Journal of Universal Computer Science, vol. 20, no. 13 (2014), 1791-1812
submitted: 30/1/14, accepted: 6/9/14, appeared: 28/11/14 © J.UCS

An Adaptive Intent Resolving Scheme for Service Discovery and Integration

Cheng Zheng
(Western University, London, ON, Canada
czheng23@uwo.ca)

Weiming Shen
(Western University, London, ON, Canada
wshen@uwo.ca)

Hamada Ghenniwa
(Western University, London, ON, Canada
hghenniwa@uwo.ca)

Abstract: Service discovery and integration is an important research area with efforts invested to explore the potential advantages of collaborative computing in general and service-oriented computing in particular. However, current technologies still limit their application within the reach of enterprise systems or privately available services. Intents is an emerging and innovative technique aimed to discover and integrate publically available services. In Intents, intent message resolving is a critical step which is deemed to decide the quality of the whole system. However, existing schemes applied in intent resolving adopt the exact-matching strategy which may rule out services desired by the user. This paper brings in information retrieval techniques and applies them to intent resolving. We take an empirical approach through extensive experiments and analyses on a real dataset to obtain guiding principles. Based on the resulting principles, an adaptive intent resolving scheme is designed. Afterwards, we integrate the scheme into the Intents user agent developed in a previous project.

Keywords: Service Discovery and Integration, Web Services, Intents, Intent Resolving
Categories: H.1.0, H.3.4, H.4.0, H.5.2

1 Introduction

Service discovery and integration is an important field attracting great interests of researchers and developers in collaborative computing [Shen, 07], pervasive computing [Zhu, 05], and mobile computing [Mian, 09; Neyem, 08]. A service provider designs, develops, and publishes a service in compliance with standard protocols. Once an application developer needs to implement some functions in his/her product, he/she may depend on existing services instead of producing a new module from scratch. This philosophy has advantages in two aspects. On one hand, adopting existing services is able to speed up the development cycle of software systems, which saves project budgets in terms of human and computing resources. On the other hand, if a published service is widely applied, it should be stable, effective and efficient. Integrating such a service will demonstrate better system performance than developing anew module.

However, most of the existing research efforts on service discovery and integration are based on the currently famous Web services ternary participant role classification. In the classification, participant roles revolving around Web services are divided into three categories: *service providers*, *service brokers* and *service consumers*. In these schemes, service consumers are usually application developers who take almost all the work for seeking and binding the desired remote service while at the same time the application end users who indirectly use the service are ruled out of the steps for determining the working service. The very limited end user participation in service discovery and integration may cause serious problems. For example, application developers may choose service A for their released product. However, some end users may prefer service B and others have interests in service C while using the product. Even worse, if A is blocked in a network, the product may be out of order to the end users in the network. As a result, the product marketing will be seriously affected.

Intents is an emerging technique for service discovery and integration which is featured in the introduction of application end user participation into the process of service discovery and integration. It was first designed and implemented in Android, known as Android Intents. However, Android Intents is specifically designed for the integration of Android application components and is limited when applied to other systems like Web services. In order to address this problem and extend the applicability of Intents, Web Intents [WebIntents, 13] was proposed to employ Intents for online application discovery and integration and exhibit huge potential for service discovery and integration. The advantages of Intents over other techniques include:

- Intents is much more flexible than existing approaches such as UDDI for service discovery and integration because of its consideration of Web dynamicity. Services are registered with decentralized private registries before their use in any application. The management actions such as adding and removing a service to a registry are under the control of the end user.
- Intents can work around the restrictions imposed by gateways on a network which may undermine some application operations. For instance, if an online application depends on some services which are blocked by its gateway, the application may not be able to run normally. Intents addresses this issue by binding services dynamically, i.e., application developers only need to specify operation semantics and let end users decide the final service to complete application operations. Thus an end user can choose the service which is permitted by his/her network gateways so that the application work flow will not be affected.
- Service description in Intents can be embedded into relevant Web pages and may employ the references in Web pages to build connections between services. This strategy provides a convenience which Web search engines can utilize to discover and collect the services scattered over the Web as a byproduct of their ordinary Web crawling procedure. In addition, the implicit links may also be exploited in service retrieval like PageRank in traditional Web information retrieval.

In Intents, when application developers want to employ external services for their applications, they can specify a construct named *intent* to represent their service needs in the applications. An intent consists of two major factors for a service need: the

intended action to perform and the data prepared for the action. When an application user uses the application, the intent is triggered and a message enclosing the intent will be sent to the application user's user agent. The user agent is responsible for resolving the intent and assists the end user in finding out a satisfied service to complete his/her application task.

However, existing intent resolving methods only apply the exact matching strategy, which means the two necessary parts of an intent, i.e., action and data type in a candidate service description must be exactly the same as that in the intent message invoked by a service consumer. This method compromises and limits the applicability of a service at least in the following aspects:

- Limit the set of potential services. For instance, if an intent with the action "share a link" is triggered, only the services with exactly the same action are returned. Those services supporting actions such as "share" or "share link" will be ruled out of the set of candidate services.
- The application developer must know exactly the action and data type while designing an Intents-supported component. Sometimes, an application developer may only know the semantic or the meaning of what operation will be performed. In the exact matching scheme, the developer has to know the action with no difference and this may become a burden for application developers.

This paper proposes an adaptive method in intent message resolving. The idea comes from techniques in information retrieval which has also been used by researchers for Web service discovery [Pan, 11], but we adapted it for Intents. Firstly we define the intent resolving problem and create combinations of similarity formulae based on the fields in intent-based service description. We take an empirical approach through extensive experiments and analyses on a real dataset to obtain guiding principles. Based on the resulting principles, an adaptive intent resolving scheme is designed. Then, we integrate the scheme into the Intents user agent developed in a previous project.

The rest of the paper is organized as follows: Section 2 reviews the background of Intents including its concepts and principles. In addition, related techniques in information retrieval are touched on. Section 3 formally defines the problem of the intent resolving process. Section 4 presents our empirical study with detailed experiments and analyses. Section 5 describes how the adaptive resolving scheme is designed and its integration to an existing Intents user agent. Section 6 concludes the paper and proposes some future work.

2 Background and Literature Review

2.1 Intents

The name "Intents" was coined with the meaning of a collection of intents and an intent is a representation that models the necessary elements when a service request is triggered:

- *Action*: the name of the service operation which is about to perform, and
- *Data*: the *data* fed to the intended service operation.

The architecture of an Intents system [Zheng, 13a] can be illustrated by Figure 1.

