Zbornik 18. mednarodne multikonference

INFORMACIJSKA DRUŽBA – IS 2015

Zvezek D

Proceedings of the 18th International Multiconference

INFORMATION SOCIETY – IS 2015

Volume D

Sodelovanje, programska oprema in storitve v informacijski družbi
Collaboration, Software and Services in Information Society

Uredil / Edited by

Marjan Heričko

http://is.ijs.si

12. oktober 2015 / October 12th, 2015
Ljubljana, Slovenia

Zbornik 18. mednarodne multikonference

INFORMACIJSKA DRUŽBA – IS 2015

Zvezek D

Proceedings of the 18th International Multiconference

INFORMATION SOCIETY – IS 2015

Volume D

Sodelovanje, programska oprema in storitve v informacijski družbi

Collaboration, Software and Services in Information Society

Uredil / Edited by

Marjan Heričko

12. oktober 2015 / October 12th, 2015
Ljubljana, Slovenia

Urednik:

Marjan Heričko
University of Maribor
Faculty of Electrical Engineering and Computer Science

Založnik: Institut »Jožef Stefan«, Ljubljana
Priprava zbornika: Mitja Lasič, Vesna Lasič, Lana Zemljak
Oblikovanje naslovnice: Vesna Lasič, Mitja Lasič

Dostop do e-publikacije:
http://library.ijs.si/Stacks/Proceedings/InformationSociety

Ljubljana, oktober 2015

CIP - Kataložni zapis o publikaciji
Narodna in univerzitetna knjižnica, Ljubljana

004.77(082)(0.034.2)

MEDNARODNA multikonferenca Informacijska družba (18 ; 2015 ; Ljubljana)
Sodelovanje, programska oprema in storitve v informacijski družbi [Elektronski vir] : zbornik 18. mednarodne multikonference Informacijska družba - IS 2015, 12. oktober 2015, [Ljubljana, Slovenia] : zvezek D = Collaboration, software and services in information society : proceedings of the 18th International Multiconference Information Society - IS 2015, October 12th, 2015, Ljubljana, Slovenia : volume D / uredil, edited by Marjan Heričko. - El. zbornik. - Ljubljana : Institut Jožef Stefan, 2015

ISBN 978-961-264-088-0 (pdf)
1. Gl. stv. nasl. 2. Vzp. stv. nasl. 3. Dodat. nasl. 4. Heričko, Marjan
281711104

PREDGOVOR MULTIKONFERENCI INFORMACIJSKA DRUŽBA 2015

Multikonferenca Informacijska družba (http://is.ijs.si) je z osemnajsto zaporedno prireditvijo osrednji srednjeevropski dogodek na področju informacijske družbe, računalništva in informatike. Letošnja prireditev traja tri tedne in poteka na Fakulteti za računalništvo in informatiko in Institutu »Jožef Stefan«.

Informacijska družba, znanje in umetna inteligenca se razvijajo čedalje hitreje. V vse več državah je dovoljena samostojna vožnja inteligentnih avtomobilov, na trgu je moč dobiti čedalje več pogosto prodajanih avtomobilov z avtonomnimi funkcijami kot »lane asist«. Čedalje več pokazateljev kaže, da prehajamo v naslednje civilizacijsko obdobje, hkrati pa so konflikti sodobne družbe čedalje težje razumljivi.

Letos smo v multikonferenco povezali dvanajst odličnih neodvisnih konferenc. Predstavljenih bo okoli 300 referatov v okviru samostojnih konferenc in delavnic, prireditev bodo spremljale okrogle mize in razprave ter posebni dogodki kot svečana podelitev nagrad. Referati so objavljeni v zbornikih multikonference, izbrani prispevki pa bodo izšli tudi v posebnih številkah dveh znanstvenih revij, od katerih je ena Informatica, ki se ponaša z 38-letno tradicijo odlične znanstvene revije.

Multikonferenco Informacijska družba 2015 sestavljajo naslednje samostojne konference:

- Inteligentni sistemi
- Kognitivna znanost
- Izkopavanje znanja in podatkovna skladišča
- Sodelovanje, programska oprema in storitve v informacijski družbi
- Vzgoja in izobraževanje v informacijski družbi
- Soočanje z demografskimi izzivi
- Kognitonika
- Delavnica »SPS EM-zdravje«
- Delavnica »Pametna mesta in skupnosti kot razvojna priložnost Slovenije«
- Druga študentska konferenca s področja računalništva in informatike za doktorske študente
- Druga študentska konferenca s področja računalništva in informatike za vse študente
- Osma mednarodna konferenca o informatiki v šolah: razmere, evolucija in perspektiva.

Soorganizatorji in podporniki konference so različne raziskovalne institucije in združenja, med njimi tudi ACM Slovenija, SLAIS in Inženirska akademija Slovenije. V imenu organizatorjev konference se zahvaljujemo združenjem in inštitucijam, še posebej pa udeležencem za njihove dragocene prispevke in priložnost, da z nami delijo svoje izkušnje o informacijski družbi. Zahvaljujemo se tudi recenzentom za njihovo pomoč pri recenziranju.

V 2015 bomo tretjič podelili nagrado za življenjske dosežke v čast Donalda Michija in Alana Turinga. Nagrado Michie-Turing za izjemen življenjski prispevek k razvoju in promociji informacijske družbe bo prejel prof. dr. Jurij Tasič. Priznanje za dosežek leta je pripadlo dr. Domnu Mungosu. Že petič podeljujemo nagradi »informacijska limona« in »informacijska jagoda« za najbolj (ne)uspešne poteze v zvezi z informacijsko družbo. Limono je dobilo počasno uvajanje informatizacije v slovensko pravosodje, jagodo pa spletna aplikacija »Supervizor«. Čestitke nagrajencem!

Niko Zimic, predsednik programskega odbora
Matjaž Gams, predsednik organizacijskega odbora

FOREWORD - INFORMATION SOCIETY 2015

In its 18th year, the Information Society Multiconference (http://is.ijs.si) remains one of the leading conferences in Central Europe devoted to information society, computer science and informatics. In 2015 it is extended over three weeks located at Faculty of computer science and informatics and at the Institute "Jožef Stefan".

The pace of progress of information society, knowledge and artificial intelligence is speeding up. Several countries allow autonomous cars in regular use, major car companies sell cars with lane assist and other intelligent functions. It seems that humanity is approaching another civilization stage. At the same time, society conflicts are growing in numbers and length.

The Multiconference is running in parallel sessions with 300 presentations of scientific papers at twelve conferences, round tables, workshops and award ceremonies. The papers are published in the conference proceedings, and in special issues of two journals. One of them is Informatica with its 38 years of tradition in excellent research publications.

The Information Society 2015 Multiconference consists of the following conferences:

- Intelligent Systems
- Cognitive Science
- Data Mining and Data Warehouses
- Collaboration, Software and Services in Information Society
- Education in Information Society
- Facing Demographic Challenges
- Cognitonics
- SPS EM-Health Workshop
- Workshop »Smart Cities and Communities as a Development Opportunity for Slovenia«
- 2nd Computer Science Student Conference, PhD Students
- 2nd Computer Science Student Conference, Students
- 8th International Conference on Informatics in Schools: Situation, Evolution, and Perspective.

The Multiconference is co-organized and supported by several major research institutions and societies, among them ACM Slovenia, i.e. the Slovenian chapter of the ACM, SLAIS and the Slovenian Engineering Academy. In the name of the conference organizers we thank all societies and institutions, all participants for their valuable contribution and their interest in this event, and the reviewers for their thorough reviews.

For 2013 and further, the award for life-long outstanding contributions will be delivered in memory of Donald Michie and Alan Turing. The life-long outstanding contribution to development and promotion of information society in our country is awarded to Dr. Jurij Tasič. In addition, a reward for current achievements was pronounced to Dr. Domnu Mungosu. The information strawberry is pronounced to the web application "Supervizor, while the information lemon goes to lack of informatization in the national judicial system. Congratulations!

Niko Zimic, Programme Committee Chair
Matjaž Gams, Organizing Committee Chair

KONFERENČNI ODBORI
CONFERENCE COMMITTEES

International Programme Committee

Vladimir Bajic, South Africa
Heiner Benking, Germany
Se Woo Cheon, South Korea
Howie Firth, UK
Olga Fomichova, Russia
Vladimir Fomichov, Russia
Vesna Hljuz Dobric, Croatia
Alfred Inselberg, Israel
Jay Liebowitz, USA
Huan Liu, Singapore
Henz Martin, Germany
Marcin Paprzycki, USA
Karl Pribram, USA
Claude Sammut, Australia
Jiri Wiedermann, Czech Republic
Xindong Wu, USA
Yiming Ye, USA
Ning Zhong, USA
Wray Buntine, Australia
Bezalel Gavish, USA
Gal A. Kaminka, Israel
Mike Bain, Australia
Michela Milano, Italy
Derong Liu, Chicago, USA
Toby Walsh, Australia

Organizing Committee

Matjaž Gams, chair
Mitja Luštrek
Lana Zemljak
Vesna Koricki-Špetič
Mitja Lasič
Robert Blatnik
Mario Konecki
Vedrana Vidulin

Programme Committee

Nikolaj Zimic, chair
Franc Solina, co-chair
Viljan Mahnič, co-chair
Cene Bavec, co-chair
Tomaž Kalin, co-chair
Jozsef Györkös, co-chair
Tadej Bajd
Jaroslav Berce
Mojca Bernik
Marko Bohanec
Ivan Bratko
Andrej Brodnik
Dušan Caf
Saša Divjak
Tomaž Erjavec
Bogdan Filipič
Andrej Gams
Matjaž Gams
Marko Grobelnik
Nikola Guid
Marjan Heričko
Borka Jerman Blažič Džonova
Gorazd Kandus
Urban Kordeš
Marjan Krisper
Andrej Kuščer
Jadran Lenarčič
Borut Likar
Janez Malačič
Olga Markič
Dunja Mladenič
Franc Novak
Vladislav Rajkovič Grega
Repovš
Ivan Rozman
Niko Schlamberger
Stanko Strmčnik
Jurij Šilc
Jurij Tasič
Denis Trček
Andrej Ule
Tanja Urbančič
Boštjan Vilfan
Baldomir Zajc
Blaž Zupan
Boris Žemva
Leon Žlajpah

KAZALO / TABLE OF CONTENTS

Sodelovanje, programska oprema in storitve v informacijski družbi/ Collaboration, Software and Services in Information Society ***1***

PREDGOVOR / FOREWORD 3

PROGRAMSKI ODBORI / PROGRAMME COMMITTEES 4

Predicting Software Defect-Proneness from Software Repository Data – A Case of Eclipse Bug Data/ Podgorelec Vili, Karakatič Sašo 5

Graph-Based Analysis and Metrics of Evolving Open-Source Software Systems/ Grbac Babić Sanja, Galinac Grbac Tihana 9

How to Evaluate Technical Quality of Python Projects / Malačič Robert 13

Different Approaches for Measuring XML Schemas/ Pušnik Maja, Rakić Gordana, Budimac Zoran. Heričko Marjan 17

Survey on Software Testing for Big Data Applications/ Karakatič Sašo, Pavlič Luka, Arumugam Gurusamy. Muthuraman Thangaraj, Heričko Marjan 21

Applying Mobile Services Guidelines for User Acceptance to Hybrid Mobile Application Development/ Ovčjak Boris, Jošt Gregor, Polančič Gregor 25

User Interface Development Challenges on Mobile Devices: A Case of Diabetes Risk Estimation Applications/ Fijačko Nino, Povalej Bržan Petra, Štiglic Gregor 29

Agroit Project: On Standards Based Cloud Integration in ICT in Farming/ Rupnik Rok 33

Review of Classification Frameworks for E-Collaboration IT Solutions/ Fluks Zmago, Heričko Marjan 37

Method for Linking Theoretical and Practical Knowledge in Expert Systems and Their Application in Welding Expert System/ Stropnik Ambrož, Vuherer Tomaž, Zorman Milan 41

Participation of Persons with Disabilities in Collaborative Work and Community / Welzer Tatjana, Lhotska Lenka, Grout Ian, Ward Tony, Družovec Marjan 45

Indeks avtorjev / Author index ***49***

Zbornik 18. mednarodne multikonference

INFORMACIJSKA DRUŽBA – IS 2015

Zvezek D

Proceedings of the 18th International Multiconference

INFORMATION SOCIETY – IS 2015

Volume D

Sodelovanje, programska oprema in storitve v informacijski družbi

Collaboration, Software and Services in Information Society

Uredil / Edited by

Marjan Heričko

12. oktober 2015 / October 12th, 2015
Ljubljana, Slovenia

PREDGOVOR / PREFACE

This year, the conference "Collaboration, Software and Services in Information Society" is being organized for the fifteenth time as a part of the "Information Society" multi-conference. As in previous years, the papers from this year's proceedings address actual challenges and best practices related to software and service engineering as well as successful collaboration.

The acceptance and success of advanced ICT-based services depends heavily on their quality and reliability. Therefore, it is important meet requirements related to internal and external quality but also quality in use. Papers in these proceedings address different aspects of quality assurance including technical measures and internal metrics, defect prediction, testing approaches, usability and user interface quality evaluation. Big data application and cloud computing offer new opportunities but also introduce new challenges. Therefore, it is crucial to combine concepts, approaches, methods and theories from different domains, to establish efficient collaboration environments that support interdisciplinary project teams and assist in linking theoretical and practical knowledge.

We hope that these proceedings will be beneficial for your reference and that the information in this volume will be useful for further advancements in both research and industry.

prof. dr. Marjan Heričko
CSS 2015 – Collaboration, Software and Services in Information Society Conference Chair

PREDGOVOR

Konferenco "Sodelovanje, programska oprema in storitve v informacijski družbi" organiziramo v sklopu multikonference Informacijska družba že petnajstič. Kot običajno, tudi letošnji prispevki naslavljajo aktualne teme in izzive, povezane s programskim in storitvenim inženirstvom ter dobrimi praksami uspešnega sodelovanja.

Sprejem in uspešna uporaba na informacijskih tehnologijah temelječih storitev je v veliki meri odvisna od njihove zanesljivosti in kakovosti. Zato je pomembno, da zagotovimo ustrezno notranjo in zunanjo kakovost ter zadovoljimo zahtevam glede kakovosti rešitev v uporabi. Prispevki, zbrani v tem zborniku, naslavljajo različne vidike zagotavljanja kakovosti sodobnih informacijskih rešitev in storitev. Naslavljajo tako tehnična merila kakovosti in programske metrike kot pristope k testiranju in vrednotenju uporabnosti ter napovedovanju in odkrivanju pomanjkljivosti. Poseben izziv, a tudi priložnosti, predstavljajo informacijske rešitve in storitve, temelječe na obsežnih podatkovnih zbirkah in računalništvu v oblaku.

Nujno je povezati spoznanja, koncepte, metode in pristope različnih področij ter hkrati vzpostaviti učinkovita okolja za podporo sodelovanju interdisciplinarnih projektnih skupin. Zato upamo, da boste v zborniku prispevkov, ki povezujejo teoretična in praktična znanja, našli koristne informacije za svoje nadaljnje delo tako pri temeljnem kot aplikativnem raziskovanju.

prof. dr. Marjan Heričko
predsednik konference CSS 2015 – Collaboration, Software and Services in Information Society Conference

PROGRAMSKI ODBOR / PROGRAM COMITTEE

Dr. Marjan Heričko
University of Maribor, Faculty of Electrical Engineering and Computer Science

Dr. Ivan Rozman
University of Maribor, Faculty of Electrical Engineering and Computer Science

Dr. Lorna Uden
Staffordshire University, Faculty of Computing, Engineering and Technology

Dr. Gabriele Gianini
University of Milano, Faculty of Mathematical, Physical and Natural Sciences

Dr. Hannu Jaakkola
Tampere University of Technology Information Technology (Pori)

Dr. Mirjana Ivanović
University of Novi Sad, Faculty of Science, Department of Mathematics and Informatics

Dr. Zoltán Porkoláb
Eötvös Loránd University, Faculty of Informatics

Dr. Vili Podgorelec
University of Maribor, Faculty of Electrical Engineering and Computer Science

Dr. Aleš Živkovič
Innopolis University, Faculty of Computer Science

Dr. Boštjan Šumak
University of Maribor, Faculty of Electrical Engineering and Computer Science

Dr. Gregor Polančič
University of Maribor, Faculty of Electrical Engineering and Computer Science

Dr. Luka Pavlič
University of Maribor, Faculty of Electrical Engineering and Computer Science

Predicting Software Defect-Proneness from Software Repository Data – A Case of Eclipse Bug Data

Vili Podgorelec
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
+386 2 220 7353
vili.podgorelec@um.si

Sašo Karakatič
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
+386 2 220 7360
saso.karakatic@um.si

ABSTRACT

In this paper, we present the results of predicting software errors within a set of software engineering datasets, namely the publicly available Eclipse Bug Data, using our MPGT classification algorithm. Previous studies showed that obtaining high predictive accuracy of positive cases (defect-prone software modules) while preserving low predictive error rate of negative cases (not defect-prone modules) is not an easy task. As the MPGT algorithm was designed for constructing decision trees which should provide balanced predictions (in terms of averaged class accuracy), we decided to test the MPGT's performance on predicting the software defect-proneness. The results show that MPGT outperformed other standard decision tree induction algorithms.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning – *induction, knowledge acquisition.* I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search – *heuristic methods.* D.2.5 [**Software Engineering**]: Testing and Debugging – *diagnostics, monitors, testing tools.*

General Terms

Algorithms, Reliability, Experimentation, Verification.

Keywords

Machine learning, Software errors prediction, Decision trees, Evolutionary algorithms.

1. INTRODUCTION

Recent research results from the fields of machine learning and data mining have provided practical tools for using data miners to automatically learn predictors for software quality [1]. In a never-ending quest of optimizing the software projects' budgets, such predictors can be used to focus the testing on parts of the system that seem defect-prone. For this purpose, the predictors are built upon static code attributes from the software code repositories. In general, many different static code attributes can be used for learning predicting models. Although there are some researchers who oppose the value of static code attributes as defect predictors [2], most of them nevertheless endorse them [3, 4, 5].

On the other hand, there are numerous learning methods that can be used to construct the predictors, like neural networks, support vector machines or decision trees, to name a few. To choose the right one is not the easiest task to do, and the choice of learning method is even far more important than which subset of the available data is used for learning [1]. Different decision tree (DT) learning methods were often used in prior work [3, 4], due to their efficiency and straightforwardness. However, the big problem of the majority of the existing learning methods is that they generally tend to maximize the predictive accuracy. Accuracy is a good measure of a learner's performance when the possible outcomes occur with similar frequencies. In the software engineering datasets, however, this is not generally the case. Therefore, a good learner to be used for predicting software errors should provide high predictive accuracy of positive cases (defect-prone software modules) while preserving low predictive error rate of negative cases (not defect-prone software modules).

In our previous work, we have designed a multi-population genetic algorithm for construction of DTs called MPGT [6]. Its main advantage over the existing DT induction algorithms is the ability of constructing prediction models which should provide balanced predictions. For this purpose, we performed an experiment in which we tested the predictive performance of MPGT on three versions of Eclipse Bug Data datasets and compared the results with the best known traditional DT induction methods.

2. MINING THE SOFTWARE DATASETS

Predicting defect-prone software components is an economically important activity and so has received a good deal of attention. Current defect prediction work focuses on 1) estimating the number of defects remaining in software systems, 2) discovering defect associations, and 3) classifying the defect-proneness of software components, typically into two classes, defect-prone and not defect-prone [7]. This paper is concerned with the third approach.

The prediction result can be used as an important measure for the software developer [8] and can be used to control the software process. Being able to predict which components are more likely to be defect-prone supports better targeted testing resources and can therefore improve efficiency. However, in order to use the prediction results, one needs to evaluate them beforehand thoroughly, and possibly also to validate them. Within the machine learning community there are many measures that can be used to evaluate the quality of a predicting model. However, how to adequately validate the learning methods and their learned models still represents a major challenge. Incomplete or inappropriate validation can result in unintentionally misleading results and over-optimism on the part of the researchers [7].

For this purpose, in [1] authors discussed about how a learned predictor could be evaluated. A predictor is learned to identify defect-prone modules. In this task, a predictor can either identify truly erroneous modules as defect-prone – such modules are then marked as true positives. On the other hand, predictor can also mark as defect-prone modules which are not erroneous – such modules are then marked as false positives. An ideal predictor would predict all defect-prone modules as erroneous, achieving the true positive rate (*tpr*) of 100%, and simultaneously all the not defect-prone modules as non-erroneous, achieving the false positive rate (*fpr*) of 0%. In practice, engineers balance between *tpr* and *fpr*. To operationalize this notion of balance, in [1] authors defined a new performance measure called *balance*, which is used to choose the optimal (*tpr*, *fpr*) pairs:

$$balance = 1 - \frac{\sqrt{(0-fpr)^2+(1-tpr)^2}}{\sqrt{2}} \qquad (1)$$

3. MULTI-POPULATION GENETIC TREE

3.1 Multi-Population Genetic Algorithms for Decision Tree Induction

Genetic algorithm (GA) is capable of exploring a wide range of search space when the selection pressure is properly controlled, while crossover and mutation evolve solutions towards local optima, keeping the needed genetic diversity. The evolutionary search for the solution is directed towards the optimal solution based on a predefined fitness function.

In this paper, our aim is to find the best DT for a given software engineering dataset. As we want to optimize the DT's predictive performance regarding several criteria (overall accuracy, average class accuracy, true positives rate, false positives rate, etc.) as well as achieve low complexity of the constructed solution (a small DT), the optimization problems becomes a multi-objective one.

In our previous work, we have designed and evaluated a multi-population genetic algorithm called MPGT (multi-population genetic tree) for the purpose of constructing DTs which are able to provide balanced solutions regarding different criteria [6].

3.2 Defining the fitness function

MPGT consists of two co-evolving subpopulations which employ the same initialization, tournament selection, crossover and mutation operators. Each subpopulation, however, has a different fitness function, which optimizes a different objective. In a regular cycle, after a predefined number of generations (migrate interval) the exchange of DTs between the two subpopulations occurs according to a predefined parameter (migrate rate). MPGT is outlined in Fig. 1.

Originally, MPGT uses the following two fitness functions for the two co-evolving sub-populations (note that both fitness functions are penalty functions, which means that the lower value represent a better solution):

$$ff_{pop_1} = (1 - fsc) + \frac{1}{S} \cdot \frac{n}{sf} \qquad (2)$$

$$ff_{pop_2} = \left(1 - \frac{1}{K}\sum_{i=1}^{K} acc_i\right) + \frac{1}{S} \cdot \frac{n}{sf} \qquad (3)$$

where S is the total number of data instances, n is the number of nodes in the tree, sf is a size factor that defines how many additional nodes outweigh one misclassified instance (we set it to 10), K is the number of decision classes, acc_i is the accuracy of the i-th class, and fsc is the F-measure (also called F-score) criterion – a harmonic mean of the precision and recall values.

For the use with EBD datasets in this paper we, however, replaced the F-score measure from the ff_{pop1} with the balance measure explained above, obtaining:

$$ff_{pop_1} = (1 - balance) + \frac{1}{S} \cdot \frac{n}{sf} \qquad (4)$$

By making this change, we wanted of course the evolution to search for DTs with high balance measure – achieving high true positive rate (correct prediction of defect-prone modules) while keeping the false positive rate (false prediction of not defect-prone modules as being erroneous) low.

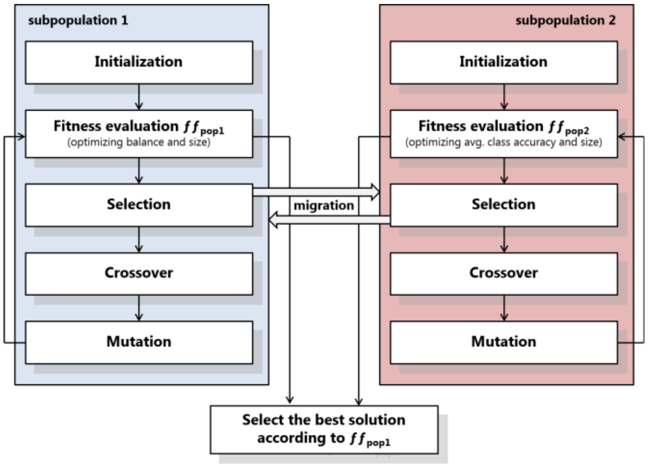

subpopulation 1
Initialization
Fitness evaluation ff_{pop1}
(optimizing balance and size)
Selection
Crossover
Mutation
subpopulation 2
Initialization
Fitness evaluation ff_{pop2}
(optimizing avg. class accuracy and size)
Selection
Crossover
Mutation
migration
Select the best solution according to ff_{pop1}

Figure 1. The outline of MPGT learning algorithm.

4. EXPERIMENT

To assess the performance of the MPGT algorithm on software engineering data, we performed an experiment over three different versions (2.0, 2.1, and 3.0) of publicly available datasets from the Eclipse Bug Data (EBD) project [9]. The EBD datasets were prepared by mining the Eclipse bug and version databases to map failures to Eclipse components. The resulting datasets list the defect density of all Eclipse components and can be used to relate code, process, and developers to defects. EBD 2.0 contains 6.740 files, EBD 2.1 contains 7.900 files, and EBD 3.0 contains 10.600 files. Each case (description for one file) contains the following information: name, pre-release defects, post-release defects, complexity metrics, and structure of abstract syntax tree(s). There are altogether 200 attributes, and each case can be determined as either defect-prone (containing some known post-release defects) or not defect-prone (there are no known post-release defects). All three datasets are very imbalanced: in EBD 2.0 there are only 14.49% defect-prone classes, in EBD 2.1 10.83%, and in EBD 3.0 14.80%.

First, we constructed DTs over all three versions of EBD using some of the best and most known DT induction algorithms: J48, CART, Random Tree (RT) and Random Forest (RF), using the Weka framework [10]. To be precise, RF method does not exactly construct DTs but rather an ensemble of DTs; however, we decided to include it because of its known predictive power. Then, we constructed DTs with our MPGT algorithm, using the default settings for all the evolutionary parameters [6] and the two fitness functions described above.

