Discovering Web Services to Improve Requirements Specifications: Does It Help?

Konstantinos Zachos, Neil Maiden, and Rhydian Howells-Morris

Centre for HCI Design, City University, London
kzachos@soi.city.ac.uk, n.a.m.maiden@city.ac.uk,
rhydianmorris2002@hotmail.com

Abstract. Service-centric systems pose new opportunities for when engineering requirements. This paper reports an evaluation of software tools with which to exploit discovered services to improve the completeness of requirements specifications. Although these tools had been evaluated previously in facilitated industrial workshops, industrial users had not used the tools directly. In this paper we report 2 industrial uses and evaluations in which experienced analysts used the tools directly on 2 real-world requirements projects. Results reveal that analysts used the tools to retrieve web services that could implement specified requirements, but analysts were less able to improve these requirements in light of the retrieved services. Results have implications for iterative service discovery processes and service discovery algorithms.

1 Developing with Web Services

Web and software services are operations that users access via the internet through a well-defined interface independent of where the service is executed [1]. Service-centric systems integrate software services from different providers into applications that discover, compose and monitor these services. Developments in service-centric computing have been rapid [2], but there has been little reported research to address how to engineer service-centric systems.

As we have reported previously [3], one consequence of service-centric systems is that requirements processes might change due to the availability of services. Discovering candidate services can enable analysts to increase the completeness of system requirements based on available service features. We have researched new tools and techniques to form service queries from incomplete requirements specifications as part of the EU-funded SeCSE Integrated Project. Although the effectiveness of these tools to increase requirements completeness was demonstrated in workshops, in which stakeholders worked with the tools through facilitators and scribes [4], we still lacked empirical evidence of whether analysts can use and benefit from these tools directly. Therefore we made the tools available for use by SeCSE's industrial partners on service-centric systems development projects. This paper reports results from the requirements phases of projects at 2 of these partners – a large multi-national consultancy and a small software house providing applications. Results were used to investigate 2 research questions about the usefulness of the SeCSE tools:

Q1. Can the tools retrieve specifications of services compliant with requirements specified by analysts in service queries?

Q2. Can analysts make requirement specifications more complete using the retrieved service specifications?

To answer these 2 questions we collected data about the requirements specified by analysts, the service specifications retrieved using service queries composed of these requirements, analysts' decisions to retain or reject each of these services, changes to requirements in light of service specifications, and qualitative statements made by analysts. Q1 was answered using analyst decisions to retain or reject each retrieved service. Q2 was answered using post-retrieval changes that analysts make to use case and requirement specifications.

Sections 2 and 3 of this paper describe SeCSE's service-centric requirements process and tools. Section 4 introduces the 2 industrial users of these tools and the evaluation method, and sections 5 and 6 report results from the 2 evaluations. Section 7 answers the 2 research questions and reports some threats to validity. The paper ends with future work on new software modules to support the specification of requirements from retrieved web services.

2 Discovering Services in SeCSE

In previous SeCSE work we had reported an iterative and incremental requirements process for service-centric systems [5]. Requirements analysts form service queries from a requirements specification to retrieve services that are related to the requirements. Descriptions of these retrieved services are explained to stakeholders, then used to refine and complete the requirements specification to enable more accurate service retrieval, and so on.

Relevance feedback, as it is known, has important advantages for the requirements process. Stakeholders will rarely express complete requirements at the correct levels of abstraction and granularity to match to the descriptions of available services. Relevance feedback enables service consumers and analysts to specify new requirements and re-express current ones to increase the likelihood of discovering compliant services. Furthermore accurate relevance feedback provides information about whether requirements can be satisfied by available services, to guide the analysts to consider build, buy or lease alternatives or to trade-off whether requirements can be met by the available services.

The process has 2 important features. Firstly, to ensure its industrial uptake, the process uses established specification techniques based on structured natural language. For example, to specify system behaviour the process supports UML use case specifications. To specify the required properties in a testable form for generating service monitoring policies it supports the VOLERE requirements shell [6]. As such the process extends the Rational Unified Process (RUP) without mandating unnecessary specification or service querying activities.

Secondly the process uses services that are discovered from service registries to challenge system boundaries and discover new requirements. For example, if no services are found with an initial query, SeCSE provides advice on how to broaden the query to find services that, though not exactly matching the needs of the future system, might provide a useful basis for further specification.

To support the iterative and incremental requirements process we implemented the SeCSE service discovery environment. The next section describes this environment.

3 SeCSE's Service Discovery Environment

The environment has 4 modules: (i) service registries; (ii) UCaRE, a module to document requirements and generate service queries; (iii) EDDiE, the service discovery engine, and; (iv) the Service Browser module for reviewing and selecting retrieved services. We describe these 4 modules in turn.

3.1 SeCSE's Service Registries

The environment discovers services from federated SeCSE service registries that store both the service implementation that applications invoke and one or more facets that specify different aspects of each service. Current service registries such as UDDI are inadequate for discovering services using criteria such as cost, quality of service and exception handling. Therefore SeCSE has defined 6 facets of a service – signature, description, operational semantics, exception, quality-of-service, cost/commerce, and testing [7] – that specify features that are important when discovering services. Each facet is described using an XML data structure. The environment uses the description and quality-of-service facets of each service. Figure 1 shows part of the service description facet of one service retrieved in 1 of the 2 reported evaluations. The quality-of-service facet is used to refine selection once services are discovered. SeCSE's service registries are implemented using eXist, an Open Source native XML database featuring index-based XQuery processing, automatic indexing.

