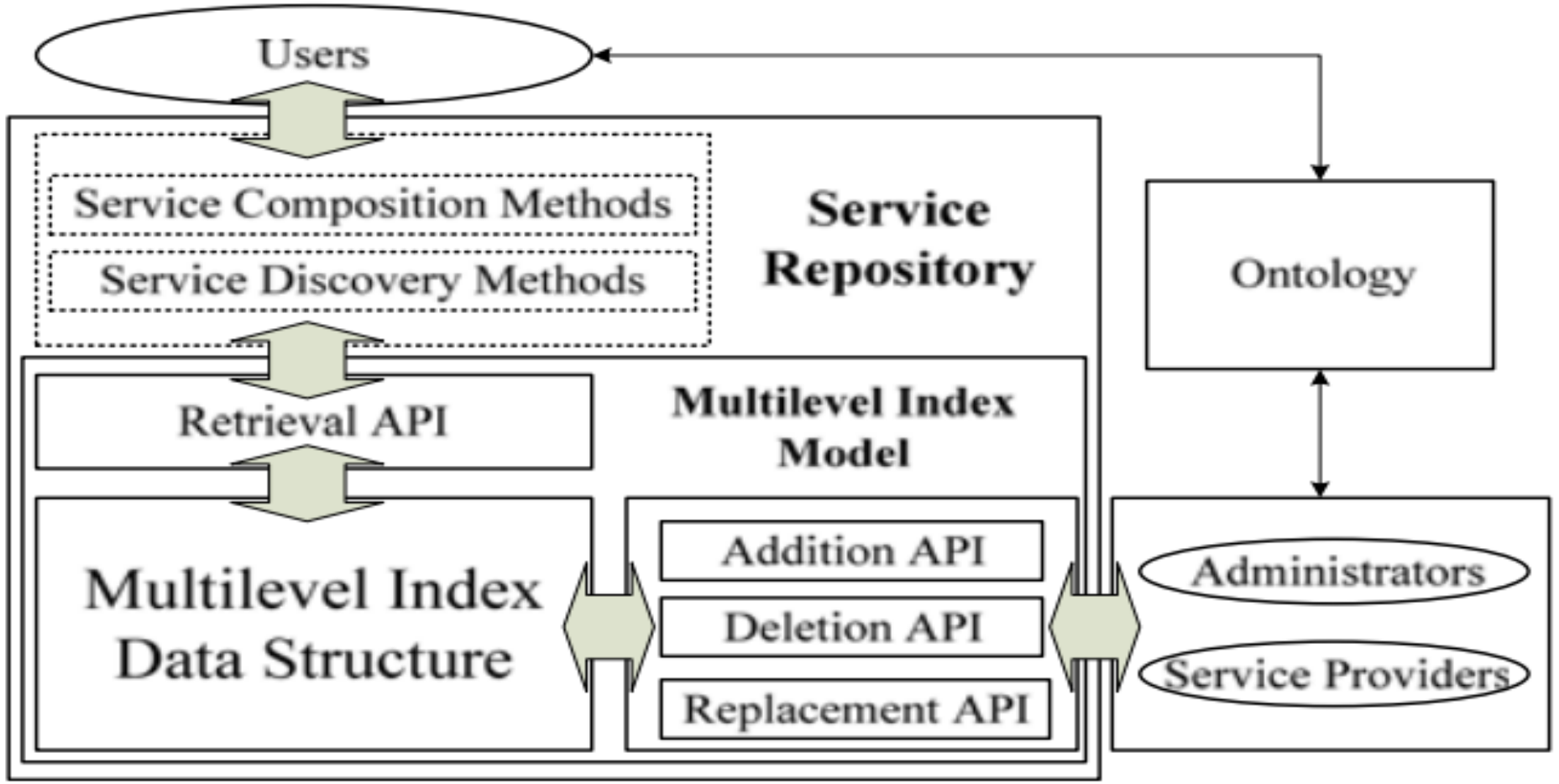


Fig. 1. Architecture of the proposed model.

Theoretical foundation of multilevel index:

I. Equivalence Based Indexes
II. Redundancy removal Indexes
III. Selection of a key

Operations on Multilevel Index :

I. Retrieval
II. Addition
III. Deletion and Replacement

Pros:

i. Multilevel index model developed is more efficient than existing structures such as sequential and inverted index structures.

ii. Due to multilevel index model, the time required for service discovery and composition has reduced leading to performance improvement.

Cons:

i. Various factors such as number of input parameters for each service, number of services containing the same parameter as input, probability of each service not taken into consideration in details.

ii. Adjustment of addition operation to handle the above different factors maintaining the high efficiency not taken into consideration in depth.

