SERVICE COMPUTATION 2015

The Seventh International Conferences on Advanced Service Computing

ISBN: 978-1-61208-387-2

March 22 - 27, 2015

Nice, France

SERVICE COMPUTATION 2015 Editors

Marcelo de Barros, Microsoft Corporation, USA

Claus-Peter Rückemann, Leibniz Universität Hannover / Westfälische Wilhelms-Universität Münster / North-German Supercomputing Alliance (HLRN), Germany

SERVICE COMPUTATION 2015

Forward

The Seventh International Conferences on Advanced Service Computing (SERVICE COMPUTATION 2015), held between March 22-27, 2015 in Nice, France, continued a series of events targeting computation on different facets.

The ubiquity and pervasiveness of services, as well as their capability to be context-aware with (self-) adaptive capacities posse challenging tasks for services orchestration, integration, and integration. Some services might require energy optimization, some might require special QoS guarantee in a Web-environment, while others a certain level of trust. The advent of Web Services raised the issues of self-announcement, dynamic service composition, and third party recommenders. Society and business services rely more and more on a combination of ubiquitous and pervasive services under certain constraints and with particular environmental limitations that require dynamic computation of feasibility, deployment and exploitation.

The conference had the following tracks:

- Web services
- Empirical methods in system and service management
- Service innovation, evaluation and delivery

Similar to the previous edition, this event attracted excellent contributions and active participation from all over the world. We were very pleased to receive top quality contributions.

We take here the opportunity to warmly thank all the members of the SERVICE COMPUTATION 2015 technical program committee, as well as the numerous reviewers. The creation of such a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors that dedicated much of their time and effort to contribute to SERVICE COMPUTATION 2015. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations and sponsors. We also gratefully thank the members of the SERVICE COMPUTATION 2015 organizing committee for their help in handling the logistics and for their work that made this professional meeting a success.

We hope SERVICE COMPUTATION 2015 was a successful international forum for the exchange of ideas and results between academia and industry and to promote further progress in the

area of computation. We also hope that Nice, France provided a pleasant environment during the conference and everyone saved some time to enjoy the charm of the city.

SERVICE COMPUTATION 2015 Chairs

SERVICE COMPUTATION 2015 Advisory Chairs

Mihhail Matskin, KTH, Sweden
Hideyasu Sasaki, Ritsumeikan University - Kyoto, Japan
Bernhard Hollunder, Hochschule Furtwangen University – Furtwangen, Germany
Paul Humphreys, Ulster Business School/University of Ulster, UK
Arne Koschel, Hochschule Hannover, Germany
Michele Ruta, Politecnico di Bari, Italy
Alfred Zimmermann, Reutlingen University, German
Aida Omerovic, SINTEF, Norway
Martin Wynn, University of Gloucestershire, UK
Annett Laube, Bern University of Applied Sciences (BUAS), Switzerland
Claus Pahl, Dublin City University, Ireland

SERVICE COMPUTATION 2015 Industry/Research Chairs

Ali Beklen, CloudArena, Turkey
Steffen Fries, Siemens Corporate Technology - Munich, Germany
Emmanuel Bertin, Orange Labs, France
Matthias Olzmann, noventum consulting GmbH - Münster, Germany
Sergey Boldyrev, HERE Berlin, Germany
Rong N. Chang, IBM T.J. Watson Research Center, USA
Wasif Gilani, SAP Research, UK
Alexander Kipp, Robert Bosch GmbH, Germany
Marcello Coppola, ST Microelectronics - Grenoble, France
Jan Porekar, SETCCE, Slovenia

SERVICE COMPUTATION 2015

Committee

SERVICE COMPUTATION Advisory Chairs

Mihhail Matskin, KTH, Sweden
Hideyasu Sasaki, Ritsumeikan University - Kyoto, Japan
Bernhard Hollunder, Hochschule Furtwangen University – Furtwangen, Germany
Paul Humphreys, Ulster Business School/University of Ulster, UK
Arne Koschel, Hochschule Hannover, Germany
Michele Ruta, Politecnico di Bari, Italy
Alfred Zimmermann, Reutlingen University, German
Aida Omerovic, SINTEF, Norway
Martin Wynn, University of Gloucestershire, UK
Annett Laube, Bern University of Applied Sciences (BUAS), Switzerland
Claus Pahl, Dublin City University, Ireland

SERVICE COMPUTATION 2015 Industry/Research Chairs

Ali Beklen, CloudArena, Turkey
Steffen Fries, Siemens Corporate Technology - Munich, Germany
Emmanuel Bertin, Orange Labs, France
Matthias Olzmann, noventum consulting GmbH - Münster, Germany
Sergey Boldyrev, HERE Berlin, Germany
Rong N. Chang, IBM T.J. Watson Research Center, USA
Wasif Gilani, SAP Research, UK
Alexander Kipp, Robert Bosch GmbH, Germany
Marcello Coppola, ST Microelectronics - Grenoble, France
Jan Porekar, SETCCE, Slovenia

SERVICE COMPUTATION 2015 Technical Program Committee

Witold Abramowicz ,Poznan University of Economics, Poland
Saeed Aghaee, University of Lugano, Switzerland
Antonia Albani, University of St. Gallen, Switzerland
Riyad Alshammari, KSAU-HS University, Saudi Arabia
Dimosthenis S. Anagnostopoulos, Harokopio University of Athens, Greece
Julian Andrés Zúñiga, University of Cauca, Colombia
Ismailcem Budak Arpinar, University of Georgia, USA
Johnnes Arreymbi, School of Architecture, Computing and Engineering - University of East London, UK

Arne Koschel, Hochschule Hannover, Germany
Yousri Kouki, ASCOLA - INRIA, France
Natalia Kryvinska, University of Vienna, Austria
Patricia Lago, VU University Amsterdam, Netherlands
Ulrich Lampe, Technische Universität Darmstadt, Germany
Annett Laube-Rosenpflanzer, Bern University of Applied Sciences - Biel/Bienne, Switzerland
Guanling Lee, National Dong Hwa University, Taiwan
Keqin Li, SAP Product Security Research, France
Kuan-Ching Li, Providence University, Taiwan
Noura Limam, University of Waterloo, Canada
Cho-Chin Lin, National Ilan University,Taiwan
Damon Shing-Min Liu, National Chung Cheng University, Taiwan
Qing Liu, The Commonwealth Scientific and Industrial Research Organisation (CSIRO), Australia
Shih-His (Alex) Liu, California State University - Fresno, USA
Welf Löwe, Linnaeus University, Sweden
Hui Ma, Victoria University of Wellington, New Zealand
Khaled Mahbub, City University, UK
Sabri A. Mahmoud, King Fahd University of Petroleum and Minerals, Saudi Arabia
Kurt Maly, Old Dominion University, USA
Lefteris Mamatas, University College London, UK
Gregorio Martinez, University of Murcia, Spain
Mihhail Matskin, KTH, Sweden
Manuel Mazzara, Innopolis University, Russia / ETH Zurich, Switzerland
Viktor Medvedev, Vilnius University, Lithuania
Souham Meshoul, University Constantine 2, Algeria
Lars Mönch, FernUniversität in Hagen, Germany
Fabrizio Montesi, IT University of Copenhagen, Denmark
Fernando Moreira, Universidade Portucalense, Portugal
Debajyoti Mukhopadhyay, Maharashtra Institute of Technology, India
José Neuman De Souza, Federal University of Ceará, Brazil
Francisco Javier Nieto De-Santos, Atos Research and Innovation - Bilbao Spain
Artur Niewiadomski, Institute of Computer Science - Siedlce University of Natural Sciences and Humanities, Poland
Mara Nikolaidou, Harokopio University of Athens, Greece
Roy Oberhauser, Aalen University, Germany
Matthias Olzmann, noventum consulting GmbH - Münster, Germany
Hichem Omrani, CEPS/INSTEAD - GEODE dept., Luxembourg
Claus Pahl, Dublin City University, Ireland
Ingo Pansa, iC Consult, Germany
Namje Park, Jeju National University, Korea
Petra Perner, Institute of Computer Vision and applied Computer Sciences, Germany
Dana Petcu, West University of Timisoara, Romania
Willy Picard, Poznan University of Economics, Poland
J Brian Pickering, IT Innovation Centre, UK

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

An Application of Stochastic Models To Monitoring of Dynamic Web Services

Marcelo De Barros, Manish Mittal
Bing Customer Experiences Engineering
Microsoft Corporation
Redmond, USA
marcelod@microsoft.com, manishm@microsoft.com

Abstract - Web search engines are very dynamic in nature; not only are the backend and data powering the site evolving, but the frontend is always adapting to different browsers, devices and form-factors, and experiments are often running in production. In fact, when it comes to User Experience (UX), it is likely that users are always falling into some live experiment in production: variation of colors, fonts, typography, different Java Scripts and so on. Issues (software bugs) can occur on the live site for very particular contexts, where a context is defined as a particular configuration of browser, market and experiment. As an example, a JavaScript error can occur on a certain page, for certain types of queries, against a certain market on a particular browser. The problem that we're trying to solve is to devise a probabilistic methodology to monitor and detect these particular software bugs in production environments by maximizing the chances of detecting the most relevant issues from the application users' standpoint. For this purpose, we at the Microsoft Bing Experiences Team developed a concept of synthetic exploratory monitoring that can focus on the important features on the sites and pages, and use invariants (conditions that should always hold true, or always hold false, for specific contexts), such as security-related invariants, to detect potential anomalies in the current context. We make use of stochastic models to ensure maximum relevant coverage of contexts and devices. We use the power of the Selenium testing framework to drive end-to-end automation on browsers and devices, the notion of exploratory tests, and a set of heuristics and invariants (text-based and image-based) that can auto-detect problems on the live site in very particular contexts. We compare and contrast two machine–learning models: Markov Chains and Time-Based Artificial Neural Networks (ANNs). We implemented the idea explained in this paper to monitor large-scale web sites such as Bing Search Engine where alerts are generated automatically whenever the anomaly conditions are detected. The solution is easily expandable to other sites. We envision, as future work, moving this technology to the cloud that would allow easy customization of all parameters (browsers used, definition of the finite-state machine, heuristics and invariants). This paper explains the fundamental principles to create a stochastic monitoring model and demonstrates how to apply the principles to large-scale web sites and services. We will utilize Bing Search Engine to illustrate the techniques explained here.

Keywords-software testing; large-scale services; quality of services; markov chains; artificial neural networks; selenium; testing in production; monitoring; stochastic models

I. SCALING SYSTEMS TO DEVICES, BROWSERS AND MARKETS

In today's world, whenever a new online system is launched, it is usually available across several devices (devices that display web contents), browsers and markets instantaneously and simultaneously. This poses a significant development challenge since:

a) Different browsers, devices and markets have specific requirements and resources that may differ from each other, and there is no enforced global standard across them.
b) Support for Cascading Style Sheet (CSS) and HTML5 compatibility and support vary significantly from browser to browser.
c) The form-factor for the different devices varies significantly. Because of smaller screens, code might need to be optimized to show the user different data or presentation of the information. Despite the development and adoption of responsive web design techniques [10], very often there is still a need for small code customizations. For instance, some devices are large enough to display data into two vertical panes (columns), where others require the use of a single pane.
d) Markets are also another important dimension given the differences in language grammars as well as geo-cultural differences. Large-scale systems such as Google, Bing and Facebook are always dealing with such challenges.

Many large-scale web sites are now making use of "flights" or "experiments". An experiment is a way to expose a percentage of the site's users to a different treatment of the site (which can be differences in the User Interface, middle-tier, backend or even data differences) in order to collect early feedback and then make an informed decision about the upcoming features for the system. For example, a search engine might want to expose 2% of its users to a SEarch Results Page (or SERP) that shows only eight "blue links" by default instead of ten blue links. The telemetry for that experiment is then collected and analyzed against the "control" (the ten blue links) and data analysts work on distilling the positive and negative aspects of the experiment, where positive aspects correspond to user metrics moving towards the expected direction (such as page load time being reduced, user abandonment reduced,

 ISBN: 978-1-61208-387-2

increased dwell time [16], amongst others) and negative correspond to the converse. Experiments can overlap with each other. At any point in time, there might be tens or even hundreds of experiments running in production environments [15].

The paper is organized as follows: in Section I, we describe the complexities involved with monitoring large-scale services. In Section II, we describe the current state of the art. In Sections III and IV, we introduce the ideas of Markov Chains and Selenium, respectively. In Sections V, we define the concept of exploratory runs. In Section VI, we define the concept of subscription-based validation methods. In Section VII, we devise the strategy for exploratory runs utilizing a stochastic model (such as Markov Chains), Selenium and the pre-defined concept of subscription-based validation methods to solve the monitoring problem. In Section VIII, we explore other stochastic models that can be used to solve the monitoring problem, such as Artificial Neural Networks. In Section IX, we provide a summary of the work as well as the direction for future research.

II. MONITORING COMPLEXITIES AND STATE OF THE ART

Since the code is somewhat customized to different user experiences (experiments, browsers, devices and markets), there is a possibility of encountering specific issues on any of these and worst, on combination of these dimensions: a specific problem may only happen on an experiment, on a given browser, on a given device and for a particular market. Some simple lower-bound calculations show the complexity and the scale of this problem. If we have around 30 experiments, 30 browsers, 30 devices and 200 markets, the number of possible combinations (assuming no overlaps on the experiments) becomes 30*30*30*200 = 5,400,000 different permutations. Even using well known monitoring techniques, such as Gomez [2] or Keynote [3], it becomes impossible to monitor all these variations. In reality, though, most of these contexts are either not significantly crucial to the business or are not valid at all (for example, most of the time experiments are limited to either a group of markets or a group of browsers), hence understanding the valid and important permutations can prune the combinatorial space considerably. Notice the usage of the terms "testing" and "monitoring" are interchangeable in this paper, both indicating the ability to proactively detect anomalies in real production environments.

The current state of the art for monitoring strategies consists basically of three approaches:

a) Synthetic Transactions [2]: market tools such as Keynote and Gomez give the capability of building custom, synthetic transactions to monitor particular features of web services and sites. However, synthetic transactions only target very limited set of features that need to be known a-priori which limits its effectviness when monitoring complex and dynamic systems. They are very inefective for highly dynamic services.

b) Performance Counters [14]*:* web services developers have the ability to implement performance counters on the server side which can give indications of potential system malfunctions. For example, a performance counter that tracks "75%tile server side latency" can be the initial lead to investigate real user issues with the service (in case of spikes or drops, for example). However, performance counters have the disadvantages that they usually fail to track client-only problems (such as javascript errors) and since they aggregate data across all the users, it only detects problems when a signficant number of users are affected by the problem – issues that affect only a very small percentage of users usually go undetected by performance counters.

c) Telemetry: telemetry consists of analysis of time series of logs from user activity as well as system probes in order to detect anomalies in production [12]. Although telemetry analysis has the capability of detecting widespread issues with one's service, it is not a real-time monitoring system since the collection, aggregation and availability of the data are tasks that usually take signficant time to be performed, limiting its ability in detecting anomalies in a timely manner.

None of the current state of the art methoodologies hence is comprehensive enough to actually model the user's behavior and detect in real-time relevant anomalies based on the true users' patterns observed in production environments. The work described in this paper is an attempt to address this gap.

III. MARKOV CHAINS

We use Markov Chains [4] to model the behavior of the system, limiting the monitoring space to the most probable paths. A Markov Chain is a type of stochastic model based on the concept of Finite-State Machines [17]) that undergoes transitions from one state to another on a state space. It is a random process usually characterized as memory-less: the next state depends only on the current state and not on the sequence of events that preceded it. For search engines, the states in a Markov Chain are web site landing pages, such as: the search engine Home Page, the search engine Web Results Page, Videos Results Page, Images Results Page, Settings Page, and any the other page type included in the search engine substrate.

The actions that lead to a state transition are the different actions that can be performed by the end user, mainly Searches, Clicks, Tabs, Hovers, and so on. With enough anonymous log-based information about the different states and actions, one can build a comprehensive Markov Chain diagram modelling the proper behavior of the average user of the web system in questions. The assumption is that most web sites nowadays log information about their users' iterations with the page (in an anonymized manner). The picture below (Figure 1) gives an example of a state transition, and the table below (Table I) gives an example of a simple Markov Chain. Notice that the key aspect here is

 ISBN: 978-1-61208-387-2

that each action is associated with a certain probability (the "Probability Weightings" column in Table I), calculated based on the number (percentage) of users who triggered that respective action based on captured data. For example, the second row in Table I shows the state as being the "home page" and the probability weighting as being "20%". The semantics of such information is that when a user lands (or is) in the "Home Page" state, there is a probability of 20% that the user will perform the action of "typing a query" and hitting enter. The third row tells us that if the user is in the same state ("Home Page") there is also a probability of 15% that the user will perform the action of clicking on the "Top News" link. The table only shows a partial view of the probability weighting distribution.

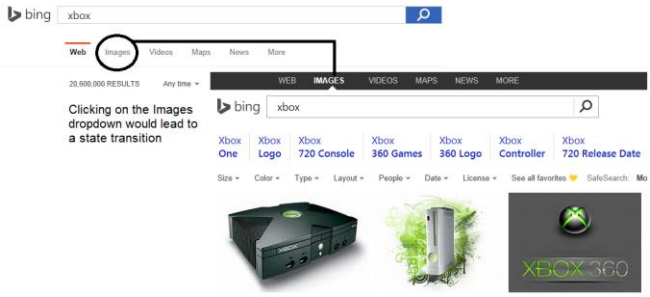


Figure 1. Example of state transitions

TABLE I. EXAMPLE OF MARKOV CHAINS STATES AND WEIGHTED TRANSITIONS

Initial State	Final State	Action	Probability Weightings
Home Page	Search Results Page	Typed Query	20%
Home Page	Search Results Page	Clicked on "Top News"	15%
Home Page	Home Page	Refresh the Page	5%
Search Results Page	Search Results Page	Typed Query	60%
Search Results Page	Non-Search Page	Clicked on Ads	15%
Search Results Page	Non-Search Page	Clicked on Algo Result	33%
Images Results Page	Videos Results Page	Click on the Videos link	10%
Images Results Page	Images Results Page	Click on Related Images	25%
Images Results Page	Images Results Page	Click on Related Entities	7%
Images Results Page	Images Results Page	Refresh the Page	13%

The granularity of the states and the actions is something that varies depending on the applications. In the example above, the Typed Query action could certainly be further refined by specifying the category/class of query being typed, such as "Local Query", "Adult Query" or "Electronics". Likewise the "clicked on" event can be grouped into categories (such as "clicked on Algo Results") or further refined down to the domain of the link being clicked (such as "clicked on an amazon.com link"). The important aspect is to create the chain in such a way that it truly encompasses the users' behavior but keeping it concise enough to prune the overall search space. For our project, we also added some random aspects to our testing in order to provide extra coverage. For example, when the action is "Send a new query" we take a random query from a pool of pre-defined queries, usually a combination of head and tail queries (See "Web Search Queries" [9]).

Some states are outside the scope of the pages being tested. For example, if the scope being tested is all the pages under the bing.com domain, any site outside that domain would be considered an out-of-scope state. It is important to model the chain in such a way that once out of the scope, actions will lead to in-scope states (such as clicking the back button, or navigating back to the initial state).

With the Markov Chain created, the monitoring approach can be tweaked to randomly follow the paths and probabilities specified by the chain. Notice that the approach will necessarily focus on the most probable paths (assuming a random distribution), which is the desired approach.

In addition to using a Markov Chain for transitions, another important aspect that needs to be taken into consideration is the overall distribution of browsers, devices, markets and flights (experiments).

There are two different approaches to integrating Markov Chains for these additional dimensions into the monitoring system:

1) Create the Markov Chain to take into account browsers, devices, markets and flights (experiments). In such cases, there can be multiple Markov Chains for each dimension, or combination of dimensions, or one Chain where states and transitions take into account these dimensions; or, alternatively
2) Create the Markov Chain without the particular data about browsers, devices, markets and flights, and use an orthogonal table with the distribution of the population across these dimensions, and randomly switch to a certain dimension as you navigate the chain.

The approach we have taken is the second one. The Markov Chain is created with the overall usage pattern across all the users in the system. At the same time we get the distribution of users across all browsers, devices, markets and experiments. In the following hypothetical example (Table II), we see several user context distributions across browsers, devices, markets and experiments. We then combine these two sources of data (the Markov Chain and the User Context Distributions) in order to come up with the proper stochastic model for the exploratory tests. Section VII explains the details of how these two data sources come together. Section VIII explains how the User Context

 ISBN: 978-1-61208-387-2

Distribution can be used as input into an Artificial Neural Network.

TABLE II. EXAMPLE OF USER CONTEXT DISTRIBUTIONS

Browser	Percentage of users
Internet Explorer 7	6%
Internet Explorer 8	8%
Internet Explorer 11	15%
Firefox	9%
Others	*62%*

Device	Percentage of users
Windows Phone	34%
iPhone	17%
Kindle Fire	17%
Android	9%
Others	*23%*

Market	Percentage of users
United States	52%
China	17%
Brazil	4.5%
Canada	7%
Others	*19.5%*

Experiment	Percentage of users
Experiment #1: light-blue background color	2%
Experiment #2: larger font size for titles	3%
Experiment #3: larger images	20%
Experiment #4: new relevance ranker	1%

A potential limitation of the Markov Chains is the fact that transitions from one state to the other do not depend on the path taken to get to the current state. This might be seem as a limitation of the model if there is a need to build more complex, "state-full" scenarios. That can easily be overcome by developing more detailed states inside the Markov Chain (adding complexity to it). For example, if there is a need to model a scenario where users come from page A through page B, we can build a state named "AB" that reflects that path.

IV. SELENIUM

Selenium [6] is a portable software testing framework for web applications that provides a record/playback tool for authoring tests without learning a test scripting language (Selenium IDE). It also provides a test domain-specific language (Selenese) to write tests in a number of popular programming languages, including Java, C#, Groovy, Perl, PHP, Python and Ruby. The tests can then be run against most modern web browsers. Selenium deploys on Windows, Linux, and Macintosh platforms. The way we use Selenium for exploratory tests and monitoring is through Selenium WebDrivers. Selenium WebDriver accepts commands and sends them to a browser. This is implemented through a browser-specific browser driver, which sends commands to a browser, and retrieves results. Most browser drivers actually launch and access a browser application (such as Firefox or Internet Explorer). Selenium WebDriver does not need a special server to execute tests. Instead, the WebDriver directly starts a browser instance and controls it. There is an ongoing effort by the inventors of Selenium to make it an internet standard.

Selenium provides an easy interface to interact with the browser, and the same test scripts can be used against many supported browsers. The ability to perform clicks, hovers, navigation manipulation, simulate different keyboard commands to the browser, scroll, change the browser settings and even detect and manipulate pop-up windows make it ideal for web automation.

In order to provide extra reliability, one can make use of a Selenium Grid. Selenium Grid is a server that allows tests to use web browser instances running on remote machines. With Selenium Grid, one server acts as the hub. Tests contact the hub to obtain access to browser instances. The hub has a list of servers that provides access to browser instances (WebDriver nodes), and lets tests use these instances. Selenium Grid allows running tests in parallel on multiple machines, and to manage different browser versions and browser configurations centrally (instead of in each individual test).

V. EXPLORATORY RUNS

The term Exploratory Runs here is loosely used to define the process of semi-randomly exploring different parts of a system while performing different verifications and validations that are pertinent to the current part of the system in question. The semi-random nature is accomplished via two methods: walking the generated Markov Chain, and modifying the context based on the users' distribution of markets, browsers, devices and experiments. The process usually starts at the initial page of the system, such as the user's home page. At that point a frequency-weighted random set of actions gets triggered based on the weight (probability) of the actions in the Markov Chain. It continues from that point on following the same approach indefinitely or until a certain time amount elapses. The transition of the states is implemented via commands in Selenium. Figure 2 below illustrates a simple Markov Chain being walked probabilistically:

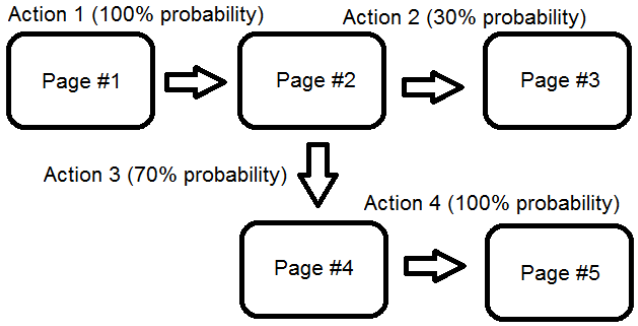


Figure 2. Schema depicting a simple Markov Chain

Orthogonally to the walk of the Markov Chain, we make use of the contexts distribution in the following manner:

 ISBN: 978-1-61208-387-2

a) Markov Chain traversal keeps happening randomly for a period of time (say N minutes)
b) After that period of time elapses, a change of context happens based on the distribution table

We use N = 30 minutes, which is based on our observations with real Bing user data, 30 minutes is the average time for a user web session. After 30 minutes, the contexts in which the tool is running may change: browser, device, market or experiment. The change is random but weighted based on the distribution tables. We utilize a number of Selenium Grids, one for each type of Internet Explorer (IE) browsers (from version 7 to the latest version), and all the grids also contain other browsers, such as Chrome and Firefox. Markets also change based on a set of pre-defined markets (around 200 in our case). The device is simulated on the desktop browsers by manipulating the user-agent. This simulation isn't ideal as some issues only appear or repro on the actual devices, but it is a good stopgap solution to catch some types of issues (like features being under/over triggered). We also force the exploratory run to fall into one (or combination of) experiments by using test hooks (in our case query-string parameters that are only enabled/visible inside the Microsoft corporate network). The automation keeps running indefinitely as a monitoring mechanism against production.

VI. SUBSCRIPTION-BASED VALIDATION MODULES

It is common to see the schema of a validation module (or test case) as a self-contained unit that performs all the steps necessary to set up the proper pre-validation before the validation takes place, followed by the validation itself, culminating with the post-validation (or teardown). Schematically we have:

```
SampleTestCase()
{
  Pre-ValidationSetup();
  Validation();
  Post-ValidationSetup(); //Teardown
}
```

There are many advantages of such scheme: simplicity, standard pattern, readability, reproducibility, determinism, to name a few. However, such a model does not fit well into the exploratory runs mentioned previously. Instead, what we want is a subscription-based model where the test case subscribes to the current state (or action) if the current state (or action) meets certain criteria pertinent to that test. Schematically, subscription-based test cases have the following format:

```
SubscriptionBasedSampleTestCase()
{
  If(IsRelevantState(this.CurrentState))
    Validation();
}
```

In essence, we are proposing a separation of the validation method from the configuration. The test becomes opportunistic rather than deterministic: if we reach a situation during the traversal of the Markov Chain where the test is applicable, then it runs; otherwise it ignores the current state.

An example of a subscription-based test case would be the following: suppose that we want to write a test case to validate behaviors for a certain segment of queries called navigational queries, which are queries that seek a single website or web page of a single entity. A query such as "sales force" is a navigational query. There are several types of validation that can be performed for navigational queries. As per the example in Figure 3, when searching on "sales force", we can base validation on:

a) Correctness of the algorithmic first result returned
b) Proper attribution for "Official Site"
c) Proper number, format, truncation for deep-links
d) Proper placement and usage of inner-search boxes

The picture below (Figure 3) depicts the items that can be subjected to validation:

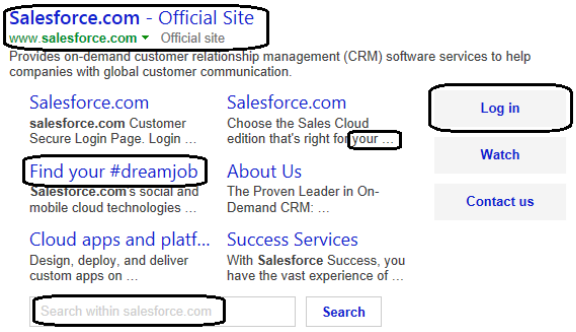


Figure 3. Validation aspects for web-search deep-links

There are two types of tests that can be used in this model:

1) *Custom Tests* are specific for only certain states (or actions). For instance, the deep-links validation shown above is an example of custom test since it only applies to pages originated from navigational queries
2) *Invariant Tests* verify general invariants that should always be true (or always be false) no matter what state we are

Invariant Tests are very powerful since they apply to all states (or actions). It is important and recommended that the product being tested be properly instrumented with test hooks in order to enable invariant conditions that can then be tested through invariant tests. An example of an invariant test would be a test that looks for java script errors. No state (or action) should lead to a Java script error on the page. We instrumented some of the Bing pages so that whenever inside the Microsoft corporate network and when a certain query string parameter is passed in the URL, any Java script

 ISBN: 978-1-61208-387-2

error is caught via a global try/catch and written into a hidden HTML div tag [5]. With such instrumentation implemented, the invariant test for java script errors becomes trivial – basically checking for the presence of the java script error div tag. Other types of invariant tests are:

a) Links: no links should lead to 404 pages
b) Server Error: no state/action should lead to server errors
c) Security: no state/action should expose any security flaw (such as cross-site scripting [13])
d) Overlapping: no state/action should contain overlapped elements

Security invariants for instance are implemented by scanning the page and attempting to exploit potential vulnerabilities. An example would be cross-site scripting [13]: all the links and JavaScripts on the page are exercised with custom parameters handcrafted in order to exploit cross-site scripting vulnerabilities. Since Selenium allows the test to actually open and run the browser, if a cross-site scripting vulnerability is found the monitoring validation can then detect it based on the handcrafted parameter passed to the link or JavaScript.

Selenium also provides a capability of taking the screenshot of the current page. This allows the engineers to implement image-based test methods, some of which can be custom methods (such as the rendering and placement of some objects on the page specific to certain contexts) or invariants (such as the space between blocks on the page). Also, it is important to notice that some of the methods only apply to certain contexts (browsers, devices, markets or experiments). In such cases, the test needs to verify that the current context is relevant for the test in question to be executed.

VII. METHODOLOGY

Combining all the approaches described in this paper, we come up with the following methodology for synthetic exploratory testing or monitoring of large-scale web systems:

1) Mine the logs to create the user's profile Markov Chain. A user profile reprsents the states, actions and states transitions based upon mining of the logs

2) Retrieve the percentage distribution of different contexts (browsers, devices, markets and experiments)

3) Create custom and invariant tests that adhere to the subscription-based model

4) Stochastically run through the Markov Chain using Selenium or Selenium Grid. When testing search engines a key aspect is the generation of relevant queries to be used. It can be a combination of top queries based on frequency as well as segment-specific queries (such as queries that trigger local results or movie results)

5) Sporadically (time-based) switch contexts based on the distribution from #2

6) At each state (and action), apply the subscription-based tests from the library (#3). Alert in case of failures.

We differentiate monitoring from testing in terms of running the tests post-production and pre-production, respectively. The approach can be used for either one. However, we prefer to have deterministic tests as a pre-production mechanism, leaving the non-deterministic ones (such as the stochastic ones based on Markov Chains) as a monitoring mechanism (post-production). Also, the different tests have different priorities, so not all the tests will lead to an escalation (usually the invariant ones are deemed higher priority than the custom ones).

As the approach above executes, over time the critical monitoring paths will certainly be covered. Given that the approach follows a weighted-probability model, the critical paths will be covered more often than the non-critical ones. That is desirable since in today's fast-pace development environment of large-scale web systems, only the critical problems (the ones affecting the vast majority of users) get real attention; others are treated as low priority. The stochastic model is an elegant way to ensure highly-probable coverage of critical scenarios, and yet also cover some low-key scenarios.

Below are two examples of invariant failures when the model was applied to Bing.com. We used a set of 5 high-end servers executing around 1,000,000 state transitions per day, and running over 100 validation methods (of which 15% were invariant ones). The first example (Figure 4) is an invariant that looks for HTTP 500 server errors, in this case, generated by a combination of experiment and different interactions with the site:

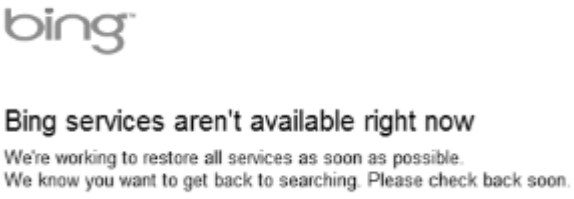


Figure 4. Issue discovered through an invariant test method

The second one is a low-priority invariant test based on image processing. In this case, the area to the right of the end of the search box should always contain background color only. But in the case of the German market, whenever search filters are present due to the long words in German, the placement of the filters (bottom left inside the top right rectangle) are going beyond the limits of the search box, breaking the pre-specified requirement (the requirements consist of User Interface principles and rules determined by designers that the code should always adhere to. In this particular case the specification clearly calls out that only background color can show up at the right side of the search box. Such a rule is violated in the case of German strings given that strings in the German language are usually longer

 ISBN: 978-1-61208-387-2

than the ones in English). Figure 5 shows an example of such an issue:

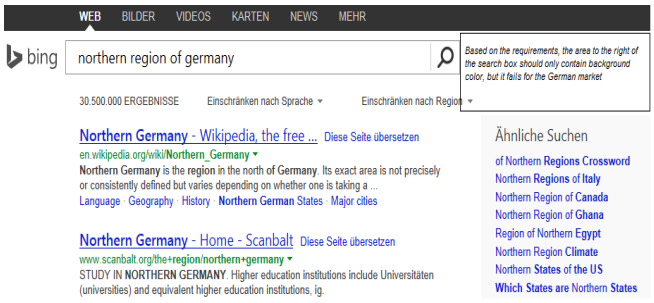


Figure 5. Example of an image-based error related to markets

Notice that the use of Markov Chains and context distributions allows the monitoring system to be highly adaptive: as the user patterns and context distributions change over time, the system will adapt itself based on the new data. The other important aspect is that the validation and monitoring mechanisms can certainly be extended to more than functional use, such as covering security concerns. At each step during the traversal of the Chain, we can also plug-in penetration tests which would be characterized as invariant methods.

VIII. TIME-BASED NEURAL NETWORKS

One of the limitations of Markov Chains is the fact that there is a need to introduce into the chains specific weighted-random events in order to account for the different contexts [11]. An alternative to overcome such limitation is to use a prediction model to, given a particular state and time for a user, predict the next state that the user is going to be based on the training data. The model that we selected was Artificial Neural Networks (ANNs [7]). The idea is to use features related to the current and previous states (pages), current and previous actions, current context, and generate the next most probable state, action and contexts. However, since the execution will have a temporal factor, there was an attempt to introduce a time-based feature into the ANN: the information about the user is segmented on a per-time unit, in our case every second. Such approach is similar to a Time Delay Neural Network (TDNN [8]). Mathematically speaking, the function F that the ANN will implement would then be:

F(States, Actions, Context, Time) = {State', Action', Context')

Here, we use the previous three states that the user has been previously (three previous pages visited), the corresponding three previous actions, the current context (which is a tuple consisting of browser, device, market and experiment) and the current time unit of the day in seconds (from 0 to 86,399). Figure 6 below depicts the ANN used.

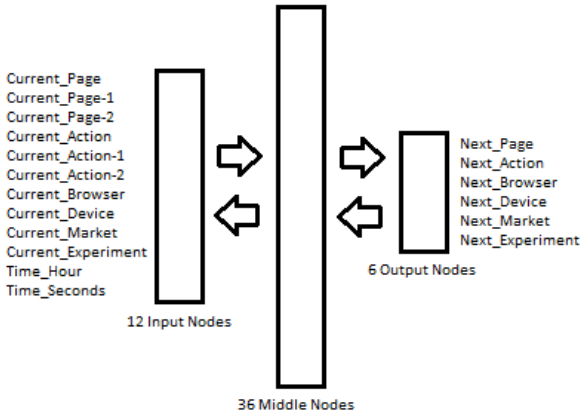


Figure 6. Time-based ANN

We decided to try the feed the network with the three previous pages and actions in order to have more accurate prediction. Trying with further states and actions did not improve its precision numbers (around 65% precision, see Table III below). We obtained the maximum precision numbers with the schema aforementioned: 12 input nodes where we split the time feature into two features: hour of the day (0-23) and seconds of the hour (0-59), 36 nodes in the middle (hidden) layer and 6 output nodes consisting of the state, actions, browser, device, market and experiment that the user should be in. During the execution, the ANN is fed with the current information about the user and the new information is given, changing the current user. Notice that the benefit of this new model is that it can be expanded to use other features too, and we do not need to rely on a weighted-random parallel mechanism to take into account the user's context. Also, given that the execution of the action usually takes over a second, it is very likely that the next input to the ANN will contain a different time parameter hence likely leading to a different output (the concern was that during the execution mode, the input would be consistently the same, hence producing the same output. It was not the case since the execution of each step took longer than 1 second). The learning model utilized was the error back-propagation [9]. We utilized a data set consisting of 1.5 million impressions in a 24h timeframe, proportionally sampled and distributed over the 24h window. 80% of this data was used as the training set whereas the remaining 20% was used as a test set.

With the Time-Based ANN fully trained, we swapped the model in the execution engine (Section VIII.4) with the Time-Based ANN. everything else in the methodology remained the same as described in Section VIII. Table III below depicts some comparative data between the Markov Chain Model and the Time-Based ANN model:

 ISBN: 978-1-61208-387-2

TABLE III. COMPARISON BETWEEN MARKOV CHAINS AND ANN FOR MONITORING

	Markov Chains	*Time-based ANN*
Training Time	~10min	~60min
Execution Time	400ms	30ms
Precision	N/A	65%
Min-Time-To-Failure (MTTF)	30min	108min

As one can see, due to the nature of the error back-propagation algorithm (with the high number of interactions for convergence), the time for the function to converge takes approximately six times longer compared to the training of the Markov Chain (which consists primarily of creating the weighted transitions). On the other hand, once the ANN is properly trained, its execution is significantly faster than the Markov Chain (likely attributed to the heavy weighted-random computations on the Markov Chains). The precision achieved for the ANN was not very high, around 65% for the test set, likely due to the fact that the time-based concept does not give a very predictable aspect to the back-propagation function despite its convergence. The Min-Time-To-Failure (MTTF) is characterized as the minimum time during the monitoring aspect to find the first monitoring failure or potential failure. In this aspect the Markov Chain converges faster than the Time-Based ANN. We believe this fact is related to the low precision for the Time-Based ANN. Our conclusion is that the Time-Based ANN gives a more elastic and expandable model where more features can be added in order to improve its precision; however the Markov Chain still gives the best outlook in terms of speed of training as well as better modeling the user's behavior. The Markov Chains are also significantly easier to implement compared to ANNs. Future work will be focused on augmentation of the ANN in order to improve its precision.

IX. CONCLUSION AND FUTURE WORK

Monitoring large-scale dynamic web sites across multiple browsers, devices, markets and experiments is a very complex task. In this paper, we have proposed a way to model the users' behavior via two stochastic methods: Markov Chains and Time-based Artificial Neural Networks. We compared these two methods in terms of their complexities, precisions and overall fitness for the problem of monitoring large-scale services. We combined Markov Chains and Selenium to recreate the same conditions experienced by real users in production. In addition, the validation approach is also changed from self-contained validation methods to a subscription-based model where the validation method subscribes to only the applicable states. Finally, validations can be invariant ones (applicable to all states) or custom ones (applicable to specific states). Future work will be focused on the time-based artificial neural network in order to achieve higher precision and better suitability for the problem of monitoring of web services. We presented an instance of the solution to monitor web search engines, but the same approach can be used to monitor other types of dynamic web services.

REFERENCES

[1] M. De Barros and C. Alex "Agile quality-centric development process of large-scale web systems", Swiss Testing Day 2014, March. 2014

[2] Gomez Network [Online]. Available from https://www.gomeznetworks.com/?g=1 2014.12.29

[3] Keynote [Online]. Available from http://www.keynote.com/ 2014.12.29

[4] M. De Barros, J. Shiau, C. Shang, K. Gidewall, H. Shi, and J. Forsmann, "Web Services Wind Tunnel: On Performance Testing Large-Scale Stateful Web Services", 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2007

[5] The HTML tag [Online]. Available from http://www.w3schools.com/tags/tag_div.asp 2015.01.12

[6] Selenium HQ Browser Automation [Online]. Available from http://docs.seleniumhq.org/ 2014.12.29

[7] Neural Networks [Online]. Available from http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html 2015.01.12

[8] Neural Network, Component of Measuring Set for Error Reduction [Online]. Available from http://www.measurement.sk/2004/S1/Vojtko.pdf 2015.01.12

[9] Error Backpropagation [Online]. Available from http://www.willamette.edu/~gorr/classes/cs449/backprop.html 2015.01.12

[10] E. Marcotte, "Responsive Web design", A List Apart, May. 2005

[11] M. De Barros, "Automated Synthetic Exploratory Monitoring of Dynamic Web Sites Using Selenium", PNSQC 2014, October. 2014

[12] R. Ramakrishnan, "Big Data @ Microsoft" [Online]. Avaiable from http://research.microsoft.com/en-us/events/fs2013/raghu-ramakrishnan_bigdataplatforms.pdf 2015.01.15

[13] S. Cook, "A Web Developer's Guide to Cross-Site Scripting" [Online]. Available from https://www.grc.com/sn/files/A_Web_Developers_Guide_to_Cross_Site_Scripting.pdf 2014.12.29

[14] Web Service Counters for the WWW Service [Online]. Available from https://technet.microsoft.com/en-us/library/cc786217(v=WS.10).aspx 2015.01.16

[15] User Experience at Google "Focus on the user and all else will follow", CHI 2008 Proceedings, April. 2008

[16] C. Liu, R. W. White, and S. Dumais, "Understanding Web Browsing Behaviors through Weibull Analysis of Dwell Time

[17] M. Arbib, A. Theories of Abstract Automata (1st ed.). Englewood Cliffs, N.J.: Prentice-Hall, Inc. ISBN 0-13-913368-2, 1969

 ISBN: 978-1-61208-387-2

Context Sensitive Web Service Engineering Environment for Product Extensions in Manufacturing Industry

Dragan Stokic, Ana Teresa Correia
Institute for Applied System Technology, ATB-Bremen,
Bremen, Germany
e-mail: dragan@atb-bremen.de, correia@atb-bremen.de

Abstract— **Modern industrial companies aim to extend their products with services as fundamental value-added activities and reduce the product to be just a part of the offering. Web based services offer excellent opportunities for such product extensions. To build such services and to meet requirements of mass customization, the manufacturers of machines and equipment for production of mass customized products need powerful service engineering environments to allow for multi-directional exchange of knowledge between product design, service design and manufacturing, as well as customers and other relevant organizations across the value chain, distributed all over the globe. Specifically, they need feedback from their business customers to whom they sell their equipment, as well as from the final-product customers. The objective of this research is to create a new context sensitive, product-(web)service engineering environment based on a combination of Cloud Manufacturing, Product Data Management system and social software solutions, as well as a set of tools to support real time sharing of knowledge among various actors, from the designer up to the customer, aimed at companies producing machines for mass product manufacturers. The application of the environment in a shoe machine producer is presented.**

***Keywords-product extension services; web services; service engineering; context sensitivity; cloud manufacturing*.**

I. INTRODUCTION

To support dynamic building of new web based services around products, i.e., to build Product Extension Services, there is a need for strong collaboration among various actors across the value chain [1][2]. In today's rapidly changing and globalizing markets, with new emerging technologies to support the mass production for manufacturing and service industries, the new paradigm, called "Mass Customization", represents the trend towards the production of highly customized products/services. Providing customers with the ability to co-design products/services based on their own preferences has been considered one of the most distinctive features of mass customization. However, to meet requirements of building Product Service Systems (PSS) for mass customization, the manufacturers of machines and equipment for production of mass customized products need feedback from their business customers to whom they sell their equipment and services, as well as from the final-product users/customers. Real time exchange of knowledge between the web service designers, product manufacturers, maintenance experts, as well product-service users, is unavoidable for the modern PSS design. This includes automatic data gathering and exchange along the value chain (e.g. data on energy consumption), but also tacit knowledge from various actors (e.g., experience of the maintenance staff), relevant for building services.

