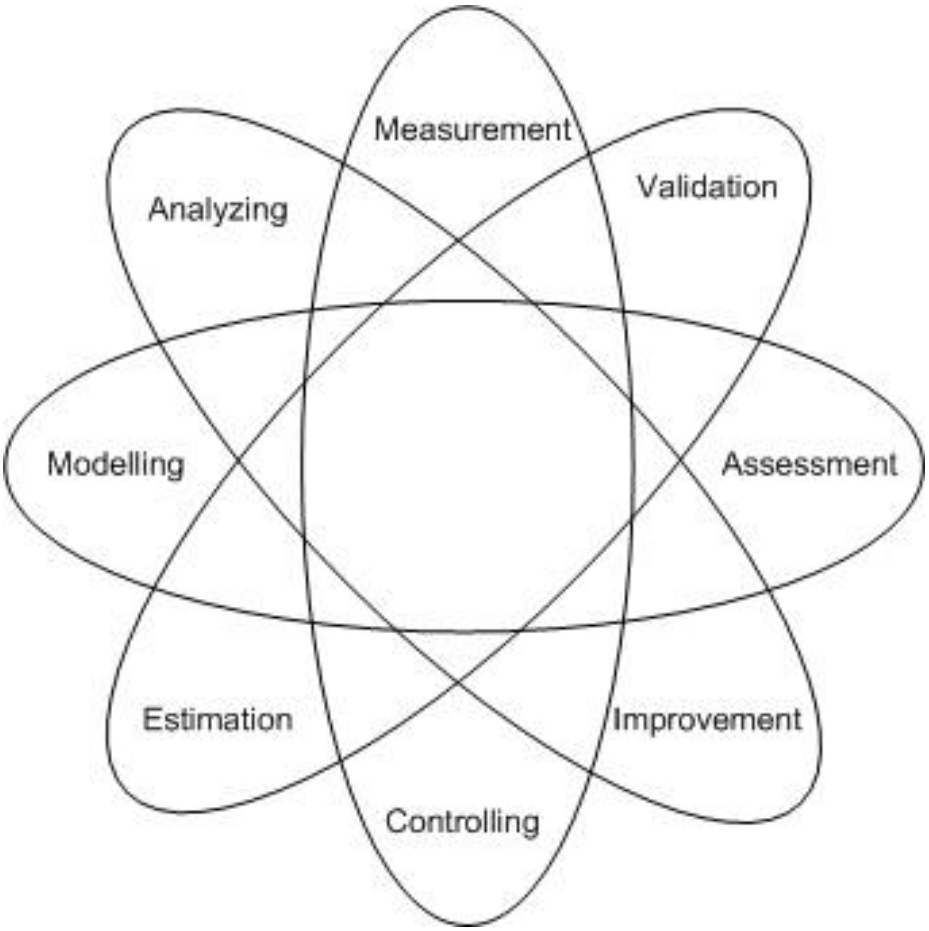
Volume 26, Number 1, March 2021

Software Measurement News

Journal of the Software Measurement Community



Editors:

Alain Abran, Manfred Seufert, Reiner Dumke, Christof Ebert, Cornelius Wille

CONTENTS

Editors:

Alain Abran
Professor and Director of the Research Lab. in Software Engineering Management
École de Technologie Supérieure - ETS, 1100 Notre-Dame Quest,Montréal, Quebec, H3C 1K3, Canada, **alain.abran@etsmtl.ca**

Manfred Seufert
Chair of the DASMA, Median ABS Deutschland GmbH
Franz-Rennefeld-Weg 2, D-40472 Düsseldorf,
manfred.seufert@mediaan.com

Reiner Dumke
Professor on Software Engineering, University of Magdeburg, FIN/IKS
Postfach 4120, D-39016 Magdeburg, Germany,
dumke@ivs.cs.uni-magdeburg.de, http://www.smlab.de

Christof Ebert
Dr.-Ing. in Computer Science, Vector Consulting Services GmbH
Ingersheimer Str. 20, D-70499 Stuttgart, Germany,
christof.ebert@vector.com

Cornelius Wille
Professor on Software Engineering, University of Applied Sciences Bingen
Berlinstr. 109, D-55411 Bingen am Rhein, Germany,
wille@fh-bingen.de

Editorial Office: University of Magdeburg, FIN, Postfach 4120, 39016 Magdeburg, Germany
Technical Editor: Dagmar Dörge
The journal is published in one volume per year consisting of two numbers.

Students Challenge Overview

Alain Abran, ETS Montreal, Canada

see last year:

Saturday March 27th 2021, 9:00 to 12:00

(Montreal-New-York time zone)

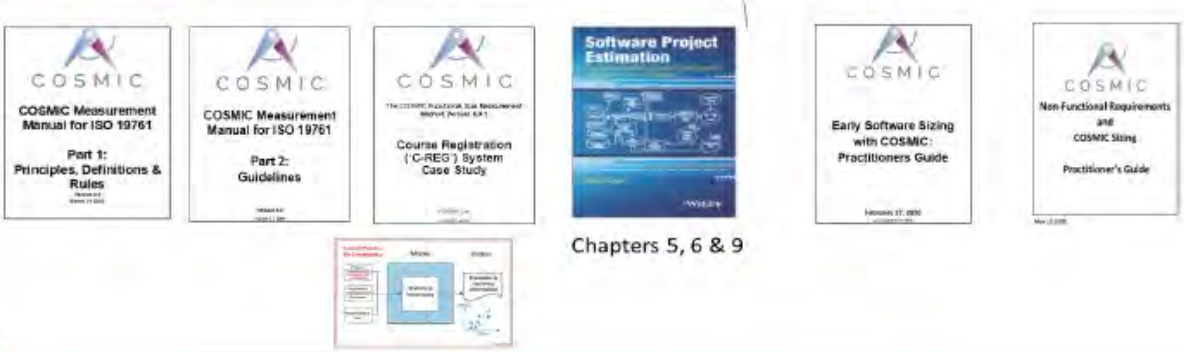
Challenge rewards:

- ✓ 1st prize: 500 Euros
- ✓ 2nd prize: 300 Euros
- ✓ 3rd prize: 200 Euros

Challenge Inputs

1. A sub-set of detailed functional requirements.
2. A sub-set of functional requirements at unspecified levels of completeness.
3. Some non-functional requirements.
4. Description of a development environment.
5. A data set with historical data on development productivity.

Useful resources



Challenge tasks

Using the Requirements document the teams have to:

1. Size with COSMIC the detailed functional requirements.
2. Approximate the size of the functional requirements.
3. Size the non-functional requirements allocated to software functions.
4. Develop an estimation model from the historical data.
5. Estimate the effort to develop both functional & non-functional requirements.

see: https://profs.etsmtl.ca/aabran/English/index.html

Announcement of the GI-FG 2.1.10

Gesellschaft für Informatik e.V. (GI) und Plattform Lernende Systeme (PLS)

Data Science: Industrieerfahrung und Praxistipps

Virtueller Workshop am 20.5.2021, 14-17 Uhr

Zunehmend mehr Daten sind im beruflichen und privaten Umfeld verfügbar und laden zu neuen Geschäftsmodellen ein. Big Data und Künstliche Intelligenz versprechen vollkommen neue Produkte und Lösungen: vom autonomen Fahren bis hin zu Industrie 4.0. Die Potenziale, mit Daten direkt Geld zu verdienen oder basierend auf Daten intelligente Dienste und Produkte aufzubauen, scheinen unendlich.

Dazu werden Kompetenzen benötigt, um konkrete datenbasierte Use Cases umzusetzen. Doch viele Unternehmen erreichen Ihre Ziele nicht oder nur sehr langsam:

- Unklare Use Cases und Geschäftsmodelle
- Zu wenige oder qualitativ unzureichende Daten
- Unklar, was in welcher Form gespeichert and analysiert werden soll
- Die KI hält in der Praxis nicht, was zuvor versprochen wurde
- Mangelnde Verfügbarkeit von Data Scientists und Software-Ingenieuren

Sie sind eingeladen, im Rahmen eines virtuellen Workshops mit Anwendern aus Wirtschaft und Wissenschaft die folgenden Fragestellungen praxisorientiert zu diskutieren:

- Wie findet man geeignete Use Cases und woher bekommt man die Daten?
- Worin liegt der Wert von Daten und wie kann man diesen greifbar machen?
- Wie erreicht man ein nachhaltiges Geschäftsmodell?

Nutzen Sie den Workshop, um ...

- *…Impulse zum Thema Data Science aus erster Hand zu bekommen*
- *…Ihr Netzwerk zu erweitern und sich mit Gleichgesinnten auszutauschen*
- *…Lösungen für Ihre aktuellen Fragen und Herausforderungen zu erhalten*

Die Ergebnisse des Workshops sind die Basis für ein gemeinsames Positionspapier.

Zielgruppe

Der Workshop richtet sich an alle, die Data Science im Unternehmen umsetzen und Erfahrungen dazu austauschen möchten. Wir wollen dabei insbesondere auf die Belange von KMU eingehen.

Anmeldung

Der Workshop ist kostenlos.
Bitte vorherige Anmeldung an **Jens.Heidrich@iese.fraunhofer.de** bzw. **Christof.Ebert@vector.com**

Agenda

14:00 Begrüßung: Christof Ebert (Vector & GI), Maximilian Hösl (PLS)
Moderation: Jens Heidrich (Fraunhofer IESE & GI)
14:10 Impulsvorträge mit Fragerunde
- Matthias Patz (DB Systel, VP Innovation & New Ventures)
- Michael Weyrich (Universität Stuttgart, Direktor IAS, und robo-test)
- Alexander Löser (Beuth Hochschule für Technik Berlin, Leiter FZ Data Science)
- Julien Siebert (Fraunhofer IESE, Data Scientist)

15:15 Pause
15:30 **World-Café** (Diskussionsrunde zu den Fragestellungen an drei virtuellen Tischen)
16:30 Zusammenfassung und Abschluss
17:00 Ende der Veranstaltung

Siehe auch: https://fg-metriken.gi.de

30th IWSM/Mensura 2020

was held at October 29 – 30 in Mexico City

Proceedings are available online

Title of Papers	Paper	Presentation
Ahmed Darwish and Hassan Soubra. COSMIC Functional Size of ARM assembly programs	Download	Download
Hassan Soubra, Yomna Abufrikha and Alain Abran. Towards Universal COSMIC Size Measurement Automation	Download	Download
Wilder Perdomo-Charry, Julia Prior and John Leaney. How do Colombian software companies evaluate software product quality?	Download	Download
Nebi Yilmaz and Ayça Kolukısa Tarhan. Meta-models for Software Quality and Its Evaluation: A Systematic Literature Review	Download	Download
Tuna Hacaloglu, Hüseyin Ünlü, Onur Demirors and Alain Abran. Using COSMIC Light instead of COSMIC Functional Size Measurement Manual: An Application on Multiple Cases	Download	Download
Philipp Haindl and Reinhold Plösch. Specifying Feature-Dependent Maintainability Requirements in an Operational Manner – Results From a Case Study with Practitioners	Download	Download