Name: ViaMichelinFindNearByPOIwebservice
Service goal: ServiceGoal: The FindNearbyPOI Web Service allows you to search a list of POI matching specified criteria located around a central point
Short service description: The "FindNearbyPOI" Web Service allows searching for a certain number of ad-dresses or locations closest 'as the crow flies' to a particular address or place of interest within a user-definable search radius. Then displaying any detailed poi information is possible. For example, it can look for car dealers closest to a given location or find competitors that are closest to your sales points and, as a result, analyze catchment areas that are the least well served

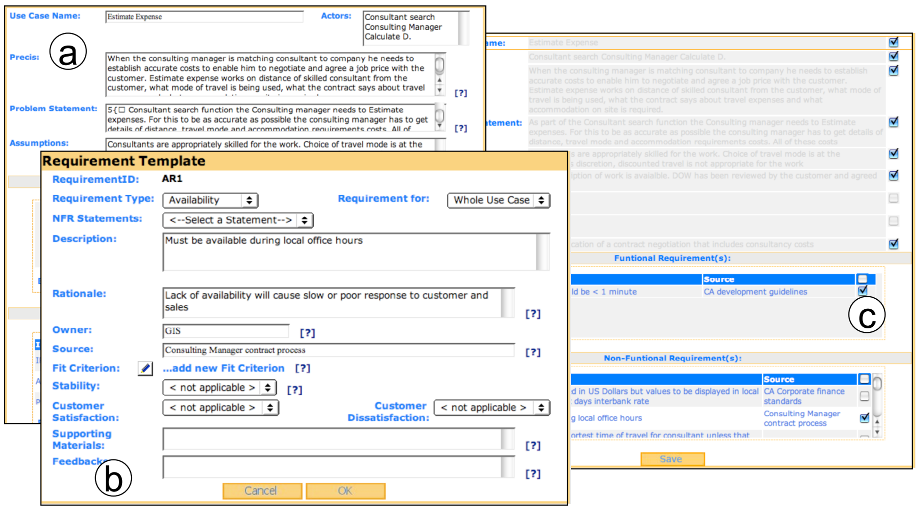
Fig. 1. Example of part of one service with SeCSE's description facet

3.2 The UCaRE Requirements Module

Analysts express requirements for new applications using UCaRE, a web-based .NET application. UCaRE supports tight integration of use case and requirements specifications – a requirement expressed using VOLERE can describe a system-wide requirement, a requirement on the behavior specified in one use case, or a requirement on the behavior expressed in one use case action. UCaRE allows analysts to create service queries directly from use case and requirements specifications.

At the start of the requirements process, analysts work with stakeholders to develop simple use case précis that describe the required behaviour of a new system. Figure 2(a) shows a use case précis expressed in UCaRE, taken from one of the reported evaluations, to specify how a user shall use an on-line ticket searching application. A second précis also used in the evaluation is reported in a readable form in Figure 3. Figure 2(b) shows a requirement, also from these partners, associated with the précis expressed using the UCaRE VOLERE shell. Larger versions of the screenshots in Figure 2 are available at [8]. Requirements also used in the evaluation are reported in readable form in Figure 3.

The analyst then uses the simple tick-box feature shown in Figure 2(c) to select attributes of use cases and requirements to include in a service query. Each service query is formed of one or more elements of a pre-defined type such as a requirement description or rationale, or a use case précis, pre-condition or action. UCaRE maps these element types to service query elements to deliver the seamless integration of service querying with requirements specification, as described at length in [3]. The



Use Case Name: Estimate Expense
Actors: Consultant search Consulting Manager Calculate D.
Precis: When the consulting manager is matching consultant to company he needs to establish accurate costs to enable him to negotiate and agree a job price with the customer. Estimate expense works on distance of skilled consultant from the customer, what mode of travel is being used, what the contract says about travel
Problem Statement: 5{□ Consultant search function the Consulting manager needs to Estimate expenses. For this to be as accurate as possible the consulting manager has to get details of distance, travel mode and accommodation requirements costs. All of
Assumptions: Consultants are appropriately skilled for the work. Choice of travel mode is at the

Requirement Template

RequirementID: AR1
Requirement Type: Availability
Requirement for: Whole Use Case
NFR Statements: <--Select a Statement-->
Description: Must be available during local office hours
Rationale: Lack of availability will cause slow or poor response to customer and sales
Owner: GIS
Source: Consulting Manager contract process
Fit Criterion: ...add new Fit Criterion
Stability: < not applicable >
Customer Satisfaction: < not applicable >
Customer Dissatisfaction: < not applicable >
Supporting Materials:
Feedback:

ame: Estimate Expense ✓
Consultant search Consulting Manager Calculate D. ✓
When the consulting manager is matching consultant to company he needs to establish accurate costs to enable him to negotiate and agree a job price with the customer. Estimate expense works on distance of skilled consultant from the customer, what mode of travel is being used, what the contract says about travel expenses and what accommodation on site is required. ✓
atement: As part of the Consultant search function the Consulting manager needs to Estimate expenses. For this to be as accurate as possible the consulting manager has to get details of distance, travel mode and accommodation requirements costs. All of these costs ✓
s are appropriately skilled for the work. Choice of travel mode is at the s discretion, discounted travel is not appropriate for the work ✓
iption of work is avaialble. DOW has been reviewed by the customer and agreed ✓
☐
☐
cation of a contract negotiation that includes consultancy costs ✓