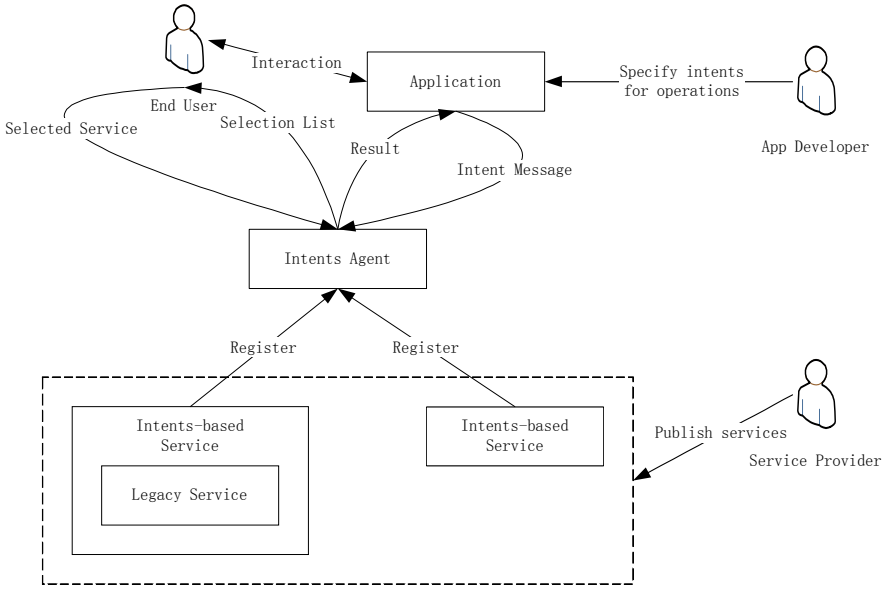

Figure 1: Architecture of Intents-based Systems

The participants revolving the Intents system include service providers, application developers and end users. A service provider takes the responsibility of creating and publishing Intents-based services. An application developer designs and implements some application which may be dependent on Intents-based services. The end user is the ultimate consumer of the application.

Each end user has a private Intents agent. The user can register any services he/she is interested in with the agent. This step is analogous to "software installation". Once some operation of an application relying on Intents is triggered, an intent message is created and sent to the Intents agent. The agent then resolves the intent message and generates a list of candidate services for the user to choose. After the user makes a decision, the intent message is forwarded to the selected service to complete the task.

An Intents-based service is capable of absorbing an intent message, validating its data, processing the data and returning a result or showing a resulting page. The service description for an Intents-based service can be embedded into a Web page with a customized HTML tag named *intent*. Each tag specifies a service and its related attributes such as identifier, supported action and data, service title and others. This strategy is different from the existing approach which utilizes RDF to link the data of services [Pedrinaci, 10]. Each service is required to be single topic which means it only contains one operation compared with multiple operations support in WSDL. Single-topic services are clearer than multi-topic services in that the operations may be inter-interfered in the latter pattern.

The intent messages triggered by the application are classified into three categories: explicit intents, authoritative intents, and naïve intents. An explicit intent

contains a service identifier to invoke the service directly. An authoritative intent is an intent specified by an authoritative organization or institution. The fields of an authoritative intent are determined by authorities and should not be changed by application developers. The naïve intent is the most flexible type among the three categories. A service provider can design and specify any contents in the action field of an intent that a service supports and an application developer can also specify any contents in the action field of the desired intent bound to his/her product if he/she fails to have enough knowledge about the desired service but the meaning of the action to take.

2.2 Information Retrieval Models

In the past few decades, a variety of methods for modelling the similarity between a query and a document in a collection have been proposed and developed. These models are called retrieval models. Vector space model, probabilistic model, and language model are representative models with verified effectiveness by both analytical and empirical evidence.

A document is usually modelled as a set of terms while ignoring their original ordering in the document. This reduction is known as the *bag of words model* [Manning, 09]. The vector space model [Salton, 75; Salton, 83] treats such representation as a vector and the set of vectors for all the documents constitutes a vector space. Thus the similarity between a query and a document can be measured by the cosine value of their vector representations. Lucene [McCandless, 10] provides a default retrieval model which is a modified vector space model implementation. In addition to the Lucene's default model implementation, Singhal et al. [Singhal, 96] proposed pivoted document length normalization in the vector space model and their method has achieved widely recognition in the past few years.

Probabilistic model [Fuhr, 92; Robertson, 76] derives the similarity from the perspective of probability theory. It thinks of the relevance relationship between a query and a document as uncertain and manages to estimate a probability of the relationship which is employed as the similarity measure between the query and the document. Okapi BM25 [Robertson, 95; Robertson, 99] is a highly effective probabilistic model implementation.

Language model [Lavrenko, 01; Ponte, 98] also creates a probability of query and document. However, instead of directly inferring the relevance relationship, language model assumes a statistical distribution for each document and estimates the conditional probability of generating the query given a document model. However, language model has the disadvantage of lowering the probability of words absent from a document while raising the probability of words appearing in the document. Thus smoothing methods such as Dirichlet priors are needed in the language model to compensate for such deficiency. Zhai and Lafferty [Zhai, 01] made a study on various smoothing methods of language model including Jelinek-Mercer smoothing, absolute discounting, and Dirichlet priors.

Fang [Fang, 07] proposed and discussed seven constraints in constructing a reasonable formula for similarity computing. She then analyzed, modified and improved the above existing methods and came up with an axiomatic approach for modelling relevance based on the seven principles and achieved a comparatively better results on some public datasets.

Traditional techniques in IR have achieved satisfactory effectiveness on plain text and assorted Web contents, especially when the length of documents is long enough. However, because Intents is a newborn concept, the effects of applying these IR techniques directly on Intents still call for further research work.

2.3 Retrieval on Short Text

Since the action field in an intent is usually a short text string, service retrieval in Intents may utilized the techniques in the field of short text retrieval. This section will investigate the existing techniques on short text retrieval and discuss how they are related to intent resolving.

One way of short text retrieval is to expand the word set by semantically similar words. Thesaurus looking-up is a straightforward methodology in such direction. A thesaurus is a reference work that lists words grouped together according to similarity of meaning [Theaurus, 13]. A thesaurus can be locally generated or globally maintained [Imran, 09; Walker, 01]. WordNet [Fellbaum, 98] is a popular global thesaurus which contains more than 100,000 synsets. Global thesauri are usually well maintained but they may not apply to some corpora, especially those with newly created words. So an automatically generated thesaurus from a specific corpus is complementary in some cases. Word co-occurrence which means two different words occurring in a document are likely to be semantically related is leveraged in automatically generating thesaurus [Manning, 09].