Luigi Lavazza, Liu Geng and Roberto Meli. Productivity of Software Enhancement Projects: an Empirical Study	Download	Download
Sara Elmidaoui, Laila Cheikhi, Ali Idri and Alain Abran. Predicting Software Maintainability using Ensemble Techniques and Stacked Generalization	Download	Download
Francisco Valdés-Souto and Jorge Valeriano-Assem. Exploratory Study: Simulating the Productivity Control in Software Projects using Feedback Loop Control Theory	Download	Download
Francisco Valdés-Souto, Roberto Pedraza-Coello and Fabiola Cristina Olguín-Barrón. COSMIC Sizing of RPA Software: A Case Study from a Proof of Concept Implementation in a Banking Organization	Download	Download
Francisco Valdés-Souto, Daniel Torres-Robledo and Hanna Jadwiga-Oktaba. Product Delivery Improvement in a Software Factory Contract Applying Learning Curves	Download	Download
Olga Ormandjieva, Mandana Omidbakhsh and Sylvie Trudel. Measuring the 3V's of Big Data: A Rigorous Approach	Download	Download
Duygu Deniz Erhan, Ayça Kolukısa Tarhan and Rana Özakıncı. Selecting Suitable Software Effort Estimation Method	Download	Download

Hela Hakim, Asma Sellami, Hanene Ben Abdallah and Alain Abran. Improving the Structural Size Measurement Method Through the Nested (Multi-Level) Control Structures Assessment in UML Sequence Diagram	Download	Download
Özden Özcan Top, Onur Demirors and Fergal Mc Caffery. Challenges and Working Solutions in Agile Adaptation: Experiences from the Industry	Download	Download
Harold van Heeringen. Portfolio Cost Estimation and Performance measurement in the context of a SAFe Scaled Agile Framework	Download	Download
Abdelaziz Sahab and Sylvie Trudel. COSMIC Functional Size Automation of Java Web Applications Using the Spring MVC Framework	Download	Download
Sylvie Trudel and Olga Ormandjieva. Lean Measurement: A Proposed Approach	Download	Download

ABOUT IWSM MENSURA

The **IWSM Mensura** conference is the result of the joining of forces of the *International Workshop on Software Measurement* and the *International Conference on Software Process and Product Measurement*. Together they form the conference where new ideas from the world of academic research meet practical improvements from industry on topics of measuring software.

Each year practitioners and researchers from all over the world gather together to learn about new developments, test new ideas and exchange possible new solutions and applications. *more*

If you like the content, please share it with your network on Twitter, Facebook or LinkedIn using #IWSM2019.

IN 2020 WE WILL BE IN

MÉXICO

see: https://www.iwsm-mensura.org/

Workshop
"Evaluation of Service-APIs – ESAPI 2020"
Motto: APIs als Klebstoff einer umfassenden Digitalisierung

November 2020, Berlin

Sandro Hartenstein[2], Konrad Nadobny[1, 2, 3,]
Steven Schmidt[2, 4,] Andreas Schmietendorf[1, 2]
[1]OvG-Universität Magdeburg, [2]HWR Berlin,
[3]Bayer AG, [4]Deutsche Bahn AG

1. Motivation und Themen des Workshops

Die Gartner Group[1] geht davon aus, dass im Jahr 2021 mehr als 60% aller Anwendungsentwicklungen von eingesetzten Web-APIs profitieren. Diese mit Hilfe klassischer Internettechnologien zur Verfügung gestellten Web-APIs bieten die Möglichkeit eines konsistenten Zugriffs auf fachlich begründete Informationen und Funktionen aber auch auf komplette Geschäftsprozesse.

Neben einer unternehmens- und branchenübergreifenden Integration existierender Softwarelösungen wird dabei auch die Zielstellung einer kompositorischen und damit agilen Softwareentwicklung verfolgt. Aufgrund der ggf. „ad hoc" zusammengesetzten Lösungen muss auch der Betrieb mit diesen Herausforderungen umgehen können. Daher kommt der Themenstellung „DevOps" als Klammer zwischen Entwicklung und Betrieb eine besondere Bedeutung zu.

Der ESAPI-Workshop im Jahr 2020 fokussierte die folgenden Themen:

- Bewertung von Vertrauen und Sicherheit bei Web-APIs.
- Branchenspezifische Ansätze zur Spezifikation von Web-APIs.
- Lowcode bzw. Codeless Softwareentwicklung mit Web-APIs.
- Effiziente Ansätze zur „API-fizierung" von Altanwendungen.
- Risiken bei über Web-APIs bezogenen KI-Algorithmen.
- Vor- und Nachteile von GraphQL-basierten Web-APIs.
- Elemente eines DevOps-orientierten API-Managements.
- Serverless bereitgestellte Web-APIs – Fiktion oder Wirklichkeit?

Der ursprünglich als Präsenzveranstaltung geplante Workshop wurde im Jahr 2020 erstmals online durchgeführt. Mehr als 40 Teilnehmer hatten sich im Rahmen der

[1] Quelle: Zumerle, D. et al. 2019. API Security. What You Need to Do to Protect Your APIs [online]. Verfügbar unter https://www.gartner.com/en/documents/3956746/api-security-what-you-need-to-do-to-protect-your-apis

virtuellen Konferenz zusammengefunden. Erstmals wurde die Veranstaltung von der Bayer AG in Berlin gehostet.

2. Beiträge zum Workshop

Im Vorfeld wurde ein entsprechender Call for Paper innerhalb der Community verteilt, auf dessen Basis 11 Beiträge für den Workshop ausgewählt wurden. Um den speziellen Herausforderungen einer Online-Veranstaltung zu genügen, wurden die Beiträge als Keynote, als 10minütige Themen-Pitches oder mit Hilfe parallel bereitgestellter Poster präsentiert.

Anja Fiegler, Andreas Schmietendorf

Entwicklung smarter Anwendungen mit Hilfe cloudbasiert angebotener KI-Algorithmen;

Niko Zenker, Daniel Paschek, Marvin Leine

Einsatz einer gewichteten Graphendatenbank zur Abbildung komplexer Unternehmensarchitekturen;

Konrad Nadobny

Vergleich von Enterprise API-Management-Lösungen;

Steven Schmidt

Schaffung eines vertrauenswürdigen, öffentlichen WLAN -Herangehensweise und Teilergebnisse;

Michael Petry, Volker Reers, Frank Simon

Reaktive, minimal destruktive API-Härtung am Beispiel von GraphQL;

Jens Borchers

Zero Trust-Architektur und -Kultur;

Daniel Kant, Andreas Johannsen

Exemplarische API-Schwachstellen bei IoT-Geräten auf der Grundlage von OWASP API Security TOP 10;

Gabriel Landa, Sandro Hartenstein

Bitcoin Blockchain via Satelliten;

Kadir Ider

Effective Privacy Management Concepts: Increasing Privacy Control by Reducing Complexity;

Maximilian Müller, Matthias Dobkowicz, Andreas Johannsen, Allan Fodi
Konzeption eines Objektkonfigurators zur Erstellung von Auszügen einer Objektbibliothek;

Sandro Hartenstein
Entwicklung vertrauenswürdiger Web-APIs.

3. Ergebnisse der Breackout-Diskussionen

Der Tradition des Workshops entsprechend galt es, ein World Cafe erstmals virtuell, mit Hilfe von „Breack Out Sessions durchzuführen. Die Teilnehmer wurden zunächst in drei Gruppen aufgeteilt und dann jeweils einem Diskussionsraum zugeteilt. Jedem dieser Diskussionsräum war ein fester Moderator zugeteilt, welcher den Austausch leitete und die Ergebnisse auf einem gemeinsamen Whiteboard dokumentierte. Nach 15 Minuten wechselten die Gruppen den Diskussionsraum, so dass sie sich nun zu einem weiteren Thema austauschen konnten. Dabei bauten sie auf den Ergebnissen der vorherigen Gruppe auf. Nach weiteren 15 Minuten wurde der Diskussionsraum abermals gewechselt.

Massive APIfizierung von Legacy Applikationen

Das Thema der massiven APIfizierung von Legacy Applikationen wurde kontrovers diskutiert, wobei das Ergebnis in der Abbildung auf der folgenden Seite dargestellt ist. Das Thema gliedert sich in die Teilbereiche Standardisierung, Vernetzung und Verschlankung. Im Laufe der Diskussion wurde zudem herausgearbeitet, dass auch Aspekte wie Kultur und Innovation in diesem Kontext eine große Rolle spielen. So bedarf es einer offenen, kollaborativen Kultur mit dem Ziel, dass möglichst alle Daten und Funktionalitäten als API angeboten und standardkonform implementiert und dokumentiert werden (Open API). Die Zielvorstellung ist somit ein Konnektivitäts-Ökosystem, in dem alle Akteure sich einfach und effizient austauschen können. Systeme sind im Idealfall in Echtzeit integierbar und ermöglichen ein Agieren ohne Medienbrüche und Inkonsistenzen. Die Motivation für eine API-getriebene IT-Architektur im Allgemeinen und die dementsprechend benötigte massive APIfizierung von Altsystemen im Speziellen begründet sich im folgenden Sachverhalt:

Etablierung gemeinsamer Standards und Normen zur Reduktion der Komplexität. Auf dieser Grundlage lassen sich Standardlösungen einfacher und vor allem effizienter bereitstellen. Darüber hinaus bietet sich die Möglichkeit, historisch gewachsene Systemlandschaften zu entwirren (d.h. entkoppeln) und die Strukturen sukzessive zu modernisieren.

In Bezug auf Legacy-Applikationen muss beachtet werden, dass die Systeme oftmals über komplexe, historisch gewachsene Businesslogiken verfügen, die nicht verloren gehen dürfen. Mithilfe eines API-Wrappers können diese zum Beispiel hinter einer API-Fassade verborgen werden, so dass das Altsystem modernisiert und endkoppelt werden kann. Die ursprüngliche Kernfunktion bleibt dabei erhalten, so dass es wie gehabt weiter betrieben werden kann. API-Wrapper können somit als Brückentechnologie genutzt werden, um monolithische Systeme zunächst mit einer Standardschnittstelle zu ertüchtigen und dann nach und nach aufzubrechen. Dies erleichtert nicht nur die Systemintegration, sondern ermöglicht auch einen Wandel weg von lokal optimierten Lösungen (Best of Breed) hin zu ganzheitlichen, integrierten Systemlandschaften (Best of Suite).