iii. The above cons are planned as a part of future work

4. Web Services Discovery Approach Based on Mining Interface Semantics

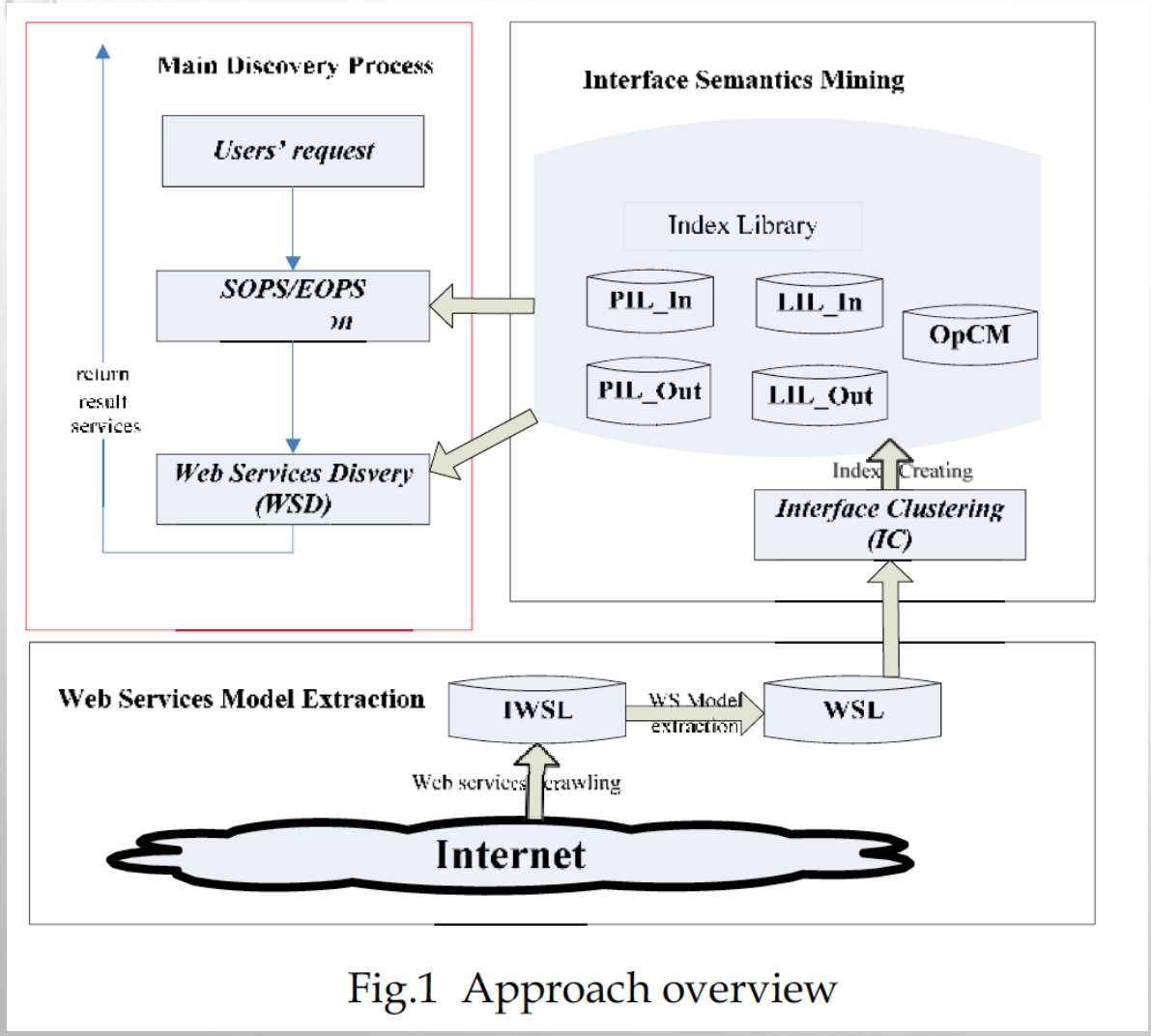


Fig.1 Approach overview

- This paper provides a unique way for Web services discovery(WSD) by mining the underlying semantic structures of interface parameters.
- The paper first builds a Keyword Semantic Body and then does indexing of it.