In addition to words expanding, some researchers proposed taking the advantage of mature Web search engines [Efron, 12; Sahami, 06]. They fed the short text to a Web search engine, and selected a set of most relevant documents from the returned document set. Then a set of relevant words determined from these documents were substituted for the original short text in retrieving activities.

The methods mentioned in this section indicate that adding extra information such as similar words or words from relevant documents are keys to short text retrieval. This hinted us that only depending on the action field may not be enough and leveraging other useful fields in an intent may bring in benefits.

3 Definition of the Intents Resolving Problem

The service description for an Intents-based service contains the fields of *identifier, action*, *data type*, *title* and other information which may be employed in intent resolving. Moreover, the Web page which the service description tag is embedded can be counted as related details about the service because its content can be introductory or help information about the service. The content is intuitively more informative than other fields in a service description tag as a result of its comparatively long length and abundance of text.

On the other hand, an intent message contains fields of *action*, *data type* and *data*. Resolving an intent message means retrieving a list of candidate services from the service registry by utilizing the information included in service descriptions and the intent message. The resolving process involves the following steps:

1) Check the action field content to see if there is any feature showing that this is an explicit intent message. In our current design, a URI-like action is to be

recognized as a possible service identifier. Some organizations may also prefer URI-like strings to be the actions of their authoritative intents. For instance, "webintents.org" use strings with such style as the content of their intents.

2) If the intent is not an explicit intent, then it will be checked against a registry of public authoritative intents. Exact matching also takes effect in this step. Once it is detected to be an authoritative intent message, the registered services which provide supports to such intent will be returned to the user as candidate services.
3) If the intent message is neither an explicit intent nor an authoritative one, we need to first filter out the services whose data type is not exactly the same as in the intent message. Since Intents adopts MIME types as the way of specifying a data type, if the data type field content of an intent message fails to find a match in any service description, the resolving process should trigger a stop.
4) As for the survivors from the above stage, the action field in the intent message is leveraged as a query representation to retrieve the relevant services based on the content of action, title and other parts in its service description. This stage is an analogy to the process of retrieving a relevant document from a collection in IR wherein false positive errors should be permitted. That means not all the retrieved services in this stage are relevant but most of the top few should satisfy the user service need. Existing IR retrieval model can be applied in this stage to return a ranked list of candidate services.

The first three steps are straightforward and easy to be implemented. The last one needs more investigation and we focus on it most in this work. Formally speaking, if the action field of an intent message is regarded as a query Q and a service description as a document D, a similarity score $S(Q,D)$ needs to be calculated based on the two parts. The score is then employed to order the services: the higher the score, the more relevant the service. We apply an existing IR retrieval model to generate $S(Q,D)$.

Since a service description may contains multiple text fields such as action, title or other parts, taking into consideration one or multiple fields in computing the score may be significant to the overall performance. To examine the differences, we select three fields in a service description: action, title and Web page content among which the Web page content is a long-text field while the other two are short. Seven similarity formulae are constructed by combinations of the three fields, as list in Table 1.

In a service description, the action field is required but the other two are optional. We assume that *only* considering the action field may not achieve an optimal performance to satisfy the user's service needs. As a result, we will explore the effectiveness of all the combinations of the three fields through extensive experiments.

Similarity Formulae	*Explanations*
$S_1(Q,D) = S(action_Q, action_D)$	Query is the action part of intent messages while document only consider the action part in service descriptions
$S_2(Q,D) = S(action_Q, title_D)$	Query is the action part of intent messages while document only consider the title part in service descriptions
$S_3(Q,D) = S(action_Q, webpage_D)$	Query is the action part of intent messages while document only consider the Web page part in service descriptions
$S_4(Q,D) = S_1(Q,D) + S_2(Q,D)$	Equally weighted sum of S_1 and S_2
$S_5(Q,D) = S_1(Q,D) + S_3(Q,D)$	Equally weighted sum of S_1 and S_3
$S_6(Q,D) = S_2(Q,D) + S_3(Q,D)$	Equally weighted sum of S_2 and S_3
$S_7(Q,D) = S_1(Q,D) + S_2(Q,D) + S_3(Q,D)$	Equally weighted sum of S_1, S_2 and S_3

Table 1: Similarity formulae and explanation

4 Empirical Research and Analyses

In this section, we demonstrate a series of experiments conducted to explore the effectiveness when traditional IR techniques are applied to Intents resolving.

4.1 Experiment Set-up

The dataset we employed in the experiments was extracted from a public intent registry on OpenIntents [OpenIntents, 13]. Each entry in the dataset consists of an action field, a title field, a description field and other parts which can be counted as a service description. We selected the action field, the title field and the description field in our experiments. Since the first two are short text while the third one is a field of long text, we used the description field to take the place of the Web page to keep the pattern of two short text fields and one long text field in the dataset. To be consistent with terms in IR, we call the entry with the three fields a *document*. An overview of the dataset is shown in Table 2.

Name	*Statistics*
Documents	83
Queries	88
Query-relevant-document judgments	119
Average relevant document per query	1.35

Table 2:Intents dataset statistics

It is observed that this is a comparatively small dataset. However, all the data are in practice and come from realistic open Android applications registered by their developers with OpenIntents. Therefore, we believe it can reveal realistic

characteristics in intent resolving. We manually generate a set of queries (empirically, more than 50 queries is enough for testing) and their relevant document judgments in the dataset. These queries are in the form of the action field in intent messages.

We adopted Apache Lucene [Lucene, 13] and modified it to become our experiment platform. Lucene has inner implemented retrieval models including vector space model, probability model, and language model. Lucene's implementations of these models are representative. In addition, we implemented an axiomatic retrieval model based on the interfaces provided by Lucene.

Lucene's default retrieval model is an implementation of vector space model which is empirically successful and widely recognized. Its calculation is described as in Formula (1).

$$S(Q,D) = coord(Q,D) \times queryNorm(Q) \sum_{t \in Q} tf(t,D) idf^2(t) boost(t) norm(t,D) \quad (1)$$

More details about the formula are in Table 3.