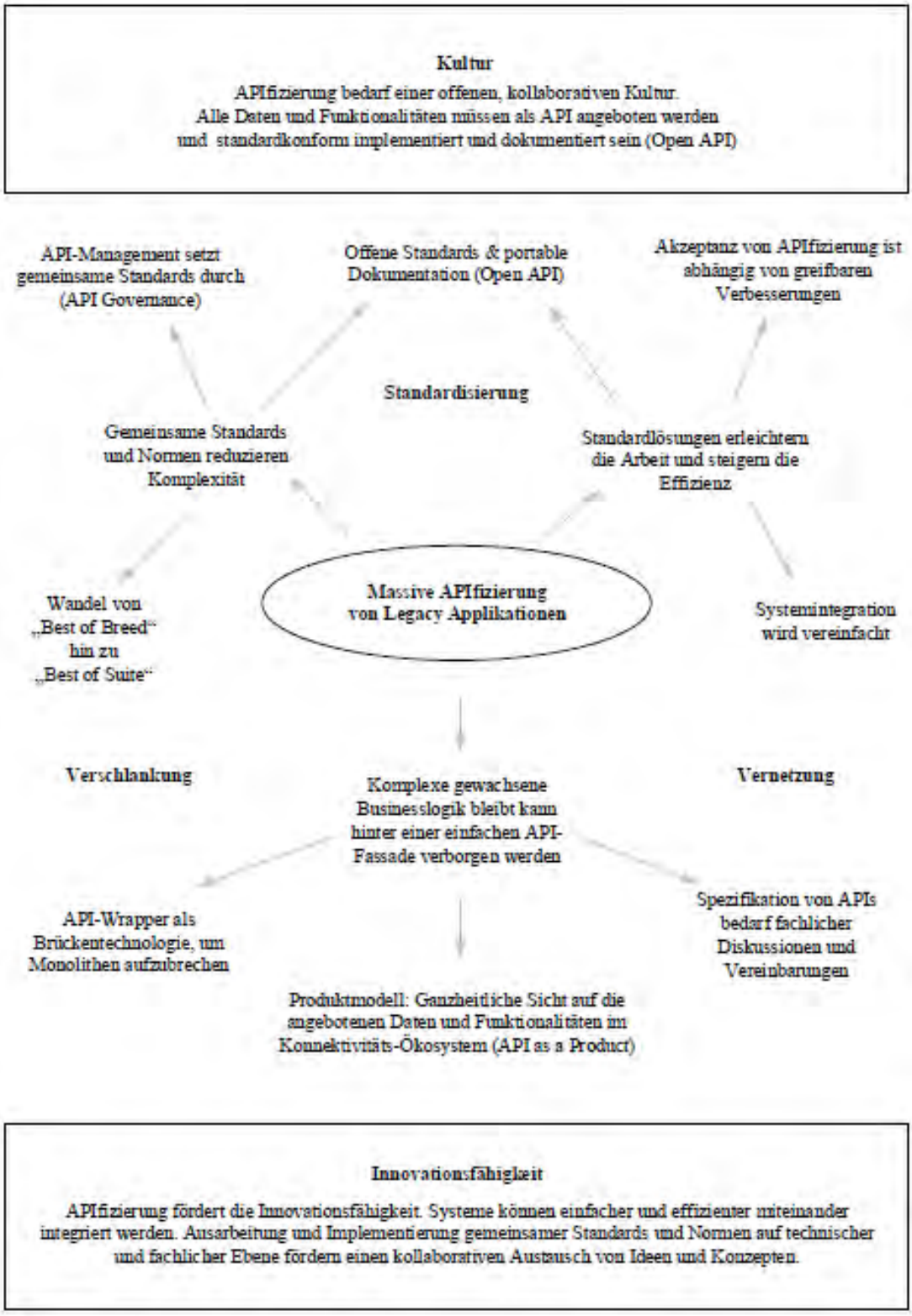


Abbildung 1: Ergebnis der Diskussion

Die Akzeptanz einer API-getriebenen Transformation ist abhängig von greifbaren Verbesserungen. Aus technischer, betrieblicher und organisatorischer Sicht ist zunächst ein professionelles API-Management zum Durchsetzen der gemeinsamen Standards unerlässlich (API Governance). In Bezug auf Daten und Funktionalitäten müssen diese Standards auf fachlicher Ebene diskutiert und vereinbart werden. Diese gemeinsamen Standards, Regeln und Normen reduzieren im Nachgang die Komplexität und vereinfachen den späteren Austausch. Idealerweise werden dabei bereits existierende Industriestandards, wie zum Beispiel domänenspezifische Datenmodelle, implementiert.

Vertrauen in Public WIFI-Infrastrukturen

In diesem World Cafe wurden zentrale Fragen zur Einstellung der Diskussionsteilnehmer gegenüber der Vertrauenswürdigkeit öffentlicher WLANs diskutiert. Einstiegspunkt war dabei die Grundsatzfrage, inwiefern überhaupt öffentlichen WLANs vertraut wird. Das Feedback über die verschiedenen Diskussionsrunden hinweg war stark diversifiziert. Im Wesentlichen wird öffentlichen WLANs nicht vertraut. Häufige Antworten haben aber teilweise nach Anbieter bzw. angebotenem Serviceumfang unterschieden, oder die Entscheidung einer eigenen, kurzen Prüfung verschiedener Datensicherheitseigenschaften vorbehalten.

Im Hinblick auf die Fragestellung zur Relevanz der Vertrauenswürdigkeit für die Nutzung des jeweiligen WLAN-Angebotes müsse nach Meinung der Teilnehmer grundsätzlich zwischen beruflicher und privater Verwendung unterschieden werden. Im privaten Kontext war die Auswirkung auf die tatsächliche Nutzung häufiger irrelevant, im beruflichen oder professionellen Kontext gab es jedoch starke Abhängigkeiten.

Der dritte Diskussionsgegenstand bewegte sich im Bereich vertrauensschaffender Maßnahmen, welche der Service aufweisen müsste, um als relativ vertrauenswürdig zu gelten. Hierbei war in allen Diskussionsrunden der grundsätzliche Konsens erkennbar, dass sich Maßnahmen nicht auf eine rein technische Dimension beschränken dürfen. Kommunikative Aspekte zur transparenten Darstellung von Nutzungsrisiken und entsprechenden - gegebenenfalls betreiberseitigen – Lösungen sind hier sehr häufig genannt worden. Auch die Rolle des Staates als Aufklärer über diese Sachverhalte kam zum Tragen, ebenso wie eine Zertifizierung einer relativen Vertrauenswürdigkeit seitens einer unabhängigen Institution. Eine Anmeldung im öffentlichen WLAN mit einem durch den Betreiber gestellten Zertifikat auf dem eigenen Endgerät ist abhängig von der Anbieterreputation zur Etablierung einer verschlüsselten Kommunikation dabei weitläufig akzeptiert.

Die abschließende Frage für die Teilnehmer des World Cafes befasste sich mit einer potenziell gesteigerten Nutzungsrate öffentlicher WLANs bei eventueller Umsetzung der zuvor diskutierten Maßnahmen und Eigenschaften. Hier war das Feedback überwiegend positiv. Eine Anmerkung bestand darin dass ein Belohnungssystem für die Nutzung eines sichereren Angebots einen zusätzlichen Anreiz darstellen könnte.

Herausforderungen beim KI-Bezug via Web-APIs

Das Angebot von webbasierten APIs, die KI-Algorithmen zugänglich machen, wächst täglich. Aufgrund der zumeist cloudbasierten Bereitstellung dieser technischen Hürden für einen Einsatz von Algorithmen der künstliche im Rahmen der Azure-Plattform oder auch bei der IBM im Rahmen der Bluemix-Plattform. Die Bedenken von Seiten der Anwender, entsprechende Angebote produktiv zum Einsatz zu bringen, sind in Deutschland allerdings enorm. In anderen Regionen wie z.B. im asiatischen oder auch nordamerikanischen Raum steht man dem Einsatz weitaus unkritischer gegenüber. Dementsprechend profitieren innovative Lösungen im Zusammenhang mit fachlichen Anwendungsszenarien, die eher durch den Endanwender bzw. durch potentielle Kunden getrieben werden. Die eher abwartende Haltung in Deutschland impliziert die Gefahr, den Anschluss zu verlieren.

Die folgenden Ausführungen charakterisieren die wesentlichen Eckpunkte der innerhalb des World Cafes durchgeführten Diskussion:

Die Erwartungen der Web-API-Consumer (d.h. Entwickler) sind zum einen eine hohe Security-Grundabsicherung. In diesem Zusammenhang wird typischerweise auf die Authentifizierung und Autorisierung, die netzwerkorientierte Verschlüsselung, das Versionsmanagement sowie ein transparentes Vertragsmanagement Bezug genommen. Zum anderen erwarten die Anwender spezielle Transparenz hinsichtlich des konkret eingesetzten KI-Algorithmus. Die Nachvollziehbarkeit, Verständlichkeit, Genauigkeit und das Vertrauen in die vortrainierten Modelle sind wichtige Anforderungen der Nutzer. Sie können und sollten vom Service Provider adressiert werden. Das Vertrauen in die trainierten Modelle und die Absicherung gegen „böses“ Training sollte von unabhängigen Dritten geprüft und mit Hilfe anerkannter Zertifikate bestätig werden. In diesem Zusammenhang sollten sich auch Standards hinsichtlich des API-Managements bzw. der Spezifikation (Beschreibung) etablieren. Grundsätzlich wurde durch die Teilnehmer festgestellt, dass eine ausschließlich technische Sicht auf die Vertrauenswürdigkeit von Web-APIs nicht ausreicht. Darüber hinaus bedarf es einer Entmystifizierung eingesetzter KI-Algorithmen.

4. Tagungsband und weitere Informationen

Auch für das Jahr 2021 ist die Durchführung eines ESAPI-Workshops vorgesehen. Aktuell gehen wir davon aus, das dieser in Köln (angefragter Gastgeber: Zurich Versicherungsgruppe Deutschland, HS Köln) durchgeführt werden kann. Weiterführende Informationen werden zeitnah unter der folgenden URL im Internet bereitgestellt:

https://blog.hwr-berlin.de/schmietendorf/

Abbildung 2: Tagungsband zum Workshop ([Schmietendorf/Nadobny 2020])

Quelle: https://www.shaker.de/de/content/catalogue/index.asp?lang=de&ID=8&ISBN=978-3-8440-7515-1

5. Quellenverzeichnis

[Schmietendorf/Nadobny 2020] Schmietendorf, A.; Nadobny, K. (Hrsg.): ESAPI 2020
4. Workshop Evaluation of Service-APIs, Berlin – 03. November 2020, 140 Seiten, in Berliner Schriften zu modernen Integrationsarchitekturen, Shaker-Verlag, Düren, November 2020, ISBN 978-3-8440-7515-1

Dank

Unser Dank gilt den Referenten und Teilnehmern, aber auch den Partnern (HWR Berlin, OvG-Universität Magdeburg), Sponsoren (Bayer AG Berlin, Deutsche Bahn AG, Delivery Hero) und Unterstützern im Programmkomitee, die eine solche Veranstaltung ermöglicht haben. Ein herzlicher Dank geht auch an die beteiligten Medienpartner SIGS DATACOM GmbH aus Köln und an den Shaker Verlag GmbH aus Aachen.