Funtional Requirement(s):

	Source	☐
ld be < 1 minute	CA development guidelines	✓

Non-Funtional Requirement(s):

	Source	☐
d in US Dollars but values to be displayed in local days interbank rate	CA Corporate finance standards	☐
g local office hours	Consulting Manager contract process	✓
ortest time of travel for consultant unless that		

Fig. 2. Specification of a use case (a) and requirement (b) in UCaRE, and selection of use case and requirements attributes to generate service queries (c)

Precis:	All users can search events and tickets by use of search function. She can search event and ticket on base of event date or event name or event place or interpreter (band, sport team, etc.) name. She can specify also type of event. If search result is more then one, they are displayed like list. Number of list rows on page can be limited by application variable.
AR:	Must be available during local office hours.
FR:	For registered and logged user must be possibility to book or purchase tickets directly from list.

Fig. 3. A simple use case précis and requirements for the ticket searching application used in one of the industrial evaluations, which are used to formulate queries and discover services

analyst then refines each generated service query using the names and locations of registries to search, the maximum number of services to retrieve, and the parts of speech (e.g. noun, verb and adjective) in the service query text to search on.

An analyst using UCaRE can generate one or more service queries from the specification of a system. Each query is a structured XML file containing structured natural language statements. Because these statements are derived from requirements and use cases, each is potentially ambiguous and incomplete. EDDiE, the service discovery engine, was designed to handle this ambiguity and incompleteness.

3.3 The EDDiE Module

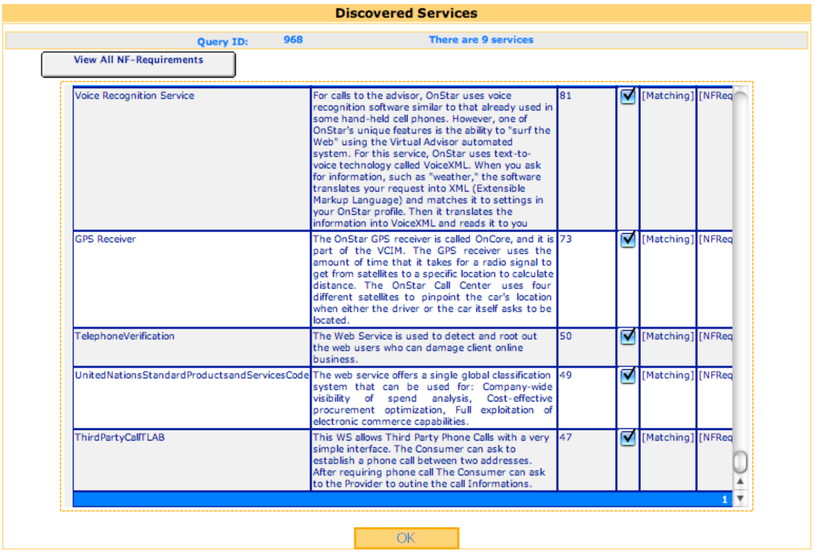
The purpose of EDDiE is to discover descriptions of candidate services using the service description facet with queries composed of information such as that in Figures 2 & 3. Other requirement types and service facets such as quality-of-service and cost fulfil important roles during service selection once discovered using the Service Browser module.

The EDDiE algorithm has the 4 key components. In the first the service query is divided into sentences, then tokenized and part-of-speech tagged and modified to include each term's morphological root (e.g. *displayed* to *display*, and *tickets* to *ticket*). Secondly, the algorithm applies procedures to disambiguate each term by defining its correct sense and tagging it with that sense defined in the WordNet online lexicon [9] (e.g. defining a *ticket* to be a *a commercial document showing that the holder is entitled to something (as to ride on public transportation or to enter a public entertainment* rather than *a list of candidates nominated by a political party to run for election to public offices*). Thirdly, the algorithm expands each term with other terms that have similar meaning according to the tagged sense using WordNet, to increase the likelihood of a match with a service description (e.g. the term *ticket* is synonymous with the term *order* or *voucher* which is also included in the query). This query expansion enables the algorithm to retrieve service specifications for service queries that share no common terms. In the fourth component the algorithm matches all expanded and sense-tagged query terms to a similar set of terms that describe each candidate service, expressed using the service description facet, in the SeCSE service registry. Query matching is in 2 steps: (i) XQuery text-searching functions to discover an initial set of services descriptions that satisfy global search constraints; (ii) traditional vector-space model information retrieval, enhanced with WordNet, to further refine and assess the quality of the candidate service set. This two-step approach overcomes XQuery's limited text-based search capabilities. The algorithm returns a set of retrieved service specifications and match scores ranked according to the semantic distance to the service query.

The EDDiE algorithm is described at length in [3].