Item	*Calculation*
$coord(Q,D)$	$\frac{\lvert Q \cap D \rvert}{\lvert D \rvert}$
$queryNorm(Q)$	Not applicable to scoring documents
$tf(t,D)$	$\sqrt{f(t,D)}$
$idf(t)$	$1 + \log \frac{N}{df(t)+1}$
$boost(t)$	Set to 1 in the experiment
$norm(t,D)$	$\frac{1}{\sqrt{\lvert D \rvert}}$

Table 3: Lucene's default retrieval model explanation

In Table 3, $|Q \cap D|$ is the number of terms both in Q and D, $|D|$ the length of D, $f(t,D)$ the count of term t in D, N the number of all indexed documents and $df(t)$ the count of documents containing term t. If only the effective parts are considered, the formula for Lucene's default retrieval model is reduced to Formula (2).

$$S(Q,D) = \frac{|Q \cap D|}{|D|^{\frac{3}{2}}} \sum_{t \in Q \cap D} f^{\frac{1}{2}}(t,D) (1 + \log \frac{N}{df(t)+1})^2 \quad (2)$$

The implementations selected for other models are Okapi BM25 for the probability model, an implementation with Dirichlet prior smoothing for the language model and the F2-EXP one for the axiomatic model. All the implementations and their formulae are listed in Table 4.

In Table 4, *avgdl* is the average length of all the indexed documents. k, b, μ and s are parameters for user setting up. In our experiments we adopted their default settings which averagely achieve best results. Their values are shown in Table 5.

Implementation Name	***Model***	$S(Q,D)$
Lucene Default Model	Vector space model	$\frac{\lvert Q \cap D \rvert}{\lvert D \rvert^{\frac{3}{2}}} \sum_{t \in Q} f^{\frac{1}{2}}(t,D)(1+\log \frac{N}{df(t)+1})^2$
Okapi BM25	Probabilistic model	$\sum_{t \in Q} \log \frac{N-df(t)+0.5}{df(t)+0.5} \times \frac{f(t,D)(k+1)}{f(t,D)+k(1-b+b\frac{\lvert D \rvert}{avgdl})}$
LM Dirichlet	Language model	$\sum_{t \in Q} \log(1+\frac{f(t,D)}{\mu P(t \mid C)}) + \lvert Q \rvert \log(\frac{\mu}{\lvert D \rvert + \mu})$
F2-EXP	Axiomatic model	$\sum_{t \in D} f(t,D) \times \left(\frac{N}{df(t)}\right)^k \times \frac{f(t,D)}{f(t,D)+s+\frac{s\lvert D \rvert}{avgdl}}$

Table 4: Model implementations and their formulae

Implementation Name	***Parameter Settings***
Okapi BM25	k=1.25 b=0.75
LM Dirichlet	μ=2000
F2-EXP	s=0.5 k=0.35

Table 5: Parameter settings for retrieval model implementations

Before retrieval models take effects, some pre-processing must be applied to both queries and documents, including removing stop words and stemming. In addition to the default stop word set in Lucene, we added other stop words such as "com", "intent", and "org" which commonly appear in the action field. As for stemmer, we employed the widely accepted and recognized Porter stemmer [Porter, 80].

4.2 Evaluation Measure

The metrics for evaluation measured in the experiments contains *recall*, *precision*, *F-measure, mean average precision* (MAP) and *mean reciprocal rank* (MRR). Precision is the fraction of retrieved documents that are relevant and recall is the fraction of relevant documents that are retrieved. These two measures are always inversely related which means if one of them increases the other drops. As a result, F-measure is invented as a value by considering both recall and precision. In this work we adopt the F_1 measure, as shown in Formula (3).

$$F_1 = 2\frac{precision \times recall}{precision + recall} \tag{3}$$

Lucene can return ranked results and is capable of setting up the parameter of maximal returned documents. Thus we can control the size of returned top documents and compute recall, precision, and F-measure for the first k documents or at rank k.

MAP is the mean of average precision over all queries and the average precision is the average of the precisions for the top k documents after each relevant document is retrieved. If $R(Q)$ is the set of all the documents which are relevant to query Q and $rank(d)$ is the rank at which document d is retrieved then MAP is formalized as in Formula (4)

$$MAP = \frac{1}{|A_Q|}\sum_{Q\in A_Q}\frac{1}{|R(Q)|}\sum_{D\in R(Q)} Precision(rank(D)) \tag{4}$$

where A_Q is the set of all queries, $rank(D)$ the rank at which document D is retrieved and $Precision(k)$ the precision at rank k. Many experiments have demonstrated that MAP is stable and excels in differentiating retrieval models. The calculation of the MAP in our experiments is slightly apart from what is touched on here. We will explain it late in the paper.

MRR is the average multiplicative inverse of the rank of the first correct retrieved document, as shown in Formula (5).

$$MRR = \frac{1}{|A_Q|}\sum_{Q\in A_Q}\frac{1}{rank_{1st}(Q)} \tag{5}$$

where A_Q is the set of all queries and $rank_{1st}(Q)$ is the rank of the first correct retrieved document of Q. MRR is high when most of relevant documents take the first position in the returned ranking.

4.3 Results and Analyses

The experiments measured and calculated recall, precision, F-measure, MAP, and MRR for the four retrieval model implementations over the seven similarity formulae listed in Table 1. This section presents what we found from the conducted experiments.

In the experiments, we found that it is very difficult to retrieve all the relevant documents for a query in practice. Once the recall comes to a specific value point (always close to but less than 1), it will keep the same until no more document is returned. The reason for this phenomenon is that scoring all the indexed documents is time-consuming because their total number is very huge in practice. As a result most retrieval model implementation usually selects a reasonable fraction of the indexed documents containing most relevant documents and scoring them. Thus the recall at rank k may keep the same when k passes a value point, as illustrated in Figure 2 which is a set of tested recalls in our experiments for Lucene default model on S_7. When k is larger than 8, the recall stops increasing. This means the extra returned documents are irrelevant and noisy once the recall for the top k documents comes to the maximum. Therefore only the top documents with an increased recall are effective because new retrieved documents have the possibility of relevance within such a range. We call these top documents with recall increase *effective top documents*.

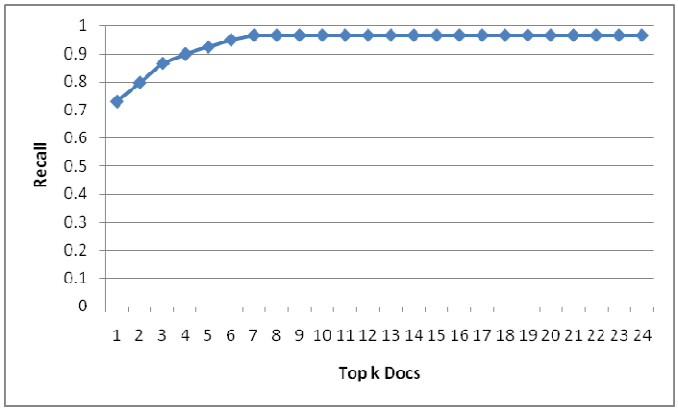

Figure 2: Recall for the Lucene default model on S7

On the other hand, if a relevance model produces a larger maximum recall but has relatively fewer effective top documents, it should also have a better average precision within those documents because they include a larger ratio of relevant documents distributed over a smaller returned document set. This heuristic is also verified by our results listed in Figure 3.