The classical product engineering systems do not meet their requirements concerning neither effective support of concurrent web service design, nor facilitating acquisition and re-use of the tacit knowledge. The industrial companies require a structured approach offered by such classical Information and Communication Technology (ICT) solutions, but, on the other hand, they need high flexibility from tools to allow capturing of dynamically changing requirements and experience of various actors [3]. Cloud Manufacturing (CMfg) provides new possibilities for collaborative design of PSS within such distributed enterprise, easily adaptable to highly dynamically changing conditions under which enterprises are developing and manufacturing their product-services [4]. On the other hand, tremendous experiences in social SoftWare (SW) solutions offer new opportunities for enterprises to capture and share experienced based knowledge among all actors across the value chain, highly relevant for design of web services for product extension. Such social SW solutions include social networks allowing for mass customers' feedback (through opinion mining) on a global market and wiki-like solutions allowing for flexible organization of knowledge/experience capturing/documenting by non-IT experts in companies [5], but such solutions have not been systematically used in a combination to classical engineering for PSS engineering.

The objective of the research presented, is to build a new web Service engineering environment for Product Extensions as a combination of classical product engineering tools, CMfg and social SW solutions, as such combination is likely to meet the requirements of distributed enterprises to allow for utilizing manufacturing intelligence and experience of all actors in the value chain, including both business customers / companies and product / service consumers.

The structure of the paper is as follows: In Section II a brief analysis of the state-of-the-art is provided. In Section III, the basic proposed concept is described, while in Section IV the application in industry is indicated. Section V describes the expected innovations and benefits, as well as future work.

II. STATE OF THE ART

Product Service Systems (PSS). Based on the analysis of the research dealing with PSS, it can be concluded that there

 ISBN: 978-1-61208-387-2

are no tools to develop SW for designing and managing PSS appropriate for machine vendors acting at the global market faced with mass customization requirements, even though some academic SW has been making breakthroughs, potentially in relation to collaborative engineering of PSS, in e.g. the capability to address integration of SW and hardware [6], or dynamic simulation [7]. Existing commercial SW mostly addresses the designing of physical products. Some SW solutions do support some single phases in development of PSS, but no engineering environment for PSS is provided. In other words, interchangeability between product and service has yet to be realized in commercial SW. However, this is exactly where a new type of SW is required in order to provide more opportunity to create an offering with higher value or an innovative solution.

Context sensitivity. With the recent advance of context-aware computing, an increasing need arises for developing formal context modelling and reasoning techniques. The basis for context-aware applications is a well-designed Context Model (CM). A CM enables applications to understand the user's activities in relation to situational conditions. Typical context modelling techniques include key-value models, object-oriented models, and ontological methods [8]. By context modelling, the problem of how to represent the context information is solved. However, how to extract context from the knowledge process and how to manipulate the in-formation to meet the requirement of knowledge enrichment remains to be solved. Since it is planned to model context with ontology, context extraction mainly is issue of context reasoning and context provisioning: how to inference high level context information from low level raw context data [9]. The application of context sensitivity for web service engineering in manufacturing industry has not been explored.

Analysis of big data volumes as customer feedback. The rise of social media has enabled citizens to express their opinions online about everything. Companies want to tap this source of information to understand reviews, ratings, recommendations, in order to identify new marketing opportunities, and manage their reputations. As businesses look to automate the process of filtering out the noise, understanding the conversations, and identifying the relevant content, many are now looking to the field of sentiment analysis (also known as opinion mining). Sentiment analysis can be separated in two categories: manual or human sentiment analysis and automated sentiment analysis. Many companies will need a combination of these methods to combine the capabilities of human interpretation witch computational capability of automatic search and analysis. Automated sentiment analysis of digital texts can be performed combining elements from machine learning, e.g. support vector machines, and semantic orientation, e.g. ontology. This research is interested in understanding the polarity of a given text or document, i.e. if the author has a positive, negative or neutral opinion [10]. The use of such technique for web service engineering in manufacturing industry has not been sufficiently examined.

III. OBJECTIVE AND BASIC CONCEPT

The objectives of the research are to develop:

- New Service Oriented Architecture (SOA) - based engineering environment for design of web services around products based on real time sharing of knowledge among product design, service design and manufacturing within distributed enterprises based on a combination of CMfg and Social SW solutions, allowing for involvement of customers on the global market, both business and final-product, where a novel PSS ontology is a key bonding element.
- Set of SW services to support context sensitive capturing and searching of knowledge for service design functionality and reusability, as well as for context sensitive analysis of big volumes of data gathered over the globally distributed customers, design, manufacturing and suppliers.

Concretely, the envisioned developments include: a) an open and extensible environment built on the foundations of Product Data Management/Product Lifecycle Management (PDM/PLM), and b) a methodology and accompanying tools to support the collaborative product-service design. The environment is specifically focusing on conceptual product-service design, but, to a large extent, it will be applicable for detailed web services design.

A new *engineering environment* for real time sharing of knowledge among various actors involved in service design within distributed enterprises is under development. The conceptual architecture of the environment is presented in Fig. 1. For many manufacturing companies services development starts to be or will be in a near future a core, value-generating process. To satisfy the need to differentiate them in the market place, nearly every company's development processes have unique properties. The diversity and spectrum of methods and skills required to perform web service development processes is huge. Engineers from different disciplines and specialists with various backgrounds have to address a part or an aspect of PSS by having a partial model and collaborating with other experts. It is necessary to define common concepts and language used to interface between these models. However, it is not reasonable to totally integrate those models into a complete, holistic view of PSS, covering each and every aspect of all the methods used to define PSS. Furthermore, it is sensible that every discipline uses the best available tools to perform their tasks.

The new engineering environment is therefore built as an open solution with loose coupling of different tools. The requirements of several industrial companies in different sectors (all proving equipment for the manufacturing of mass customized products) concerning PSS development have been analyzed and serve as a basis for the definition of new engineering environments and set of tools. The aim is to provide an environment for supporting web services development with ICT solution that embraces the diversity of users, methods, tools in use. The backbone of the environment are (PDM/PLM) solutions (e.g. open enterprise ARAS [11]) and social SW solutions. The advantages of the PDM/PLM systems are well defined structures of data corresponding to the well-organized industrial processes [12]. However, such systems, due to their often "rigid"

 ISBN: 978-1-61208-387-2

structures, may restrict possibilities to collect experience from various actors in the value chain, especially in dynamically changing conditions in highly flexible manufacturing. Especially experience from shop-floor (e.g., in adding sensors), and customers, is very difficult to be captured using classical PDM/PLM systems. Social SW allows for flexible capturing and presentation of experience of different actors, but often suffers from missing structures limiting reuse of these experiences.

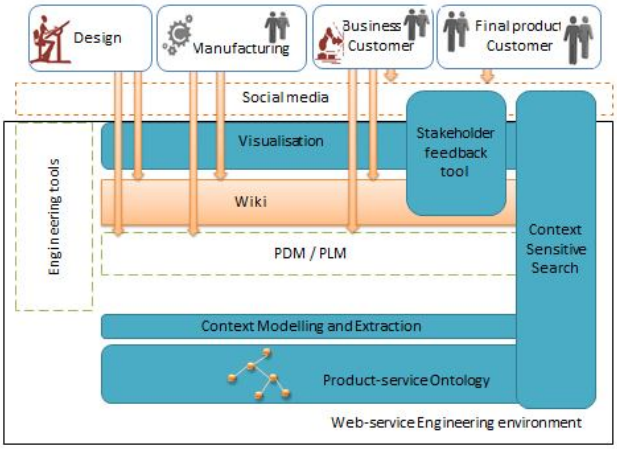


Fig.1. Concept of web service engineering environment

These solutions allow non-IT experts to provide their experiences in any form it fits them, e.g. problems in manufacturing of parts or assembly can be documented by the shop floor in a form which best suits specific manufacturing line/machine (each group can define its own 'form'). The feedback from customers on the PSS use can be also provided in any form which suits specific customer and/or service experts (maintenance staff, etc.). On the other hand, social media networks such as LinkedIn, Facebook, Twitter, Pinterest, Instagram, etc. allow gathering experiences of mass customers in very "free" forms.

The environment is a SOA-based open environment which combines open PDM/PLM and engineering systems and social SW, both wiki based solution for knowledge capturing of knowledge/experience of experts from manufacturing area, maintenance / service providers and business customers, and social networks/multimedia. The experience - based knowledge capturing via social SW is directly interrelated with the structured knowledge, collected within classical tools and by automatic monitoring of manufacturing processes, the product itself (e.g., sensors at machines) and usage (e.g., manufacturing processes where machines are used). The environment includes powerful visualization of knowledge, as well as a middleware to interface to various systems for automatic data gathering (e.g. Manufacturing Execution Systems - MES, Supervisory Control and Data Acquisition - SCADA, equipment, etc.) and Security, Trust & Privacy framework (for the sake of simplicity, middleware and security framework are not depicted in Fig. 1).

The objective is to allow engineers/designers, when using various engineering tools and/or PDM/PLM solutions to re-design/document/reconfigure product parts and service components, to have direct access to the experience from manufacturing area and business customers related to the corresponding part/service, but also to opinion/feedback of customers of final-product (e.g., features of shoes produced by the machine /component); see Fig. 2. The collaborative features allowing for real time sharing of knowledge between design teams and other actors are included.

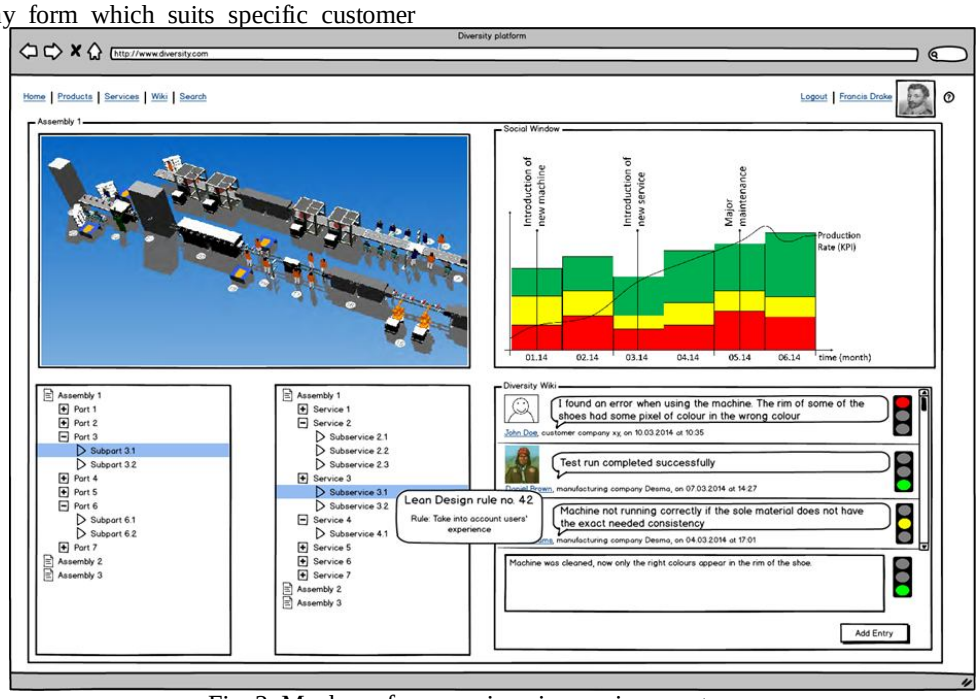


Fig. 2. Mockup of new engineering environment

 ISBN: 978-1-61208-387-2

The whole environment is made in CMfg, i.e., in order to allow for product–service co-design, the manufacturing environment (both manufacturing of machines/equipment and use of machines/equipment at the business customers) is virtually represented [4][13]. This PDM/PLM system is embedded in a cloud where all the actors in the value chain plus the machines and processes are able to provide data/knowledge and access the knowledge that is both saved in the PDM/PLM systems, as well as in the social SW solutions in the cloud. By this, dynamic feedback among various actors can be established, as well as life cycle aspects can be efficiently taken into account. The social SW is extended with a number of plugs-ins and additional functionality to allow for effective capturing of the unstructured knowledge. Security aspects, especially critical for CMfg, are analyzed, and appropriate mechanisms will be integrated.

As explained above, the bonding element between the classical engineering tools, PDM/PLM and social SW is a common *ontology* for product-service. Research on product-service has been carried out for many years and in various disciplines; however, even a consolidated set of terminologies has not been established [14]. A common ontology has not been released in industrial practice. Thus, one of the goals of the research lies in developing flexible, open engineering environment and ontology to realize the environment and interconnect various SW approaches and tools. An overall ontology will be built starting from [14].

In order to support designers to easily obtain information/knowledge relevant for her/his current task within the enormous amount of data gathered by automatic data collection, knowledge within PDM/PLM and different engineering tools, knowledge /experience collected within social SW from manufacturing end customers, etc., a *context sensitive search* functionality is developed to support all other tools in the environment. Therefore, the whole solution is context sensitive. The context sensitivity includes four key elements: CM, context monitoring services, context extraction services and search [9]. CM represents an abstract description of collaborative work relevant for PSS design activities in general. To enable its direct use by context-sensitive services, CM is formally expressed as ontology. Its concepts, attributes and relations are directly derived from the collaborative situations in building web services in dynamic industrial settings. CM is based on the collaboration patterns, i.e., typically occurring forms of collaborative work relevant for PSS and extended with scenario specific concepts (processes, products, technology).

A set of generic context monitoring services provide basic monitoring functionalities to monitor processes in which data/knowledge is acquired (e.g. design, manufacturing area both were machines are produced and where they are used), needed to identify/extract context. The Context Extraction Services observe activities within the new environment using Monitoring services. Context Extraction Services analyze structured and unstructured data to determine the context of the current situation and to identify what activity the users are currently involved in. The Context Extractor uses Context Association Network and Context Hierarchy Tree approaches. By using an appropriate CM and unstructured information provided by users or devices, the context extraction services process this unstructured information, to automatically annotate it.

Tools for analysis of stakeholder feedback enable context sensitive analysis of the knowledge gathered in wiki, social media networks and classical engineering tools. Specifically, the feedback from final-product customers, where mass customization plays central role, needs to be analyzed and provided to machine/equipment vendors in order that they can improve design of their services to accommodate mass customization requirements. Information sources on social networks are continuously mined to identify opinions about products, services and specific features. Using opinion mining, keywords identify products and features and associate the opinion expressed by customers.

IV. APPLICATION

The new environment is aimed at manufacturing companies producing equipment and delivering them at the global market in various mass production sectors (shoe, food packaging, etc.), which require new solutions for effective collaboration among various actors, as a most critical aspect of the PSS design process.

The environment is currently tested and applied at a German company producing machines for the shoe industry, which needs to combine feedback from their business partner (shoe manufacturers) and feedback from the shoe buyers in order to improve their machines and various services around these machines and allow for mass customization of shoes, which is one of the key requirement in today's global shoe market. They intend to improve the design of their machines and web services around their products (e.g. maintenance and monitoring of their machines/systems). They also intend to use these web services (e.g., service for remote diagnostics) to automatically collect, in real time, data/knowledge on performance of their machines at their business customers, aiming to further enhance design of the machines and services. In order to allow for effective building of services by adding sensors at their machines, they need to establish feedback from their manufacturing area of their machines to design processes. Besides data which can be collected by automatic measuring in processes and over services, it is important to gather experience of people involved in these processes. Of special importance is to collect knowledge/experience of their service teams and business customers (shoe manufacturers) distributed all over the world. Therefore, they need to allow different actors to easily document their experience using web services, taking into account cultural differences in different world regions and highly variable conditions under which the machines are used (e.g., low education levels of the machine operators, etc.). The social SW solution proposed will allow collecting experience from maintenance, while the stakeholder feedback tool will analyze feedback from shoe buyers. Both types of knowledge will be correlated with the PDM/PLM solution and support building of various web services, i.e. the proposed combination of CMfg and social SW allows collecting of the experience of actors in an easy and less formalized way but structured enough to be re-usable for

 ISBN: 978-1-61208-387-2

improved web services around machines (e.g., for improved diagnostics). By this, they will build a unique system where knowledge from both machine manufacturers and customers are collected to be re-used for optimization of service design.

V. CONCLUSION AND FUTURE WORK

The main innovation lies in solving the crucial problem of how to support PSS design process at machine/equipment vendors, faced with the challenge of mass customization of final-products produced by their machines/equipment, by provision of appropriate knowledge, both formalized and experience based knowledge, based on combination of classical engineering tools, PDM/PLM systems and social SW solutions. This includes data mining of high volume of data provided by the shop-floor experts (manufacturers of the machines), business customers and final-product customers, as well as the context extraction from the content created/used within dynamic collaborative work and manufacturing processes and/or the data provided by different services. In order to achieve such a solution, the research provides several innovations: (a) It brings a step towards development of an engineering environment supporting development of PSS. As indicated in Section II, to date, there have been insufficient attempts to provide such an environment. However, the research in recent years has created a number of methods and SW useful for the development of such PSS engineering environment [15]. The research will focus on the development of ontology for PSS [12]. (b) This is one of the first attempts to combine classical engineering tools, PDM/PLM solutions with CMfg and social SW to efficiently provide experience and knowledge from shop-floor and user feedback from the global market to the web service designers' desks. (c) The new solutions are context sensitive, in order to support the user to cope with enormous amount of knowledge to be managed and allow for higher re-usability of components and services [16]. (d) The research contributes to bringing data mining algorithms to higher maturity level applicable in (manufacturing) industry by enhancing existing and developing novel ones to meet the application requirements. The proposed combination of advanced technological solutions will bring considerable benefits to the manufacturing companies in terms of reducing time to market in building new and/or upgrading existing machines with web services as different manufacturability, environmental and final-product mass customization aspects will be effectively taken into account already in conceptual service design. The approach will allow for improvements in knowledge sharing across product-service lifecycle, as well as better product-service offerings addressing customer needs. It will also allow for better addressing sustainability across the entire service lifecycle.

The new environment and tools are currently under development and first testing trials by the users are going on.

ACKNOWLEDGMENT

This work is partly supported by the DIVERSITY (Cloud Manufacturing and Social Software Based Context Sensitive Product-Service Engineering Environment for Globally Distributed Enterprise) project of EU's H2020 framework, under the grant agreement no. 636692. This document does not represent the opinion of the European Community, and the Community is not responsible for any use that might be made of its content.

REFERENCES

[1] F. H. Beuren, M. G. Gomes Ferreira, and P. A. Cauchick Miguel, "Product-service systems: a literature review on integrated products and services," J. Clean. Prod., vol. 47, May 2013, pp. 222–231.

[2] M. Boehm and O. Thomas, "Looking beyond the rim of one's teacup: a multidisciplinary literature review of Product-Service Systems in Information Systems, Business Management, and Engineering & Design," J. Clean. Prod., vol. 51, , Jul. 2013, pp. 245–260.

[3] M.P. Sorli, and D. M. Stokic, Innovating in Product/Process Development, Springer-Verlag, Heidelberg, 2009.

[4] X. V. Wang and X. W. Xu, "ICMS: A Cloud-Based Manufacturing System," in Cloud Manufacturing, W. Li and J. Mehnen, Eds. Springer London, 2013, pp. 1–22.

[5] D. Ortiz-arroyo, "Chapter 2: Discovering Sets of Key Players in Social Networks," in Computational Social Network Analysis, A. Abraham, A.-E. Hassanien, and V. Snágel, Eds. London: Springer London, 2010.

[6] H. Meier, R. Roy, G. Seliger , Industrial Product-Service Systems - IPS2. In: CIRP Annals - Manufacturing Technology 59 (2), 2010, pp. 607–627.

[7] L. Pesonen, S. J. Salminen, J-P Ylén, P. Riihimäki, Dynamic simulation of product process, in Journal Simulation Modelling Practice and Theory Vol. 16 No. 8., 2008, pp. 1091-1102.

[8] Strang, T. and C. Linnhoff-Popien. A Context Modeling Survey. in Workshop on Advanced Context Modelling, Reasoning and Management as part of the Conference on Ubiquitous Computing - The Sixth International Conference on Ubiquitous Computing. 2004. Nottingham, England.

[9] D. M. Stokic, S. Scholze, and O. Kotte, "Generic Self-Learning Context Sensitive Solution for Adaptive Manufacturing and Decision Making Systems", ICONS 2014, Nice, 2014.

[10] B. Snyder, R. Barzilay. "Multiple Aspect Ranking using the Good Grief Algorithm". Proceedings of the Joint Human Language Technology/North American Chapter of the ACL Conference (HLT-NAACL), 2007, pp. 300–307.

[11] ARAS PDM Software [Online], Available from: http://www.aras.com/standards/standard.aspx?name=PDM-Software, last accessed on 19.02.2015

[12] A. Saaksvuori, and A. Immonen, Product Lifecycle Management, 2nd ed., Springer, Berlin, 2008.

[13] A. Weiss, Computing in the clouds, Networker, vol. 11, 2007, pp. 16-25.

[14] G. Annamalai, et.al. An Ontoloogy for PSS, Chapter in Functional Thinking for Value Creation, Springer-Verlag, 2010, pp. 231-236.

[15] J. Lee, and M. AbuAli, "Innovative Product Advanced Service Systems (I-PASS): meth-odology, tools, and applications for dominant service design," Int. J. Adv. Manuf. Technol., vol. 52, no. 9–12, , Feb. 2011, pp. 1161–1173 .

[16] Y. Geum and Y. Park, "Development of technology roadmap for Product-Service System (TRPSS)," in 2010 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), 2010, pp. 410–414.

 ISBN: 978-1-61208-387-2

Developing and Adopting Trust-aware Collaborative Prediction of QoS for Service-based Systems

Feng-Jian Wang, Chen-Yang Chen, Po-Han Chen
Dept. of Computer Science
National Chiao Tung University
Hsinchu, Taiwan
Email: {fjwang@cs.nctu.edu.tw, admachen@gmail.com, sihalon@gmail.com}

***Abstract*—An application (service) can be composed of existing services. Generally, an appropriate service might be selected according to the predicted Quality of Service (QoS) values, and the most common approach for service selection is using collaborative filtering for prediction. In this paper, we present a trust-aware QoS prediction method for service selection, which is inspired by trust relationships in social networks. We use direct and indirect similarities of opinion among users, each of which are combinations of belief, disbelief, and uncertainty. Then, we can derive similarities among users, select similar users, and predict the QoS value of a service. The experimental results show that the accuracy of QoS values determined using our method is better than that using other methods.**

Keywords–Trust-aware; QoS Prediction; Service-Based Systems; Opinion.

I. INTRODUCTION

Web services are widely adopted to provide various functions in fields such as scientific research, e-commerce, healthcare, and aerospace. Developers can create applications with low costs and short development times by adopting existing services. However, individual services to be adopted for composition must meet not only functional but also non-functional requirements such as response time, reliability, and cost constraints. For better service selection, quality-of-service (QoS) values can be used. Several approaches have been presented to determine the QoS values of services [1]-[2]. Liang *et al*. [1] proposed a framework to predict QoS values by extracting service features. Zhen *et al*. [3] adopted matrix factorization to predict QoS. Zheng *et al*. [4] applied collaborative filtering (CF) to predict unknown QoS values. Their approach selects similar users based on the Pearson Correlation Coefficient (PCC). Moreover, two studies have been conducted for improving their CF-based approach by applying factors: neighborhood [2] and time-aware [5]. However, the predicted results are inaccurate when the number of invocation records is too low.

In general, similarity has propagative characteristics, so similarity exploitation of users that reside at longer distances is reasonable. To improve QoS prediction value accuracy, we propose an approach based on the concept "the more similar the users selected for prediction, the more accurate the predicted QoS value". For example, consider three users having the following properties: the first two users are not similar to each other, but both are similar to the third. With a two-hop distance [6], the first user may have some similarity to the second; in other words, there is an indirect similar relationship between them.

In our approach, prediction accuracy is improved based on a social trust network. For example, we might be able to know with certainty whether a proposition will be true or false [7]. Both *direct* and *indirect similarities* comprise the kernel adopted in our work. User can be trusted more with higher direct similarity. Also, inspired by Josang's paper [8], "Opinion" has been introduced to express the extent of belief in some events, and direct opinion can be defined to be composed of three factors: *uncertainty*, *belief*, and *disbelief*. An *indirect opinion* can be determined from the recommendation of other users based on their *direct opinion* [8]. Finally, the value for representing the indirect similarity of two users, can be estimated according to other users' indirect opinions.

We then present an algorithm to select users based on direct (i.e., PCC) and indirect similarity, and adopt their opinion to help improve QoS prediction by modifying QoS values using Resnick's formula correspondingly. The experimental results indicate that our approach is better than other approaches when the number of invocation records is small [4].

The rest of the paper is organized as follows. Section II describes the background and related work. Section III introduces our trust-aware approach. Section IV presents the experiments and makes comparisons. Finally, Section V concludes the whole paper and future work.

II. BACKGROUND

In this section, we describe service-based systems (SBS), QoS calculations, social trust, and related QoS evaluation and prediction approaches. Part A presents a review of QoS evaluation and prediction approaches for an application (service). Part B introduces trust and related work in recommendation systems.

A. QoS Evaluation and Prediction for SBS

SBSs are applications composed of individual services in more than one website. From the viewpoint of a composition workflow, a service can be treated as a process. During SBS development, it is better to select the most appropriate one from among web service candidates. For example, some consumers prefer cheap services, while others prefer highly reliable ones. It might be better to select a cheaper service if budget is a major consideration. Moreover, functionality might be quantified based on the values of necessary quality factors. With the rising popularity of SBSs, an increasing number of services with similar functions are supported by different service providers. Therefore, it is better to adopt an effective approach to evaluate Web services for selection, composition, and so on [9].

Collaborative filtering approaches [4][10] are widely adopted for predicting attribute values in recommendation systems. QoS values can be predicted using QoS records of different Web services from similar users, including failure probability, performance, and cost. Usually, similarities between users are defined in terms of PCC [11], which is a measure of the linear correlation between two variables X and Y, as expressed by (1), where $\overline{X}$ is the mean of X, $\overline{Y}$ is the mean of Y, and the coefficient value is between -1 and 1.

 ISBN: 978-1-61208-387-2

$$r = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}} \quad (1)$$

Data sparsity is one of the main challenges for current CF-based approaches [6]. When both numbers of users and services increase quickly, the *user-service matrix* (which will be explained later) commonly used in CF approaches could become very large and sparse. The cold-start problem is encountered because a new user invokes only a few services, which may lead to insufficiency of QoS values. To alleviate the cold-start problem, factors such as location [12] and time [7][11] have been adopted to improve prediction.

In conventional CF approaches, similar users are located within a distance of one hop. Similarity has propagative characteristics, so information exploitation of users located at longer distances would be reasonable. An example in Figure 1 shows that User 1 is similar to User 2, User 2 is similar to User 3, and both similarities are of one distance. It is reasonable to treat both User 1 and User 3 as indirectly similar, and there is a similarity of two-hop propagations between them [13].

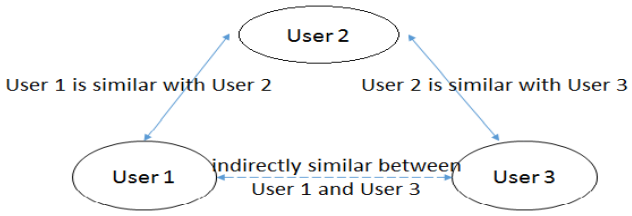


Figure 1. Example of similarity propagation.

B. Social Trust in Cloud Computing and Recommendation system

Social networks are used to reflect real-world relationships, allow users to share information, and form connections among users [14]. Chard *et al*. [14] designed a social cloud system and implemented it. In the social cloud system, users can discover storage services contributed by their friends based on existing trust relationships. In the study of recommendation systems, Singla *et al*. [15] applied data mining techniques to study the relationships among users, and observed that users are more likely to share interests with similar users. Pitsilis *et al*. [16] announced that the performance of a recommendation system can be enhanced based on potential trust among users.

Each of above trust models is adopted to solve one or more specific problems in social cloud systems or recommendation systems. In contrast, Subjective Logic [8] is a general trust model for representation and reasoning of trust. It operates on subjective beliefs and uses "*opinion*" to denote the representation of a subjective belief. Pitsilis *et al*. [13] and O'Donovan *et al*. [17] used subjective logic for reasoning the trustworthiness among users, and their experiments indicate that the accuracy of recommendation values can be improved with trustworthiness.

Because of the incompleteness and inconsistency of knowledge, it is impossible to know for sure whether a proposition would be true or false [7]. Thus, *opinion* [8] has been defined to express the extent of belief in some events, and the definition includes three factors: belief (b), disbelief (d), and uncertainty (u). The mathematical definition and computation of direct opinion are described below.

Let ω_e^i, $user_i$'s opinion about an event e, be a three tuple, where b_e^i is $user_i$'s belief in event e, d_e^i is $user_i$'s disbelief in event e, u_e^i is $user_i$'s uncertainty in event e and $\omega_e^i = \{b_e^i, d_e^i, u_e^i\}$. Figure 2 indicates the profile of (2).

$$b_e^i + d_e^i + u_e^i = 1, \quad \{b_e^i, d_e^i, u_e^i\} \in [0,1]^3 \quad (2)$$

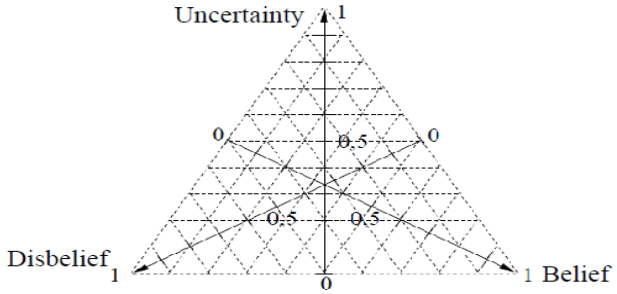


Figure 2. Opinion triangle.

Two operators are provided by *Subjective Logic: recommendation* $\otimes$ in (3) and *consensus* $\oplus$ in (4) [8]. Both can be used for deriving opinions regarding other users' opinions. Based on recommendation $\otimes$, $user_i$'s opinion about event e due to $user_j$ is derived from $user_j$'s recommendation. In (4), let the degree of trust of $user_i$ to $user_j$ be b_j^i. Then, b_e^{ij}, d_e^{ij} and u_e^{ij} are the belief, disbelief, and uncertainty values, respectively, about e of $user_i$ being persuaded by $user_j$ and can be determined as $b_j^i \times b_e^j$. d_e^{ij} is determined by multiplying b_j^i and d_e^j. $user_j$'s certainty can be defined as $1 - u_e^j$ (i.e., $b_e^j + d_e^j$). Let ω_e^{ij} be $user_i$'s opinion about e recommended by $user_j$, and let ω_j^i be $user_i$'s opinion about the degree of trust in $user_j$. Mathematically, ω_e^{ij} can be defined as follows:

$$\omega_e^{ij} = \omega_j^i \otimes \omega_e^j = \{b_e^{ij}, d_e^{ij}, u_e^{ij}\},$$
$$where \begin{cases} b_e^{ij} = b_j^i \times b_e^j \\ d_e^{ij} = b_j^i \times d_e^j \\ u_e^{ij} = 1 - b_j^i \times (1 - u_e^j) \end{cases} \quad (3)$$

The effect of the consensus operator $\oplus$ can help reduce uncertainty by applying opinions from both $user_i$ and $user_j$. The consensus opinion $\omega_e^{i,j}$ between $user_i$ and $user_j$ on event e can be defined as follows [8]:

$$\omega_e^{i,j} = \omega_e^i \oplus \omega_e^j = \{b_e^{i,j}, d_e^{i,j}, u_e^{i,j}\},$$
$$where \begin{cases} b_e^{i,j} = (b_e^i u_e^j + b_e^j u_e^i)/(u_e^i + u_e^j - u_e^i u_e^j) \\ d_e^{i,j} = (d_e^i u_e^j + d_e^j u_e^i)/(u_e^i + u_e^j - u_e^i u_e^j) \\ u_e^{i,j} = (u_e^i u_e^j)/(u_e^i + u_e^j - u_e^i u_e^j) \end{cases} \quad (4)$$

A consensus operator combines evidences for different users, and several approaches [13][16][18][19] provide different methods to combine opinions based on the consensus operator.

III. PROPOSED METHODOLOGY

This section presents an approach to provide a trust-aware QoS method for predicting the QoS values of web services. Figure 3 shows a global viewpoint of our approach, including both input and output of two functions: 1) to help service selection with QoS prediction 2) to help evaluate a new composite service. Our work in this paper is focused on function one (prediction part), which involves three stages: In Stage 1, indirect similarity between two users is determined by applying Subjective Logic [8]. In Stage 2, a set of similar users is selected according to their direct and indirect similarities to

 ISBN: 978-1-61208-387-2

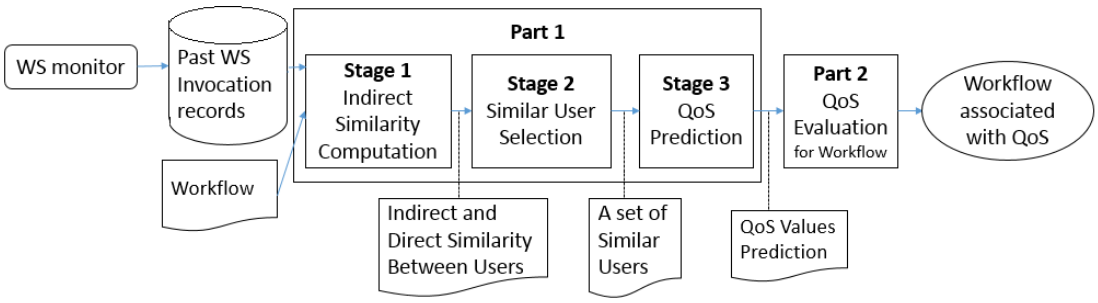


Figure 3. Workflow of trust-aware collaborative QoS prediction and evaluation.

the designated user. In Stage 3, QoS values are predicted by employing the QoS values of similar users. The three stages are described below.

A. Stage 1: Indirect Similarity Computation

The first stage introduces the *user-service* matrix, for calculating direct similarity and subjective logic [8] to determine indirect similarity between users in our approach.

1) Direct Similarity Calculation

A user-service matrix is a 2-dimensional matrix, where each row represents a distinct service to be invoked and each column represents a distinct user. Each entry in the matrix contains some recorded QoS values, which are usually called *invocation* records.

In our work, such entries in a user-service matrix are defined to contain three tuples, where the first tuple is reliability value, second is response time, and third is throughput for the corresponding service and invoker (user). The entry is called *null* when all values inside it are zero. For example, $user_1$ has never invoked $service_2$, and entry $E_{1,2}$ is null. $service_1$ has been invoked by $user_1$, and $E_{1,1}$ contains (90%, 100ms, 24), i.e., the reliability of $service_1$ is 90%, response time is 100 ms, and throughput is 24 kbps.

By applying PCC [11], direct similarity between $user_i$ and $user_j$ is computed by employing (5) according to QoS values of the services invoked by both of them.

$$Sim(i,j) = \frac{\sum_{s \in S_i \bigcap S_j} (E_{i,s} - \overline{E_i})^T (E_{j,s} - \overline{E_j})}{\sqrt{\sum_{s \in S_i \bigcap S_j} (|E_{i,s} - \overline{E_i}|)^2} \sqrt{\sum_{s \in S_i \bigcap S_j} (|E_{j,s} - \overline{E_j}|)^2}} \quad (5)$$

$$\overline{E_i} = \frac{1}{|s_i|} \sum_{s \in S_i} E_{i,s} \quad (6)$$

where i and j represent $user_i$ and $user_j$ separately, S_i and S_j are two sets of services invoked by $user_i$ and $user_j$ separately, $S_i \bigcap S_j$ is a set of services invoked by both of them and $\overline{E_i}$ is the average QoS value of all services invoked by $user_i$ shown in (6).

2) Indirect Similarity Calculation

In general, the greater the number of services invoked by both users, the greater is the number of common invocation records owned by them. Consider two users $user_i$ and $user_j$. To determine $user_i$'s uncertainty toward the notion that "applying $user_j$'s invocation records to predict QoS might be useful," the number of services invoked by both users may be adopted as evidence. When the number of common invocation records is larger, $user_i$ might be more sure that applying $user_j$'s invocation records to predict QoS is useful. Furthermore, the uncertainty can be determined using (7), a formula derived from [6]:

$$u(i,j) = (n_{i,j} + 1)^{-m} \quad (7)$$

where $n_{i,j}$ denotes the number of services invoked by $user_i$ and $user_j$. m is a positive number.

Practically, the number of services invoked is very small [4]. If m increases (e.g., to be greater than 2), the uncertainty determined using (7) decreases dramatically. Moreover, this indicates the lower uncertainty is achieved when fewer services are invoked by both users. Such an uncertainty value is impractical, and it is better and reasonable to assign m as 1 in our experience.

In the case that two users have high similarity values (e.g., high PCC value), the invocation records of one user might be useful for predicting the QoS value of the other user. To derive $user_i$'s belief value for "Applying $user_j$'s invocation records to predict QoS is useful," direct similarity (i.e., PCC value) between $user_i$ and $user_j$ may be adopted as evidence. Such a volume for $user_i$ can be defined according to (8), which is derived from [6]:

$$b(i,j) = \frac{1}{2}(1 - u(i,j))(1 + Sim(i,j)) \quad (8)$$

In (8), the certainty value is $1 - u(i,j)$. When $user_i$ and $user_j$ have a higher PCC value, they might have higher belief as well. Thus, $1 + Sim(i,j)$ is adopted to compute the weight for deciding the ratio of belief to disbelief. Moreover, equation (8) includes a one-half operation to restrict $b(i,j)$ within the interval [0,1]. Correspondingly, the disbelief of $user_i$ to such an application can be determined according to (2) (i.e., $b(i,j) + d(i,j) + u(i,j) = 1$).

An opinion can be direct or indirect. The value of a direct opinion can be derived by (7) and (8), which compute uncertainty and belief values [20]. The value of an indirect opinion can be derived from the recommendations of other users using the recommendation operator [8]. In (1), the PCC value between $user_i$ and $user_j$ is assigned as zero when neither of them invokes a service. In this case, $user_j$ is not put into the set of similar users for predicting the QoS value of $user_i$. However, when both users are similar to a third user, the invocation records of $user_j$ may be helpful for predicting the QoS value of $user_i$ because the similarity can be propagated via the third user [13]. To introduce the effect of indirect similarity, indirect opinion may be adopted.

An example in Figure 4 shows how to derive an indirect opinion. In the example, the belief values of trust degree for $User_i$ to ($User_k$, $User_l$ and $User_m$) and ($User_k$, $User_l$ and $User_m$) to $User_j$ are both larger than a threshold. Thus,

indirect opinion between $User_i$ and $User_j$, $\omega_j^{i(k,l,m)}$, can be determined using the recommendations of $User_k$, $User_l$, and $User_m$ [8].

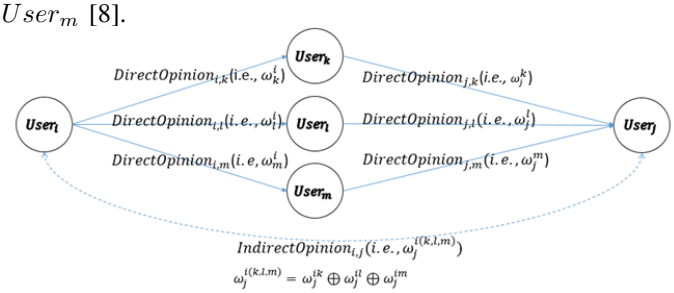


Figure 4. Example for Determining Indirect Opinion.

The indirect opinion of two users $user_i$ and $user_j$ can be defined as the consensus of direct opinions of users who have direct opinions about both aforementioned users. According to the example shown in Figure 4, $\omega_j^{i(k,l,m)}$ can thus represent the indirect opinion from $User_j$ to $User_j$. The indirect similarity between $user_i$ and $user_j$ can determined using indirect opinion by applying (9). Figure 5 shows the distribution of indirect similarity values and indicates that the growth rate of belief to indirect similarity value depends on the uncertainty value.

$$Sim'(i,j) = 2 \times (1 - u_j^i) \times b_j^i - 1 \quad (9)$$

1) where u_j^i is the uncertainty value and b_j^i is the belief value for $user_i$ to $user_j$. 2) direct opinions are obtained from users selected following the process described in the next paragraph. $1 - u_j^i$ is $user_i$' s certainty, employed to be a weighting value for adjusting the growth rate of belief to indirect similarity by multiplying b_j^i and $(1-u_j^i)$. Then, we define Sim' as multiplication by 2 and subtraction of 1 to restrict its value to [-1,1].

To select appropriate users to recommend the opinion, we define an opinion threshold (OThreshold) between 0 to 1 in order to determine the users whose opinions can affect the designated user effectively. OThreshold is helpful for predicting QoS.