Currently COSMIC News

I am delighted to announce that the number of countries with a COSMIC representation has risen to 31 because of the addition of:

Cameroon
In Cameroon, COSMIC is represented by Donatien Moulla (donatien.moulla@cosmic-sizing.org). For more information, visit cosmic-sizing.org/organization/local/cameroon

Jordan
In Jordan, COSMIC is represented by Khalid Al-Sarayreh (khalid.alsarayreh@cosmic-sizing.org). For more information, visit cosmic-sizing.org/organization/local/jordan

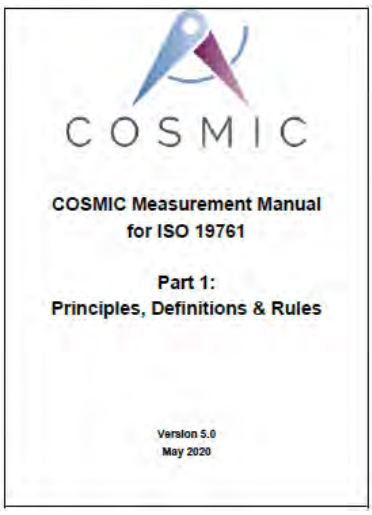
Morocco
In Morocco, COSMIC is now represented by Ali Idri (ali.idri@cosmic-sizing.org). For more information, visit cosmic-sizing.org/organization/local/morocco

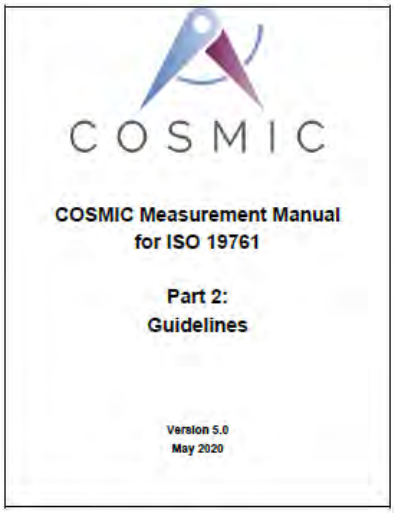
Frank Vogelezang
Chairman

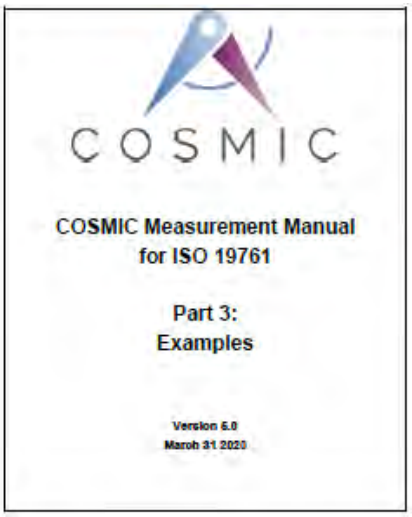
The COSMIC version 5.0 is now available

(see: www.cosmic-sizing.org/)

Part 1: *Principles, definitions & rules* (17 pages)*
Part 2: *Guidelines* (17 pages)*
Part 3: *Examples of COSMIC concepts and measurements (28 pages)*







Software Sizing Tools

There are a growing number of open source and commercial tools that have been created to support the COSMIC methodology. The COSMIC organisation takes no responsibility for these tools. All are web applications or SAAS unless otherwise stated.

Record/Measure CFP counts

- ScopeMaster *Sizer*
- SoftwareLite – App for Android
- Mensura
- VisualFSM – Windows Application

Automated COSMIC sizing from Requirements

- ScopeMaster – *Analyser, sizing and QA of user stories (English, Italian, Spanish & Portuguese).*
- ScopeMaster Story Analyser for Jira – App For Jira Cloud

Tools that perform early estimations in CFP

- Software Risk Master, Namcook Analytics – pre-requirements model-based estimation.
- ScopeMaster *Analyser* – estimations from automated requirements sizing.

Electronic Manual

- COSMIC Docs* – Mobile app version of the COSMIC Documentation. English and Spanish

Tools for CFP base estimation

- SEER for Software, Galorath – Windows Application
- SoftwareExpert – App for Android

Software Requirements Tools that can use CFP sizes

- Jira – custom fields, Trello – custom fields power up.

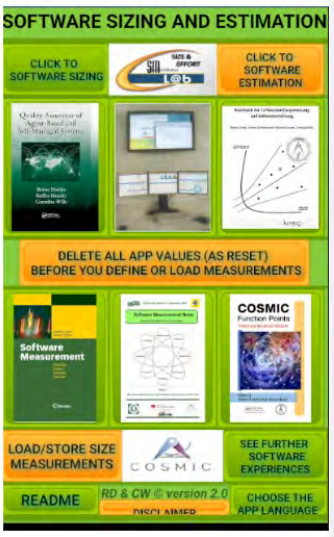
USE OUR APP IN YOUR LANGUAGE

SoftwareExpert in the Google Play Store

CONTENTS

SoftwareExpert **can be applied to quickly and easily determine important project key data for the development of software systems and can be used in German, English, French, Spanish and Italian.**

The software estimation is based on a Software size such as (COSMIC/IFPUG) function points or Story points or Lines of Code or Feature points etc.



further information see in ***www.smlab.de***

Purpose of Software Measurement

Harry M. Sneed
Technical University of Dresden
Ziviltechnik-Prentner, Vienna

1. Why measure software?

There are many reasons for wanting to measure software. Contrary to what many users believe, it is not only to control the quality. Quality is only one of the properties of a software system. There are also other properties such as size, complexity, conformance, compatibility, adaptability and sustainability. All of these properties need to be quantified if they are to be judged and compared. Planned features must be compared with actual features in order to determine if the software fulfills its non-functional requirements.

- *To determine what is done*

What is meant by "**done**"? When is software done, i. e. finished? To answer that question one needs numbers and not just subjective opinions. Of course it is good to have an opinion, but that is not enough. There must be numbers to support that opinion. One man's opinion can be another man's imagination. There must be some way to distinguish between fact and fiction. This is where measurement comes into play. Measurement brings objectivity into decision making. It delivers facts upon which decisions can be based, for instance the decision as to whether to release a software version or not.

- *To predict the costs of doing it*

Another reason to measure software, one which is particularly important for user management is to predict the costs of a potential project. In the case of smaller projects where only a few people are involved, it may be possible to estimate the costs of a project based on expert opinion or the combined predictions of project participants, but with larger projects where many persons are involved this no longer holds, The effort required has to be systematically calculated using some proven estimation method. Such estimation methods rely on numeric data to calculate the time and costs. Some may say that this is just another form of witchcraft, but this is all we have in a world of uncertainty. This form of algorithmic cost calculation is only as good as the numbers which go into it. If the method used to predict effort is based on the size of the code then one must expect that size measurement, whether it is in lines of code or in source statements, to be accurate. The same applies when counting the number of data items processed as with the Data-Point method or the number of process inputs and outputs as with the Function-Point method. The elementary entities must be counted correctly if the cost prediction is to be reliable. The larger the project is, the more important it is to measure accurately.

- *To compare performance*

A further reason for wanting to measure software is to compare the productivity of the software producing teams. Our free market economy is based on competition. Teams and firms compete with one another to produce more software at less cost, or as Tom DeMarco once put it – more bang for your bucks. How can one know if a team is really earning its money, or if it is just spending its time? Team output, i.e. productivity, must be in some way compared with a bench mark. That productivity benchmark can be obtained from past performance. Software producers should keep records of how much software they have produced in the past and what effort was required to produce it. This is referred to as their productivity rate. They might also take the productivity rate from other service providers working under similar conditions to use as a benchmark. By comparing the current productivity of their teams with past productivity and with the productivity of foreign teams they can determine if their teams are working as they should be. If not, they can study the reasons why and come to a conclusion as to how to raise their productivity. In any case, productivity measurement is an essential prerequisite to any means of process improvement.

- *To judge the quality of the software*

Finally, there is a reason to want to judge the quality of a software product. How good is a piece of software? How does it compare with other software of the same type? To answer these questions the quality of the software must be measured. Software quality is both static and dynamic. **Static quality** is the degree to which the code and the documents fulfill the static quality requirements. These requirements are the standards to be fulfilled. Once they are accepted and approved, it is only a question of checking them. Any violation of these standards, i.e. rules is considered to be a deficiency in the software, regardless of whether it seems to be meaningful or not. As long as the rules have been approved by a legitimate organizational unit, it is up to the projects to abide by them. Software standards are like traffic rules. If they are passed and approved by the local government then they apply to the territory under jurisdiction of that local government. Anyone passing through that territory is obliged to abide by them. There can be no exceptions. The same is with software rules. They must be followed by all those developing software within the jurisdiction of the local standards board. Standards are enforced by measuring the code and documents they apply to.

Dynamic quality is the degree to which the software performs according to the specification. This is determined by testing. The software is executed and its performance measured. To be measured is the execution time, the amount of memory required, the code, data and functional coverage and the error rate. The software should execute within the time limits set and not use more memory than what is allowed. The code, data and functions should be covered to the extend specified. In the case of code that is the percentage of statements or branches executed. In the case of data that is the percentage of data values set or used. In the case of functions that is the ration of functions tested relative to functions specified. The number of errors detected should be below the maximum error limit set. The execution time, the coverage levels and the error rate are recorded, i.e. measured.

These measurements are essential to assessing the dynamic quality of the software in question. The software should only be released when it has fulfilled the minimum quality criteria. Thus, as pointed out here, there are many good reasons for measuring software, namely

- to determine the degree to which it is done
- to predict its costs
- to ascertain the productivity of those developing it
- to assess its quality.

There may be other reasons as well, but these are the main ones.

2 What software should be measured?

A software product is made up of different partial products, which are equivalent to views of that product. There is the view point of the users, the view point of the analysts, the view point of the architects, the view point of the programmers and the view point of the testers. Users see the user interface and the instructions for using it; analysts see the requirement specification; architects see the design model and its documentation; testers see the test cases and the test documentation. Each of these partial products has its own language and its own rules for using that language. They also have their own measurements. Requirements specifications are normally made in natural language, e.g. English or German. Design models are now made mostly in UML, although there are alternative modelling languages such as ODL. Code can be written in one of many programming languages ranging from Basic Assembler to Java Script and PHP. Testware can be made with scripts, tables or mark-up languages like XML. The biggest challenge to software measurement lies in processing so many different languages, each with its own unique features.
Fortunately no one will have to deal with all of them. At any one user site only a subset of the many languages will be used. There will be at most the local natural language plus English, one modelling language most likely UIML, one or two programming languages and perhaps a test language. The key objects to be measured are

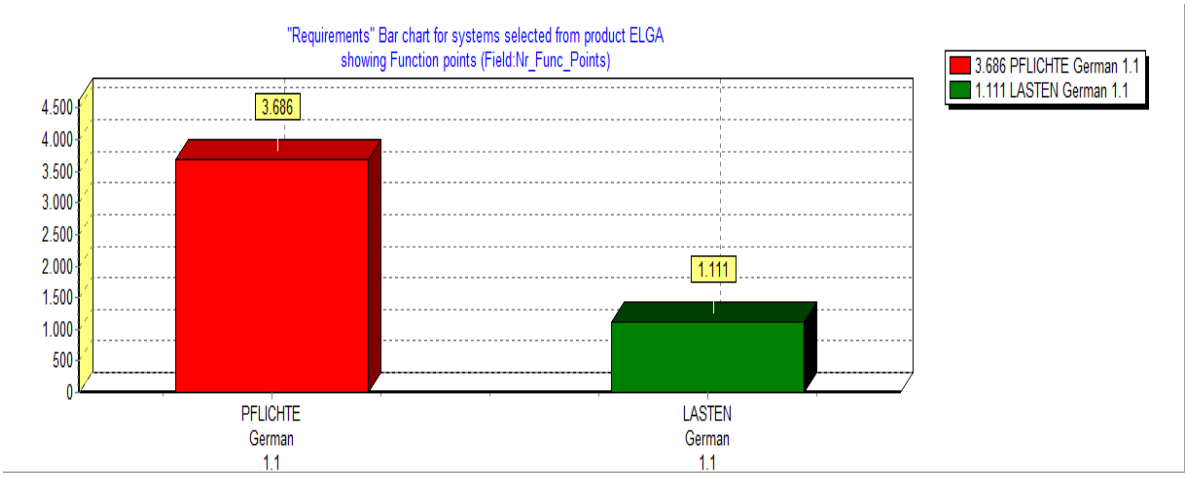
- the requirement specification
- the design model
- the source code and
- the test cases.