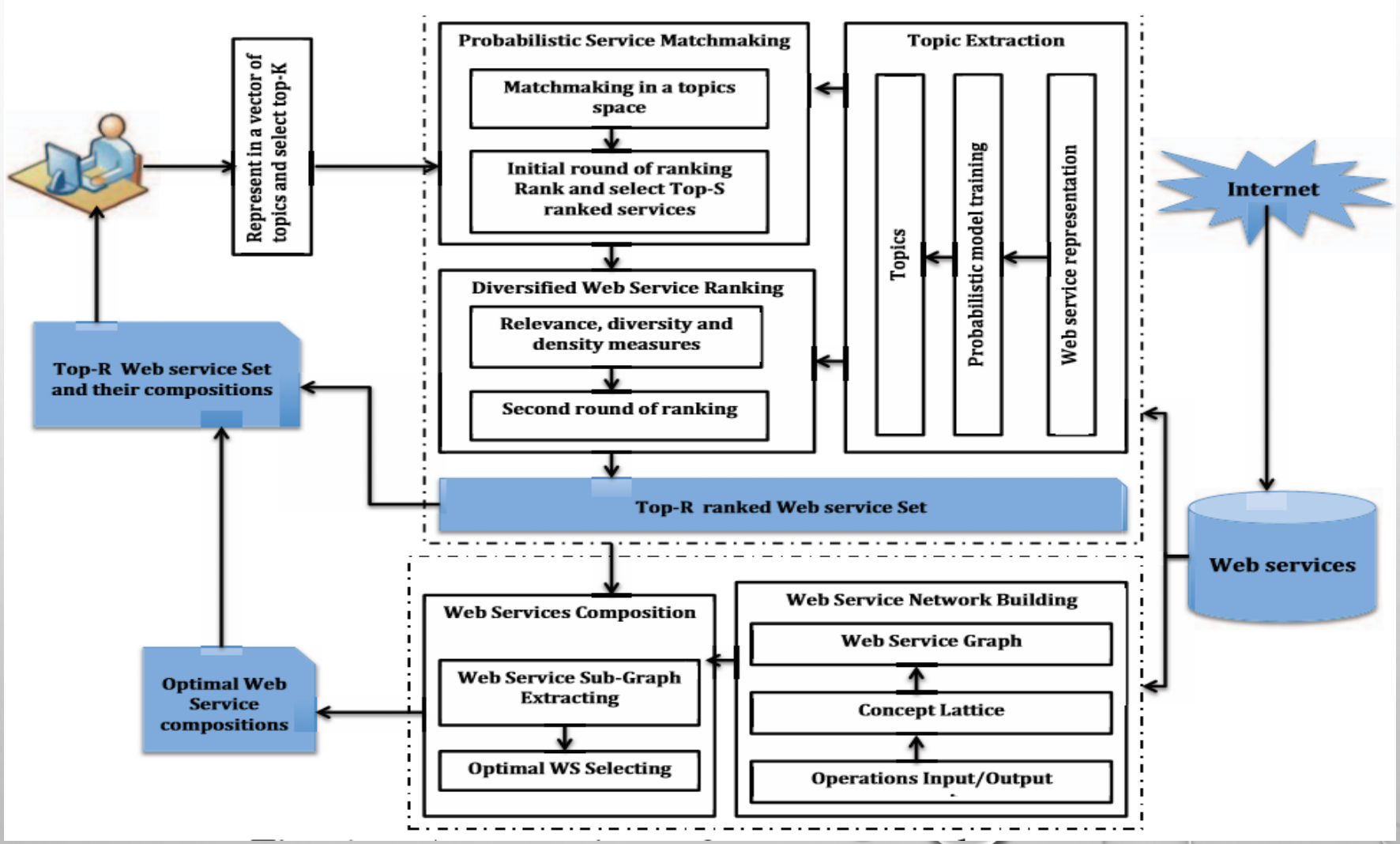
Pros:

- WS are buried deep inside large number of web pages as they are stored in the web service library and lots of WSs that are highly correlated get missed
- This information is majorly not present in the natural language but rather in a coarse-grained description. So, this semantic information is not caught by normal search engines during WSs discovery
- It also provided a method to search composite WSs based on user's request.

Cons:

The mining performed in the description model is based on the efficiency of the algorithm used and an inefficient algorithm can give misleading results.

4. Web Service Discovery Using Probabilistic Approach



Pros:

- Using probabilistic approach for service discovery, redundancy has reduced in web service search results.
- Due to reduced redundancy, user satisfaction has increased
- Performance of web service recommendation in terms of precision has improved.

Implemented Paper

User's Social Profile Based Web Service Discovery

Human element in web service discovery

- The concept of using Social Networks is novel.
- Human-related factors play more and more important role in service search and composition extracting properties & relations from people's web pages.
- Studies pay more attention to the functional requirements, while the relationships among services, providers and users are neglected.
- web services and users in the same social network and examining their influence on each other have not been investigated yet in previous work.
- SOAF: Integration of services into social networks (SOAF - Service Of A Friend) to leverage the creation of the Internet of Services.

UDDI vs Decentralized registry

- Need a lot of time to browse his entire social network to search a web service.
- Type of machine readable ontology.
- No need of a centralized database. Described using RDF and OWL.
- Major example is its use in an Identity System known as WebID.
- It helps us form own ego-centric network as this paper makes use of ego-centric data and past invocation histories.

FOAF

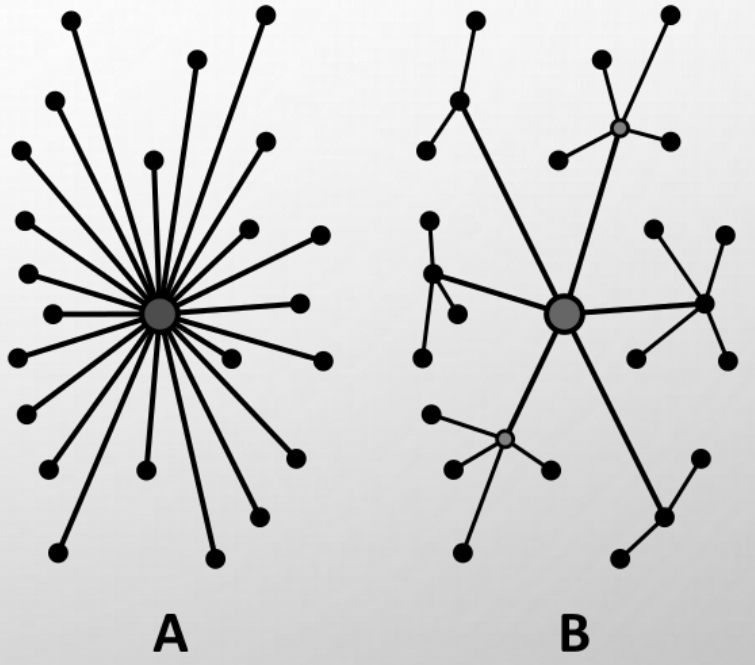
- Need a lot of time to browse his entire social network to search a web service.
- Type of machine readable ontology.
- No need of a centralised databaseDescribed using RDF and OWL.
- Major example is its use in an Identity System known as WebID.
- It forms its own ego-centric networkThis paper makes use of ego-centric data and past invocation histories.