3.4 The Service Browser Module

The Service Browser presents retrieved services to analysts and stakeholders. Services that attain a minimum threshold of match value are presented in a ranked order. The analyst can view all properties of the service description facet and corresponding use



Voice Recognition Service	For calls to the advisor, OnStar uses voice recognition software similar to that already used in some hand-held cell phones. However, one of OnStar's unique features is the ability to "surf the Web" using the Virtual Advisor automated system. For this service, OnStar uses text-to-voice technology called VoiceXML. When you ask for information, such as "weather," the software translates your request into XML (Extensible Markup Language) and matches it to settings in your OnStar profile. Then it translates the information into VoiceXML and reads it to you	81	✓	[Matching]	[NFReq
GPS Receiver	The OnStar GPS receiver is called OnCore, and it is part of the VCIM. The GPS receiver uses the amount of time that it takes for a radio signal to get from satellites to a specific location to calculate distance. The OnStar Call Center uses four different satellites to pinpoint the car's location when either the driver or the car itself asks to be located.	73	✓	[Matching]	[NFReq
TelephoneVerification	The Web Service is used to detect and root out the web users who can damage client online business.	50	✓	[Matching]	[NFReq
UnitedNationsStandardProductsandServicesCode	The web service offers a single global classification system that can be used for: Company-wide visibility of spend analysis, Cost-effective procurement optimization, Full exploitation of electronic commerce capabilities.	49	✓	[Matching]	[NFReq
ThirdPartyCallTLAB	This WS allows Third Party Phone Calls with a very simple interface. The Consumer can ask to establish a phone call between two addresses. After requiring phone call The Consumer can ask to the Provider to outine the call Informations.	47	✓	[Matching]	[NFReq

1

Fig. 4. The Service Browser module, showing retrieved services, their specifications and match scores

case and requirement properties to enable understanding and selection. The analyst can also filter the services according to compliance or otherwise with non-functional requirements included in the service query. A screenshot showing candidate services retrieved for queries in one evaluation is depicted in Figure 4. A larger version of the screenshot in Figure 4 is available at [10].

Previously SeCSE's service discovery environment had been tested with industrial users for its usability and core functionality [11]. It was also evaluated successfully in a half-day workshop at FIAT's research centre to discover requirements for new automotive applications [4]. However, the FIAT analysts did not use the tool directly. In the remainder of this paper we report the next phase of evaluation, in which analysts used the environment on their own to specify requirements and retrieve candidate service specifications.

4 The Industrial Users and Evaluation Method

The 2 industrial users – both members of the SeCSE project – were CA and KD Software. CA is one of the world's largest IT management software providers. It undertakes core systems integration roles for clients seeking secure, service-oriented architectures. KD Software is an independent software developer in the Czech Republic that develops business systems using service-centric techniques.

Two experienced analysts – 1 from the UK office of CA and 1 from the Czech office of KD Software – undertook the evaluations. Both had extensive analytic experience and were familiar with UML. Both worked remotely at their offices, accessing SeCSE tools on servers based in London using thin web clients through their organizations' firewalls. The CA analyst received initial on-site training with the tools whereas the KD Software analyst learned to use the tools independently using SeCSE user guides. The SeCSE tools searched 4 federated service registries located in Italy and Spain. They contained 154 service specifications for applications including weather reporting, flight booking and route planning taken from existing public UDDI registries and generated by service providers in SeCSE. Each service description was written by the original service provider and not modified prior to use.

The CA analyst specified requirements for a travel cost estimation application for CA consultants to use. The KD Software specified requirements for an outline ticket searching application. The analysts worked independently of each other, but undertook the same 6-step evaluation method. Each step is described in turn:

1. The application was analyzed using UML use case diagrams to generate use case specifications of the required behaviour of the application;
2. One or more use case specifications for use cases described in the diagrams were entered directly by each analyst into UCaRE, and requirements were entered using the VOLERE requirements shell;
3. Each analyst used UCaRE functions to generate one or more service queries per use case specification defined in UCaRE during the second step;
4. Each analyst used EDDiE to retrieve services from SeCSE service registries using the service queries generated in the previous step. KD Software had specified and published one web service, called *KDTicketDataDelivery2a*, in the service registries which the analyst aimed to discover in the evaluation. CA had not published any service specifications, hence the evaluation investigated whether an uncontrolled set of available services could be used to support CA's requirements process;
5. Each analyst used the Service Browser to understand the service specifications retrieved from the registries and select those relevant to the generated queries;
6. Each analyst experimented with changes to use case and requirements specifications in light of the discovered services, documenting changes in use case and requirements specified in UCaRE. Each analyst could then repeat steps 3-6 again until the evaluation was complete.

Each analyst undertook all 6 steps. The 6 steps provide reference steps during the descriptions of the 2 evaluations reported in the next 2 sections.

5 Results from the CA Evaluation

The CA travel cost estimation application was specified to support its consultants who travel to client sites then bill their time and expenses. The use case diagram for the application contained 15 use cases and 3 actors. The primary actor in the model was the consulting manager, who seeks to achieve goals such as create draft description of work, establish price, agree project terms and conditions, search for consultants, and review projects.