For all three EBD datasets and for all the learning methods we used the 10-fold cross-validation. Each method was tested on each fold (a pre-divided training/test set combination) on all the datasets. All results are thus based on the average of all ten folds, whereas the results of MPGT are additionally averaged from 10 evolutionary runs per fold (giving 100 evolutionary runs per dataset). All the reported results are for the test sets.

4.1 Results

For all three EBD datasets and for all five learning methods used we calculated the following measures: accuracy (*acc*), average F-measure (*afm*, average of both classes' F-measure), average class accuracy (*aca*, average of both classes' accuracy), true positive rate (*tpr*), false positive rate (*fpr*), balance (*bal*), and tree size (*size*).

From Table 1 we can see that Random Forest produced DTs with the highest accuracy for all three versions of EBD. On the other hand, MPGT constructed DTs with the lowest accuracy. If the datasets would have been balanced, these results would be very unfavorable for the MPGT method. However, considering the high imbalance of the data, a trivial (useless) model that would identify each case as not defect-prone would achieve between 85% and 89% percent accuracy.

For this purpose we compared the averaged F-measure (Table 2). The F-measure, a harmonic mean of precision and recall, somewhat reduces the problem of accuracy measure in the case of non-balanced data; but in case of highly imbalanced datasets, also F-measure can be somewhat misleading. From Table 2 it can be seen that J48 in two cases and CART in one case achieved the best result. The MPGT ranked somewhere in the middle within this measure. It is interesting that Random Forest ranked much worse than in the case of accuracy – in two cases it scored worse than MPGT.

The third measure is the average class accuracy (Table 3), calculated as the mean of each decision class' accuracy. The frequency of instances of a single class within the dataset is not important here – each single decision class contribute equally to the final score. As it can be seen from Table 3, the MPGT outperformed all other learning methods considerably in this measure.

The next three measures, true positive rate (*tpr*), false positive rate (*fpr*), and *balance*, probably give the best information about the quality of produced predictors (Tables 4 and 5). As it can be seen from Table 4, by far the highest *tpr* (73% on average) was achieved by MPGT. All the other learning methods failed considerably in this measure, achieving on average from 23.9% (Random Forest) up to 37.8% (J48). A reasonable *fpr* of 25.1% was achieved by MPGT, while the other methods achieved from 1.5% (Random Forest) up to 10.5% (Random Tree). These numbers are also reflected in the *balance* results (Table 5). It can be seen that MPGT again outperformed all other methods considerably (almost by a third), achieving an average *balance* of 73.8%, while the other methods achieved from 46.2% (Random Forest) up to 55.7% (J48).

Table 1. Accuracy for five learning methods.

	MPGT	J48	CART	RT	RF
EBD 2.0	74.2	85.9	87.9	82.6	88.7
EBD 2.1	73.9	86.7	89.7	83.5	89.9
EBD 3.0	75.8	84.4	87.1	80.7	87.3

Table 2. Average F-measure for five learning methods.

	MPGT	J48	CART	RT	RF
EBD 2.0	64.6	69.7	69.8	65.2	69.7
EBD 2.1	60.5	62.2	57.9	58.8	59.9
EBD 3.0	65.3	67.1	66.4	61.7	63.9

Table 3. Average class accuracy for five learning methods.

	MPGT	J48	CART	RT	RF
EBD 2.0	75.1	68.7	66.7	65.3	65.6
EBD 2.1	73.2	61.2	56.1	59.1	57.3
EBD 3.0	73.5	66.1	63.2	61.7	60.7

Table 4. TPR and FPR for five learning methods.

	MPGT	J48	CART	RT	RF
EBD 2.0	76.3 26.1	44.5 7.1	36.8 3.4	41.0 10.4	33.1 1.9
EBD 2.1	72.3 25.9	28.7 6.3	13.2 1.1	27.9 9.7	15.6 1.0
EBD 3.0	70.2 23.2	40.0 7.9	29.4 3.0	34.8 11.4	22.9 1.6

Table 5. Prediction balance for five learning methods.

	MPGT	J48	CART	RT	RF
EBD 2.0	74.9	60.4	55.2	57.6	52.7
EBD 2.1	73.1	49.4	38.6	48.5	40.3
EBD 3.0	73.2	57.2	50.0	53.2	45.5

The comparison of *tpr* and *fpr* can be even better observed on Figure 2, and the comparison of *balance* for all learning methods on Figure 3.

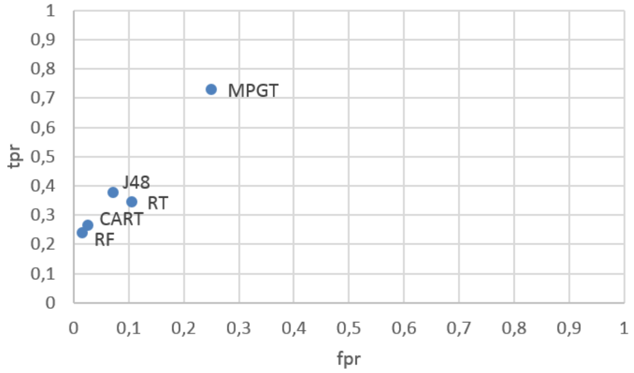


Figure 2. The (tpr, fpr) plot for all five learning methods. Note that the ideal method would achieve (tpr=1, fpr=0).

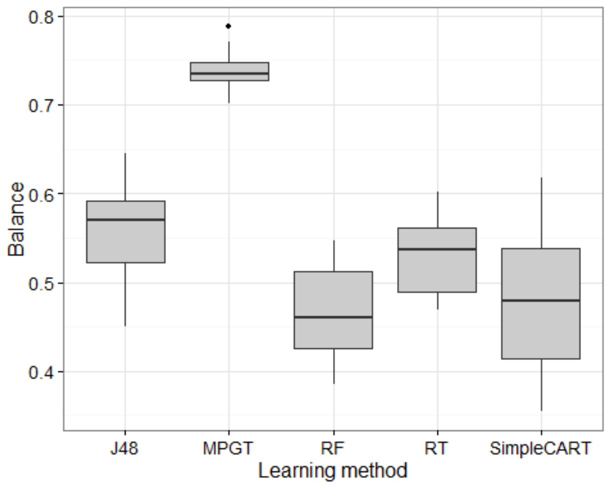


Figure 3. The comparison of balance for five learning methods.

The last measure is the size of the constructed DTs (Table 6), expressed with the number of nodes (Random Forest produces a combination of trees, with higher complexity than a single DT, so its size is not reported). It can be seen that MPGT produced the smallest DT in each of three EBD datasets. Only CART, which constructs oblique DTs as opposed to axis-parallel DTs of other methods, produced DTs of comparable size to MPGT, while J48 and especially Random Tree produced DTs of much bigger size.

Table 6. Tree size for five learning methods.

	MPGT	J48	CART	RT	RF
EBD 2.0	8.2	438.2	34.2	1272.2	-
EBD 2.1	7.7	427.0	7.8	1452.2	-
EBD 3.0	9.3	695.0	27.0	2315.8	-

5. CONCLUSIONS

Our research confirmed the difficulty of predicting the defect-prone software modules. The majority of the existing learning methods are overly optimistic and identify more than a half of actually erroneous modules as not being defect-prone. The consequence is the low *balance* of the constructed predictors. By employing our evolutionary MPGT algorithm for construction of DTs, we succeeded, however, to achieve considerably higher *balance*, while keeping the complexity of the constructed predictor very low. These results give us a sound platform to build upon in our quest of linking the worlds of machine learning and software engineering and providing software developers with an efficient software defect-proneness tool.

6. REFERENCES

[1] Menzies, T., Greenwald, J., and Frank, A. 2007. Data mining static code attributes to learn defect predictors. *IEEE Trans. Software Eng*. 33, 1 (Jan. 2007), 2-13. DOI= http://dx.doi.org/10.1109/TSE.2007.256941.

[2] Fenton, N., and Ohlsson, N. 2000. Quantitative analysis of faults and failures in a complex software system. *IEEE Trans. Software Eng*. 26, 8 (Aug. 2000), 797-814. DOI= http://dx.doi.org/10.1109/32.879815.

[3] Porter, A., and Selby, R. 1990. Empirically guided software development using metric-based classification trees. *IEEE Software*. 7, 2 (Mar. 1990), 46-54. DOI= http://dx.doi.org/10.1109/52.50773.

[4] Khoshgoftaar, T. M., Allen, E. B., Jones, W. D., and Hudepohl, J. P. 2000. Classification-tree models of software-quality over multiple releases. *IEEE Trans. Reliability*. 49, 1 (Mar. 2000), 4-11. DOI= http://dx.doi.org/10.1109/24.855532.

[5] Menzies, T., Di Stefano, J. S., Chapman, M., and McGill, K. 2002. Metrics that matter. In *Proceedings of the 27th Annual NASA Goddard Software Engineering Workshop*. SEW-27'02. IEEE Computer Society Washington, DC, 51-57. DOI= http://dx.doi.org/10.1109/SEW.2002.1199449.

[6] Podgorelec, V., Karakatič, S., Barros, R. C., and Basgalupp, M. P. 2015. Evolving Balanced Decision Trees with a Multi-Population Genetic Algorithm. In *Proceedings of the 2015 IEEE Congress on Evolutionary Computation*. CEC2015. IEEE Computer Society, 54-61. DOI= http://dx.doi.org/10.1109/CEC.2015.7256874.

[7] Song, Q., Jia, Z., Shepperd, M., Ying, S., and Liu, J. 2011. A General Software Defect-Proneness Prediction Framework. *IEEE Trans. Software Eng*. 37, 3 (May 2011), 356-370. DOI= http://dx.doi.org/10.1109/TSE.2010.90.

[8] Kenney, G. Q. 1993. Estimating Defects in Commercial Software during Operational Use. *IEEE Trans. Reliability*. 42, 1 (Mar. 1993), 107-115. DOI= http://dx.doi.org/10.1109/24.210280.

[9] Zimmermann, T., Premraj, R., and Zeller, A. 2007. In *Proceedings of the Third International Workshop on Predictor Models in Software Engineering*. PROMISE’07. IEEE Computer Society Washington, DC, 9-15. DOI= http://dx.doi.org/10.1109/PROMISE.2007.10.

[10] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. 2009. The WEKA Data Mining Software: An Update. ACM SIGKDD Explorations Newsletter, 11, 1 (Jun. 2009), 10-18. DOI= http://dx.doi.org/10.1145/1656274.1656278.

Graph-based analysis and metrics of evolving open-source software systems

Sanja Grbac Babić
Polytechnic Pula, College of Applied Sciences
Riva 6
HR-52100 Pula, Croatia
sanja.grbac@politehnika-pula.hr

Tihana Galinac Grbac
Faculty of Engineering, University of Rijeka
Vukovarska 58
HR-51000 Rijeka, Croatia
tihana.galinac@riteh.hr

ABSTRACT

Since the software systems are constantly evolving and becoming more complex, in recent years have been developed new methods and approaches which allows research, development and prediction in complex systems. Therefore, the complex network theory is used in recent years in analysis of software systems. Here in this paper we will analyze selected metrics from complex network theory on large and complex open source software in ten versions of its evolution.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics complexity measures

General Terms

Measurement, Experimentation

Keywords

Network Science, complex network, software analysis, graph, metrics, small-world, centrality.

1. INTRODUCTION

Many real systems can be represented as network, where the elements of the system are nodes and interactions between elements are edges. Representing a system as a graph allows us to process and analyse data from different scientific fields. So recently, graph-based analysis of complex systems has become known as Network Science [1]. A graph can be used to model a wide range of real systems represented as networks, where a node could be a person, city, airport, neuron, species, or webpage etc. and an edge could represent a relationship, road, line, synapse, relation or hyperlink etc., respectively. In Table 1 we give some examples of real-world networks.

Therefore, network science is very common and used in computer science, sociology, transportation, economics and finance, biology, ecology and other sciences, and it's become a crucial component of modern science [1].

In this paper we discuss the results of the graph-based analyses for ten versions of evolving open-source software system. We aim to study software evolution by analyzing network properties of large-scale software systems.

Table 1: Real-world types of complex networks

	Node	Edge
Biological network	cell, neuron, protein, gene, specie	metabolic pathway, protein interaction, prey-predator relation
Social network	people, companies	contact, friendship interaction, business relation
Technological network	infrastructure, city, station, airport, website, consumer	rail, road, airline, router, power grid
Information network	www document, data	URL link

The paper is organized in four sections. In section 2 we briefly introduce complex networks, and graph theory, we define metrics and describe small-world network properties, proposing applications of the metrics in the real-world networks analysis. In section 3 we present the results of our preanalysis, discussing then in section 4. Finally in section 5 we conclude proposing our future work.

1.1 About work

In this paper, we introduce and analyze different metrics that appear in software graph based analyses. We investigate these properties for a selection of real-world networks and how they can be applied in other complex networks. We put on comparison between different versions of Java development tools (JDT).

2. COMPLEX NETWORKS

Studying complex networks today is a major challenge in many scientific disciplines, because it is very applicable to different networks, but the complex network theory is also used in recent years in analysis of software systems [1, 2]. The results of the graph-base analysis are known as graph metrics, and they give us lots of information about network.

We can define a graph as an ordered pair *G (V, E)*, where *V* are the nodes or vertices, and *E* is a set of edges (links) between those nodes. A graph can be undirected or simple if the edges have no orientation, so they just connect nodes, or can be directed if the edges are oriented, so they are arrows from one node to another.

In computer science, a graph is the basic object of study in graph theory and it can represent all kinds of networks.

In software engineering we can apply graph theory for modelling software in many different ways. For example, nodes may represent modules, classes, objects or functions, and edges may encode dependencies between them, calls, heritage, etc. One may then study software through such graphs, called complex networks. In this way, we have a different vision and we opened up new horizons in the analysis of software.

2.1 Definition of network metrics

Complex networks can be more clearly understood if we know to interpret some value resulting from the analysis of the correspondent graph. Here we define metrics that we use in our preanalysis. Some of this metrics are also analyzed in [1, 2, 3, 5].

The degree (***D***) of a node is the number of edges connected to it. For the directed graph we distinguish in-degree as the number of edges entering in the node, and out-degree as the number of edges coming out from the node.

The average degree is the average of node degrees in the graph. Graphs with high average degrees tend to be tightly connected [3].

Average Path Length or distance (***L***) is the average of all the shortest paths for all possible pairs of nodes in the network. We can say that it is also defined as the average number of steps it takes to get from one node of the network to another, so it indicates the global connectivity [5].

We can define the **diameter *D(G)***of a network as the longest of all the calculated shortest paths in a network [3], but can also say that the diameter is the longest distance between two nodes in the network. [1].

The clustering coefficient (***C**ᵢ*) of a node measures clustering degree of complex networks, and that is the degree of local transmission between network nodes [1, 3].

We can define the clustering coefficient for each node i, supposing that k_i is the number of nodes connected to it (degree of a node i), and there are e_i links between those nodes:

$$C_i = \frac{e_i}{k_i(k_i - 1)}$$

The clustering coefficient (***C***) of the whole network is the average of all N nodes’s clustering coefficient.

$$C = \frac{1}{N}\sum_{i=1}^{N} C_i$$

So, the clustering coefficient can assume just values $0 \leq C \leq 1$, and it is significant for nodes with at least two neighbors [3]. This means that when $C = 0$ all the nodes are isolated nodes, and when $C = 1$ the entire network is connected, and any pair of two nodes are connected directly.

The higher network clustering coefficient, the greater is the flow of energy and information among all nodes in the network [3].

Betweenness centrality (***BC***) is a fraction of shortest paths that pass through the node. It measures how often a node appears on shortest path between nodes in the network. A node can be considered highly important if it forms bridges between many other nodes.

Closeness centrality (***CC***) is defined as the average distance from a given starting node to all other nodes in the network.

The **eccentricity** (***ECC***) of a node i in a graph *G* is the maximum distance from i to any other node. According to this definition the diameter *D(G)* of a graph *G* is the maximum eccentricity over all vertices in a graph.

In complex network and social network analysis, centrality is a measure that indicates the importance of nodes in a network. Special attention should be given to the nodes with high *BC* during the development, and those with high *CC* can be adopted for effective and efficient software testing [4].

2.2 Small-world network

A small-world network is a model used to explain many real large complex networks [1, 6]. It is type of graph in which any two nodes have a high probability of being reached through a short path of intermediate nodes, but can be easily reachable. A small-world network is defined as a network where the distance *L* between any two nodes is a logarithmic function of the number of nodes *N* in the network:

$$L =\propto \log N$$

In a small-world network most nodes are not neighbors of one another. A network is characterized by the small-world phenomenon has the diameter of the corresponding graph relatively small and high clustering coefficient [1, 5].

2.3 Applications of the metrics in the real-world networks

These graph metrics are also applicable for analyzing biological networks, where the type of centrality metrics depends on the type of the network. For example, considering gene regulatory networks we can apply centralities degree, shortest-path, closeness centrality, betweenness centrality.

Technological network – airport network can be analyzed using this metrics: degree, betweenness centrality, closeness centrality. This method of analysis can help in identifying nodes (airport) whose connectivity needs to be improved, or to find out redundancy in the network.

In social network metrics to be considered are node degree because it measures network activity for a node. Most active nodes are those with highest degree, the shortest paths in the network are more important, because all information flows along the network's shortest paths only, high betweenness centrality shows us node with great influence over, high closeness centrality of a node allows to access all the nodes in the network more quickly than from other nodes.

2.4 Software evolution employing network models

Since the software system can be represented and threated as a network graph, where the nodes are software modules (classes) and communication links (relations) are edges, we can apply graph-based analyses of network. Using graph metrics we can ponder software system properties and predict the future evolution. From the previous work [2] it has been ascertained that the graph metrics are giving us much more information than the classic metrics, and it has been demonstrated in [3] how graph metrics can be used to predict software evolution.

Previous results [2] indicate that after a first period of intensive growth of size and complexity of software system network in the later evolution version network measures tend to stabilize, so we analysed the defined metrics for an evolving open-source software system in his 10 versions.

Table 2: Metric values

Version	V	E	D	Av. Path length	Av. degree	Clustering coefficient	Eccentricity	Closeness Centrality	Betweenness Centrality
1.0	803	2650	4	1,6681	6,6002	0,0613	1,2328	1,0188	6,3972
2.0	1429	5871	5	1,6107	5,4334	0,0404	1,3830	0,9901	5,4334
3.0	2463	11817	6	1,6679	9,5956	0,0632	1,4433	1,1318	8,2400
3.1	2787	13752	6	2,5430	9,8704	0,5381	2,7333	1,7836	32,6823
3.2	3305	16482	6	2,5919	9,9739	0,0491	2,8003	1,9035	26,3470
3.3	3460	17568	5	2,5818	10,1549	0,0486	2,8090	1,9067	26,7396
3.4	3614	18834	5	1,6573	10,4228	0,0520	1,5182	1,1330	8,0910
3.5	3736	19550	5	2,5693	10,4657	0,0525	2,8160	1,9028	27,4389
3.6	3780	19879	5	2,5670	10,5180	0,0520	2,8190	1,9043	27,4613
3.7	3808	19987	5	2,5679	10,4974	0,0504	2,8190	1,9043	27,4330

3. ANALYSIS

For our research we have chosen ten versions of Java development tool, because that gives us good sample for our statistical analysis. The source code files have been collected from Eclipse[1] repositories, but for obtaining network files has been used rFind tool. We have chosen to analyze network files for open source projects written in Java provided from previous work [7, 8].

In our work we have analyzed ten versions of JDT software system using Gephi[2] as a tool for exploring and manipulating networks [6]. We analyzed this JDT open-source projects as evolving open-source software systems that are big enough to measure metrics of a large-scale systems and compare it with the analysis provided in smaller software project [1, 3, 5].

The results of our graph-based preanalysis are provided in the Table 2.

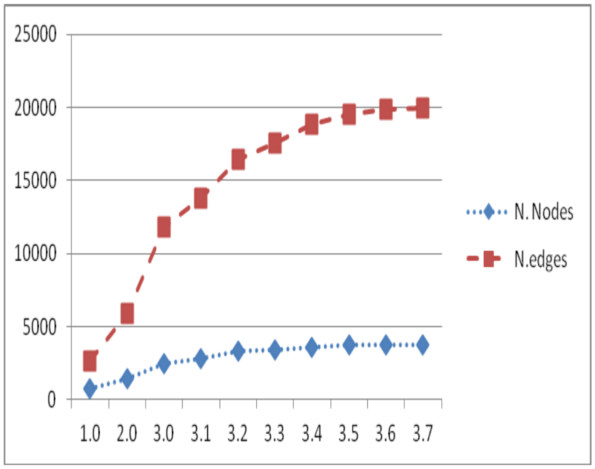


Figure 1: The growth of number of nodes ande edges over the versions

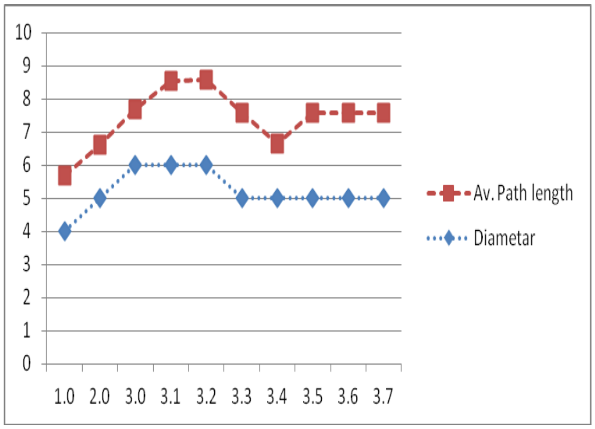


Figure 2: Average Path Length and Diametar over versions

The results show that the number of nodes and links is considerably growing from version to version, as shown in Figure 1, but we can notice a reduced growth in the last versions of JDT software systems. Despite this rise we notice that for our software the diameter tends to be 5 constantly in the last five versions as shown in Table 2 (column 4) and in Figure 2.

Related to this values is the average path length of each version which slightly increases after the first three versions, but after the increase in the last versions it has relatively stable value.

Average node degree indicates in some way popularity of the node. In our preanalysis the average degree is increasing from initial 6,6 value to 10,5 in the last version and it is relatively high, indicating that the nodes have a good connection between each other, and are becoming more connected although the entire network is much bigger.

The clustering coefficient of the entire network is very small, around 0,05 and we notice the tendency of slight deterioration in last versions. For each version we noticed that almost half of all the nodes have C_i equal to zero. This is confirming a good software engineering practice [3] which refers to the less complicated software testing, maintaining and evolving.

[1] http://git.eclipse.org/c/

[2] http://gephi.github.io/

It is very significant to consider the centrality of the nodes in complex network analysis, because it indicates the important nodes in the network. But the problem is how to interpret importance, because of different point of considering centrality we have different metrics. We can consider a node important if it has lot of connections with other nodes (high degree), or because it forms bridges between many other nodes (high betweenness centrality), for this reasons we clarify the results of our analysis on three graph metrics that are significant for considering node centrality, and are *BC, CC, ECC*.

From our analyses we noticed that those nodes that have *ECC* equal to zero have also *CC* and *BC* equal to 0, and in every version we have a lots of nodes having *BC, CC* and *ECC* equal to zero. The obtained results of *CC* and *ECC* for each node in every version are correlated with the average path length and in accordance with the definition.

Although almost all the nodes in each version have *BC* less than 1, the remaining nodes assume different values, and we noticed that in every version of software system we have a few nodes with high betweenness centrality, that means that those nodes are some kind of centre of the network and we can consider it important, so the changes on that nodes have impact on many other nodes in the network.

4. DISCUSSION

Although the system is increasing throw the version the diameter is constantly low, and in relation to him the average path length is low and almost stable. As a small-world phenomenon implies the diameter is relatively small, although the network is big and our results are in accordance with the logarithmic function provided in the definition of a small-world network.

From the Table 2 we can notice one exception during evolution of our software system, precisely the version 3.4. Software system version 3.4 has values of some graph metrics (Av. Path length, Eccentricity, Closeness Centrality, Betweenness Centrality) which do not follow the other trends, so we can confirm that during the evolution not always the same metrics are evolving from one version to another.

We notice a high number of nodes and edges in our software-system much bigger than in other analyses [1, 3, 5], but small diameter, and high average degree. Our values of clustering coefficient are much smaller in comparison with the previous analyses [1, 3, 5]. Using centrality metrics we can determine isolated and popular nodes, and then focus on this nodes to predict evolution of the software system.

Our preanalysis confirms that the metrics do not change significantly through the evolution of the same open-source software system, so we have observed that indeed, software structure is increasing throw the evolution the graph metrics values are similar, which suggests homogeneity of the versions of the same software system.

5. CONCLUSIONS AND FUTURE WORK

As the software systems continuously enhance and become more complex it is necessary to enable the development with minimum costs and prevent changes that lead to failure. If we are based on complex networks approach to analyse the graph and its metrics we can achieve the desired expectations. If the network has a small number of nodes that have strong impact on network it will be easier for us to point on them and we can better manage development and maintenance of software. We analyzed basic metrics of complex network on ten versions of open-source software system. Our software is selected to be with large amount of nodes and edges, because we aimed to confirm recent findings on software networks, but on large networks and in software evolutions. It is even possible that the version numbers do not match the sequence of evolution, as demonstrated by the results of the analysis in Table 2. We can see that some versions (3.4) come in a jump as shown in Figure 2.

In future we will expand our analyses on more complex metrics, such as degree distribution, and try to see if it is in accordance to the power-law degree distribution. Also, in future, we will include in analysis new evolutions of open-source software systems. This will enable the analysis of a larger set of versions and improve the prediction of software evolution.

6. ACKNOWLEDGMENTS

This work has been supported in part by Croatian Science Foundation’s funding of the project UIP-2014-09-7945.

7. REFERENCES

[1] Wen, L., Kirk, D., and Dromey, R.G. 2007. *Software System as Complex Networks. On* Cognitive Informatics, 6th IEEE International Conference on , pages 106-115, Lake Tahoo, CA, 2007.