FoaF and Decentralization of registry

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

<#JW>
    a foaf:Person ;
    foaf:name "Jimmy Wales" ;
    foaf:mbox <mailto:jwales@bomis.com> ;
    foaf:homepage <http://www.jimmywales.com> ;
    foaf:nick "Jimbo" ;
    foaf:depiction <http://www.jimmywales.com/aus_img_small.jpg> ;
    foaf:interest <http://www.wikimedia.org> ;
    foaf:knows [
        a foaf:Person ;
        foaf:name "Angela Beesley"
    ] .

<http://www.wikimedia.org>
    rdfs:label "Wikimedia" .
```



JACCARD'S CO-EFFICIENT

- Jaccard Index / Similarity Coefficient::
- Range from 0 to 100%
- DIS: extremely sensitive to small samples sizes, so should have used some other logic
- Formula: Jaccard Index = (the number in both sets) / (the number in either set) * 100
- $J(X,Y) = |X \cap Y| / |X \cup Y|$
- In case there are missing values: Replace the missing values with the median for the set,
- Use a k-nearest neighbor or EM algorithm.
- Link: http://www.statisticshowto.com/jaccard-index/

SIMILARITY DEGREE: α

- We define a Similarity Degree(SD) or α measure in order to compute and find for each current user his best friends.