Several of the use cases in the diagram were expanded into use case specifications entered into UCaRE. One such use case specification, *Estimate expense*, is shown in Figure 5. The problem statement outlined the existing problem. The précis described the required behaviour using unstructured text. The author also specified 4

Use Case ID	Estimate Expense
Actors	Consultant search, Consulting Manager
Problem statement	As part of the Consultant search function the Consulting manager needs to Estimate expenses. For this to be as accurate as possible the consulting manager has to get details of distance, travel mode and accommodation requirements costs. All of these costs.
Precis	When the consulting manager is matching consultant to company he needs to establish accurate costs to enable him to negotiate and agree a job price with the customer. Estimate expense works on distance of skilled consultant from the customer, what mode of travel is being used, what the contract says about travel expenses and what accommodation on site is required.
Functional Requirements	Response time should be < 1 minute.
Non-Functional Requirements	All costs to be calculated in US Dollars but values to be displayed in local currency based on that days interbank rate. Must be available during local office hours. Travel type must be shortest time of travel for consultant unless that gives a disproportionate cost increase.
Added Value	Currently done manually with online mapping and some rule of thumb measures.
Justification	Accurate estimate generate confidence and correct pricing of jobs
Triggering event	Sales notification of a contract negotiation that includes consultancy costs.
Preconditions	Draft description of work is available. DOW has been reviewed by the customer and agreed in principle.
Assumptions	Consultants are appropriately skilled for the work. Choice of travel mode is at the consultants discretion, discounted travel is not appropriate for the work.
Normal Course	1. Draft description of work review complete.
	2. Distance between consultant and customer is calculated.
	3. Cost of daily travel is estimated.
	4. Mode of transport is decided.
	5. Accommodation is accepted or rejected.
	6. Total cost calculated.

Fig. 5. The *Estimate expense* use case specification from the CA application

requirements – 1 functional and 3 non-functional – on the behaviour specified in the use case. A post-study review revealed that the specified functional requirement should be a performance requirement and at least one of the non-functional requirements can be interpreted as a functional one, so service discovery took place using incorrectly typed requirements. The use case normal course was composed of 6 actions that describe the required behaviour of the new expense estimating application.

During step 2 of the process the CA analyst commented that UCaRE was robust but necessitated the user guide to specify use cases. He also commented that there was no discernible difference in tool performance between access from the server site and remotely at CA offices, but remote access suffered from some Internet lag.

During step 3 the CA analyst successfully generated service requests and queries from the use case specification. The service query retrieved 14 candidate service specifications from the registries containing the 154 service specifications. Table 1 lists the names of the retrieved services in match order and the CA analyst's decisions to retain or reject each service using the Service Browser module.

During step 5 of the process the CA analyst retained 6 and rejected 7 of the 14 retrieved services to invoke in the future travel cost estimation application. An 8th was also rejected but identified as potentially useful to a related CA application. Over half of the services retrieved by EDDiE were deemed to be incorrect by the analyst for the specified service query.

Table 1. Ranked services retrieved by EDDiE for the CA *Estimate expense* use case specification, and the decision to retain or reject each of the services

Discovered Service Name	Decision to retain or reject service
Weblogwebservice	- Rejected -
AdressMeister	+ Retained +
Webservice search	- Rejected -
XigniteCurrencies	+ Retained +
FoldCalc	- Rejected -
XgniteEdgar	Rejected, but could be useful in another application
QuoteAndMarketData	- Rejected -
ThirdPartyCallTLAB	- Rejected -
SendSMSTLAB	+ Retained +
Mobile7NavigationKit	+ Retained +
CreditCardVerifyer	- Rejected -
XnavigationCEFRIEL	+ Retained +
TimeServiceCEFRIEL	+ Retained +
EmailVerifier	- Rejected -
XigniteDataSet	- Rejected -

There was no relationship between the services retained and the EDDiE ranking of these services. Short descriptions of the 6 retained services and the reasons for retaining them are reported in Table 2.

Table 2. Retained services and the CA analyst's rationale for their retention as relevant to the CA application

Service Name	Service Description	Rationale for retention
AdressMeister	Address Meister is a web-service for postal address verification and correction. It provides current, high-quality address data and verification logic without the cost and complexity of maintaining the nation's address database in-house. The service can be used by e-businesses to verify the addresses provided to them on their websites.	Verifying if the address is correct of both consultant and customer.
XigniteCurrencies	This web service provides real-time currency data (foreign exchange rate) for more than 170 currencies. Convert the US dollar amount to other currencies using real-time currency exchange rates returned by this currency web service	All internal currencies in USD, therefore currency conversion is required
SendSMSTLAB	This WS allows sending and monitoring SMS with a very simple interface. The Consumer can ask to send an SMS text to a list of addresses through the GSM network. After requiring SMS sending The Consumer can ask to the Provider to outline the delivery status of his request.	Communication of authorisation to consultant
Mobile7NavigationKit	ROUTE 66 Mobile 7 determines its position using an advanced high sensitive wireless GPS receiver, guiding the user with turn-by-turn voice instructions and on-screen directions to its destination. A new navigation display has been developed providing users with all vital travel information on a single clear screen of their smartphone including 3D map display, turn arrows and navigation guidance, as well as the ability to dial points of interest directly from the map.	Used to calculate distance between consultant and customer locations
XnavigationCEFRIEL	Especially useful for car drivers. You may want to know the duration in time of your trip, given the geographical positions of the departure and arrival places.	Required for estimating journey costs/time
TimeServiceCEFRIEL	This service computes the difference in time of two given moments.	Used in many places, for delivering time differences, e.g how long has the negotiation been progressing

The reasons given by the CA analyst demonstrate the usefulness of the retrieved services for estimating expenses. The *AdressMeister* service could be invoked to verify client addresses, the *XigniteCurrencies* service was needed to compute costs in a single currency, invoking the *SendSMSTLAB* service would deliver a means of communicating system outputs to the consultant, the *Mobile7NavigationKit* service would calculate distances between locations deemed pivotal to the application, the *XnavigationCEFRIEL* service could be invoked to compute the travel time – also pivotal to the application, and the *TimeServiceCEFRIEL* service would provide important timing data to the application.