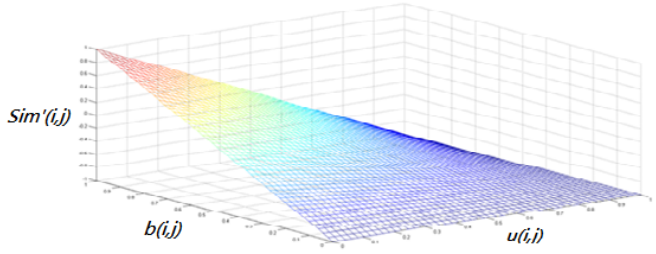


Figure 5. Distribution of (9).

Algorithm 1 details the procedure for deriving indirect similarity. Lines 3-7 select the users whose belief values from $User_i$ to them and from them to $User_j$ are larger than $OThreshold$ and place them into $TrustUserSet$. Indirect opinion between $User_i$ and $User_j$ is initialized by their direct opinion. Lines 8-11 represent a loop that considers all recommendations from the users in the trust user set and determines indirect opinion of each user for $User_i$ to $User_j$. At each turn of the loop, the indirect opinion value due to or via recommendation of $User_l$ is determined using the recommendation operator at line 9. Each indirect opinion is combined with the consensus operator at line 10. Because u_j^i and b_j^i are two attributes in ω_j^i, $Sim'(i,j)$ is computed after ω_j^i is computed.

Algorithm 1 Deriving Indirect Similarity

Input: $User_i$, $User_j$ and $OThreshold$
Output: Indirect similarity $Sim'(i,j)$ between users

1: **function** CALC_INDIRECTSIMILARITY()
2: $TrustUserSet \leftarrow \emptyset$,$w_l^i \leftarrow \emptyset$, $w_j^i \leftarrow \emptyset$;
3: **for** each user $User_k$ **do**
4: **if** both $b(i,k)$, $b(k,j) > OThreshold$ **then**
5: $TrustUserSet \leftarrow TrustUserSet \bigcup \{User_k\}$
6: **end if**
7: **end for**
8: **for** each user IN $TrustUserSet$ named as $User_l$ **do**
9: $\omega_l^i \leftarrow \omega_l^i \otimes \omega_j^l$
▷ recommended by $User_l$ with recommendation operator
10: $\omega_j^i \leftarrow \omega_j^i \oplus \omega_l^i$ ▷ ω_l^i is added to ω_j^i by consensus operator
11: **end for**
12: **return** $Sim'(i,j)$ ▷ defined in (9)
13: **end function**

Figure 6. Deriving Indirect Similarity

B. Stage 2: Similar Users Selection

To select similar users who may help predict QoS values for the designated user, the first step is to divide users into two groups according to a given number k, where the PCC values between the first-group users (i.e., similar users) and the designated user are larger than those between the second-group users and the designated user. In general, QoS value predictions are inaccurate when k is unsuitable. For example, if the value of k is too large, users not similar to the designated user might be selected. Because there is no effective method to determine a suitable k value for division currently, an eigenvalue k can be adopted to be the number of first-group users instead. We study the influence of k and find that QoS values are more accurate and suitable when k is set to be within a specific range (set as 15 here). Besides, in a cold-start situation, PCC may not be able to find an adequate number of similar users. Our approach adopts indirect similarity values to improve user selection in the case there are less than k users selected owing to direct similarities. The selection approach associated with indirect similarity is the same as the first one except 1) the users are those who were not selected before, 2) selection data is indirect similarity, and 3) amount being selected is equal to k–the number of users selected in the first stage.

Algorithm 2 is designed for this selection. Each user in $AllUsers$ contains PCC values and indirect similarity values, determined using (5) and Algorithm 1, respectively, and the calculations are set in lines 2-5 of said algorithm. Lines 7-14 constitute a while loop for selecting similar users according to their PCC values to $User_i$. Lines 16-23 perform selection based on indirect similarity if the first while loop cannot get k users.

C. Stage 3: QoS Prediction

The QoS value predictions are determined according to (10) based on Resnicks's formula [11][21]. Let $\widehat{E_{i,s}}$ be $user_i$'s QoS value prediction for $service_s$, $\overline{E_i}$ and $\overline{E_j}$ represent the mean QoS values of all services invoked by $User_i$ and $User_j$

Algorithm 2 Selecting Similar Users

Input: $User_i$, an eigenvalue k, and $AllUsers$
Output: a set of similar users to $User_i$

```
1: function SIMILAR_USER_SELECTION()
2:     for each user named User_j in AllUsers do
3:         User_j.Similarity ← Sim(User_i, User_j)
4:         User_j.IndirectSimilarity ← Calc_IndirectSimilarity(User_i, User_j, OThreshold)
5:     end for
6:     select_count = 0
7:     while select_count < k do
8:         Fetch a user from AllUsers who has the greatest PCC value and is not in Similar_Users, and name it as User_j
9:         if User_j.Similarity ≤ 0 then
10:            break
11:        end if
12:        Similar_Users ← Similar_Users ⋃ {User_j}
13:        select_count = select_count + 1
14:    end while
15:    if select_count < k then
16:        while select_count < k do
17:            Fetch a user from AllUsers who has the greatest indirect similarity value and is not in Similar_Users, and name it as User_j
18:            if User_j.IndirectSimilarity ≤ 0 then
19:                break
20:            end if
21:            Similar_Users ← Similar_Users ⋃ {User_j}
22:            select_count = select_count + 1
23:        end while
24:    end if
25:    return Similar_Users
26: end function
```

Figure 7. Selecting Similar Users

respectively, and $S(i)$ be a set of users similar to $user_i$ generated by Algorithm 2 and $W_{i,j}$ be a weight value indicating the similarity between $user_i$ and $user_j$. Here $Similarity_{i,j}$ refers to the similarity between $user_i$ and $user_j$.

$$\widehat{E_{i,s}} = \overline{E_i} + \sum_{j \in S(i), E_{j,s} \neq null} W_{i,j} \times (E_{j,s} - \overline{E_j}) \quad (10)$$

$$W_{i,j} = \frac{Similarity_{i,j}}{\sum_{k \in S(i)} Similarity_{i,k}} \quad (11)$$

Algorithm 3 is designed to predict the QoS value of $user_i$. In the algorithm, a set of similar users is selected at line 2 by applying Algorithm 2. The predictive QoS value is initialized with the average QoS value of all services invoked by $User_i$ in line 3. Lines 4-9 constitute a loop for updating the predictive QoS value according to the QoS values of all similar users for $Service_s$. Finally, line 10 returns the predicted QoS values of $Service_s$ invoked by $user_i$.

IV. EXPERIMENTS FOR QOS PREDICTION

In this section, we introduce our experiments and results, and compare the results with those from existing works. Part A describes the technique for comparing the evaluation results of QoS prediction. Then, we discuss the influence of parameters such as the number of invoked services, eigenvalue k, and opinion threshold in Part B.

Algorithm 3 Predicting QoS values of a service

Input: $User_i$, and $Service_s$
Output: predicted QoS values of $Service_s$ for $User_i$

```
1: function QOS_PREDICTION()
2:     similar_users ← Similar_User_Selection(User_i, k)
3:     Ê_{i,s} ← Ē_i
4:     for User_j in similar_users do
5:         if E_{j,s} ≠ null then
6:             W_{i,j} ← Similarity_{i,j} / Σ_{k∈S(i)} Similarity_{i,k}        ▷ by (11)
7:             Ê_{i,s} ← Ê_{i,s} + W_{i,j} × (E_{j,s} − Ē_j)
8:         end if
9:     end for
10:    return Ê_{i,s}
11: end function
```

Figure 8. Predicting QoS values of a service

A. Comparison of Prediction Accuracy

To evaluate the accuracy of a QoS prediction technique, Mean Absolute Error (MAE) [4] is adopted for calculating the difference between real and predicted QoS values. In general, the smaller the calculated MAE value, the more accurate is the QoS prediction for a service. MAE is defined as follows [22]:

$$MAE = \frac{\sum_{S \in PS_i} |\hat{E}_{i,s} - E_{i,s}|}{|PS_i|} \quad (12)$$

where PS_i is a set of services whose QoS values are derived by the prediction for $user_i$, and $E_{i,s}$ and $\hat{E}_{i,s}$ is a pair of real and predicted QoS values of $Service_s$ for $user_i$ respectively.

In our evaluation work, the calculation of real QoS values is based on the invocation records collected in [23], and widely adopted as real QoS values for evaluation of QoS prediction methods [4][12][20]. The data in [23] were obtained from www.wsdream.net, a service broker website, and they represent 100 real-world Web services. To monitor the QoS values of these services, the system deployed 150 service consumers on Planet-Lab [24] distributed across 20 countries to invoke the services, and recorded 1,500,000 Web service invocations executed a hundred times by 150 distributed users on 100 Web services. Each invocation record indicates the QoS values of a service invoked by a user, and includes three items: response time, throughput, and execution state (i.e., failure or success).

The computation methods for QoS values are quality dependent. For example, reliability can be determined based on execution state. To minimize errors in our experiments, the response time of a service for a user was defined as the average of the response time from all invocation records of the service for the user. So does throughput, which is defined as the average of all throughput value. The reliability value of $service_j$ for $user_i$ is calculated according to (13):

$$R_{i,j} = \frac{S_{i,j}}{N_{i,j}} \quad (13)$$

where $S_{i,j}$ is the successful invoking times of $service_j$, and $N_{i,j}$ is the invoking times of $service_j$. That is, a smaller $R_{i,j}$ means higher reliability.

Given that each user invokes some services only in the real world, the user-service matrix is sparse. However, it is difficult

TABLE I. PREDICTION ACCURACY COMPARISON

Metric	Methods	Reliability			Response time			Throughput		
		Density 5%	Density 10%	Density 15%	Density 5%	Density 10%	Density 15%	Density 5%	Density 10%	Density 15%
MAE	UMEAN	0.0701	0.0755	0.067	1504.90	1545.41	1419.11	2557.50	2561.75	2548.39
	IMEAN	0.0877	0.0863	0.091	1665.00	1705.83	1758.32	2690.27	2637.06	2732.41
	UPCC	0.0442	0.0398	0.037	2076.14	974.48	859.48	1505.07	1361.50	821.41
	IPCC	0.0426	0.0435	0.0390	1371.27	1072.75	955.25	1635.31	1676.25	1608.47
	HPCC	0.0711	0.0538	0.0456	1663.59	1279.14	1278.79	713.04	532.45	308.76
	MF	0.4899	0.4293	0.3886	1721.41	1487.87	1426.77	1813.59	333.91	222.95
	Our Approach	0.0371	0.0362	0.0350	1363.01	879.75	794.67	1306.74	1153.64	636.70

to derive appropriate sparse matrixes (e.g., in different density) to carry out the experiments from real world databases. In our experiment, the QoS values of 100 Web services invoked by 150 users calculated according to the invocation records are stored in a 150×100 user-service matrix called *answer* matrix and we define an *experimental sparse* matrix, a 150×100 user-service matrix. The QoS values in the experimental sparse matrix are constructed based on the answer matrix. For example, if the density of the sparse matrix is 5%, 5% of the entries in the experimental sparse matrix are filled with real QoS values from the answer matrix and the others are filled with NULL. The experimental sparse matrix is built in steps:

1) A new user-service matrix is built and all entries are set to NULL.
2) Let NE be the number of entries (e.g., 15,000 in our experimental sparse matrix), $density$ be the density of the experimental sparse matrix, and NS be the number of selected entries given as multiplication of NE and $density$. A set of numbers SN is derived randomly by selecting NS non-repetitive numbers within the interval [1,NE].
3) The indexes of selected entries according to the number RN in SN can be derived by (14) and (15)
4) The selected entries the of experimental sparse matrix are filled with the QoS values contained in the corresponding entries of the answer matrix.

$$index_{service} = (RN \div columns\ of\ the\ matrix) + 1 \quad (14)$$

$$index_{user} = RN\ \%\ rows\ of\ the\ matrix \quad (15)$$

To compare the accuracy of our approach with other approaches, we implemented a sequence of methods based on 1) user-based CF approach using PCC (UPCC) [22], 2) item-based CF approach using PCC (IPCC) [25], 3) hybrid PCC (HPCC) [26], 4) user-mean (UMEAN), 5) item-mean (IMEAN), 6) matrix factorization (MF) [3] and 7) our trust-aware QoS prediction approach to predict the QoS values of services. MAE values were calculated based on the predicted QoS values and QoS values from the answer matrix. To study the accuracy under different densities, experimental sparse matrixes of 5%, 10%, and 15% were adopted as the training matrices. To minimize errors, each approach was looped 50 times for 10 randomly selected users, and the average MAE was calculated. Table I summarizes the contributions of our work:

1) Our QoS prediction approach obtains better prediction accuracy in terms of reliability values and response time for densities of 5%-15%. However, MAE values of reliability are accurate and so close in these applied methods because the calculation of service reliability are really high in our dataset.
2) The prediction accuracy in terms of throughput was also better than that of other methods under 5% density. However, when the number of invocation records is large (i.e., density is 10% or 15%), the matrix factorization [3] and hybrid PCC [26] approaches obtained better prediction accuracy for QoS attributes with values close to each other for different users (e.g., throughput).
3) A few previous studies [3][26] dealt with the prediction of one or two distinct QoS attributes. However, our approach is suitable for three QoS attributes.

B. Impacts Observed for Interval Factors

Table I also shows that the MAE values of each QoS attribute are in a distinct range in our approach (e.g., the MAE values of reliability and response time are 0.035-0.0371 and 794-1363, respectively). Observation on line charts with the MAE values in a graph is difficult because the range differences are large. We adopted Normalization Mean Absolute Error (NMAE) [2] to depict the line chars for studying the influence of number of invoked services, k, and opinion threshold, as follows [2]:

$$NMAE = \frac{MAE}{\sum_{S \in PS_i} \frac{\hat{E}_{i,s}}{|PS_i|}} \quad (16)$$

The number of services invoked by a user may change the prediction accuracy in our approach. To study this effect, the number of services was varied from 5 to 50, and the data were incremented by 5. Figure 9 shows the NMAE values of response time, throughput, and reliability in our approach. Their values declined from 0.449 to 0.288, 0.24 to 0.01, and 0.020 to 0.015, respectively, thus indicating that our approach can be improved when a (designated) user invokes a greater number of services.

To study the influence of eigenvalue k on the prediction accuracy, k was varied from 2 to 20. Figure 10 shows that the NMAE value of our approach is the smallest when k was 12 or 14. This indicates that the predicted QoS values were more accurate when eigenvalue k was within a specific range.

To study the influence of opinion threshold ($OThreshold$) on the prediction accuracy, the data were detected in the opinion threshold at increments of 0.1 from 0.1 to 0.9. Figure 11 shows that the NMAE value obtained using our approach is the smallest when the opinion threshold is 0.2-0.4. This indicates that the predicted QoS values were more accurate when the opinion threshold was within a specific range.

 ISBN: 978-1-61208-387-2

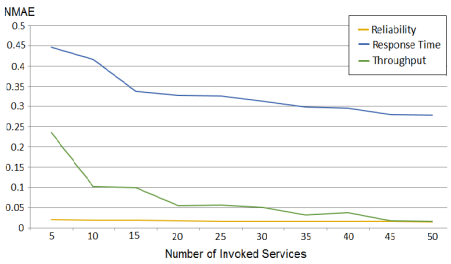


Figure 9. Influence of invoked services.

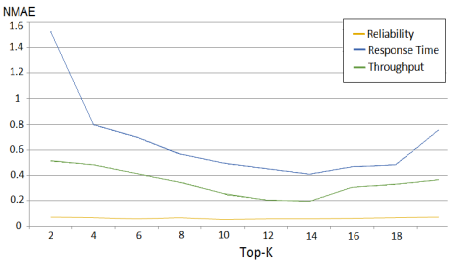


Figure 10. Influence of eigenvalue k.

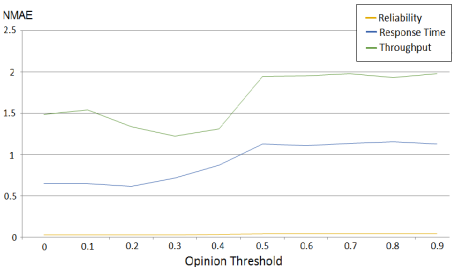


Figure 11. Influence of opinion threshold.

V. Conclusion and Future Work

In this paper, we present a trust-aware approach to predict QoS values of services more accurately. In our approach, the opinions of selected users are used to improve PCC values. These opinions are selected based on three factors, namely, belief, disbelief, and uncertainty, of the designated users who share a greater number of common services. Moreover, we introduce the indirect similar property to help select users for cold-start services. The experiments indicate that our approach provides better prediction for service selection.

There are at least two issues that warrant further study: 1) Only three QoS attributes were adopted in our approach. The accuracy of QoS values might be improved by applying a greater number of QoS attributes. 2) Indirect similarity might improve with the use of additional features such as user location, reputation of provider, and user preferences.

References

[1] Z. Liang, H. Zou, J. Guo, F. Yang, and R. Lin, "Selecting web service for multi-user based on multi-qos prediction," in Services Computing, 2013 IEEE International Conference on, June 2013, pp. 551–558.

[2] J. Wu, L. Chen, Y. Feng, Z. Zheng, M. C. Zhou, and Z. Wu, "Predicting quality of service for selection by neighborhood-based collaborative filtering," Systems, Man, and Cybernetics: Systems, IEEE Transactions on, vol. 43, no. 2, March 2013, pp. 428–439.

[3] Z. Zheng, H. Ma, M. Lyu, and I. King, "Collaborative web service qos prediction via neighborhood integrated matrix factorization," Services Computing, IEEE Transactions on, July 2013, pp. 289–299.

[4] Z. Zheng and M. Lyu, "Collaborative reliability prediction of service-oriented systems," in Software Engineering, 2010 ACM/IEEE 32nd International Conference on, vol. 1, May 2010, pp. 35–44.

[5] Z. Liu, Z. Liu, and T. Lu, "A location and time related web service distributed selection approach for composition," in Grid and Cooperative Computing, 9th International Conference on, Nov 2010, pp. 296–301.

[6] G. Pitsilis and S. J. Knapskog, "Social trust as a solution to address sparsity-inherent problems of recommender systems," Recommender Systems and the Social Web, 2009, pp. 33–40.

[7] A. JÃÿsang, "Reliability analysis with uncertain probabilities," in Proceedings of the 4th International Conference on Probabilistic Safety Assessment and Management (PSAM4). Springer, Heidelberg, 1998.

[8] A. Jøsang, "A logic for uncertain probabilities," Int. J. Uncertain. Fuzziness Knowl.-Based Syst., vol. 9, no. 3, Jun. 2001, pp. 279–311. [Online]. Available: http://dl.acm.org/citation.cfm?id=565980.565981

[9] Z. Zheng, Y. Zhang, and M. Lyu, "Investigating qos of real-world web services," Services Computing, IEEE Transactions on, vol. 7, no. 1, Jan 2014, pp. 32–39.

[10] R. Burke, "Hybrid recommender systems: Survey and experiments," User Modeling and User-Adapted Interaction, vol. 12, no. 4, Nov. 2002, pp. 331–370.

[11] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, ser. CSCW '94. ACM, 1994, pp. 175–186.

[12] J. Zhu, Y. Kang, Z. Zheng, and M. Lyu, "Wsp: A network coordinate based web service positioning framework for response time prediction," in Web Services (ICWS), 2012 IEEE 19th International Conference on, June 2012, pp. 90–97.

[13] G. Pitsilis and S. J. Knapskog, "Social trust as a solution to address sparsity-inherent problems of recommender systems," in Proceedings of 2009 ACM Conference on Recommender Systems, 2009, pp. 33–40.

[14] K. Chard, S. Caton, O. Rana, and K. Bubendorfer, "Social cloud: Cloud computing in social networks," in Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, July 2010, pp. 99–106.

[15] P. Singla and M. Richardson, "Yes, there is a correlation: - from social networks to personal behavior on the web," in Proceedings of the 17th International Conference on World Wide Web, ser. WWW '08. New York, NY, USA: ACM, 2008, pp. 655–664. [Online]. Available: http://doi.acm.org/10.1145/1367497.1367586

[16] G. Pitsilis and L. Marshall, "Modeling trust for recommender systems using similarity metrics," in Trust Management II, ser. The International Federation for Information Processing, Y. Karabulut, J. Mitchell, P. Herrmann, and C. Jensen, Eds. Springer, 2008, vol. 263, pp. 103–118.

[17] J. O'Donovan and B. Smyth, "Trust in recommender systems," in Proceedings of the 10th International Conference on Intelligent User Interfaces, ser. IUI '05. New York, NY, USA: ACM, 2005, pp. 167–174. [Online]. Available: http://doi.acm.org/10.1145/1040830.1040870

[18] A. Jøsang, R. Hayward, and S. Pope, "Trust network analysis with subjective logic," in Proceedings of the 29th Australasian Computer Science Conference - Volume 48, ser. ACSC '06. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2006, pp. 85–94. [Online]. Available: http://dl.acm.org/citation.cfm?id=1151699.1151710

[19] A. Josang and T. Bhuiyan, "Optimal trust network analysis with subjective logic," in Emerging Security Information, Systems and Technologies, 2008. SECURWARE '08. Second International Conference on, Aug 2008, pp. 179–184.

[20] Y. Zhang, Z. Zheng, and M. Lyu, "Wspred: A time-aware personalized qos prediction framework for web services," in Software Reliability Engineering, IEEE International Symposium, Nov 2011, pp. 210–219.

[21] P. Resnick and H. R. Varian, "Recommender systems," Commun. ACM, vol. 40, no. 3, Mar. 1997, pp. 56–58. [Online]. Available: http://doi.acm.org/10.1145/245108.245121

[22] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, ser. UAI'98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 43–52. [Online]. Available: http://dl.acm.org/citation.cfm?id=2074094.2074100

[23] Z. Zheng, Y. Zhang, and M. Lyu, "Distributed qos evaluation for real-world web services," in Web Services (ICWS), 2010 IEEE International Conference on, July 2010, pp. 83–90.

[24] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "Planetlab: An overlay testbed for broad-coverage services," SIGCOMM Comput. Commun. Rev., vol. 33, no. 3, Jul. 2003, pp. 3–12. [Online]. Available: http://doi.acm.org/10.1145/956993.956995

[25] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in Proceedings of the 10th International Conference on World Wide Web, ser. WWW '01. New York, NY, USA: ACM, 2001, pp. 285–295. [Online]. Available: http://doi.acm.org/10.1145/371920.372071

[26] Z. Zheng, H. Ma, M. Lyu, and I. King, "Wsrec: A collaborative filtering based web service recommender system," in Web Services, 2009. ICWS 2009. IEEE International Conference on, July 2009, pp. 437–444.

 ISBN: 978-1-61208-387-2

Finding Optimal REST Service Oracle Based on Hierarchical REST Chart

Li Li, Wu Chou
Shannon IT Lab
Huawei
Bridgewater, New Jersey, USA
{li.nj.li, wu.chou}@huawei.com

***Abstract*—Based on the hypertext-driven nature of REST API, this paper presents a structural approach for REST client design and implementation, in which a REST client is decomposed into two reusable functional modules: a client oracle that selects hyperlinks to follow for a given goal, and a client agent that carries out the interaction as instructed by the oracle. This decomposition has several advantages over a monolithic REST client where the two functions are intertwined and inseparable. To automatically find an optimal client oracle from a machine-readable description of a REST API, we introduce the path selection framework and apply Dijkstra's Shortest Path algorithm to Hierarchical REST Chart, which is an enhancement and extension to the original REST Chart that describes REST API based on Colored Petri-Net. The proposed method has been implemented in Java and tested on two sets of Hierarchical REST Charts. Experimental results indicate that the proposed approach is effective and promising.**

Keywords—REST API; Hierarchical REST Chart; REST Oracle; Petri-Net; Shortest Path

I. INTRODUCTION

In recent years, the REST architectural style [1] has become increasingly popular, and it has been applied widely to API designs in various areas, including Real-Time Communications [2][3], Cloud Computing [4], and Software-Defined Networking [5]. It provides an efficient and flexible way to access and integrate large-scale complex systems and distributed applications. REST is based on the principle that *any client of a REST API should be driven by nothing but hypertext*. This principle seems abstract but it is easy to understand if we treat a REST API as a distributed finite state machine, where the states are resource representations and the transitions are the links between the representations. In this model, hypertext-driven means that a client should enter a REST API from an entry point, and then be guided by the hypertext from the resources to reach a final representation.

This principle makes it possible for a user without any technical background to use the Web by following the hyperlinks on the pages until the desired page is retrieved. In this process, the user decides which links to follow based on the information on the page, and the user agent carries out these decisions by interacting with the resources identified by the links. This separation of users from the user agents make both of them "reusable" in the sense that a user can use any user agent, and a user agent can use any user, to navigate the Web.

To mirror this process for a REST client that navigates a REST API without user involvement, it would be beneficial to decompose the REST client into two functional components: a client oracle responsible to select links to follow from the resource representations, and a client agent responsible to interact with the resources based on the selected links. This decomposition of a REST client has several advantages over a monolithic REST client where these two functions are intertwined and inseparable:

- a client oracle can be reused with different versions of a REST API, especially if a REST API version update changes some resource representations and identifications, but does not change the link relations in the representations;
- a client oracle can be reused across different service description languages of the same REST API;
- a client oracle can drive client agents in different programming languages to achieve consistent behavior;
- a client agent can be driven by different client oracle to accomplish different tasks;
- a client oracle can significantly reduce the size of a client agent if only a small portion of the resource representations are selected by the client oracle.

To realize these benefits for a given REST API, a client oracle and client agent can be written by developers manually. But this can be difficult, time consuming, and error prone. A better approach is to generate a REST oracle and agent automatically from a machine-readable service description of a REST API, such as REST Chart [6]. If the manual programming process becomes unnecessary or greatly reduced, we can significantly speed up the REST client development process. To tackle REST client generation in two phases, this paper describes a method to find optimal client oracles based on the Hierarchical REST Chart, an extension to REST Chart [6], and client agent generation will be our future work.

The rest of this paper is organized as follows. Section II reviews related work in REST service description languages. Section III describes the Hierarchical REST Chart. Section IV introduces the optimal REST oracle framework. Section V discusses the implementation and experimental results, and our findings are concluded with Section VI.

 ISBN: 978-1-61208-387-2

II. RELATED WORK

Since 2009, several new service description languages, including WADL [7], RAML [8], Swagger [9], RSDL [10], API-Blueprint [11], SA-REST [12], ReLL [13], REST Chart [6], RADL [14], and RDF-REST [15] have been developed independently for REST API, but none of them is yet standardized. All these description languages are encoded in some machine-readable languages, such as XML, and most of them are standalone documents, except a few of them, such as SA-REST, are intended to be embedded within a host language, such as HTML.

RAML is a YAML language that organizes a REST API as a tree rooted at a base URI (template or reference) that denotes a REST API entry point. Underneath the root are a set of URI (templates or references) that identify available resources. Each URI may be associated with the access methods that define the input and output representations. While RAML offers a minimalist structure and several interesting mechanisms, such as inline documentation, resource traits and types, it could lead to inadvertent violation of the REST constraints [6] by exposing a list of fixed resource locations. Also, RAML does not seem to have a way to tie the hyperlinks in hypertext representations with the URI templates in the REST API description tree. Without these ties, it would be difficult for a REST client to know the method to access a hyperlink and the response representation.

Swagger has bindings to both YAML and JSON, and its REST API descriptive structure is very similar to RAML, except using a different set of vocabularies. For this reason, it has the same problems as RAML.

RSDL is a XML language that organizes a REST API around a list of <resource> elements, each of which may contain elements <location> that define its URI, <link> that links it to other resources, and <method> that define the access. As a result, RSDL could also inadvertently violate the REST constraints [6] by exposing a fixed set of resource locations to the clients. Moreover, there are no ties between the <link> elements, the <method> elements, and the actual resource representations, such that a hyperlink in a representation can point to its access method and response representation.

RADL is a XML language that organizes a REST API around the <resource> element that defines the resource location in the child element <uri> and resource methods in the child <interface> element. The request and response of a method are defined by <document> elements, which may contain <link> elements pointing to other <document> elements. Like RSDL, this resource centric design could lead to fixed resource locations. Moreover, even if a client knows the interface of a resource, it may not know the method in the interface to access a hyperlink of a document, if two or more methods exist in the interface.

Some open source toolkits are available for some service description languages [8][9][11] to generate client and server skeleton code in Java and node.js, such that the developers can edit the generated source code to complete the implementation. However, when generating the client code, these toolkits do not separate client oracle and client agent, as far as we know.

In addition, none of these service description languages supports nested REST API descriptions, such that a complete REST API description can be incrementally refined or composed seamlessly with the same mechanism. Breaking a large service description file into small parts can be helpful but it is still not sufficient, since the large service description is incomplete without the parts.

III. HIERARCHICAL REST CHART

We adopt the REST Chart model [6] as the basis for finding optimal client oracles. A major feature of REST Chart is the ability to combine the static aspects of a REST API, e.g. media types and link relations, with the dynamics of the REST API, e.g. the hypertext driven client-server interactions, into one coherent model. The REST Chart models a REST API as a Colored Petri Net where the places are "colored" by types. A typed place denotes a media type schema that defines valid resource representations. A transition denotes a valid resource interaction following a hyperlink in a schema. A token in a typed place denotes a valid resource representation defined by the schema. In this model, the connected schemas collectively define the hypertext media types of a REST API without creating any out-of-band dependences to the resource organization of the REST API.

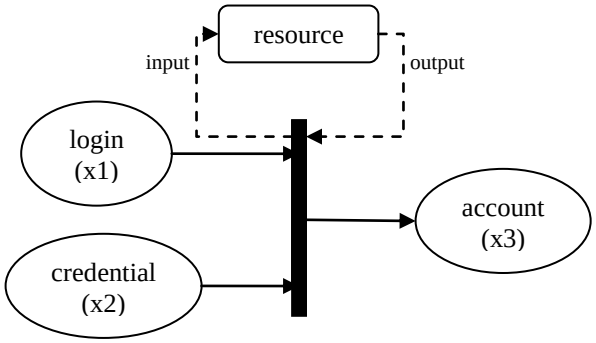


Figure 1. Example of a basic REST Chart

A basic REST Chart defines the contract between the server and client for a single interaction. This contract consists of two server places and a client place connected by a transition, as illustrated in Figure 1. This REST Chart indicates that a client can transfer its representational state from the login place (server place) to the account place (server place) if the client can create a token for the credential place (client place). To make the transition, the server first puts a token x1 in the login place (i.e. returns a valid login page), to provide a login hyperlink to the client. Then the client selects that link and puts a token x2 in the credential place (i.e. enters valid username and password). At this point, the transition (solid bar) fires and the user credential is sent to the login resource identified by the hyperlink. On success, the server deposits a token x3 in the account place (i.e. returns the valid account information).

 ISBN: 978-1-61208-387-2

These token markings capture the essential interaction procedure as sanctioned by the REST Chart.

The resource involved in the interaction is identified by a URI template, and there is no fixed resource location, relation, or interface that could lead to violations of the REST constraints R3-R5 [6]. A REST API with more than one interaction can be described by connecting appropriate places to form a large Petri-Net with a single entry place, which is the designated entry point of the REST API.

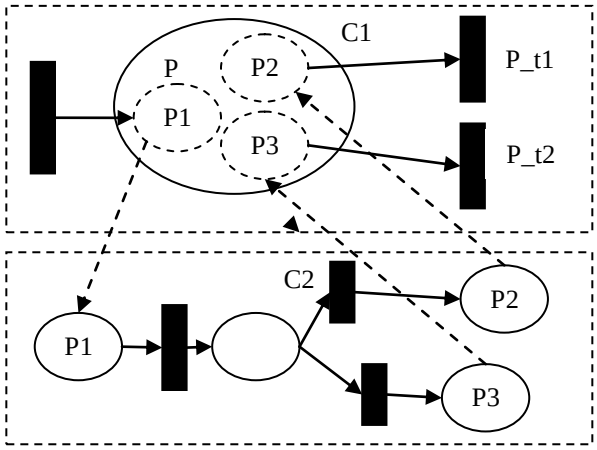


Figure 2. Hierarchical REST Chart C1 that nests REST Chart C2

```
<rest_chart id=”C1”>
  …
  <representation id="P" href="URI_to_C2" />
  <transition id="P_t1">
    <input>
      <representation ref="P/P2" link="P2_k1" />
    </input>
    …
  </transition>
  <transition id="P_t2">
    <input>
      <representation ref="P/P3" link="P3_k1" />
    </input>
    …
  </transition>
  …
</rest_chart>
```

Figure 3. XML of REST Chart C1 that nests C2 by its interface

```
<rest_chart id="C2">
  <representation id="P1" initial="true">…</representation>
  …internal topology…
  <representation id="P2">…</representation>
  <representation id="P3">…</representation>
</rest_chart>
```

Figure 4. XML of REST Chart C2’s interface places

To promote reusability and modularization of REST API, this paper extends the REST Chart to Hierarchical REST Chart based on Hierarchical Petri-Net. In Hierarchical REST Chart, a typed place can contain another REST Chart, as shown in Figure 2 where place P of REST Chart C1 contains REST Chart C2. The REST Charts C1 and C2 communicate as follows: 1) when place P receives a token of type P1, it is moved to place P1 of C2; 2) C2 will fire as usual; 3) when place P2 or P3 has a token, then the token is moved back to place P; 4) C1 will continue to fire as usual. In general, Hierarchical REST Chart can be nested to any number of levels.

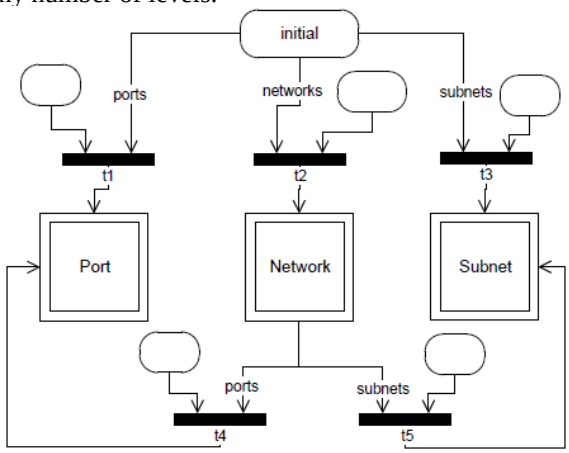


Figure 5. Top-level REST Chart for the network REST API with nested representations in double framed boxes

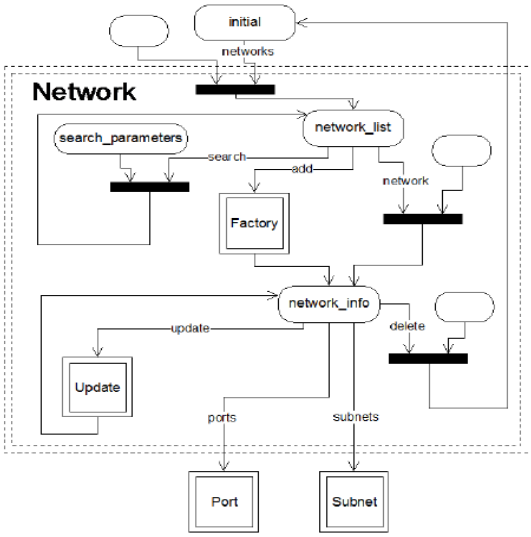


Figure 6. REST Chart nested inside at the Network place of the top-level REST Chart

To represent Hierarchical REST Chart, the XML of C1 is modified but there is no modification to the XML of C2. In C1, we modify place P and its outgoing transitions in REST Chart C1 to point at C2. The relevant modifications to C1 XML are illustrated in Figure 3 in bold font, where the <representation> element has an attribute “href” that points to the location of REST Chart C2, and the two <transition> elements “P_t1” and “P_t2” use notations “P/P2” and “P/P3” to reference the places P2 and P3 respectively in C2 that is nested in place P. The places P1, P2, and P3 of Chart REST C2 act as its interface to hide the topology of C2 as shown in Figure 4, such that the internal changes in C2 will not impact C1.

 ISBN: 978-1-61208-387-2

We have successfully applied Hierarchical REST Chart to describe several practical REST APIs, including the Network Management REST API of OpenStack [16]. Figure 5 depicts a top-level REST Chart with three nesting representations: Port, Network and Subnet, and the nested REST Chart for the Network is depicted in Figure 6, whose interface place Port and Subnet are used by C1 in transition t4 and t5 respectively. In both figures, the empty places indicate the client requests to dereference the hyperlinks.

IV. OPTIMAL REST ORACLE

REST Oracle is based on the idea that a REST client can be divided into two reusable functional modules: a client oracle that decides the resources to interact with, and a client agent that carries out the resource interactions instructed by the client oracle. The decisions made by a client oracle obviously depend on what goal the client is trying to achieve with the REST API. For example, a client oracle that tries to check bank account balance probably should select different resources than a client oracle that tries to deposit a paper check to an account.

If a REST API is described by Hierarchical REST Chart, the goal of a client can be defined in terms of the places it needs to visit. For instance, when a client wants to deposit a check to an account, it must reach these places in the correct order: login place to authenticate itself, the place to deposit a check, the place to scan the check, the place to upload the check, the place to verify the information, and the place for the positive acknowledgement of the entire process.

If there is more than one sequence of places to reach a goal, as most REST API does, a REST client needs to consider which path is optimal. For this purpose, we map each transition in a Hierarchical REST Chart to a positive real number that represents the cost for a REST client to take that transition. The cost can be related to network latency, the message size, the processing time, or a combination of such factors that can be measured based on the environment in which the REST API operates. With the cost factor, an optimal path can be defined as the shortest path from the initial place to a goal place in a Hierarchical REST Chart. A uniform cost of 1.0 at every transition would produce an optimal oracle that takes the smallest number of messages to reach the goal place.

It is possible to find such shortest paths in REST Chart based on Petri-Net reachability algorithms [18] or coverability algorithms [19]. However, these algorithms are usually complex or may take exponential space as they compute all possible token markings in arbitrary Petri-Net. For this reason, we decide to use graph search algorithm, whose time complexity is not dependent on token markings and is polynomial to the number of places and transitions of a Petri-Net.

To apply this approach, we convert the Hierarchical REST Chart to a nested directed graph. The server places become the vertices, and each edge is labeled with corresponding transition including the client place and a cost as illustrated in Figure 7. This process is recursively applied to the nested REST Charts.

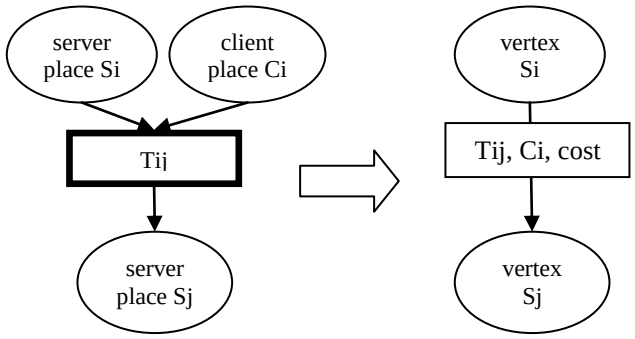


Figure 7. Transforming a REST Chart to a directed graph with cost

It is often useful to direct a REST client to not just one, but a series of goal places when interacting with a workflow. To realize this requirement, we adapt Dijkstra's Shortest Path algorithm [17] to the nested directed graph to find the shortest path from the initial place to the first goal place, and then repeat the algorithm from current goal place to the next goal place in the series, until the last goal place is reached. If a goal place is not reachable, then the process will stop. The shortest path found by this process is an oracle that reaches these goals in the given order.

```
1.  Client_Oracle(C, A, Pi, Pj): Oracle
2.    C: REST Chart
3.    A: adjacency matrix for C
4.    Pi: source place
5.    Pj: target place
6.    IN = {Pi}
7.    For each Pk in P of C do d[Pk] = A[Pi,Pk]
8.    While Pj not in IN do
9.      Pk = a place X in P–IN with minimum d[X]
10.     s[Pk] = C.transition(Pi)
11.     IN += Pk
12.     For each place X in P–IN do
13.       dist = d[X]
14.       d[X] = min(d[X], d[Pk] + A[Pk, X])
15.       if (d[X] < dist) then s[X] = C.transition(Pk)
16.     End
17.   End
18.   T = s[Pj]
19.   Oracle = (C.server_place(T), T, C.client_place(Y))
20.   While C.server_place(T) ≠ Pi do
21.     T = s[T]
22.     Oracle += (C.server_place(T), T, C.client_place(T))
23.   End
24.   Return reverse(Oracle)
25. End
```

Figure 8. Client Oracle algorithm adapted from Dijkstra's Shortest Path Algorithm

The Client Oracle algorithm is outlined in Figure 8. The core of the algorithm (lines 2-17) uses Dijkstra's Shortest Path algorithm on the directed graph A, which is converted from the REST Chart C, to find a shortest path from an initial place Pi to a final place Pj and record the transitions on the path (e.g. s[X] = transition(Pk)). The rest of the algorithm (lines 18-24) reconstructs from the recorded

 ISBN: 978-1-61208-387-2

transitions of the oracle as a sequence of triples (Server_Place, Transition, Client_Place).

If a vertex Si contains a nested directed graph, then we will find the shortest paths from the initial vertex of the nested graph to all its final vertices, and add the total cost of each shortest path to the cost of the corresponding edge from Si. For example in Figure 2, the cost of the shortest path from P1 to P2 will be added to edge P_t1 of P, and the cost of the shortest path from P1 to P3 will be added to edge P_t2 of P.

The following diagram (Figure 9) illustrates 3 shortest paths (oracles) superimposed on an automated coffee service REST Chart with uniform cost 1 labeled on the edges, and the corresponding oracles are summarized in Table I.