[2] Chaikalis, T. and Chatzigeorgiou, A. 2015. *Forecasting Java Software Evolution Trends Employing Network Models*. On Software Engineering, IEEE Transactions on (Volume:41 , Issue: 6), pages 582-602, 2015.

[3] Bhattacharya, P., Iliofotou, M., Neamtiu, I., and Faloutsos, M. 2012. *Graph-Based Analysis and Prediction for Software Evolution,* ICSE 2012: 419-429

[4] Šubelj, L., and Bajec, M. 2012. *Software Systems through Complex Networks Science: Review, Analysis and Applications,* On First International Workshop on Software Mining, pages 9-16, Beijing, China, 2012.

[5] Valverde, S. and Solé R.V. 2007. *Hierarchical Small-Worlds in Software Architecture,* Special Issue on Software Engineering and Complex Networks Dynamics of Continuous, Discrete and Impulsive Systems Series B, 2007.

[6] Bastian, M., Heymann, S. and Jacomy, M. 2009. *Gephi: An Open Source Software for Exploring and Manipulating Networks,* On Third International ICWSM Conference 2009., San Jose, California 2009.

[7] Petrić, J. and Galinac Grbac, T. 2014. *Software structure evolution and relation to system defectiveness,* EASE '14 Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, London 2014.

[8] Petrić, J., Galinac Grbac, T. and Dubravac, M., 2014. *Processing and data collection of program structures in open source repositories,* Proceedings of SQAMIA 2014, Lovran 2014.

How to Evaluate the Technical Quality of Python Projects

Robert Malačič
Business School Murska Sobota
Murska Sobota, Slovenia
robert.malacic@gmail.com

ABSTRACT

Software metrics give us the possibility of evaluating a software project on the base of its design quality, which means the ability to assess the quality of its program code. To assess a software solution we must first determine and have an overview of its metrics. Metrics give us numerical data on the state of software code, its reliability and the state of the code comments. With the use of information from software metrics, it is possible to evaluate software projects. Metrics provide us with information about whether certain software solutions meet standards, as well as which of the chosen metrics are suitable in its measure, etc.

The use of software metrics also allows us to calculate technical debt, which gives us, as a result, the value of how many hours of software design a software engineer would need to fix the created debt. The monitoring of technical debt during the development process of the project allows us to control the quality of the source program code and therefore contributes to a reduction in costs.

In our preliminary work, we will research, if there are any software tools capable of calculating technical debt for Python projects. In addition, we will search for tools that support the calculation of object-oriented metrics for Python projects.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics complexity measures

General Terms

Management, Measurement, Design, Economics, Languages, Theory.

Keywords

Technical debt, Software metrics, Object oriented metrics, Programming languages, Python.

1. INTRODUCTION

For quality assurance, software solutions metrics should be surveyed, which are an important indicator of the critical details in the development, implementation and maintenance of software solutions. In the life cycle of software solutions, and during the process of developing a software solution, we have the ability to monitor internal metrics and to survey the external metrics after the implementation of software solutions. [1]

By combining all software life cycle development phases into one organization, it is possible that the development and maintenance creates synergies. By collecting the measurements of metrics in a development cycle, we can influence the issues that might occur during maintenance time. [1]

1.1 Software Metrics

Software metrics are used to identify poor planning and poorly designed software code. Metrics help programmers make it easy to follow their work and progress in a programming cycle; they also ensure the satisfactory quality of the projects. [2] Even monitoring the development process can determine whether a sudden increase in the number of code rows in a given time period is the result of copying and pasting the code. [2] [3]

In the process of choosing what metrics are to be used in measurement in order to predict software design quality, the first thing that has to be considered is from which viewpoint the measurements of metrics are to be evaluated, i.e. what is the main goal of the measurements. [3]

Object-oriented (OO) metrics (e.g. CK – Chidamber and Kemerer) can help users understand the complexity of object-oriented model, in anticipating the occurrence of errors in the testing and maintenance of software solutions. [1] [4]

Classic metrics are used to determine the index of maintenance and for calculating computational complexity. [5] [6] [7]

We want to determine which metrics authors of articles and publications are using in their research and compare their results with our own findings.

1.2 Technical Debt

Ward Cunningham invented the term technical debt in the year 1992. [8] The term technical debt describes the price of wrong approaches and the use of shortcuts in the development and maintenance of software. The term technical debt is more of a metaphor, and not a scientific theory or concept. While it is not a new phenomenon, it is associated with older definitions, such as software decay, software aging, risk management, the research of software quality, etc. However, the technical debt metaphor guides research in architecture, software metrics, the quality of software, and risk management software. [9] [10] [11] [12] [13]

The most common metrics used for monitoring technical debt are as follows: [3] [14]

- metrics to calculate code coverage;
- coupling and cohesion between objects – for the code to be maintainable it can't be too complex, i.e. contain too many predicates, too much coupling and cohesion between objects or be messily written in general. Good understandable code means that it is well commented and written in a structured way with proper indenturing. These design issues do not reflect any part of the object-oriented design in a system and are therefore not covered in this report.
- cyclomatic complexity – the primary purpose of the metric is to evaluate the test and maintainability of software modules. Another practical use of the metric is that it can be used as an indicator of reliability in a software system. Experimental studies have indicated a strong correlation between the McCabe metric and the number of errors that exist in the source code, as well as the time required to find and correct such errors;
- Halstead software metrics – these metrics are computed statically from the code. Although these metrics are often referenced in software engineering studies and have had a great impact in the area, the metrics have been the subject of criticism over the years. However, one metric that uses parts of Halstead's metrics and has good maintainability predictors, is the maintainability index metric.

1.3 Python

Python is a widely used, general-purpose, high-level programming language, with fully dynamic data types, that automatically manages memory and supports a functional, procedural, structured and object-oriented computer programming paradigm. Python was developed as an open source project, which is operated by the nonprofit organization Python Software Foundation. [15] Python programming language is used by Google, NASA, YouTube, Yahoo, the New York Stock Exchange and many more in the construction and operation of their websites [16] Python interpreters are available for installation on all major operating systems including MS Windows, Linux distributions, Mac OS X, Android, etc.

2. PRELIMINARY RESEARCH GOALS

The aim of preliminary research is to find, with the help of a metric analysis of open source solutions, the reference values for individual metrics, with which we can then assess whether an open source solution is of average, poor (below average) or good (above average) quality. The objectives of the preliminary survey are as follows:

- To select a software tool to calculates metrics for determining the quality of software.
- To select Python open-source software solutions and determine their metrics.
- To examine and determine which metrics can assist in the selection of optimal solutions among similar Python projects.
- To discover which metrics ues authors of articles and publications in their research and to compare the results with our own findings.

The research questions that we set ourselves were:

1. Is it possible to evaluate a Python solution using "classic" object-oriented metrics (and can the same be done for Java, C #) – which ones can and which cannot?
2. What tools do other authors use to obtain metric properties?
3. Is the concept of technical debt used in other publications for the evaluation of Python projects?

In the following chapters, we presents related research, which deals with technical debt and various software metrics. At the end, one can find the results of our preliminary research.

3. RELATED WORK

In various articles, there are comparisons of methods of metrics computations for the software solutions presented. Software solutions in various programming languages were developed and the metrics for those projects were calculated. Most of those software solutions were projects in Java, C ++, while Python occurs rarely. [9] Python often appears as an additional reference programming language in an attempt to interpret the metrics of various methods of project development. In some articles, however, the authors, due to a lack of open source solutions for measuring metrics of Python projects, have proposed their own metrics. [17] [18]

3.1 Software metrics in Agile Software and Empirical Study

In the article *Software metrics in Agile Software and Empirical Study* the authors answer the following research questions: [9]

- Is it possible to recognize the use of agile methodologies through the analysis of software metrics?
- Are metrics distributions generated from software, and developed using agile methodologies, similar to metrics distributions of software, developed using plan-driven methodologies?
- It is possible to assert that metric distributions generated from agile methodologies are related to a better quality of software?

In the article, the authors are trying to present the results of software metrics distributions for eight open source projects. Five projects were developed using agile methodologies, three projects were developed using plan-driven methodologies, five projects were developed using Java and three projects were developed using Python. They computed ten metrics:

- IFANIN – (Number of immediate base classes);
- NOC – (Number of Children (CK);
- NIM – (Number of instance methods);
- NIV – (Number of instance variables);
- WMC – (Weighted methods per class);
- RFC – (Response For a Class);
- LOC – (Lines of code of the class);
- CLOC – (Lines of comments of the class);
- NOfS – (Number of declared statement);
- DIT – (Depth of Inheritance Tree).

The first result is related to metrics differences between systems developed in Java and systems developed in Python. Python is a general-purpose, high-level OO programming language. Its design philosophy emphasizes code readability and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as Java. The language provides constructs intended to enable clear programs on both a small and large scale. Other results shows that systems developed using Python have a low average value of CLOC than systems developed in Java. One tentative proposal might be that the design philosophy of Python language emphasizes code readability reducing comments. [9] [17] [19]

These empirical results suggest that the use of agile methodologies and programming practices does not influence the distribution of metrics in the classes. In conclusion, the development methodology does not seem to affect metric distributions. [9]

3.2 An Approach to Improve the Quality of Software Using Metrics and Technical Debt

In the master thesis *An Approach to Improve the Quality of Using Software Metrics and Technical Debt*, the author deals with the difficulty of measuring and monitoring the quality of software solutions. [20] He describes the development of a tool that should be used in a Volvo factory. The tool uses software metrics and the concept of technical debt for monitoring the quality of software solutions. Nine metrics were chosen:

- LOC (*Lines of Code*);
- CC (*Cyclomatic Complexity*);
- NCA (*Number of Non Constant Attributes*);
- NOM (*Number of Methods*);
- CPD (*Comment Percentage of Descriptions*);
- SOS (*Size of Statechart*);
- DIT (*Depth of Inheritance Tree*);
- NOC (*Number of Children*);
- NOD (*Number of Descendants*).

With the exception of the CPD metric, the remaining metrics are used for the calculation of technical debt. The threshold values of

metrics were established in order to discover technical debt. [20] [21]
The author also stresses the importance of data visualization that enables everyone involved to survey the development process. [20]

3.3 The Analysis of Object-Oriented Metrics in C++ Programs

In research *The Analysis of Object-Oriented (OO) Metrics in C++ Programs* the author measures the quality of software solutions designed with the C ++ programming language. [22] One research question was which of the object-oriented metrics correlate with defect density in the research chosen open source systems?

The focus of research is on the use of CK (Chidamber & Kemerer) metrics:

- WMC (*Weighted methods per class*);
- DIT (*Depth of inheritance tree*);
- NOC (*Number of children*);
- CBO (*Coupling between object classes*);
- RFC (*Response for class*);
- LCOM (*Lack of cohesion metric*).

This research describes the application of mentioned design metrics in an analysis of 30 open source systems, and presents the results. The work is mainly focused on systems written in C++, and all of them were collected from Sourceforge.net. In order to get the metrics data, the source code of the systems was analyzed using the metric extraction tool Understand. An initial analysis of the data shows that the distribution is not normal. Therefore, a Spearman correlation analysis was done. The results show that only RFC and NOC are significant in predicting defects. [22]

3.4 Estimating the Complexity of Programs in Python Language

In this scientific paper, the authors formulate a complexity metric for the Python Language. Their case study is evaluated in three different object-oriented languages. They apply their metric on a real project to prove its real applicability and usefulness. [17]

As they discover in a review of existing literature, Python is a programming language that lets programmers work more quickly, and to integrate systems more effectively. In cases of an embedded system, where inexpensive components and maintenance are demanded, Python may provide the best solution. However, there are no proper tools or techniques available to evaluate the quality of Python code. All of the existing tools are effective in evaluating the quality of the Python language only up to an extent. Most of them are confined to compute simple metrics, which give only an idea for some specific attributes, none of which is capable of evaluating the majority of attributes in a single metric. [17]

In the paper, the authors are trying to investigate all the factors, that are responsible for increasing the complexity of code written in Python language. There is a proposal for a new unified metric SMPy (*Software Metric for Python*). The practical applicability of the metric is demonstrated on a case study. [17]

4. PRELIMINARY WORK

Technical debt and software metrics have been calculated for twelve open source projects written in the programming language Python. [23] [11] [10] [8] To measure the values of object-oriented (OO) metrics for Python projects, we used the software tool Understand from the company Sciitools. The open-source platform SonarQube with the plugin Plyint was used to calculate the technical debt of Python projects. [23] [19] [24] The results are presented in Table 1. In Table 2 there are represented values, calculated with the software tool Understand for each object-oriented metric of the Python projects.

For most projects covered by our preliminary survey, it is not possible to get a number of detected errors that occurred during the development of a project. Data about errors could be useful to discover correlations between OO metrics and the levels of technical debt. Instead, it may be tested if it is possible to find a correlation between values of metrics and values of technical debt in different versions of project cycle development.

By comparing the values of technical debt and values of OO metrics in individual versions of the project development history, one could discover a correlation between OO metrics and technical debt.

Table 1: Technical debt of Python projects, calculated with a software tool Sonarqube

Project	Issues	Major issues	SQALE rating	Technical dept	Ratio
AstroBin	**369**	**298**	**A**	**7d 2h**	**0,2%**
AstroML	**475**	**414**	**A**	**5d 1h**	**0,3%**
Blender	**10.413**	**7.960**	**A**	**157d**	**1,3%**
Django	**907**	**726**	**A**	**19d**	**0,6%**
ErpNext	**624**	**612**	**A**	**14d**	**0,8%**
GadFly	**1.530**	**1.396**	**A**	**30d**	**3,3%**
HomeAssistant	**63**	**43**	**A**	**1d 2h**	**0,5%**
Odoo	**6.387**	**5.487**	**A**	**126d**	**1,2%**
PyCharm	**2.275**	**176**	**A**	**7d 6h**	**1,1%**
PyFF	**998**	**792**	**A**	**17d**	**1,0%**
Tryton	**1.589**	**1.589**	**A**	**37d**	**0,9%**
Zope	**23.432**	**21.548**	**A**	**473d**	**2,1%**

Table 2: Calculated values of OO metrics for Python projects

	OO metrics								
Project	Classes	Code lines	DIT	IFANIN	NOC	RFC	NIM	NIV	WMC
AstroBin	463	137.538	0,91145	0,88337	0,14039	2,15551	1,73434	0,26350	1,73434
AstroML	42	23.010	1,28571	1,00000	0,35714	10,38095	4,95238	3,50000	4,95238
Blender	2.632	208.079	1,12310	1,15198	0,97302	3,89096	2,33055	0,97796	2,33055
Django	595	62.940	1,70756	0,98992	2,96807	10,81513	3,25546	0,54118	3,25546
ErpNext	292	41.642	1,37671	1,00342	0,11301	10,24315	4,36986	1,17808	4,36986
GadFly	165	18.770	1,58788	0,67273	0,55758	15,83636	5,76364	2,42424	5,76364
HomeAssistant	39	5.028	1,41026	1,02564	0,28205	6,56410	4,30769	2,74359	4,30769
Odoo	2.075	572.937	1,86361	1,01012	0,48723	46,66651	3,89590	0,80482	3,89590
PyCharm	326	15.131	2,35890	1,00000	0,73620	18,21472	6,38957	0,00000	6,38957
PyFF	528	47.866	1,98864	0,82197	0,77083	10,20076	3,30492	0,98295	3,32576
Tryton	1.040	148.965	0,89231	1,04135	1,18558	4,93558	4,00288	0,54904	4,00288
Zope	6.846	364.306	1,68157	0,98744	1,30178	12,21750	3,41192	1,03418	3,41192

5. FINDINGS AND CONCLUSIONS

Technical debt is an abstract concept for everything that should be done in a project but is not, and which then affects all the outcomes during development, making the project more and more difficult to test, maintain, develop, etc. Calculating the technical debt helps to monitor the curve of the development projects and to influence the slope of the curve itself, while helping everyone involved in the project, especially developers who can oversee their own development throughout the project.

In the articles that we have examined, the authors used different tools, which are mainly used in helping determine the value of metrics. With the tools with which we calculated the technical debt and the values of object-oriented metrics, we have sufficiently covered the need for the calculation of wanted metrics.

Solutions in the programming language Python stand out as the most optimal due to the nature of the programming language, which, because of its simple code (which comments on itself) reduces the complexity of the program code. In the future, it will be necessary to pay attention to the correlation between the value of different metrics, which with its values are pointing to the problems that may arise in the maintenance (in coupling and cohesion of objects).

The conclusions of the preliminary study are as follows:

Is it possible to evaluate a Python solution using "classic" object oriented metrics (as well as for Java, C #) - which may or may not?

The same metrics for Python solutions can be used as with other object-oriented languages.

What tools do other authors use to obtain metric properties?

Among tools that calculate the value of technical debt, the software tool SonarQube is suitable for Python solutions. The software tool NDepend is intended for software solutions in the MS .Net environment (C#, Visual Basic). The calculation of technical debt for projects in different programming languages is possible with the software tool CAST, but not for Python projects. The calculation of OO metric values for Python projects supports the software tool Understand (SciiTools).

Is the concept of technical debt used to evaluate Python projects?

The calculation of technical debt for Python projects is very well integrated into the software tool SonarQube, which offers programmers – together with the plugin Pylint (Python Code Analyzer) - excellent opportunities for project monitoring. It also offers timely actions in the event of increasing technical debt, which SonarQube closely oversees and analyzes in each phase of development.

How to deal with technical debt? There is no simple answer to this question. It can dealt with in a simple manner, by paying it off as soon as possible, or all at once. On the other hand, it can be repaid gradually by selecting urgent or important items of technical debt, which affects the life cycle of software solutions. Managers (CIO), which have been in their positions for a long time, will understand based on their experience, which technical debt needs to be repaid and which can be temporarily, or permanently, allowed.

Our opinion is that it makes sense to continue with research on the appropriateness of existing metrics and application of the concept of technical debt for Python project evaluation and to find the reference values for individual metrics.

6. REFERENCES

[1] J. Walters and K. M. Adams, "Improving Software through Metrics while Providing Cradle to Grave Support," CrossTalk, pp. 26-29, 2014.

[2] D. P. Darcy and C. F. Kemerer, "OO Metrics in Praktice," IEEE Software, pp. 17-19, 2005.

[3] M. Andersson and P. Vestergren, Object-Oriented Design Quality Metrics, Uppsala: Uppsala University, 2004.

[4] L. Cheikhi, R. E. Al-Qutaish, A. Idri and A. Sellami, "Chidamber and Kemerer Object-Oriented Measures: Analysis of their Design from the Metrology Perspective," International Journal of Software Engineering and Its Applications, Vols. Vol. 8, No. 2, pp. 359-373, 2014.

[5] A. Abram, Software Metrics and Software Metrology, New Jersey.: John Wiley & Sons, Inc., 2010.

[6] E. Urbani, "Metriken zur statischen Analyse objektorientirten Source-Codes," pp. 58-66, 1 julij 2004.

[7] N. Fenton and J. Bieman, Software Metrics - A Rigorous and Practical Approach, Third Edition, CRC Press, 2014.

[8] Subramanyam, Jitendra;, "Technical Debt in Business Applications," 2013. [Online]. Available: http://www.castsoftware.com/resources/document/whitepapers/world-quality-report-2011-2012?qad=ES). [Accessed 17 03 2015].

[9] G. Destefanis, S. Counsell, G. Concas and R. Tonelli, Software Metrics in Agile Software: an Empirical Study, Cagliary: Universitiy of Cagliary.

[10] D. J. Stuertevant, Technical Debt in Large Systems: Understanding the Cost of Software Complexity, Massachusetts: Massachusetts Institute of Technology , 2013.

[11] M. Fowler, "Technical Debt," 2003. [Online]. Available: www.martinfowler.com/bliki/TechnicalDebt.html. [Accessed 26 januar 2015].

[12] R. Griffith, Technical debt management in release planning – a decision support framework, Montana: Montana State University, 2014.

[13] E. Lim and N. Taksande, "What Software Practitioners Have to Say about Technical Debt," 2012.

[14] C. Sterling, Managing Software Debt: Building for Inevitable Change, Boston: Addison-Wesley, 2011.

[15] G. v. Rossum, "Wikipedia," Python Software Foundation, Februar 2015. [Online]. Available: http://en.wikipedia.org/wiki/Python_(programming_language). [Accessed 28 april 2015].

[16] T. Gaddis, Starting Out with Python, Third Edition, Essex: Pearson Education Limited, 2015.

[17] F. Cafer and S. Misra, Estimating complexity of Programs in Python Language, Atılım: Atılım University, 2010.

[18] H. Washizaki, R. Humaira, K. Sakamoto, A. Ohashi and Y. Fukazawa, Towards a Unified Source Code Measurement Framework Supporting Multiple Programming Languages, Tokyo: Dept. Coputer Science and Engineering.

[19] S. Misra and F. Cafer, "A software metric for python language," Heidelberg, 2010.

[20] B. Petersson and S. Zhang, An Approach to Improve Quality of Software Using Metrics and Technical Debt, Göteborg: University of Computer Science and Engineering, 2013.

[21] K. A. Ferreira, M. A. Bigonha, R. S. Bigonha, L. F. Mendes and H. C. Almeida, "Identifying thresholds for object-oriented software metrics," The journal of System and Software, vol. 85, pp. 244-257, 2011.

[22] N. Sham and A. Bakar, "The Analysis of Object-Oriented Metrics in C++ Programs," Lecture Notes on Software Engineering, vol. Vol. 4, 2015.

[23] J. L. Letouzey and M. Ilkiewicz, Managing Technical Debt with the SQALE Method, Arcueil, France: IEEE Software Magazine, 2012.

[24] A. G. Campell and P. P. Pappetrou, SonarQube in Action, New York: Manning Publications Co., 2014

Different approaches for measuring XML Schemas

Maja Pušnik
Faculty of Electrical Engineering
and Computer Science,
University of Maribor,
Maribor, Slovenia
maja.pusnik@uni-mb.si

Gordana Rakić
Faculty of Sciences
University of Novi Sad
+381 21 458888
goca@dmi.rs

Zoran Budimac
Faculty of Sciences
University of Novi Sad
+381 21 458888
zjb@dmi.rs

Marjan Heričko
Faculty of Electrical Engineering
and Computer Science,
University of Maribor,
Maribor, Slovenia
marjan.hericko@uni-mb.si

ABSTRACT

In this paper, we will describe different approaches to measuring different aspects of the three level architecture of software development, where XML Schemas play an important role, particularly in the data and the logic layer. We measured the quality aspect of (1) XML Schema as a special XML document, (2) XML Schema objects, (3) classes, generated based on XML Schemas and (4) relational databases, generated based on the mapping of XML Schemas. This paper is preliminary research, and thus a proposal for further development and integration of the four metric groups, focused on the quality of XML Schemas from several aspects as well as the effects an XML Schema might have on higher architecture levels. The overview of metrics will be used for the further development and optimization of metrics, applied to XML Schemas that are nevertheless suitable for other aspects of information systems and software components.

Categories and Subject Descriptors

D.2.8 [**Metrics**]: Complexity measures

General Terms

Measurement, Standardization.

Keywords

Measurement, metrics, XML Schemas.

1. INTRODUCTION

Assuring quality in all aspects of the software development lifecycle is of crucial importance if we want to create a well-accepted product with a positive user experience. There are several factors affecting user experience, acceptance and favourable behaviour [18], intention for future use, and loyalty to a product developer. Some of these factors include efforts in product maintenance and feedback on how the changing/upgrading of software affects overall quality. For this, quality metrics are needed. Since information systems or software products are complex systems with fairly heterogeneous components, we focused on only one component: XML Schemas, being an important indicator of quality in the data interchange in e-business and other fields [1]. XML Schemas and XML technologies in general are present in all aspects of software, however they are also a part of a legacy that needs to be modernized (the first XML Schema were introduced around the year 2000). Although we have focused primarily on XML Schemas and their side products, their characteristics can be applied on general XML technologies, including XML documents, XSLT transformation, XSL-FO and other distinctive XML related documents.

This paper addresses the possible metrics that can be applied to different stages or levels of software development, focusing on XML Schemas. The structure and content of these widely used document definitions has a significant influence on the quality of XML data and data in general, therefore the quality of XML Schemas is also an important challenge for knowledge management. If achieved, a good XML Schema directly and indirectly leads to the higher efficiency of a product, as well as the simplification of information solutions, simplified maintenance and higher quality of data in organizations. This paper focuses on measuring the quality of XML Schemas.

The paper is organized into six sections. After the introduction of the problem area, the goal of this paper is presented: finding the most optimal measuring process. The third section presents four possible metric types, which are integrated and compared in the fourth section. A conclusion and future plans are given in the fifth section.

2. MEASURING PROCESS

The history of (software) metrics begins with metric proposals from the 80s and 90s, when the first software solutions and computers in general were introduced into widespread use. Metrics are already widely used, however there is still a debate over the accuracy of the results, as well as if the results can be generalized. There are still aspects of software that are not properly addressed (the quality of GUI for example, user experience, acceptance etc.). The metrics and their categorization, developed based on research in [1], regarding the three-level architecture are presented in Figure 1.

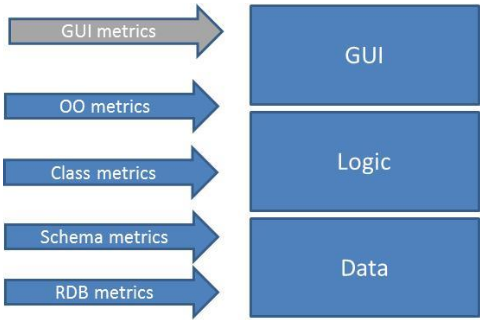


Figure 1 Metrics of the 3 level architecture.

The focus of the paper is on measuring XML documents as building blocks of the entire software architecture. A modern software systems consist of many hundreds or even thousands of interacting entities at different levels of abstraction [2]. Therefore, measuring and monitoring of each entity is important. Knowledge about different aspects of software quality during software evolution can be valuable information for developers and project managers. It helps to reduce the number of defects and improves the internal structure of software [15]. The main goal of this paper is to detect the main difficulties in the application of software metrics in practice and to deal with them through the development of a more useful software metrics tool, as already addressed in [16]. The four main metric types will be introduced in the following section.