All of these partial products are objects of measurement. All of them have a size, a complexity and a quality that can be expressed in terms of quantitative values such as pages of documentation, lines of code and number of table entries. These are examples of physical quantities. There are also logical quantities such as the number or requirements, the number of design entities, the number of modules, the number of statements and the number of test cases. In measuring software, one must distinguish between physical and logical measurements. There is a big difference between counting lines of code and counting statements, just as between counting data values and counting data definitions. Those who measure software must be aware of those differences.

2.1 Requirement documents

Measuring a requirement document requires a model of the requirements. It must be possible to count the model entities like processes, objects, use-cases and rules. Currently, there is no generally accepted model for specifying requirements such as UML for the system design. The burden of defining a requirement model is put on the individual user. Whoever is measuring the requirements must take over that model and define measurements for it. Not even the often cited function-points have a generally accepted model behind them. So it means that requirement measurement have only a local significance. They cannot be compared among different organizations. Nevertheless requirements should be measured if only for comparison of past and present projects. It is up to those responsible for their measurement to define a model based on current requirement texts and to identify the model entities within that text. Once they are identified, the entity types can be counted and measurements made with those counts such as the number of business processes, business objects and business rules. The important thing is that they are counted in the same way in all of the requirement documents of that particular user organization. These elementary counts can then be used to compute more complex metrics such as function-points data-points, object-points and use-case points. These are the metrics used to make cost calculations based on requirements. This is the foremost goal of requirement measurement, but there are other goals as well such as determining the completeness and consistency of the requirements, and assessing the requirement quality.

Tab. 1: Sample of Size Measurement based on Requirement Documentation

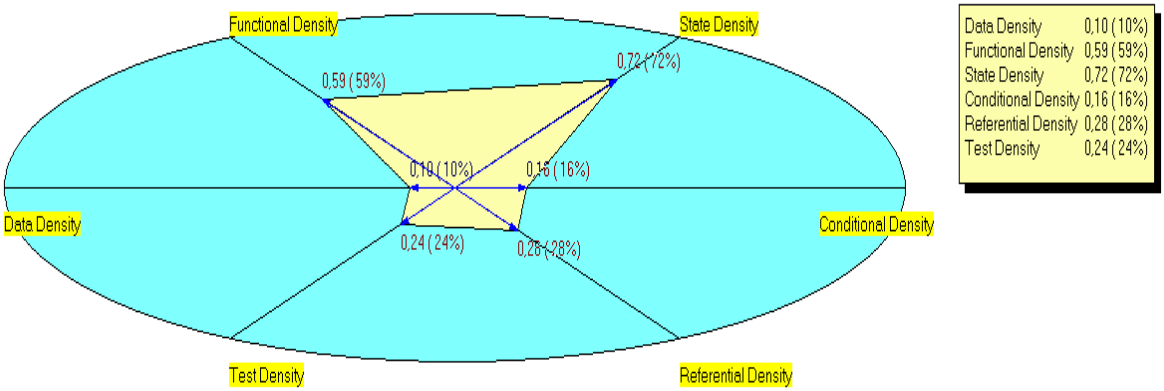


2.2 Design models

Measuring design models is much easier than measuring requirement documents because they are formally defined, mostly in the form of an XMI schema. UML models appear to the user as a series of interrelated diagrams but behind these diagrams is an XMI schema which can be processed and parsed. The XML parser can easily recognize the model entities and count them, entities such as classes, interfaces, methods and data attributes. In the UML-2 language there are 13 different

entity types. Counting them makes it possible to compute metrics for design size, complexity and quality. In this way it is possible to measure object points as well as function points and to estimate effort based on these adjusted sizes. It is also possible to compare design models and to judge the quality of the system design. If the design is found to be faulty it is still possible to correct it before coding begins. This is the main advantage of design measurement. It can also be used to calculate the costs of coding and testing, but for calculating the overall costs of development, it comes too late.

Tab. 2: Sample of Complexity Measurement based on Design Documentation



Spiderweb chart for product ELGA, system LASTEN v 1.1 "Requirements"

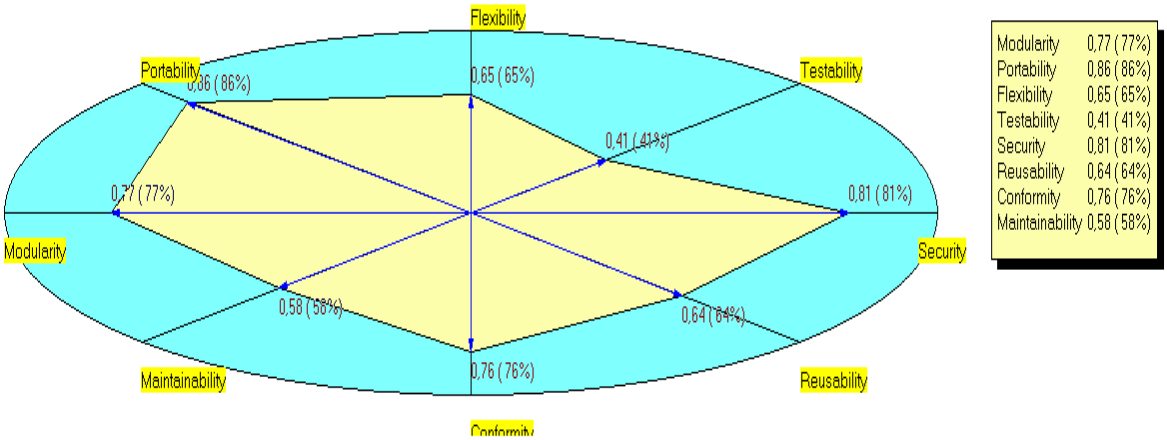
2.3 Source Code

The source code is considered by naïve users to be the software per se. It is all that matters and is all that is worth maintaining. This is a very narrow definition of software. If the code is really all that matters, it makes no sense to document the requirements or to create a design model. The fact is that if we only retain the source code, there is nothing to test the source code against. Since we cannot compare it with anything else, we have no way of determining whether it is right or wrong. We are testing the code against itself. Besides our view of the software system is restricted to this one perspective, namely to that of the coder. Users, analysts, designers and testers are excluded from understanding the software. If they want to understand it, they had best learn the programming language it is written in.

For some conniving coders this might be what they want. They don't want anyone else to understand what they are doing. This is definitely not good for the application system and in the long run it is not good for the programmers. Their narrow view of the system should be enhanced by other views at higher levels of abstraction. The saying that one cannot see the forest because of the trees fits very well here. To truly comprehend a complex software system one must view it from different perspectives. Not only the programmers but also the analysts, designers data modelers and testers must be able to understand the system and to follow its evolution. This and the need for a base line to test against is reason enough to maintain the adjacent documents.

Still, we need to be able to comprehend ant to measure the source code itself. Measuring the source code means we have to model the structure of the code and to count the code entities – statements, procedures, data declarations, parameters, etc. With these counts we can then measure the code size, complexity and quality. The literature is rich with metrics for measuring code. One must however collect the numbers for feeding those metrics by parsing the code. The problem with that is there are so many different programming languages and each must have its own parser. Even within the same user organization there may be several languages used. To estimate and trace the costs of software evolution, all of them have to be measured according to the same rules. This is the greatest barrier to code measurement.
The positive aspect of code measurement is that almost every software property can be derived from the code including such abstract properties as function-points and object-points. Standards are now emerging for counting them and, since the programming languages are formally defined, the counting algorithms can be automated. There may be excuses for not measuring the requirement and the design, but there is no excuse for not measuring the code. It is the first step to installing a software measurement program.

Tab. 3: Sample of Quality Measurement based on the Source Code



Spiderweb chart for product ELGA, system $SYSOO v 1.1 "Code"

2.4 Testware

Only recently has the testware of a system become an object of measurement. Up till now test ware was considered to be an unmeasurable substance because it was not definable. Thanks to the work of the testing community founded by Gelperin and Hetzel and institutionalized in the ISTQB test ware has now become measurable. It consists of the test cases and the procedures for executing them as well as the test data and the procedures for generating and validating that. These procedures are documented in the form of test scripts, test case tables, XML forms, test data tables and assertions. In so far as these documents are at least semi formally defined, they can be parsed and their contents measured. The first step is to define a set of test case attributes common to most tests, attributes such as the parameters to and the results from a test case, with value ranges, as well as the exception conditions.

The goal is to determine the size of the test procedures and the number of test cases with their attributes. With this information, converted to test-points, it should be possible to estimate the costs and time of testing a system before testing begins.
Once testing has begun, it becomes possible to document the progress of the test. On the one hand the test coverage is recorded both for the functionality and the data. Functional test coverage is measured in terms of number of functions tested relative to the number of functions specified. Data coverage is measured by comparing the number of data items defined with the number of data values generated. Code coverage is based on the ratio of statements executed relative to the sum of all statements. This data reveals the extent to which the target system has been tested. It is up to the testers to install and to use the tools for recording the test coverage measures.

Parallel to recording the test coverage, testers should also be recording the errors that occur. Any time a result occurs which does not match the result specified in the test case definition, an exception is triggered. These exceptions are recorded as potential errors to be validated by the tester. Once they are validated by the responsible tester, the potential errors become real errors and are placed in the project error log. The error log becomes part of the test documentation along with the test coverage reports, the test scripts and the test case specifications. The contents of the test documentation become objects of measurement. In fact many counts and metrics are already included in that documentation. So it is only a question of copying and aggregating them.

Metrics from a Java Service Test

Service	Opers	Stmts	Logic Branchs	Params	FuncPt	Test Paths	Tester Hours
Calendar	3	473	31	38	12	15	8
OrderEntry	16	625	187	43	29	92	37
BauSparer	17	276	47	64	35	22	13
BeautySalon	24	429	72	54	18	33	21
Geometry	5	510	73	19	9	36	18
Authorize	27	573	265	19	22	130	65
MailService	48	3317	762	211	126	278	88
Total	140	6203	1437	448	251	606	250

Test Productivity = 0.41 Test Cases per Hour or 2.4 Hours per Test Case

3 How should software be measured?

Once it is clear what to measure, the next question is how to measure it. How to measure is of course dependent on what we are measuring.

Measuring a requirement document requires other metrics than measuring source code, but the measurement techniques are similar for all software artefacts, be it code, documents or testware. The artefacts are texts, diagrams or tables. Behind the formal diagrams – such as UML – are texts in some kind of markup language like XML. Also the tables can be converted to text. Whatever is in a text format can be converted scanned or parsed automatically. This leaves only the free form graphics to be measured manually. In the case of such diagrams one must visually examine the graphics and count the different graphic objects. Since this is a very time consuming activity, it is seldom done. Normally is too expensive to practice. That means that only software which can be automatically processed can be feasibly measured. This applies to source code, design models, test cases and requirement texts in so far as they are properly prepared. Only source code and standard design models can be measured with no preprocessing. Requirement texts and test cases have to be pre-processed.

The prerequisite to measuring requirements is to mark up the requirement text. The requirement model entities must in some way be made recognizable. Either they are labeled from the beginning when writing the requirements or their identifiers are inserted into the finished text. One sample solution used by this author is to insert label lines before each text segment or table to which the label applies. (see Sample). A text parser should be able to recognize the markers and identify them as measurable entities. Besides the labels which mark the beginning of a model entity, there should also be some sort of delimiter to mark the end of that entity. Everything between the beginning marker and the end marker is considered to belong to that entity. The goal is to count the occurrences of each entity type as a means of measuring system size without rewriting the document. Without knowing the system size of a software system it will never be possible to estimate the costs of doing something with that system whether it is to develop it, to maintain it, to reengineer it or to convert it. In the case of a new development it may only be possible to measure the requirements. Therefore we will need requirement metrics. For maintenance, reengineering or conversion we can measure the source code.