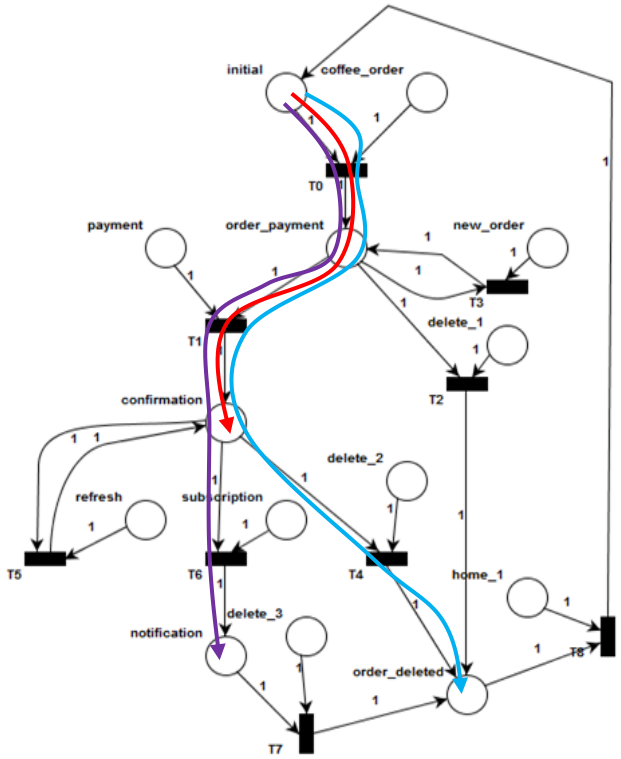


Figure 9. Three client oracles (shortest paths) for three goal series

TABLE I. ORACLES FOUND FOR DIFFERENT GOALS

	Goal Series	*Oracle*
1	**{confirmation}**	**(initial, T0, coffee_order) (order_payment, T1, payment)**
2	**{confirmation, order_deleted}**	**(initial, T0, coffee_order) (order_payment, T1, payment) (confirmation, T4, delete_2)**
3	**{notification}**	**(initial, T0, coffee_order) (order_payment, T1, payment) (confirmation, T6, subscription)**

The oracle triples in Table I contain the crucial information to implement a fully functional oracle program. The server place in a triple specifies the representations that a REST client must understand to select a hyperlink. The transition in a triple specifies the interaction with the selected hyperlink. The client place specifies the representations, e.g. a form definition, that the REST client must supply for that interaction. The only missing information is the actual representation, e.g. the form data, for the client place, which can be saved statically with the client oracle, or input to the client oracle dynamically when it is needed. Alternative optimal paths to a goal can also be included in a client oracle for fail over or load balancing purposes.

V. IMPLEMENTATION AND EXPERIMENTAL RESULTS

We have developed a Java implementation of the Client Oracle algorithm in Figure 8. The overall flow of the implementation is illustrated in Figure 10. Only the top-level REST Chart is need by the tool, which will automatically load any nested REST Chart. Our Java tool implementation accepts a single or a series of goals. In addition to finding a client oracle for a given set of goals, it can also find all the client oracles from the entry place to all other places in a Hierarchical REST Chart. The output of the tool includes the oracles found for the top-level and the nested REST Charts.

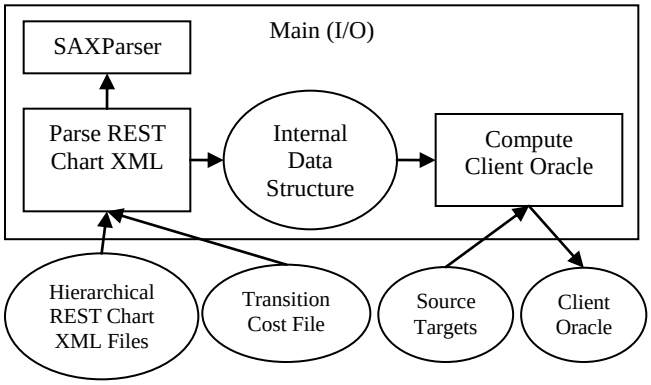


Figure 10. Java implementation of Client Oracle

We ran our Java tool on two sets of Hierarchical REST Charts with randomly generated costs to find all possible client oracles from the initial place. The first set contains 4 Hierarchical REST Charts nested in three levels, where the numbers in the parentheses indicate the number of places and transitions in the REST Chart:

```
Chart1 (8,7)
    Chart3 (4,3)
    Chart2 (6,5)
          Chart4 (4,3)
```

The second set contains 4 REST Charts nested in 3 levels as follows:

```
ABC1 (8,14)
    ABC2 (6,8)
          ABC3 (5,7)
          ABC4 (4,5)
```

The execution time (in millisecond) includes parsing the multiple XML files into the internal data structure, finding all client oracles based on the data structure, and saving them to a log file. The times measured by Java function `System.currentTimeMillis()` averaged over 5 runs on

 ISBN: 978-1-61208-387-2

a 32-bit Windows 7 machine (Intel i5 M560 dual core at 2.67 GHz and 4.00 GB memory) are summarized in Table II, where the numbers in the parentheses indicate the total number of places (V) and transitions (E) in each REST Chart.

TABLE II. PERFORMANCE SUMMARY

	average (ms)	***std***
Set 1 (22, 18)	**59**	**6.8**
Set 2 (23, 34)	**66**	**6.7**

Since the Dijkstra's Shortest Path algorithm has O(E+V*log(V)) time complexity for a graph with E edges and V vertices, the time ratio of these two sets are very close to the time complexity ratio: 66/59=1.1 while (34+23*log(23))/(18+22*log(22))=1.3. For this reason, the the performance is satisfactory and consistent. More important than the performance measurements that can be improved in many ways, the results demonstrate that the approach is feasible in finding optimal client oracles with any Hierarchical REST Chart in polynomial time.

VI. CONCLUSION

The three main contributions of this paper are: 1) a structural approach to REST client design based on two reusable functional modules, i.e. a client oracle that selects hyperlink to follow for a given goal, and a client agent that carries out the interaction as instructed by the oracle; 2) the new modeling mechanism and XML language to support Hierarchical REST Chart, which is a significant improvement over the original REST Chart for REST service modeling; and 3) a path selection framework for finding the optimal REST oracle and the implementation of the path selection framework based on Dijkstra's Shortest Path algorithm to Hierarchical REST Chart. Our approach has several advantages over a monolithic REST client design approach. Experimental results indicated that this approach is feasible and promising.

For future work, we plan to continue improving the framework of the Hierarchical REST Chart and apply the REST oracle approach in automated goal-driven generation of fully functional REST clients based on the REST Chart description of REST API.

ACKNOWLEDGMENT

The authors would like to thank Anita Kurni for implementing the Java Client Oracle tool while working as a contractor for Huawei.

REFERENCES

[1] R. T. Fielding, Architectural Styles and the Design of Network-based Software Architectures, Dissertation, University Of California, Irvine, 2000.

[2] Twilio REST API, http://www.twilio.com/docs/api, retrieved: February, 2015.

[3] GSMA OneAPI, http://www.gsma.com/oneapi/voice-call-control-restful-api/, retrieved: February, 2015.

[4] Amazon Simple Storage Service REST API, http://docs.aws.amazon.com/AmazonS3/latest/API/APIRest.html, retrieved: February, 2015.

[5] Floodlight REST API, http://www.openflowhub.org/display/floodlightcontroller/Floodlight+REST+API, retrieved: February, 2015.

[6] L. Li and W. Chou, "Design and Describe REST API without Violating REST: a Petri Net Based Approach," ICWS 2011, July 4-9, 2011, pp. 508-515.

[7] M. Hadley, Web Application Description Language, W3C member Submission, 31, August 2009, http://www.w3.org/Submission/wadl/, retrieved: February, 2015.

[8] RAML Version 0.8, http://raml.org/spec.html, retrieved: February, 2015.

[9] Swagger 2.0, https://github.com/swagger-api/swagger-spec, retrieved February, 2015.

[10] J. Robie, R. Cavicchio, R. Sinnema, and E. Wilde, "RESTful Service Description Language (RSDL), Describing RESTful Services Without Tight Coupling, Balisage," The Markup Conference 2013, http://www.balisage.net/Proceedings/vol10/html/Robie01/BalisageVol10-Robie01.html, retrieved: February, 2015.

[11] API Blueprint Format 1A revision 7, https://github.com/apiaryio/api-blueprint/blob/master/API%20Blueprint%20Specification.md, retrieved: February, 2015.

[12] K. Gomadam, A. Ranabahu, and A. Sheth, SA-REST: Semantic Annotation of Web Resources, W3C Member Submission 05 April 2010, http://www.w3.org/Submission/SA-REST/, retrieved: February, 2015.

[13] R. Alarcon and E. Wilde, "Linking Data from RESTful Services," LDOW 2010, April 27, 2010, pp 100-107.

[14] J. Robie, RESTful API Description Language (RADL), https://github.com/restful-api-description-language/RADL, 2014, retrieved: February, 2015.

[15] P.-A. Champin, "RDF-REST, A Unifying Framework for Web APIs and Linked Data," Services and Applications over Linked APIs and Data (SALAD) workshop at ESWC, May 2013, pp.10-19.

[16] OpenStack API References, http://developer.openstack.org/api-ref.html, retrieved: February, 2015.

[17] J. L. Gersting: Mathmatical Structures for Computer Science, third edition, 1993, pp. 422-423.

[18] C. G. Cassandras and S. Lafortune, Introduction to Discrete Event Processing, 2nd edition, Springer, 2008, pp. 246-246.

[19] J. Esparza and M. Nielsen: Decidability Issues for Petri Nets, BRICS Report Series, RS-94-8, ISSN 0909-0878, May 1994.

 ISBN: 978-1-61208-387-2

Robust Interactions under System Crashes and Network Failures of Collaborative Processes with Arbitrary Control Flows

Lei Wang,
Luís Ferreira Pires
and Marten J. van Sinderen

CTIT, University of Twente,
the Netherlands
Emails: {l.wang-1, l.ferreirapires, m.j.vansinderen}@utwente.nl

Andreas Wombacher

Achmea, the Netherlands
Postbus 866
3700 AW Zeist
Email: andreas.wombacher@achmea.nl

Chi-Hung Chi

CSIRO, Australia
3-4 Castray Esplanade,
Hobart, Tasmania, 7000
Email: chihungchi@gmail.com

***Abstract*—Due to the possibility of system crashes and network failures, the design of robust interactions for collaborative business processes is a challenge. If a process changes state, it sends messages to other relevant processes to inform them about this change. However, server crashes and network failures may result in a loss of messages. In this case, the state change is performed by only one process, resulting in global state/behavior inconsistencies and possibly deadlocks. Our idea to solve this problem is to (automatically) transform the original processes into their robust counterparts. We illustrate our solution using a subset of WS-BPEL. A WS-BPEL process is modeled using a so called Nested Word Automata (NWA), to which we apply our transformation solution and on which we perform correctness proof. We have also analyzed the performance of our prototype implementation. In our previous work, we assumed that a certain pre-defined interaction follows the failed interaction. In this work, we lift this limitation by allowing an arbitrary behavior to follow the failed interaction, making our solution more generally applicable.**

Keywords–robust, collaborative processes, WS-BPEL, interactions, system crash, network failure, automata

I. INTRODUCTION

The electronic collaboration of business organizations has grown significantly in the last decade. Often data interchange is based on processes run by different parties exchanging messages to synchronize their states. If a process changes state, it sends messages to other relevant processes to inform them about this change. However, server crashes and network failures may result in a loss of messages. In this case, the state change is performed by one process, resulting in global state/behavior inconsistencies and possible deadlocks. In general, a state inconsistency is not recovered by the process engine that executes the process. This can be seen from a screen dump of errors from the Oracle process engine 12c which sends message to an unavailable server (see Figure 1).

Figure 2a shows that normally, a business process is deployed to a process engine, which runs on the infrastructure services (OS, database, networks, etc.), where system crashes and network failures may happen. Our solution to recover from failures is to transform business processes into their robust counterparts, as shown in Figure 2b. The robust process is deployed on the unmodified infrastructure services and is recoverable from some interaction failures caused by system crashes and network failures. Our solution has the following properties: (1) the application protocols are not modified. We do not modify the message format nor message sequence, e.g., by adding message fields that are irrelevant for the application logic or adding acknowledge messages to the original message sequence. The service autonomy is kept in that if one party transforms the process according to our approach and the other party does not, they can still interact with each other, although without being able to recover from system crashes and network failures. (2) the process transformation is transparent for process designers. In this paper, we illustrate our solution using WS-BPEL [1]. However, other process languages may be applicable as long as they support similar workflow patterns [2].

This paper is an extension of our previous work [3][4][5][6], where we assumed that a certain pre-defined interaction follows the failed interaction, i.e., the only sequence control is assumed that the further interaction is sequentially following the failed interaction. A technical report [7] is provided as an online version with more details on the formalism used and the transformation methods of our approach. In this paper, we lift this limitation by allowing an arbitrary behavior to follow the failed interaction, making our solution more generally applicable. We support conditional control flow and loops and their arbitrary combination as possible further interaction after interaction failure. The structure of the paper is the following: Section II analyzes possible interaction

 ISBN: 978-1-61208-387-2

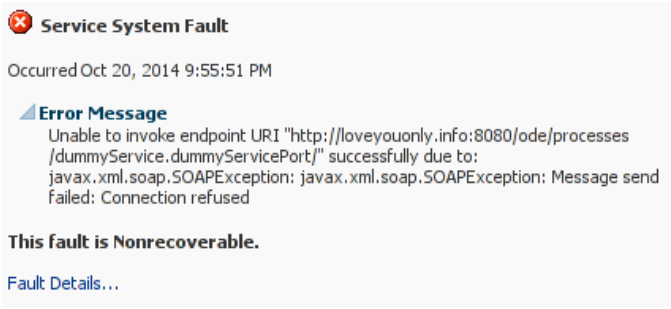
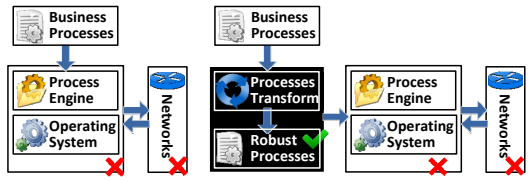
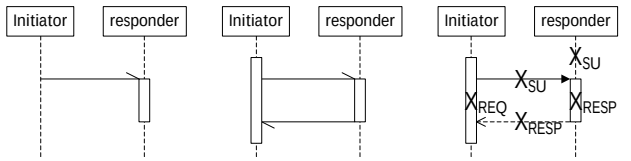


Figure 1. Oracle SOA engine interaction errors.



(a) Interaction failures (b) Robust process transformation

Figure 2. Our idea to cope with failures.



(a) *send* and *receive* (b) *send-receive* case I (c) *send-receive* case II

Figure 3. Process interaction patterns.

failures. Section III proposed our process transformation-based solution. Section IV validates our solution. Section V discusses related work and Section VI concludes our paper.

II. Interaction Failure Analysis

This section analyzes possible interaction failures of collaborative processes caused by system crashes and network failures.

A. Process Interaction Patterns

Process interaction failures are specific to interaction patterns. In [8], 13 interaction patterns are identified. In this paper, we focus on the *send*, *receive* and *send-receive* patterns. This limitation is not severe because more complex patterns can be composed using these basic interaction patterns. Figure 3a shows an initiator that sends a one-way message to a responder. The initiator behavior corresponds to the *send* pattern, while the responder behavior corresponds to the *receive* pattern. In pattern *send-receive* in Figure 3b the initiator combines one *send* and one *receive* pattern. We call this pattern asynchronous interaction in the sequel of the paper. In Figure 3c, the initiator starts a synchronous interaction by sending a request and getting a response, which characterize the *send-receive* pattern.

B. Process Interaction Failures

Interaction failures caused by system crashes and network failures are *pending request failure*, *pending response failure* and *service unavailable* [4]. As all failures possible in the interaction patterns of Figure 3a and Figure 3b are covered by Figure 3c, we look only into the interaction failures of the interaction pattern in Figure 3c. *Service unavailable* (marked as X_{SU}) is caused by a responder system crash or a network failure of the request message delivery. At process level, the initiator is aware of the failure through a catchable exception of the process implementation language. *Pending request failure* (marked as X_{REQ}) is caused by initiator system crashes after sending a request message. The initiator is informed of the failure after restart, e.g., through catchable exceptions. However, the responder is not aware of the failure, so that it replies with the response message and continues execution. *Pending response failure* (marked as X_{RESP}) is caused by a responder system crash or a network failure of the response message delivery. In both cases, the responder replies with the response message (after a restart if the responder system crashes) and continues execution. However, the connection gets lost and the initiator cannot receive the response message. The initiator is aware of this failure after a timeout.

Due to the heterogeneous infrastructure, e.g., different process engine implementations or network environments, we have to make the following assumptions concerning the failure behavior of the infrastructure: *1) Persistent execution state.* The state of a business process (e.g., values of process variables) are kept persistent and survive system crashes. *2) Atomic activity execution (e.g., invoke, receive, reply).* A system crash means that the execution is stopped only after the previous activity is finished and the next activity has not started. A restart means that execution resumes from the previous stopped activity. These assumptions correspond with the default behavior of the most popular process engines, such as Apache ODE or Oracle BPEL Process Manager (released as a component of Oracle SOA Suite). In Apache ODE's term, this is named as persistent processes in their default configuration. Otherwise this configuration can be modified to "in-memory" at deployment time [9]. For Oracle BPEL Process Manager, this is named as "durable" processes, otherwise is named as "transient" processes. By default all the WS-BPEL processes are durable processes and their instances are stored in the so called dehydration tables, which survives system crashes [10]. *3) Network Failures* interrupt the established network connections and the messages that are in transit get lost.

III. Process Transformation Based Solution

A. Business Processes

We choose WS-BPEL as process specification language in our work. A WS-BPEL process is a container where relationships to external partners, process data and handlers for various purposes and, most importantly, the activities to be executed are declared. As an OASIS standard, it is widely used by enterprises. We use Nested Word Automata (NWA) [11] to describe the underlying semantics of WS-BPEL and use them as a basis for our formal evaluation. We choose NWA because we need to model the nested structure of WS-BPEL syntax. While traditional finite state automata can be used for

 ISBN: 978-1-61208-387-2

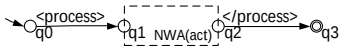


Figure 4. NWA model of a process.

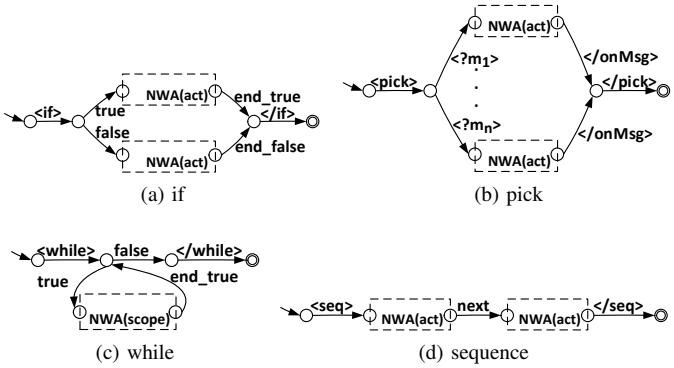


Figure 5. NWA model of WS-BPEL structured activities.

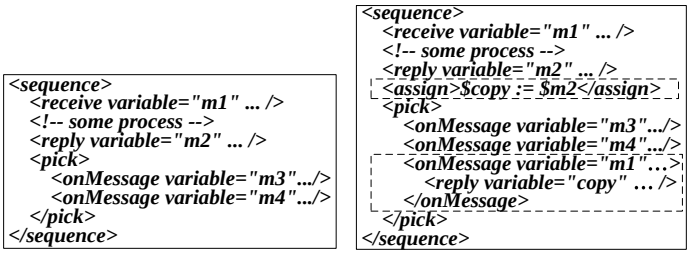


Figure 6. WS-BPEL example of our solution.

describing all possible states of messages, and their sending and receiving sequences, they lack the capability of describing nested structures of activities.

An NWA is an automaton that has hierarchical nesting structures. Formally, an NWA A over an alphabet Σ is a structure $(Q, q_0, Q_f, P, p_0, P_f, \delta_c, \delta_i, \delta_r)$ consisting of
- a finite set of (linear) states Q,
- an initial (linear) state $q_0 \in Q$,
- a set of (linear) final states $Q_f \subseteq Q$,
- a finite set of hierarchical states P,
- an initial hierarchical state $p_0 \in P$,
- a set of hierarchical final states $P_f \subseteq P$,
- a call-transition function $\delta_c : Q \times \Sigma \mapsto Q \times P$,
- an internal-transition function $\delta_i : Q \times \Sigma \mapsto Q$, and
- a return-transition function δ_r: $Q \times P \times \Sigma \mapsto Q$.

The definition of Q, q_0, Q_f, δ_i corresponds to the definition of a finite state automata over an alphabet Σ [12]. The alphabet Σ represents all possible process behaviors, e.g., $<$*process*$>$ $\in \Sigma$ represents the starting of a business process, $?m_i \in \Sigma$ represents receiving a message while $!m_j \in \Sigma$ represents sending a message. An internal transition $\delta_i(q_i, !m_i) = q_j$ represents that the process replies a message $!m_i$ at the state q_i and then enters the state q_j. The hierarchical states P, p_0, P_f are used to describe the nesting structure of an NWA. A *call transition* δ_c enters the nested automaton while a *return transition* δ_r leaves the nested automaton. A nested structure is graphically represented as dashed box. The NWA model of a WS-BPEL process is shown in Figure 4. A call transition $\delta_c(q_0, <$*process*$>) = (q_1, p_a)$ starts from the initial state and a return transition $\delta_r(q_2, p_a, <$*/process*$>) = q_3$ leads to the accepted state. The NWA model of an activity *NWA(act)* is nested within the NWA of the process. This is described by the hierarchical state p_a.

WS-BPEL activities are divided into two categories, namely *basic* and *structured* activities. The currently supported *structured* activities are *if*, *pick*, *while* and *sequence*, as shown in Figure 5. The *flow* (concurrent execution) or the other forms of loops (*RepeatUntil, ForEach*) are not considered now and will be considered in future work. Each structured activity model has exactly one call transition and one return transition to *enter* and *leave* its nested structure(s), which is represented as a dash box. The detailed explanation of our formalization is presented in [7].

B. Transformation Method

An operation that can be safely repeated is called idempotent [13]. Idempotent operations can be recovered by resending the request message. However, a request resent to non-idempotent operations (such as bank transfer operations) triggers potentially incorrect executions. Our solution for non-idempotent operation is that when a failure happens, a resent message is replied with a copy of the previous processing result.

In the example of Figure 6a, the WS-BPEL snippet receives a message $m1$, performs some (non-idempotent) processing, then replies with a message $m2$. The next incoming messages could be $m3$ or $m4$. If the initiator sends request $m3$ or $m4$, this implies that the initiator has successfully received the response message $m2$. If due to an interaction failure, for example, the initiator crashes and fails to receive the response message $m2$, the initiator can recover by resending request message $m1$. We then transform the responder process to use a copy of the previous result as response. Figure 6b shows that in order to make a copy of the response message, we use an *assign* activity to keep the result value in a process variable $\$copy$. In the *pick* activity, we add an *onMessage* branch to accept the resent message $m1$ and use the variable $\$copy$ as the response. However, a resent message could be sent multiple times before the response is ultimately received. We nest the *pick* activity in a *while* to cope with the duplicate resent message. A detailed discussion is in [7]. Our process transformation algorithm is presented as follows.

1) Responder Transformation Algorithm: For a WS-BPEL process, given its *NWA* model $(Q, q_0, Q_f, P, p_0.P_f, \delta_c, \delta_i, \delta_r)$ over the alphabet Σ, we assume that the alphabet that represents the response messages is Σ_{resp} and the alphabet that represents the request messages is Σ_{req}, thus $\Sigma_{req} \subseteq \Sigma$ and $\Sigma_{resp} \subseteq \Sigma$. The transformation algorithm is as Figure 7.

The algorithm iterates through all combinations of a state q, a request message $?m_{req}$ and a response message $!m_{resp}$. In line 2, we check if the message pair $(?m_{req}, !m_{resp})$ corresponds to the request and response for a synchronous operation and at state q, the response message $!m_{resp}$ is sent,

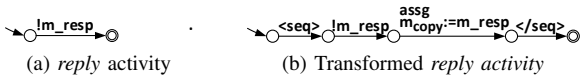
1: **for all** $q \in Q$, $?m_{req} \in \Sigma_{req}$ **and** $!m_{resp} \in \Sigma_{resp}$ **do**
2: **if** (m_{req}, m_{resp}) is a synchronous message pair **and** $\delta_i(q, !m_{resp})$ is defined in *NWA* **then**
3: $save\_reply(q, !m_{resp})$
4: N $\leftarrow next\_receive(q, !m_{resp})$
5: **for all** $?m_{next} \in N$ **and** $q_{next} \in Q$ **do**
6: **if** $\delta_i(q_{next}, ?m_{next})$ is defined in *NWA* **then**
7: $transform\_receive(q_{next}, ?m_{next})$
8: **else if** $\delta_c(q_{next}, ?m_{next})$ is defined in *NWA* **then**
9: $transform\_pick(q_{next}, ?m_{next})$
10: **end if**
11: **end for**
12: **end if**
13: **end for**

Figure 7. Responder process transformation algorithm

represented by a transition $\delta_i(q, !m_{resp})$. This is the failure point that the response message may be lost due to interaction failures and where our transformation method applies. As defined in line 3, we first make a copy of the response message, as shown in Figure 8. The NWA model of *reply* activity in Figure 8a is replaced by an NWA model of a *sequence* activity in Figure 8b, in which a *reply* activity model and an *assign* activity model are nested. The *assign* activity model represents the copy of the reply message into the variable m_{copy}. In order to process the possible resent request message $?m_{req}$ due to the lost of the message $!m_{resp}$ sent at state q, we calculate the set of all possible next incoming messages, which is defined as $next\_receive(q, !m_{resp})$ in line 4. We construct an automaton $A(!m_{resp}, ?m_{next})$ as in Figure 9 to describe that a process replies with a message $!m_{resp}$ and waits for some possible next incoming message $?m_{next}$. $\delta(q_0, !m_{resp}) = q_1$ models the reply of the response message $!m_{resp}$. $\delta(q_1, \Sigma/\Sigma_{req}) = q_1$ represents some process execution in which no messages are received. $\delta(q_1, ?m_{next}) = q_2$ represents that the process receives an incoming message $?m_{next}$. $\delta(q_2, \Sigma) = q_2$ models any process execution. For the process *NWA* model, at some state q, a reply of a message $!m_{resp}$ is represented by an internal transition $\delta_i(q, m_{resp})$. We change the initial state of the process *NWA* model to from q_0 to q, and call this automaton *NWA*(q). Starting at q, after replying the message $!m_{resp}$, if one possible next incoming message is $?m_{next}$, then *NWA*$(q) \cap A(!m_{resp}, ?m_{next}) \neq \emptyset$, i.e., the process modeled by *NWA* has the behavior described by $A(!m_{resp}, ?m_{next})$.

The intersection operation $\cap$ between an *NWA* and an finite state automaton is defined to check whether the business process modeled by the *NWA* has the message sending and receiving behavior modeled by the automaton. The intersection operation is based on finite state automata. We "flatten" an *NWA* to a finite state automaton by skipping hierarchical information, described as follows. Given a NWA $(Q, q_0, Q_f, P, p_0, P_f, \delta_c, \delta_i, \delta_r)$ over the alphabet Σ, the "flattened" automaton is $A(Q, q_0, Q_f, \Sigma, \delta)$, where Q, q_0, Q_f and Σ are the same as the NWA, the transition function δ is defined as

1) $\delta(q_{i1}, a) = q_{i2}$, if the NWA has an internal transition $\delta_i(q_{i1}, a) = q_{i2}$.
2) $\delta(q_{c1}, a) = q_{c2}$, if the NWA has a call transition $\delta_c(q_{c1}, a) = (p, q_{c2})$.



(a) *reply* activity (b) Transformed *reply activity*

Figure 8. Responder process transformation, *reply* activity.

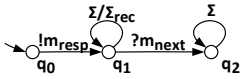
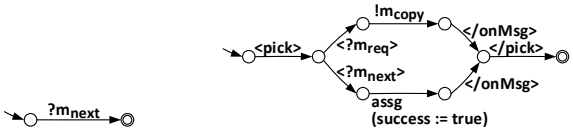
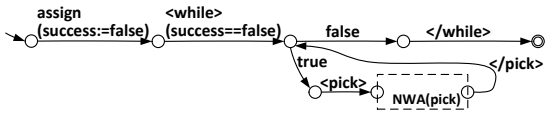


Figure 9. The automaton $A(!m_i, ?m_{next})$.



(a) Responder, *receive* activity (b) Transformed responder, part I



(c) Transformed responder, part II

Figure 10. Responder process transformation, *receive* activity.

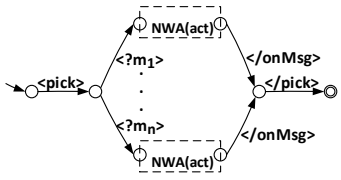
3) $\delta(q_{r1}, a) = q_{r2}$, if the NWA has a return transition $\delta_r(q_{r1}, p, a) = q_{r2}$.

Both call transitions and return transition are treated as flat transitions that the hierarchical state p is not considered. The intersection operation can be done between two finite state automata, as defined in [12].

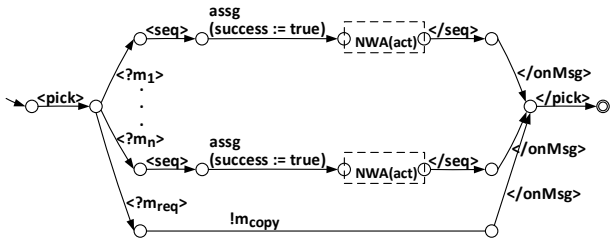
We define the set of all possible next incoming messages as $next\_receive(q, !m_{resp}) = \{?m_{next} | ?m_{next} \in \Sigma_{req} \wedge$ *NWA*$(q) \cap A(!m_{resp}, ?m_{next}) \neq \emptyset\}$.

For all $?m_{next} \in next\_receive(q, !m_{resp})$ and $q_{next} \in Q$, if at the state q_{next} the next incoming message $?m_{next}$ is received, two cases of transition may be defined in *NWA*: in a model of a *receive* activity as an internal transition $\delta_i(q_{next}, ?m_{next})$ or in the model of a *pick* activity as a call transition $\delta_c(q_{next}, ?m_{next})$. For the first case (line 6), as shown in Figure 10a, the procedure $transform\_receive(q_{next}, ?m_{next})$ is introduced as follows. We replace the transition with a *pick* activity with two branches, as shown in Figure 10b. One *onMessage* branch models the receive of the resent message $?m_{req}$ and the reply of the result message m_{copy}. The other *onMessage* branch models the receive of the message $?m_{next}$, and after that we set the flag $success$ to $true$ to indicate that the previous interaction is finished successfully. However, a possible loss of the response message m_{copy} triggers multiple resending of the request m_{req}. Therefore, the *pick* activity is defined in a *while* iteration so that multiple requests $?m_{req}$ can be accepted. Figure 10c shows that the *while* iteration ends when the flag $success$ is set to *true*.

For the second case (line 8), as shown in Figure 11a, the message $?m_{next}$ is one of the messages in $m_1, ..., m_n$. Figure



(a) Responder, *pick* activity



(b) Transformed responder, part I

Figure 11. Responder process transformation, *pick* activity.

11b shows that we then add a call transition $<?m_{req}>$ to model that the process accepts the resent message, and an internal transition $!m_{copy}$ to represent the reply using a copy of the previously cached result m_{copy}. In the other branches, the nested NWA(act) is replaced by the model of a *sequence* activity, in which we model the assignment of the flag variable *success* to *true*, followed by the original NWA(act). Similarly, in order to cope with a possible loss of the response message m_{copy}, the *pick* activity model is nested in a *while* iteration to handle multiple resent messages, as shown in Figure 10c. We do not directly prove the correntness of this algorithm, however, the correctness of the transformed processes by this algorithm have been proved in section IV.

After the transformation, at some states the responder can receive more messages than the original process, because the resent message can be accepted and be replied. However, the request is not processed again. In this sense, we do not give malicious initiators any chance of jeopardizing the process by changing the sequence of requests or sending the same request multiple times.

2) Initiator Transformation: The initiator transformation makes it resend the request message whenever a failure happens. The detailed transformation method is presented in [7].

3) Recoverable Assumption: Assume that $(?m_{req}, !m_{resp})$ is a pair of synchronous request and response messages, the process receives request message $?m_{req}$, then at state q, the process sends the response message $!m_{resp}$. However, if $?m_{req} \in next\_receive(q, m_{resp})$, then one of the next possible messages is still $?m_{req}$, in this case, the responder cannot distinguish a resent message due to a failure from a normal request message. Thus we have to require that in the process design the condition $?m_{req} \notin next\_receive(q, !m_{resp})$ can be met. However, by following a few process design principles during the design of the original process, this condition can be met. An example is, a split of message $?m_{req}$ into two different messages, $?m_{req1}$ and $?m_{req2}$ (for example, one message is used to send request, the other asks for results). The initiator sends the two messages back to back. If a responder receives $?m_{req1}$, then it waits for $?m_{req2}$, rather than waiting for $?m_{req1}$ again.

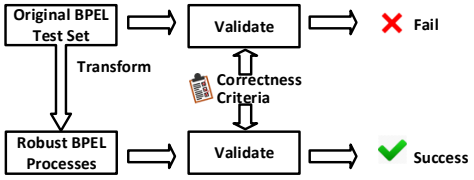


Figure 12. The setup of the correctness proof.

IV. EVALUATION

This section presents the correctness validation of our solution. We also evaluate the performance overhead of our prototype under different workloads.

A. Correctness Validation

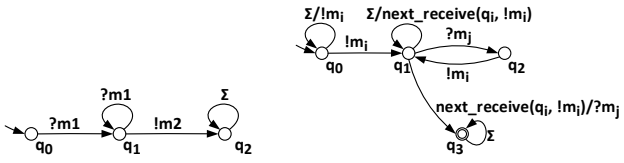
The correctness proof shows that the solution introduced in Section III provides a robust process. The core of the proof is to define the correctness criteria for asynchronous and synchronous interactions and represent them such that they can be automatically evaluated.

Figure 12 shows the setup of our correctness proof. We take the test set of 726 example WS-BPEL processes which implement all possible Internet Open Trading Protocol (IOTP) interactions [14], and we transform them into the *NWA* model of the corresponding robust business process. The proof is finished by checking that the *NWA* process model is a subset of the correctness criteria, which are modeled as automata. Given an automaton $A(Q, \Sigma, \delta, q_0, F)$, each state in Q represents a state of the messages sending and receiving status. Set Σ models of all possible process behaviors, e.g., sending and receiving messages. δ is transition function: $Q \times \Sigma \rightarrow Q$ that models the state transition triggered by process execution, e.g., for states $q_i, q_j \in Q$ and $!m \in \Sigma$, $\delta(q_i, !m) \rightarrow q_j$ represents that at state q_i, the process replies with message $!m$ and then enters state q_j. The correctness criteria automata models the set of correct message sending and receiving sequences. We present the correctness criteria as follows.

1) Initiator Side Correctness Criteria: There are two criteria due to the interaction patterns: the criteria for a single message sending and the criteria for synchronous request and response message pair. In this paper, we discuss only the latter due to the page limitations. The automaton is visualized as Figure 13a. A correct interaction is regarded as, a request message $?m1$ can be sent at state q_0 and can be resent multiple times at state q_1 until a response message $!m2$ is received at state q_2.

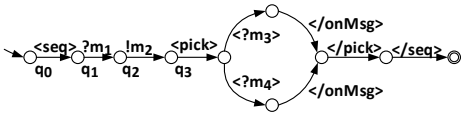
2) Responder Side Correctness Criteria: The property we want to validate is that any resent message can be accepted by the transformed process and replied. Given a process, assume that the synchronous request and response messages are m_j and m_i. If the transformed robust process model is NWA', for state q_i and transition $!m_i$, we have $?m_j \in next\_receive(q_i, !m_i)$. The idea behind it is that after the reply message $!m_i$ is sent, the robust process can accept possible resent messages due to failures. In this case, the response should be sent without reprocessing. The criteria are shown

 ISBN: 978-1-61208-387-2

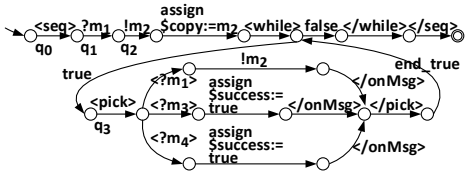


(a) Initiator (b) Responder

Figure 13. Correctness criteria of process transformation.



(a) Automaton model of the original process



(b) Automaton model of the transformed process

Figure 14. Illustration of the correctness validation, robust *NWA* model of Figure 6.

as Figure 13b. We omit the detailed discussion of this criteria due to page limitations.

The process control flows can be designed in arbitrary ways, and since we cannot exhaust all possibilities, we use a WS-BPEL test set that implements all possible IOTP interactions, which is a total of 726 BPEL processes. After the transformation of the test processes into automata, we apply the subset check to evaluate the correctness of the WS-BPEL test processes, i.e., we prove that for all processes and their *NWA* model and criteria automaton A, *NWA* $\in A$, i.e., all messages sending and receiving sequences are correct.

We take the process in Figure 6 to illustrate the correctness validation. An original WS-BPEL snippet shown in Figure 6a is transformed into the robust counterpart, and their automata models are shown as Figures 14a and 14b, respectively. At state q_2, where the message $!m_2$ is to be replied, the set of the possible next incoming messages of the responder is $next\_receive(q_2, !m2) = \{?m1, ?m3, ?m4\}$. The criteria for the synchronous request $?m1$ and the response $!m2$ is shown as Figure 15. First, we do a subset check to prove that the original is not a subset of the criteria. Actually, we can see that the message sequence $(.., ?m1, !m2, .., ?m1, !m2, ..?m3, ..)$ can be accepted by the criteria automaton. However, this sequence cannot be accepted by the model of the original process, since there is no transition defined for the second $?m1$. Second, we do a subset check to prove the transformed automata model is a subset of the criteria, i.e., all sending and receiving message sequences are correct.

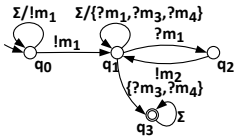


Figure 15. Correctness criteria for the illustrative process.

Figure 16. Setup of Performance Test.

TABLE I. PERFORMANCE OVERHEAD.

Origin	Trans	Overhead	Origin	Trans	Overhead
Workload $\lambda = 5$			Workload $\lambda = 10$		
287 ms	379 ms	92 ms	322 ms	452 ms	130 ms

B. Performance Evaluation

In Figure 2 of the whole setup, if the infrastructure (OS, process engine, hardware and network configuration) is the same, performance depends mainly on the process design and the workload, i.e., $performance = Test(ProcessDesign, workload)$.

We use similar setup of our performance test with our previous performance tests [15], which is shown in Figure 16. We use the cloud infrastructure from Amazon EC2. The initiator and responder processes are deployed on two computing instances and we use a local client to collect the performance data.

We evaluated the performance overhead of our transformed process under different workloads. The number of requests sent per minute by the local client complies with a Poisson distribution with parameters $\lambda = 5$ and $\lambda = 10$ requests per minute. We used these workloads because according to our tests under the available hardware and software configurations, higher workloads would exhaust the server resources. We use the open source Apache ODE process engine where an embedded Derby [16] database is used. The Amazon EC2 instance type is t1.micro with 1 vCPU and 0.594GiB memory. Each test run lasted for 60 minutes, but only the response times in the 30 minutes in the middle of this period have been considered (steady state).

The performance data is shown as Table I. Under the workload of $\lambda = 5$, the average performance overhead of our transformation mechanism is 92 ms. Under the workload of $\lambda = 10$, the average overhead is 130 ms. We conclude then that the performance overhead increases with the workload. However, we expect lower performance overhead when the infrastructure is scalable, like in a cloud environment.

 ISBN: 978-1-61208-387-2

V. Related Work

Solutions based on exception handling [17][18] is process-specific. WS-BPEL supports compensations of well-defined exceptions using exception handlers. However, elaborate process handler design requires process-specific knowledge of failure types and their related recover strategies. Alternatively, we try to ease the process designers from dealing with synchronization failures by a transparent process transformation from a given business process to its recovery-enabled counterpart.

A fault tolerant system can be built by coping with the occurrence of failures by applying redundancy [13]. Three kinds of redundancy are possible: information redundancy, time redundancy and physical redundancy. However, the existing solution either require more effort of the business process designers, or additional infrastructure support, or both.

On physical layer, the robust solutions on process engine level [19][20] dependend on a specific process engine. We defined our solution based on the WS-BPEL building blocks without requiring extensions at the engine level. However, the transformed process can still be migrated to other standard process engines. Reliable network protocols such as HTTPR have been proposed to provide reliable synchronization. However, the deployment of these solutions increases the complexity of the network infrastructure. We assume that system crashes and network failures are rare events, thus extending the infrastructure may introduce too much overhead. Further, the solutions are not applicable in some outsourced deployment environments. For example, in some cloud computing environments, user-specific infrastructure configuration to enhance synchronization is not possible. Dynamic service substitution [21][22] is a way to perform recovery by replacing the target services by equivalent services. In [23][24], the QoS aspects of dynamic service substitution are considered. In our work, we do not change the business partners at runtime.

Information redundancy recovery is based on replication. Our cache-based process transformation is information redundant because a cache is a kind of replication. Time redundancy solutions include web services transactions. The WS-AT [25] standard specifies atomic web services transactions, while WS-BA [26] standard specifies relaxed transactions so that the participant can choose to leave the transaction before it commits. However, if a transaction rolls back, a process-specific compensation is required. Actually, transactions can deal with well-defined failures. The 2-phase commit distributed transaction protocol can not deal with system crash (referred to as *cite failure* in [27]). However, in a special case of process in which all participants send vote results to a coordinator, if the coordinator crashes before sending the vote results to any participant, all the participants are blocked and the final results of the transaction remain unknown.

VI. Conclusion and Future Work

In this paper, we have identified three types of interaction failures caused by system crashes and network failures and we have proposed a process interaction failure recovery method to cope with system crashes and network failures. This paper is an extension of our previous work [3][4][5][6], where we assumed that a certain pre-defined interaction follows the failed interaction. In this paper, we lift this limitation by allowing an arbitrary behavior to follow the failed interaction, making our solution more generally applicable. The challenge is to accept the resent message due to failures with the arbitrary control flow of the responder process. We transformed the business process design into nested word automata model. At a state that models the reception of an incoming message, we add an additional transition to accept the resent message due to failure. We have proved the correctness of our process transformations and we implemented a prototype to test the runtime performance of our method. Currently, the transformation process is semi-automatic. We have implemented the automatic transformation from a WS-BPEL process to the NWA model, however, the transformation of the NWA model to the robust counter part is manually. In future, we will automate the transformation process and we will investigate more complex process interaction patterns.

References

[1] OASIS, Web Services Business Process Execution Language, 2nd ed., OASIS, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html, [retrieved: Feb., 2015], Apr. 2007.

[2] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros, "Workflow patterns," Distributed and Parallel Databases, vol. 14, no. 1, Jul. 2003, pp. 5–51.

[3] L. Wang, A. Wombacher, L. Ferreira Pires, M. J. van Sinderen, and C.-H. Chi, "An illustrative recovery approach for stateful interaction failure of orchestrated processes," in IEEE 16th EDOC Workshops, 2012, pp. 38–41.

[4] ——, "A state synchronization mechanism for orchestrated processes," in IEEE 16th Intl. EDOC Conf., 2012, pp. 51–60.