3. METRIC TYPES

The primary focus is on measuring the core XML document definition, the XML Schema. However, an XML Schema can be generated based on a relational database, classes are generated based on XML Schema building blocks and the Object Oriented structure is based on predefined XML Schemas. Therefore, all measurements are focused on the features and characteristics of XML Schemas, where the complexity of their structure and other parameters is presumed to have an effect on other levels as well. Compared to the expanded set of characteristics, a comparison was made with other architecture level of XML Schema metrics. The comparison will be used for the further integration of expanded metric sets, and to create a composed metric system evaluating the characteristics of a product at all levels of development (Figure 1) of a database and business logic. GUI metrics and user experience of the product has not yet been included in our preliminary research.

3.1 XML Schema metrics

Based on preliminary research [1], quality aspects of XML Schemas were defined. Each quality aspect was presented through measurable variables within a composite metric and validated based on several representative XML Schemas. 25 variables were measured for a set of 250 standard XML Schemas within different fields (domains) of use. The schemas were attained through available search portals (Google) filtered by standard schemas in 2013. The variables were included within 6 proposed XML Schema oriented metrics: (M1) structure, (M2) clarity, (M3) optimality, (M4) minimality, (M5) reuse, and (M6) flexibility.

Variables were measured for each attained XML Schema, often receiving a non-standard value from 0 to over 1000. For comparing individual variables, a standardization of values was conducted [1]. Figure 2 presents the connection and interpretation of XML Schema quality aspects.

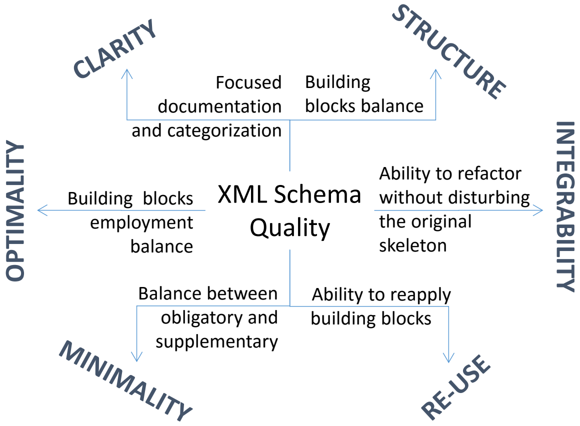

CLARITY
STRUCTURE
Focused documentation and categorization
Building blocks balance
OPTIMALITY
INTEGRABILITY
Building blocks employment balance
XML Schema Quality
Ability to refactor without disturbing the original skeleton
Balance between obligatory and supplementary
Ability to reapply building blocks
MINIMALITY
RE-USE

Figure 2 XML Schema Quality metrics.

3.2 OO metrics

Objects in Object Oriented programming are basically data structures, closely connected with the predefined XML Schema. They are connected to element and attribute types, group types and other XML Schema concepts. Object Oriented representation of the core structural and constraint-related features of XML Schema is presented in [17]. In the following subsections, a summary of different OO metric proposals is presented.

3.2.1 Chidamber and Kemerer's metrics

The authors defined six metrics for OO design [3]:

- Weighted Methods per Class (WMC) - the sum of the complexities of all class methods. It is an indicator of how much effort is required to develop and maintain a particular class.
- Depth of Inheritance Tree (DIT) - Depth of inheritance, also called depth of inheritance tree (DIT), is defined as the maximum length from the node to the root of the tree [7].
- Number of children (NOC) - Inheritance, otherwise called generalization, is one of the fundamental concepts of object models; the number of immediate subclasses (children) subordinated to a class (parent) in the class hierarchy [8].
- Coupling between object classes (CBO) - links between classes define the detailed architecture of the application.
- Response for a Class (RFC) - number of Distinct Methods and Constructors invoked by a Class. The response set of a class is the set of all methods and constructors that can be invoked as a result of a message sent to an object of the class [9].
- Lack of Cohesion in Methods (LCOM) - is a measure for the number of not connected method pairs in a class representing independent parts having no cohesion. It represents the difference between the number of method pairs not having instance variables in common, and the number of method pairs having common instance variables [8].

3.2.2 MOOD metrics

Metrics for Object Oriented Design is defined based on [5] and [6]:

- Method Hiding Factor (MHF) - measures the invisibilities of methods in classes. The invisibility of a method is the percentage of the total classes from which the method is not visible.
- Attribute Hiding Factor (AHF) - measures the invisibilities of attributes in classes. The invisibility of an attribute is the percentage of the total classes from which the attribute is not visible. An attribute is called visible if it can be accessed by another class or object.
- Method Inheritance Factor (MIF) - inherited methods/total methods available in classes.
- Attribute Inheritance Factor (AIF) - inherited attributes/total attributes available in classes.
- Polymorphism Factor (PF) - measures the degree of method overriding in the class inheritance tree. It equals the number of actual method overrides divided by the maximum number of possible method overrides.
- Coupling Factor (CF) - is evaluated as a fraction. The numerator represents the number of non-inheritance couplings.

3.2.3 Other Traditional Complexity Metrics and Models

Other known Object Oriented metrics were also included and will be evaluated and integrated more thoroughly in our future work:

- McCabe's Cyclomatic Complexity (CC) - is the number of linearly independent paths within it.
- Halstead's Software Science,
- Length Equation,
- Quantification of Intelligence Content, and
- Programming Effort

3.3 Class metrics

Since XML Schemas can be directly generated into classes, we included the overview of the class related code metrics [4]. At the class level we look not just at metrics which measure aspects of the class but also look at metrics that give us information on the interaction between classes. Metrics which measure these class interactions tell us far more about the design than about our code, for example how good the 'division of labour' is between methods. Others explain how much a change to a particular class will affect code in another class. Changes to one class should have minimal effects on other classes, and the number of other classes affected should be minimal. Where classes do have a high level of dependency on one another they should be in the same package.

At a basic level, we are interested in metrics accumulated from the method-related metrics e.g. the numbers of Methods and Statements in the class. We are also interested in the Total, Average (per method) and Maximum Cyclomatic Complexity and the Total Halstead Effort. Maintainability Index is important as well. The following metrics are outlined based in [10] and [4]:

- Lack of cohesion of methods (LCOM) - number of not connected method pairs in a class representing independent parts having no cohesion. It represents the difference between the number of method pairs not having instance variables in common, and the number of method pairs having common instance variables.
- Weighted Class Size (WCS) - the number of methods plus the number of attributes of a class.
- Coupling metrics: Response For Class (RFC) that measures the complexity of the class in terms of method calls, Message Passing Coupling (MPC) that measures the number of messages passing among objects of the class, Coupling Between objects (CBO), the total of the number of classes that a class referenced plus the number of classes that referenced the class, Fan Out or Efferent Coupling (Ce), the number of other classes referenced by a class, and Fan In or Afferent Coupling (Ca), the number of other classes that reference a class.
- Reuse ratio - number of super classes above this class in the class hierarchy divided by the total number of classes in the class hierarchy.
- Specialization Ratio - Specialization ratios measure the extent to which observations contained within a category are representative of the population of those observations as a whole [14] and is calculated as the number of subclasses below this class in the class hierarchy and number of super classes above this class in the class hierarchy.

Additionally, several parameters are calculated, providing information about the level and complexity of interaction between the class and other classes (both in the classes' hierarchy and external to it): the Number of External Methods Called, Number of Methods Called in the class hierarchy and the Number of local methods called. These figures are used in calculating metrics such as RFC, LCOM, MPC, Fan In and Fan Out. The number of instance variables, number of modifiers, number of interfaces implemented and number of packages imported give additional information about the class's level of semantic complexity. As with methods, large values for these can suggest that a class is doing too much.

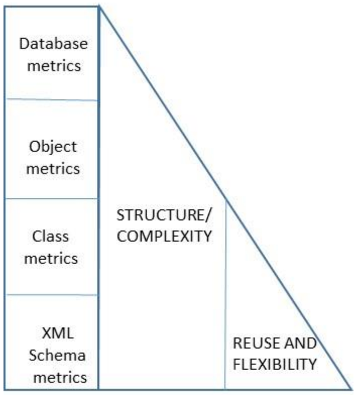
3.4 Relational database metrics

XML Schemas can be mapped between each other and directly to a relational database. Considering the main characteristics of a relational table, the following metrics are proposed in [12] and [13]:

- Number of attributes (NA) - the number of attributes in all the tables of the schema.
- Depth Referential Tree (DRT) - the length of the longest referential path in the database schema. Cycles are only considered once.
- Number of Foreign Keys (NFK) - defined as the number of foreign keys in the schema.
- Cohesion of the schema (COS) - defined as the sum of the square of the number of tables in each unrelated sub-graph of the database schemata.
- Referential degree (RD) of a table [11].
- Schema Size (SS).
- Complexity of references between tables (DRT, NFK).

4. COMPARING/INTEGRATING

We compared and evaluated positive and negative sides to all metric approaches, trying to compose a holistic approach, addressing all aspects of XML technologies and a variety of versions in which XML is transformed. Table 1 presents all metrics, categorized into similar groups. Figure 3 presents the three major quality aspects that appeared in all metric groups: complexity of structure, reuse and flexibility.



. Figure 3 Integrated metrics.

The focus of this paper is to prepare an initial integration of several dispersed metrics. Connecting all proposed and presented metrics into a combined and extended quality index as well as extending the metric group is the following step. Nevertheless, Figure 3 and Table 1 present a preliminary classification of the multilevel metric groups.

Table 1. Integrated metrics

M0 XML Schema Quality metrics	M1 OO metrics	M2 Class metrics	M3 DB metrics
Structure	Complexity Metrics and Models Traditional Metrics	Un-weighted Class Size Number of External Methods Called, Number of Methods Called in class hierarchy Number of local methods called Number of instance variables Number of modifiers Number of interfaces implemented Number of packages imported	NA DRT NFK COS SS Complexity of references between tables
Reuse		Reuse ratio	
Flexibility		Coupling Lack of cohesion of methods	

5. CONCLUSION

There are several metrics and measuring approaches to measure software quality, focused on different architecture levels. Our primary motivation was to build an extensive set of metrics, applying all aspects of XML Schema quality, addressing both structure and content. The results in this paper open the possibilities for further scientific and applicative research in the field of XML Schema quality as in the field of XML data knowledge management. Based on our research, 30% of XML Schemas are not properly built and need adaptation mostly in the fields of structure and transparency, indicating that other aspects of software development are insufficient in quality as well.

Future work will include an in-depth analysis of existing metrics, the identification and evaluation of all measured parameters and the search for correlations among them. The goal is to provide a holistic approach to evaluate the quality of software as well as information systems as well as a tool for early identification of errors or entities with low quality. We will try to create a combined, optimal, integrated and structured metric, combining and extending existing knowledge of quality metrics.

6. REFERENCES

[1] Pušnik, M., Heričko, M., Budimac, Z. and Šumak, B. 2014. XML Schema metrics for quality evaluation. *Computer Science and Information Systems*. Vol. 11, No. 4, 1271–1289. DOI= 10.2298/CSIS140815077P.

[2] Savić, M., Rakić, G., Budimac, Z., and Ivanović, M. 2014. A language-independent approach to the extraction of dependencies between source code entities. *Information and Software Technology*. Volume 56, Issue 10, October 2014, Pages 1268–1288. doi:10.1016/j.infsof.2014.04.011

[3] Jamali, S.M. 2006. *Object Oriented Metrics, http://www.cs.sfu.ca/~sja25/personal/resources/papers/ObjectOrientedMetrics.pdf.*

[4] Virtual Machinery, *Object-Oriented Software Metrics - Class Level Metrics*, Technical report accessed on 1.9.2015 at http://www.virtualmachinery.com.

[5] An Introduction to Object-Oriented Metrics, *Object oriented metrics, Software engineering Materials,* Technical report accessed on 1.9.2015 at http://agile.csc.ncsu.edu.

[6] MOOD and MOOD2 metrics, Technical report accessed on 1.9.2015 at http://www.aivosto.com/.

[7] Naboulsi, Z. *The Ultimate Visual Studio Tips and Tricks Blog* accessed on 1.9.2015 at http://blogs.msdn.com/b/zainnab.

[8] Lincke, R. and Löwe, 2007 W. *Compendium of Software Quality Standards and Metrics*, accessed on 1.9.2015 at http://www.arisa.se/compendium.

[9] Brooks, A. Technical report accessed on 1.9.2015 at http://staff.unak.is/andy/StaticAnalysis0809/metrics/rfc.html.

[10] Kumari, D. and Rajnish, K. 2015. Investigating the Effect of Object-oriented Metrics on Fault Proneness Using Empirical Analysis. *International Journal of Software Engineering and Its Applications*, Vol. 9, No. 2 (2015), pp. 171-188.

[11] Piattini, M., Calero, C. and Genero, M. 2001. Table Oriented Metrics for Relational Databases. *Software Quality Journal*. June 2001, Volume 9, Issue 2, pp 79-97.

[12] Calero, C., Piattini, M and Genero, M., 2001. Database Complexity Metrics, *Quality of Information and Communications Technology*. Lisboa, Portugal, March 12-14, 2001.

[13] Jamil, B. and Batool, A. 2010. SMARtS: Software Metric Analyzer for Relational Database Systems, January 2010. DOI: 10.1109/ICIET.2010.5625716.

[14] United Nations Glossary of Classification Terms. Prepared by the Expert Group on International Economic and Social Classifications and ISIC Rev. 3, para. 155.

[15] Gerlec, Č., Rakić, G., Budimac, Z. and Heričko, M. 2012. A Programming Language Independent Framework for Metrics-based Software Evolution and Analysis. *Computer Science and Information Systems 9 (03), 1155-1186*.

[16] Budimac, Z., Rakić, G., Heričko, M. and Gerlec, Č. 2012. Towards the Better Software Metrics Tool. *Software Maintenance and Reengineering (CSMR)*. 2012 16th European Conference, pages 491-494.

[17] Algič, S., Bernstein, P.A. and Jairath, R. Object-Oriented Constrains for XML Schema. *Objects and Databases*. Volume 6348 of the series Lecture Notes in Computer Science pp 100-117.

[18] International Organization for Standardization (2009). Ergonomics of human system interaction - Part 210: Human-cantered design for interactive systems (formerly known as 13407). ISO F±DIS 9241-210:2009

Survey on Software Testing for Big Data Applications

Sašo Karakatič
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
saso.karakatic@um.si

Luka Pavlič
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
luka.pavlic@um.si

Gurusamy Arumugam
Department of Computer Science
Madurai Kamaraj University
Madurai, India
gurusamyarumugam@gmail.com

Muthuraman Thangaraj
Department of Computer Science
Madurai Kamaraj University
Madurai, India
thangarajmku@yahoo.com

Marjan Heričko
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
marjan.hericko@um.si

ABSTRACT

In this paper we researched the topic of software testing on applications that utilize big data. The term big data is becoming more popular with the increasing usage of large, high varied data sources, but so far there is limited knowledge on testing these kinds of the applications.

This paper is a preliminary study in which we conducted an initial systematic literature review and present the results gathered from this search. The initial results were filtered to limit the scope to relevant papers only. We also conducted a basic overview of the type of research, the source of the research paper and the time distribution of the published work.

The results of this literature review suggest that this is a promising research topic, as the number of published papers has been steadily rising since 2013, which coincides with the rise in popularity of the term big data and the usage of big data sources.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.5 [**Testing and Debugging**]: Testing tools, Code inspections; D.2.8 [**Software Engineering**]: Metrics—complexity measures, performance measures

General Terms

Measurement, Performance, Reliability, Theory

Keywords

Big data, software testing, performance testing

1. INTRODUCTION

Over the last decade, data has increased in many aspects including its volume, the speed with which it is generated and so on. A new term, *big data*, was coined to capture the specifics of the data that has to be processed and analyzed. There is no agreed definition of the term. However, one of the mostly used definitions, written by Gartner [8] includes three *V*-s. Gartner defines big data by *high volume*, *high velocity* and *high variety*. If one has to deal with impractically high volumes of the data, in short (almost real) time, and the data is represented in numerous formats, then one is dealing with big data. In this type of situation, there is no opportunity for a ”store and analyze” approach, which is used in traditional business intelligence, but instead new approaches are needed in order to analyze the data.

Since big data approaches and applications have recently entered the mainstream [4] and are not limited to a controlled laboratory environment anymore, fundamental development issues have been raised. One of those is definitely the possibility of performing software testing activities over big data applications. That includes unit testing, black- or white-box testing, acceptance testing, performance testing and others. It seems almost impossible to employ traditional software testing activities and approaches for big data applications as well. The data is constantly flowing, and the volume is one such a scale that one does not even have the possibility of copying or backing up the data. These and other issues need to be addressed if big data applications are to become mainstream from a software engineering point of view. The aim of this paper is to review state of the art developments in current software testing approaches that are designed for big data applications. The initial systematic literature review in the field will be presented in detail. However, in this paper, we will not present a new or improved software testing approach, but remain limited to reviewing the state of the subject.

Big data applications, as with every other bit of software, need to be delivered with quality, on time and within a budget. But given the nature of big data, is it reasonable to expect established engineering approaches to work at the moment? One of the key aspects in software engineering is quality assurance, which is usually narrowed down to testing activities. According to Galop, there are numerous challenges that have to be tackled when developing big data applications [6, 7] with regard to the automation and virtualization of processes, and understanding and properly managing data with a lack of expert knowledge.

The rest of this paper is structured as follows: the next section describes the research method for a systematic literature review in detail. Literature sources, the methods of their retrieval, acceptance and rejection criteria are described. The core of this paper is captured in section 3, where the results of the systematic literature review are presented. We conclude with an interpretation of the results and summarize current state of the art developments in the area.

2. RESEARCH METHOD

In this section we present the methods of the preliminary systematic literature review with regard to software testing on applications with big data. The goal of this research is to determine if the research topic is worth investigating and what direction the research body heading. In order to develop a complete understanding of the problem and solutions in the field of software testing for big data applications we were trying to answer several research questions. The questions addressed during this paper are:

- What are the general problems regarding software testing in the field of big data applications?
- Are there sufficient solutions to testing issues in the field of big data testing?
- Which big data testing field is addressed mostly in the literature?
- What are the best practices regarding software testing for big data applications?

2.1 Data sources

Directly used literature sources are academic databases. We have chosen the following databases, which are considered as mainstream venues for global research on software development:

- ScienceDirect (*http://www.sciencedirect.com*)
- ACM Digital library (*http://dl.acm.org*)
- IEEE Xplore (*http://ieeexplore.ieee.org*)
- SpringerLink (*http://link.springer.com*)
- Scopus (*http://www.scopus.com*)

2.2 Data retrieval

In order to perform a systematic literature review of the presented field, a common set of keywords was used. The syntax of search string was adapted to a particular database. Search strings were (1) *software AND testing AND ”big data”*, and (2) *application AND testing AND ”big data”*.

Normally, for software testing formal methods, one would search in the area of *”verification and validation.”* However, in our case, after performing preliminary searches it was revealed that the term *”testing”* is more appropriate, since *”verification”* or *”validation”* terms resulted in a very different research scope (such as hypothesis validation).

Since preliminary searching resulted in the majority of results having the *”testing”* term related to other concepts, instead of software testing (testing hypothesis, testing environment, training and testing dataset in machine learning) we addressed two specific testing areas as well: *performance testing* and *acceptance testing*. This is why we added two more search strings: (3) *software AND (”performance testing” OR ”acceptance testing”) AND ”big data”*, and (4) *application AND (”performance testing” OR ”acceptance testing”) AND ”big data”*.

2.3 Studies selection

Primary studies were included according to the following criteria:

- The paper should be written in the English language,
- The paper should be available online (including free studies and studies, available through our individual subscriptions with IEEE, ACM and Springer),
- The paper should be published between 2010 and 2015,
- The paper should discuss challenges and solutions in the testing of big data applications - whether it be in general or within a special case.

Additionally, if studies conformed to the following criteria, they were excluded from our body of research papers:

- were duplicate or repeated studies or
- were not directly related to the objective of the research.

From all studies found in this initial search process, a primary selection was performed based on the summaries of the studies. A final selection was performed via paper review. No additional selection process was made – the review of the papers in detail were left for future research on the topic.

After the initial search, we conducted a review of the papers gathered by a search based on the titles. We were left with 22 relevant papers. These papers were studied in detail and constituted the result of this detailed study.

Table 1: Results of search string employment.

Search string	ScienceDirect	ACM	IEEE	Scopus	SpringerLink
software AND testing AND ”big data”	1997	1438	136	45	3326
application AND testing AND ”big data”	3033	1910	252	94	4874
software AND (”performance testing” OR ”acceptance testing”) AND ”big data”	71	53	1377	2	124
application AND (”performance testing” OR ”acceptance testing”) AND ”big data”	84	57	2508	7	146

3. Review of the Results

In this section we will present the results of the study search and retrieval based on the presented search criteria and search strings. First, we present the search string and a summary of the initial search and then we make present the filtered results based on the criteria presented in the previous section. The data retrieval was performed in late August 2015 and the search results are summarized in Table 1.

3.1 Results by topic

The vast amount of research found in the data retrieval process is naturally too much to handle and review entirely, so we filtered the results based on the title and abstract, in order to narrow it down to relevant research. After filtering the results, we gathered 20 papers and 2 books. Books [19, 12] and a survey [4] addressed the general topics regarding this systematic literature review. Twelve papers focused on the theoretical and practical aspects of performance testing (expected response time, memory consumption etc.), while one of them addressed security issues [3]. We examined two case studies and some papers that described testing issues in big data applications in general or merely mention research opportunities including the testing of big data applications [17, 1, 15, 16, 18, 2].

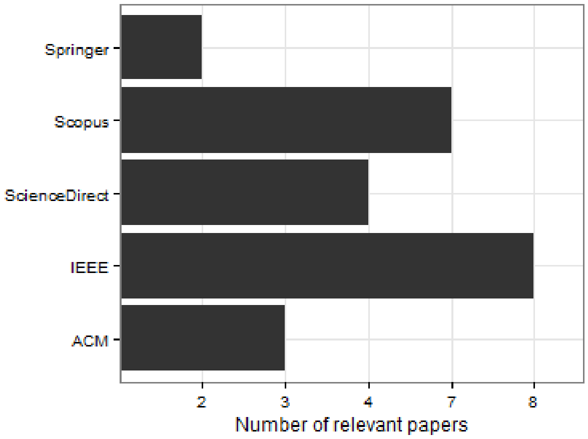


Figure 1. Comparison of sources where relevant papers were found.

In general, the papers' conclusions included the clear observation that traditional software testing activities are not possible across a big data area. Verifying the results of massive data stream processing is impractical for humans, if not impossible – this is the reason why big data software exists in the first place [2, 5, 13, 14, 21]. Many papers focused on the performance issue of these kinds of applications [9, 10, 11, 20, 23, 22].

Basic statistics about the relevant papers are shown in Figure 1, 2 and 3, where we present the research search engine, the year of publishing and type of paper, respectively. Figure 1 shows that most papers were found in research search engines by IEEE (8 papers), followed closely by Scopus (7 papers). Note that two of the papers were found on both the ScienceDirect and Scopus search sites.

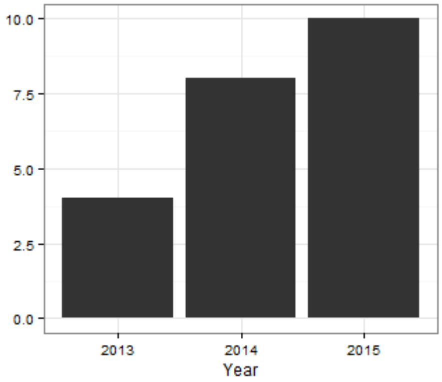


Figure 2. Year of the publishing of relevant papers.

In the chart in Figure 2 it can be seen that the topic of software testing is gradually emerging. Before the year 2013, there were no papers on the topic published, while in 2013 there were only four papers on the topic. In recent years (2014 and 2015) we can observe a trend where papers on the topic were published; there are even two more in 2015 than in the year 2014. These findings proposed that there are no relevant papers outside of our search criteria, where we limited the search only to papers publicized in the years 2010 and 2015.

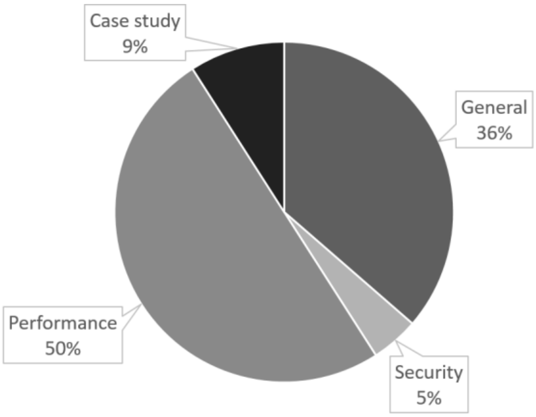


Figure 3. Types of research found.

Next, Figure 3 depicts the proportions of topics of research conducted in relevant papers. As the figure shows, the most frequent topic of software testing in applications with the use of big data is performance. This is natural as the biggest difference in comparison to regular applications is the large quantity of data, which can hinder performance significantly. Second, the most frequent topic in relevant papers is about testing in general, which applies to all subjects on software testing. “Case study” and “security testing” were the least-frequent topics found in relevant papers.

4. CONCLUSIONS

In this paper, we presented a brief overview of the research done on the topic of software testing of applications that use big data. We made a preliminary systematic literature review, which will also be used as a basis for future studies. A literature review of the topic reveals that the subject is relatively new, since the oldest paper on the subject is from the year 2013. This is not so surprising when we take into account that the term ”big data” is also new and was not commonly used in the research community before that time.

A detailed review of the relevant papers found that software testing, when big data is involved, is indeed different than testing on ”regular” applications or applications that do not utilize big data. Some of the papers presented use cases on how the researchers tackled the problem of testing, others provided a useful overview of techniques from regular software testing methods that could have theoretically been used when big data is applied, and some presented frameworks and suggestions on how to test performances for this kind of software. As applications utilizing big data become more frequent, the topic of testing these kind of applications becomes more important for the scientific community.