- This degree is useful in filtering all the user's social relationships and finding only his closest friends.
- Example:: Similarity Degree:
- SD (U1, U2) =0.8;
- SD (U1, U3) =0.75;
- SD (U1, U4) =0.5 and
- SD (U1, U5) =0.3.where Where SD (u , v) ∈ [0..1].

EXAMPLE

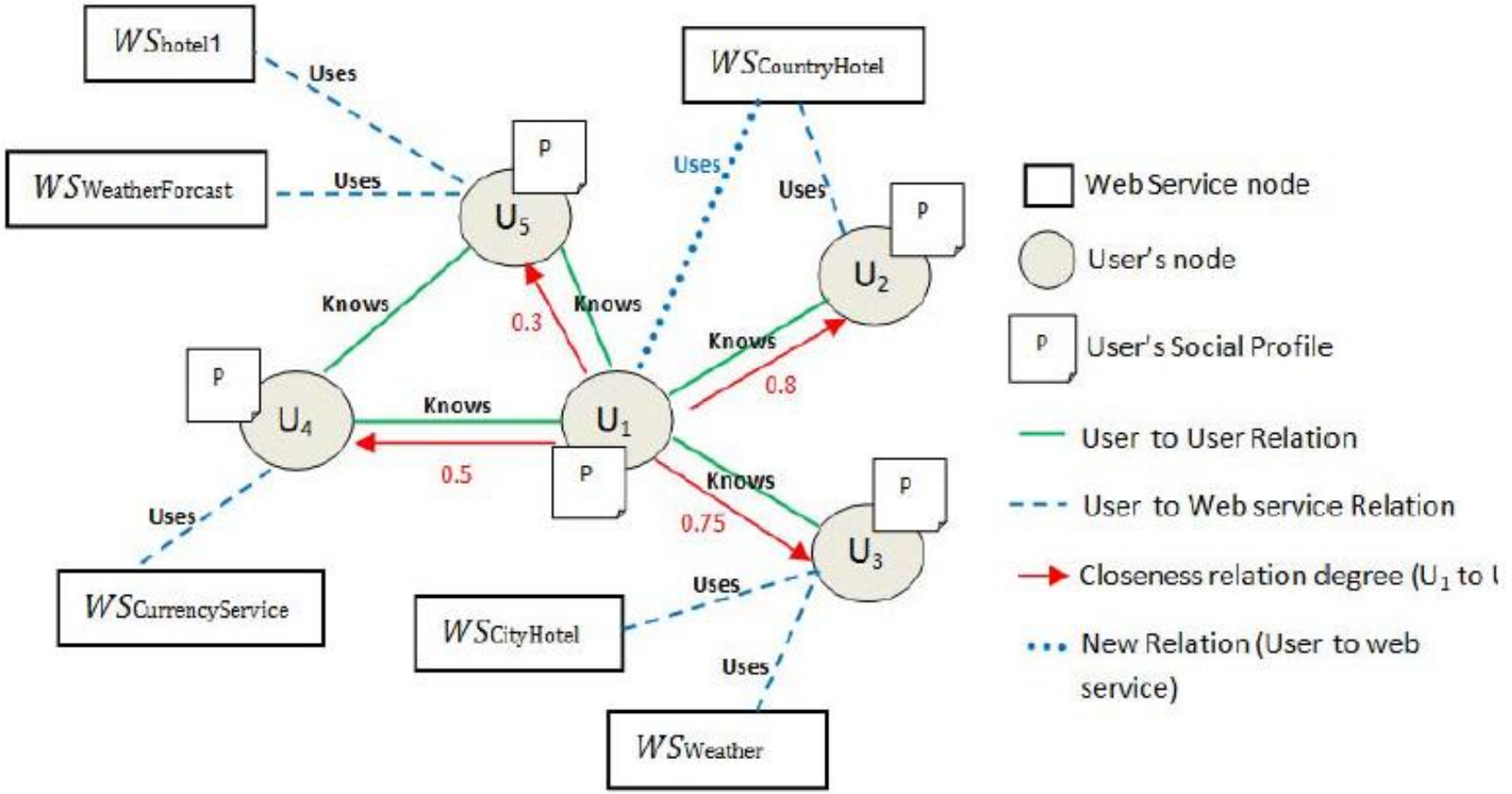
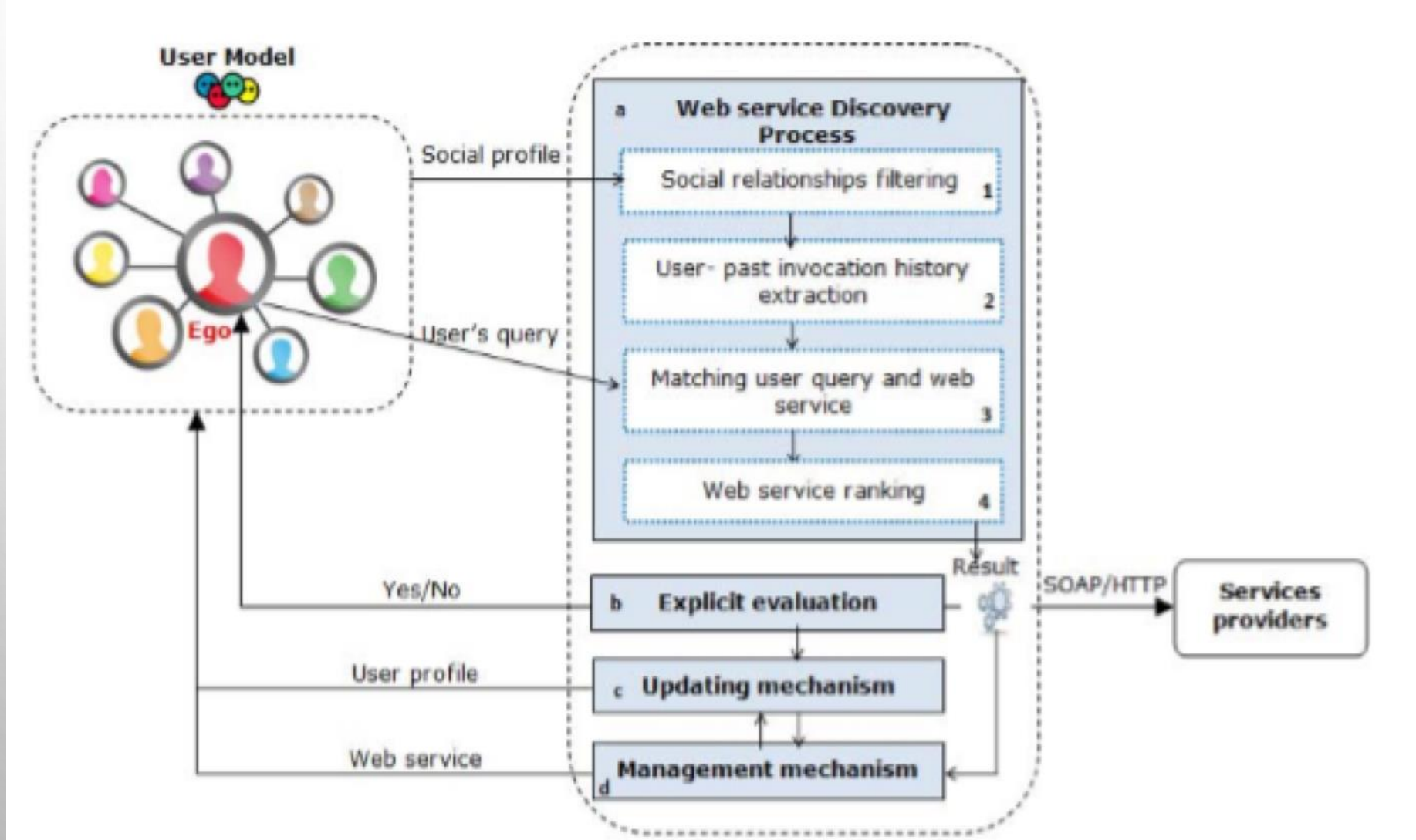


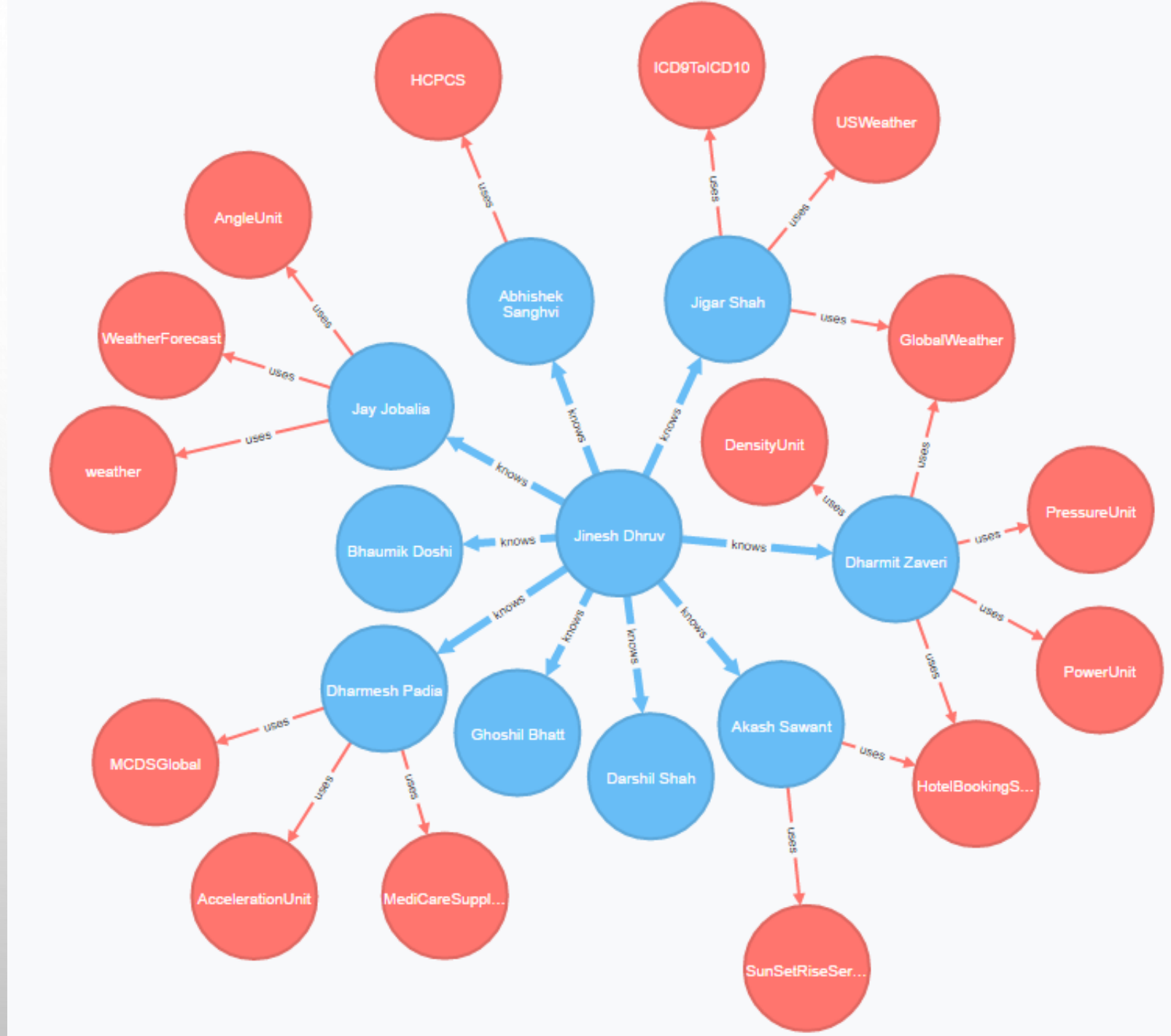
Fig. 1. Example of a motivating scenario.

ARCHITECTURE



Implementation

1. Build User Model (Neo4j Graph)



GRAPH REPRESENTATION

- Socio-centric vs Ego-centric approach: Can as well make use of centralized registries.
- Takes time to manually search everything and hence not dynamic in nature
- The types of friends and age groups are very variable, so likability of something is dependent on those factors.
- **Nodes:** Represent users and web services(atomic or composite)Contains:: name, endpoint, invocations: Extracted from WSDL document.
- Can also be stated as:
  - Contains: Static OR Dynamic data::Static: name, age, country, city, etc.