[5] ——, "Robust client/server shared state interactions of collaborative process with system crash and network failures," in 10th IEEE Intl. Conf. on Services Computing (SCC), 2013.

[6] ——, "Robust collaborative process interactions under system crash and network failures," Intl. J. of Business Process Integration and Management, vol. 6, no. 4, 2013, pp. 326–340.

[7] ——, "Robust interactions under system crashes and network failures of collaborative processes with arbitrary control flows," CTIT, University of Twente, Tech. Rep., 2014. [Online]. Available: http://documentations123.appspot.com/sc2015/techrep.pdf, [retrieved: Feb., 2015]

[8] A. Barros, M. Dumas, and A. Hofstede, "Service interaction patterns," in Business Process Management, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, vol. 3649, pp. 302–318.

[9] Apache ODE, "Create a process," https://ode.apache.org/creating-a-process.html#in-memory-execution.

[10] SOA Technology for beginners and learners, "Transient vs. durable bpel processes," http://ofmxperts.blogspot.nl/2012/11/transient-vs-durable-bpel-processes.html, Nov. 2012.

[11] R. Alur and P. Madhusudan, "Adding nesting structure to words," J. ACM, vol. 56, no. 3, May 2009, pp. 16:1–16:43.

[12] J. E. Hopcroft, Introduction to Automata Theory, Languages, and Computation, 3rd ed. Pearson Addison Wesley, 2007.

[13] A. S. Tanenbaum and M. van Steen, Distributed Systems: Principles and Paradigms, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2006, ch. 8, pp. 321–375.

[14] J. Schiedung, "Analysing and modelling of IOTP transactions by CPNs and BPEL," Master's thesis, Darmstadt University of Technology, 2004.

[15] L. Wang, A. Wombacher, L. Ferreira Pires, M. J. van Sinderen, and C.-H. Chi, "A colloborative processes synchronization method with regards to system crashes and network failures," in the 29th Symp. on Applied Computing (SAC), 2014.

[16] Apache Software Foundation, "Ode database setup," http://ode.apache.org/databases.html.

 ISBN: 978-1-61208-387-2

[17] N. Russell, W. Aalst, and A. Hofstede, “Workflow exception patterns,” in Advanced Information Systems Engineering, ser. Lecture Notes in Computer Science, E. Dubois and K. Pohl, Eds. Springer Berlin Heidelberg, 2006, vol. 4001, pp. 288–302.

[18] B. S. Lerner, S. Christov, L. J. Osterweil, R. Bendraou, U. Kannengiesser, and A. E. Wise, “Exception handling patterns for process modeling,” IEEE Transactions on Software Engineering, vol. 36, no. 2, 2010, pp. 162–183.

[19] S. Modafferi, E. Mussi, and B. Pernici, “Sh-bpel: a self-healing plug-in for ws-bpel engines,” in the 1st workshop on Middleware for Service Oriented Computing. NY, USA: ACM, 2006, pp. 48–53.

[20] A. Charfi, T. Dinkelaker, and M. Mezini, “A plug-in architecture for self-adaptive web service compositions,” in IEEE Intl. Conf. on Web Services, Jul. 2009, pp. 35–42.

[21] M. Fredj, N. Georgantas, V. Issarny, and A. Zarras, “Dynamic service substitution in service-oriented architectures,” in IEEE Congress on Services - Part I, Jul. 2008, pp. 101–104.

[22] L. Cavallaro, E. Nitto, and M. Pradella, “An automatic approach to enable replacement of conversational services,” in Service-Oriented Computing, L. Baresi, C.-H. Chi, and J. Suzuki, Eds. Springer Berlin Heidelberg, 2009, vol. 5900, pp. 159–174.

[23] O. Moser, F. Rosenberg, and S. Dustdar, “Non-intrusive monitoring and service adaptation for ws-bpel,” in the 17th intl. conf. on World Wide Web. NY, USA: ACM, 2008, pp. 815–824.

[24] F. Moo-Mena, J. Garcilazo-Ortiz, L. Basto-Diaz, F. Curi-Quintal, S. Medina-Peralta, and F. Alonzo-Canul, “A diagnosis module based on statistic and qos techniques for self-healing architectures supporting ws based applications,” in Intl. Conf. on Cyber-Enabled Distributed Computing and Knowledge Discovery, Oct. 2009, pp. 163 –169.

[25] OASIS Web Services Transaction (WS-TX) TC, Web Services Atomic Transaction (WS-AtomicTransaction), http://docs.oasis-open.org/ws-tx/wstx-wsat-1.2-spec.html, [retrieved: Feb., 2015], OASIS Standard, Rev. 1.2, Feb. 2009.

[26] ——, Web Services Business Activity (WS-BusinessActivity), http://docs.oasis-open.org/ws-tx/wstx-wsba-1.2-spec-os/wstx-wsba-1.2-spec-os.html, [retrieved: Feb., 2015], OASIS Standard, Rev. 1.2, Feb. 2009.

[27] M. T. Ozsu, Principles of Distributed Database Systems, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2007, ch. 12.

 ISBN: 978-1-61208-387-2

Hybrid Approach to Abstract Planning of Web Services

Artur Niewiadomski
Institute of Computer Science
Siedlce UPH
Siedlce, Poland
e-mail: artur.niewiadomski@uph.edu.pl

Wojciech Penczek
Institute of Computer Science
Polish Academy of Science and UPH
Warsaw, Poland
e-mail: penczek@ipipan.waw.pl

Jaroslaw Skaruz
Institute of Computer Science
Siedlce UPH
Siedlce, Poland
e-mail: jaroslaw.skaruz@uph.edu.pl

***Abstract*—The paper deals with the abstract planning problem – the first stage of the Web Service Composition in the PlanICS framework. Abstract planning consists in finding (multisets of) service types which can potentially satisfy the user query. We introduce a novel planning technique based on a combination of Genetic Algorithm with a Satisfiability Modulo Theories Solver, which allows to improve the efficiency of each separate method. The paper presents also some experimental results which show the advantages of the hybrid method when applied to large search spaces with many alternative solutions.**

Keywords-Web Service Composition; Abstract Planning; Genetic Algorithm; Satisfiability Modulo Theories; Hybrid Algorithm

I. INTRODUCTION

The Web Service Composition (WSC) problem [1][2][3] consists in composing simple functionalities, accessible via well-defined interfaces, in order to achieve more complex objectives. This is one of the main ideas of Service-Oriented Architecture (SOA) [1]. Unfortunately, WSC is a complex and hard problem and therefore requires an application of quite sophisticated methods and tools.

PlanICS [4] is a framework aimed at WSC, easily adapting existing real-world services. The main assumption in PlanICS is that all the Web services in the domain of interest as well as the objects that are processed by the services, can be strictly classified in a hierarchy of *classes*, organised in an *ontology*. Another key idea is to divide the planning into several stages. The first phase deals with *classes of services*, where each class represents a set of real-world services, while the other phases work in the space of *concrete services*. The first stage produces an *abstract plan* composed of service classes [5]. Then, the Offer Collector (OC), i.e., a module of PlanICS, interacts with instances of the service types constituting the abstract plan and retrieves data used in the concrete planning (CP) phase. As a result of CP, a *concrete plan*, i.e., a sequence of offers satisfying the predefined optimization criteria is obtained. Such a multiphase approach enables to reduce dramatically the number of Web services to be considered and inquired for offers.

This paper deals with the Abstract Planning Problem (APP), which is known to be NP-hard [5]. Our previous works employed several heuristic methods to solve APP: Genetic Algorithm (GA) [6][7], a translation to Petri nets [8], and Satisfiability Modulo Theories (SMT) Solvers [5]. The results of the extensive experiments show that the proposed methods are complementary, but every single one suffers from some disadvantages.

The main disadvantage of an SMT-based solution is often a long computation time, which is not acceptable in the case of a real-world interactive planning tool. The translation to Petri nets seems to be an efficient planning method, but only for some specific types of ontologies. On the other hand, a GA-based approach is relatively fast, but the probability of finding a solution, as well as the number of solutions found, decrease with the increase of the plan lengths.

Thus, our aim consists in exploiting the advantages of the two abstract planning methods – based on GA and SMT – by combining them into one hybrid algorithm. The main idea of our hybrid approach involves a modification of the standard GA in such a way that after every iteration of GA several individuals are processed by the SMT-based procedure, which aims at modifying them in order to obtain solutions of APP.

In our previous papers [9][10], we showed several variants of hybrid algorithms for solving the Concrete Planning Problem (CPP). However, in the case of CPP we dealt with the constrained optimisation problem. The main goal was to find a concrete plan satisfying all the constraints and maximizing the quality function. Here, in case of APP, the main aim is to find *all* abstract plans with a fixed number of service types. In practice, this means finding as many alternative plans as possible, using available resources (e.g., computer memory and computation time). Therefore, the main contribution of this paper is a new version of a hybrid algorithm combining GA with SMT, which finds abstract plans. Since we build upon our previous work, the general idea is somehow similar to the one applied in [9]. However, due to the fundamental differences between CPP and APP, the realisation of the hybrid abstract planner is substantially different than the hybrid concrete ones. The details are discussed at the end of the next subsection.

A. Related Work

The existing solutions to the WSC problem are divided into several groups. Following [11] our hybrid algorithm belongs to the AI planning methods. Other approaches include: automata theory [12], situation calculus [13], Petri nets [14], theorem proving [15], and model checking [16]. In what follows, we

 ISBN: 978-1-61208-387-2

shortly review the literature on composition methods related to our non-hybrid and hybrid algorithms.

A composition method closest to our SMT-based method [5] is presented in [17], where the authors reduce WSCP to a reachability problem of a state-transition system. The problem is encoded as a propositional formula and tested for satisfiability using a SAT-solver. This approach makes use of ontologies describing a hierarchy of types and deals with an inheritance relation. However, we consider also the states of the objects, while [17] deals with their types only. Moreover, among other differences, we use a multiset-based SMT encoding instead of Propositional Satisfiability Problem (SAT).

As far as our GA-based solution of APP is concerned, the approach closest to ours is given in [18], where GA is used to one phase planning, which combines an abstract and a concrete one. The idea of a multiset representation of a GA individual has been also used in [19]. However, contrary to our approach, no linearization of a multiset is generated in order to compute the fitness value.

Hybrid algorithms for WSC are also known in the literature. Jang et al. [20] use a combination of Ant Colony Algorithm with GA to find a concrete plan. While the experimental results are better than these obtained using a simple GA, the number of service types and offers used are not sufficient to draw general conclusions about efficiency of this approach. A combination [21] of two evolutionary algorithms, Tabu Search and GA, was also used to CP. Although this method allowed to find good quality concrete plans, our hybrid algorithm allows for dealing with much larger search spaces.

In our previous papers [9][10], we presented several variants of hybrid algorithms combining GA and SMT. One of them exploits an SMT-solver in order to (partially) generate the initial populations for GA. The other versions of hybrid algorithms are sightly modified GAs using an SMT-solver as a local search engine. That is, given an individual with a relatively high fitness value, an SMT-based procedure tries to change values of several genes in order to improve the individual. This approach is the closest to the method proposed in the present paper. Here, we also exploit an SMT-solver to improve some individuals of a population maintained by GA, but that is where the similarities end. The main differences result from various domains and the definitions of the abstract and concrete planning. Another difference is in what "the improvement" means in each case. In the case of CPP one is searching for such values of genes that satisfy the (predefined) constraints. However, satisfiability of the constraints is just one of the fitness function components, so such an improved individual is usually helpful for GA, but it is unlikely a final solution. On the other hand, in the case of the hybrid method applied to solve APP, if the SMT-based procedure returns an improved individual, then it already represents an abstract plan. The next group of differences are the technical details of the algorithms, like, e.g., the strategies regarding when, how often, and how many times SMT-solver should be run, which and how many genes are to be changed, and how to choose the individuals being good candidates to an improvement. Moreover, according to our best knowledge, there are no other approaches combining symbolic and evolutionary methods into hybrid algorithms.

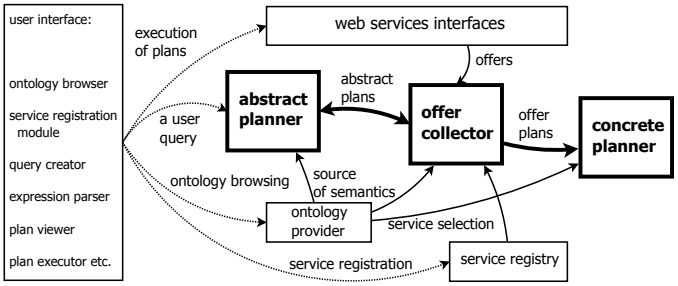


Figure 1. A simplified diagram of the PlanICS system architecture

The rest of the paper is structured as follows. In Section II, the PlanICS framework is introduced and APP is formulated. Section III presents the main ideas of our hybrid approach as well as some technical solutions. Next, the preliminary experimental results are given and discussed. The paper ends with some conclusions.

II. PLANICS FRAMEWORK

This section sketches the main ideas behind the PlanICS framework and gives all the intuitions necessary to formulate the abstract planning problem. The formal definitions can be found in [5].

An ontology contains a system of *classes* describing the types of the services as well as the types of the objects they process. A class consists of a unique name and a set of the attributes. By an *object*, we mean an instance of a class. By a *state* of an object, we mean a valuation of its attributes. A set of objects in a certain state is called a *world*.

The main goal of the system is to find a composition of services that satisfies a user *query*. The query interpretation is defined by two sets of worlds: the initial and the expected one. Moreover, the query may include several additional constraint sets used at different planning stages. Figure 1 shows the general PlanICS architecture. The bold arrows correspond to computation of a plan while the thin arrows model the planner infrastructure. While this paper concerns APP, in what follows we focus on the abstract planning phase.

A. Abstract Planning Problem

The first stage of the composition in the PlanICS framework is the abstract planning. It consists in matching services at the level of input/output types and the abstract values. That is, because at this stage it is sufficient to know if an attribute does have a value, or it does not, we abstract from the concrete values of the object attributes, and use two special values $\mathbf{set}$ and $\mathbf{null}$.

Thus, for a given ontology and a user query, the goal of the abstract planning is to find such a (multi)set of service types that allows to build a sequence of service types transforming an initial world of the user query into some *final world*, which has to be consistent with an expected world, also defined as a part of the query. The consistency between a final world and an expected one is expressed using the notion of the *compatibility* relation, formally defined in [5]. Intuitively, a final world W_f is compatible with an expected world W_e if the following

 ISBN: 978-1-61208-387-2

conditions are met: (i) for every object $o_e \in W_e$ there exists a unique object $o_f \in W_f$, such that both the objects are of the same type or the type of o_f is a subtype of o_e; (ii) both the objects agree on the (abstract) values of the common attributes.

The result of the abstract planning stage is called a Context Abstract Plan (CAP). It consists of a multiset of service types (defined by a representative transformation sequence), contexts (mappings between services and the objects being processed), and a set of final worlds. However, our aim is to find not only a single plan, but many (significantly different, and all if possible) abstract plans, in order to provide a number of alternative ways to satisfy the query. We distinguish between abstract plans built over different multisets of service types. See [5] for more details.

III. HYBRID APPROACH TO SOLVE APP

In this section, we recall some details of our GA-based abstract planner, as a base of the new hybrid algorithm. Then, an analysis of our previous experimental results obtained using GA and SMT is provided. Finally, basing on this analysis, the hybrid algorithm is proposed.

A. Application of GA to solving APP

The base of our hybrid approach is the standard GA aimed at solving APP. GA is a non deterministic algorithm maintaining a population of potential solutions during an evolutionary process. A potential solution is encoded in a form of a GA individual, which, in case of APP, is a sequence of natural values representing service types. However, since each abstract plan makes use of the multiset concept, the individuals differing only in the genes ordering are indistinguishable. In each iteration of GA, a set of individuals is selected for applications of genetic operations, such as the standard one-point crossover and mutation, which leads to obtaining a new population passed to the next iteration of GA. The selection of an individual and thus the promotion of its offspring to the next generation depends on the value of the fitness function.

Before the fitness function is calculated, every individual is processed by a procedure which is trying to find such an ordering of genes that an individual starts with a transformation sequence of a maximal length (see [6] for details). Moreover, such an executable prefix of an individual consists of *good service* types. Intuitively, a service type is *good* if it produces or modifies objects that are a part of some expected world, or they are an input of another good service type. Note that all the abstract plans are built solely of good service types only.

More details on applying GA to abstract planning can be found in [6][7].

B. Analysis

Let us begin with an analysis of the GA behaviour. Our previous works on an application of GA to solve APP show that a short computation time is certainly its advantage, but on the negative side the probability of finding a solution decreases along with the length of an abstract plan. Moreover, GA is unable to find solutions for instances with more than 6 services and several abstract plans in a search space.

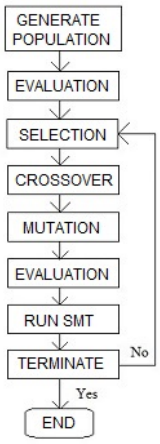


Figure 2. Hybrid algorithm flowchart.

On the other hand, the computation time of the SMT-based algorithm applied to APP is longer (comparing to GA), but all the solutions are found for instances with up to 9 services. Moreover, SMT performs a (symbolic) exploration of the whole search space to make sure that there is no other plan up to the given length. However, for instances with 12 services the SMT-solver is able to finish the computation within the given time only for search spaces containing one plan. For the remaining cases it indeed returns at least one solution, but it does not guarantee that more plans do not exist. The above discussion together with our previous successful applications of the hybrid algorithms to CPP [9][10], lead to a conclusion that a promising approach should consist in a combination of both approaches exploiting their advantages. To this aim we have to identify (and encode as an SMT formula) such a sub-problem of APP, which is solvable for an SMT-solver in a reasonable short time.

C. Algorithm

The problem we have encountered while solving some hard APP instances is as follows. Quite frequently GA ends up with unfeasible solutions containing only a few 'unmatched' services. This leads to the main idea of our hybrid approach. It relies upon an application of SMT to improve some number of the best individuals in each iteration of GA. An improvement consists in finding such good service types that can replace the unmatched ones. If such service types exist, then the modified individual represents an abstract plan. Figure 2 shows the consecutive steps of the hybrid algorithm.

Our hybrid algorithm mimics the standard steps of GA, which are modified in the following way. In each iteration of GA, after an evaluation step, a fixed number of top individuals (which do not yet constitute a complete abstract plan) are considered as candidates for an SMT-based improvement. Let I be an individual consisting of k genes. Let $e(I)$ denote the length of the maximal executable prefix of I and $g(I)$ denote the number of good service types of I. Then, the individual I is passed to the SMT-based procedure if the following conditions are met:

- $\lceil \frac{k}{2} \rceil \leq e(I) < k$, and

 ISBN: 978-1-61208-387-2

- $g(I) \geq \lceil \frac{k}{2} \rceil$.

Thus, only the individuals consisting at least in a half of good service types are considered as candidates for an improvement. Moreover, such a candidate should contain an executable prefix consisting of at least $\lceil \frac{k}{2} \rceil$ service types, but also having at least one gene to be changed.

If an optimal solution is found by SMT, then it is passed back to the GA population. The fixed number of individuals passed to the SMT procedure assures that only the individuals that have a real chance to be improved are considered for passing to SMT. Otherwise, this could result in a too long computation time of the SMT-based procedure. While an optimal solution is unknown a priori, it is very difficult to depict whether an individual is near to an abstract plan in the search space and thus if it is suitable for an improvement. In our approach this assessment is based on the two features of an individual: the length of its maximal executable prefix and the number of its good service types. Note that in the case of an abstract plan both the values are equal to k.

D. SMT-based problem encoding

The SMT procedure combined with GA is based on the encoding exploited in our "pure" SMT-based concrete planner [5]. However, now the task for an SMT-solver is to check whether the maximal (non-executable) suffix of an individual can be modified to make an improved individual a solution of APP. Thus, in addition to the user query q and the ontology, the SMT-based procedure takes also as input an individual and the length of its maximal executable prefix. The general idea of the SMT task is depicted in Figure 3.

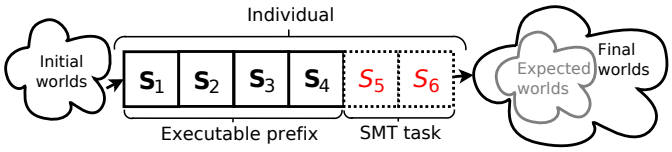


Figure 3. SMT role in the hybrid algorithm

Let I denote an individual in the form of a sequence of service types and I_j be the service type at the j-th position of the sequence. Let e denote the length of the maximal executable prefix and let k be the total length of I. Then, the task of an SMT-solver is reduced to checking satisfiability of the following formula:

$$\varphi_k^q = \mathcal{I}^q \bigwedge_{j=1..e} \left(\mathcal{C}_j \wedge (s = I_j) \wedge \mathcal{T}_j^s \right) \bigwedge_{i=(e+1)..k} \left(\mathcal{C}_i \bigvee_{s \in \mathbb{S}} \mathcal{T}_i^s \right) \wedge \mathcal{E}_k^q \wedge \mathcal{B}_k^q \quad (1)$$

where $\mathcal{I}^q$ and $\mathcal{E}_k^q$ are the formulas encoding the initial and the expected world of the user query, respectively, $\mathcal{C}_i$ encodes the i-th context function, $\mathbb{S}$ is the set of all service types from the ontology, and $\mathcal{T}_i^s$ encodes the worlds transformation by a service type s. $\mathcal{B}_k^q$ stands for a formula preventing from finding the already known solutions, if there are any. See [5] for more encoding details. Thus, an SMT-solver tries to find a sequence of service types of length $(k-e)$, which can replace the non-executable suffix of I in such a way that the improved individual is a (previously unknown) solution of APP.

IV. Experimental Results

We have evaluated the efficiency of the hybrid algorithm against the GA and SMT-based planners using the ontologies and the user queries generated by our software - Ontology Generator (OG). All ontologies are generated by OC in a random manner meeting the semantic rules. Each query is also generated randomly in such a way that the number of various abstract plans equals to the value of a special parameter of OG. This guarantees that we know a priori whether the planners have found all solutions.

A. Configuration of the Experiments

In order to evaluate the efficiency of our planners, we have conducted experiments using twenty four instances of APP generated by OG, with the following parameter values: the solution length (the number of service types constituting a plan): $k \in \{6, 9, 12, 15\}$, the number of the service types in the ontology: $n \in \{64, 128, 256\}$, and the number of the plans in the search space: $sol \in \{1, 10\}$. Thus, the size of the search space varies from $64^6 = 2^{36}$ for the first and the fourth experiment, up to $256^{15} = 2^{120}$ in the case of the 21st and the 24th instance.

Each experiment has been repeated 20 times on a standard PC with 2GHz CPU, 8GB RAM, and Z3 [22] version 4.3 as an SMT-solving engine. The experiments involving the "pure" GA and hybrid planner have been performed using the following parameters: the population size equals 100 for the instances with one solution only and 500 for the instances with ten plans in the search space, the number of the iterations equals 100, whereas the probabilities of crossover and mutation are at levels of 95% and 0.5%, respectively. We impose the 2000 sec. time limit for every experiment.

B. Experimental Results Evaluation

A comparison of the experimental results of all the three planners is presented in Table I. The best results are marked with bold. The first four columns from the left contain the experiment number and the OG parameters used to generate the instances. The next six columns present the results of our hybrid planner, such as the time consumed by the SMT-based procedure calls and consumed by GA, the total computation time, the average and the maximal number of plans found, as well as the probability of finding a plan. Note that the average number of plans found is computed taking into account only these cases, where at least one solution has been found. The next four columns contain the results obtained using the GA-based abstract planner. That is, from left to right: the computation time, the average and the maximal number of plans found, and the probability of finding a plan. Finally, the last two columns display: the time consumed by the SMT-based abstract planner in order to find the first plan and the total computation time. Since the SMT-based planner always finds all the plans (provided it has enough time and memory), we do not report any other results here. Note that the memory usage does not exceed 2GB, even during the experiments with the largest instances and the "pure SMT" as the planning method. The SMT-based planner seems to be the most memory-demanding in the last phase of searching, i.e., while checking that there is no more plans and (symbolically)

 ISBN: 978-1-61208-387-2

TABLE I. EXPERIMENTAL RESULTS

				Hybrid						Pure GA				Pure SMT	
Exp	**k**	**n**	**sol**	**SMT** [s]	**GA** [s]	**Total** [s]	**Avg** plans	**Max** plans	**Prob.** [%]	**Time** [s]	**Avg** plans	**Max** plans	**Prob.** [%]	**First** [s]	**Total** [s]
1	6	64	1	4.05	8.24	12.29	1	1	100	**5.71**	**1**	**1**	**100**	6.31	12.8
2		128		5.77	8.78	14.55	1	1	100	**8.07**	**1**	**1**	**100**	7.29	14.8
3		256		10.79	13.29	24.07	1	1	100	**13.62**	**1**	**1**	**100**	16.66	27.1
4		64	10	3.04	25.74	28.78	3.25	10	100	24.29	5.4	10	100	**5.22**	**18.3**
5		128		6.52	32.15	38.67	3.15	8	100	31.21	6.25	10	100	**8.54**	**26.6**
6		256		13.85	43.33	57.18	3.65	8	100	45.95	5.55	9	100	**11.93**	**38.1**
7	9	64	1	12.08	11.67	23.75	1	1	85	**11.83**	**1**	**1**	**95**	19.49	58.7
8		128		25.65	15.68	41.33	1	1	90	**13.43**	**1**	**1**	**100**	41.01	90.1
9		256		43.61	28.88	72.49	1	1	90	**26.74**	**1**	**1**	**90**	54.99	133
10		64	10	**17.54**	**56.9**	**74.43**	**3.15**	**10**	**100**	57.69	1.77	4	65	21.09	295
11		128		**30.64**	**63.38**	**94.02**	**4.16**	**10**	**95**	69.94	1.54	4	65	49.93	553
12		256		**61.64**	**113.05**	**174.69**	**4.32**	**10**	**95**	113.15	1.33	2	30	113.3	977
13	12	64	1	55.09	21.77	76.86	1	1	45	**21.22**	**1**	**1**	**65**	156.4	781
14		128		86.48	30.15	116.62	1	1	85	**28.12**	**1**	**1**	**60**	203.2	1962
15		256		118.7	46.82	165.52	1	1	55	**46.31**	**1**	**1**	**60**	315.4	1947
16		64	10	**78.98**	**118.56**	**197.54**	**2.79**	**10**	**95**	118.29	0	0	0	113.5	> 2000
17		128		**109.89**	**139.96**	**249.84**	**2.38**	**10**	**80**	148.65				250.5	
18		256		**193.17**	**253.22**	**446.39**	**1.85**	**6**	**65**	260.94				325.8	
19	15	64	1	119.09	33.68	152.77	1	1	25	**34.56**	**1**	**1**	**30**	469.7	
20		128		185.34	43.17	228.51	1	1	30	**40.45**	**1**	**1**	**25**	382.1	
21		256		247.3	68.26	315.56	1	1	35	**68.69**	**1**	**1**	**35**	1018	
22		64	10	**168.46**	**237.57**	**406.03**	**1.67**	**3**	**30**	216.6	0	0	0	413	
23		128		**309.53**	**267.83**	**577.36**	**3**	**5**	**10**	261.21				1850	
24		256		**304.88**	**450.63**	**755.5**	**3**	**3**	**5**	437.59				931	

exploring the whole search space. The memory usage of the "pure" GA algorithm remains below several hundred MB, and in the case of the hybrid method it does not exceed 1GB for the largest instances.

It is easy to observe that in the case of the instances with only one plan in the search space the "pure" GA algorithm is superior to the other algorithms due to its relative low computation time. However, it is worth noticing that the probability of finding a solution by GA drops rapidly with the increase of the lengths of plans. On the other hand, analysing the experiments where there are ten plans in the search space, one can notice that the "pure" SMT algorithm is faster than the other algorithms only for APP instances having the shortest solutions. In all the other cases the hybrid planner is clearly superior. Concerning the experiments 10, 11, and 12, the computation time of the hybrid planner is much lower than the time consumed by the "pure" SMT-based algorithm. Comparing the results obtained using the hybrid and the "pure" GA algorithm it is easy to see that admittedly GA is about 30% faster, but the hybrid one outperforms GA if all other measures are considered.

The analysis of the remaining experiments, where the search spaces contain ten solutions is even simpler. These are the most important results related to the instances which turned out be hard for the pure GA and SMT planner, as GA does not yield any result while the SMT-based planner is running out of time. The hybrid algorithm outperforms the other planners for the instances, for which the plans consist of more than 6 service types. In case of 12 services, the maximal number of the abstract plans found equals to 10, while for 15 services it is 5. Summarizing, the experiments 16, 17, 18, 22, 23, and 24 prove that applying the hybrid algorithm to APP one can obtain substantially better results.

C. Comparison with Other Approaches

We have done our best to compare efficiency of our tool with another system. Nam et al. [17] report 7 experiments performed on a set of 413 *concrete* Web services, where SAT-time consumed for every composition varies from 40 to 60 sec. However, the composition consists only in simple type matching, the plans consist of a few services only, and the main goal is to find the shortest sequence of services satisfying the user query. We have repeated these experiments translating first the input data to the PLANICS ontology. We treated each concrete service as a service type, and we modelled the service parameters type hierarchy as the object types. Our results have appeared to be better. PLANICS is able to find the shortest solution in just fractions of a second of SAT-time and in several seconds of the total computation time. Overall, our approach is more complex and the composition by just types matching is a special, simplified case of our planning. Moreover, instead of searching just for the shortest solution, we focus on finding a number of alternative plans.

For the second comparison we have used the Fast Downward tool [23] that is aimed at solving problems specified in PDDL (Planning Domain Definition Language) [24]. We developed a tool which translates PLANICS ontologies and user queries into PDDL domains and problems, respectively. Then, we translated the benchmarks described in Sec. IV and we used them as input for FD. The FD tool was able to find a solution only for smallest instances with 64 service types. For these cases an interesting observation can be made. Namely, the amount of time and memory consumed by FD is not so strongly related to the length of the plan, like in the case of PLANICS, but they clearly depend from number of services in the ontology. For larger benchmarks the whole available memory has been quickly consumed and the computation had to be aborted. It shows the advantage of PLANICS while reasoning in large ontologies.

V. CONCLUSION AND FUTURE WORK

A new hybrid algorithm based on GA and SMT has been proposed to solve APP. The algorithm has been implemented and some preliminary experiments have been performed for

 ISBN: 978-1-61208-387-2

benchmarks having different sizes of ontologies and solutions in the search space. The very first results show that using a combination of the SMT- and GA-based approach, one can obtain quite good results, especially for problems that are hard to solve using the "pure" planning methods, i.e., with large search spaces and many alternative solutions.

We plan to further improve the efficiency of our hybrid approach in terms of lower computation times and higher probabilities of finding solutions. The successful application of the hybrid algorithm to abstract planning problem shows that the proposed method has a high potential. Thus, another important task to be addressed in a future work is an application of similar algorithms to other hard problems involving an exploration of huge search spaces.

ACKNOWLEDGMENT

This work has been supported by the National Science Centre under the grant No. 2011/01/B/ST6/01477.

REFERENCES

[1] M. Bell, Introduction to Service-Oriented Modeling (SOA): Service Analysis, Design, and Architecture. John Wiley & Sons, 2008, ISBN: 978-0-470-14111-3.

[2] S. Ambroszkiewicz, "Entish: A Language for Describing Data Processing in Open Distributed Systems," Fundam. Inform., vol. 60, no. 1-4, 2003, pp. 41–66.

[3] J. Rao and X. Su, "A Survey of Automated Web Service Composition Methods," in Proc. of SWSWPC'04, ser. LNCS, vol. 3387. Springer, 2005, pp. 43–54.

[4] D. Doliwa et al., "PlanICS - a Web Service Compositon Toolset," Fundam. Inform., vol. 112(1), 2011, pp. 47–71. [Online]. Available: http://dx.doi.org/10.3233/FI-2011-578

[5] A. Niewiadomski and W. Penczek, "Towards SMT-based Abstract Planning in PlanICS Ontology," in Proc. of KEOD 2013 International Conference on Knowledge Engineering and Ontology Development, September 2013, pp. 123–131.

[6] J. Skaruz, A. Niewiadomski, and W. Penczek, "Evolutionary Algorithms for Abstract Planning," in PPAM (1), ser. Lecture Notes in Computer Science, R. Wyrzykowski, J. Dongarra, K. Karczewski, and J. Wasniewski, Eds., vol. 8384. Springer, 2013, pp. 392–401.

[7] ——, "Solving the abstract planning problem using genetic algorithms," Studia Informatica, vol. 1-2(17), 2013, pp. 29–48, ISSN: 1731-2264.

[8] A. Niewiadomski and K. Wolf, "LoLA as Abstract Planning Engine of PlanICS," in Proceedings of the International Workshop on Petri Nets and Software Engineering, co-located with 35th International Conference on Application and Theory of Petri Nets and Concurrency (PetriNets 2014) and 14th International Conference on Application of Concurrency to System Design (ACSD 2014), Tunis, Tunisia, June 23-24, 2014, pp. 349–350. [Online]. Available: http://ceur-ws.org/Vol-1160/paper26.pdf

[9] A. Niewiadomski, W. Penczek, and J. Skaruz, "Genetic Algorithm to the Power of SMT: a Hybrid Approach to Web Service Composition Problem," in Service Computation 2014 : The Sixth International Conferences on Advanced Service Computing, 2014, pp. 44–48.

[10] ——, "A Hybrid Approach to Web Service Composition Problem in the PlanICS Framework," in Mobile Web Information Systems, ser. Lecture Notes in Computer Science, I. Awan, M. Younas, X. Franch, and C. Quer, Eds. Springer International Publishing, 2014, vol. 8640, pp. 17–28. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-10359-4_2

[11] Z. Li, L. O'Brien, J. Keung, and X. Xu, "Effort-Oriented Classification Matrix of Web Service Composition," in Proc. of the Fifth International Conference on Internet and Web Applications and Services, 2010, pp. 357–362.

[12] S. Mitra, R. Kumar, and S. Basu, "Automated Choreographer Synthesis for Web Services Composition Using I/O Automata," in ICWS, 2007, pp. 364–371.

[13] V. Chifu, I. Salomie, and E. St. Chifu, "Fluent calculus-based Web service composition - From OWL-S to fluent calculus," in Proc. of the 4th Int. Conf. on Intelligent Computer Communication and Processing, 2008, pp. 161 –168.

[14] V. Gehlot and K. Edupuganti, "Use of Colored Petri Nets to Model, Analyze, and Evaluate Service Composition and Orchestration," in System Sciences, 2009. HICSS '09., jan. 2009, pp. 1 –8.

[15] J. Rao, P. Küngas, and M. Matskin, "Composition of semantic web services using linear logic theorem proving," Inf. Syst., vol. 31, no. 4, Jun. 2006, pp. 340–360.

[16] P. Traverso and M. Pistore, "Automated composition of semantic web services into executable processes," in The Semantic Web ISWC 2004, ser. LNCS, 2004, vol. 3298, pp. 380–394.

[17] W. Nam, H. Kil, and D. Lee, "Type-Aware Web Service Composition Using Boolean Satisfiability Solver," in Proc. of the CEC'08 and EEE'08, 2008, pp. 331–334.

[18] F. Lecue, M. D. Penta, R. Esposito, and M. Villani, "Optimizing QoS-Aware Semantic Web Service Composition." in Proceedings of the 8th International Semantic Web Conference, 2009, pp. 375–391.

[19] I. Garibay, A. S. Wu, and O. Garibay, "Emergence of genomic self-similarity in location independent representations," Genetic Programming and Evolvable Machines, vol. 7(1), 2006, pp. 55–80.

[20] Z. Jang, C. Shang, Q. Liu, and C. Zhao, "A Dynamic Web Services Composition Algorithm Based on the Combination of Ant Colony Algorithm and Genetic Algorithm," Journal of Computational Information Systems, vol. 6(8), 2010, pp. 2617–2622.

[21] J. A. Parejo, P. Fernandez, and A. R. Cortes, "QoS-Aware Services composition using Tabu Search and Hybrid Genetic Algorithms," Actas de los Talleres de las Jornadas de Ingenieria del Software y Bases de Datos, vol. 2(1), 2008, pp. 55–66.

[22] L. M. de Moura and N. Bjørner, "Z3: An efficient SMT solver," in Proc. of TACAS'08, ser. LNCS, vol. 4963. Springer-Verlag, 2008, pp. 337–340.

[23] M. Helmert, "The Fast Downward Planning System," Journal of Artificial Intelligence Research, vol. 26, 2006, pp. 191–246.

[24] A. E. Gerevini, P. Haslum, D. Long, A. Saetti, and Y. Dimopoulos, "Deterministic planning in the fifth international planning competition: PDDL3 and experimental evaluation of the planners," Artificial Intelligence, vol. 173, no. 5, 2009, pp. 619–668.

 ISBN: 978-1-61208-387-2

Appropriate Machine Learning Methods for Service Recommendation Based on Measured Consumer Experiences Within a Service Market

Jens Kirchner
Karlsruhe University of Applied Sciences
Linnaeus University
Email: Jens.Kirchner@hs-karlsruhe.de,
Jens.Kirchner@lnu.se

Philipp Karg and Andreas Heberle
Karlsruhe University of Applied Sciences
Moltkestr. 30, 76133 Karlsruhe, Germany
Email: kaph1014@hs-karlsruhe.de,
Andreas.Heberle@hs-karlsruhe.de

Welf Löwe
Linnaeus University
351 06 Växjö, Sweden
Email: Welf.Lowe@lnu.se

***Abstract*—The actual experience of the performance of services at consumers' side is a desirable foundation for service selection. Considering the knowledge of previous performance experiences from a consumer's perspective, a service broker can automatically select the best-fitting service out of a set of functionally similar services. In this paper, we present the evaluation of machine learning methods and frameworks which can be employed for service recommendation based on shared experiences of previous consumers. Implemented in a prototype, our approach considers a consumer's call context as well as its selection preferences (expressed in utility functions). The implementation of the framework aims at the time-critical optimisation of service consumption with focus on runtime aspects and scalability. Therefore, we evaluated and employed high-performance, online and large scale machine learning methods and frameworks. Considering the Internet as a service market with perpetual change, strategies for concept drift have to be found. The evaluation showed that with the current approach, the framework recommended the actual best-fit service instance in 70 % of the validation cases, while in 90 % of the cases, the best or second best-fit was recommended. Furthermore, within our approach employing the best method, we achieved 94.5 % of the overall maximum achievable utility value.**

Keywords–Service Selection; Service Recommendation; Machine Learning; Big Data.

I. INTRODUCTION

Service-Oriented Computing (SOC), Software as a Service (SaaS), Cloud Computing and Mobile Computing indicate that the Internet develops itself to a market of services where service consumers can dynamically and ubiquitously consume functionality from services with limited or no knowledge about the implementation or the system environment of the provided service. Besides the functionality, service consumers are interested in the performance of a service, which is expressed in its non-functional properties (NFPs) such as response time, availability or monetary charges. Within a service market, service functionality may be provided by several competing service instances. Among these similar services, service consumers are interested in the consumption of the service instance which fits best towards their preferences. In [1], we described that service selection has to be based on the actual experience of NFPs at consumer side. We defined *Service Level Achievement* to be the general performance of a service at consumer side (consumer side measured NFPs). In the case of basing service selection on Service Level Achievements, a service broker can automatically select the best-fit service among functionally similar services. In contrast, when the selection is based on Service Level Agreements (SLAs), one can only state the fact that SLAs have not been met; mitigating the issue, however, requires human action, hence, time. Perpetual change is one of the major characteristics of service markets such as the Internet. NFPs of service instances can be volatile (e. g., high load at certain times and limited resources), new functional equivalent instances enter the market; others are temporary or permanently not available. A collaborative knowledge base of consumption experiences benefits single users and helps them to optimise service selection towards their needs and based on their call context. Call contexts consider aspects that have an influence on NFPs at consumer side such as location, time, weekday, etc. When recommendation is based on actual experienced NFPs, their preferred weight from consumer side and the call context, the knowledge base has to be built on a potentially high load of measurement data, which needs to be processed and learned promptly. Within our framework, we develop a service broker which bases its decisions on consumption measurements of previous service calls [1]. In particular, service recommendations are based on a call context and a user's preferences which are expressed in a utility function. Basing service selection on Service Level Achievements requires measuring the NFPs of services at the moment of the actual service call. E. g., measuring response time of a service at a point in time called from a certain location. This can be easily integrated in SOC/SaaS infrastructures. However, it also requires the aggregation of individually measured data to turn it into knowledge about the expected performance of services in the future. Machine learning is an obvious candidate for this aggregation. For this task, machine learning methods have to cope with a high load of data efficiently in real-time or short periods.

Machine learning cannot be used out of the box until we find answers to the following questions: 1) Which concrete machine learning algorithm can be applied effectively, i. e., with high accuracy in the prediction of achievements? 2) Are there differences in the accuracy of different algorithms in the prediction of achievements? 3) How can we apply effective machine learning efficiently, i. e., with a minimum impact on service (selection) time?

In this paper, we seek to answer 1) by selecting reasonable candidates and experimentally evaluating their prediction accuracy. Addressing 2), for the selected algorithms, we even assessed prediction accuracy and its effect on the actual utility

 ISBN: 978-1-61208-387-2

of the consumers (here, response time and availability). Finally, in order to address 3), we defined an architecture that is split into a foreground and background model. The service recommendation knowledge in the foreground model is used for fast service selection. It is asynchronously updated with the knowledge output of the background model, which, in turn, is gained by (potentially not quite as fast) machine learning on service call measurements at consumer side.

For the evaluation of appropriate machine learning approaches and their implementations in machine learning frameworks which can be employed for best-fit service recommendation, our focus was set on speed, real-time/online processing, accuracy and concept drift mechanisms. “Concept drift occurs when the values of hidden variables change over time. That is, there is some unknown context for concept learning and when that context changes, the learned concept may no longer be valid and must be updated or relearned” [2]. The evaluation was conducted with a prototype implementation of a holistic recommendation component for a service broker in order to determine the most appropriate method and framework for this purpose.

In Section II, the architecture of our service selection framework is introduced. Section III describes the challenges and optimisation focus within service selection. Section IV focuses on the pre-selection and evaluation of machine learning methods and frameworks. In Section V, the conducted evaluation of machine learning methods regarding appropriateness for service selection in a service market is described. Related work is outlined in Section VI. Finally, the conclusion is done in Section VII.