During step 6 of the process the CA analyst used the specifications of the 6 retrieved services to generate 2 new requirements and improve a 3rd one in UCaRE. Table 3 lists the 2 new requirements and changed third one, along with the rationale for these requirements provided by the CA analyst.

Table 3. New and changed requirements generated from relevant feedback from the 6 retained services retained within the Service Browser

Type of Edit	Requirement Description	Rationale
New requirement	Web based access for the remote use by consulting manager	Consulting manager may have to authorise travel when travelling themselves
New requirement	The application must present 2 alternative methods of travel, Lowest cost and shortest time	Lowest cost requirement is not the only requirement, see changed requirement below
Changed requirement	Travel type modified so that the travel is short only if it is cost effective	Removed requirement for lowest cost travel.

After the evaluation we examined the granularity of the specified requirements and retrieved web services. Both new and original requirements were coarser grain than retrieved web services that implemented atomic functions that, if invoked, were insufficient on their own to satisfy a requirement. Therefore EDDiE was able to retrieve web services that were finer-grain than the requirements in the service queries, but this difference in granularity might have impacted on further requirements generation.

Using the revised use case and requirement specifications the CA analyst returned to step 3 of the process and generated a new service query. EDDiE returned the same 14 service specifications in the same order, although some of the MatchValues were slightly different to the MatchValues returned for the original service query. Because of this second result, the CA analyst concluded the evaluation. After the evaluation he reported that he was unable to use the Service Browser to review the service specifications and their similarities with requirement and use case attributes in the service query effectively, and this made service selection activities difficult.

6 Results from the KD Software Evaluation

During steps 1 and 2 the KD analyst used the SeCSE tools to specify 10 use cases and the associated requirements on the ticketing selection application, then discover services from registries (steps 3 & 4) that were then browsed, selected (step 5) and used to revise the use cases and requirements (step 6). One of these use case specifications is reported in Figure 6. The specification reveals evidence that the KD analyst had also attributed the wrong type to some of the requirements used to generate service queries.

Use Case ID	KD UC Ticket Searching
Actors	User (Unregistered User, Registered User, Administrator)
Problem statement	Providing events and ticket information search
Precis	All users can search events and tickets by use of search function. She can search event and ticket on base of event date or event name or event place or interpreter (band, sport team, etc.) name. She can specify also type of event. If search result is more then one, they are displayed like list. Number of list rows on page can be limited by application variable.
Functional Requirements	FR: Application UI must be standard internet browser. FR: Ticket data are provided by any Ticket data delivery service. FR: Ticket data can be searched in own or third party database. FR: Tickets or events can be searched according to several criteria (ticket database connection, event name, date, place, event type, etc.). FR: The search results must be in list form with possibility to limit size (rows number).
Non-Functional Requirements	NFR: Availability 99,9% NFR: Delay max.30 second
Added Value	Searching functionality is open, it can works with several ticket databases.
Justification	User needs information for ticket purchase.
Triggering event	User needs information for ticket purchase or for event attending planning.
Preconditions	Internet access, standard browser, ticket booking application started on web server, ticket database is accessible.
Assumptions	Internet browser software, internet access
Successful end states	List of events, tickets. Message "No suitable event or ticket found".
Unsuccessful end states	No internet access. Ticket database is not available. Too many users work with database (database is busy).
Normal Course	User chooses Ticket Search option FR1: Application must have this option (button or link).
	User chooses Ticket Searching criteria (ticket database connection (it should be in parameters data), date, event name, maximal ticket price, event place, event type, max.returned results number, etc.) FR2: Application must have possibility to fill search criteria.
	System returns list of events or list of tickets for events. FR3: Application must represent returned data in list or grid format. When user click on rows in list, details are displayed. FR4: For registered and logged user must be possibility to book or purchase tickets directly from list.
Variations	If [no connection to internet] then [application cannot run (browser accessibility message)] (related to Action 1). If [Registered and Logged User] then [User can continue to book or purchase tickets (those options are visible)] (related to Action 3).
Alternatives	If [no ticket database is available] then [application returns message about data availability] If [exotic browser] then [application returns standard info message window and recommends to change browser] If [no internet access] then [application returns standard info message window] If [no ticket data fits to criteria] then [application returns message about and recommends to change search criteria]

Fig. 6. One example KD Software use case specification – specifying the use case *Ticket searching*

The main difference between the KD and CA evaluations was that KD Software had earlier published one service specification in the registries, called *KDTicketDataDelivery2a,* which could be invoked in the ticketing application. The short description of the service was:

> *Service provides ticket data delivery in several formats (list, one concrete item, etc.). Service also has operation for data searching.*

The KD analyst generated 2 service queries from 2 of the 10 use case specifications entered into UCaRE. Both were composed of text extracted from the use case name,

Table 4. Top 10 service specifications retrieved for service queries generated from 2 KD Software application use case specifications