There are now well established techniques for measuring source code. We need only a parser for that particular language which recognizes specific language elements such as lines of code statements, data objects, procedures, modules, etc. These have to be counted and weighed. Many of the same elements can be found in the design model, only at a higher level of abstraction. In an UML-XMI schema each entity type is identified. The same applies to the test case schema. The problem there is that there is no universal standard for naming the test entities. The user has to define requirement and test entities for himself. This can be done with a test entity table.

As already pointed out the main barrier to a complete measurement of all software artefacts from the requirements to the testware is the lack of a universal requirement model. For the time being users must help themselves by defining their own model in the form of a domain specific language which can be processed and measured.

4 Conclusions on Software Measurement

It has been stated before that without measurement there can be no engineering. Measurement is a condition sine qua non to any engineering discipline. This applies also to software engineering, and what is more to requirements engineering. Most of the currently used software artefacts can already be measured. Even the requirements can be measured if they are put into at least a semiformal model with recognizable, well defined entities and relationships. With the data gathered from the documents, the code and the testware it is possible to fulfil the goals of software measurement at all levels of abstraction:

- requirement level
- design level
- code level and
- test level,

namely to

- predict future project costs
- determine current project status and
- assess the quality of products.

5 References on Software Measurement

Ebert, C., Dumke, R, Bundschuh, M., Schmietendorf, A.(2005): *Best Practices in Software Measurement,* Springer Verlag, Berlin.

Ebert, C., Dumke, R.(2007): *Software Measurement,* Springer Verlag, Berlin.

Sneed, H., Seidl, R. (2010): *Software in Zahlen*, Hanser Verlag, München

Large Scale Software Systems and Their Project Indicators

Reiner Dumke, Anja Fiegler, Cornelius Wille
University of Magdeburg, Microsoft Germany,
and Technical University of Applied Science Bingen, Germany

1 Introduction

The following paper considers the large scale software systems that are used of us every day. Software permeates our lives to an ever greater extent, as already expressed by *pervasive computing.* For the most part, software systems are available constantly and everywhere (as *ubiquitous computing*) and influence us in a permanent way.

Especially in the commercial sector, the relationships and dependencies between software systems continue to increase, as can be clearly seen, for example, in the ***Gallery of e-Business Systems*** by R. Neumann [15].

On the other hand, the consequences of poor software quality with regard to a software system or its development and application process are often dangerous and sometimes dramatic (see the sufficiently well-known collection of software application errors by P. Neumann [14]).

This makes it all the more important to ensure the quality and controllability of extensive software systems. For this purpose, there are extensive quality criteria and measures that signal the possibility of quality deficiencies or can also identify them from the experience of empirical software engineering.

The estimation of important project indicators in this context is supported by different tools. In this paper, the estimations of these essential project data are based of our ***SoftwareExpert App*** (available in the Google Play Store) (see [4] and [6]). The estimation could be select from a list of authors for a special kind of project metrics or you can use all shown metrics building the average values. In our paper we have chosen the average metrics values (the project costs excluded).

The way our app is used, especially for estimating important project and product features, is illustrated in the following figure. It shows that for the application of the estimation formulas, a scope measure such as Lines of Code, Story Points, Feature Points and others are necessary to perform the respective estimations. For this, a conversion into the scope measure of this app (the COSMIC Function Points) is only necessary at the beginning (see figure 1).

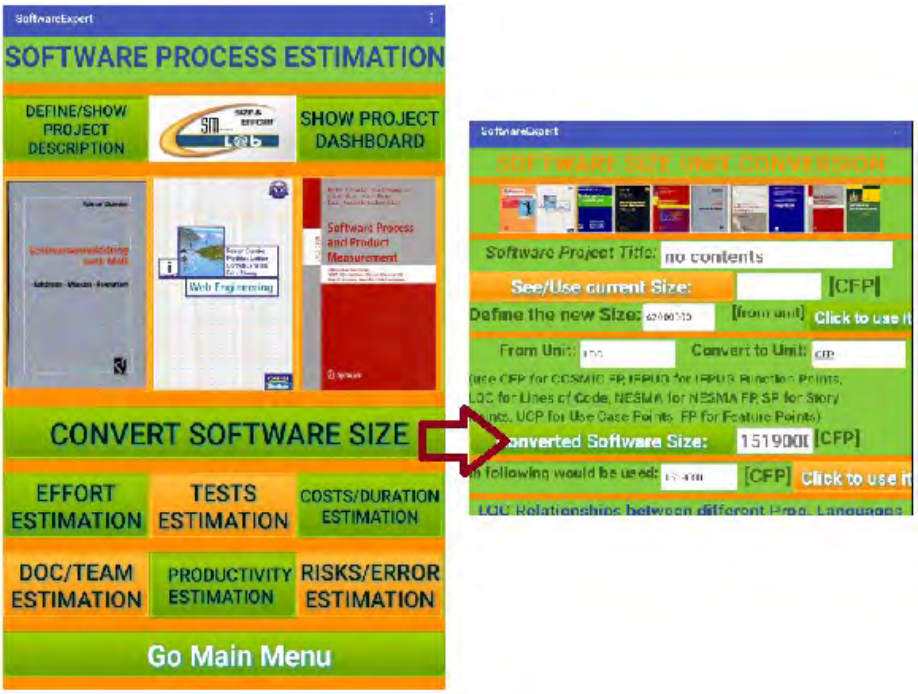


Fig.1: Software Size Conversion with our *SoftwareExpert* App

Especially for the estimation in our app, we have used the already very extensive literature collection of our community, such as [1], [2], [5], [7], [8], [11], [12], [13], [16], [18], [19] and many other more. There is also the possibility to select certain preferred estimation or to define your own estimation formula or estimation factor in out app.

2 Considered Software Systems

Our analysis is based on a review by Desjardins [3] of very large software systems and their size in lines of code. We have selected some of these huge software systems, as they mostly accompany us in our daily lives.

We use our smartphones every day, marvel at flights into space, edit our videos and photos, file with modern aeroplanes, communicate via Facebook, dream of autonomous driving and Google daily for terms, topics or localities in the world. All these software systems surround us constantly, and most of them expect a high level of reliability.

The following table shows the selected systems, their scope in Lines of Code and the CFPs (COSMIC Function Points) already converted by our app, which are the basic scope measure for applying the estimates.

Software System	Lines of Code	COSMIC Function Points
Our SoftwareExpert app	15 000	367
Average iPhone app	40 000	980
Space Shuttle Software	400 000	9800
CESM Climate Model	1 200 000	29400
Hubble Space Telescope	2 000 000	49000
Photoshop C.S. 6	4 000 000	98000
Windows NT 3.1	4 200 000	102900
HD DVD Player on XBox	4 500 000	110250
Google Chrome	5 300 000	129850
Boeing 787	13 000 000	318500
F-35 Fighter jet	22 000 000	539000
Microsoft Office 2013	44 000 000	1078000
Facebook	62 000 000	1519000
Mac OS X "Tiger"	84 000 000	2058000
Car Software	100 000 000	2450000
Google	2 000 000 000	49000000

Tab. 1: Chosen Large Scale Software Systems from [3]

3 Measurement and Estimation Intentions

As a measurement result, we have already taken the software size as a given (here we have also immediately converted the LOC into the CFP for the application of our app).

For the estimates we can select the following two metrics or measures [4]:

1. ***Product measures***: Of course, this includes the *software size*, the required *test cases*, the d*ocumentation size*, and the *product defects* (assuming professional development of the systems).

2. ***Process and project measures:*** For this, we can use the project data known from the literature on the required *team size*, *development effort*, *testing effort* and *documentation effort*, *project duration*, *productivity* and *project risk*.

The following table shows an example of our ***SoftwareExpert*** application characterizing the DVD player software.

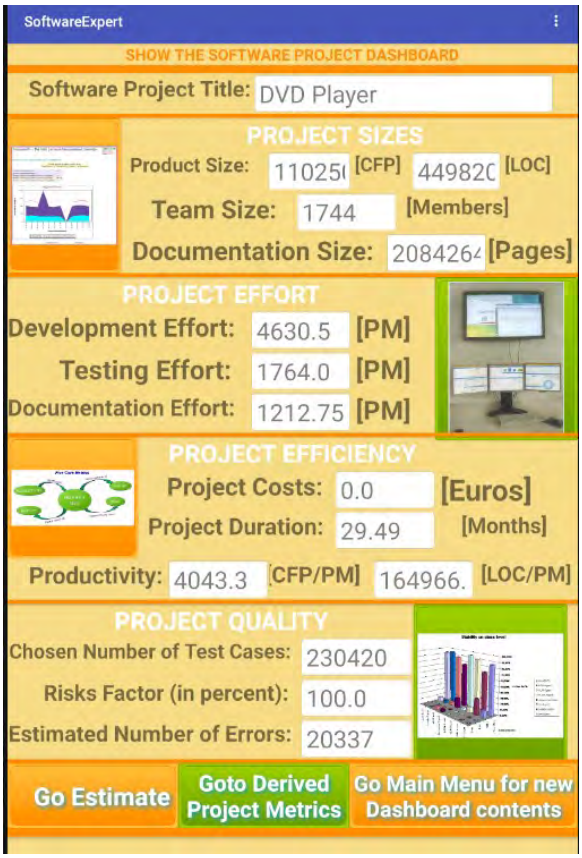


Fig. 2: Example of Estimated Project Data using our SoftwareExpert App

For the estimation of the very large software systems considered here, it is recommended to use the tablet version of our app.

4 Measurement and Estimation Results

Of course, it is not a surprise that small examples leads to small project data and large ones to big data. Furthermore, the examples have different motivations and intentions.

But, some estimations could be interesting and meaningful. Note, the size based estimation don't differ between the kind of system (real-time or business etc.).

In following we show some tables and leave the interpretation to the reader for his own needs (PM means personal month). We have added the estimation results of our *SoftwareExpert app* itself in order to characterize the quality and project data of the measurement tool.

Software System	Team Size	Documentation Size (in pages)	Estimated Number of Errors
Our SoftwareExpert app	6	6937	67
Average iPhone app	16	18524	180
Space Shuttle Software	155	185266	1807
CESM Climate Model	465	555803	5422
Hubble Space Telescope	775	926338	9038
Photoshop C.S. 6	1550	1852678	18077
Windows NT 3.1	1628	1945313	18981
HD DVD Player on XBox	1744	2084264	20337
Google Chrome	2054	2454800	23952
Boeing 787	5039	6021209	58752
F-35 Fighter jet	8527	10189730	99427
Microsoft Office 2013	17054	2037948	198855
Facebook	24031	2871654	280204
Mac OS X "Tiger"	32558	3890628	379632
Car Software	38759	4631700	451943
Google	775180	92634000	9038866

Tab. 2: Chosen Large Scale Software Systems and their required team size, documentation size and their potential defectiveness

Team size and the ***documentation size*** are classic orientations that are primarily intended to illustrate a respect for the complexity of scope in these software systems. However, special attention should be paid to the ***Number of errors*** contained in the software system, especially under the condition of a professional development of these systems. This does not mean that the software developers or programmers did a bad or inadequate job of development, but is mainly due to the complexity of these software systems.