II. Framework architecture

In [1], we introduced our framework which optimises service selection based on consumer experience, call context and preferences (utility functions). In this paper, we focus on the recommendation component of our framework, which is depicted in Figure 1. The illustration shows the service consumer’s realm where a local component manages dynamic bindings and requests for best-fit service instances. The second task is to measure a service call’s performance (measurement of experienced NFPs) and to provide this information to the central broker component for learning purpose. Within the scope of this paper, our main focus is set on the foreground and background model combination, which is important for the overall recommendation process.

When a measurement feedback ticket arrives, the collected data is persisted and the pre-processing of the data is conducted. Afterwards, the data is used for the online learning of the NFPs. There exists a learning model for each NFP and service instance combination. In time or amount intervals, for each utility function and call context, the utility values for each service instance are calculated. Once all utility values for all instances are calculated, the service instance with the highest utility value within each utility function and call context will be updated in the foreground table. This foreground table contains the recommendation information for each call context and preference (utility function). When a service recommendation request arrives, the broker only has to look up this information in the recommendation table of the foreground model, while the actual learning and calculation is done asynchronously in the background model.

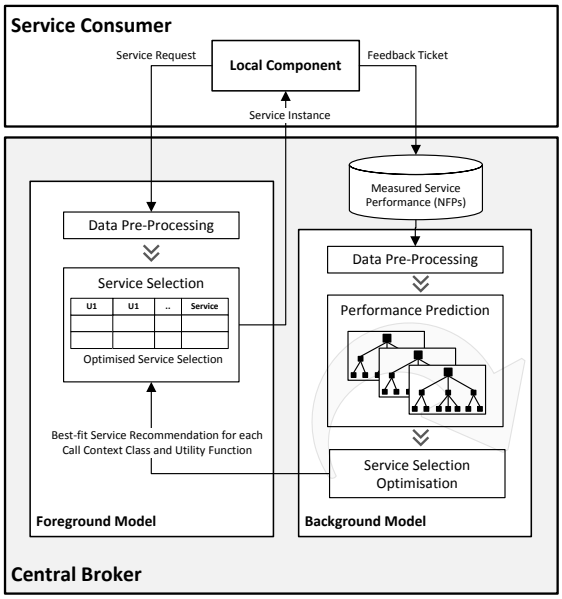


Figure 1. Foreground and background model within our framework.

The background model activities comprise machine learning, calculation and determination for the optimisation at consumer side. However, these tasks are time-consuming. The decoupling of the time-consuming tasks of the background model from the foreground model, which handles the time-critical recommendation tasks, reduces or even avoids the costs in terms of (service) time.

III. Optimisation focus and challenges

Following the Cross Industry Standard Process for Data Mining (CRISP-DM) [3], before machine learning methods can be employed, it is mandatory to have a clear understanding of the optimisation goals as well as the data which is used. As introduced above, the service broker and its learning component are supposed to recommend the best-fit service instance among functionally equivalent candidates. In particular, they are supposed to be aware of the call context and a consumer’s preferences and base their recommendation on the experiences of previous service calls by consumers of similar call contexts.

In general, the selection of a service instance is based on one or more NFPs. NFPs have different scales of measurement with different optimisation functions. For example, response time is a ratio scale with an optimisation towards the minimum. The availability for a service at a specific time is nominal: a service is either available or not. In such a case, the optimisation focus is to select a service instance which has the highest (maximum) probability of being available. When the selection of a service instance is based on more than one NFP, NFP data has to be normalised in order to be comparable and calculable. Usually, in such a case, not all NFPs are equally important, so their importance has to be weighted and taken into account. Within our framework, utility functions are used to calculate the utility value for each service instance based on the expected NFPs of each candidate and their weighted impact. “Utility functions can be captured in a vector of real values, each representing the weight of a corresponding quality metric [NFP]. The idea is based on a weighted scoring model”

 ISBN: 978-1-61208-387-2

[1]. For instance, lowest response time is more important (weighted: 60 %) than lowest price (weighted: 40 %) would results in a utility function $U(ResponseTime, Price) = 0.6 \times ||ResponseTime|| + 0.4 \times ||Price||$, where $|| \cdot ||$ normalises $ResponseTime$ and $Price$, respectively, between 0 and 1 [1]. Since preferences vary, the utility functions also vary among all service consumers. Therefore, within the overall optimisation/recommendation process, the overall utility is individual. However, within a single tier recommendation, the NFPs of service instances are based on consumers' contexts. Hence, within the same call context (e. g., time, weekday, location, type/size of input data), consumers with different preferences experience statistically similar NFPs, but the calculated utilities are different due to different utility functions. Within our framework, the expected NFPs of service instances are learned and each individual utility value is (pre-)calculated, which is then used for the actual individual service recommendation.

Analysing the market, change is the most important characteristic regarding the data. Besides new service instances, existing ones may temporarily not be available or cease to exist for good. But one has also to take into account that changes in infrastructure, network or market participants' behaviour arise, which also affect the NFPs within certain call contexts. These things are in general not evident to consumers, but have to be taken into account for recommendation tasks. Therefore, learning components for service brokers have to cope with rapid change. Service recommendation is in total a time-critical challenge. First, change has to be discovered quickly, and secondly, the recommendation query itself is supposed to be part of a service call and service time is one of the major optimisation goals. However, service recommendation is time-consuming, especially the learning and calculation part of it. Within our framework, we tackle this drawback by introducing two approaches. The first approach splits the component for direct dynamic service recommendation requests into a foreground and a background model. The learning of the expected NFPs is done in the background model, while its results together with the pre-calculated utility values, and therefore the best-fit service instances within each utility cluster and call context class, are stored in the foreground model, which can be easily and quickly retrieved. The second approach focuses on dynamic service binding, which is mainly based on the idea that service bindings are updated at consumers' side. For this, we developed a plug-in for middleware/SOA products. The actual service binding addresses within the dynamic service binding are updated by the central component by the usage of the publish–subscribe pattern [1].

We predefined the following optimisation goals for the recommendation unit within our framework:

1) High-performance service determination
2) High recommendation accuracy of the best-fit service instance
3) Continuous machine learning process
4) Adaptation to performance (NFP) changes

IV. MACHINE LEARNING METHODS AND FRAMEWORKS

This section focuses on the pre-selection and evaluation of machine learning methods and frameworks.

A. Pre-selection of machine learning methods

Based on [4][5], there are several aspects for the evaluation of machine learning methods such as speed, accuracy, scalability, robustness and interpretability. The requirements listed in Table I were defined for the pre-selection of appropriate machine learning methods.

TABLE I. REQUIREMENTS FOR THE SELECTION OF MACHINE LEARNING METHODS

Speed	describes how efficient the machine learning method performs concerning the training and prediction time. Furthermore, this aspect also concerns the overall machine learning process as a 'critical path' from end-user side.
Accuracy	describes how effective the machine learning method performs: Degree of correct classification or coefficient of determination in regression [5].
Scalability	considers the ability of the method to be efficiently applied to a large data set [5].
Robustness	describes the ability to make correct classifications and predictions, given noisy or missing data value. It also considers whether the method is able to run automatically in a changing environment [5].

There is a broad variety of learning methods that address classification and regression. The major aim of our recommendation unit is to recommend the best-fit service. Although the prediction of a certain value can be employed for pre-calculation purpose, the goal within the overall recommendation process is to recommend best-fit service instances, which is a nominal result. For classification in general, the costs of the learning phase are cheaper than for regression. Therefore, we focused primarily on classification methods in the initial phase. In [4], the author published a comprehensive overview of established supervised machine learning classification techniques. This overview provides useful information for method selection, highlighting the benefits and drawbacks of each method which helped us to find appropriate methods for further evaluation. Table II is an extraction reduced to the requirements we outlined in Table I.

As previously mentioned, although in general *accuracy* is an important desire in machine learning, in our time-critical domain, *speed* is as important within our recommendation process. However, due to the separation of the foreground and background model, speed is not as important anymore, since learning is done asynchronously in the background model. Therefore, we are able to put a stronger focus on accuracy again. *Scalability* and *robustness* are also important criteria in our framework. Although Naïve Bayes and Decision Trees are not highly rated on accuracy in Table II, their rating in all other criteria is high and, among all criteria, they have the highest overall rating. After initial pre-tests with test data sets of various kinds, we selected Naïve Bayes, Hoeffding Tree [6] and Fast Incremental Model Trees with Drift Detection (FIMT-DD) [7] for the implementation in our framework, and therefore, for the evaluation within our broker scenario.

The Naïve Bayes classifier is a simple probabilistic classifier based on Bayesian statistics (Bayes' theorem) with strong independence assumptions [8]. The Hoeffding tree or Very Fast Decision Tree (VFDT) is an incremental, anytime decision tree induction algorithm that is capable of learning from massive data streams, assuming that the distribution generating examples do not change over time. It exploits the fact that a small sample can often be enough to choose an optimal

 ISBN: 978-1-61208-387-2

TABLE II. COMPARISON OF THE MAJOR MACHINE LEARNING METHODS
(**** REPRESENT THE BEST AND * THE WORST PERFORMANCE) (BASED ON [4])

	Decision Tree	Naïve Bayes	Neural Networks	kNN	SVM	Rule Learners
Speed of Learning/Training	***	****	*	****	*	**
Speed of Classification/Prediction	****	****	****	*	****	****
Scalability / Incremental Learning	**	****	***	****	**	*
Accuracy	**	*	***	**	****	**
Robustness/Tolerance to missing values	***	****	*	*	**	**
Robustness/Tolerance to noise	**	***	**	*	**	*

splitting attribute. This idea is supported mathematically by the Hoeffding bound, which quantifies the number of observations needed to estimate some statistics within a prescribed precision [6][9]. FIMT-DD focus on time-changing data streams with explicit drift detection [7].

B. Machine learning frameworks

Besides the machine learning methods, their implementation within libraries and frameworks are as important. For the selection of machine learning frameworks, we considered the requirements listed in Table III to be relevant.

TABLE III. REQUIREMENTS FOR THE SELECTION OF MACHINE LEARNING FRAMEWORKS/LIBRARIES

Integration	Easy integration of the library in Java; Capability of online processing within a machine learning workflow (online learning)
Automation	High degree of automation; Ability to adapt to changes (e. g., changing service instances, service consumers with new call contexts)
Usage	Strong dependency between the library and the machine learning methods; Framework targets general approaches and is not limited to specific purposes
Open Source	Framework should be open source and freely available to the public

In order to get an overview about popular and wide-spread software in the data analytic and data mining area, we took the results of KDnuggets' online poll about software that their open community members had used within the past 12 months in real projects [10]. The results of the poll, in which 3,000 had participated, contain open source as well as commercial products. Although these results give an overview about the software in this field, it has to be considered critically, since the poll was open and everyone could have taken part.

The following frameworks were pre-evaluated based on our requirements listed in Table III, but were not suitable for our purpose. RapidMiner [11] is a well-known and widely-used desktop application for solving a variety of machine learning, data and text mining tasks. It offers a comprehensive selection of machine learning methods and integrates third-party libraries. Despite of all the benefits, we declined RapidMiner mainly because of the missing capability of incremental/online learning. A reason for that could be that RapidMiner comes from the area of classical batch processing and analytics. Furthermore, an integration in Java is possible but requires additional efforts for automation tasks. R [12] is an open source software environment for statistical computing and graphics. With a classical statistical background, R does not provide modern machine learning approaches and does not focus on incremental learning. Apache Mahout [13] is a suite of machine learning libraries with algorithms for clustering, classification and collaborative filtering on top of scalable and distributed systems. Despite the overall advantages, it was not selected because of the little selection of machine learning methods and the specific use cases. Other libraries such as Apache Spark [14], KNIME [15], Shogun [16], Shark [17], scikit-learn [18] and Vowpal Wabbit [19] were reviewed but not considered for further evaluation, because of early misfits to our requirements.

We selected Weka [20] and MOA [21] because of their extensive collection of classical machine learning methods as well as new algorithms with state of the art concepts for incremental learning. Because of their native Java integration ability, they provide a high degree of automation. Furthermore, Weka is also used by other software in this sector, such as RapidMiner. Both frameworks are open source and developed by the University of Waikato. Weka contains different methods and algorithms for pre-processing, classification, regression, clustering, association rules and visualisation. MOA stands for Massive Online Analysis, which focuses on online algorithms for data streams. It includes several state of the art algorithms for classification, regression and clustering.

V. EVALUATION

This section describes the evaluation scenario, evaluation criteria, the evaluation environment and method as well as the results of the evaluation of machine learning methods regarding their appropriateness for service selection in a service market.

A. Evaluation scenario

The evaluation of the learning methods and frameworks implies that the actual best-fit service instance is known at each call context (location, weekday, time, etc.) with each utility function. This is a challenge when it comes to a real-world validation. In reality, service calls in a service market, especially the Internet, cannot be repeated under the exactly identical conditions as the Internet's network behaviour as well as a service's infrastructure are complex systems in terms of factors that influence service calls. For instance, at a certain, unique moment, the load of a service instance's system environment and the network load or any incident are combinations of coincidences and can therefore not be repeated. In order to gain exact reproducible situations, all service calls which are supposed to be compared need to be made at the same moment which is practically infeasible, especially when there are several competitive service instances.

To get a situation where the validation process retrieves exactly the best-fit service instance for validation at each moment considering call context and utility function, we developed a simulator which creates service instance measurements for a certain time period based on predefined behaviour profiles. The

 ISBN: 978-1-61208-387-2

implementation of this framework follows a periodic behaviour influenced by statistical random-based deviation. Currently, the periodic behaviour of the simulated Web services follows our initial measurements in [1] and considers: day/night time, weeks, months, work days and weekends. The random-based deviation is supposed to simulate unexpected incidences such as network traffic jams, high/low usage of a service's limited infrastructure. The random-based influence over a period was also evidenced in our real-world service tests [1]. At the moment, two NFPs are simulated which are response time and availability.

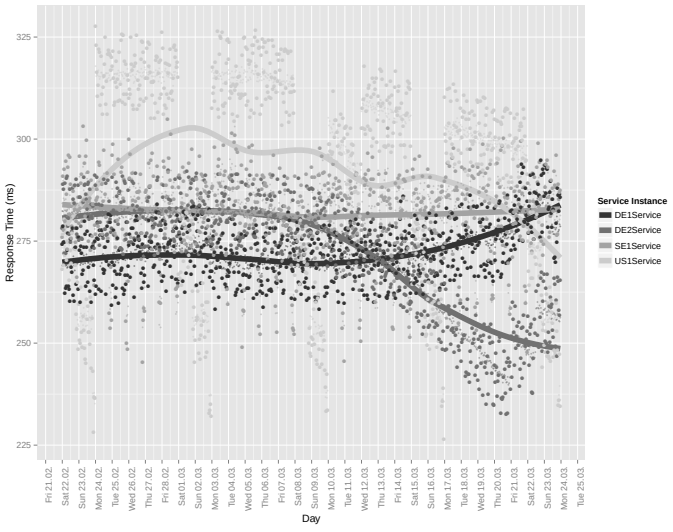


Figure 2. Overview about the simulated response time of four service instances and their trend over the whole period.

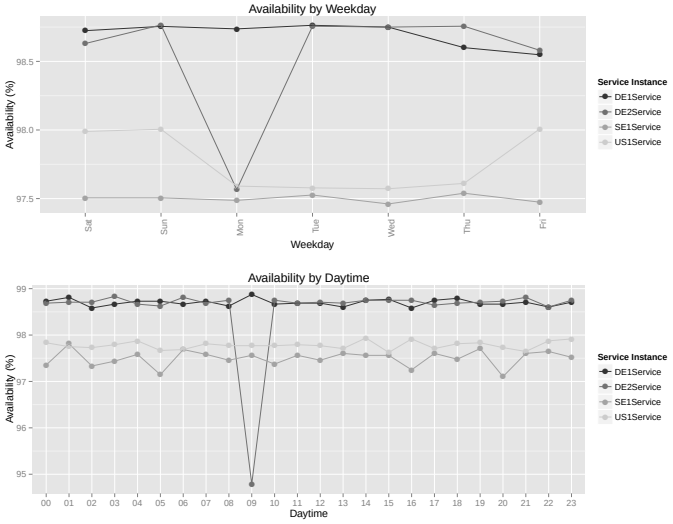


Figure 3. Overall periodic behaviour regarding availability of the simulated service instances with weekday and daytime aspects.

Figure 2 and Figure 3 depict an overview about the simulated NFPs. The simulated validation data set comprises a period of 30 days and has a total set of 460,800 records (40 records/hour $\times$ 24 hours/day $\times$ 30 day $\times$ 16 unique clients). The records contain information about day, time, response time in millisecond and availability (Boolean). Within the simulation, between each record there is a time interval of 90 seconds. Figure 2 shows in a condensed form the response time of all services instances within the whole period. Note that the line is only the trend. Within the recommendation process, the actual best-fit service instance at each time is important and not the averaged value of each service instance. The line is therefore only a visual orientation for us to determine the concept drift of each service instance within the period (e. g., DE2Service). Figure 3 shows the statistical value of availability with a focus on weekday and daytime periods.

B. Evaluation criteria

For the evaluation of the appropriateness of machine learning algorithms and their implementation in frameworks, each framework has to recommend the best-fit service instance based on a user's utility function and call context, i. e., the instance with the highest utility value. Implemented in our foreground-/background-model scenario, the machine learning algorithms have to estimate the expected NFP behaviour for all NFPs and all service instances. As described above, this results in $n \times m$ models (while n is the amount of considered NFPs and m is the amount of service instances). Since the number of modes can increase significantly, the recommendation table of the foreground model is updated asynchronously by the background model. In the background model, NFPs are learned continuously using the incremental learning functionality of the machine learning methods. However, although the expected NFPs per service instance are learned online, the utility values for each utility function and call context, and therefore the determination of the best-fit service instance, are calculated asynchronously in time- and count-based intervals. Besides the fact whether the actual best-fit service instance was recommended at each time, we also focus on how good the pre-calculated utility value was. For this, we took two indicators: *TOP1 accuracy* gives evidence on the recommendation quality as percentage of the correctly determined best-fit services with respect to the actual best-fit services instances, whereas the *TOP2 accuracy* shows the percentage of the determined best-fit instance within the actual top-two best-fit service instances.

C. Evaluation environment and method

For the execution of the evaluation, we used a test machine running Linux (Ubuntu 12.04 LTS) as its operating system, equipped with an i5-3340M CPU @ 2.70GHz x 4 (64 bit) and 15.6 GiB RAM.

The evaluation focuses on the overall recommendation. Recall that the overall process is split into the learning of the actual expected NFPs in a certain call context, for which we employ machine learning frameworks, and the calculation of the utility values according to service consumers' individual utility functions (preferences), from which the best-fit service can be determined. The machine learning frameworks learn each incoming record online, while the pre-calculation and determination part is conducted in intervals.

The evaluation is supposed to evaluate the accuracy within the period of examination at each point in time. Therefore, the data is not split into a training and validation set. In fact, with each learning interval, which also contains the update of the foreground table, the recommendation entry for a call

 ISBN: 978-1-61208-387-2

context and utility function is validated with the actual best-fit service for the upcoming service call. Hence, at each time, with the previously learned records, the recommendation quality for future recommendation requests in the same call context and with the same utility function is measured. The idea is that change in general can be evaluated.

D. Evaluation results

As written above, the overall learning for the recommendation of the best-fit service instance is split into the learning of the expected NFPs, for which we employ the machine learning methods/frameworks, and the determination of the best-fit instance based on pre-calculation. The first part is done continuously, while the second part is done in intervals. For the overall evaluation, we conducted two rounds, one with an interval of 10 records and one with 100 records. Table IV shows the results for 100 records, since there were only minor differences between both rounds; however, the 10 records round achieved slightly better results. As shown, the results of the FIMT-DD achieved around 70 % of correct predictions (with 10 record intervals, we achieved over 70 %). Note that the calculated utility ranges from 0–100. Comparing all methods, there is not much difference between Naïve Bayes and Hoeffding Tree. The FIMT-DD shows very good results. It has the highest update rate of the foreground table, which is an indication that it reacts quicker and more fine-grained on change than the other methods.

TABLE IV. EVALUATION RESULTS OF THE MACHINE LEARNING METHODS NAÏVE BAYES, HOEFFDING TREE AND FIMT-DD WITHIN THE OPTIMISED SERVICE SELECTION/RECOMMENDATION

	Naïve Bayes	Hoeffding Tree	FIMT-DD
TOP1 Accuracy (in %)	58.634	59.837	69.287
TOP2 Accuracy (in %)	90.163	90.421	93.471
Mean Absolute Error (Utility)	1.656	1.660	1.049
Recommend. Table Updates	659	647	1.189

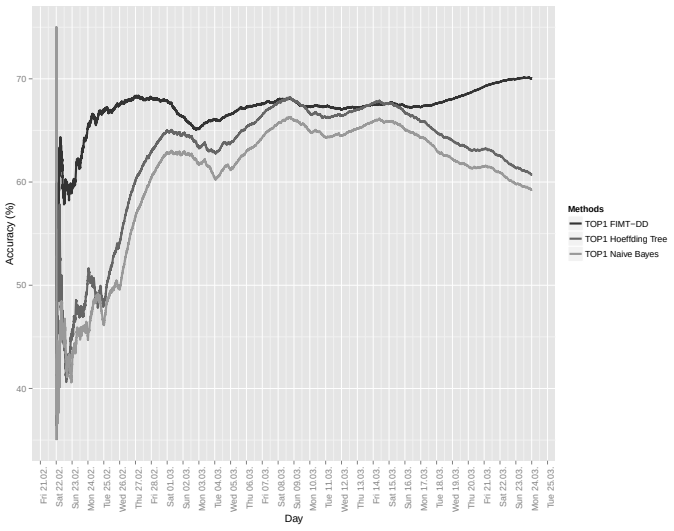


Figure 4. Service recommendation accuracy of the FIMT-DD, Hoeffding Tree and Naïve Bayes algorithm in the course of time.

The cold start problem applies to service recommendation, which means that good recommendation results are also supposed to be achieved with a small set of records at the beginning. Depicted in Figure 4, we can see that for the TOP 1 indicator in the overall recommendation process, the FIMT-DD quickly achieves a high accuracy. The drift detection of the FIMT-DD seems to work at the end of the period where some service instances change their performance behaviour (see Figure 2).

Our recommendation approach is supposed to be scalable. Table V shows the processing time of the overall process (NFP learning, utility pre-calculation, best-fit determination and update of the foreground table) for incoming measurement records. As we can see, a single record is processed in less than three milliseconds by a total of 460 thousand records. For all machine learning methods, we used the MOA framework. The figures show that, in term of learning overhead, there is not much difference between the methods. However, comparing the figures, Naïve Bayes would be able to process approximately 900 records more in an hour than the FIMT-DD. Comparing the numbers of the 100 record and the 10 record intervals (factor 8.8), it reveals that since the NFPs are learned continuously for every incoming record, the time-consuming part is related to the pre-calculation and best-fit determination. Although this is also due to the fact that these figures also include the evaluation steps (calculation of the actual NFPs, calculation of the actual utility values and determination of the actual best-fit instance), there is a high optimisation potential within the pre-calculation/-determination steps such as in-memory databases instead of hard disk databases. However, since these steps are asynchronous, they do not harm the recommendation process and are still good enough for background processing.

TABLE V. TIME FOR PROCESSING A SINGLE INSTANCE IN MILLISECONDS OF THE MACHINE LEARNING METHODS NAÏVE BAYES, HOEFFDING TREE AND FIMT-DD

Processing per record (ms)	Naïve Bayes	Hoeffding Tree	FIMT-DD
100 record intervals	2.602	2.614	2.621
10 record intervals	22.997	23.080	23.129

Figure 5 reveals more insight in the accuracy measure. The figure shows the degree of accuracy of the utility prediction. Once again, the best-fit service instance is the instance with the highest utility value regarding a service consumer's individual preferences (utility function). So, the closer the prediction towards the actual utility value is, the better the method. Comparing the method's charts, we see that for Naïve Bayes and Hoeffding Tree the predicted utility values at each time are both quite similar and do not reflect the curve of the actual values. In contrast, the chart for FIMT-DD depicts that the prediction is very close to the actual values. The intercepts of the curves show, that FIMT-DD does cope with change rapidly. In all cases, intercepts – which denote a change in the best-fit ranking – are also reflected in the prediction quite accurately.

For the evaluation of service recommendation in general, the actual utility gain is an important measure. Since the selection of service instances are based on several NFPs, the utility value as a basis for the individual preferences is an appropriate measure to benchmark service recommendation. In Table VI, the average experienced utility value after the service recommendation based on the FIMT-DD algorithm

 ISBN: 978-1-61208-387-2

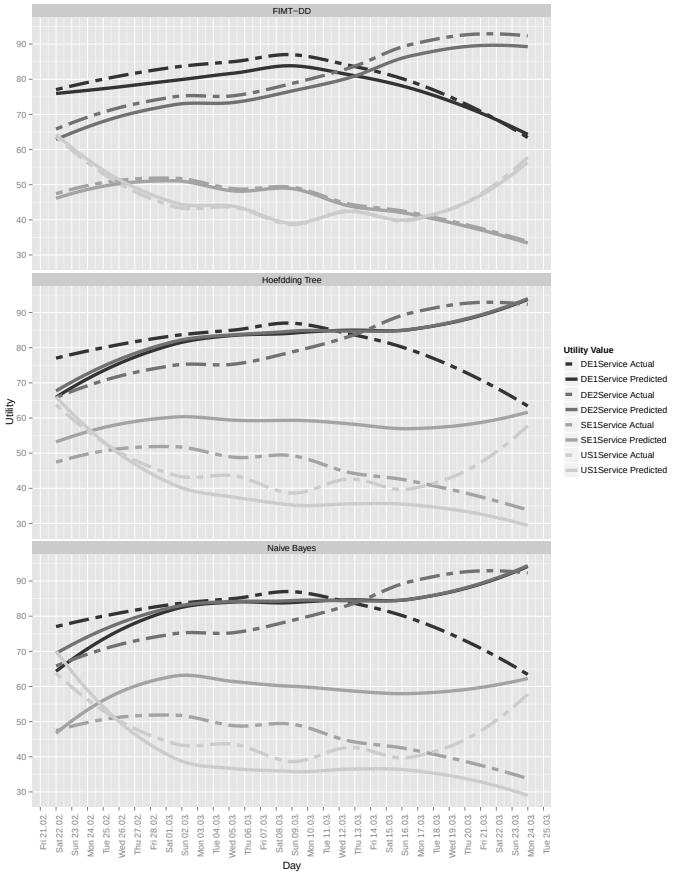


Figure 5. Detailed overview about the predicted and actual utility values.

TABLE VI. UTILITY GAIN WITH SERVICE RECOMMENDATION USING THE FIMT-DD ALGORITHM IN COMPARISON

After selecting ...	Average experienced utility value	FIMT-DD comparison in per cent
the FIMT-DD recommended instance	86.79	100.0 %
the perpetual best instance at each time	91.86	94.5 %
the perpetual worst instance at each time	29.22	297.0 %
the statistically best instance statically	81.96	105.9 %
an instance randomly	64.08	135.4 %

is compared with other scenarios. The table reveals good results. As written above, within this evaluation scenario, the overall best and worst services can be determined at each time. Once again, such comparisons are only possible within such a scenario; this is not possible in reality. Comparing the figures, we see that the FIMT-DD-based recommendation is able to achieve 94.5 % of the maximum achievable utility value. It is 35.4 % better than a random selection approach and even 5.9 % better than the statistically best service instance when statically using it. Note, that statically choosing the statistically best service instance is also a kind of learning.

VI. Related work

In [1], we introduced the overall concept of how knowledge can benefit service selection/recommendation in general. In that work, we presented the framework on an abstract level and introduced the recommendation component as a black box. In this work, however, we highlighted exactly this recommendation component and demonstrated the application of a concrete machine learning approach and how Service Level Achievements can be turned into knowledge for the benefit of a consumer-centric optimised service recommendation. In [1], we introduced a multi-stage selection with multi-stage dependencies of (compound) services. In this work, however, our focus was set on the general, appropriate and feasible approach of employing machine learning methods within service recommendation.

Further details about the evaluation can be read in [22]. This initial approach focuses mainly on the evaluation of machine learning algorithms which are appropriate for service recommendation. However, this work does not focus on some major characteristics of a service market such as the granularity of contexts and preferences (utility functions) and the availability of measurement data according to the actual usage of services. Furthermore, service recommendation also influences the behaviour of NFPs of service instances. For instance, best-fit service instances are more often consumed, which might affect response time. So, in case of limited service resources, these best-fit instances' NFPs can change for the worse.

Collaborative filtering (CF) approaches for service recommendation also focus on the exploitation of shared knowledge about services in order to recommend services to similar consumers before the actual consumption on an automated basis [23][24][25][26]. The major drawback of CF is that consumer-related preference similarities have to be found beforehand. With our call context and utility function approach, new consumers can already benefit from existing knowledge. CF approaches also do not take into account that consumers can have different optimisation goals or preferences and only some approaches [24][25] consider differences between consumers regarding their context. In [27], the authors tackle the lack of consideration of a consumer's preferences and interests; however, they do not take consumer context into account. The authors of [28] describe an approach to tackle the mentioned cold-start problem within CF.

Some work focuses on the prediction of NFPs for the detection of SLA violations such as [29]. This work mainly focuses on SLAs. In [1] (also cf. [26]), we argued that SLAs are not a good basis for service selection, considering the fact that service providers are profit-oriented, it is tempting to embellish their SLAs in order to be consumed. Also, as SLAs of consuming and providing services (e. g., compound services) depend on the SLAs of their sub-providers, deviations of actual non-functional characteristics and those specified in SLAs may propagate and spread even unintendedly and without the control of the providers. Furthermore, the performance experience at consumers' side is also dependent on a consumer's call context, which is also not reflected in SLAs.

We recently conducted a paper study about relevant NFPs during service selection, for which we processed over 4,000 conference papers in the SOC domain. In this study, we discovered only a very few papers considering more than one NFP during service selection/recommendation. Our approach considers the optimisation (recommendation) with several NFPs which is challenging due to the fact that the determination of the best-fit service instance according to service consumers' individual preferences result in a calculation task. Furthermore,

 ISBN: 978-1-61208-387-2

NFPs have different scales of measurement and different optimisation focuses. Therefore, the complete recommendation process cannot be left to machine learning alone.

VII. Conclusion

We evaluated appropriate machine learning frameworks for the recommendation of best-fit services according to consumer preferences and call contexts within a service market. Implemented in the recommendation component for a service broker, the FIMT-DD showed very good results in the evaluation. Furthermore, its implementation in the MOA framework fulfilled a high degree of our requirements in terms of recommendation accuracy, speed of learning, scalability and robustness. With a high degree of automation, Java integration and being open source, the MOA framework also fulfilled all software-related requirements.

This initial evaluation was based on a continuously generated simulation data set. This data set allowed us to compare the learning/recommendation results with the overall optimum and on a reproducible basis, which would have not been possible with real-world services. The characteristics of the simulated data set, however, are based on real-world services, which we observed in former studies. The records in the data set consisted of continuous data. Within our framework and in reality, this continuity of records can in general not be assumed, since the recommendation framework uses measurement of actual invoked service calls. However, service instances cannot be assumed to be invoked equally often. Furthermore, best-fit services are more often recommended and therefore consumed, while underdogs never get the chance to prove themselves. Also, often recommended service instances might be affected by the over-consumption if infrastructure resources are exceeded. In order to tackle this, our framework needs further strategies and tests of how big the impact of this potential drawback is in reality and how this can be solved. The implemented prototype of the overall recommendation process also needs further performance improvements for the pre-calculation and pre-determination part. Finally, the overall approach has to be validated to a real-world scenario with a realistic number of clients and services.

Nonetheless, the prototype and the employment of the FIMT-DD within the MOA framework build a good foundation for service brokers recommending a best-fit service towards service consumers' preferences. With the foreground/background architecture of the framework, the time-consuming overall learning process is decoupled from the actual time-critical recommendation process. So, our approach only produces minimal overhead to service times.

References

[1] J. Andersson, A. Heberle, J. Kirchner, and W. Löwe, "Service Level Achievements - Distributed knowledge for optimal service selection," in Ninth IEEE European Conference on Web Services (ECOWS), 2011, pp. 125–132.

[2] C. Sammut and M. Harries, "Concept drift," in Encyclopedia of Machine Learning, C. Sammut and G. Webb, Eds. Springer US, 2010, pp. 202–205. [Online]. Available: http://dx.doi.org/10.1007/978-0-387-30164-8_153

[3] C. Shearer, "The CRISP-DM model: The new blueprint for data mining," Journal of Data Warehousing, vol. 5, no. 4, 2000, pp. 13–22.

[4] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," Informatica, no. 31, 2007, pp. 249–268.

[5] J. Han and M. Kamber, Data Mining: Concepts and Techniques, 2nd ed. Elsevier, Morgan Kaufmann, 2006.

[6] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2001, pp. 97–106.

[7] E. Ikonomovska, J. Gama, and S. Džeroski, "Learning model trees from evolving data streams," Data Mining and Knowledge Discovery, vol. 23, no. 1, 2011, pp. 128–168.

[8] E. J. Keogh and M. J. Pazzani, "Learning augmented bayesian classifiers: A comparison of distribution-based and classification-based approaches," 1999. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.55.7726

[9] Weka, "Weka Javadoc – Hoeffding Tree," date of retrieval: 25 Oct 2014; http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/HoeffdingTree.html.

[10] KDnuggets.com – Data Mining Community's Top Resource for Data Mining and Analytics Software, "What analytics, data mining, data science software/tools you used in the past 12 months for a real project poll," June 2014, date of retrieval: 22 Oct 2014; http://www.kdnuggets.com/polls/2014/analytics-data-mining-data-science-software-used.html.

[11] "RapidMiner," https://rapidminer.com/, http://sourceforge.net/projects/rapidminer/.

[12] "The R project for Statistical Computing," http://www.r-project.org/.

[13] "Apache Mahout," http://mahout.apache.org/.

[14] "Apache Spark," http://spark.apache.org/.

[15] "KNIME," http://www.knime.org/.

[16] "The SHOGUN Machine Learning Toolbox," http://www.shogun-toolbox.org/.

[17] "Shark machine learning library," http://image.diku.dk/shark/sphinx_pages/build/html/index.html.

[18] "scikit-learn – Machine Learning in Python," http://scikit-learn.org/.

[19] "Vowpal Wabbit (Fast Learning)," http://hunch.net/~vw/.

[20] Machine Learning Group at the University of Waikato, "Weka – Data mining with open source machine learning software in Java," http://www.cs.waikato.ac.nz/ml/weka/.

[21] University of Waikato, "MOA Massive Online Analysis," http://moa.cms.waikato.ac.nz/.

[22] P. Karg, "Evaluation and Implementation of Machine Learning Methods for an Optimized Web Service Selection in a Future Service Market," Master's thesis, Karlsruhe University of Applied Sciences/Linnaeus University, Germany/Schweden, 2014.

[23] Z. Zheng, H. Ma, M. Lyu, and I. King, "QoS-aware Web service recommendation by collaborative filtering," Services Computing, IEEE Transactions on, vol. 4, no. 2, 2011, pp. 140–152.

[24] M. Tang, Y. Jiang, J. Liu, and X. Liu, "Location-aware collaborative filtering for QoS-based service recommendation," in Web Services (ICWS), IEEE 19th International Conference on, 2012, pp. 202–209.

[25] L. Kuang, Y. Xia, and Y. Mao, "Personalized services recommendation based on context-aware QoS prediction," in Web Services (ICWS), IEEE 19th International Conference on, 2012, pp. 400–406.

[26] R. Yang, Q. Chen, L. Qi, and W. Dou, "A QoS evaluation method for personalized service requests," in Web Information Systems and Mining, ser. Lecture Notes in Computer Science, vol. 6988. Springer Heidelberg, 2011, pp. 393–402.

[27] G. Kang, J. Liu, M. Tang, X. Liu, B. Cao, and Y. Xu, "AWSR: Active Web service recommendation based on usage history," in Web Services (ICWS), IEEE 19th International Conference on, 2012, pp. 186–193.

[28] Q. Yu, "Decision tree learning from incomplete QoS to bootstrap service recommendation," in Web Services (ICWS), IEEE 19th International Conference on, 2012, pp. 194–201.

[29] P. Leitner, B. Wetzstein, F. Rosenberg, A. Michlmayr, S. Dustdar, and F. Leymann, "Runtime prediction of service level agreement violations for composite services," in Service-Oriented Computing. ICSOC/ServiceWave 2009 Workshops, 2010, pp. 176–186.

 ISBN: 978-1-61208-387-2

Towards a Compiler for Business Processes — A Research Agenda

Thomas M. Prinz, Thomas S. Heinze,
and Wolfram Amme

Chair of Software Technology, Friedrich Schiller University
Jena, Germany
Email: {Thomas.Prinz, T.Heinze,
Wolfram.Amme}@uni-jena.de

Johannes Kretzschmar
and Clemens Beckstein

Artificial Intelligence Group, Friedrich Schiller University
Jena, Germany
Email: {Johannes.Kretzschmar,
Clemens.Beckstein}@uni-jena.de

***Abstract*—Business process management (BPM) and service-oriented architectures (SOA) promise the development, application, maintenance, and improvement of business processes, i.e., service compositions, as it is done in software engineering. However, BPM is currently more similar to an unfinished patchwork and an overall system supporting BPM is missing since it requires a unified execution engine (a virtual machine), a common intermediate representation, and eventually a compiler. In this paper, we motivate the construction of such a system for BPM and propose an approach including the mentioned sub systems. Additionally, we show the gaps in current approaches and why some techniques are not yet fully applicable. We encourage that system with state-of-the-art approaches and our own ideas of BPM, compiler construction, and artificial intelligence. Such a system finally will encourage processes for small and medium-sized enterprises and for SOA applications.**

Keywords–Business Process Management; Compiler; Intermediate Representation; Planning; Service-oriented Architecture.

I. INTRODUCTION

It seems that time has come for the efficient and successful application of business process management (BPM) and service-oriented architectures (SOA). There are promising approaches and techniques for each step of the BPM life cycle [1], i.e., (1) requirements analysis, (2) design, (3) implementation, (4) verification and testing, and (5) ongoing improvements. However, Koehler et al. have already emphasized gaps in the BPM life cycle (especially the missing automation of process translations into executable processes) which hinder companies in exploiting the benefits of BPM [2]. Therefore, BPM is more similar to an unfinished patchwork and a unified system seriously supporting BPM is needed.

In this paper, we argue for a compiler for business processes, i.e., service compositions. Koehler et al. have already argued for a compiler for business IT-systems and provide interesting approaches and ideas. Their compiler follows a top-down approach. However, the implementation of a compiler for business processes needs both: A well-defined business process modeling language as input and a machine that executes a process for the output. For example, the programming language Java would not have been so successful if it had not used its own virtual machine abstracting from the real physical design.

A virtual machine for business processes is similar to a unified and sufficient intermediate representation (IR) (like a bytecode for processes). Current process description languages like the Business Process Model and Notation 2.0 (BPMN) [3] and Event-driven Process Chains (EPCs) [4] are primarily designed for high level descriptions of business processes rather than for technical implementations. Although we do *not* want to translate abstract processes into executable ones (since there is the need for developers supporting that transformation), a well-defined and common technical basis for the definition of usable constructs and expressions is needed to guarantee such a transformation without later risking high additional effort.

As the IR is *not* suitable for the development of processes in general (like machine code or Java Bytecode for programs), it is necessary to provide a more high-level but IR-conform processing language (like a subset of BPMN and EPCs). Therefore, that language has to be automatically transformable into the IR. Such a transformation can basically be done by a compiler. The compilation process should provide powerful tools and analyses with useful failure and diagnostic information about the process for the developer. These kinds of information are currently undetailed and so new algorithms have to fill the gap. Additionally, a tool can handle such information to provide several options for (semi-) automatic error handling. Within a dialog between the developer and the system, the developer can choose the best fitting solution.

Although there are already approaches considering parts of the mentioned system, little attention has been paid to their interaction. We identified four important subsystems for a general overall BPM system: *(1)* a simple and unified process engine, i.e., a (virtual) machine, *(2)* a unified IR, *(3)* a verifying compiler translating business processes into that IR, and *(4)* an error handling which provides correction proposals being applied to processes.

In this paper, we consider the state-of-the-art of BPM systems (Section II) in short. We describe a system consisting of a compiler, an IR, and a process engine with regard to state-of-the-art approaches, solutions (Section III) and provide own research approaches to finally enable the implementation of such a new approach for a system. Section IV summarizes and concludes the paper.

II. STATE OF THE ART

There are many tools providing BPMN for BPM e.g., Activiti BPM Platform [5], Redhat jBPM [6], IBM WebSphere [7], AristaFlow BPM Suite [8], and BonitaBPM [9]. These tools allow for the development, simulation, and execution of business processes. Furthermore, they have additional features supporting parts of the BPM lifecycle.

 ISBN: 978-1-61208-387-2

However, most of those tools use different subsets of and (intermediate) representations for BPMN such that processes are not interchangeable between tools without additional effort. Considering that fact for the programming language Java for example: It would be strange if there would be a variety of virtual machines for Java, for each accepting a different subset of Java bytecode instructions. *Apromore* [10] is an advanced process model repository based on a common intermediate representation (canonical representation) to handle different process model languages. However, although the repository benefits from that representation, common parts of process modeling languages like exceptions, exception handling, signals, transitions, etc. are not accurately representable.

Our approach calls for the implementation of a core virtual machine for business process modeling languages that is extensible by additional tools. Such a core virtual machine asks for a greatest possible subset of process elements being accepted by the machine (the IR) and also asks for a compiler, which constructs the IR form for current process modeling languages (e.g., *Web Services Business Process Execution Language 2.0* (BPEL) [11], *Yet Another Workflow Language* (YAWL) [12], EPCs or BPMN). The compilation process is not always straightforward as some modeling languages (e.g., BPMN) only provide subsets of executable process models (e.g., BPMN Process Execution Conformance [3] for BPMN).

Compilers for business processes are rare in existing tools. Most of them take a process as it is and interpret it stepwise. Sometimes, however, additional information is needed for the execution of a process which can be derived from a compiler, e.g., data types, soundness or reference safety. Additionally, most business process modeling language's output formats are not suitable for fast and efficient analyses and compilers therefore have to create a more compact format.

For this purpose, our approach calls for the implementation of a compiler that transforms and analyzes a process in an intermediate and interchangeable representation. The most common used intermediate representations for business processes are specification conform exchange formats or BPEL. BPEL has the great advantage to be a block-based language. That, however, is the largest problem for simply transforming processes of graph-based languages (like BPMN) to BPEL [13][14]. Furthermore, BPEL was designed to orchestrate different web services and not to directly execute tasks.