The scarcity of research and the topics of available research papers indicate that this is a promising filed of research and should provide the basis for future research topics. Also, the trend of published papers indicates that more papers are being published every year.

5. REFERENCES

[1] P. Amirian, A. Basiri, and A. Winstanley. Evaluation of data management systems for geospatial big data. In *Computational Science and Its Applications–ICCSA 2014*, pages 678–690. Springer, 2014.

[2] I. Chana and P. Chawla. Testing perspectives for cloud-based applications. In *Software Engineering Frameworks for the Cloud Computing Paradigm*, pages 145–164. Springer, 2013.

[3] G. Chen, S. Wu, and Y. Wang. The Evolvement of Big Data Systems: From the Perspective of an Information Security Application. *Big Data Research*, 2(2):65–73, June 2015.

[4] M. Chen, S. Mao, and Y. Liu. Big data: A survey. *Mobile Networks and Applications*, 19(2):171–209, 2014.

[5] R. DeLine. Research Opportunities for the Big Data Era of Software Engineering. In *2015 IEEE/ACM 1st International Workshop on Big Data Software Engineering*, pages 26–29. IEEE, May 2015.

[6] Galop. 2 major challenges of big data testing, Sept. 2015.

[7] Galop. 5 big data testing challenges you should know about, Sept. 2015.

[8] Gartner. What is big data? Gartner it glossary, Sept. 2015.

[9] I. Hartung and B. Goldschmidt. Performance analysis of Windows Azure data storage options. In *Large-Scale Scientific Computing*, pages 499–506. Springer, 2014.

[10] N. Li, A. Escalona, Y. Guo, and J. Offutt. A scalable big data test framework. In *Software Testing, Verification and Validation (ICST), 2015 IEEE 8th International Conference on*, pages 1–2. IEEE, 2015.

[11] Z. Liu. Research of performance test technology for big data applications. In *2014 IEEE International Conference on Information and Automation (ICIA)*, pages 53–58. IEEE, July 2014.

[12] W. McKnight. Information Management: Strategies for Gaining a Competitive Advantage with Data. Nov. 2013.

[13] S. Nachiyappan and S. Justus. Getting ready for BigData testing: A practitioner’s perception. In *Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on*, pages 1–5. IEEE, 2013.

[14] S. Nachiyappan and S. Justus. Cloud Testing Tools and its Challenges: A Comparative Study. *Procedia Computer Science*, 50:482–489, 2015.

[15] R. Nambiar. A standard for benchmarking big data systems. In *Big Data (Big Data), 2014 IEEE International Conference on*, pages 18–20. IEEE, 2014.

[16] S. Nativi, P. Mazzetti, M. Santoro, F. Papeschi, M. Craglia, and O. Ochiai. Big Data challenges in building the Global Earth Observation System of Systems. *Environmental Modelling & Software*, 68:1–26, June 2015.

[17] C. E. Otero and A. Peter. Research directions for engineering big data analytics software. *Intelligent Systems, IEEE*, 30(1):13–19, 2015.

[18] A. Rajasekar, R. Moore, S. Huang, and Y. Xin. Big Data Operations: Basis for Benchmarking a Data Grid. In *Advancing Big Data Benchmarks*, pages 123–137. Springer, 2014.

[19] A. Reeve. *Managing Data in Motion: Data Integration Best Practice Techniques and Technologies*. Newnes, 2013.

[20] S. Sioutas, E. Sakkopoulos, A. Panaretos, D. Tsoumakos, P. Gerolymatos, G. Tzimas, and Y. Manolopoulos. D-P2P-Sim+: A novel distributed framework for P2P protocols performance testing. *Journal of Systems and Software*, 100:211–233, Feb. 2015.

[21] H. M. Sneed and K. Erdoes. Testing big data (Assuring the quality of large databases). In *2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 1–6. IEEE, Apr. 2015.

[22] T. Vanhove, G. V. Seghbroeck, T. Wauters, F. D. Turck, B. Vermeulen, and P. Demeester. Tengu: An Experimentation Platform for Big Data Applications. In *2015 IEEE 35th International Conference on Distributed Computing Systems Workshops*, pages 42–47. IEEE, June 2015.

[23] M. Yesudas, G. M. S, and S. K. Nair. High-Volume Performance Test Framework using Big Data. In *Proceedings of the 4th International Workshop on Large-Scale Testing - LT ’15*, pages 13–16, New York, New York, USA, Feb. 2015. ACM Press.

Applying Mobile Services Guidelines for User Acceptance To Hybrid Mobile Application Development

Boris Ovčjak
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
boris.ovcjak@um.si

Gregor Jošt
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
gregor.jost@um.si

Gregor Polančič
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
gregor.polancic@um.si

ABSTRACT

In this paper we investigated the appliance of mobile service guidelines for user acceptance to hybrid mobile application development. This is reasonable since hybrid mobile applications are gaining on popularity because of their “develop once, run everywhere” approach. To this end we implemented a simple hybrid entertainment application. The results show that while hybrid mobile applications are not as responsive as their native counterparts, we could still apply majority of the factors that affect acceptance of mobile services.

Categories and Subject Descriptors

H.1.2 [**Information Systems**]: Models and principles – Human factors
D.3.3 [**Programming Languages**]: Language Constructs and Features

General Terms

Measurement, Performance, Design, Reliability, Experimentation, Human Factors, Theory

Keywords

Mobile services, User acceptance, Hybrid mobile application development.

1 INTRODUCTION

Mobile technologies already represent a major part of our daily life. The statistics show, that in 2014, worldwide traditional PC, tablet, ultramobile and mobile phone shipments grew by 4.2 percent, nearing the number 2.4 billion devices. It is also estimated that the number will reach 2.6 billion at the end of the year 2015. Out of that, nearly 2 billion devices are represented by mobile phones [5]. The advancement of mobile technologies, especially the introduction of fast affordable mobile data connections (3G and 4G), boosted the growth of mobile data traffic, reaching the growth of 59% in the year 2015 [4]. These affordable data connections represent the opportunity for the development of mobile solutions (applications) that use resources available through internet connections and therefore reach out of the confines of mobile device itself. The resulting solutions are represented by mobile applications, which deliver various mobile data services to their users, enhancing their flexibility, mobility and efficiency within business and everyday life domains. Such mobile data services are defined differently by researchers. E.g. Hong, Thong, Moon, and Tam (2008) define mobile data services as an assortment of data communication services that can be accessed using a mobile phone over a wide geographic area via mobile telephone networks [7]. On the other hand, Lu, Liu, Yu, and Wang (2008) refer to mobile services as all types of digital services via wireless networks, accessible through any type of mobile device [11]. They provide wireless access to the digitalized contents of the internet via mobile device [10]. Therefore, by using mobile terminal equipment, consumers may conduct a vast area of activity comprised of transactions of services, goods and information with a monetary value via wireless network [17]. As there exists a variety of different mobile services, the researchers developed different categories that divide mobile services. The most commonly used categories by researchers are: (1) Communication Services (e.g. e-mail, SMS, MMS, etc.), (2) Information Services (e.g. weather information, headlines, maps, traffic information, etc.), (3) Entertainment Services (e.g. mobile games, music, TV, ringtones, etc.) and (4) Transactional Services (e.g. making purchases, reservations, banking transactions, etc.) [1][6][19].

Consequentially, the extensive usage of mobile technologies and Wi-Fi-enabled portable devices has also convinced businesses and governments to prepare for transition from electronic to mobile services [16]. However, it is important to stress that the popularity of mobile applications varies between users, which encouraged the mobile data services acceptance research. The acceptance of technology innovations is important for purchasing and use of new products and the same stands for mobile services. There are many studies and researches that are trying to find factors that can influence the acceptance of mobile services with the aim of bringing the best possible experience to users. These researches are often relying on the acceptance models, mainly the Technology Acceptance Model (TAM), Theory of Planned Behavior (TPB), Theory of Reasoned Action (TRA), Unified Theory of Acceptance and Use of Technology (UTAUT) [13].

Besides, if we want to develop a mobile application that would address as many users as possible, we need to take into the consideration many different mobile platforms, e.g. Android, iOS and Windows Phone. However, developing a single application for a wide variety of mobile platforms can be both costly and time consuming. To this end, alternative approaches to the native mobile development have been introduced, which allow developers to implement business logic once and deploy it on several different mobile platforms. Such applications are also known as hybrid mobile application and support the “write once, run anywhere” approach.

To summarize, we can address the issue of multiple mobile platforms with hybrid mobile applications, which affect both developers of mobile applications as well as the end-users. However, the user acceptance still remains a challenge. To this end, we will examine in this article the mobile service acceptance factors for the chosen category as well as analyze how and to what extend can we apply them to hybrid mobile application development. We will achieve this by prototype implementation of a hybrid mobile application.

2 RESEARCH BACKGROUND

In light of research background we will firstly address the mobile data services acceptance, followed by the proposed model for entertainment category and the definition of hybrid mobile application development.

2.1 Mobile Data Services Acceptance and Guidelines

The research of mobile services acceptance is extensive. Summarized data gathered from literature shows that regarding different categories, transactional services are the most investigated (51%), following by information services (22%), entertainment services (9%) and communication services (8%) [13]. In our previous research, we analysed the existing literature with the goal of developing a proposed acceptance model for each of the aforementioned mobile service categories. The proposed models were constructed using only factors that in all cases showed significant influence on the acceptance of an individual mobile service.

The proposed models represent the basis for researchers as a standing point from which to advance the research and also for developers of mobile solutions by providing them the key factors to consider when developing new mobile apps. This study is intended mostly for the developers, by providing a case example of applying factor based guidelines on the development of a mobile application using mobile data services.

Our previous study presented with four proposed acceptance models for each of the individual mobile service categories. To achieve our goal we have chosen to implement a simple mobile game application by considering the factors that impact the acceptance of entertainment mobile services based on our proposed model (Figure 1).

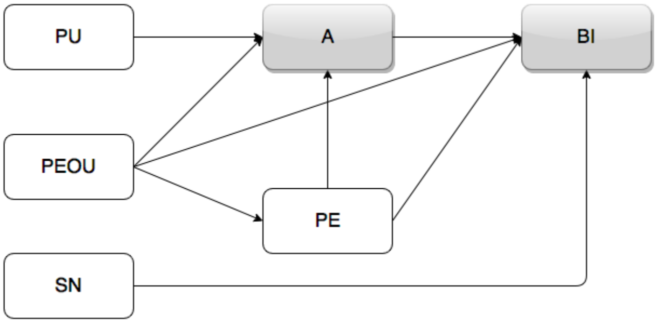


Figure 1: Proposed model for entertainment category

The proposed model consists of six factors and seven causal relations. The majority of the model consists of the factors from the original Technology Acceptance Model (Perceived usefulness (PU), Perceived ease of use (PEOU), Attitude (A), Behavioral intention (BI)). In addition to constructs specific to the original TAM, the generic model also include two additional factors: Subjective norm (SN) from TAM2 and Perceived enjoyment (PE).

The proposed model for the acceptance of mobile entertainment services shows that user acceptance is dependent on the following four latent variables (factors):

- PU - The degree to which a person believes that using mobile services would enhance his or her job performance [2]
- PEOU - The degree to which a person believes that using a system would be free of effort [2]
- SN - The person's perception that most people who are important to him think he should or should not perform the behavior in question [2]
- PE - The extent to which the activity of using a particular system is perceived to be enjoyable in its own right, aside from any performance consequences resulting from system use [2].

To devise guidelines for the development of a prototype mobile application, we researched the most common user complaints about mobile apps and statistics about their usage. These were applied to factors analysed in our study. The resulted proposed guidelines are listed in the Table 1.

Table 1: Proposed guidelines for hybrid mobile services development

Factor	**User complaints [9]**	**Rationale [3]**	**Proposed guidelines**
Perceived usefulness	-App crashing -Compatibility issues -Lack of features -Disliked features -Network problems or slow response time -Unresponsive app	75% users think the app should do exactly what they want it to do.	Developers should: -try to minimize the chance for an app crash -ensure the compatibility between different mobile platforms -include all expected functionalities -ensure fast responsive apps especially in relation to network connections
Perceived ease of use	-Features not working like expected -Complaints about design, controls or visuals	74% user think the app should be easy to use 57% think the app should be well designed	-provide a simple intuitive design -the functionality of features based on platform standards
Social Norm		66% of users downloaded the app based on a review or recommendations 57% of users recommended an app based on their positive experience	-provide a way for app to interact with social media with the goal of app promotion and sharing
Perceived enjoyment	Uninteresting content		-provide user exciting content

These proposed guidelines were in the next stage used to implement a hybrid mobile solution to validate the applicability of these guidelines in the hybrid mobile application development.

2.2 Hybrid Mobile Application Development

As already stated in the introduction, mobile applications can be developed in three distinctive ways, namely: (1) platform specific mobile application, (2) mobile web application and (3) hybrid mobile application.

Platform specific development is the most common way to develop a mobile application. Such applications are designed for a specific mobile platform, making the migration to other platforms challenging and sometimes impossible. On the other hand, mobile web applications, designed for mobile devices, are executed in a browser, so we need only an active web connection to run such applications. Hybrid mobile applications combine the strengths of both approaches. Generally, we integrate mobile web application within a platform specific application. The result behaves like a mobile web application but with all the advantages of platform specific application, e.g. supporting the offline mode, access to advanced, platform specific features and access to all mobile sensors. The extent of such support is dependent on the framework that supports the development of the hybrid mobile applications. While the main business logic of any hybrid mobile applications is written once, we still need to embed the final application in the native mobile application, so that we can deploy it to each different platform [12].

3 APPLYING MOBILE SERVICES GUIDELINES FOR USER ACCEPTANCE TO HYBRID MOBILE APPLICATION DEVELOPMENT

In this chapter we will firstly define which hybrid framework we selected for the implementation. Furthermore, we will explain the development, based on user expectations. Finally, we will provide the results of the implementation in light of the applying the factors to the prototype.

3.1 Selection Of A Hybrid Framework

For the implementation of our prototype application we selected one of the most popular and leading frameworks for hybrid mobile application development, PohneGap [18]. PhoneGap was developed in 2008 by Nitobi and was acquired by Adobe System four years later. The most prominent PhoneGap investors were IBM, RIM and Microsoft [15].

PhoneGap enables the development of hybrid applications with the usage of common web technologies, such as HTML5, CSS3 and JavaScript.

Besides the most common web technologies, PhoneGap enables the access to mobile sensors as well. This is achieved by using a WebView (an element that is used by native applications to display local HTML files and is supported by all major mobile platforms). The PhoneGap libraries therefore enable the usage of native APIs, where we have the access to mobile sensors. All mobile sensors are supported for the leading mobile platforms (iOS, Android and Windows Phone) [14].

3.2 Application Development Based On User Expectations

Mobile application, developed for this study, is a simple mobile game prototype, based on a known letter based game - The hangman (Figure 2). As already stated, the goal was to take into account the proposed guidelines, therefore the aim was to keep the application functional, easy to use, exciting and connected with social media. The prototype enables users to play the game, view the results, share the results and recommend the application to others.

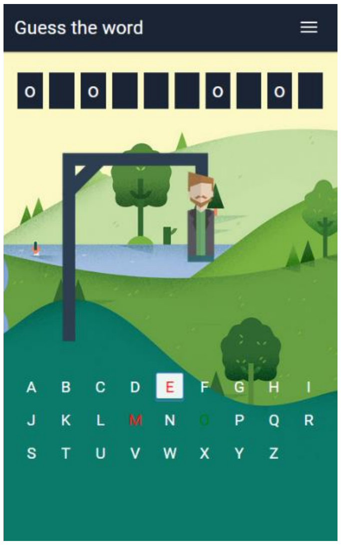


Figure 2: Prototype application

The backend of the prototype mobile application was developed entirely by using Node.js, a server-side JavaScript. Node.js is a software platform that is built on Chrome's V8 JavaScript runtime and uses event-driven non-blocking I/O model. The platform consists of three layers, namely the base layer (contains all the core components), middle layer (acts as a middleware and establishes communication from lower to top layer) and final top layer (consists of JavaScript API) [8]. Node.js exposed REST services that our mobile application consumed, i.e. authentication and all the business logic.

The front end of the application was developed by using PhoneGap and the corresponding technologies, namely HTML5, CSS3 and JavaScript, along with Bootstrap. The latter is a framework composed from the three before mentioned web technologies intended to develop responsive mobile first projects on the web.

3.3 Findings

The implementation of the prototype represents an example of considering mobile data services acceptance factors in the mobile application development. As already stated, our prototype application tried to satisfy four acceptance factors: (1) Perceived ease of use, (2) Perceived usefulness, (3) Social norms and (4) Perceived enjoyment. The results of applying these factors to an application can be observed in the Table 2.

Table 2: Implementation results

Factor	Result	Success
Perceived usefulness	The hybrid development enables the use of the application on all major mobile platforms. The application includes all expected functionalities. The app is responsive, although while new mobile devices show faster execution time old devices are significantly slower when running the app. In addition, the responsiveness of a hybrid application is not as elegant as of a native app.	Partial
Perceived ease of use	The application uses simple intuitive design, customized for a mobile game.	Yes
Subjective norm	The application includes functionalities intended to support promotion and sharing.	Yes
Perceived enjoyment	The factor is by definition subjective in nature, therefore the developer does not have a direct influence on the user's perceived enjoyment.	No

The results from table 2 show that most of the factors can be applied to the development of a hybrid mobile application, but to a different extent. While the developer of a hybrid application can to a certain point achieve the perceived ease of use and subjective norm, the perceived usefulness can be only partially achieved due to the restraints of a hybrid development. On the other hand, the perceived enjoyment cannot be implemented as it is subjective for every user and is not based on the quality of the application implementation.

4 CONCLUSION

The article probed the question of whether we can apply mobile services guidelines for user acceptance to hybrid mobile application development. We investigated the applicability of mobile service guidelines for user acceptance regarding a specific mobile service category (entertainment) when developing such applications. We achieved this by implementing a prototype application in PhoneGap, a framework for developing hybrid mobile applications. The backend was developed in Node.js, a server-side JavaScript platform. In this section, we will address the implications of the aforementioned implementation, and provide future directions.

The implications of this study are as follows. The implementation resulted in successfully applying almost all mobile service guidelines for use acceptance, except for those regarding perceive enjoyment, since the guidelines are motivated solely by subjective motives. All this shows that the hybrid applications can benefit from all the strengths and opportunities a native application can. In this light, we can conclude that the gap between hybrid and native application is growing smaller. However, the performance of hybrid applications is still worse than their native counterparts, which is an important aspect of mobile service guidelines for user acceptance. Additionally, the result of this study showed that such mobile service guidelines for use acceptance, introduced in this article, can be used by developers for developing user acceptable mobile solutions. Besides, this article can serve as a basis for other future work. This implementation analysed only the application of factors related to entertainment mobile services category, the future studies could analyse other categories as well. In addition, the implemented solutions could be verified by the users of these applications in order to confirm the benefits of applying mobile service acceptance factors to mobile development.

6. REFERENCES

[1] Al-Debei, M. and Al-Lozi, E. (2014). Explaining and predicting the adoption intention of mobile data services: A value-based approach. Computers in Human Behavior, 35, pp.326-338.

[2] Davis, F. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. MIS Quarterly, 13(3), p.319.

[3] Effectiveui.com, (2015). New Study Finds Mobile Application Users Prefer Usability and Good User Experience over Brand Names Alone. [online] Available at: http://www.effectiveui.com/company/newsroom/press-releases/11-10-2010/ [Accessed 8 Sep. 2015].

[4] Gartner.com, (2015). Gartner Forecasts 59 Percent Mobile Data Growth Worldwide in 2015. [online] Available at: http://www.gartner.com/newsroom/id/3098617 [Accessed 8 Sep. 2015].

[5] Gartner.com, (2015). Gartner Says Worldwide Traditional PC, Tablet, Ultramobile and Mobile Phone Shipments to Grow 4.2 Percent in 2014. [online] Available at: http://www.gartner.com/newsroom/id/2791017 [Accessed 8 Sep. 2015].

[6] Hong, S., Thong, J., Moon, J. and Tam, K. (2008). Understanding the behavior of mobile data services consumers. Information Systems Frontiers, 10(4), pp.431-445.

User Interface Development Challenges on Mobile Devices: A Case of Diabetes Risk Estimation Applications

Nino Fijačko
Faculty of Health Sciences,
University of Maribor
Maribor, Slovenia
nino.fijacko@um.si

Petra Povalej Bržan
Faculty of Health Sciences,
University of Maribor
Maribor, Slovenia
petra.povalej@um.si

Gregor Štiglic
Faculty of Electrical Engineering and Computer Science,
University of Maribor
Maribor, Slovenia
gregor.stiglic@um.si

ABSTRACT

Designing an effective graphical user interface (GUI) is a task full of challenges in modern mobile application development. Developers are confronted with specifics of different operating systems as well as different characteristics of hardware even in the same operating system. In this paper we provide a systematic review of mobile applications in the rapidly expanding healthcare mobile application market. More specifically, we observe the functionality of 31 mobile application for diabetes type 2 risk estimation focusing on the user interface design. As the user interface strongly depends on the risk estimation methodology, we also observe which mobile applications rely on validated predictive models and which models are most frequently used.

Unfortunately, we found out that only minority of applications disclose the underlying methodology or validated diabetes risk calculator name. Additionally, given the fact that mobile application development allows developers to enhance the user experience of the mobile application in comparison to classical paper and pencil tests, we observe that very few applications exploit this possibility. In this study, we observed the use of textual, numerical or graphical representation of results in type 2 diabetes risk estimation mobile applications. Again, the results show that very few applications combine all three approaches.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces - *Graphical user interfaces.* D.2.10 [**Software Engineering**]: Design - Methodologies. J.3 [**Computer Applications**]: Life and Medical Sciences - *Health.*

General Terms

Measurement, Performance, Reliability, Experimentation.

Keywords

Android, iOS, Windows Phone, type 2 diabetes, risk estimation, mobile health, user interface design.

1. INTRODUCTION

Identifying persons at increased risk of developing Type 2 Diabetes (T2D) as early as possible is of high importance to avoid possible complications at later stages. A recent study by Shaw et al. [1] points at a rapidly increasing numbers of diabetes patients that are supposed to rise over 350 million by the year 2030. Therefore, it is of high importance to act early and provide advice on healthy lifestyle, especially to people with high estimated risk.

Nowadays, there are multiple diabetes associations present online that see the opportunity to reach a large number of people by letting them use the online risk test calculators where no manual scoring is needed. Furthermore, with the rapid increase of the mobile application market, there is also more and more diabetes risk calculators available in all three major mobile application stores. Not only in case of diabetes risk estimation, one can also find a large number of mobile applications aiming to help users with different chronic diseases. For example, Donsa et al. [2] present how computerized decision support systems and the role of machine learning can help improve the personalization of the patient's diabetes treatment on different levels. Hanauer et al. [3] developed Computerized Automated Reminder Diabetes System for blood glucose monitoring. They argue that using cell phone text messaging offers a highly portable, well-accepted, and inexpensive modality for engaging management of diabetes.

For several decades, healthcare experts mostly performed the screening tests by asking questions or by filling in paper and pencil questionnaires. The risk score was obtained by summing the points corresponding to specific answers to questions ask in such questionnaires. Studies comparing paper to electronic questionnaires date back over a decade, with most of them focusing on user experience and perception of the electronic questionnaires in comparison to their paper-and-pencil versions. Cook et al. [4] compared electronic to paper questionnaires for chronic pain assessment in a randomized, crossover study. Their results support the validity and acceptance of electronic versions with majority of users rating e-questionnaires as easier and preferred.

This paper reviews the functionality of 31 mobile application for diabetes type 2 risk estimation focusing on the user interface design. As the user interface strongly depends on the risk estimation methodology, we also observe which mobile applications rely on validated predictive models and which models are most frequently used.

2. METHODS

In this paper, we follow a Preferred Reporting Items for Systematic reviews and Meta-Analyses (PRISMA) proposed by Moher et al. [5] to perform a systematic review of diabetes type 2 risk estimation mobile applications. Some adaptations were needed due to specifics of mobile applications in comparison to research papers where PRISMA is usually applied. PRISMA checklist includes 27 items that pertain to the content of a systematic review and meta-analysis, which include the title, abstract, methods, results, discussion and funding. Flow diagram in PRISMA depicts the flow of information through different phases of a systematic review and starts with Identification phase where number of records identified through database(s) search is stated. In case of mobile applications, we use mobile application stores as databases to search for records – i.e. mobile applications.

In the next phase all duplicated records are removed and unique records are screened on the basis of inclusion criteria. Records that are not consistent with the inclusion criteria are excluded from further analysis. In the third phase all discovered mobile applications are assessed for eligibility. The applications that do not meet the eligibility criteria are excluded from further analysis.

Table 1 shows a list of devices and software used to conduct a review of type 2 diabetes risk estimation mobile applications. All application searches were executed on a personal computer using the most recent versions of the corresponding web browsers.

Table 1. List of equipment used to evaluate the applications

	Android	**iOS**	**Windows Phone**
Phone	LG Nexus 5	iPhone 5S	Nokia Lumia 1320
Tablet	Asus Nexus 7 (2013)	iPad mini 1	HP ElitePad 1000 G2
Operating System	Lollipop 5.0.2	iOS 8.1.1	Windows Phone 8.1
Web Browser (version)	Chrome browser (39.0.2171.99)	Safari 8.0.3 (10600.3.18)	Chrome browser (39.0.2171.99)

Three experts in the field of healthcare related mobile applications were involved in the review of the mobile applications. Initially, a set of search keywords that were used to search for T2D risk estimation applications was defined. The selected search terms were “diabetes”, “risk” and “health” that were used in combination with keywords “test”, ”calculator”, ”checker”, ”tool”, and ”score”. The first search results included a list of more than 1,500 mobile applications that were then manually inspected for adequacy by title and description by each reviewer. After the first filtering, each of the three reviewers presented a list of resulting applications. All applications chosen by all three reviewers were automatically included in the final list, however the inclusion of the applications that were identified by only one or two reviewers was discussed by all three experts.