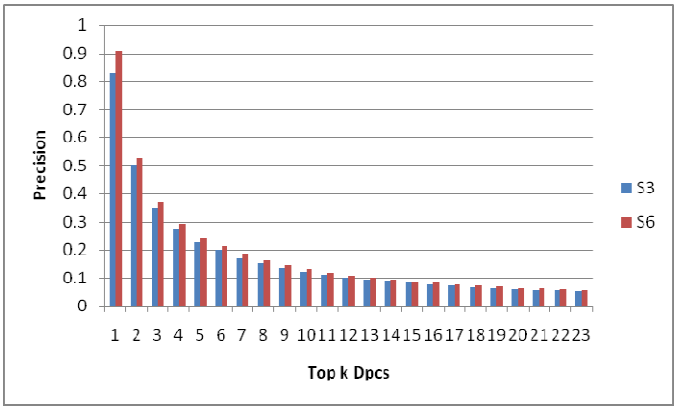

Figure 3: Precision for S3 and S6 of the F2-EXP model

Figure 3 shows the precision comparison of the F2-EXP model on S_3 and S_6. The two executions returned same-sized document sets. However, for S6, F2-EXP produced a larger maximum recall while keep a smaller set of effective top documents. We can see at any rank its precisions on S_6 beats over that of S_3.

In the experiments, all the four retrieval model implementations returned a same sized set of documents and maximum recalls on any similarity formula, but their effective top document sets are different. Figure 4 displays the total retrieved

documents, the effective top documents, and the maximum recall for all the four retrieval model implementations on all the seven similarity formulae.

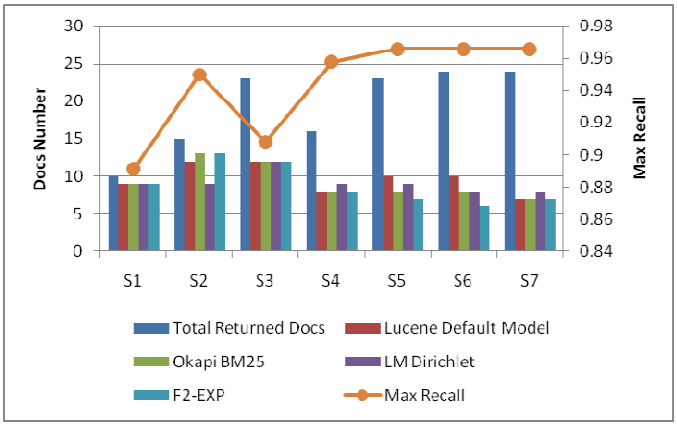

Figure 4: Returned documents, top effective documents and maximum recall

In Figure 4, it is obvious that for S_3 the maximum recall is relatively lower compared to its counterparts while at the same time the amount of effective top documents is larger. The formula only comprises the description field (a long text field) which simulates a related Web page containing the intent-based service description tag. This implies that the description field or the simulated Web page field may not be appropriate for intent resolving on its own. However, it is observed from the chart that $S_5 - S_7$ have a large maximum recall but with just a few effective top documents. These similarity formulae are composite formulae comprising the description field which indicates the description field may significantly improve the effectiveness when combined with other fields such as action, title or both of them.

We also list the results of precision and F-measure to verify the above heuristic in Table 6 and 7. Since S_1 only totally returned 10 documents, we only show the results within top 10 documents for all the participants.

Top k Docs	**Similarity Formulae**						
	S_1	S_2	S_3	S_4	S_5	S_6	S_7
1	0.761	0.920	0.804	0.941	0.904	**0.946**	**0.949**
2	0.469	0.522	0.492	0.528	0.526	0.539	**0.541**
3	0.337	0.372	0.352	0.377	0.372	0.381	**0.386**
4	0.270	0.293	0.276	0.297	0.293	0.302	**0.303**
5	0.230	0.240	0.230	0.248	0.245	**0.253**	0.250
6	0.195	0.202	0.200	0.210	0.213	**0.216**	0.214
7	0.167	0.176	0.172	0.182	0.185	0.185	**0.187**
8	0.148	0.157	0.151	0.162	**0.163**	**0.163**	**0.163**
9	0.134	0.141	0.134	0.144	**0.145**	**0.145**	**0.145**
10	0.120	0.127	0.120	0.130	**0.131**	**0.131**	**0.131**

Table 6: Precision

Top k Docs	**Similarity Formula**						
	S_1	S_2	S_3	S_4	S_5	S_6	S_7
1	0.647	0.783	0.683	0.800	0.773	0.804	**0.807**
2	0.560	0.622	0.587	0.631	0.631	0.642	**0.646**
3	0.465	0.513	0.486	0.520	0.512	0.525	**0.533**
4	0.403	0.437	0.413	0.445	0.437	0.451	**0.452**
5	0.361	0.377	0.361	0.390	0.379	**0.398**	0.393
6	0.318	0.330	0.326	0.343	0.346	**0.352**	0.349
7	0.280	0.295	0.288	0.304	**0.313**	0.310	0.312
8	0.253	0.268	0.258	0.277	**0.279**	**0.279**	**0.279**
9	0.233	0.245	0.233	0.250	**0.252**	**0.252**	**0.252**
10	0.212	0.223	0.212	0.228	**0.230**	**0.230**	**0.230**

Table 7: F-measure

The best results in every row are marked in bold type. All of them fall in the columns of $S_5 - S_7$ which are field combinations including the description field. Here we draw out the first heuristic acquired based on the above analyses:

Rule 1. *The description or related Web page field should not be used in intent resolving alone, but it is highly recommended that it be used collaboratively with other fields.*

Since the retrieval model implementations on all the similarity formulae are incapable of returning all relevant documents in the experiments, we changed the MAP metric slightly to make it fit for the real situation. Instead of averaging precision over each retrieved relevant document, we make means over the top document sets whose size is smaller than the effective top document set. If $E(Q)$ is the set of the effective top documents for query Q, $Precision(k)$ is the precision at rank k, then Formula (4) becomes

$$MAP = \frac{1}{|A_Q|} \sum_{Q \in A_Q} \frac{1}{|R(Q)|} \sum_{k \le |E(Q)|} Precision(k) \qquad (6)$$

Formula (6) is the MAP we applied in the experiments. The results are listed in Table 8.