Our approach relies on an intermediate representation for business processes. That representation should allow and outperform existing analyses for the verification of pre-defined process properties since process validation is very expensive by currently supported BPM tools. State-of-the-art research considers those verification mechanisms. For example, in previous work [15], we have focused on compiler-based mechanisms for finding deadlocks and missing synchronizations. These techniques are so efficient that we were able to perform the analyses after each modification of a process model and to give detailed diagnostic information as shown by our tool implementation [16].

Besides these compiler-based mechanisms, we argue for semantic analyses of processes by artificial intelligence (AI) planning methods. These methods rely on semantic descriptions of process-activities. Semantic descriptions are already widely used in the field of service-oriented architectures through service description standards, like *Web Service Modeling Language* (WSML) [17] and *Web Ontology Language for Semantic Web Services* (OWL-S) [18]. In contrast to previous service description standards, these languages allow the specification of requirements and impacts of a service regarding a descriptive domain model. With such descriptions, an AI planner is able to goal-oriented generate an ordered set of services, which can be executed as a BPEL-like service composition by workflow engines [19][20]. Because of the fast growing complexity of planning problems, there are usually assumptions of the domain models concerning time, execution, observability and influence aspects [21]. Therefore, AI planning for workflow generation is only practical in particular use-cases with simple workflow models. Our approach focuses on cheaper methods for evaluating a process, which are part of planning algorithms. Thereby, AI planning could be used also for more comprehensive workflow descriptions and could enhance a comprehensive semantic analysis.

III. Compiler-Engine Architecture for Business Processes

In the following, we propose a new BPM architecture. For this purpose, we explain the overall system first and subsequently describe each subsystem in detail.

Figure 1 gives a structural overview of the complete system. The system has two sides inspired by Amme et al. [22]: a *producer* and a *consumer side*. The producer side is a compiler being adaptable to each business process development tool. It accepts an entire process in different process modeling languages, where a specific front-end containing a parser and a transformation exists for each language. The internal format is an IR, which allows for the application of semantic analyses whose output in turn can be used as input for an error handler, a coder, and an annotator.

The consumer side consists of an engine and virtual machine, respectively. It loads a compiled process and extracts the IR. Then, it executes the IR and performs dynamic semantic analyses and, in the case of an error, it provides an error handling which enables for a "rescue" of the running process.

The interface between the producer and consumer side is a *business process repository*. The producer side stores compiled processes within that repository whereas the consumer side can load them. Furthermore, error handling systems on both sides utilize the same repository for their analyses.

A. Producer Side

The producer side is divided into a *parser*, a *transformation*, an *IR*, *semantic analyses*, and an *error handling* as well as a *coder* and an *annotator*.

1) Parsing and Transformation: The parser's task is to structurally analyze and verify the entire process against its description language, i.e., a conformance check. Afterwards, the transformer translates that process into an IR. For this purpose, the front-end, consisting of the parser and the transformer, depends on the process language.

Two approaches can be distinguished: (1) Defining a mapping from one language to the other [23][24][25] or (2) creating a parse tree (process structure tree, PST) which is then used for a translation [26]. Both approaches have their roots in compiler theory. However, since the PST is similar

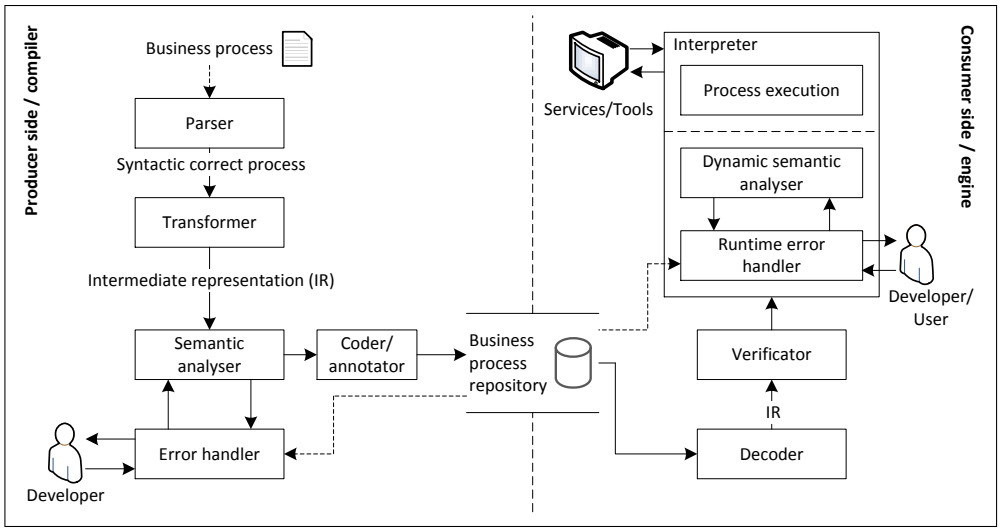


Figure 1. System overview containing a compiler and an engine.

to an abstract syntax tree (AST), it provides more structural information and therefore we prefer the PST as a mapping is still possible at a later time.

The inclusion of instructions and variables of the business process constitutes the major problem during the parsing and transformation. Process instructions can be translated as shown by Amme et al. [27], using the Concurrent Static Single Assignment Form (CSSA form) [28]. However, the creation of CSSA form is currently only suitable for structured graphs. Most business processes are unstructured, so we have to define a transformation for those processes and their instructions.

2) Intermediate Representation: A common IR must cover (almost) all constructs and instructions of currently popular process languages. Furthermore, it must provide and support efficient techniques for detailed semantic analyses. Currently, Petri nets [29] and workflow graphs [30] are commonly used to represent language-independent and analyzable processes. Since workflow graphs have Petri net semantics but provide more structural information, workflow graphs should be used as they are very similar to (concurrent) control flow graphs of compiler theory [28].

In previous work, we have defined an *extended workflow graph* (eWFG) based on CSSA form [27][31]. In the next steps, we plan to extend those eWFGs with advanced language constructs like OR-joins, events, signals, transactions, exception handling, and roles.

3) Semantic Analyses: The task of the semantic analyser is to verify the IR against properties and to restructure the IR, e.g., for the encoding into a mobile format. Semantic analyses consist of structural, context-sensitive, content-related, and goal-oriented analyses. Structural analyses consider only the control flow of the process without regarding instructions. Context-sensitive and content-related analyses include those instructions. Goal-oriented analyses require additional information from the developer in which the developer describes the goals of the process.

Traditionally, process analyses focus on structural process properties, e.g., soundness [29]. The soundness property guarantees the absence of deadlocks in non-deterministic processes. We have developed a new approach to detect such deadlocks in conjunction with all the necessary information to repair them [15][16]. That approach has to be extended for the IR's additional language constructs. Although structural analyses consider only the control flow, they are suitable pre-processors for advanced analyses as they reduce the solution and failure space, c.f. SESE decomposition [32]. One has to show that it is possible to find further structural information, e.g., nodes with possible race conditions.

Since structural analyses can result in false-positive and false-negative analysis results [33], the consideration of data and instructions is essential to seriously support a process developer. However, less attention has been paid to process data and instructions in the literature. Sidorova [33] and our previous work [27][31] describe ways to include data in semantic analyses. Approaches of compiler theory can improve context-sensitive and content-related analyses by deriving predicate-logic expressions, by using path-sensitive data-flow analyses [34], by using instruction ordering techniques [35], or by using demand-driven approaches with backward traverses [36]. Especially, (C)SSA form is predestined for state space techniques since each variable is defined once and therefore the state space of a variable can be directly attached to it. All those approaches have to be reconsidered in the context of process analysis.

Goal-oriented analyses use approaches of AI planning. For this purpose, one has to define a precise *process domain* (properties which are in the focus of the process) that is used by the developer to describe the changes in this domain and the goal of the process. Then, the reachability of the goal can be verified. Furthermore, analyses can make suggestions to complete a process with respect to its goals. Our major focus lies on the adaption of AI planning techniques for the context of processes.

4) Error Handling: The error handler can improve the IR process by error correction and restructuring. The results of the semantic analyses are visualized and explained to the process's developer, and the error handler derives proposals for correction which then can be applied. To this end, the process has to be decomposed in such a way that failures can

 ISBN: 978-1-61208-387-2

be corrected locally without side effects.

The application of methods for automatic correction is not in the main line of research in BPM, resulting in only a handful of related approaches [37][38][39]. Most approaches consider preconditions of tasks within the process and how they can lead to deadlocks. Automatic correction then means to introduce weak conditions to avoid such deadlocks. Other methods derive a more general model of the process and afterwards construct a new process representing that model.

We follow another approach, in which the desired behavior of the process can never be completly derived from the process since the intention of the process is only in the mind of the developer. In this case, error handling has to interact with the developer to identify the best fitting solution. For this, approaches of AI planning are considered, which use basic rules to generate good solution proposals.

5) Coder and Annotator: If the process is correct, such that all verifiable properties hold, the coder and annotator enrich the IR with the results of the semantic analyses and afterwards possibly encode it into a mobile format. Both, the annotation of business processes as well as a mobile format, have not been in the focus of research. We want to consider approaches of compiler theory in which the annotation of programs (e.g., Java bytecode) or mobile formats are well understood. Our mobile format *SafeTSA*, for example, provides approaches to efficiently transform programs with a tree structure (AST) and (C)SSA form [40]. As mentioned before, each process can be represented by its PST and therefore it is possible to generate a mobile format, similar to SafeTSA, for business processes. In summary, we have to generate a SafeTSA conform mobile format for business processes to encourage their exchange.

B. Business Process Repository

After the process becomes executable, has been verified and possibly encoded into a mobile format, it can be stored within a business process repository. Thereby, the repository should provide features to find fitting process interaction partners by the use of its (semantic) annotations. For this purpose, promising proposals for methods [41][42] exist which can be extended and applied to the development of our system.

C. Consumer Side

As most steps of the consumer side are similar to those of the compiler, we want to only briefly discuss the engine in the following. On the consumer side, the business process is taken from the repository and is transformed back into the common IR (*decoder*). During that transformation, the process has to be verified once again (*verificator*) with the help of the annotated information, to guarantee a flawless transfer. As the results of the semantic analyses on the producer side have been annotated to the IR, that can be done fast.

Subsequently, an *interpreter* starts executing the process, for which Petri net-based or straight-forward (and almost sequential) approaches have been described in practice. However, we prefer to use a virtual machine as process engine since this approach is sufficient for Java. We imagine a main control unit loading the process and monitoring its execution. Furthermore, it starts a subprocess for each new control flow (e.g., after parallel branches) such that each control flow is handled by a seperate control unit with its own local memory, arithmetic logic unit, and input/ouput unit. The resulting architecture provides full parallelism and is able to execute sub processes on hetergenous subsystems (e.g., in a network). It can request services and other tools to support (user) tasks. The main control unit performs *dynamic semantic analyses* during the execution of a process to find runtime errors as early as possible. Since those analyses have full runtime information, indications of failure situations can be detected *before* they occur. A user has then the possibility to correct those failures with the help of a *runtime error handler*. Both, the dynamic semantics analyses and the runtime error handling mechanism, are based on the process's annotations, the analyser, and the error handling techniques of the compiler with the advantage of having full information about actual variable assignments.

IV. Conclusion and Future Work

In this paper, we have motivated the construction of a compiler, a common IR, a virtual machine, and detailed failure analyses for business process management. We have proposed a system which allows for the compilation, storing, and execution of processes based upon its own IR. That system uses state-of-the-art approaches and ideas from business process management, compiler construction, and artificial intelligence. There already exist approaches to realize such a system.

For the future, we recommend to develop and evaluate a compiler-based development and runtime environment for business processes without the consideration of data. Those environments should then be extended for processes with data. With this in mind, there are four main aspects being sequential considered: (1) an intermediate representation based on extended workflow graphs with regard to a virtual machine and its execution semantics, (2) process properties with static and dynamic analyses for their verification, (3) an error visualization, handling, and correction, and (4) process annotations for an efficient information transfer. The major goal is to show that such a business process management system is possible, applicable, and efficient. We are sure that such a system is the future of process development and will support processes for small and medium-sized enterprises and for the development of SOA applications.

References

[1] W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, "Business process management: A survey," in Business Process Management, International Conference, BPM 2003, Eindhoven, The Netherlands, June 26-27, 2003, Proceedings, ser. Lecture Notes in Computer Science, W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, Eds., vol. 2678. Springer, 2003, pp. 1–12.

[2] J. Koehler, T. Gschwind, J. M. Küster, H. Völzer, and O. Zimmermann, "Towards a compiler for business-it systems - A vision statement complemented with a research agenda," in Software Engineering Techniques - Third IFIP TC 2 Central and East European Conference, CEE-SET 2008, Brno, Czech Republic, October 13-15, 2008, Revised Selected Papers, ser. Lecture Notes in Computer Science, Z. Huzar, R. Kocí, B. Meyer, B. Walter, and J. Zendulka, Eds., vol. 4980. Springer, 2008, pp. 1–19.

[3] OMG, "Business Process Model and Notation 2.0," formal/2011-01-03, 2011, last access: February 18, 2015.

[4] A. Scheer, "Architecture of integrated information systems (ARIS)," in Information Infrastructure Systems for Manufacturing, Proceedings of the JSPE/IFIP TC5/WG5.3 Workshop on the Design of Information Infrastructure Systems for Manufacturing, DIISM '93, Tokyo, Japan, 8-10 November, 1993, ser. IFIP Transactions, H. Yoshikawa and J. Goossenaerts, Eds., vol. B-14. North-Holland, 1993, pp. 85–99.

 ISBN: 978-1-61208-387-2

[5] Alfresco, "Activiti BPM Platform," http://activiti.org/, last access: February 18, 2015.

[6] redhat, "jBPM - Open Source Business Process Management - Process engine," http://www.jbpm.org/, last access: February 18, 2015.

[7] IBM, "IBM WebSphere software - United States," http://www.ibm.com/software/websphere/, last access: February 18, 2015.

[8] AristaFlow, "AristaFlow - Aristaflow BPM Suite fr den BPM-Roundtrip in einem einzigen Werkzeug," http://www.aristaflow.com/bpmsuite.html, last access: February 18, 2015.

[9] Bonitasoft, "Bonitasoft - Open Source Workflow & BPM software," http://www.bonitasoft.com/, last access: February 18, 2015.

[10] M. L. Rosa, H. A. Reijers, W. M. P. van der Aalst, R. M. Dijkman, J. Mendling, M. Dumas, and L. García-Bañuelos, "APROMORE: an advanced process model repository," Expert Syst. Appl., vol. 38, no. 6, 2011, pp. 7029–7040.

[11] OASIS, Web Services Business Process Execution Language Version 2.0, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html, OASIS Std. 2, Rev. 0, apr 2007, last access: February 18, 2015.

[12] W. M. P. van der Aalst and A. H. M. ter Hofstede, "Yawl: yet another workflow language," Inf. Syst., vol. 30, no. 4, 2005, pp. 245–275.

[13] M. Weidlich, G. Decker, A. Großkopf, and M. Weske, "BPEL to BPMN: the myth of a straight-forward mapping," in On the Move to Meaningful Internet Systems: OTM 2008, OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008, Monterrey, Mexico, November 9-14, 2008, Proceedings, Part I, ser. Lecture Notes in Computer Science, R. Meersman and Z. Tari, Eds., vol. 5331. Springer, 2008, pp. 265–282.

[14] W. Zhao, R. Hauser, K. Bhattacharya, B. R. Bryant, and F. Cao, "Compiling business processes: untangling unstructured loops in irreducible flow graphs," IJWGS, vol. 2, no. 1, 2006, pp. 68–91.

[15] T. M. Prinz and W. Amme, "Practical compiler-based user support during the development of business processes," in Service-Oriented Computing - ICSOC 2013 Workshops - CCSA, CSB, PASCEB, SWESE, WESOA, and PhD Symposium, Berlin, Germany, December 2-5, 2013. Revised Selected Papers, ser. Lecture Notes in Computer Science, A. Lomuscio, S. Nepal, F. Patrizi, B. Benatallah, and I. Brandic, Eds., vol. 8377. Springer, 2013, pp. 40–53.

[16] T. M. Prinz, N. Spieß, and W. Amme, "A first step towards a compiler for business processes," in Compiler Construction - 23rd International Conference, CC 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings, ser. Lecture Notes in Computer Science, A. Cohen, Ed., vol. 8409. Springer, 2014, pp. 238–243.

[17] J. de Bruijn, H. Lausen, A. Polleres, and D. Fensel, "The web service modeling language WSML: an overview," in The Semantic Web: Research and Applications, 3rd European Semantic Web Conference, ESWC 2006, Budva, Montenegro, June 11-14, 2006, Proceedings, ser. Lecture Notes in Computer Science, Y. Sure and J. Domingue, Eds., vol. 4011. Springer, 2006, pp. 590–604.

[18] W3C, OWL Web Ontology Language for Services, http://www.w3.org/Submission/2004/07/, World Wide Web Consortium W3C Std. 1, Rev. 0, nov 2004, last access: February 18, 2015.

[19] F. Henni and B. Atmani, "Dynamic web service composition. use of case based reasoning and AI planning," in Proceedings of the 4th International conference on Web and Information Technologies, ICWIT 2012, Sidi Bel Abbes, Algeria, April 29-30, 2012, ser. CEUR Workshop Proceedings, M. Malki, S. Benbernou, S. M. Benslimane, and A. Lehireche, Eds., vol. 867. CEUR-WS.org, 2012, pp. 22–29.

[20] H. Nacer and D. Aïssani, "Semantic web services: Standards, applications, challenges and solutions," J. Network and Computer Applications, vol. 44, 2014, pp. 134–151.

[21] M. Ghallab, D. S. Nau, and P. Traverso, Automated planning - theory and practice. Elsevier, 2004.

[22] W. Amme, T. S. Heinze, and J. von Ronne, "Intermediate representations of mobile code," Informatica (Slovenia), vol. 32, no. 1, 2008, pp. 1–25.

[23] W. M. P. van der Aalst, "The application of petri nets to workflow management," Journal of Circuits, Systems, and Computers, vol. 8, no. 1, 1998, pp. 21–66.

[24] S. Hinz, K. Schmidt, and C. Stahl, "Transforming BPEL to petri nets," in Business Process Management, 3rd International Conference, BPM 2005, Nancy, France, September 5-8, 2005, Proceedings, W. M. P. van der Aalst, B. Benatallah, F. Casati, and F. Curbera, Eds., vol. 3649, 2005, pp. 220–235.

[25] N. Lohmann, "A feature-complete petri net semantics for WS-BPEL 2.0," in Web Services and Formal Methods, 4th International Workshop, WS-FM 2007, Brisbane, Australia, September 28-29, 2007. Proceedings, ser. Lecture Notes in Computer Science, M. Dumas and R. Heckel, Eds., vol. 4937. Springer, 2007, pp. 77–91.

[26] J. Vanhatalo, H. Völzer, and J. Koehler, "The refined process structure tree," Data Knowl. Eng., vol. 68, no. 9, 2009, pp. 793–818.

[27] W. Amme, A. Martens, and S. Moser, "Advanced verification of distributed ws-bpel business processes incorporating cssa-based data flow analysis," International Journal of Business Process Integration and Management, vol. 4, no. 1, 2009, pp. 47–59.

[28] J. Lee, S. P. Midkiff, and D. A. Padua, "Concurrent static single assignment form and constant propagation for explicitly parallel programs," in Languages and Compilers for Parallel Computing, 10th International Workshop, LCPC'97, Minneapolis, Minnesota, USA, August 7-9, 1997, Proceedings, ser. Lecture Notes in Computer Science, Z. Li, P. Yew, S. Chatterjee, C. Huang, P. Sadayappan, and D. C. Sehr, Eds., vol. 1366. Springer, 1997, pp. 114–130.

[29] W. M. P. van der Aalst, A. Hirnschall, and H. M. W. E. Verbeek, "An alternative way to analyze workflow graphs," in Advanced Information Systems Engineering, 14th International Conference, CAiSE 2002, Toronto, Canada, May 27-31, 2002, Proceedings, ser. Lecture Notes in Computer Science, A. B. Pidduck, J. Mylopoulos, C. C. Woo, and M. T. Özsu, Eds., vol. 2348. Springer, 2002, pp. 535–552.

[30] W. Sadiq and M. E. Orlowska, "Analyzing process models using graph reduction techniques," Inf. Syst., vol. 25, no. 2, 2000, pp. 117–134.

[31] T. S. Heinze, W. Amme, and S. Moser, "A restructuring method for WS-BPEL business processes based on extended workflow graphs," in Business Process Management, 7th International Conference, BPM 2009, Ulm, Germany, September 8-10, 2009. Proceedings, ser. Lecture Notes in Computer Science, U. Dayal, J. Eder, J. Koehler, and H. A. Reijers, Eds., vol. 5701. Springer, 2009, pp. 211–228.

[32] J. Vanhatalo, H. Völzer, and F. Leymann, "Faster and more focused control-flow analysis for business process models through SESE decomposition," in Service-Oriented Computing - ICSOC 2007, Fifth International Conference, Vienna, Austria, September 17-20, 2007, Proceedings, ser. Lecture Notes in Computer Science, B. J. Krämer, K. Lin, and P. Narasimhan, Eds., vol. 4749. Springer, 2007, pp. 43–55.

[33] N. Sidorova, C. Stahl, and N. Trcka, "Soundness verification for conceptual workflow nets with data: Early detection of errors with the most precision possible," Inf. Syst., vol. 36, no. 7, 2011, pp. 1026–1043.

[34] J. Fischer, R. Jhala, and R. Majumdar, "Joining dataflow with predicates," in Proceedings of the 10th European Software Engineering Conference held jointly with 13th ACM SIGSOFT International Symposium on Foundations of Software Engineering, 2005, Lisbon, Portugal, September 5-9, 2005, M. Wermelinger and H. Gall, Eds. ACM, 2005, pp. 227–236.

[35] E. Duesterwald and M. L. Soffa, "Concurrency analysis in the presence of procedures using a data-flow framework," in Symposium on Testing, Analysis, and Verification, 1991, pp. 36–48.

[36] K. Winter, C. Zhang, I. J. Hayes, N. Keynes, C. Cifuentes, and L. Li, "Path-sensitive data flow analysis simplified," in Formal Methods and Software Engineering - 15th International Conference on Formal Engineering Methods, ICFEM 2013, Queenstown, New Zealand, October 29 - November 1, 2013, Proceedings, ser. Lecture Notes in Computer Science, L. Groves and J. Sun, Eds., vol. 8144. Springer, 2013, pp. 415–430.

[37] M. Gambini, M. L. Rosa, S. Migliorini, and A. H. M. ter Hofstede, "Automated error correction of business process models," in Business Process Management - 9th International Conference, BPM 2011, Clermont-Ferrand, France, August 30 - September 2, 2011. Proceedings, ser. Lecture Notes in Computer Science, S. Rinderle-Ma, F. Toumani, and K. Wolf, Eds., vol. 6896. Springer, 2011, pp. 148–165.

 ISBN: 978-1-61208-387-2

[38] A. Awad, G. Decker, and N. Lohmann, “Diagnosing and repairing data anomalies in process models,” in Business Process Management Workshops, BPM 2009 International Workshops, Ulm, Germany, September 7, 2009. Revised Papers, ser. Lecture Notes in Business Information Processing, S. Rinderle-Ma, S. W. Sadiq, and F. Leymann, Eds., vol. 43. Springer, 2009, pp. 5–16.

[39] C. Wagner, “A data-centric approach to deadlock elimination in business processes,” in 3rd Central-European Workshop on Services and their Composition, Services und ihre Komposition, ZEUS 2011, Karlsruhe, Germany, February 21-22, 2011. Proceedings, ser. CEUR Workshop Proceedings, D. Eichhorn, A. Koschmider, and H. Zhang, Eds., vol. 705. CEUR-WS.org, 2011, pp. 104–111.

[40] W. Amme, N. Dalton, M. Franz, and J. von Ronne, “Safetsa: A type safe and referentially secure mobile-code representation based on static single assignment form,” in Proceedings of the 2001 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), Snowbird, Utah, USA, June 20-22, 2001, M. Burke and M. L. Soffa, Eds. ACM, 2001, pp. 137–147.

[41] F. Klan and B. König-Ries, “A user-centered methodology for the evaluation of (semantic) web service discovery and selection,” in 4th International Conference on Web Intelligence, Mining and Semantics (WIMS 14), WIMS ’14, Thessaloniki, Greece, June 2-4, 2014, R. Akerkar, N. Bassiliades, J. Davies, and V. Ermolayev, Eds. ACM, 2014, p. 18.

[42] H. Si and Y. Zhao, “A structured p2p-based approach to semantic web services publication and discovery,” JSW, vol. 9, no. 7, 2014, pp. 1930–1940.

 ISBN: 978-1-61208-387-2

An Approach for a Web-based Analysis Solution with MUSTANG

Mirco Josefiok* and David Korfkamp† and Jan Witt‡

OFFIS – Institute for Information Technology
Data Management and Data Analysis
Escherweg 2, 26121 Oldenburg
Email: †josefiok@offis.de, †korfkamp@offis.de, ‡witt@offis.de

***Abstract*—Analytical Information Systems (AIS) are comprehensive solutions for analyzing large data sets. The operation of an AIS usually requires an extensive infrastructure. Moreover, usually only specialist users are capable of performing analyses. In this paper, we present an approach for a web-based analysis solution which can be deployed either in a web-based environment or as an on-premise solution. We strongly emphasize self service capabilities by adding knowledge-based components in the form of an additional metadata layer.**

Keywords–web-based system; analysis information systems; self-service business intelligence.

I. INTRODUCTION

For coping with an increasing flow of information in companies and a growing complexity in planning business decisions, Data Warehouses (DWH) with Online Analytical Processing (OLAP) have been established. With these technologies and concepts, a company-wide provision of information should be guaranteed. Analytical Information Systems (AIS) are the logical bracket around the concepts and technologies DWH, OLAP, Data Mining and the respective analysis tools [1].

Development and operation of an infrastructure for processing, storing and analyzing large amounts of data requires substantial investments. To be able to react promptly and appropriately and to acquire new data whenever available, the necessary capacities must be maintained. Therefore, small and medium enterprises, as well as other entities whose core competencies are not information technology, do not use dedicated analysis platforms or business intelligence solutions commonly [2].

Analytical Information Systems (AIS) (see section II-B) usually suffer from some shortcomings when it comes to self-service of business users. Notably, typical business users are not capable of performing analyses by themselves, because the software solutions are too complex and lack the necessary guidance. Moreover, the underlying data model is often too complex to comprehend. Usually, no metadata is present, which might help business users to gain insights about structure and semantics of the underlying multidimensional data [3].

This raises the demand for a solution which is easily accessible, scales with an increasing amount of users but does not lack the analytical capabilities specialist users demand.

In this paper, we present a web-based approach for an analysis solution. Our approach is based on the MUSTANG (Multidimensional Statistical Data Analysis Engine) framework [4] and is developed within the WAIS project [5]. MUSTANG is used, for example, by epidemiological cancer registries (ECRs) in several German federal states in an instance called CARESS [6]. Whilst MUSTANG and its instances were originally developed as standalone applications, recent changes made it possible to deploy it as a service-based application in cloud environments. This results in the opportunity to make the analysis solution available to a broader audience, which is backed up by the new approach of the presented solution. With curated data, even more value can be added to our approach. Moreover, we introduced an additional metadata layer to foster self-service operations.

This paper is organized as follows. Section II introduces the foundations of our work, namely Software as a Service (SaaS), AIS and the MUSTANG framework. Section III analyzes the problem statement and introduces the architectural requirements for a service-based AIS. Section IV gives an overview of selected related works. Section V presents our approach and gives an overview of the developed prototype and its architecture. Section VI describes the evaluation and points out remaining research and development challenges. Finally, the paper concludes with Section VII.

II. FOUNDATIONS

Cloud computing is currently considered one of the most important topics for the information and communications technology (ICT) sector which has emerged in recent years. It already has a significant impact on how most IT-related projects are pursued, especially regarding the design and implementation of new software products. In addition, increasing innovation may be possible with the new deployment options available through cloud computing [7].

Several conceptual frameworks to describe and characterize cloud service offers exist. As they are developed by different groups and organizations, they differ in their intention, type of formulation, the level of description, and in terms of which key issues of cloud service evaluation are addressed. The American National Institute of Standards and Technology (NIST) provides a definition of cloud computing which is well accepted in industry and science [8]. A recent survey of different frameworks and reference models for the description of cloud offerings and taxonomy of cloud service offers is provided by Gudenkauf et al., with the aim to help overcome the skepticism of enterprises regarding cloud service offers by identifying their key factors [9].

 ISBN: 978-1-61208-387-2

A. Software as a Service (SaaS)

According to Babar and Chauhan [10] SaaS can be defined as follows: "The capability provided to the consumer [...] to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings." Our approach can be regarded as a special form of SaaS, in which analytical systems are deployed at a hosted location and accessed by potential users with regular internet connections.

For a potential user, this model has a wide range of benefits. Besides the benefits which result from the usage of cloud services, like cost alignment, cost reduction, streaming payment and compliance implementation, potential end users can focus on their core competencies and do not need to bother about building and maintaining a rather complex infrastructure for running an AIS.

B. Analytical Information Systems (AIS)

AIS have been established in science and industry for performing analyses on large data sets. AIS should support analytical activities by summarizing different concepts and technologies (DWH, OLAP, Data Mining, Business Intelligence solutions) and present them to the user due to a unified view. The DWH acts as central database in which all relevant data should be integrated via ETL processes (extract, transform, and load). With OLAP a multi-perspective view on the data can be enabled. Data Mining is an umbrella term for different methods in the field of data analysis. End users have access to those concepts and methods via business intelligence tools. AIS provide those concepts, methods and tools to end users and support them in the process of information retrieval and decision making [1].

C. Self-Service Business Intelligence

Self-service business intelligence (SSBI) can be defined as a BI environment, which empowers business users to perform complex analyses without needing the help of an expert [11]. This allows users to acquire information in a more timely manner and makes them more independent from dedicated BI departments. Major requirements for establishing a SSBI solutions are understandable and comprehensible results, accessible software solutions, fast response times when performing complex analyses and quick and easy access to the source data.

D. MUSTANG

MUSTANG is a framework for creating AIS. The system supports non-technical users by providing tools for data analysis in a highly accessible user interface, enabling them to carry out explorative analyses, ad-hoc queries and reporting activities. A usual MUSTANG application consists of three layers. The Data Integration Layer provides a unified physical integration of various heterogeneous data sources, e.g., data warehouses, metadata and geographic information systems. The Component Integration Layer provides services

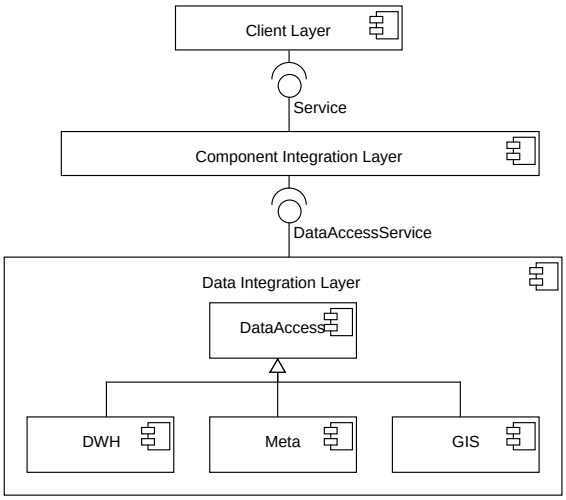


Figure 1. MUSTANG Architecture

to perform different types of analyses and to expose metadata structures like saved analysis configurations and the structure of the underlying data warehouse. The client layer provides an user interface (UI) which orchestrates the services of the Component Integration Layer and visualizes their results. By now only a desktop rich client existed - in the course of this work we developed a new web-based client described in the upcoming sections. Figure 1 shows a simplified overview of the MUSTANG architecture.

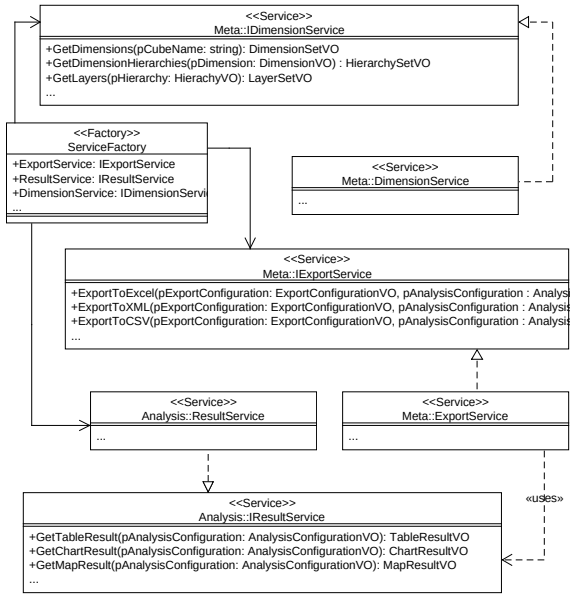


Figure 2. Services classes of the component integration layer

The individual layers are encapsulated from each other. Especially the component integration layer provides a service facade to potential client applications. Figure 2 shows an example for the services classes of the component integration layer. The figure depicts three exemplary MUSTANG services

 ISBN: 978-1-61208-387-2

as well as the service factory which maintains instances of all MUSTANG services. As visible in the figure, all services in MUSTANG are written against interfaces and the service factory exposes only these interfaces in order to maintain interchangeability of services. The central artifact of a MUSTANG analysis is the *AnalysisConfigurationVO* which is used across all services of the Analysis namespace. The exemplary services *ExportService* and *ResultService* use this object to calculate analysis results and provide it for visualization or to export it into various formats. The *DimensionService* is used to expose the dimension structure of the underlying data warehouse which is one piece of an analysis configuration in MUSTANG [6].

III. Key Business Drivers and Requirements

Migrating an existing software system to a cloud environment or even providing it as a service is a very difficult task. First, it has to comply with the service-oriented architecture (SOA) paradigm. A major advantage of doing this is, that with a SOA multiple services can be orchestrated for higher-value services [12][13]. Cloud based applications can be considered a collection of services. Software systems which have been implemented in a service oriented way, should be able to adopt cloud computing more easily [14].

AIS are usually very complex systems, dealing with various different data sources and varying user groups. In most cases, only specialist users conduct analyses with AIS. With this in mind, in the course of the project, the following business drivers and requirements could be identified. Several one-day workshops with potential users were held. The following list gives an overview of the most relevant business drivers and requirements.

Multidimensional analyses: Multidimensional analyses capabilities are required to allow the exploratory analysis of complex data.

Performance: Performance comparable to the CARESS Desktop Client [6] is desired.

Modularity and expandability: It should be possible to subsequently add complementary functionality to the prototype.

Web standards: The prototype should use web standards and should conform to the RESTful architecture paradigm [15].

Metadata: Meaningful metadata should be provided to increase efficiency and accessibility when performing analyses.

User empowerment: It should be possible for inexperienced users to perform complex analyses.

We will address the implementation of each individual business driver in Section V.

IV. Selected Related Work

In the field there exist several approaches that address the aforementioned requirements more or less. In this section we present a selection of related workings and measure their degree of fulfilment of these requirements. This selection consists of CARESS, an instance of MUSTANG with a strong focus on the requirements of epidemiological cancer registries in Germany [6], Tableau, a software tool for analyzing different data sources of all kinds, focusing on self-service BI [16], KNOBI (Knowledge-based Business Intelligence), which is an instance of MUSTANG enriched with an ontology-based semantic layer knowledge [11] and Super Data Hub a web-based analysis solution for big data [17]. While the requirement *multidimensionality* has been met by all related approaches, the other requirements are only fulfilled partially which is especially true for *Metadata* and *Web standards*. See table I for an overview of the related workings and their requirements coverage.

	CARESS	Tableau	KNOBI	Super Data Hub
Multidimensional analyses	x	x	x	x
Performance		x		
Modularity and expandability	x		x	
Web standards				x
Metadata	x		x	
User empowerment		x	x	x

TABLE I. Requirements coverage of related work

V. Approach

In this section, we present our approach for a web-based analysis solution based on the MUSTANG framework. In Section V-A, we describe the server side of our application and especially how we utilize the MUSTANG framework for our purpose. In section V-B, we explain the implementation of our client with particular emphasis on self-service and the usage of metadata.

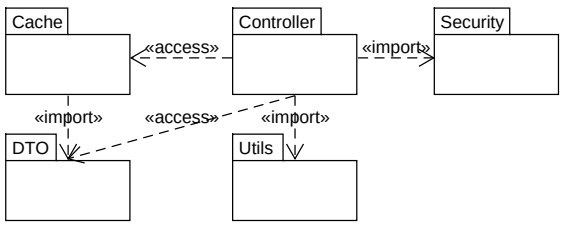


Figure 3. Server Packages

MUSTANG itself started with a monolithic design, but over the course of the past two years it was reengineered towards a more service-oriented architecture. The system itself was divided into loosely coupled components which access each other through a newly added service layer. Moreover, this RESTful service layer was added to encapsulate view logic even further. This allows using different existing cloud services for hosting different parts of the system. Figure 1 shows a simplified architecture of a possible system based on MUSTANG. Each data source, the core application and the view or client application may be deployed using different cloud services.

A. Prototypical Application - Server Side

On foundation of the reengineered MUSTANG framework we developed a prototypical application. The MUSTANG framework already offers rich functions for multidimensional data analysis that we utilized in order to contribute to the business driver *Multidimensional analyses*. We added a completely

 ISBN: 978-1-61208-387-2

new developed HTML5/JavaScript frontend to the desktop application. Therefore, we relied on RESTful web services for data exchange between client and server. We decided to use these technologies in order to comply with the business driver *Web standards*. Moreover, we added the capability to work securely in a multi-client environment.

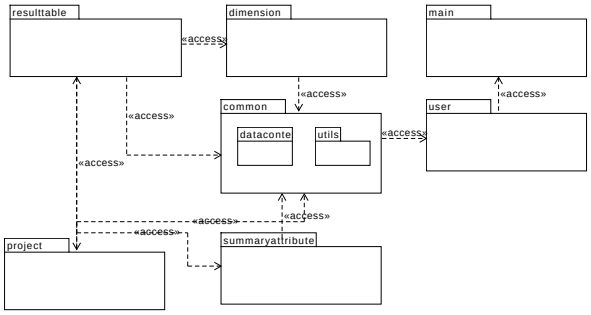


Figure 4. Client Structure

Figure 3 shows the package structure of the server-side application. It is developed on top of the component integration layer of the MUSTANG framework and provides an interface for accessing multidimensional data analysis via web services. For implementing the application, we used the Microsoft ASP.NET Web API 2 which offers an easy way to create RESTful web services. Our application is structured into different packages. The main point is the *controller package*, which contains the controller-classes. A controller class consists of methods which handle and answer the incoming client's HTTP requests (GET, POST, DELETE, PUT). The returning values are mostly in the form of lightweight data transfer objects (DTO) which are organized in the *DTO package*. A DTO in our application corresponds to a *Value Object* provided by the MUSTANG framework. We tailored our DTOs to have a smaller size compared to the corresponding Value Object by removing all information not required by our application. This way we gained quicker response times from communication between web browser and ASP.NET server contributing to the business driver *Performance*. Before transferring the DTOs to requesting clients, they are serialized to the JavaScript Object Notation (JSON), which is the preferred data exchange format when implementing REST APIs.

Another contribution to the business driver *Performance* is the reduction of response times for sending requested data by our implemented caching functions for OLAP-Server-access, which are concentrated in the *Cache package*. Especially the time for retrieving measures, classification nodes and comprehensive metadata-information could be reduced from minutes to seconds by instantiating them already on server start through our caching mechanism, which makes the objects present in-memory. The *Security package* holds all logic necessary for authentication and user and group management. At last, the *Utils-package* contains various utility-classes for handling and transforming multidimensional data.

B. Prototypical Application - Client Side

Our client application consumes the web services of the described server side application. It was developed using HTML5/JavaScript, especially AngularJS [18]. In order to contribute to the business driver *Modularity and expandability*, the application is organized in a modular way with reusable components representing different aspects of the view. Figure 4 gives an overview of the client application's structure.

The *main module* represents the applications starting point. It accesses the *user module* and therefore log-in, log-out and client side security logic. The central module is the *common module* which is accessed by all other modules because its *datacontext* holds the overall data which is re-used throughout the application, e.g., selected multidimensional data, analysis metadata and user data.

The *measures module* contains everything needed to select measures. Assisted by comprehensive metadata, the users can scan a measure-list by entering search terms or using filtering mechanism based on facet classifications to identify proper measures for their analysis intention. The metadata for measures include, for example, information about theme, origin of data, area coverage or available dimensions. In this way, the business drivers *Metadata* and *User empowerment* are addressed.

Figure 5 shows the modal window for selecting measures. (1) shows the facet classification. Facets can be filtered by name and description and individually grouped. The facet definition is part of the Extract-Transform-Load (ETL) process. Users are able to browse, select and deselect different facets. (2) shows list of available measures. For each measure, the most important meta data is shown right away while additional information can be accessed via a tool tip. (3) shows the selected measures as well as the dimensions available with these measures.

In the *dimension module* the functionality for selecting dimensions and appropriate classification nodes is located. Here, the user can navigate by choosing fitting layers or by expanding a hierarchy to find suitable classification nodes. A function to merge different nodes for an ad-hoc aggregation is included as well. Again, it is possible to display additional metadata for individual elements.

The main point for analysis purposes is organized in the *resulttable module*. After selecting measures, dimensions and classification nodes the resulting table is generated here. The user can manipulate the table via drag and drop in an interactive way. Dimensions and measures can be moved and arranged in columns and rows. For a better table view, it is possible to change to a full screen mode, in which all nonessential information is faded out. Furthermore, functionality to export table data into Excel or comma separated value (CSV) format is included. Figure 6 shows a generated table. (1) shows user interface elements for manipulating the table (e.g. moving dimensions from one axis to another). (2) shows the table itself and the result data. Finally (3) shows the selected measures and the user interface elements to determine the presentation of the measures.

The functionality to save analysis data is implemented in the *project module*. Analyses can be organized in collections called projects and enriched with metadata like descriptions, editors and visibility declarations (private vs. public).