An additional criteria for exclusion was language of the application. Only applications in English language were included in the final review. The applications for T2D risk test in German (one for Android and one for iOS), Spanish (two for Android) and Chinese language (one for Android and one for iOS) were therefore excluded from the review.

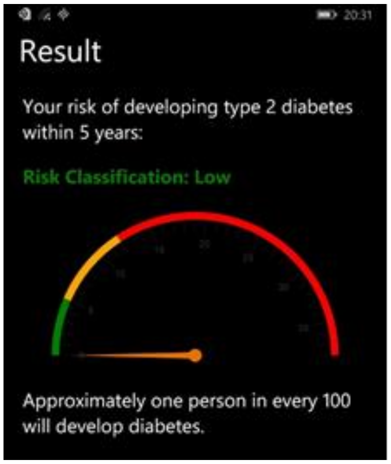
We also excluded applications based on a technical exclusion criteria such as:

- application did not start (two Android and one iOS application),
- disabled login to the application (one Android and one Windows Phone application),
- the functionality and description of the application did not match (one Android and one Windows Phone application),
- location specific application (one Android application).

3. RESULTS

The results of this study are based on the search in three major mobile application stores that was performed in September 2015. We found 31 (16 Android OS; 8 iOS and 7 Windows Phone) (Table 1) eligible applications for T2D risk estimation that met all inclusion criteria. Together we compared 26 freely available and 5 applications where payment was required to download the application (one from Google Play Store, two from iTunes Store and three from Windows Phone Store; range from €0.80 to €1.16; total €4.99).

Our study also focused on the representation of results in mobile T2D risk applications (Table 2). We defined three types of results – i.e. textual, graphical and numerical form. In case of textual form of results, a user is usually presented with feedback on the severity of the risk and additional recommendations on preventing T2D by changing the lifestyle. Numerical results can be represented in form of real numbers, integer values, percentages and numbers inside a textual message. Most often numerical form represents a sum of points for specific answers where the final decision and recommendation depend on the threshold values. Graphical form of results can be represented by a chart visualizing the risk estimation in percentage or even displaying the estimation of risk for the future period at different time points. As it can be seen from Table 2, there are five applications that include all three types of results. However, it has to be noted that in two cases we observed the same application developed for two different operating systems. However, it would be difficult to say that the three applications with a complete representation of results are superior to other compared applications – mainly due to somehow limited graphical user interface in comparison to modern-looking applications. An example of a graphical interface from AUSDRISK (A14) application representing all three types of result representation can be seen in Figure 1.



Result

Your risk of developing type 2 diabetes within 5 years:

Risk Classification: Low

Approximately one person in every 100 will develop diabetes.

Figure 2. An example of a result representation using textual, numerical and graphical representation.

Table 3. List of type 2 diabetes risk estimation applications with corresponding risk score method and representation of results

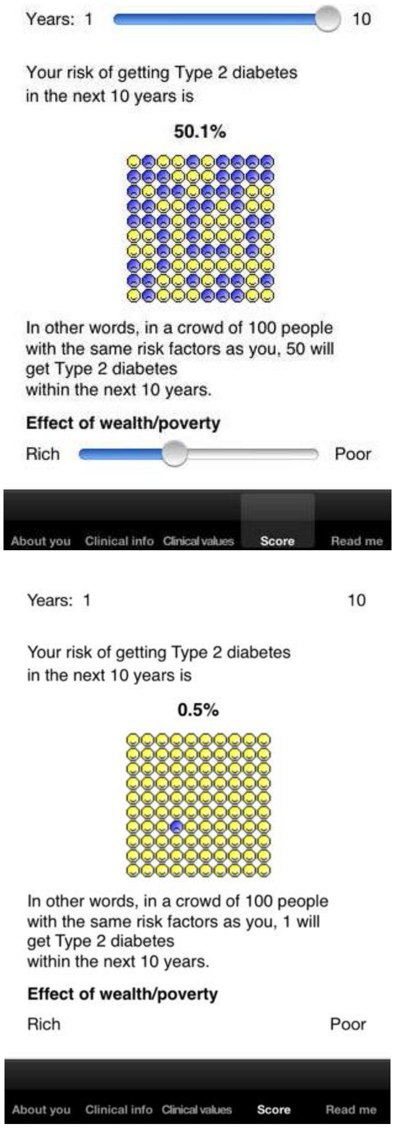
Apps ID	**Short Name**	**Risk score method**	**Risk period**	**Representation of results**			**Available**
				Text	**Numerical**	**Graphic**	
Diabetes risk	A1	NA	5 years	+	+		Free
Diabetes risk calculator	A2	NA	10 years	+	+		Free

Diabetes risk	A3	FINDRISC	10 years	+	+		Free
Diabetes Risk Survey	A4	NA	NA	+	+		Free
Diabetes risk checker	A5	ClinRisk qdiabetes	10 years		+	+	€0.80
Diabetes Risk Score	A6	NA	5 years	+	+	+*	Free
Diabetes test	A7	NA	5 years	+	+		Free
Diabetes risk test (ADA)	A8	FINDRISC	10 years	+			Free
Lloydspharmacy Diabetes Check	A9	NA	NA	+	+		Free
Find risk	A10	FINDRISC	10 years	+	+		Free
FindRisk Diabetes	A11	NA	NA	+	+		Free
Meditas	A12	ADA	NA	+	+		Free
Screening for Type 2 Diabetes	A13	NA	10 years	+	+		Free
AUSDRISK Diabetes Risk Monitor	A14	AUSDRISK	5 years	+	+	+	Free
Are you at risk?	A15	NA	10 years	+	+		Free
BMI & WAIST CheckUp ENG	A16	NA	NA	+			Free
Your diabetes risk calculator	I1	NA	NA		+	+	Free
Qdiabetes	I2	QDiabetes	1-10 years	+	+	+	€1.08
Diabetes risk checker	I3	ClinRisk qdiabetes	10 years		+	+	€1.08
Findrisc	I4	FINDRISC	10 years		+	+	Free
Diabetes test	I5	NA	5 years	+	+		Free
Diabetes risk score	I6	NA	5 years	+	+	+*	Free
Diabetes Risk	I7	FINDRISC	10 years	+	+		Free
CANRISK Diabetes questionnaire	I8	CANRISK	10 years	+	+		Free
Diabetes calculator	W1	NA	NA	+	+		Free
Meditas	W2	ADA	NA	+	+		Free
Diabetes predictor	W3	NA	NA	+	+		€1.16
Diabetes risk checker	W4	ClinRisk qdiabetes	10 years		+	+	€0.87
Diabetese II Test	W5	NA	NA	+			Free
Ausdrisk	W6	AUSDRISK	5 years	+	+	+	Free
Diabetes risk tool	W7	NA	NA	+	+		Free

*After re-calculation of the risk.

Table 2 also shows the underlying risk scoring method where one can observe some most frequently used methods from paper and pencil questionnaires such as FINDRISC [6], AUSDRISK [7], ADA [8] or Qdiabetes [9]. The most widely used questionnaire (i.e. FINDRISC) is also the most frequent among the reviewed mobile applications. FINDRISC is followed by Qdiabetes, AUSDRISK, CANRISK and ADA screening methods. However, the most concerning fact from this review reveals that the majority (55%) of the reviewed mobile applications did not disclose the methodology they are using to calculate the risk of type 2 diabetes.

As already mentioned, we compared mobile applications using different hardware as well as operating systems. Different mobile hardware and software characteristics are not only challenging to review, but also represent a great challenge for developers. Therefore, we believe one of the important factors in mobile applications is also frequency of updates that keep the application up to date with the current software and hardware available on the market. As an example, one can observe that an update in the operating system often results in unexpected behavior of the graphical interface of the application. Figure 2 represents an application tested using iOS version 8.4 (lower screenshot) and an older version of the iOS 8.1.1 (upper screenshot) where a sliders control simply disappears with an upgrade to a newer version of the operating system.



Years: 1 10

Your risk of getting Type 2 diabetes
in the next 10 years is

50.1%

In other words, in a crowd of 100 people
with the same risk factors as you, 50 will
get Type 2 diabetes
within the next 10 years.

Effect of wealth/poverty

Rich Poor

Years: 1 10

Your risk of getting Type 2 diabetes
in the next 10 years is

0.5%

In other words, in a crowd of 100 people
with the same risk factors as you, 1 will
get Type 2 diabetes
within the next 10 years.

Effect of wealth/poverty

Rich Poor

Figure 2. An example of an unwanted impact on graphical interface with an upgrade of the operating system.

4. DISCUSSION AND CONCLUSIONS

The results of the analysis are a little surprising since we found out that more than half of mobile T2D risk estimation applications does not provide the user with the information on the empirical scientific background of the risk estimation test used in the application. We believe that this is an essential information for the user in order to be able to asses in what extend the calculated risk is accurate. An additional link to the scientific paper that can offer more information for the user about the risk estimation test used, is also a useful addition to the user.

Another important factor for the end user is presentation of the results. We observed three different types of presentation (i.e. textual, numerical and graphical). There were only a few applications that included all three types of presentation of the results (A6, A14, I2, I6, W6). However, it would be difficult to say that advanced representation of the results makes those applications superior to other compared applications in general – mainly due to somehow limited graphical user interface in comparison to modern-looking applications.

Additionally to risk assessment score, the user would benefit also from more detailed description why the risk is high / low and in case of increased risk the directions for reduce it. A list of scientific papers about user specific risk factors would be a good for obtaining first information about the risk factors and how to avoid them.

5. REFERENCES

[1] Shaw, J. E., Sicree, R. A., and Zimmet, P. Z. 2010. Global estimates of the prevalence of diabetes for 2010 and 2030. *Diabetes. Res. Clin. Pract.* 87, 1 (Jan. 2010), 4-14.

[2] Donsa, K., Spat, S., Beck, P., Pieber, T. R., and Holzinger, A. 2015. Towards personalization of diabetes therapy using computerized decision support and machine learning: some open problems and challenges. In *Smart Health. Springer International Publishing* 237-260

[3] Hanauer, D. A., Wentzell, K., Laffel, N., and Laffel, L. M. 2009 Computerized Automated Reminder Diabetes System (CARDS): e-mail and SMS cell phone text messaging reminders to support diabetes management. *Diabetes. Technol. Ther.* 11, 2 (Feb. 2009), 99-106.

[4] Cook, A. J., Roberts, D. A., Henderson, M. D., Van Winkle L. C., Chastain, D. C., and Hamill-Ruth, R. J. 2004. Electronic pain questionnaires: a randomized, crossover comparison with paper questionnaires for chronic pain assessment. *Pain.* 110, 1-2 (Jun. 2004), 310-317.

[5] Moher, D., Liberati, A., Tetzlaff, J., Altman, D. G., and PRISMA Group. 2010. Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. *Int. J. Surg.* 8, 5 (Feb 2010), 336-341.

[6] Lindström, J., and Tuomilehto, J. 2003. The diabetes risk score: a practical tool to predict type 2 diabetes risk. *Diabetes. Care.* 26, 3 (Mar. 2003), 725-31.

[7] Chen, L., Magliano, D. J., Balkau, B., Colagiuri, S., Zimmet, P. Z., Tonkin, A. M., Mitchell, P., Phillips, P. J., and Shaw, J.E. 2010. AUSDRISK: an Australian type 2 diabetes risk assessment tool based on demographic, lifestyle and simple anthropometric measures. *Med. J. Aust.* 192, 4 (Feb. 2010), 197-202.

[8] Bang, H., Edwards, A. M., Bomback, A. S., Ballantyne, C. M., Brillon, D., Callahan, M. A., Teutsch, S. M., Mushlin, A. I., and Kern, L. M. 2009. Development and validation of a patient self-assessment score for diabetes risk. *Ann. Intern. Med.* 151, 11 (Dec. 2009), 775-783.

[9] Hippisley-Cox, J., Coupland, C., Robson, J., Sheikh, A., and Brindle, P. 2009. Predicting risk of type 2 diabetes in England and Wales: prospective derivation and validation of QDScore. *BMJ.* 338 (Mar. 2009), b880.

AgroIT project: Standards-based ICT cloud integration for farming

Rok Rupnik
University of Ljubljana, Faculty of
Computer and Information Science
and
Datalab Tehnologije d.d.
1000 Ljubljana, Slovenia
+386 41 719 209
rok.rupnik@datalab.eu

ABSTRACT

This paper introduces EU funded AgroIT project which has several goals which all aim at setting up on standards based cloud integration platform for integrating various applications and devices. In the first part of the paper the AgroIT project and its goals are briefly introduced. After that the paper introduces the concept of pilot projects which represent a PoC (proof of concept) for the use of applications and devices which are integrated over a cloud integration platform. The core of the paper provides discussion about the need for standards in data exchange in farming. The need for merging efforts in defining standards for data exchange is presented as the answer to several EU projects and initiatives which all in general aim at defining standard for data exchange in farming.

Categories and Subject Descriptors

D.2.12 [**Interoperability**]

D.2.11 [**Software architectures**]

H.2.4 [**Systems**]

D.4.2 [**Types of systems**]

General Terms

Management, Economics, Standardization.

Keywords

Cloud integration platform, farming, integration, standard for data exchange.

1. INTRODUCTION

Over recent decades farming has been witnessing the deployment of various sophisticated technologies in the areas of machinery and automation. We believe that the deployment of IT is the next step forward which will change farming and enable the transition to higher quality and higher efficiency farming. An increase in EU and national regulations, such as cow registers, FADN, etc., already demands significant level of bureaucracy from farmers. Recent publications showed that on average a farmer in Germany spends 20 hours a week for administrative and paper work, third of which is spent on accounting and tax information return [1]. The average time spent on administrative and paper work in the four countries studies was of 7 hours per week material. European farmers are obligated to provide more and more (electronical) reports to EU and national institutions. As a consequence they need farming adapted ERP systems with various other applications which will provide various data needed for reporting to EU and national institutions. Other applications must be integrated with ERP systems in order to provide extensive options for reporting. Hence, integration is with no doubt an important direction which development of IT in agriculture should follow. There are various IT solutions available on the market and they cover variety of niche areas which lack a very important characteristics: they are mostly not integrated.

The AgroIT project is an EU funded project which tackles the issue of the integration of IT solutions and various systems in agriculture. The overall objective of the project is to implement cloud integration platform which will enable standardized integration of various IT solutions and systems.

2. OVERVIEW of PROJECT GOALS

2.1 First objective: Implementation of advanced decision support for farming

Various modules using various methods and algorithms for data analysis will be developed: data in ERP system of a farm, data gathered from various sensors and devices, and data gathered through cloud services. It is more than a classical BI system which shows data from various aspects. Decision support system will also generate recommendations for the farmer and planning through generating recommended daily scenarios.

2.2 Second objective: Implementation of mobile applications

The second objective is to implement various mobile applications for farming that enable simple and efficient input of data during the execution of farmer's daily activities. Analysis has shown that a significant part of data to track daily activities of the farmer can be entered on as-you-go concept through mobile applications: either through simple input or automatically through technologies like GPS, RFID and NFC.

2.3 Third objective: The integration of services and applications for enabling extensive data collection

The third objective is to integrate monitoring systems, which will enable the collection of data from sensors and other devices through wireless communication technologies. There are various sensors which we will integrate into the platform through monitoring systems: sensors for irrigation, humidity and temperature in the barn, water level, air pressure, gasses in barn, etc. We will also integrate various devices through their monitoring systems into the platform: milking robots, traps for pests/insects and based on that pest/insect monitoring system and agriculture specialized weather stations. The integration of the mentioned elements has the following added value: various data will be available to ERP and decision support system to enable decision support. ERP will enable electronically reporting of statistical data to government institutions and EU institutions.

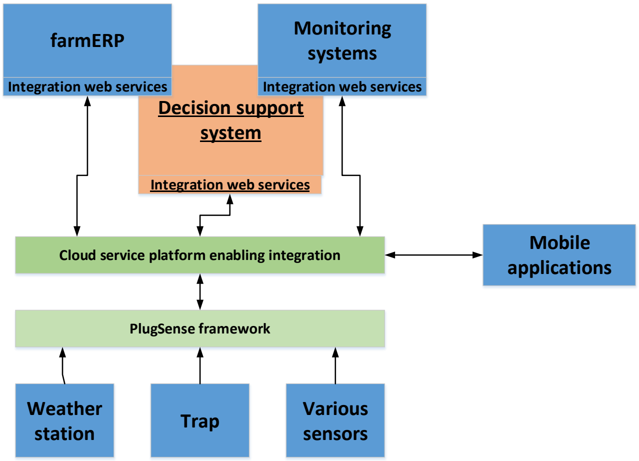


Figure 1: The scehma of AgroIT platform and systems integrated

2.4 Fourth objective: Integration and creating open standards for integration

Fourth objective is to integrate individual elements into platform as a whole and to create open standards for integration: within AgroIT platform and with AgroIT platform. Standards will be published under public domain or creative-commons license to allow collaboration, large support within the industries and communities. This objective will enable the long term expansion of AgroIT platform and through that various and new benefits to all stakeholders. Integration principle of platform and possibility for integration of any ICT based service and application will enable faster implementation periods for demands of EU (and national governments): shorter time from defining element of legislation (directive, agenda, etc.) to ICT based implementation in farming.

2.5 Overall objective: To implement and integrate AgroIT platform

The overall objective of the project is to implement AgroIT platform. AgroIT platform is an open cloud integration platform based on open standards. AgroIT will in long term deliver applications and services to various stakeholders: farmers, local communities, state institutions, consulting institutions in farming (government funded and private) and EU institutions.

As part of the project open standards will also be defined. This will accelerate the transfer of innovative applications and services to the market by: quicker integration of new ICT elements (applications, services, and monitoring systems) into the platform and as a consequence quicker transfer of integrated ICT solutions to market. After the project farmers will not be limited to selection of "project products", i.e. products, which will be finalized based on prototypes implemented within AgroIT project.

3. Pilot projects

In AgroIT project consortium there are several pilot partners. The role and responsibility of software partners in to implement software products, while the role of pilot partners is to organize and run pilot projects.

The goal of pilot projects is to:

- Select proper pilot farms: there are 100 pilot farms planned in all pilot countries. In the interest of the project is to have various types of pilot farms with characteristics spread over the following areas: size of farm, regions within the pilot country, area of farm main stream activity (livestock, fruit growing, vineyard growing, …)
- Organize pilot environments in pilot countries. There are several pilot countries: Poland, Slovenia, Romania, Denmark and Austria
- Install software products and devices at pilot farms
- Workshops and seminars execution to teach farmers use software an devices
- Provide support for pilot farms in advanced using of software and devices
- Execute detailed analyses of pilot projects

Pilot projects will start in October 2015 and will finish in December 2016. The duration of 15 months will cover all seasons and enables pilot farms to close year 2016 in accountancy.

In the final stage of pilot projects the following analyses will be performed:

- Scalability and sustainability analysis
- Final functional reports for each country
- Final technical reports for each pilot country
- Mobile applications usability report

4. PROJECT CONSORTIUM

Figure below shows the EU countries of consortium partners. Software partners come from:

- Slovenia:
  - Datalab: project coordinator; responsible for ERP system for farming,
  - Efos: responsible for electronic trap, i.e. the platform to detect pests,
  - Sinergise: responsible for GIS
  - University of Ljubljana, Faculty of Computer and Information Science: responsible for cloud integration platform
- Portugal: FreedomGrow, responsible for sensors
- Austria: Pessl Instruments, responsible for mobile weather station
- Denmark: SEGES, responsible for ERP system for farming

Pilot partners come from:

- Slovenia
- Austria
- FYRM
- Poland
- Denmark

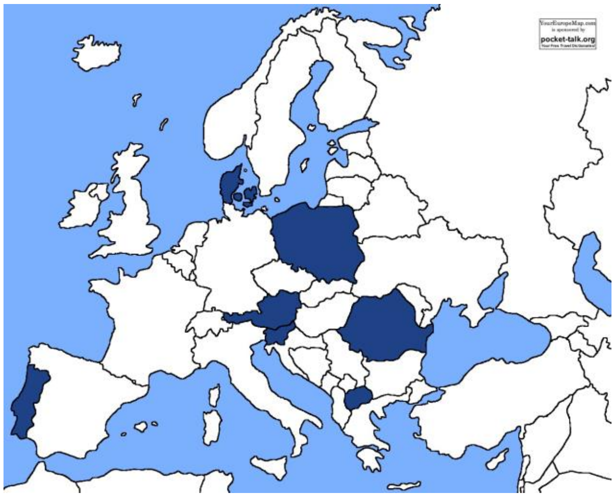


Figure 2: Consortium partners

5. STANDARDS FOR INTEGRATION

Important goal of the project is to define standards for integration of any systems (application or device) to the platform. The main idea of the platform is to enable the integration of any system with the platform and through the platform a particular system integrated will be integrated to any other integrated system: there will be no one-to-one integrations of various systems, every system will integrate only with platform. For that reason standard for integration need to be defined.

We plan to submit standard proposal to Organization for the Advancement of Structured Information Standards (OASIS). More specifically, we will define a standard for the exchange of data within systems from the AgroIT domain (ERP, sensors, decision support systems, etc.). This standard will be an extension of the Open Data Protocol (OData) and submitted to Organisation for the Advancement of Structured Information Standards (OASIS).

Furthermore, we will examine the possibility to develop a standard for definition of domain specific schemas based on a data description language Extensible Markup Language (XML) Schema, a standard for transformation between the domain specific data formats based on a data transformation language Extensible Stylesheet Language Transformations (XSLT), and to submit it to OASIS. Next, we will examine the possibility to develop a standard for integration of SaaS/XaaS solutions, which would include a standardized data format, data transformations and data exchange protocol for descriptive-driven autonomous data integration and synchronization from the authoritative SaaS/XaaS solution to subordinate SaaS/XaaS solutions. This standard candidate is to be submitted to OASIS as a new standard proposal. Another candidate for standardisation, i.e. extension of existing standards Web Services Description Language (WSDL) and/or Web Application Description Language (WADL), Web Services Business Process Execution Language (WS-BPEL) and/or Business Process Model and Notation (BPMN) and/or Yet Another Workflow Language (YAWL), and other Web Services standards (WS-*), is the protocol for integration of user interfaces from different SaaS/XaaS solutions to support execution of human activities in the automated workflows, regardless which integrated SaaS/XaaS solution the user interface implementation (or its part) belongs to. The will be submitted to OASIS.

A program steering committee will be founded after the project within AgroIT open standards consortium, comprising of top experts for research and development processes. The committee will be responsible for definition and usage of best practices for development and research activities. These best practices include standardized data and service integration approaches and top-down/contract-first development. Throughout the project, the committee will assure a satisfying quality of final results to be presented in a form of standard proposals.

5.1 Other projects, organisations and initiatives working on standards for data exchange in farming

There are several projects, organisations and initiatives working on standards for data exchange in ICT for farming.

FOODIE project (Farm Oriented Open Data in Europe) will deliver a platform hub on the cloud. Spatial and non-spatial data related to agricultural sector will be available for agri-food stakeholder groups. The following data groups will be covered by their standard [2]:

- GIS data
- Sensor data, data from machines (tractors) and weather stations
- Products (pesticides)
- Subsidies
- Satellite data

agroXML is a markup language for agricultural data providing elements and XML data types for representing data on work processes on the farm including accompanying operating supplies like fertilizers, pesticides, crops and the like. It is defined using W3C's XML Schema. agroRDF is an accompanying semantic model that is at the moment still under heavy development. It is built using RDF. We analysed agroXML and believe that its concept is good. It only seems that the authors of agroXML didn't manage to make a breakthrough and reach the critical mass of the use of standard [3].

agriXchange is a EU-funded coordination and support action to setup a network for developing a system for common data exchange in the agricultural sector [4].

There are only key projects and initiatives mentioned above, there are also some other examples which could be mentioned. As it can be revealed from discussion above there are several standards: existing or emerging. We believe that having many standards is like have no standard. It is essential to merge efforts in the area of standardization of data structures and data exchange in farming and agriculture.

6. CONCLUSION

Agriculture and farming are areas where the expansion of ICT investments will continue. On standards based integration of various systems is the direction which assures farmers the highest value added of the use of the systems.

AgroIT is one of EU funded projects with goals that will in short-, mid- and long term enable on standards based integration of various systems (applications and devices) in ICT for farming. As the paper revealed there are various existing and emerging standards in area of data exchange in farming. In order to accomplish the general mission of standards, efforts in this area must be merged due to the simple fact: having many standards is like having no standard.

7. REFERENCES

[1] Lartey G. Lawsona, Søren Marcus Pedersen, Claus Grøn Sørensen, Liisa Pesonen, Spyros Fountas, Armin Wernere, Frank W. Oudshoorn, Luzia Herold, Thanos Chatzinikos, Inger Marie Kirketerp. 2011. A four nation survey of farm information management and advanced farming systems: A descriptive analysis of survey responses. *Computers and Electronics in Agriculture*. 77, (Mar. 2011), 7-20

[2] Foodie project web site. http://www.foodie-project.eu/

[3] agroXML web site. http://www.agroxml.de/

[4] agriXchange web site. http://www.agrixchange.org/

Method for linking theoretical and practical knowledge within expert systems, and the application of an expert system for welding

Ambrož Stropnik
KiviCOM d.o.o.
Kidričeva 3a
Slovenj Gradec, Slovenia
ambroz@kivi.si

Tomaž Vuherer
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
tomaz.vuherer@um.si

Milan Zorman
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
milan.zorman@um.si

ABSTRACT

Modern companies hold lots of knowledge in different forms (e.g. documents, knowledge of employees …). Although many companies have implemented knowledge management support systems for different purposes (e.g. developing new products), not all of the expertise is captured within a knowledge management system because some of that knowledge is based on personal experience. In this article, we propose a new method for capturing the experiences and knowledge of employees, and present an example of an expert system for welding.

Categories and Subject Descriptors

I.2.6 [**Artificial intelligence**] Learning – *Knowledge acquisition, Parameter learning*.