- **Dynamic:** interests of items(movies, music, books, etc.), #links that define social relationships(knowledge or friendship), feedback(user's social activities), past invocations.
- **Edges:** Represents links between nodes. Define social relationship between users(knowledge relation) Or between users and services(usage relation).

Implementation

Algorithm 1 URPI-Disc

Input: profileU$_i$_FOAF : RDF document, User_QueryU$_i$: String
Output: Disc_WS
Begin
1: list_persons, Disc_WS= ArrayList,
2: list_ persons = **ExtractCloseRelations**(profileU$_i$_FOAF);
3: **for each** User$_h$ in list_persons **do**
4: Disc_WS = **ExtractInvokedWS**(User_QueryU$_i$, User$_h$);
5: **end for**
6: **return** Disc_WS;
End

1: Iterating through your friends list.

Algorithm 2 ExtractCloseRelations

Input: profileU$_i$_FOAF: RDF document
Output: Cfr

Begin
1: Cfr : ArrayList
2: WSRel=0;
3: friends = **extractFriendRelation**(profileU$_i$_FOAF);
4: **for each** User$_j$ in friends **do**
5: WSRel= **SimilarityDegree**(U$_i$, User$_j$);
6: **if** (WSRel> = threshold α) **then**
7: add User$_j$ into Cfr
8: **end if**
9: **end for**
10: **return** Cfr;
End