From theory we know that a (correct) specification model ***M*** as algorithms ***A*** written in a modelling language ***ML*** results in a programme ***P*** in a programming language ***PL*** during implementation compiled with an (optimizing) compiler ***C,*** using a (class/asset) library ***B***, running of a chosen operating system ***O*** in a server grid ***G,*** using some services ***S*** of a cloud etc., i.e. as a simplified formula

$$S(A)_{ML} \rightarrow_{reification} P_{PL} \rightarrow_{optimizing} P_{C,B} \rightarrow_{execution} O(P)_{G,S} \quad \text{(eq. 3.1)}$$

This concise presentation is also necessary in theory in order to fundamentally consider the problems of verifiable transfer of a specification into a programme implemented in a special programming language (such as the reification from specification to formal design and finally to formal implementation [9]).

The complexities analysed here consider, for example, the processing effort (whether polynomial or exponential) or even the realisability in general (as non-determinism or non-feasibility) etc. The main forms of complexity are summarised in the ***Rogers Gallery of Complexity*** (see in [10]).

In practice, however, there are other features or influences on complexity characterised by empirical software engineering. Consequently, the overall implementation task above is:

> *A **programme P** developed by the **developer W** with the **training A** and the **experience E** of the **department I** of the **company F** in the **programming language L** with the **programme library B** in the **development time T** and the **development methodology M** to the **current (modern) paradigm G**, the **test procedure V**, with the **hardware H** from the **manufacturer R** from an underlying **algorithm S** with the **complexity K** and a proven **quality Q** for the **customer C,** and thus*

$$S \rightarrow_{W,A,E,I,F,L,B,T,M,G,V} P \rightarrow_{H,R,K,Q,C} O(P) \qquad \text{(eq. 3.2)}$$

Capers Jones has summarised these further complexity features, among others, in a list of 24 types of complexity [11]. Of course, the complexities corresponding to (eq. 3.1) belong to it, such as *Algorithmic complexity, Computational complexity and Problem complexity.* Furthermore, complexities such as *Code complexity, Data complexity and Flow complexity* are already involved.

However, such complexities as *Cyclomatic complexity, Combinatorial complexity, Diagnostic complexity, Entropic complexity, Essential complexity, Fan complexity, Function point complexity, Graph complexity, Halstead complexity, Information complexity , Logical complexity, Mnemonic complexity, Organizational complexity, Perceptional complexity, Process complexity, Semantic complexity, Syntactic complexity and Topological complexity* are relevant could be identified by any metrics or measures and could apply to all components as ***W, E, M, G*** etc.

Hence, the consideration of the number of errors in Table 2 should therefore on the one hand also take into account the complexity of this extensive software, but on the other hand should also be understood as an indication of the "only" achievable quality.

> ***Software or programme errors are the price to pay for the challenge of incorporating and mastering high complexities.***

The next table summarises the respective estimated ***efforts of the software systems*** considered.

Software System	Development Effort (in PM)	Test Effort (in PM)	Documentation Effort (in PM)
Our SoftwareExpert app	15	6	4
Average iPhone app	41	16	11
Space Shuttle Software	412	157	108
CESM Climate Model	1235	470	323
Hubble Space Telescope	2058	784	539
Photoshop C.S. 6	4116	1568	1078
Windows NT 3.1	4321	1646	1131
HD DVD Player on XBox	4630	1764	1213
Google Chrome	5454	2077	1428
Boeing 787	13377	5096	3503
F-35 Fighter jet	22638	8624	5929
Microsoft Office 2013	45276	17248	11858
Facebook	63798	24304	16709
Mac OS X "Tiger"	86436	32928	22638
Car Software	102900	39200	26950
Google	2058000	784000	539000

Tab. 3: Chosen Large Scale Software Systems and their Development, Test and Documentation Effort (PM means personal month)

The large expenditure figures are of course based on the assumption of a typical (classic) team size and the considerable effort required for (classic) detailed documentation of all project processes.

Of course, the entire Google development did not take 170000 years. These efforts are rather to be seen as the challenge that the respective software developers or providers face and have to overcome.

Also, we did not produce the usual 6000 pages of documentation for our app either.

Finally, the last table shows the necessary ***test cases*** for a complete test of the system, the usual ***project duration*** and the so-called respective ***risk factor*** according to Jones [12].

Software System	Number of Test Cases	Project Duration (in Month)	Risks Factor (in percent)
Our SoftwareExpert app	762	4	65
Average iPhone app	2044	6	65
Space Shuttle Software	20479	15	100
CESM Climate Model	61444	19	100
Hubble Space Telescope	102410	22	100
Photoshop C.S. 6	204820	28	100
Windows NT 3.1	215059	29	100
HD DVD Player on XBox	230420	29	100
Google Chrome	271385	31	100
Boeing 787	665665	43	100
F-35 Fighter jet	1126519	52	100
Microsoft Office 2013	2253020	66	100
Facebook	3174710	75	100
Mac OS X "Tiger"	4301220	83	100
Car Software	5120500	89	100
Google	10241000	263	100

Tab. 4: Chosen Large Scale Software Systems and their Number of Test Cases, Project Duration and their Risks Factor

The ***number of test cases*** is also only intended to acknowledge the effort involved in extensive software systems. Likewise, the ***project duration*** is also due to the estimated team size above. For the entire Google, therefore, about 22 years would be required here as well in the context of the literature on project experiences. Jones' ***risk factor*** indicates the probability of failure of a project in percent. This means that the development of such extensive software systems has mostly failed (100 per cent).

Of course, the risk factors have very different influencing variables, as can be seen, for example, in the ***Gallery of Risk Factors*** according to Richter [17].

5 Conclusions

Our paper was dedicated to very large software systems that we use every day and for which the programme lines (as LOC) are known. Based on our previous experience in the IT sector, we estimated the corresponding project indicators with our *SoftwareExpert* app. In doing so, we documented above all the immense

resource expenditure, but also characterised the problem of real systems not being free of errors.

We should always be aware of this when we see the ever increasing use of complex software systems in all areas of society.

6 References

1. Buglione, L.; Ebert, C.: *Estimation Tools and Techniques.* IEEE Software, May/June 2011, S. 91-94
2. Bundschuh, M.; Dekkers, C.: *The IT Measurement Compendium.* Springer Publ., 2008
3. Desjardins, Jeff: *How Many Millions of Lines of Code Does It Takes?* see: https://www.visualcapitalist.com/millions-lines-of-code/
4. Dumke, R.: *The SoftwareExpert App.* Software Measurement News, 25(2020)1, pp. 5 – 6 and 24(2019)2, pp. 35 – 41 (see in the Google App Store)
5. Dumke, R.; Abran, A.: *COSMIC Function Points - Theory and Advanced Practices.* CRC Press, Boca Raton, 2011
6. Dumke, R.; Fiegler, A.; Wille, C.: *COSMIC Examples – What could they mean as an IT project?* Software Measurement News, 25(2020)2, pp. 43-52
7. Dumke, R.; Schmietendorf, A.; Seufert, M.; Wille, C.: *Handbuch der Softwareumfangsmessung und Aufwandschätzung.* Logos-Verlag, Berlin, 2014
8. Ebert, C.; Dumke, R.: *Software Measurement – Establish, Extract, Evaluate, Execute.* Springer Publ., 2007
9. Habrias, H.; Frappier, M.: *Software Specification Methods.* ISTE Ltd. Publ., 2006
10. Hemaspaandra, L. A.; Ogihara, M.: T*he Complexity Companion*. Springer Publisher, 2002
11. Hill, P. R.: *Software Project Estimation – A Workbook for Macro-Estimation of Software Development Effort and Duration.* Kwik Kopy Printing, Australien, 1999
12. Jones, C.: *Estimating Software Costs – Bringing Realism to Estimating.* McGraw-Hill Verlag, New York, 2007
13. McConnell, S.: *Software Estimation.* Microsoft Publ., 2006
14. Neumann, P.: *The Risks Digest*, see http://catless.ncl.ac.uk/Risks
15. Neumann, R.: *The Internet of Products.* Springer-Verlag, Wiesbaden, 2013
16. Putnam, L. H.; Myers, W.: *Five Core Metrics – The Intelligence Behind Successful Software Management.* New York: Dorset House Publishing 2003
17. Richter, K., Dumke, R.: *Modeling, Evaluating and Predicting of IT Human Resources Performance,* CRC Press, Boca Raton, 2013
18. Sneed, H. M.: *Software Projektkalkulation.* Hanser-Verlag München, 2005
19. Sneed, H. et al.: *Software in Zahlen*. Hanser-Verlag München, 2010

Analyse internetbasierter Datenspuren mit Hilfe des Web Scrapings - Möglichkeiten, Technologien, Tests und Problemstellungen

Andreas Schmietendorf, Walter Letzel
HWR Berlin & OvG-Universität Magdeburg

1 Motivation und Ziele

Die Analyse von im Internet zur Verfügung gestellten Informationen kann auf eine lange Tradition zurückblicken. Bereits in den frühen Jahren des Internets wurden entsprechende Crowler-Mechanismen in Suchmaschinen oder auch (Preis-) Vergleichsportalen verwendet. Auch im Zusammenhang mit Projekten des Data Science finden sich entsprechende Ansätze, wobei hier zumeist vom so genannten Web Scraping gesprochen wird. Die klassische Zielstellung des Web Scrapings besteht darin, Informationen im Internet zu „schürfen“, um diese dann weiteren Analysen (z.B. mit Methoden des Data Minings) zu unterziehen. Korrespondierende Daten, die sich so gewinnen lassen, können hinsichtlich ihrer Struktur und Semantik bekannt, aber auch unbekannt sein. Entscheidend für aussagefähige und valide Datenanalysen ist nicht die Datenmenge sondern vielmehr die Güte der über das Web Scraping gewonnenen Informationen. Typische Aspekte im Zusammenhang mit der Datengüte beziehen sich z.B. auf die Datenkonsistenz, d.h. inwieweit repräsentieren die Daten zu untersuchenden Sachverhalte oder auch auf die Vertrauenswürdigkeit der genutzten Datenquellen.

Im Rahmen des vorliegenden Papers werden die Erfahrungen im Umgang mit scapingorientierten Datenanalysen wiedergegeben werden, die innerhalb eines entsprechenden Forschungsprojekts gewonnen wurden. Im Projekt sollte mit den technologischen Möglichkeiten des Web Scrapings im Zusammenhang mit einem realen Anwendungsszenario bzw. Forschungsbedürfnis experimentiert werden. Aus fachlicher Sicht ging es beim Projekt um die Bewertung des aktuellen Mediationsangebots im deutschsprachigen Raum[1]. Dementsprechend konnten sowohl technologisch orientierte Erkenntnisse als auch fachlich orientierte Analyseergebnisse gewonnen werden. Im vorliegenden Paper soll bewusst auf beide Ergebnistypen eingegangen werden.

Vergleichbare Analysen wurden innerhalb des Forschungsteams bereits im Diskurs des Software Engineerings (vgl. [Hentschel et. al 2016]) bzw. bei einer Sentimentanalyse zur Kundenzufriedenheit mit den Angeboten der Deutschen Lufthansa und British Airways durchgeführt (vgl. [Hentschel 2015]). Auch für die

[1] Auftraggeber: BAFM e.V. - https://www.bafm-mediation.de

Forschung im Bereich der Psychologie wurde der Einsatz des Web Scrapings unter [Landers 2016] thematisiert. Die Autoren gehen u.a. auf theoretische Vorrausetzungen zum Einsatz des Web Scrapings und auf eine hypothesengesteuerte Vorgehensweise ein, wobei getroffene Annahmen mit Hilfe von Datenanalysen bestätigt oder verworfen werden.