General Terms

Algorithms

Key words

Linking knowledge, expert system, ontology, welding process optimisation

1. INTRODUCTION

Nowadays, successful companies are confronted with the need for using advanced knowledge management systems, mostly within the field of business processes such as the development of new products, manufacturing production planning, project management, etc. Every business process consists of two skills – basic theoretical knowledge and practical knowledge, resulting from the experiences of every employee. Usually, there is a gap between theoretical and practical knowledge because the theoretical knowledge is accessible to everybody (e.g. books, documents, etc.), while practical knowledge is limited to a certain number of employees. In most cases, practical knowledge is limited to only one employee or varies greatly amongst employees, resulting in significant variations during planning and decision-making (e.g., two planners in the same company create different plans for the same product or they make completely different decisions). However, nowadays such deviations regarding decisions and knowledge are unacceptable.

This paper presents our approach for linking theoretical and practical knowledge within the field of expert systems, and for process support when welding.

Chapter 2 introduces the theoretical background of an expert system and different approaches when using it during manufacturing and production planning processes. Chapter 3 presents our approach for linking theoretical and practical knowledge based on a correction factor. The theoretical background and method of the proposed expert system for welding is also described here. Chapter 4 presents the technical background of a welding expert system and its application within the Nieros Metal Company. Finally, the results of the proposed method close the article in Chapter 5.

2. BACKGROUND AND RELATED WORK

The development of expert systems started in the 1960s. The first known expert systems were DENDRAL and MYCIN, solving complex problems in medicine. Later on, many expert systems were developed, primarily for research purposes [1]. Only a few expert systems were developed “in practice”, especially in the field of manufacturing production processes (e.g. welding). New technological opportunities for the developments of expert systems emerged with the development of the Internet. Using Semantic Web, technologies gave expert systems a new meaning [2]. Hence, new theoretical classifications of expert systems emerged called “Internet expert systems” [3]. The progresses of technologies and the Semantic Web also considerably increased the uses of expert systems in practice. Within the field of knowledge management systems, Baloh and others presented the pilot knowledge portal based on semantic web technologies for supporting a new product development regarding the Slovenian white goods company Gorenje d.d. [4]. An expert system for aluminium welding – WELDES is known within the field of welding [5]. The WELDES expert system is used as an industrial tool for the identifications, assessments, and corrections of aluminium welding defects. The system consists of two modules: the Diagnostic Module and the Adviser Module [5]. There are also many semantic models of knowledge bases for capturing and organising different sources of data during manufacturing production processes [6, 7].

However, there is presently no such system that could link theoretical and practical knowledge in useful form.

3. METHOD FOR KNOWLEDGE LINKING

The proposal method consists of the following steps that are crucial for accurate functioning of the entire system:

- creating welding documentation (e.g. WPS document),

- transferring welding documentation to the welder during the manufacturing production process,
- using feedback from the parameters during the welding process.

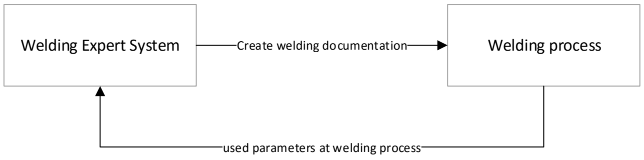

Welding Expert System
Create welding documentation
Welding process
used parameters at welding process

Figure 1 : Feedback loop

Feedback loop presents the crucial part within the expert system.

3.1 Knowledge formalisation

Decision tables form the entire knowledge needed for the preparation of welding documentation (see Table 1). They are based on thickness of material data (both for A and B materials) and type of weld (e.g. S-I weld). The values in decision tables are numeric and determined by the welding coordinator. Values based on welding standards and the practical experience of welding experts.

However, there is a set of decision tables used for creating welding documentation:

- Welding current
- Speed of welding
- Gas flow
- Diameters of the electrodes
- Diameter of the nozzle
- Diameter of the filler material
- Number of welds
- Gas
- Voltage

Table 1: Example of decision table structure

Type of weld (e.g. S-I)					
	*Thickness of metal sheet of base material **B** [mm]*				
*Thickness of metal sheet of base material **A** [mm]*		**0.8**	**1**	**...**	**n**
	0.8	11	11		
	1	11	9		
	2	10	8		
	...				
	N				

From the technological point of view, all decision tables are being converted to decision rules by an expert system.

Decision rules have the following general form:

IF A AND B AND C THEN D

In practice, the welding coordinator manages all decision tables and sets all of the values.

3.2 Correction factor

The correction factor represents a link between those values purposed in the decision tables of an expert system and those values used in the manufacturing production process (i.e. in the field of the welding process). It is used for retrieving values from decision tables as a product of the correction factor and the original value. In order to determine the correction factor, a feedback loop of an expert system, through which parameters are used during a welding process (e.g., speed, etc.) are obtained, and this is crucial. The correction factor is calculated based on values during the welding process; it is also calculated as the ratio between the average of the used value and the value determined from decision table. However, the correction factor depends on the thickness of material, type of weld (i.e. S-I), and type of decision table (i.e. speed).

4. IMPLEMENTATION AND CASE STUDY

The proposed method was implemented as an upgrade of the existing expert system for welding support in the Nieros Metal Company. This expert system is a web application, based entirely on Microsoft technologies, with DotNetNuke CMS framework, written in .NET C#. Semantic web technologies, implemented in Jena.NET library, were used for knowledge representation and reasoning [8]. The expert system was integrated within the Company's existing ERP system (see Figure 2 for system architecture) and therefore cannot run independently.

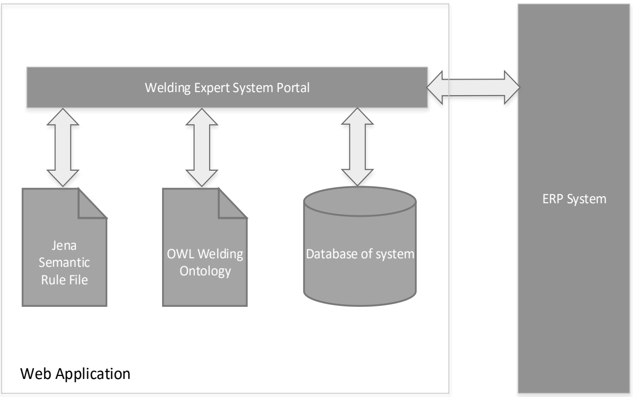
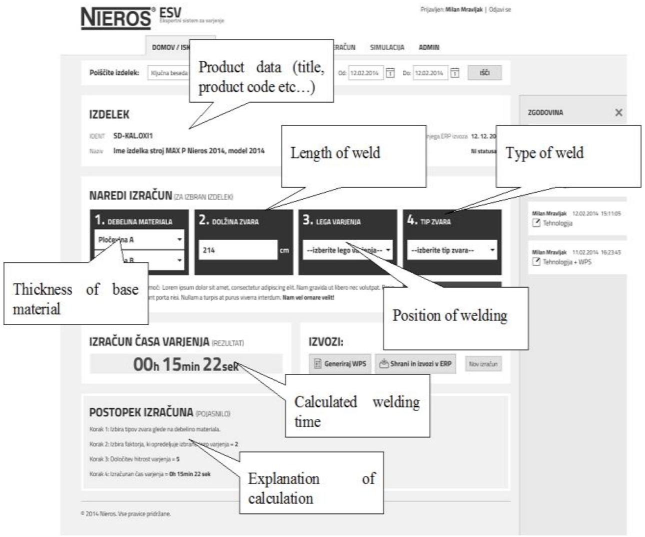


Figure 2: System Architecture

The application of our expert system is focused on the product design phase of Nieros Metal. The system relies on different parameters such as type of material and type of weld in order to calculate the time of welding and prepare underlying documents for the production process – the WPS – Welding Process Specification document.



IZDELEK

IDENT SD-KAL.OXI1

Naziv Ime izdelka stroj MAX P Nieros 2014, model 2014

Ni statusa

NAREDI IZRAČUN (ZA IZBRAN IZDELEK)

1. DEBELINA MATERIALA: Pločevina A

2. DOLŽINA ZVARA: 214 cm

3. LEGA VARJENJA: --izberite lego varjenja--

4. TIP ZVARA: --izberite tip zvara--

Milan Mravljak 12.02.2014 15:11:05 Tehnologija

Milan Mravljak 11.02.2014 16:23:45 Tehnologija + WPS

IZRAČUN ČASA VARJENJA (REZULTAT)

00h 15min 22sek

IZVOZI:

POSTOPEK IZRAČUNA (POJASNILO)

Korak 1: Izbira tipov zvara glede na debelino materiala.

Korak 2: Izbira faktorja, ki opredeljuje izbrano lego varjenja = 2

Korak 3: Določitev hitrost varjenja = 5

Korak 4: Izračunan čas varjenja = 0h 15min 22 sek

© 2014 Nieros. Vse pravice pridržane.

Figure 3: ESV – Expert System for Calculating Welding Time

5. EXPERIMENT AND DISCUSSION

The proposed method for connecting theoretical knowledge and practical experiences was piloted in Nieros Metal on a limited and controlled set of materials and required types of welds.

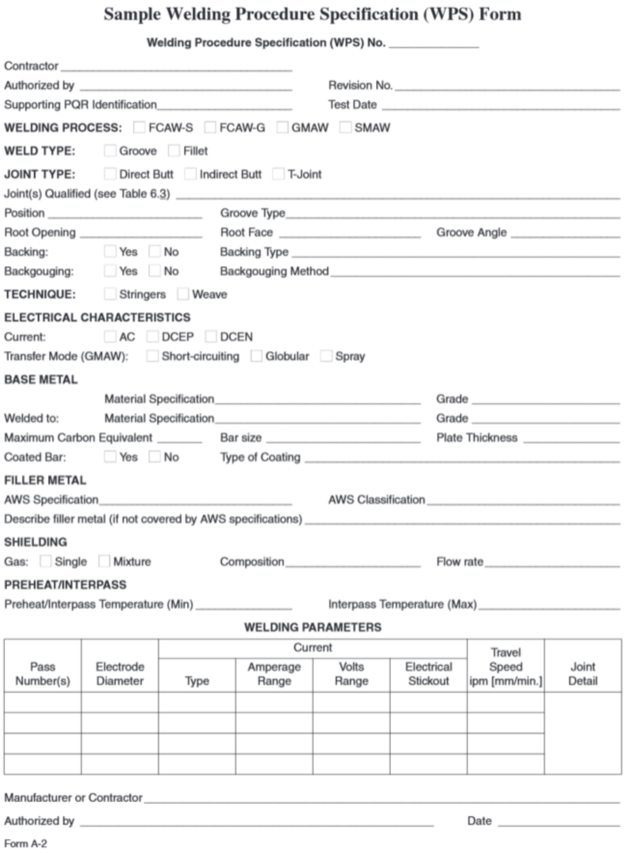
In order to maintain greater control over the experiment, we decided to test our proposed method on a specific type and thickness of the material, using a specific weld (stainless steel; 1.25mm thick; S-I weld). The expert system generated the WPS document with detailed welding instructions (see Figure 4). The key data for welding on a WPS document are the welding parameters, which we observed during the experiment.

During the manufacturing process, we followed the welders who were welding the components according to the documentation, provided by the expert system. Our main task was to focus on the actual welding parameters used during the welding process, as they can deviate from the target parameters for different reasons, such as changes in voltage.

With such an approach, we had at our disposal two datasets: one with theoretical knowledge, represented by the WPS document; and the other with the practical knowledge, acquired through observation of the welding parameters during the actual welding process.

During our experiment, we observed the welding processes, based on 18 different WPS documents and the collected data during their execution. That data was then used for verification of the proposed method.

The first results of the usage were very positive and encouraging.



Sample Welding Procedure Specification (WPS) Form

Welding Procedure Specification (WPS) No. ______________

Contractor ________________________________

Authorized by ________________________________ Revision No. ________________________________

Supporting PQR Identification ____________________ Test Date ________________________________

WELDING PROCESS: □ FCAW-S □ FCAW-G □ GMAW □ SMAW

WELD TYPE: □ Groove □ Fillet

JOINT TYPE: □ Direct Butt □ Indirect Butt □ T-Joint

Joint(s) Qualified (see Table 6.3) ________________________________

Position ____________________ Groove Type ________________________________

Root Opening ____________________ Root Face ____________________ Groove Angle ______________

Backing: □ Yes □ No Backing Type ________________________________

Backgouging: □ Yes □ No Backgouging Method ________________________________

TECHNIQUE: □ Stringers □ Weave

ELECTRICAL CHARACTERISTICS

Current: □ AC □ DCEP □ DCEN

Transfer Mode (GMAW): □ Short-circuiting □ Globular □ Spray

BASE METAL

Material Specification ____________________ Grade ______________

Welded to: Material Specification ____________________ Grade ______________

Maximum Carbon Equivalent _______ Bar size ____________________ Plate Thickness ______________

Coated Bar: □ Yes □ No Type of Coating ________________________________

FILLER METAL

AWS Specification ____________________ AWS Classification ____________________

Describe filler metal (if not covered by AWS specifications) ________________________________

SHIELDING

Gas: □ Single □ Mixture Composition ____________________ Flow rate ______________

PREHEAT/INTERPASS

Preheat/Interpass Temperature (Min) ______________ Interpass Temperature (Max) ______________

WELDING PARAMETERS

Pass Number(s)	Electrode Diameter	Current				Travel Speed ipm [mm/min.]	Joint Detail
		Type	Amperage Range	Volts Range	Electrical Stickout		

Manufacturer or Contractor ________________________________

Authorized by ________________________________ Date ______________

Form A-2

Figure 4: Example of WPS document

As expected, the values in the decision tables were being modified according to information from the production process. We also detected some issues that must be addressed, such as notifying the welding coordinator about the changes of some specific values (e.g. the suggested value of the exert systems differed from the value that was actually used in the welding process).

The results of the experiment showed that the difference between the theoretical knowledge and practical knowledge reduces through every run. The method itself is constructed so that it calculates the arithmetical mean between the target value and actual value. In this way, the welding parameters for the next generation of WPS adapt and carry on the new knowledge.

Of course, there still remain some differences between the theoretical and practical knowledge but even if we showed that the gap decreases, it will never completely disappear. The differences between the theoretical and practical knowledge will remain present, as there are many factors that influence the process: welding devices, their technical conditions (wear), calibration, etc. Therefore we intend to continue our work and enhance our proposed method even further.

As we have already mentioned, we developed our proposed method for a specific expert system, used within a specific manufacturing company.

However, that does not mean that our method cannot be replicated and adapted to other knowledge systems. We can adapt it to knowledge systems that rely on decision tables with numerical values. Unfortunately, the method cannot be used for non-numerical knowledge systems, as yet.

6. CONCLUSION

In this paper we have presented a method for combining the theoretical knowledge and practical experiences with the usages of correction factors. The proposed method was implemented as a pilot study within the real-world environment of the Nieros Metal Company's expert system for welding.

Firstly, the results show that this method is a step in the right direction and that our research should be resumed for developing new even more enhanced expert systems that will be able to support the welding process.

7. REFERENCES

[1] E. H. Shortliffe, S. G. Axline, B. G. Buchanan, T. C. Merigan, S. N. Cohen, An Artificial Intelligence program to advise physicians regarding antimicrobial therapy, (1973) Computers and Biomedical Research, vol. 6, n. 6, pp 544–560

[2] T. Berners-Lee, Putting the Web back into Semantic Web, ISWC2005 Keynote, http://www.w3.org/2005/Talks/1110-iswc-tbl Accessed on 19th Februar 2013.

[3] O. Verhodubs, J. Grandspenkis, Towards The Semantic Web Expert System, (2011) Scientific Journal of Riga Technical University, vol. 47

[4] P. Baloh, V. Vasić, C. Desouza, A. Stropnik, Application of KM Technology in development of new multifunctional products, (2006) Knowledge Management in Organizations, Maribor 2006, pp. 75-81

[5] V. Tsoukalas, M. Kontesis, E. Badogiannis, D. Papachristos, N. Fragiadakis, Prototype of an expert system for aluminum welding, (2006) CIMMACS'06 Proceedings of the 5th WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics, pp 78-83

[6] Yuh-Jen Chen , Yuh-Min Chen , Hui-Chuan Chu, Development of a mechanism for ontology-based product lifecycle knowledge integration, (2009) Expert Systems with Applications: An International Journal, v.36 n.2, pp.2759-2779

[7] H. Panetto , M. Dassisti , A. Tursi, ONTO-PDM: Product-driven ONTOlogy for Product Data Management interoperability within manufacturing process environment, (2012) Advanced Engineering Informatics, v.26 n.2, pp. 334-348

[8] Jena.NET - Flexible .NET port of the Jena semantic web toolkit, http://semanticweb.org/wiki/Jena_.NET Accessed on 19th Februar 2013.

Review of classification systems for e-collaboration

Zmago Fluks
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
+386 68 136 889
zmago.fluks@gmail.com

Marjan Heričko
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
+386 2 220 72 95
marjan.hericko@um.si

ABSTRACT

Electronic collaboration (e-collaboration) is becoming a more and more widespread and used concept in the modern world. Therefore, we decided to examine various classifications and taxonomies in the context of e-collaboration IT solutions in our research paper. They can help us better understand different e-collaboration IT solutions, classify them, or even build a new classification.

Categories and Subject Descriptors

K4.3 [Computer-supported collaborative work]:

General Terms

Collaboration, Communication, Corporation

Keywords

e-collaboration, collaboration tools, classification.

1. INTRODUCTION

Collaboration is a recursive process where two or more people or organizations are working together at the intersection of common goals (e.g. intellectual effort) which are by nature creative-oriented, with a focus on knowledge sharing, learning and building consensus [1]. E-collaboration is defined as the collaboration between individuals who are performing common tasks with the use of information technologies [2].

Sarma in her paper [3] notes that while there are a number of classification systems for organizing e-collaboration IT solutions, none of them are comprehensive enough. Either they focus on some individual aspects of collaboration or on some specific mechanisms, which are then followed by IT solutions.

Our objective was to identify, analyze and compare different classification systems for e-collaboration. The method we used was a literature review.

1.1 Categorization, classification and taxonomy

According to the Encyclopedia Britannica, the term taxonomy represents the science of classification [4], while the dictionary of informatics [5] describes it as principles of classification in groups using an exact set of criteria. Taxonomy is also described as a science or technique that is used to build the classification [6]. Bafoutsou [7] describes the classification of e-collaboration IT solutions by the establishment of taxonomy.

Jacob [8] explains that the term classification relates to the system of classes that are arranged based on pre-determined principles that are used for the distribution of a set of entities [8]. The classification system is used as a presentation tool to organize sets of information. Categorization is the process of dividing the world into groups of entities whose members are in some way similar to each other.

This means that when talking about classification, we combine things that have something in common, and when talking about categorization, we describe the properties of the parts inside a group.

2. LITERATURE REVIEW

In the following part we will explore different taxonomies and classifications within the context of e-collaboration IT solutions. Our research is limited to sources that are freely accessible on the World Wide Web and sources that are accessible in scientific databases. The search was performed by the following search engines: Google, Google Scholar, Bing, IEEE Xplore and ScienceDirect. The search queries that were used and the number of results are presented in Table 1.

Table 1: Used search queries in the literature review

Search query	Number of results				
	Google	Google Scholar	Bing	IEEE Xplore	ScienceDirect
E-collaboration marketplace	8,350,000	10,200	37,300	3	8,897
Collaboration software market	132,000,000	523,000	27,100,000	418	24,478
E-collaboration software functionalities	14,800,000	8,130	32,700	9,070	14,930
E-collaboration software classification	4,250,000	8,160	25,800	2	23,639
E-collaboration classification	3,420,000	8,700	26,500	2	74,600
E-collaboration classification scheme	7,430,000	2,170	24,500	8,990	19,659

In our research we found out that the literature review of taxonomies for e-collaboration was already presented in the article "Review and functional classification of collaborative systems" [7] where the presented classification of dimensions is built according to different authors. Taking into consideration the classification dimensions in the article [7] and the year in which it was published (2002) we think that we could extend, supplement or update the proposed dimensions with an additional literature review.

That is why our literature review focused on searching and categorizing classification systems from other authors according to the classification dimensions that were defined in the paper by Bafoutsou [7]. In the literature review we chose a systematic approach, which helped us search for classifications and taxonomies of e-collaboration IT solutions. We tried to find articles that would describe the classification or type of IT solution in the context of e-collaboration. According to the number of search query results, we concluded that the field of e-collaboration is very wide.

While the narrowing relevant search results, we collected 9 articles [7] [9] [10] [3] [11] [12] [13] [14] [15] which deal with the classifications of e-collaboration IT solutions or with the approach for selecting IT support for e-collaboration.

3. TERMINOLOGY FOR THE DESCRIPTION OF E-COLLABORATION IT SOLUTIONS

The term groupware was very common in the past. It might be a bit unusual that now we are talking about e-collaboration since these two terms do not appear to be related. Mayrhofer [10] explains that the major difference between groupware and e-collaboration is that, in e-collaboration, the emphasis is on the collaborative process and not on the technology that supports collaboration [9]. Consequently, there is a stronger emphasis on the usage of web-based technologies that support this collaborative process. Mayrhofer [10] at the same time concludes that e-collaboration is more applicative and less technologically oriented.

Authors use different terms when describing e-collaboration IT solutions. Some of them write about systems [9] [11] [12] [13] [14] [7], others about tools [10] [9] [3] [11] [12] [14] [7] [15]. In both cases, the type of IT solutions is the same, where the emphasis is on the final product and usability and not on the technology according to which the product is made.

In some papers, the usage of both terms with which authors describe the same products [9] [11] [12] [14] [7] can be seen. However, in this paper we are using the term IT solutions to describe software products in the context of e-collaboration, which by the definition of Riemer [13] must support processes of communication, coordination and cooperation.

4. CLASSIFICATION SYSTEMS AND CLASSIFICATION DIMENSIONS

Mayrhofer [9] notes that rapid technological changes require new functionalities and that there is therefore a need for a new classification of technologies in support of e-collaboration.

Soriano [15] mentions different classification systems, which can also be called frameworks for e-collaboration IT solutions. They differ by needs that are met from a different point of view. At the end of his paper, Soriano [15] does not present any new type of classification but describes the result of classification for different IT solutions with the combination of different classification systems. In fact, the result of his work is a very good description of the meaning of a classification system for categorizing e-collaboration IT solutions. The same term is also used by Sarma [3] who at the end suggests that we could jointly use multiple classification systems in order to better distinguish e-collaboration IT solutions. Soriano [15] in this case suggests the combination of three systems at the same time: Grudin [16], Sarma [3] and Booch [17].

Bafoutsou [7] presents the results of eleven suggested classifications where she organizes classification systems in classification dimensions according to the literature. In her research, Bafoutsou [7] states four major dimensions: time/space, application, group issues, technical and other; and more sub-dimensions: usability and ergonomics, mode of interaction, scalability, software, hardware, types of group tasks, characteristics of group and group size. Classification dimensions by Bafoutsou [7] with the addition of new sub-dimension functionalities that were found in two articles [14] [7] are shown in Figure 1.

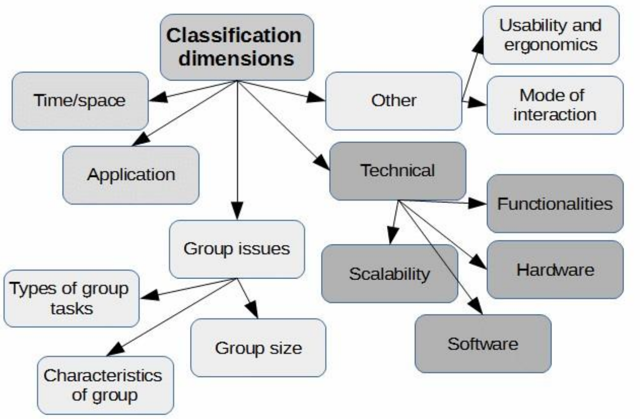


Figure 1: Classification dimensions by Bafoutsou [7] **with added sub-dimension functionalities**

4.1 Analysis

In the following chapter we will analyze classification systems which we found and allocate them to the classification dimensions which were presented by Bafoutsou in her paper [7].

In an article published in 2004, Mayrhofer [9] presents the categorization of IT solutions within the context of content management support according to the level of content management and the level of interaction within a supporting process. Based on classification dimensions by Bafoutsou [7], we can allocate this to the mode of interaction where we recognize coordination, cooperation and communication, as well as usability and ergonomics, taking into consideration the level of content management.

In a paper from 2003 [10], similar to a paper from 2004 [9] Mayrhofer presents a similar categorization. The difference is only that in this paper the background of classification, functionalities and categories of e-collaboration IT solutions are explained in more detail. The author categorizes functionalities in a pyramid shape according to the priority of implementation of e-collaboration: basic functions, fundamental e-collaboration functions, additional e-collaboration functions and add-on functions.

In her paper, Sarma [3] presents the classification in the context of software development tools. According to e-collaboration needs she distinguishes: basic, enhanced and comfort needs, as well as sets of usage of IT solutions: communication, artifact management, task management. Sets of usage of IT solutions can be recognized as types of group tasks, while e-collaboration can be recognized as usability and ergonomics.

Schauer [11] in his paper from 2011 does not introduce any new categorization but does suggest a new approach to assess the level of adequacy of IT solutions in the general context of e-collaboration. Nor does he introduce any new categorization in his paper from 2010 [12].

Riemer [13] classifies the features of IT solutions according to the typical features of e-collaboration and ways in which IT solutions will be used in the context of a group: group processes, system usage, the role of the group, types of communication, shared resources and features in awareness, as well as according to the classes of IT solutions: everyday systems, integrated systems, coordination systems, meeting systems, and specialized tools. This can be recognized as usability and ergonomics, the mode of interaction, software, and the application.

Bafoutsou [14] categorizes IT solutions for document management according to document management capability and functionality concerning collaboration. This can be recognized as dimension usability and ergonomics. Functionalities which can be seen in the sub-dimension functionalities can be recognized as new dimension functionalities, which can be part of the classification dimension: "technical."

The suggested classification in the paper by Bafoutsou [7] from 2002 is similar like to her paper from 2001 [14]. We can recognize the dimensions of usability and ergonomics, as well as technical/functionalities.