 ISBN: 978-1-61208-387-2

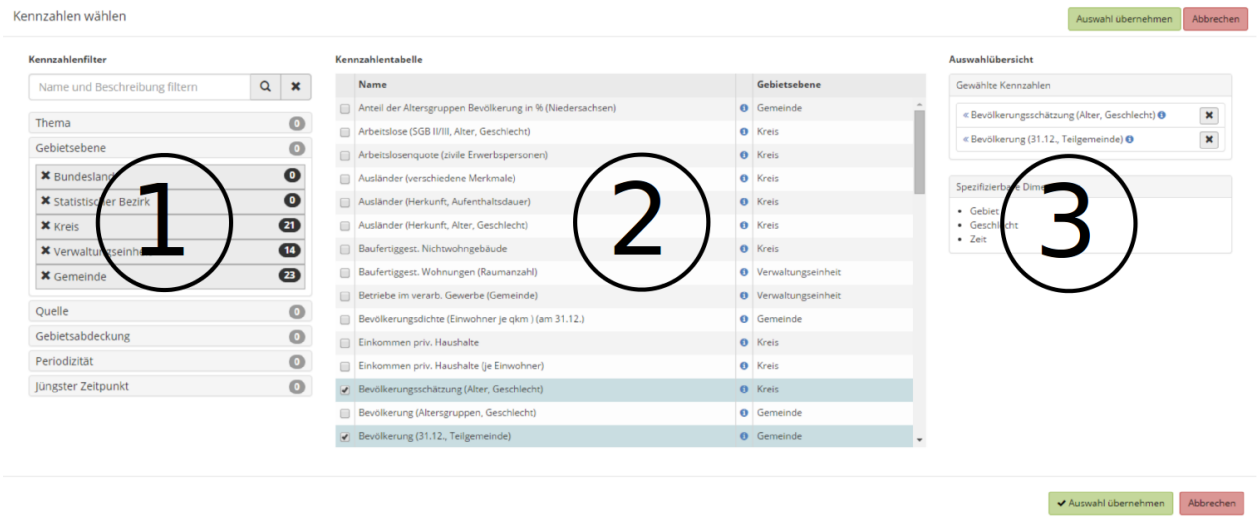


Figure 5. Measure Selection

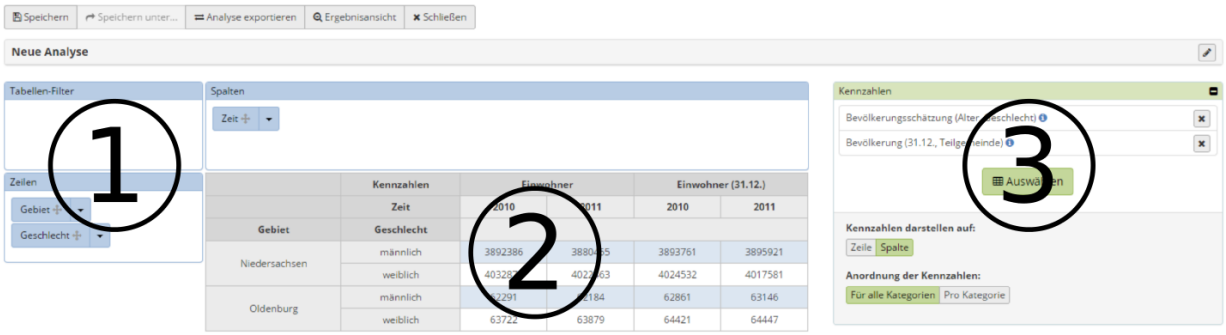


Figure 6. Result Table

VI. Evaluation and Remaining Challenges

The application has been shown to potential users on various occasions, but an extensive evaluation is currently underway and will be completed by the end of 2014. By now, the application was evaluated on four occasions with individual users or user groups. A scenario was created for the evaluation which included a typical task with a corresponding analysis questioning. The participants were encouraged to solve the task and answer the question with the system at hand. Each evaluation lasted about two hours with concluding interviews and a structured feedback process. The evaluation was recorded and will be processed and evaluated step by step. Right now only a first impression from the first evaluation appointments can be given.

The first group consisted of specialist users who are experienced in working with web-based analysis solutions and have an extensive knowledge of the analysis domain. The second group consisted solely of business users with an extensive knowledge of the domain but minor experience in working with analysis solutions. Whilst the first group does not represent the desired core audience for the application, their feedback is also valuable for improving the analysis capacity.

In preparation of the evaluation, an internal evaluation was performed. A colleague who was not associated with the project performed the evaluation. The purpose of this proceeding was to evaluate the scenario, the task to solve and to find bugs or stumbling blocks in the application itself.

Feedback from the first group has been mostly positive. As the majority of participants knew about the domain and the structure of the multidimensional data, they were able to produce valid results very quickly and to solve the given tasks. All agreed that the provided metadata allows performing valid analyses in a convenient way and producing valuable results faster. Some struggled with the concepts of facets in the dialog for selecting measures. This concept — to our knowledge — is not implemented in other applications so far. Metadata can be used in this case for filtering a measure-list. Some participants thought they could already select and specify dimensions while selecting measures. Not all provided metadata was directly accessible and was not used in the first place.

Feedback from the second group has been mostly positive as well. Before evaluation started, a short introduction was given to each participant. Some participants struggled with the overall operation of the system but were able to solve the given task with minimal help. Some participants observed inconsistencies in the operating concept regarding the specification of

 ISBN: 978-1-61208-387-2

measures.

An initial conclusion which we can derive from the evaluation so far is that our overall concept works well and the provided metadata is helpful for possible users with or without experience in working with analysis solutions. But we also found some misconceptions and inconsistencies in our operation concept, especially regarding the provisioning of even more metadata and the concept of facet classification.

VII. Conclusion and Future Work

We presented an approach for a service-based analysis solution with self-service in mind. Our approach proves to be an improvement over existing solutions. There are still a number of issues to be addressed before the solution can be used on a daily basis. We showed that our solution is accessible by business users with only minimal instructions while we preserve the analysis capabilities of an analytical information system.

Up to now we deployed the application only in a local environment. In addition, we only performed limited load tests. Therefore, we need to make sure our application scales with an increasing amount of users simultaneously performing analyses. Whilst MUSTANG is a proven and robust framework it was not tested yet in a multi-user environment.

In Section I we mentioned two main shortcomings of common analysis solutions. We addressed the issue, that typical business users are not capable of performing analyses by themselves by creating a highly accessible solution. Our solution heavily utilizes supporting metadata, which addresses the problem, that the underlying data model is too complex to comprehend. In further iterations, we plan to extend the usage of metadata and knowledge-based approaches. Our system is therefore designed and implemented in an extensible way.

This should enable business users to perform analyses without extensive training or the aid of specialist users. The evaluation showed, that in many cases metadata is sufficient, but in some cases business users require more guidance. This applies especially if they only perform analyses at fixed times (e.g., once per month or year) but not regularly. With a little more work we are sure our approach can be understood as a SSBI solution.

We mentioned some issues in Section VI, mostly regarding our operation concept. Moreover, we did not yet take security issues into account. Whilst we developed only a research prototype, a possible application which will be accessible by end users must take security issues into account. In addition, we will rework parts of our operation concept, especially regarding the facet classification as part of the measure selection.

Another planned improvement concerns the contribution of user-specific data. At the moment, it is only possible to perform analyses with the data available in the underlying DWH. A future version could incorporate an interface with an associated process which allows user groups to add their own data to a user-specific location. Particularly, this is important because certain data is not freely available and must not be available to users outside a respective group.

Acknowledgment

This work is supported by the Federal Ministry of Education and Research (BMBF) on the basis of a decision by the German Bundestag under Grant No. 01IS12042B.

References

[1] P. Chamoni and P. Gluchowski, “Analytische Informationssysteme - Einordnung und Überblick,” in Analytische Informationssysteme : Business Intelligence-Technologien und -Anwendungen, 2010, pp. 3–16.

[2] M. Mircea, B. Ghilic-Micu, and M. Stoica, “Combining business intelligence with cloud computing to delivery agility in actual economy,” Journal of Economic Computation and Economic Cybernetics Studies, vol. 45, no. 1, 2011, pp. 39–54.

[3] M. Mertens, T. Krahn, and H.-J. Appelrath, “Utilizing Structured Information from Multiple External Sources in the Context of the Multidimensional Data Model,” in BIS 2013, ser. Lecture Notes in Business Information Processing,, Abramowicz, Witold. Heidelberg: Springer, 2013, pp. 88–99.

[4] O. I. for Information Technology, “Multidimensional statistical data analysis engine,” http://www.offis.de/en/r_d_divisions/health/project/projekte/mustang.html, last visited 2015-02-09.

[5] ——, “Wais - aufbau eines wissensbasierten analytischen informationssystems für die kollaborative datenanalyse,” http://www.offis.de/en/r_d_divisions/health/project/projekte/wais.html, last visited 2015-02-09.

[6] D. Korfkamp, S. Gudenkauf, M. Rohde, E. Sirri, J. Kieschke, and H.-J. Appelrath, “Opening up Data Analysis for Medical Health Services: Cancer Survival Analysis with CARESS,” in Data Warehousing and Knowledge Discovery - 16th International Conference, Munich, Germany, 2014.

[7] P. Hoberg, J. Wollersheim, and H. Krcmar, “The Business Perspective on Cloud Computing-A Literature Review of Research on Cloud Computing,” AMCIS 2012 Proceedings, no. 5, 2012.

[8] G. Vossen, T. Haselmann, and T. Hoeren, Cloud-Computing für Unternehmen: Technische, wirtschaftliche, rechtliche und organisatorische Aspekte. Heidelberg: dpunkt.verlag GmbH, 2012.

[9] S. Gudenkauf, M. Josefiok, A. Göring, and O. Norkus, “A Reference Architecture for Cloud Service Offers,” in EDOC 2013, 2013.

[10] P. Mell and T. Grance, “The NIST Definition of Cloud Computing - Recommendations of the National Institute of Standards and Technology,” National Institute of Standards and Technology, Gaithersburg, Tech. Rep., 2011.

[11] M. Mertens, KNOBI - Knowledge-based Business Intelligence for Business User Information-Self-Service -, 1st ed. Oldenburg: OlWIR Verlag für Wirtschaft, Informatik und Recht, 2013.

[12] G. Feuerlicht, L. Burkon, and M. Sebesta, “Cloud Computing Adoption: What are the Issues,” Systems Integration, 2011, pp. 187–192.

[13] M. A. Babar and M. A. Chauhan, “A tale of migration to cloud computing for sharing experiences and observations,” in Proceedings of the 2nd International Workshop on Software Engineering for Cloud Computing, ser. SECLOUD ’11. New York, NY, USA: ACM, 2011, pp. 50–56. [Online]. Available: http://doi.acm.org/10.1145/1985500.1985509

[14] J. Lawler, “The Potential Reality of Service-Oriented Architecture (SOA) in a Cloud Computing Strategy,” Journal of Information Systems Applied Research, vol. 4, no. 1, 2011, p. 57.

[15] S. Kumaran, R. Liu, P. Dhoolia, T. Heath, P. Nandi, and F. Pinel, “A restful architecture for service-oriented business process execution,” in Proceedings of IEEE International Conference on e-Business Engineering, 2008, pp. 197–204.

[16] T. Software, “Fast analytics and rapid-fire business intelligence from tableau software,” http://www.tableau.com/, last visited 2015-02-13.

[17] SuperDataHub, “Superdatahub - cloud bi - superdatahub,” http://superdatahub.com/, last visited 2015-02-13.

[18] Google, “Angularjs — superheroic javascript mvw framework,” https://angularjs.org, last visited 2014-10-28.

 ISBN: 978-1-61208-387-2

A Conceptual Model to Evaluate Decisions for Service Profitability

Eng Lieh Ouh
Institute of Systems Science
National University of Singapore
25 Heng Mui Keng Terrace, Singapore
e-mail: englieh@nus.edu.sg

Stan Jarzabek
Department of Computer Science, School of Computing
National University of Singapore
Computing 1, 13 Computing Link, Singapore
Faculty of Computer Science,
Bialystok University of Technology
e-mail: stanjarzabek@gmail.com

***Abstract*— Service profitability depends on the cost of engineering a service for a given base of tenants, on service provisioning cost, and on the revenue gained from selling the service to that tenant base. The tenant base depends on the range of service variability, i.e., on Service Provider's ability to vary service requirements to meet tenant expectations. These various factors that have to do with service profitability form a complex web of information that makes it difficult to analyze and see the exact impact of decisions regarding the choice of service architecture or the use of service adaptation techniques. To make this analysis easier for Service Providers, we built a conceptual model that helps Service Providers identity factors affecting service profitability and the interplay among them. Based on that model, Service Providers can answer questions regarding how choices of the service architecture or tenant base affect service profitability.**

Keywords-service provider; service profittability; service architecture; service variability; tenant base; service engineering; service provisioning.

I. BACKGROUND AND MOTIVATION

Service Providers maximize service profits by looking into ways to best reduce their engineering and provisioning costs while selling the service to possibly a large number of satisfied tenants. The choice of service architecture plays a critical role in balancing the way these three forces affect service profitability. However, the most scalable and cheapest for service provisioning shared service architectures tend to restrict service adaptability. It is our goal in this paper to analyze the interplay among conflicting forces that affect service profitability, and identify the detailed factors behind these forces. The conceptual model of service profitability presented in this paper is to help Service Providers better see how decisions regarding the choices of service architecture, dynamic (at service runtime) versus static (at the service construction-time) service adaptation techniques, or the size of the tenant base affect service profitability. This conceptual model can be further extended with provisions for quantitative analysis of service profitability, which is the subject of our on-going work.

In our previous work [1], we described the decisions that Service Providers typically make during service engineering regarding the choice of service architecture, service packaging, service hosting and the use of static or dynamic binding techniques to adapt services to needs of various tenants. Service components can be encapsulated at a service or tenant specific level packaging. For service hosting, a service can be hosted on a dedicated or shared process instance. Service architectures differ in how the service code is managed during service engineering, service execution and service hosting. As compared to [1], this paper further elaborates on and formalizes earlier findings to build a conceptual model.

Section II introduces Service Variability and Service Architectures. We give our proposed Service Profitability Model in Section III, followed by an Analysis of Service Profitability in Section IV. Related work is presented in Section V. Our conclusion is in Section VI.

II. SERVICE VARIABILITY AND SERVICE ARCHITECTURES

The range of service variability is the extent to which a Service can be adapted to varying service requirements of different tenants. The larger the range the service can accommodate, the higher the number of tenants and the higher the revenue for the Service Provider. Service architecture is composed of a set of components and the relationships among them to implement a service. A service supports a set of common features [15], shared by all the tenants**,** and a set of variant features (F_{RSV}), that are in a range of service variability, i.e., features that are of interest to some but not all tenants. Each feature $f \in F_{RSV}$ corresponds to a set of variation points in service architecture components. The purpose of variation points is to enable customization of components whenever f is required by a tenant. A selected variant to be bind to a variation point is composed of service components of one or more modules as shown in Figure 1.

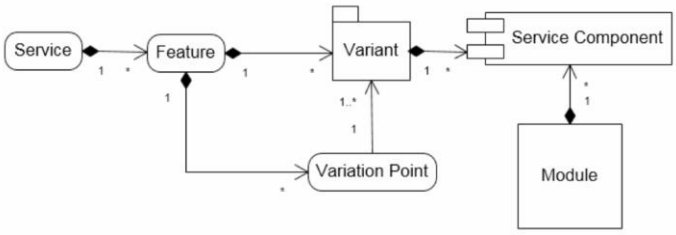


Figure 1. Service, Features and Variants

Fully-Shared (SA_{FS}) is a service architecture based on a shared process instance with service components being shared by tenants during service execution. Dynamic binding techniques are used to bind the variants to the variation points of the feature for service running in SA_{FS}. The range of service variability that is supported by SA_{FS} is as follows:

 ISBN: 978-1-61208-387-2

- At least one new variant or variation point needs to be designed and deployed onto existing process instance to support the varying requirements.

- Existing features can be configured on existing process instance to support the varying requirements

Partially-Shared (SA_{PS}) is a service architecture based on a shared process instance but, as opposed to SA_{FS}, the service components in SA_{PS} can be tenant level or service level packaged. In other words, the service components packaged in the module can be servicing a set of tenants or a particular tenant. Both static and dynamic techniques can be used to bind the variants to the variation points of the features for Service running in SA_{PS}. The range of service variability that is supported by SA_{PS} includes the variability supported by SA_{FS} and the following:

- For at least one new variant, variation points need to be designed and deployed onto a dedicated process instance to support the varying requirements.

- Existing features can be configured, but some features need to be deployed onto a dedicated process instance to support the varying requirements.

Non-Shared (SA_{NS}) is based on each tenant having its own, dedicated process instance and the service components being tenant level packaged. Both static and dynamic techniques can be used to bind the variants to the variation points of the features for Service running in SA_{NS}. The range of service variability that is supported by SA_{NS} includes the variability supported by SA_{PS} and the following:

- All required features need to be deployed onto a dedicated process instance to support the varying requirements.

The service architecture can be hybrid, comprising of a combination of existing service architectures. The SA_{FS+PS} is the hybrid service architecture comprising of SA_{FS} and SA_{PS}. $SA_{FS+PS+NS}$ is the hybrid service architecture comprising of $SA_{FS,}$ SA_{PS} and $SA_{NS.}$ The deployment diagrams of the three basic service architectures $SA_{FS,}$ SA_{PS} and SA_{NS} are shown in Figure 2. The white portions indicate components of the architecture that are not shared among tenants and the dark portions indicate components that are shared among tenants.

Service architectures that share runtime resources to minimize provisioning cost potentially limit the extent to which services can be adapted to varying requirements of tenants. For example, a tenant who requires service to be processed and data isolated due to security regulations cannot be onboard together with other tenants (that do not have such requirements) with a service architecture that does not have clear separation of runtime resources between tenants. Having clear separation of runtime resources among tenants incurs higher provisioning cost for the Service Provider, which would be likely passed on to the tenant as higher service price. On the other hand, there are also tenants who are price-sensitive with minimum variations of requirements. In this case, Service Providers can best minimize cost by engineering the service on a service architecture that shares resources. To the Service Provider, what are the factors and how they interplay can greatly impact their service profitability. We formalized these factors into a Service Profitability Model to help Service Providers analyze a complicated web of interrelated factors affecting service profitability.

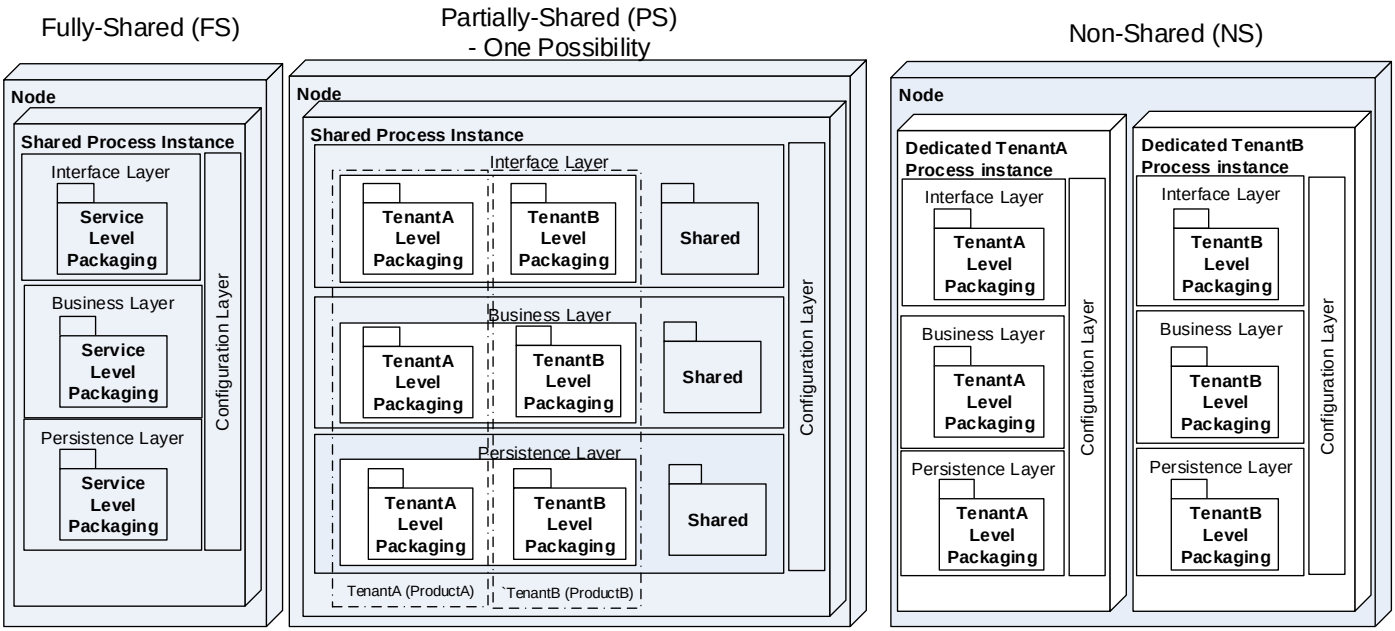


Figure 2. Service Architectures

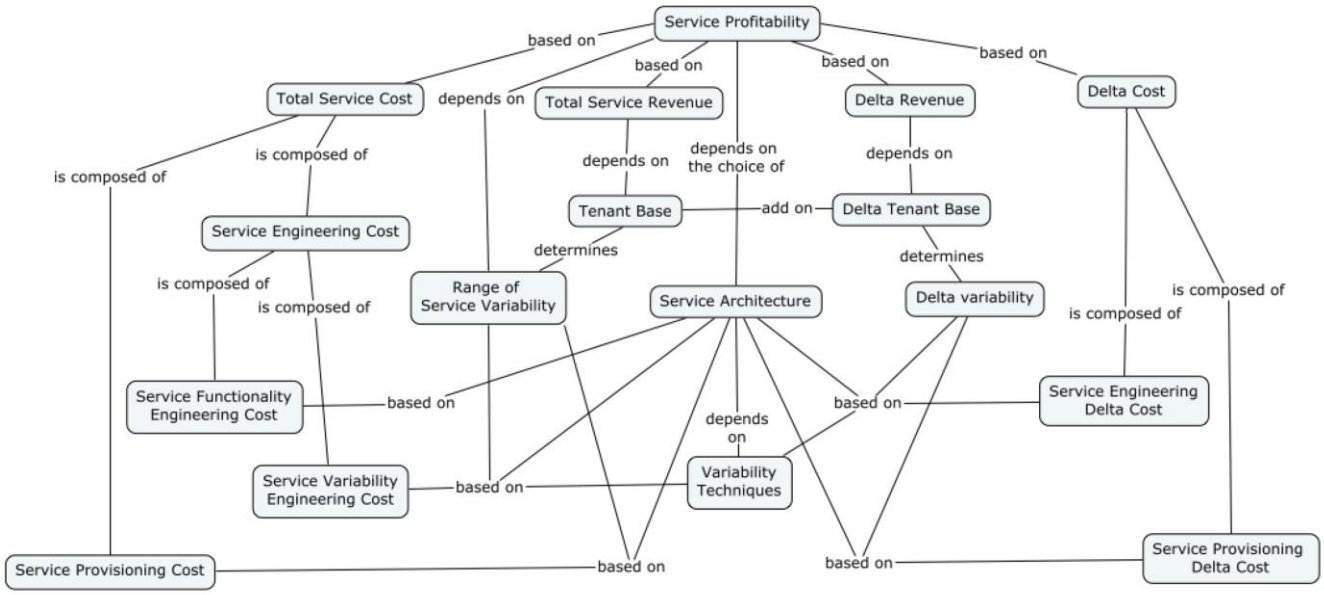


Figure 3. A conceptual model of Service Profitability

III. CONCEPTUAL MODEL OF SERVICE PROFITABILITY

Service Providers seek to maximize their profitability in the long term by minimizing their costs while maximizing their revenue. A Service Provider can re-engineer an existing application into a service or develop a service from scratch. This is the cost incurred in engineering the functionality of the service. The engineering cost varies depending on the choice of service architecture, but is independent of the number of tenants. Besides engineering service functionality, a Service Provider needs engineer variability into a service to accommodate variations in requirements among tenants. For business strategic reasons, a Service Provider may have some target tenants in mind and engineer the variability to meet the varying requirements of the target tenants. The extent of the service variability can vary with each variability technique and the selected service architecture which in turn impact the cost to engineer for variability. Service engineering costs can be estimated with software cost estimation tools (e.g., COCOMOII [2]) typically in terms of function points or lines of code. The other factor to the cost on top of service engineering cost is the provisioning cost. This cost is incurred to host the service in a hardware and network environment to serve the tenant's requests. The provisioning cost varies with the selected service architectures and the extent of service variability. The provisioning can be hosted internally or externally, virtualized or non-virtualized. To minimize provisioning cost, a virtualized environment is usually adopted. One key factor to whether the hosting is internal or external depends on each organization's security policy. Provisioning cost of virtualized environment can be estimated on the respective cloud hosting sites (e.g., Amazon Web Services [3]). During the lifecycle of the service, new tenants might be interested to subscribe to the same service. For business reasons, a Service Provider may also wish to onboard new tenants to maximize their revenue. However, the current extent of service variability might not exactly fit the requirements of the new tenants. The additional engineering and provisioning costs vary with the selected service architectures, adopted variability technique and the extent of variability of the tenant's requirements. The costs to engineer and provision for delta variability can be estimated similarly as described earlier. Service revenue will vary with the number of initial set and new set of tenants. Service profitability for providing the service in the long term can be measured by the net present value of the revenue gained minus the costs incurred over a pre-determined investment horizon, taking into account the value of money over time. For the rest of this section, we define the terms and use them in the conceptual model of service profitability. A conceptual model of Service Profitability is found in Figure 3.

A. Explanation of the terms in conceptual model

The Range of Service Variability (RSV) is the extent to which a Service can be adapted to varying service requirements of different tenants. The larger the range the service can accommodate, the higher the number of tenants and the higher the revenue for the Service Provider. The Tenant Base (TB) is composed of the users of a given Service, or onboarded tenants. To onboard a tenant, the Service must be able to meet the requirements of that tenant. As RSV reflects the Service Provider's ability to customize the Service, RSV determines the TB that can be supported. Service Providers dream to engineer a service where RSV fulfills the varying requirements of the TB, maximizing service profits. Service Profits are determined by the Total Service Cost (TSC) incurred and Total Service Revenue (TSR) gained when providing the service. Costs are incurred to engineer the functionality of the Service and to engineer the Service to support a given RSV on a given Service Architecture (SA). For this, we collectively use the term Service Engineering Costs (SEC) to denote the both the Service Functionality Engineering Cost (SFEC) and Service Variability Engineering Cost (SVEC). SVEC is based on the selected Variability Techniques (VT) to support a given RSV

 ISBN: 978-1-61208-387-2

on a given SA. There are also costs involved to provide the hardware and infrastructure resources to support a given TB on a given SA. For this, we use the term Service Provisioning Costs (SPC). TSR is the total revenue from selling the Service on a given service SA.

Delta Variability (DV) is the change to existing Service requirements required to onboard new tenant(s). We denote these newly onboarded tenants as Delta Tenant Base (DTB). Delta Cost (DC) is the cost to implement DV for a given DTB on a given SA. Similarly to TSC, DC is composed of Service Engineering Delta Cost (SEDC) and Service Provisioning Delta Cost (SPDC).

IV. An Analysis of Service Profitability

In this section, we show use cases for the conceptual model presented as ten questions illustrating specific profitability-related dilemma of a Service Provider during service planning.

A. Service Profitability Model and Service Archiectures

For $SA_{FS,}$ the service engineering cost to support the given range of variability is composed of implementing the dynamic binding techniques to bind the variants to the variation points identified for each feature of the Service. The service provisioning costs to support the given range of service variability is composed of provisioning the hardware and infrastructure resources to support the tenant base. Among the basic architectures, SA_{FS} have the lowest provisioning costs due to sharing of resources. Delta variability that is the changes to existing Service requirements by new tenants have to be supported by the service architecture to onboard the tenant. If the delta variability is outside the given range of service variability of SA_{FS} then there are two possible scenarios. One scenario is that the Service Provider incurs the service variability delta costs to engineer the delta variability into the existing $SA_{FS.}$ (e.g., implementing new variants or variation points that can be shared among tenants). The second scenario is that the tenant cannot be onboard as the delta variability cannot be supported on existing SA_{FS} (e.g., tenant require total isolation of software, process and data).

For $SA_{PS,}$ the service variability engineering costs to support the given range of service variability is higher than SA_{FS} as the Service Provider needs to implement both the dynamic and the static binding techniques. The service provisioning costs are also higher than SA_{FS} due to the tenant level packaging leading to the need to provision more resources to support more software components. If the delta variability of new tenants is outside the given range of service variability of $SA_{PS,}$ a Service Provider can onboard the tenants by incurring service variability delta costs to engineer the delta variability into the existing $SA_{PS.}$ (e.g., implementing new variants or variation points that can be shared among tenants or supporting isolation of software or data). As SA_{PS} is running on shared process instance, the Service Provider is unable to onboard the tenant if the tenant has requirements that require a dedicated process instance (e.g., isolation of processes).

For $SA_{NS,}$ the service variability engineering costs to support the given range of service variability is lower than SA_{PS} as it needs to implement only based on static binding techniques. The service provisioning costs are the highest among SA_{NS} and SA_{PS} as dedicated resources are provided for each tenant. If the delta variability of new tenants is outside the given range of service variability of SA_{NS}, SA_{NS} can support delta variability of new tenants (even for isolation of process) due to dedicated process instances.

For $SA_{FS+PS,}$ the service variability engineering costs to support the given range of service variability involves the implementation of both dynamic and static binding techniques and the need to support two basic service architectures. It is higher than any of the basic architectures $SA_{FS,}$ SA_{PS} or $SA_{NS.}$ Depending on whether the service is provisioned on SA_{FS} and/or $SA_{PS,}$ the provisioning costs for SA_{FS+PS} can be calculated from the respective provisioning costs of SA_{FS} and/or SA_{PS}. The support of delta variability is similar to the scenarios in basic architectures of SA_{FS} or $SA_{PS.}$

For SA_{FS+PS} and $SA_{FS+PS+NS,}$ the service variability engineering costs to support the given range of service variability involves the implementation of both dynamic and static binding techniques and the need to support all three basic service architectures. It is the higher than the basic service architectures. Depending on whether the service is provisioned on SA_{FS}, SA_{PS} and/or $SA_{NS,}$ the provisioning costs for SA_{FS+PS} can be calculated from the respective provisioning costs of SA_{FS}, SA_{PS} and/or SA_{NS}.

B. Key Questions to Service Profitability

Questions 1 to 3 and 4 to 6 are related to the revenue and cost of the Service Profitability Model. Questions 7 to 9 are related to service architectures, while the last question is related to the overall service profitability.

1) How many tenants will have to be onboarded so that the profit from service outweighs the cost of building it?

The answer to this question will help the Service Provider to estimate the breakeven point for service cost and revenue. For a given service architecture, the Service Provider can simulate the number of tenants against the expected cost incurred to build the service for the Service Provider to have a better insight on the breakeven point for the initial total service costs.

2) How many new delta-tenants will have to be onboarded to outweigh the cost of implementing changes (delta-requirements) required for new tenants?

To Service Providers, this is another question on the breakeven of their investment. It differs from question 7 in terms of the delta cost against the delta revenue gained from the delta tenant base. For a given service architecture, the Service Provider can simulate the number of delta tenants against the expected delta cost incurred to build the service, the Service Provider has a better insight on the breakeven point for the delta costs.

3) Which pricing stratgey should a Service Provider adopt for the service?

The right pricing strategy positively impacts the total service revenue and delta revenue to be gained. The pricing strategy can be pay as per use, subscription-based, tiered based, transaction based or mixture of the above. The pricing strategy can also vary across time. For a given service architecture, the

 ISBN: 978-1-61208-387-2

Service Provider can simulate the expected distribution of tenants for each of the pricing strategies across the investment horizon and evaluate for overall service profitability.

4) Is it better to re-engineer from existing code or develop services from scratch?

The decision to this question impacts the service functionality engineering cost, which is independent of the number of tenants. Based on simulation of the expected distribution of tenants, this question can be addressed to see if the impact of the additional costs to develop from scratch versus the savings by re-engineering from existing code.

5) Should a Service Provider provision the service internally or externally?

The impact of hosting choices on service profitability can be inferred from service provisioning and service provisioning delta costs. The cost of provisioning the service internally typically incurs higher upfront setup and maintenance cost than external provisioning. However, it provides higher degree of security in terms of privacy and isolation of data and processes. Provisioning the service externally can decrease upfront setup and maintenance costs, but the level of control is also reduced. Based on the above, simulations can be run with these costs and expected distribution of tenants to analyze overall service profitability.

6) Which variability technique should a Service Provider apply to address the required range of service variability?

A Service Provider can apply static variability techniques to address higher degree of variability, but can also be more costly as compared to only applying dynamic variability techniques to address runtime variability. The adoption of variability technique impacts the service variability engineering cost and service engineering delta cost. Additionally, due to the variability technique, new tenants with requirements that does not fit the service variability will either be unable to onboard or the service architecture needs to evolve. Assuming the service architecture remains the same, the Service Provider can simulate the expected distribution of tenants for each of the adopted variability techniques and evaluate for overall service profitability.

7) To what degree does the selection of a service architecture impact service profitability?

This is a typical profitability-related question a Service Provider tries to answer. Based on the conceptual model, a Service Provider can estimate the total service costs incurred for each choice of the service architecture. In addition, a Service Provider may simulate a set of expected delta tenants that can be onboarded within the pre-determined specified investment horizon and measure the impact to total service costs, total service revenue and overall service profits.

8) Should a Service Provider onboard new tenant if this requires the change of service architecture?

Onboarding new tenants present new business opportunities, but also incurs costs to implement extra delta variability into the Service. This cost grows further if onboarding new tenants requires the change of service architecture, e.g., from shared to dedicated. In this case, the Service Provider needs to simulate different distributions of tenants and evaluate the overall profitability. These distributions differ in their degree of varying requirements and evaluate the service profitability impact on each of the service architectures. From the results, a Service Provider can have a better insight whether to evolve their architecture or not.

9) Should a Service Provider adopt a specific service architecture or a hybrid of service architectures?

Besides deciding on the type variability techniques, adopting a hybrid of service architectures requires additional cost incurred to manage the variability across service architectures. The Service Provider can simulate the expected distribution of tenants for each service architecture including different hybrids and evaluate for overall service profitability.

10) If a Service Provider has an objective to achieve a certain level of service profitability within a certain investment horizon, what costs and revenue the Service Provider needs to incur and gain?

This question can be considered an optimization problem in terms of maximizing the total service revenue and delta revenue while minimizing the total service costs and delta cost for a given level of service profitability. From the conceptual model, factors that affect the total service cost and total service revenue can be defined as equations to be evaluated by an optimization solver. More than one optimal set of values is possible in this case. The Service Provider can evaluate and design towards one of these values in this optimal set.

Each of these questions can be addressed at the lower level of abstraction. For example, in the first question, the assumption of a given service architecture can be further decomposed to evaluate against a set of service architectures for the Service Provider to be able to evaluate more scenarios.

V. RELATED WORK

There is much literature on how to minimize service engineering and provisioning costs to improve service profitability. Existing methods focus on variability techniques to enable late binding of the service variants, customization of business process execution language (BPEL) process with variability descriptors [8] and variability modeling techniques to manage the variability in service applications [9][10]. Kwok et al. [11] propose to lower the provisioning cost through resource calculations with constraints to decide the best server to onboard a new tenant. A resource consumption estimation model to optimally place onboarding tenants is also studied in [12].

A single-instance multi-tenant service application enables a Service Provider to achieve economies of scale through runtime sharing. However, runtime sharing can make tenant-specific variations difficult to achieve in such an application as it needs to realize the variability across different tenants in the single-instance application [13]. For the software as service paradigm to truly meet its potential, Sengupta et.al. [14] propose that vendors will need to move away from building rigid “one-size-fits-all” systems, or those that offer a fixed set of available customization options from which tenants must select. Service paradigm essentially is an economic model for software consumption; hence, many of these activities would

 ISBN: 978-1-61208-387-2

have to be grounded on the basis of financial reasoning that can benefit the vendor as well as the tenants. In contrast with other works that addresses costs or benefits independently, this paper takes a holistic view of service adoption in terms of service profitability. To the best of our knowledge, there are no such economics-driven service adoption evaluation methods; therefore service adoption is frequently made without holistically evaluating whether it is economically worthwhile to invest for the long term. In this study, we seek to propose an economic model of service profitability based on high-level conceptual model of service profitability (original contribution) and existing value-added software engineering metrics and economics-driven models used in other areas. We believe that the reasoning based on service profitability is able to address the service paradigm with a more balanced view than before.

VI. CONCLUSION

Our proposed Service Profitability Model formalizes the interplay of multiple factors that influence service profitability. The model addresses decisions related to the tenant base, required range of service variability, service architecture and the use of variability techniques. Our model shows how these decisions affect service cost and revenue. We illustrated the usage of our profitability model with an analysis of service architectures and service profitability scenarios. We believe the model will help Service Providers maximize service profitability. For future work, we intend to extend our Service Profitability Model with quantitative methods and tools that can help Service Providers examine factors that affect service profitability.

REFERENCES

[1] E. L. Ouh and S. Jarzabek, “Understanding Service Variability for Profitable Software as a Service: Service Providers’ Perspective,” in 26th International Conference on Advanced Information Systems Engineering (CAiSE), 2014, pp. 9-16.

[2] “COCOMO II” http://csse.usc.edu/csse/research/COCOMOII [retrieved: Jan, 2015].

[3] “Amazon Web Services Pricing” http://aws.amazon.com/pricing/ [retrieved: Jan, 2015].

[4] A. Mili, S. F. Chmiel, R. Gottumukkala, and L. Zhang, "An integrated cost model for software reuse," in Proceedings of the 22nd international conference on Software engineering (ICSE), 2000, pp. 157-166.

[5] "The Apache Open for Business Project (OfBiz)" http://ofbiz.apache.org/ [retrieved: Jan, 2015].

[6] Stanislaw Jarzabek and Dan Daniel, "Adaptive Reuse Technique" http://art.comp.nus.edu.sg/ [retrieved: Jan, 2015].

[7] "FeatureIDE" http://wwwiti.cs.uni-magdeburg.de/iti_db/research

[8] R. Mietzner and F. Leymann, "Generation of BPEL Customization Processes for SaaS Applications from Variability Descriptors," in IEEE International Conference of Services Computing, 2008, pp. 359-366,.

[9] R. Mietzner, A. Metzger, F. Leymann, and K. Pohl, “Variability modeling to support customization and deployment of multi-tenant-aware Software as a Service applications,” in Proceedings of the 2009 ICSE Workshop on Principles of Engineering Service Oriented Systems, 2009, pp. 18-25.

[10] B. Morin, O. Barais, and J. M. Jzquel, "Weaving Aspect Configurations for Managing System Variability," in Proceedings of VaMoS, 2008, pp. 53-62,.

[11] T. Kwok and A. Mohindra, “Resource calculations with constraints, and placement of tenants and instances for multi-tenant SaaS applications,” in Service-Oriented Computing (ICSOC), 2008, pp. 633-648.

[12] Y. Zhang, Z, Wang, B. Gao, C. Guo, W. Sun, and X. Li, “An effective heuristic for on-line tenant placement problem in SaaS,” in IEEE International Conference on Web services (ICWS), 2010, pp. 425-432.

[13] I. Kumara, J. Han, A. Colman, T. Nguyen, and M. Kapuruge, "Sharing with a Difference: Realizing Service-based SaaS Applications with Runtime Sharing and Variation in Dynamic Software Product Lines," in IEEE International Conference on Services Computing (SCC), 2013, pp. 567-574.

[14] B. Sengupta and A. Roychoudhury, “Engineering multi-tenant software-as-a-service systems,” in 3rd International Workshop on Principles of Engineering Service-Oriented Systems, 2011, pp. 15-21.

[15] J. Lei, B. Sengupta and A. Roychoudhury, "Tenant Onboarding in Evolving Multi-tenant Software-as-a-Service Systems," in IEEE International Conference on Web Services (ICWS), 2012, pp. 415-422.

[16] M. Svahnberg, J. Van Gurp, and J. Bosch, ”A taxonomy of variability realization techniques. Software: Practice and Experience,” 2005, pp. 705-754.

[17] Y. Zhang, Z. Wang, B. Gao, C. Guo, W. Sun, and Xu Li, “An effective heuristic for on-line tenant placement problem in SaaS,” in International Conference on Web services (ICWS), 2010, pp. 425-432.

[18] M. Ma, and J. Huang, ”The pricing model of cloud computing services,” in Proceedings of the 14th Annual International Conference on Electronic Commerce, 2012, pp. 263-269.

[19] A. Mukhija, D. S. Rosenblum, H. Foster, and S. Uchitel, “Runtime support for dynamic and adaptive service composition. in Rigorous software engineering for service-oriented systems,” 2011, pp. 585-603.

[20] H. Jegadeesan and S. Balasubramaniam, ”A method to support variability of enterprise services on the cloud,” in Cloud Computing, 2009, pp. 117-124.

[21] D. Ma, ”The business model of" software-as-a-service". in IEEE International Conference on Services ComputingC(SCC), 2007, pp. 701-702.

[22] W. Frakes and C. Terry, "Software reuse: metrics and models," ACM Computing Surveys (CSUR), vol. 28, no. 2, pp. 415-435, 1996.

[23] V. Choudhary, "Software as a Service: Implications for Investment in Software Development," in 40th Annual Hawaii International Conference on System Sciences (HICSS), 2007, pp. 209a.

[24] B. Boehm, "Software Engineering Economics," in IEEE Transactions on Software Engineering, 1984, pp. 4-21.

[25] C. P. Bezemer, A. Zaidman, B. Platzbeecker, T. Hurkmans, and A. t Hart, “Enabling multi-tenancy: An industrial experience report,” in 2010 IEEE International Conference on Software Maintenance (ICSM), 2010, pp. 1-8.

[26] H. Wang and Z. Zheng, “Software architecture driven configurability of multi-tenant SaaS application,” in Web Information Systems and Mining, 2010, pp. 418-424.

[27] R. Mietzner, F. Leymann, and M.P. Papazoglou, “Defining composite configurable SaaS application packages using SCA, variability descriptors and multi-tenancy patterns,” in Third International Conference on Internet and Web Applications and Services (ICIW), 2008, pp. 156-161.

[28] T. Kwok, T. Nguyen, and L. Lam, “A software as a service with multi-tenancy support for an electronic contract management application,” in IEEE International Conference on Services Computing (SCC), 2008, pp. 179-186.

[29] Y. Zhang, S. Liu, and X Meng, “Towards high level SaaS maturity model: methods and case study,” in IEEE Asia-Pacific Services Computing Conference (APSCC), 2009, pp. 273-278

 ISBN: 978-1-61208-387-2