Similarity Formulae	*Lucene Default Model*	*Okapi BM25*	*LM Dirichlet*	*F2-EXP*
S_1	**0.3016**	**0.3016**	0.3009	0.3009
S_2	0.2809	0.2661	**0.3371**	0.2665
S_3	0.2642	0.2598	0.2593	**0.2644**
S_4	**0.3698**	0.3663	0.3426	0.3688
S_5	0.3246	0.3608	0.3312	**0.3916**
S_6	0.3292	0.3756	0.3629	**0.4432**
S_7	**0.4106**	0.4040	0.3658	0.4073

Table 8: MAP of retrieval models on similarity formulae

In Table 8, the best retrieval model on each similarity formulae are marked in bold type. It is interesting that S_6 achieves the best result for F2-EXP compared with other similarity formulae. We believe the restriction imposed on action design which contains some information compromises the final performance. However, the action field is a required field in service description and S_6 is a composite similarity consisting of two optional fields in service description which may be missing in practice. Thus only depending on the best strategy may not be reasonable and an adaptive design should consider a good methodology in every practical condition.

Since action is a required field and the description field is a substitute for the Web page field, we summarize the best strategy in every possible condition and list them in Table 9.

	Web page	***NOT Web page***
Title	S_6(F2-EXP)	S_4(Lucene Default Model)
NOT Title	S_5(F2-EXP)	S_1(Lucene Default Model)

Table 9: Best strategy selection in possible condition

In Table 9, "NOT" means such field is an absence. For each condition, we compared all possible strategies and selected the best solution. For instance, for the condition when the action field, the title field and the Web page field all appear, we compared all the seven similarity formulae and selected the FX-EXP (the axiomatic model) on S_6 instead of the Lucene's default model implementation (the vector space model) on S_7. This selection indicates another heuristic:

Rule 2. *Utilizing all the appearing fields in intent resolving may not achieve the best effectiveness. When the Web page field appears, FX-EXP is the best choice. Lucene's default model implementation is better if the Web page field is absent.*

User utility of a system is determined by to what degree it can satisfy the user instead of only considering its excellence on some evaluation measures. Some users may be concerned if the first few top documents containing the service they want. This requirement can be achieved by comparing the ranking of the first relevant document and selecting the best solution from among all the alternatives. MRR is a metric indicative of the ranking of the first relevant document which can assist us in accomplishing such requirement.

Similarity Formulae	***Lucene Default Model***	***Okapi BM25***	***LM Dirichlet***	***F2-EXP***
S_1	**0.834**	**0.834**	0.832	0.832
S_2	**0.95**	0.948	0.949	0.949
S_3	**0.879**	0.858	0.847	0.878
S_4	**0.974**	0.961	0.962	0.969
S_5	**0.973**	0.941	0.916	0.949
S_6	**0.981**	**0.981**	0.936	**0.981**
S_7	**0.994**	0.972	0.946	0.983

Table 10: MRR

Table 10 displays MRR measures for the four retrieval model implementations on the seven similarity formulae. The best model on each formula is marked in bold type. This time the Lucene's default model implementation wins. Similar to Table 9, Table 11 is created to shown the best strategy when MRR is applied.

	Web page	***NOT Web page***
Title	S_7(Lucene Default Model)	S_4(Lucene Default Model)
NOT Title	S_5(Lucene Default Model)	S_1(Lucene Default Model)

Table 11: Best strategy for the first correct ranking

Once the effective top documents are retrieved, the extra returned ones are irrelevant and noisy which may disturb users. The percentage of effective top documents for the selected strategies are listed in Table 12.

Strategy	***Effective Top Documents Ratio (%)***
S1(Lucene Default Model)	90
S4(Lucene Default Model)	50
S5(Lucene Default Model)	43.48
S5(F2-EXP)	30.43
S6(F2-EXP)	25
S7(Lucene Default Model)	29,17

Table 12: Effective top document ratio for the selected strategies

Table 9, 11, 12 and the two rules are the conclusions drew from the experiments based on a real dataset which constitutes the guiding principles for us to design the adaptive intent resolving methodology which will be presented in the next section.

5 An Adaptive Intent Resolving Method

In this section, we propose an adaptive intent resolving method based on the guiding principles from our empirical study and integrate it with our Intents agent which is developed on Android [Zheng, 13b].

In realistic situations, an intent message may be explicit (with service identifier taking up the action field), authoritative (where the intent structure has been specified by some authoritative institute for public use) or naïve (e.g., the flexible intent with the action field saying the meaning of what should be performed). The first two categories require exact matching while the last one only needs similar matching.

On the other hand, the service description cannot uniformly contain the same fields. It is most possible that a mixture of service descriptions with different fields missing appears in the index. Table 13 presents a sample for the possible index appearance in practice. Each row represents a service description entry and a "NULL" value means the field is missing.

Service ID	*Action*	*Title*	*Web Page Content*	*Data Type*
http://202.117.1.119/share	share a link	Facebook link share	NULL	text/url-list
http://202.117.0.200/get-a-weather	local weather	NULL	<html> <body> This is a local weather service. … </body> </html>	application /json
http://202.117.1.119/shorten	http://webintents.org/shorten	NULL	NULL	text/uri-list

Table 13: Service description index sample

An intent resolving module should be able to deal with the above issues. Figure 5 shows how to construct an intent resolving module.