Soriano [15] does not present any new type of classification, but as a result he shows the classifications of different IT solutions by means of different categorization systems. He suggests three of them: Grudin [16], Sarma [3] and Booch [17]. Grudin [16] classifies IT solutions according to time and space, as well as according to the formality of the approach. Booch [17] uses coordination, collaboration and community building. Sarma [3] uses two similar types of a pyramidal approach. In a paper from 2004 [3], the pyramidal approach is represented by the Y axis, where the categorization of an IT solution is based on collaboration needs. The higher on the pyramid, the more comfort needs are expected in e-collaboration IT solutions. In her second paper from 2005 [18] which is mentioned by Soriano [15], it is possible to see a similar pyramidal classification. However, it focuses instead on the level of coordination. This coordination level is divided into: functional, defined, proactive, passive and seamless. However on the X axis there are always three different sets of the usage of IT solutions for e-collaboration: communication, artifact management and task management. The classification system by Grudin [16] can be allocated to time/space and mode of interaction, while the system by Sarma [18] can be allocated to types of group tasks, usability and ergonomics. The system by Booch [17] can be recognized by the mode of interaction.

4.2 Classification dimensions taxonomy

According to the literature review, there is a difference between classification systems and classification dimensions. We used classification dimensions by Bafoutsou [7] to which we added sub-dimension functionalities. In some classification systems we can see that authors classify functionalities, while in papers by Bafoutsou from 2001 [14] and 2002 [7] we can see functionalities as a classification dimension and not as the subject of classification.

The classification dimensions that we found in the literature are: time/space [15], application [13], group issues; types of group tasks [3] [15], technical; software [13], technical; functionalities [14] [7], other; usability and ergonomics [9] [10] [3] [13] [14] [7] [15], other; and mode of interaction [10] [9] [13] [15].

Classification dimensions by Bafoutsou [7] which we could not recognize in the literature and also have not used are: characteristics of the group, group size, scalability, hardware. We can notice that we recognized usability and ergonomics in all classification systems. The reason is that we can look at usability and ergonomics from different points of view.

Table 2: Classification dimensions recognized in the literature

Dimension	Sub-dimension	Author						
		Mayrhofer et al. [9]	Mayrhofer et al. [10]	Sarma et al. [3]	Riemer [13]	Bafoutsou et al. [14]	Bafoutsou et al. [7]	Soriano et al. [15]
Time / space								X
Application					X			
Group issues	Types of group tasks			X				X
Technical	Software				X			
	Functionalities					X	X	
Other	Usability and ergonomics	X	X	X	X	X	X	X
	Mode of interaction	X	X		X			X

4.3 Purpose of classification systems

We realized that some classification systems have a specific purpose according to their context of use. Based on the literature we can distinctly find four intended uses of classification systems:

general use [13] [15], content management [10] [9], software development [3] and document management [14] [7].

In the review of classification systems we discovered that most authors [10] [9] [3] [15] focus on the categorization of the functionalities of IT solutions, while others [13] [14] [7] focus on the classification of products that represent IT solutions.

5. E-COLLABORATION IT SOLUTIONS

A sample set of e-collaboration IT solutions was recognized only by some authors. Unfortunately, in this research paper we cannot give a complete list of IT solutions that we found in the literature, due to space limitations. However, we can show a number of IT solutions that we recognized with individual authors: Mayrhofer et al. [10] (55 IT solutions), Schauer et al. [11] (10 IT solutions), Schauer et al. [12] (5 IT solutions), Riemer [13] (94 IT solutions), Bafoutsou et al. [14] (44 IT solutions), Bafoutsou et al. [7] (47 IT solutions).

Generally, e-collaboration IT solutions are very different across research papers. Many of them are not well recognized commercially, since they were developed by lesser known software companies, for example: CommonSpace, DocuTouch, TeamTalk, eRoom, HyperOffice, PlaceWare Conference Center, EW Meeting etc. But there are also IT solutions from well-known software companies: Microsoft Netmeeting, Microsoft SharePoint Server, Skype, phpBB, Google Calendar, Oracle Collaboration Suite, Novell GroupWise, Microsoft Outlook, Lotus Notes, OpenGroupware.org, etc.

Those IT solutions share the similarity that they are all classified as e-collaboration. For example Skype, Google Calendar and phpBB have nothing in common from the technological point of view, but we can jointly use them in e-collaboration. It was also the purpose of these research papers to categorize IT solutions in such a way that the end user could more easily select appropriate e-collaboration IT solutions.

6. CONCLUSIONS

We discovered that classifications differ regarding classification dimensions (time/space, application, group issues, technical and other; and more sub-dimensions), the purpose of classification systems (general use, content management, software development, document management) and the type of classification that is managed by the classification system (classification of functionalities and classification of IT solutions).

There is also the possibility of using different classification dimensions to build a new classification system, which could be used in a completely new way in e-collaboration. Of course, first we need to know for what purpose we would need this new classification system and what exactly we want to classify (functionalities or IT solutions).

7. REFERENCES

[1] P. Watson, *Ideas: A history of thought and invention, from fire to Freud*. HarperCollins, 2005.

[2] A. Pinsonneault and K. L. Kraemer, “The impact of technological support on groups: An assessment of the empirical research,” *Decision Support Systems*, vol. 5. Elsevier Ltd and IPMA, pp. 197–216, 1989.

[3] A. Sarma, A. Van Der Hoek, L. Cheng, and R. Street, “A Need-Based Collaboration Classification Framework 1 Introduction,” vol. Workshop a, p. 5, 2004.

[4] “Encyclopedia Britannica.” [Online]. Available: http://www.britannica.com/EBchecked/topic/584695/taxonomy. [Accessed: 05-Jan-2015].

[5] “Slovar Informatike.” [Online]. Available: http://www.islovar.org/. [Accessed: 05-Jan-2015].

[6] “Wiktionary: Taxonomy,” 2015. [Online]. Available: http://en.wiktionary.org/wiki/taxonomy.

[7] G. Bafoutsou and G. Mentzas, “Review and functional classification of collaborative systems,” *Int. J. Inf. Manage.*, vol. 22, no. 4, pp. 281–305, Aug. 2002.

[8] E. K. Jacob, “Classification and Categorization : A Difference that Makes a Difference,” vol. 52, no. 3, pp. 515–540, 2004.

[9] D. Mayrhofer and A. Back, “Classification of IS Functionalities for Workplace E-Collaboration,” *IRMA Int. Conf.*, pp. 208–213, 2004.

[10] D. Mayrhofer, “Workplace E-Collaboration Theoretical Foundations and Practical Implications Mayrhofer, David; Back, Andrea.” Institut für Wirtschaftsinformatik, pp. 1–39, 2003.

[11] B. Schauer and M. Zeiller, “E-Collaboration Systems : How Collaborative They Really Are Analysis of Collaboration Features of Electronic Collaboration Systems,” *COLLA 2011 First Int. Conf. Adv. Collab. Networks, Syst. Appl.*, p. 6, 2011.

[12] B. Schauer, M. Zeiller, and D. Riedl, “REVIEWING THE E-COLLABORATION MARKETPLACE A Survey of Electronic Collaboration Systems,” *E-bus. (ICE-B), Proc. 2010 Int. Conf.*, p. 7, 2010.

[13] K. Riemer, “THE MARKET FOR E-COLLABORATION SYSTEMS - IDENTIFICATION OF SYSTEM CLASSES,” vol. Volume 5, no. Issue 3, pp. 346–357, 2009.

[14] G. Bafoutsou and G. Mentzas, “A comparative analysis of web-based collaborative systems,” *12th Int. Work. Database Expert Syst. Appl. DEXA-01*, pp. 496–500, 2001.

[15] J. Soriano, “Characterization and Classification of Collaborative Tools,” p. 8, 2009.

[16] J. Grudin, “CSCW: History and Focus,” *IEEE Comput.*, vol. 27, pp. 19–26, 1994.

[17] G. Booch and A. W. Brown, “Collaborative Development Environments,” *Advances in Computers*, vol. 59. Elsevier Ltd and IPMA, pp. 1–27, 2003.

[18] A. Sarma, “A Survey of Collaborative Tools in Software Development,” 2005.

Participation of persons with disabilities in collaborative work and the community

Tatjana Welzer
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
tatjana.welzer@um.si

Lenka Lhotska
Czech Technical University in Prague,
Gerstner Laboratory
Prague, Czech Republic
lhotska@fel.cvut.cz

Ian Grout
University of Limerick,
Limerick, Ireland
Ian.Grout@ul.ie

Tony Ward
Department of Electronics, University of York
York, UK
tony.ward@york.ac.uk

Marjan Družovec
Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia
marjan.druzovec@um.si

ABSTRACT

In our contribution we would like to introduce the importance of including people with disabilities into the community as well as in collaborative work. Their experiences are very valuable, especially when it comes to producing software, applications and solutions in general, as well as supporting persons with disabilities. A lot of research work has been done on the topic, but not so much in the area of Electrical and Information Engineering, where we have to take a lot of care especially with regard to the practical inclusion in laboratory work and research. Last, but not least, we also have to be aware that smart devices are a significant benefit for most users but could also be a problem for those who cannot, for example, use their fingers or have limited hand movements. On the other hand, smart devices, as a part of assistive and/or adaptive technology can also support users with disabilities and make their lives easier and provide more opportunities for them.

Categories and Subject Descriptors

H.1.2 [**Information Systems**]: User/Machine Systems - *Human factors.*

General Terms

Management, Development, Human Factors.

Keywords

Disability; special needs; collaborative work; SALEIE; assistive technologies; adaptive technology

1. INTRODUCTION

Information and communication technology is marked by successful collaboration and partners in the process still play an important role. It seems that partners do not need to know each other personally, but it is enough if they have good technical support in computerized collaborative tools and software. This also means that collaboration (cooperation) is not bound by geographical location and can take place all around the world. But if we examine this situation more closely, we can encounter situations where it is very important to know your partners personally, in order to understand his/her decisions, approaches, motivations, and needs during collaboration. Namely, participants in collaborative work are not only from different topics, companies, research works, departments and cultural environments, they can also be persons with disabilities or special needs. When the results of collaborative work support the work and activities of persons with disabilities, their involvement is very valuable, especially if they are a developer as well as a user.

In this paper, we will primarily concentrate on persons with disabilities (researchers, employees, students) and awareness of their participation and collaboration including the support of legal issues and technologies.

The presented results were collected within the framework of the EU-funded project (SALEIE - Strategic Alignment of Electrical and Information Engineering in European Higher Education Institutions) with respect to requirements posed by Electrical and Information Engineering, as is evident from the title.

Additionally, we would like to point out that there are also connections to the issue with regard to the law, formal definitions and explanations of terms connected to the topic.

The present paper is organised as follows: after the introduction, we will introduce the SALEIE project. This will be followed by an initial presentation of basic legal issues, assistive and adaptive technology, while the main part will be connected to some examples that demonstrate the participation of persons with disabilities in the community and collaborative work.

In the conclusion, observations and suggestions for further work will be given with the goal of pointing out the awareness of including persons with disabilities into different systems including studies. Stress will be on recommendations that should be done.

2. SALEIE

The SALEIE consortium consists of 45 European project partners. The project explores and provides models for ways in which Higher Education Institutions in Europe, and specifically in the Electrical and Information Engineering disciplines, can respond to current challenges and introduce results to a broader community. The project is under the patronage of the EAEEIE (European Association for Education in Electrical and Information Engineering), a European non-profit organization, with members from nearly seventy European Universities, most of them teaching in the area of Electrical and Information Engineering (EIE).

The main challenges addressed by the SALEIE project are [3]:

- Ensuring that graduates are/will be prepared to enable Europe to respond to the current global technical challenges in Green Energy, Environment and Sustainability, Communications and IT, Health, and Modern Manufacturing Systems, in the sense of “new skills for new jobs” [1]. This will embrace conventional education, lifelong learning and training for entrepreneurship. This research is included in work package 3 (WP3) from the SALEIE project [3].
- Ensuring that the programme and module governance is sufficiently well understood so that issues of mobility, progression and employment are understandable by appropriate stakeholders including accrediting bodies for professional engineers [1]. This is included in work package 5 (WP5) [3].
- Ensuring all learners, irrespective of their background or personal challenges, including: dyslexia and dyspraxia; visual and audio impairments; and mental disabilities such as Asperger's, autism, depression, anxiety; are given equal opportunity to education and are appropriately supported [2]. These are the main research goals of the SALEAIE in work package 4 (WP4) [3].

Further on in the paper we will concentrate on the research and achievements from the WP4.

The main activities of the WP 4 also include: a survey of the project partners of the scale of diversity of widening participation practices across Europe, a collection of examples of best practices in supporting specific needs, the design and development of best practice support models for different types of specific needs, a collection of examples of best practices in marketing programs for students with specific needs, a review of how HEI's industrial partners view support for learners, the design of a web-based center for excellence for the support of learning in engineering across Europe - Student and staff support hub, and the translation of best practice examples into a range of European languages (including Slovenian, among others) [1], [3].

With regard to the numbered activities, collaboration and inclusion into the community, in general, at the academic level, and employability we will present some legal background for Slovenia, that formally supports and enables the mentioned activities and research steps.

3. LEGAL ISSUES, ASSISTIVE AND ADAPTIVE TECHNOLOGIES

In the Republic of Slovenia, in accordance with Article 56 of the Constitution, all citizens must be guaranteed equal rights and opportunities for education and training. The education of an individual is important for their development, participation and entry into the labour market and society [8]. Based on these facts, persons with disabilities should have equal opportunities to participate in education systems and programmes in Slovenia, either regular or special ones, on all levels of education: from preschool to a university education [8]. To fulfil these expectations, different legal documents have been adopted on a national level. Among the more important is the Action Programme for Persons with Disabilities (APPD), and specifically Chapter 3: Action programme objectives, Point 4, Objective: Education [4].

With the adoption of the APPD for 2014–2021 in January 2014[4], Slovenia received a document that represents the continuation of work previously defined in the document APPD for 2007-2013 [5]. This document was developed as a national strategy to implement the UN Convention on the Rights of Persons with Disabilities (CRPD) [9] and was ratified in the Slovenian National Assembly on April 2, 2008 [10].

While the national focal point for CPRD in Slovenia is the Ministry of Labour, Family and Social Affairs (Disabled, Veterans and War Victims Directorate, Social Affairs Directorate, Labour Market and Employment Directorate), for APPD additional responsibilities also go to the Ministry of Culture (Directorates and Services and Public cultural institutions), Ministry of Education, Science and Sport (Directorates and Services, National Education Institute, Higher Education Institutions), Association of Slovenian Training Organizations for Persons with Special Needs (Public institutions and concessionaires) and National Council of Disabled People's Organizations of Slovenia [4].

During education and also in everyday life as well as later as employed persons, people with disabilities encounter various obstacles like architectural barriers, technical obstacles such as difficulties relating to accessibility to studying literature, sign language interpreters for the deaf, induction loops for the hard of hearing, lack of information, etc. Therefore, ensuring measures to remove such obstacles are necessary. We will present some that are important for education and in a broader sense [4]:

- Equal opportunities in enrolment in all educational programmes and encouraging the increased inclusion of persons with disabilities in regular programmes of education and training.
- A network of expert institutions providing support for the education of persons with disabilities in specialized and regular programmes of education and training.
- Equal opportunities for persons with moderate, severe or serious developmental disorders to be included at all levels of a Special Programme of Education and Training, including training for life and work.
- Practical training, apprenticeships and practice within secondary schools and university study with the active participation of different employers.
- Developing and supporting life-long learning for unemployed and employed adults with disabilities.
- Ensuring text books and other teaching aids in suitable forms and that consider the type of disability.
- Ensuring spatial and technical conditions for carrying out education training and other activities with adjusted transportation.
- Provision of adequate equipment for individual use in the education and working process.

- Additional communication equipment that is necessary, owing to functional impairment.
- Physical assistance to all school participants and students who need such assistance.
- Support during higher education by means of personal assistance, accompanying and counselling for students with disabilities, the provision of suitable accommodation facilities for students with disabilities and the provision of a scholarship scheme encouraging the attainment of higher education.

Besides legal issues an important part of needed knowledge by inclusion persons with disabilities into collaborative work are connected to assistive and adaptive technologies. Assistive technology is a general term that includes assistive, adaptive, and rehabilitative technologies and devices for persons with disabilities. Assistive technology promotes and support greater independence by enabling persons with disabilities to perform tasks that they were unable to accomplish, or would had great difficulty to accomplish tasks, that are supported by technology [7].

Typical assistive technologies are used by users in everyday life. Therefore, they must satisfy requirements like intuitive control and user interface adjustable according to the handicap and needs if we mention just one example. A very illustrative example is also touch screen and person with hand tremor or problems with fine hand motor control. To support persons with those disabilities we have to develop an interface easily controllable by a person with such problems. Possible solutions can be several, some users will be satisfied with larger icons, some will prefer to use a touch pen with fixation, some would welcome hardware keyboard, or voice control can become a good option as well [7].

From those examples we can conclude that a lot of possibilities for cooperative work exists, where persons with disabilities can be only adviser and users or they are involved in the development while working on such applications and tools.

4. SALEIE EXAMPLES – PERSONS WITH DISABILITIES IN COLLABORATIVE WORK AND COMMUNITY

A list of possible examples that we collected in the frame of the SALEIE project is long, but we will briefly present two with regard to the university environment [3]. The second comes from the University of Maribor, where we have to deal with more practical experiences and the first one from the University of Ljubljana, where we have to do with formally prescribed procedures and guidelines with regard to basic definitions like students with disabilities and students with special needs.

The Statute of the University of Ljubljana recognises three groups of students with special needs: student athletes, student artists, and students with disabilities [11]. All three groups are equal in their rights and students with disabilities at the level of the University of Ljubljana have no additional rights. The status is granted after the successful application of the student.

Additionally, each faculty member of the University of Ljubljana has rights that are guaranteed by internal documents in accordance with the Statute of the University of Ljubljana, which additionally regulate the area of students with special needs.

While the University of Ljubljana has almost 30 faculties, schools and academies, we will only point to one of them as an example of additional regulations and good practices.

The Faculty of Administration is one of those faculties. They divide students who are eligible for special status into two groups. The first group includes athletes or artists or cultural workers who have achieved visible results in their respective field of activity. The second group includes students with disabilities or long-term injuries, disorders, deficits or problems that make their full and efficient integration in the execution of the study courses difficult [12].

Students from all three groups need to apply for the status by applying directly to the faculty.

Students with disabilities at the Faculty of Administration receive special additional attention. They are asked to get into personal contact, so that their problems can be worked out personally. The status of a student with disabilities is granted while bearing in mind the type of problems and the condition of the student's status, whether it be part-time or for the duration of their studies. On the basis of the personal interaction, the faculty can become familiarised with the situation of each student. Each student is different as an individual, what means that the provided options and solutions are not the only right ones. The best way to cooperate with a student with disabilities is to talk about the needs of the individual honestly and directly. The talk must be carried out in private so as not to stigmatise the disabled student, after which a common strategy discussed with the student, can provide the best results [12].

In order to achieve optimal results, the faculty has also organised a student tutorship for students with disabilities. A tutor for students with disabilities directs the students to develop the abilities they need to enable them to adapt quickly and successfully into their academic work and student life. The tutor provides assistance to students in the study activities that students with disabilities cannot perform unassisted [12].

At the University of Maribor's Faculty of Electrical Engineering and Computer Science we were faced with a problem that has to be fixed "overnight" without any experience of working with persons with disabilities and basic formal descriptions that did not provide solutions.

From one day to the next, one of the students fell ill and was no longer present in their classes. After a longer period, the student reached out to teachers and the student office, asking for help to organise lectures, meetings and exams for places where architectural barriers were not present. An additional barrier was also timing, because the student was able to organize their transportation only during a limited time, which was mostly outside of any schedule. Additionally, the student was physically unable to take part in activities at the same time as other students or even with agreed-upon regularity. The student might not appear for different reasons – schedule changes, transportation problems or health problems. Taking into account experiences with other students with disabilities, it seemed that nothing was really working and in the first period not only a personalized approach was needed but rather an ad lib personalized approach.

Most of the communication was done with the help of e-mails in direct communication with the involved teachers, with the goal of avoiding noise and misunderstandings in communication as well as to be as flexible as possible. Slowly but surely, activities started to work. The benefit of that case was the fact that the student was in the last year of their bachelor's studies and most laboratory activities were already finished before the illness of the student. In the rest of cases, Information Technology support helped the student to finalize their studies and graduate successfully.

5. CONCLUSION

After an initial presentation of the area and experiences collected in different environments, the logical question is what we should do as a next step? A lot of work has to be done and must be done, while different countries and different environments are at different level of inclusion with regard to persons with disabilities in collaborative work and community. Through research-based questionnaires and best practices our short list of recommendations, which is as general as possible and appropriate for different systems, is the following:

- Dedicated support offices.
- Staff training: Training programmes for staff to be set-up and run by the disability support office. This is to be linked to the individuals' continuing professional development.
- Institutional level information systems: Each institution should have a clear, unambiguous and useful information support system (ISS) for persons from under-represented groups. This should be set-up by the institution in accessible formats and periodically reviewed.
- Infrastructure: Laboratory and work facilities. When laboratories and work facilities are either created or updated, due care should be given to accessibility, irrespective of whether at that time persons who would require additional supports to fully engage with the laboratory and work facilities are present.
- Laboratories: Training where the technical staff of laboratories should be trained to support persons with disabilities so that they can fully engage with their laboratory work.
- Legislation awareness: All staff should be made aware of their roles and responsibilities under national legislation and institutional level regulations.

We expect that the above recommendations should be helpful and would enable further inclusion of persons with disabilities into collaborative work and the community, but the list is not final and merely constitutes the groundwork for further research and conclusions in the form of additional recommendations and changes in real environments.

6. ACKNOWLEDGMENTS

The presented results were financially supported by the EU-EACEA in the framework of the Lifelong Learning Programme, under the SALEIE grant no. 527877-LLP-1-2012-1-UK-ERASMUS-ENW entitled: Strategic Alignment of Electrical and Information Engineering in European Higher Education Institutions.

7. REFERENCES

[1] A. Frisel, T. Welzer, Z. Mrozek, A.E. Ward., M. Poboroniuc, Building a shared understanding of the skills and competences in order to respond to the current global technical challenges. V: IEEE Global Engineering Education Conference - EDUCON 2014, Istanbul, Turkey, 3-5 April 2014. EDUCON 2014. Piscataway: Institute of Electrical and Electronics Engineers: IEEE, 2014, pp. 676-679.

[2] F. Demontoux, H.Fremont, E. Woirgard, 2013, Criteria of the EIE courses accessible to disabled students. Legibility of cursus and experiences sharing for generalizing good practices, IEEE Proceedings of the 24th EAEEIE, 2013, pp.7-10

[3] SALEIE http://www.saleie.co.uk/, last visited September 25, 2015

[4] Action Programme for Persons with Disabilities 2007–2013, Ministry of Labour, Family and Social Affairs, 2007, [Online]. last visited September 25, 2015 http://www.mddsz.gov.si/fileadmin/mddsz.gov.si/pageuploads/dokumenti__pdf/api_07_13_en.pdf

[5] T. Welzer, V. Kožuh-Ledinek: Students with disabilities at the UM-Faculty of electrical engineering and computer science. In: B. Karaoglan (ed.). Proceedings of the 25th International Conference on European Association for Education in Electrical and Information Engineering: 30 May - 1 June 2014, Çeşme, Izmir, Türkiye. [S. l.]: IEEE, 2014, pp. 1-4.

[6] L. Lhotska, T. Welzer: Assistive technologies in biomedical engineering education. In: D.A. Jaffray, (ed.). World Congress on Medical Physics and Biomedical Engineering, June 7-12, 2015, Toronto, Canada: IFMBE proceedings, (IFMBE proceedings, ISSN 1680-0737, 51). Springer International Publishing, 2015, pp. 1656-1659.

[7] M. Družovec, M. Hölbl, V. Kožuh Ledinek, T.Welzer: Case study of a student with disability - personalised approach. In:ITHET 2014 Proceedings, IEEE, 2014, pp. 1-5.

[8] I. Grout, L. Grindei, T. Ward, C. Busk Marner, T. Welzer: Widening participation in EIE programmes across Europe for students with disabilities. In:ITHET 2015 Proceedings, IEEE, 2015, pp. 1-6

[9] Convention on the rights of persons with disabilities, [Online]. last visited September 25, 2015: http://www.un.org/disabilities/convention/conventionfull.shtml

[10] Sixth disability high level group report on the implementation of the UN Convention on the Rights of Persons with Disabilities, [Online]. last visited September 25, 2015: http://ec.europa.eu/justice/discrimination/files/dhlg_6th_report_en.pdf

[11] Statut Univerze v Ljubljani, http://www.uni-in-obrazci/pravilaposebnistatus.pdf , 2005, in Slovenian, last visited September 25, 2015.

[12] Special Status students, Faculty of Administration Ljubljana, http://www.fu.uni-lj.si/en/students/special-status-students/ , last visited September 25, 2015.

Indeks avtorjev / Author index

Arumugam Gurusamy 21
Budimac Zoran 17
Družovec Marjan 45
Fijačko Nino 29
Fluks Zmago 37
Galinac Grbac Tihana 9
Grbac Babić Sanja 9
Grout Ian 45
Heričko Marjan 17, 21, 37
Jošt Gregor 25
Karakatič Sašo 5, 21
Lhotska Lenka 45
Malačič Robert 13
Muthuraman Thangaraj 21
Ovčjak Boris 25
Pavlič Luka 21
Podgorelec Vili 5
Polančič Gregor 25
Povalej Bržan Petra 29
Pušnik Maja 17
Rakić Gordana 17
Rupnik Rok 33
Štiglic Gregor 29
Stropnik Ambrož 41
Vuherer Tomaž 41
Ward Tony 45
Welzer Tatjana 45
Zorman Milan 41

Konferenca / Conference

Uredili / Edited by

Sodelovanje, programska oprema in storitve v informacijski družbi / Collaboration, Software and Services in Information Society

Marjan Heričko