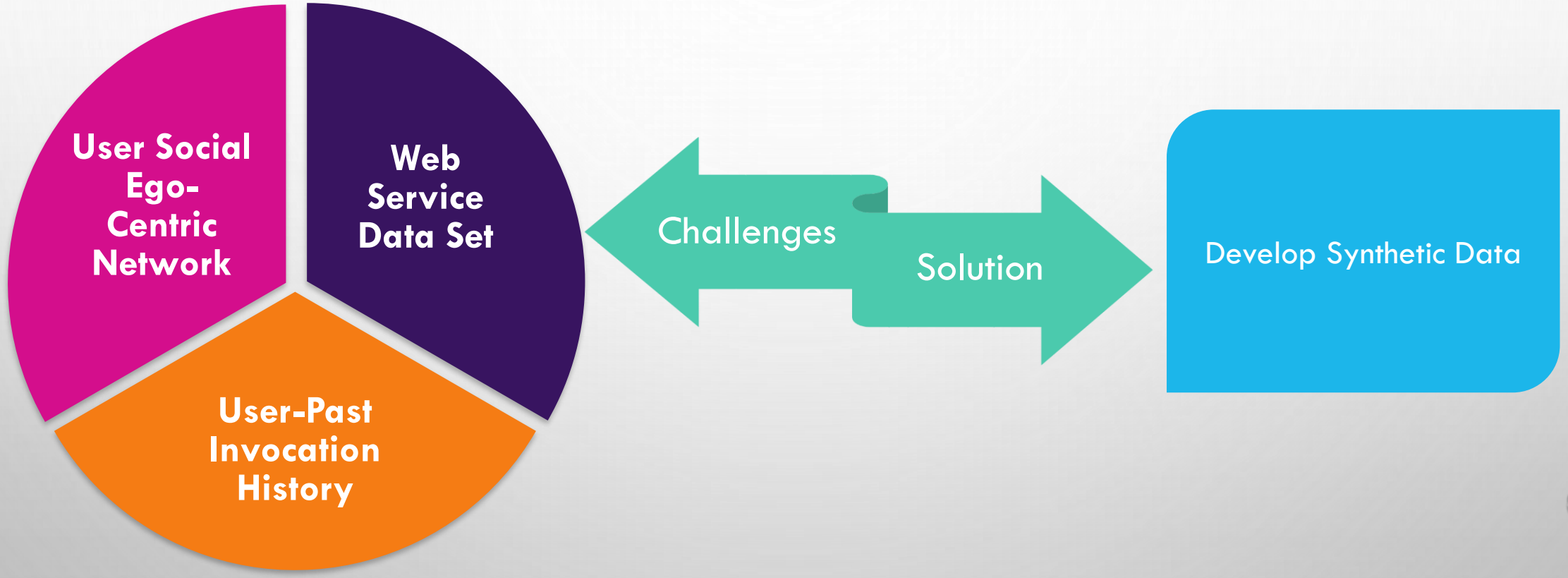
2: Find ego-centric network.

Algorithm 3 ExtractInvokedWS

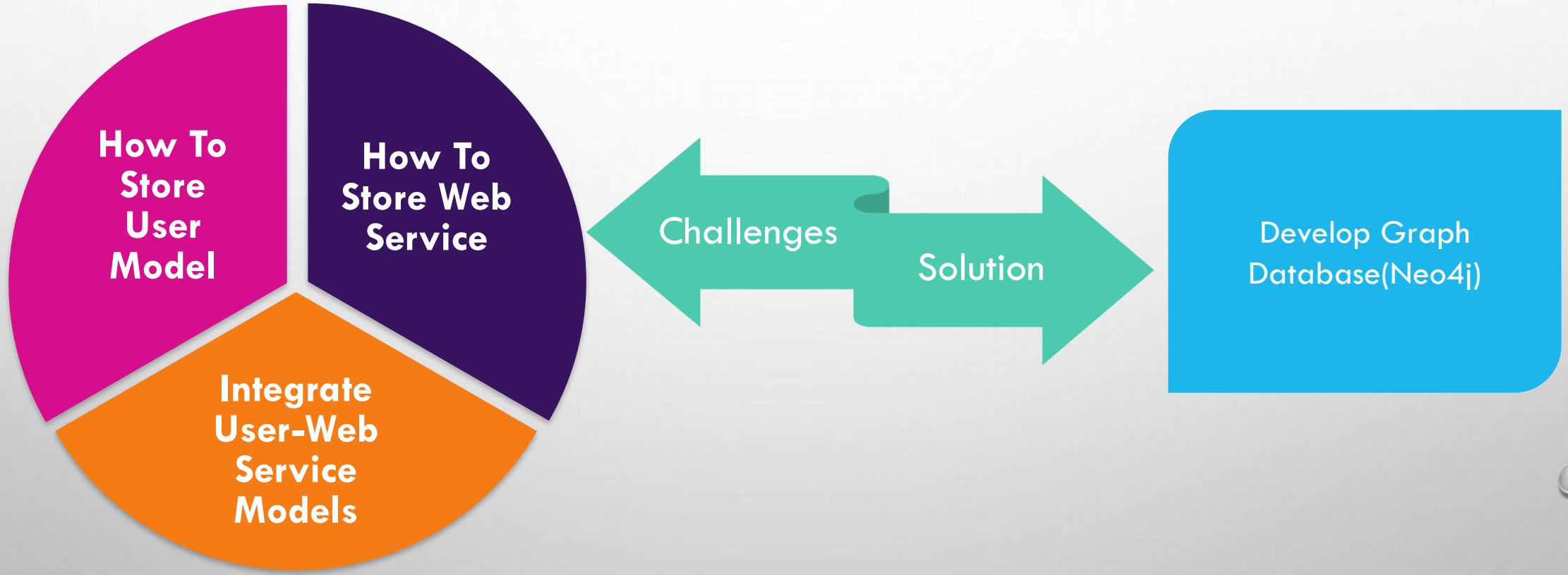
Input: User$_k$: String, User_queryU$_i$: String;
Output: InvokWS
Begin
1: InvkWS, list_nWS: ArrayList;
2: Select the social profile of User$_k$ profileU$_k$_FOAF;
3: list_nWS= **extractWS** (profileU$_k$_FOAF);
4: **For each** WS$_m$ in list_nWS **then**
5: **if** (WS$_m$ contains User_queryU$_i$ **then**
6: add name of WS$_m$ into InvkWS
7: **end if**
8: **end for**
9: **return** InvkWS;
End

3: Find the web services of your close friends using past invocation history and rank them accordingly.

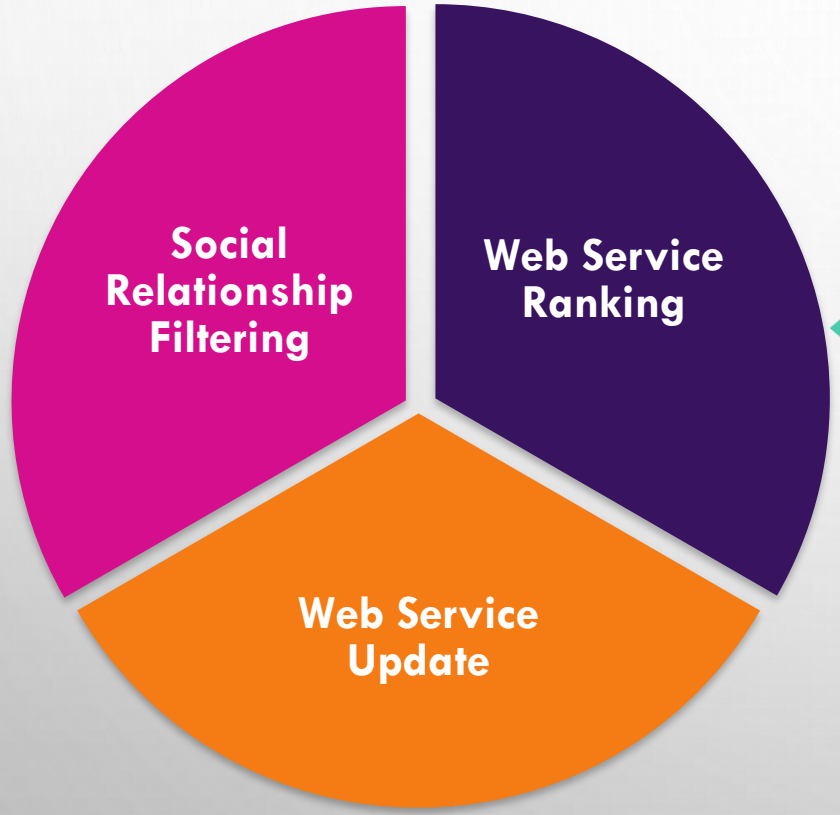
Data Collection Challenges



Data Preparation Challenges



Implementation Challenges



Challenges

Solution

Used Similarity Degree

Ranked WSS from closest friend to close friend

DRAWBACKS IN PAPER

- The use of WSDL restricts the types of web services that we are using for survey making the data less user-friendly.
- The use of 'mutual friends' as a Similarity Degree is erroneous.
- The user data of Social Networks is difficult to obtain because of privacy concerns.

FUTURE WORK

Change in Similarity Degree

- Use of an alternate co-efficient and Similarity Degree Measure: We found that the use of Similarity Degree based on mutual friends is restrictive.
- It is also misleading in a way that people with most mutual friends are not necessarily the closest friends.
- We suggest use of alternate methods like the inclusion of some other factors like the use of their page likes and groups for measurement of SD.
- This will broaden the scope of users which may not have many mutual friends with the user but may still give a better service discovery result.

FUTURE WORK

Use of other methodologies

- Use of Restful services in addition to the WSDL Document based search: The use of WSDL documents has restricted our work to only include the services that use WSDL documents.
- Also WSDL documents are less commonly available because of the use of enterprises switching to private UDDI registries.
- There are many other services being used in the market today and use of them will give a more realistic picture of the service discovery.