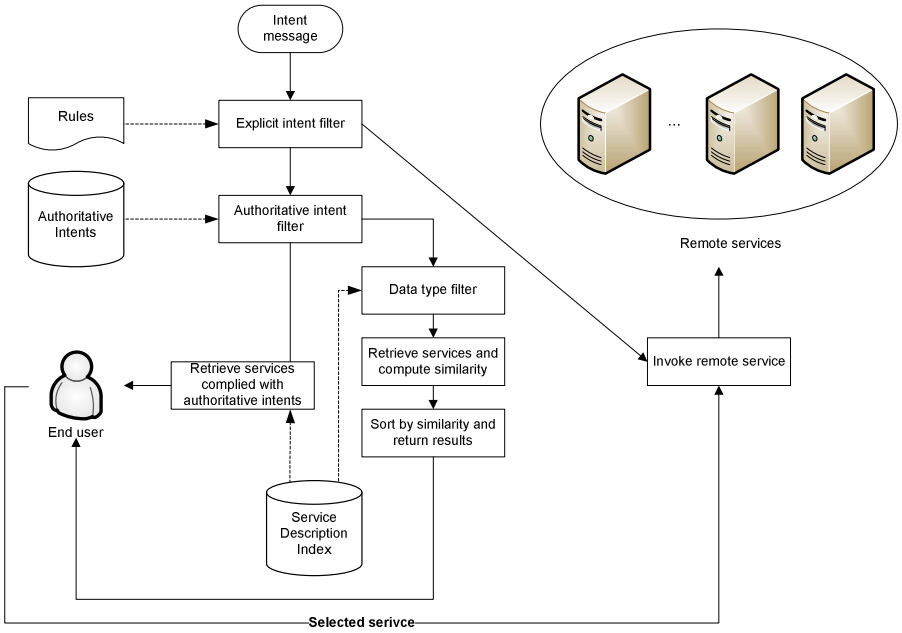

Figure 5: Intent resolving module

As shown in Figure 5, two registries are in need to resolve an intent message: a public authoritative intent registry and a service description index. The resolving procedure of an intent message starts with check if the incoming intent is explicit based on pre-defined rules. Once an intent message is detected as an explicit intent, it will be delivered to the specified service directly. Otherwise, it is checked to see

whether it is an authoritative intent according to the authoritative intent registry. If the intent is an authoritative intent, all the registered services corresponding to the authoritative protocol will return to the user as candidate services. If both of the two steps fail, the intent message is delivered for data type filtering, relevance services retrieving, similarity computing and sorting. After all a list of candidate services are returned to the user. After the user makes a choice, the selected service will be invoked.

5.1 Average Precision First Approach

If an intent message is neither an explicit intent nor authoritative intent, then techniques to retrieve a relevant service from the service description index are needed where the guiding principles obtained from our empirical research apply. Assume *I* is the service description index where every document *d* in *I* has the form, d=<action, title, webpage, datatype>. In addition, q=<action, datatype> is the intent message. Figure 6 shows the process of adaptively retrieving services from the service description index.

Input: Service description index I where every document d in I, d=<action, title, webpage, datatype>
Intent message q=<action, datatype>
Output: A ranked list of candidate services

1 Select the subset C in I where every document d in C, d.datatype = q.datatype
2 Detect the commonly not-null fields F in C
3 if webpage in F
  then set F2-EXP as the retrieval model implementation
    if title in F
      then
        set similarity formula to S_6
        set threshold to 0.25
      else
        set similarity formula to S_5
        set threshold to 0.3043
  else set the Lucene's default model implementation as retrieval model
    if title in F
      then
        set similarity formula to S_4
        set threshold to 0.5
      else
        set similarity formula to S_1
        set threshold to 0.9
4 Use the selected similarity formula and retrieval model implementation to generate a ranked list of relevant document from C
5 Use the threshold to keep the top ranked services while disposing of the remaining ones which have a very small fraction of relevant services

Figure 6: Algorithm of MAP-based adaptively retrieving services from the index based on similarity formula and retrieval model

As demonstrated in Figure 6, the algorithm begins with removing services whose data type are different from the intent message. Then it detects the commonly not-null fields in the remaining services. The similarity formula and retrieval model are adaptively set according to the principles found in the previous section. Afterwards, an ordered list of relevant documents is returned to the user through the similarity formula and retrieval model.

5.2 MRR First Approach

Some users may prefer to check the returned list in a top-down style to seek the relevant service and expect the first service on the list to be the service they really want. Therefore MRR may be more effective than MAP as a guide in such circumstance. We add a switch by which users can choose MAP or MRR as the guide to generate the recommended services.

Since ordinary users may have little idea with the concept of MRR, we adopt a text "Find service quicker" to let user switch on/off MRR retrieving. Similarly we list the MRR-based retrieving steps in Figure 7.

Input:	Service description index I where every document d in I, d=<action, title, webpage, datatype> Intent message q=<action, datatype>
Output:	A ranked list of candidate services
1	Select the subset C in I where every document d in C, d.datatype = q.datatype
2	Detect the commonly not-null fields F in C
3	Set Lucene's default model implementation to be the retrieval model
4	if webpage in F if title in F then set similarity formula to S_7 set threshold to 0.2917 else set similarity formula to S_5 set threshold to 0.4348 else if title in F then set similarity formula to S_4 set threshold to 0.5 else set similarity formula to S_1 set threshold to 0.9
5	Use the selected similarity formula and retrieval model implementation to generate a ranked list of relevant document from C
6	Use the threshold to keep the top ranked services while disposing of the remaining ones which have a very small fraction of relevant services

Figure 7: Algorithm of MRR-based adaptively retrieving services from the index based on similarity formula and retrieval model

5.3 Service Description Index Implementation

The public authoritative intent registry is much smaller than the service description index and only needs exact matching. Thus a hash table is capable of satisfying the requirement. As for the service description index implementation, we adopt Apache Lucene. As of its 4.x version, Lucene begins to employ service provider interfaces (SPI) to support flexible code-decode module loading. However, this feature prevents Lucene from applying on Android. We changed the source code of Lucene by turning off such feature and creating an Android-compatible Lucene. Because of the excellent capability in text parsing and indexing, the Intents user agent [Zheng, 13b] is now integrated with Lucene in service indexing.

6 Conclusion and Future Work

Service discovery and integration has been an active research field for about a decade. However, current techniques usually rule application end users out of this process which reduces system flexibility. Intents is an emerging and innovative technique that considers end user participation and opens a new promising direction. In Intents, Intent resolving plays a critical role. Existing strategies applied in Intents are based only on the simple exact match strategy. This paper continues our previous work on Intents [Zheng, 13a; Zheng, 13b] and attempts to bring in techniques in information retrieval and short text retrieval to improve the process of intent resolving.

This paper developed a set of composite similarity formulae based on the fields in Intents-based service description which can be used to formalize the intent resolving process. An empirical study was conducted on a real intent dataset. Based on the results of the empirical study and a detailed analysis of these results, an adaptive intent resolving approach has been developed. The proposed approach has also been integrated with a previously developed Intents user agent [Zheng, 13b].

However, significant research efforts are still required in this area. The dataset used in the work is relatively small and we need to collect more to validate the guiding design principles. Besides, with the development of information retrieval techniques, the scheme proposed in this paper may need adjustments to accommodate state-of-the-art information retrieval technologies.