Rank	Ticket Browsing Use Case	Ticket Searching Use Case
1	**KDTicketDataDelivery2a**	ViaMichelinFindNearByPOIwebservice
2	AGENDAMSD	**KDTicketDataDelivery2a**
3	ViaMichelinFindNearByPOIwebservice	Weblogwebservice
4	ImageCutOut	WebServiceSearch
5	Weblogwebservice	XgniteEdgar
6	XgniteEdgar	Mobile7NavigationKit
7	EmailVerifier	FoldCalc
8	AmazonHistoricalPricingService	ImageCutOut
9	Anagram	KDfindCarService1
10	CalendarServiceEMIC	ABAExpress

actors, précis, problem statement, assumptions, preconditions and triggering event, and 5 or 6 requirements of different types. Both were specified to search using the noun, verb, adverb and adjective parts-of-speech, but no query expansion was requested because the KD analyst explained in debriefing sessions that he did not understand the meanings of the terms *synonyms*, *hyponyms* and *glosses* in the service query.

The top 10 service specifications retrieved for the 2 service queries are reported in Table 4, ranked by MatchValues computed by EDDiE.

The target service specification - *KDTicketDataDelivery2a* - was ranked in the top two by EDDiE for both service queries. In the first query EDDiE retrieved it with full MatchValue (100). For the second use case that specified ticket searching, EDDiE also retrieved the *ViaMichelinFindNearByPOIwebservice* service with full match value that, according to the KD analyst, had no relation with the generated query. The description of this service, also taken from the SeCSE service registries, was:

the "FindNearbyPOI" Web Service allows searching for a certain number of addresses or locations closest 'as the crow flies' to a particular address or place of interest within a user-definable search radius. Then displaying any detailed poi information is possible. For example, it can look for car dealers closest to a given location or find competitors that are closest to your sales points and, as a result, analyze catchment areas that are the least well served.

The analyst's decision not to select query expansion types in the service query prevented EDDiE from generating additional query terms with the same or similar meaning to original query terms. Therefore EDDiE matched only identical or very similar terms such as *search* and *display*. We conjecture that the match values of the retrieved services including *ViaMichelinFindNearByPOIwebservice* would have been different if query expansion was enabled. This result demonstrated that, for ambiguous and incomplete queries, EDDIE generated false positives during service discovery alongside true positive discovered services.

During step 6 the KD analyst attempted to edit the use case and requirements specifications using information in retrieved services. However the changes made by the KD analyst were simple and did not lead to significant changes to new service queries or retrieved services. At this point the KD analyst ended the evaluation.

7 Research Questions Revisited

We used data from the 2 evaluations to answer the 2 research questions. The answer to Q1 – can SeCSE tools retrieve specifications of services that can implement requirements specified by analysts in service queries – was yes. In the KD Software evaluation EDDiE retrieved the target service specification with rank 1 and 2 of 154 with 2 service queries. This suggests high precision of the EDDiE algorithm in the presence of ambiguous and incomplete requirements in a real-world project. Failure to use query expansion increased the relative weighting of syntactic similarities during service discovery and, in the second query, returned one high-ranked but irrelevant service specifications. In the CA evaluation, for which there were no target service specifications, the analyst retained 6 of the 14 retrieved services to invoke in the application. This suggests that the process and environment has the potential to support applications when sufficient numbers of application-independent services are published. That said, our decision to evaluate the utility of the tools, rather than determine their precision and recall, means that we do not know whether the algorithm failed to retrieve other service specifications that might also have been retained by the analyst.

We investigated post-retrieval changes to the use case and requirement specifications to answer Q2 – can analysts using the SeCSE tools make requirement specifications more complete. There was little evidence to answer yes. The CA evaluation added 2 requirements to and changed 1 of the original 6 requirements, and there were no changes in the KD evaluation. Post-evaluation questions revealed that both analysts encountered difficulties browsing retrieved services due to complexities in the service descriptions and similarities with the requirements in the service queries. Poor expression of non-functional requirements meant that filtering services on quality-of-service compliance could not be used, and the Service Browser provided little support to each analyst to understand services. Furthermore UCaRE did not provide support for service querying that was sensitive to the recent changes to requirement and use case specifications. Because service-based changes were small in the context of the use cases specifications, the revised queries did not retrieve new services.

Of course there are numerous threats to the validity of the reported results, and important ones are reported here. The obvious threat to the validity of conclusions drawn was the small number of studies and both were participants in the SeCSE project. To minimize this threat, follow-on studies with more analysts from organizations external to the project are now taking place, and we will interpret results reported in this paper in light of the results from these future evaluations. One threat to the internal validity of the evaluations was that the application requirements and registry services were (unintentionally) aligned too well – we might not expect such alignment in public, market-oriented registries. However, results from the CA evaluation do not support this threat. The domain-independent nature of the services – for verifying addresses, calculating journey times and computing currency exchanges – made them candidates for invocation, and the SeCSE software tools retrieved and presented these services effectively enough to enable the analyst to retain them. A threat to the evaluation's construct validity also merits a mention here. Because the analysts were SeCSE partners with a vested interest in the outcome we cannot discount that they were biased to generate positive results. And one threat to the external validity of the results was our decision to align SeCSE's requirements process and service discovery tools with

UML. Whilst we chose UML for its ubiquity in software development, it does mean that the evaluation results might be less applicable to projects that adopt workflow and business process approaches to requirements analysis.