Parts of the work in this paper may be applied to integrate Web Intents and Android Intents to improve the interoperability between Web and local services which is another undergoing research work. Once Intents is applied on mobile devices, context information from their sensors [Espada, 12] is also a promising aspect for assisting the resolving process.

References

[Efron, 12] Efron, M., Organisciak, P., Fenlon, K.: Improving Retrieval of Short Texts through Document Expansion, in Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 12), Portland, OR, USA, 2012, pp. 911 – 920.

[Espada, 12] Espada, J., Martínez, O., G-Bustelo, B., Lovelle, J., Pablos, P.: A Simple Model Based on Web Services to Exchange Context Information between Web Browsers and Web Applications, Journal of Universal Computer Science, vol. 18 issue 11, pp. 1410 – 1431, 2012.

[Fang, 07] Fang, H.: An axiomatic approach to information retrieval, PhD dissertation, University of Illinois at Urbana-Champaign, 2007.

[Fellbaum, 98] Fellbaum, C.: WordNet: An Electronic Lexical Database, MIT Press, 1998.

[Fuhr, 92] Fuhr, N.: Probabilistic Models in Information Retrieval, The Computer Journal, vol. 35 issue 3, pp. 243 – 255, 1992.

[Imran, 09] Imran, H., Sharan, A.: Thesaurus and Query Expansion, International Journal of Computer science & Information Technology, vol. 1 issue 2, 89 – 97, 2009.

[Lavrenko, 01] Lavrenko, V., Croft, W. B.: Relevance-Based Language Models, in Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 01), New York, NY, USA, 2001, pp. 120 – 127.

[Lucene, 13] Apache Lucene, 2013, http://lucene.apache.org/

[McCandless, 10] McCandless, M., Hatcher, E., Gospodnetic, O.: Lucene in Action, 2nd edition, Manning, 2010.

[Manning, 09] Manning, C. D., Raghavan, P., Schütze, H.: An Introduction to Information Retrieval, Cambridge University Press, 2009.

[Mian, 09] Mian, A. N., Baldoni, R., Beraldi, R.: A survey of service discovery protocols in multihop mobile ad-hoc networks, IEEE Pervasive Computing, vol. 8 issue 1, pp. 66 – 74, 2009.

[Neyem, 08] Neyem, A., Ochoa, S. F., Pino, J. A.: Integrating Service-Oriented Mobile Units to Support Collaboration in Ad-hoc Scenarios, Journal of Universal Computer Science, vol. 14 issue 1, pp. 88 – 122, 2008.

[OpenIntents, 13] OpenIntents, 2013, http://www.openintents.org/en/

[Pan, 11] Pan, Y., Tang, Y., Li, S.: Web Services Discovery in a Pay-As-You-Go Fashion, Journal of Universal Computer Science, vol. 17 issue 14, pp. 2029 – 2047, 2011.

[Pedrinaci, 10] Pedrinaci, C., Domingue, J.: Toward the Next Wave of Services: Linked Services for the Web of Data, Journal of Universal Computer Science, vol. 16 issue 13, pp. 1694 – 1719, 2010.

[Ponte, 98] Ponte, J. M., Croft, W. B.: A Language Modeling Approach to Information Retrieval, in Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 98), New York, NY, USA, 1998, pp. 275 – 281.

[Porter, 80] Porter, M. F.: An Algorithm for Suffix Stripping, Program, vol. 14 issue 3, pp. 130 – 137, 1980.

[Robertson, 76] Robertson, S., Jones, K. S.: Relevance Weighting of Search Terms, Journal of the American Society for Information Science, vol. 27 issue 3, pp. 129 – 146, 1976.

[Robertson, 95] Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M., Gatford, M.: Okapi at TREC-3, in Proceedings of the 3rd Text REtrieval Conference (TREC-3), 1995, pp.109 – 126.

[Robertson, 99] Robertson, S. E., Walker, S., Hancock-Beaulieu, M., Okapi at TREC-7, in Proceedings of the 7th Text REtrieval Conference (TREC-7), Gaithersburg, USA, 1999, pp. 253 – 264.

[Sahami, 06] Sahami, M., Heilman, T. D.: A Web-Based Kernel Function for Measuring the Similarity of Short Text Snippets, in Proceedings of the 15th International Conference on World Wide Web (WWW 06), New York, NY, USA, 2006, pp. 377 – 386.

[Salton, 75] Salton, G., Wong, A., Yang, C.: A Vector Space Model for Automatic Indexing, Communications of the ACM, vol. 18 issue 11, pp. 613 – 620, 1975.

[Salton, 83] Salton, G., McGill, M. J.: Introduction to modern information retrieval, McGraw-Hill, 1983.

[Shen, 07] Shen, W., Hao, Q., Wang, S., Li, Y., Ghenniwa, H.: An agent-based service-oriented integration architecture for collaborative intelligent manufacturing, Robotics and Computer-Integrated Manufacturing, vol. 23 issue 3, pp. 315 – 325, 2007

[Singhal, 96] Singhal, A., Buckley, C., Mitra, M.: Pivoted document length normalization, in Proceedings of the 19th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 96), Zurich, Switzerland, 1996, pp. 21 – 29.

[Theaurus, 13] Wikipedia Theaurus, 2013, http://en.wikipedia.org/wiki/Thesaurus

[WebIntents, 13] WebIntents, 2013, http://webintents.org/

[Walker, 01] Walker, D., Query Expansion using Thesauri: Previous Approaches and Possible New Directions, University of California, Los Angeles, 2001.

[Zhai, 01] Zhai, C., Lafferty, J., A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval, in Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 01), New York, NY, USA, 2001, pp. 334 – 342.

[Zheng, 13a] Zheng, C., Shen, W., Ghenniwa, H. H.: An Intents-Based Approach for Service Discovery and Integration, in Proceedings of 2013 IEEE 17th International Conference on Computer Supported Cooperative Work in Design (IEEE CSCWD 2013), Whistler, BC, Canada, 2013, pp. 207 – 212.

[Zheng, 13b] Zheng, C., Shen, W., Ghenniwa, H. H.: Design and Implementation of Intents User Agent, in Proceedings of 2013 IEEE 17th International Conference on Computer Supported Cooperative Work in Design (IEEE CSCWD 2013), Whistler, BC, Canada, 2013, pp. 275 – 280.

[Zhu, 05] Zhu, F., Mutka, M. W., Ni, L. M.: Service Discovery in Pervasive Computing Environments, IEEE Pervasive Computing, vol. 4 issue 4, pp. 81 – 90, 2005.