8 Discussion and Future Work

Answers to the 2 research questions investigated in this paper indicate future directions of research and evaluation in SeCSE. One is the need for precision-and-recall experiments of the EDDiE algorithm and FrEDDiE, a new software module in the environment that decomposes service queries to increase the likelihood of successful retrieval with coarse-grain use cases and requirements. Controlled variables in these experiments will be predefined service query attributes such as use case précis and requirement descriptions and expansion types such as synonyms and glosses. Application experts will review retrieved services for their relevance to each query to generate precision and recall measures for different query attributes and expansion strategies. We are also extending EDDiE to retrieve other types of services, such as peer-to-peer (P2P) and grid services, thus leveraging new repositories of software services of different types on the Internet. In this extension EDDiE service queries are translated into the Universal Service Query Language [12] then fired at federations of P2P and grid service registries compliant with different standards applicable to these service types. Of course, retrieval of more candidate services from more sources amplifies 2 problems reported in the evaluations, which was how to understand and select between retrieved services, then revise requirements and service queries using relevance feedback from retained services.

The answers to the 2 research questions also provide empirical foundations for further development of the SeCSE service discovery environment, particularly in light of our answer to research question Q2. One priority is to improve the usability of the Service Browser module. To make it more usable we responded to post-review comments from the 2 analysts and developed an off-line version of the module in Microsoft Excel. Analysts can now download all data about service queries, specifications of retrieved services, match values and mappings between terms to interactive spreadsheets, to review the data off-line and manipulate it in other forms more supportive of comprehension and selection tasks.

The low number of requirements generated by both analysts when reviewing the retrieved web services contrasts with the higher number of requirements generated from services during facilitated workshops [4]. This difference indicates the need to improve tool support for analysts during this step. To this end we are developing one new software module and adding new features to a second to support service understanding and selection. The Service Browser module does not provide analysts with explicit support for generating or editing requirements in the UCaRE module. Instead the analyst is expected to flip between the 2 modules in a single web browser window, using problem analysis and requirements writing skills to document new or changed requirements in UCaRE. Therefore we designed a new software module to generate candidate new requirements descriptions from service specification text highlighted as relevant by the analyst. This new module will use mappings between terms computed by EDDiE in use case and requirement specification and retrieved service specifications to generate candidate descriptions of new requirements structured using requirements writing guidelines [13]. The analyst then selects between and edits the candidate requirements in UCaRE and links it to the service specification for traceability purposes. Of course, if successful, this requirements auto-generation module could be applied to other sources of requirement-related data such as software product descriptions.

Finally we are also adding a new feature to UcaRE to support the iterative and incremental SeCSE requirements process. In both evaluations the 2 analysts were unable to retrieve further service specifications because requirements changes from relevance feedback were small in the context of the original requirement specification. The new feature will allow an analyst to generate service queries that only include requirement and use case information that is new since the last service query(ies) were fired. We predict that the feature will enable focused searching and service retrieval, a prediction that we will investigate empirically in future user studies with the SeCSE service discovery environment.

Acknowledgements

SeCSE is funded by the EU 511680 Integrated Project.

References

1. Tetlow, P., Pan, J., Oberle, D., Wallace, E., Uschold, M., Kendall, E.: Ontology Driven Architectures and Potential Uses of the Semantic Web in Software Engineering. W3C (2005)
2. Margaria, T.: Service in the Eye of the Beholder. IEEE Computer 40(11), 33–37 (2007)
3. Zachos, K., Maiden, N.A.M., Zhu, X., Jones, S.: Discovering Web Services To Specify More Complete System Requirements. In: Krogstie, J., Opdahl, A., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, pp. 142–157. Springer, Heidelberg (2007)
4. Zachos, K., Maiden, N.A.M.: Discovering Services to Support Creative Thinking during Early Requirements Processes. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, vol. 4749. Springer, Heidelberg (2007)
5. Jones, S.V., Maiden, N.A.M., Zachos, K., Zhu, X.: How Service-Centric Systems Change the Requirements Process. In: Pastor, Ó., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 13–14. Springer, Heidelberg (2005)
6. Robertson, S., Robertson, J.: Mastering the Requirements Process. Addison-Wesley-Longman (1999)
7. Sawyer, P., Hutchinson, J., Walkerdine, J., Sommerville, I.: Faceted Service Specification. In: Proceedings SOCCER (Service-Oriented Computing: Consequences for Engineering Requirements) Workshop, at RE 2005 Conference, Paris (2005)
8. http://vega.soi.city.ac.uk/~cc559/REFSQ2008Figure2.jpg
9. Miller, K.: Introduction to WordNet: an On-line Lexical Database Distributed with WordNet software (1993)
10. http://vega.soi.city.ac.uk/~cc559/REFSQ2008Figure4.tiff
11. Deliverable A2.D10 - Evaluation of service discovery environments, v2.0, SeCSE Technical Report, available at secse.eng.it (2007)
12. SODIUM, Service-Oriented Development In a Unified fraMework, IST-FP6-004559 (2007), http://www.atc.gr/sodium
13. Alexander, I.F., Stevens, R.: Writing Better Requirements. Addison-Wesley, Reading (2002)