2 Datenquellen im Internet

Im Mittelpunkt des Web Scrapings stehen potentielle Datenquellen innerhalb des Internets. Dementsprechend gilt es, in einem ersten Schritt diese zu identifizieren und unter Berücksichtigung sowohl fachlicher als auch technischer Kriterien zu klassifizieren.

Im Zusammenhang mit einer fachlichen Auseinandersetzung kann zunächst eine sehr einfache bzw. generische Klassifikation im Internet bereitgestellter Informationen sinnvoll sein, die sich auf vielfältige Dienstleistungen übertragen lässt. Mit deren Hilfe können dann z.B. Hypothesen zur Marktpenetration, zu inhaltlichen Schwerpunkten oder auch zur Reife der zu untersuchenden Dienstleistung abgeleitet werden:

- Kontaktinformationen zu (Mediations-) Ansprechpartnern.
- Informationen zu angebotenen (Mediations-) Kompetenzen.
- Hinterlassene Kundenbewertungen zu durchgeführten Mediationen.
- Interessensbedürfnisse an (Mediations-) Leistungen.
- Informationen von (Mediations-) Interessensverbänden.
- Informationen zu (Mediations-) Weiterbildungen.
- Gesetzgeberische Festlegungen.

Über den Einsatz statistischer Methoden zur empirischen Forschung lassen sich angenommene Zusammenhänge bestätigen bzw. verwerfen. In einem weiteren Schritt gilt es, den angenommenen Informationsquellen die technisch realen Möglichkeiten gegenüberzustellen. In diesem Zusammenhang können dann z.B. die folgenden eher technologisch geprägten Informationsquellen identifiziert werden:

- Webauftritte unter Einsatz klassischer HTML-Seiten (z.B. unternehmensspezifische Webseiten).
- Web-Portale, die eigene Such- und Analyseansätze anbieten (z.B. Google oder auch Gelbe Seiten).
- Soziale Netzwerke im Zusammenhang mit beruflichen Interessen (z.B. Twitter, XING, LinkedIn via Open APIs).
- Explizite Datenangebote zu durchgeführten Mediationen (z.B. Open Data: Cross-border family mediation – vgl. https://data.europa.eu).

Entsprechend den technologisch geprägten Möglichkeiten der Datenquellen muss ein Web Scraper mit verschiedenen textlich oder auch grafisch orientierten Datenstrukturen (z.B. HTML, XML, pdf, jpg) umgehen können, wofür auch der Begriff des Parsens (z.B. Texterkennung und Umwandlung) verwendet wird. Darüber hinaus soll der Begriff des Web Scrapings auch auf den Einsatz von Open Data und Open APIs erweitert werden.

3 Existierende Technologieansätze

Der Einsatz eines Web Scrapers erfordert zumeist die Codierung der benötigten Algorithmen mit Hilfe einer Programmiersprache wie z.B. Phyton oder auch Java. Unter [Broucke 2018] findet sich eine detaillierte Auseinandersetzung mit den theoretischen und technischen Grundlagen des Web Scrapings bzw. Beispielen für potentielle Anwendungsszenarien. Im Detail muss ein klassischer Web Scraper die folgenden Funktionen erfüllen:

- Eröffnen einer Internetverbindung und lesen der URL[2].
- Parsen des über die URL bezogenen Dokuments.
- Extrahieren der benötigten Daten bzw. Schlüsselwörter.
- Bereitstellung von Metadaten wie z.B. Häufigkeiten.
- Abspeichern der Daten in einem weiter nutzbaren Format.

Aus Sicht der Autoren lassen sich die folgenden Vorgehensweisen zur Implementierung bzw. Verwendung eines Web Scrapers identifizieren. Ein Anspruch auf Vollständigkeit soll dabei allerdings nicht erhoben werden, zumal bei den aufgezeigten Alternativen potentielle Überlappungen hinsichtlich des angebotenen Funktionsumfangs existieren. Darüber hinaus impliziert die aufgezeigte Reihenfolge eine Abnahme der Freiheitsgrade hinsichtlich des realisierbaren Funktionsumfangs, im Gegensatz dazu aber auch eine Reduktion des Implementierungsaufwands bzw. der benötigten Entwicklerskills.

- ***Programmiersprachen*** wie z.B. Java bieten die Möglichkeit, Verbindungen zu Endpunkten im Internet aufzubauen. Über diese können entsprechende Daten mit Hilfe des http-Protokolls abgerufen werden. Aufwändig ist die komplette Entwicklung der zu automatisierenden URL-Navigation oder auch des „Natural Language Processings – NLP“ zur Dokumentenverarbeitung.
- ***Frameworks*** wie z.B. Scrapy https://docs.scrapy.org bieten die Möglichkeit, vordefinierte Scraping-Funktionen einzusetzen. Bei Scrapy erfolgt die Implementierung so genannter „Spider“ in Python, welche den Zugriff auf einzelne oder mehrere Webseiten aber auch das Parsing der Daten definieren. Eine Übersicht zu potentiellen Ansätzen findet sich unter [Crawlers 2021].

[2] Unified Ressource Locator

- ***Cloudbasierte Scraper APIs*** können in eigene Implementierung ohne Detailkenntnisse des Scrapermechanismus eingebunden werden. Für die Nutzung bedarf es der URL-Parametrisierung zur Festlegung zu berücksichtigender Datenelemente. Eine Übersicht zu diesen APIs findet unter [Geekflare 2020].
- ***ML-Entwicklungsumgebungen*** für Lösungen zum maschinellen Lernen bieten eine weitere Möglichkeit zum Web Scraping. Ein Beispiel für die visuelle Entwicklung (Drag & Drop vordefinierter Komponenten) entsprechender Lösungen findet sich mit dem ML Studio der Fa. Microsoft.
- ***Nutzerorientierte Scrapingansätze*** verfolgen eine manuelle Auswertung von über Webseiten bereitgestellten Informationen oder aber den Einsatz vorgefertigter Such- und Analysemethoden wie diese z.B. unter Google-Trends zur Verfügung gestellt werden.

4 No-Code und Low-Code Web Scraper

Die prototypischen Tests greifen 2 der im vorherigen Kapitel eingeführten Möglichkeiten zum Web Scraping auf. Die Gründe für die Berücksichtigung dieser Ansätze lagen in den zur Verfügung stehenden Projektressourcen, aber auch in den bei den Projektbeteiligten vorhandenen Kenntnissen. Im Detail soll sowohl ein Überblick zu den technologischen Möglichkeiten als auch ein Einblick in erzielte Analyseergebnisse gegeben werden.

5 Analysen mit Google-Trends

Mit Hilfe von Google Trends lässt sich das Suchinteresse an Themen im zeitlichen Verlauf beobachten. Konkret erfolgt dabei ein Web-Scraping der durch die Nutzer durchgeführten Google-Suchanfragen. Entsprechende Filter können den geografischen und zeitlichen Bezug (ab dem Jahr 2004), die fachlich orientierte Kategorie (z.B. Unternehmen und Industrie) oder auch die Art der über Google durchgeführten Suche (z.B. Google-News Suche) einschränken.

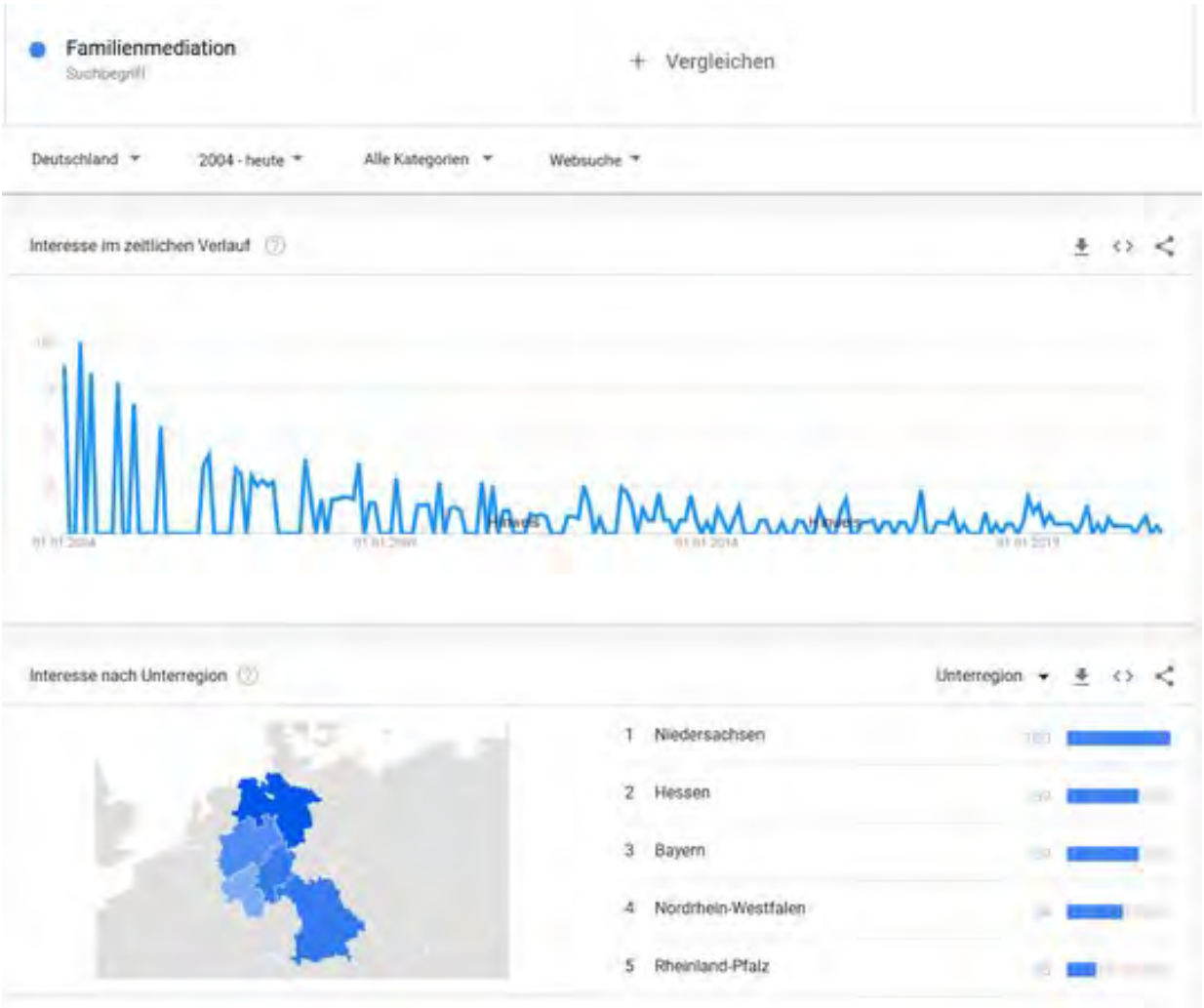


***Abbildung 1**: Interesse am Suchbegriff Familienmediation (inkl. Lokationsbezug) erstellt mit Hilfe von Google-Trends am 08.02.2021 - https://trends.google.de*

Im Falle des hier im Mittelpunkt stehenden Themas der „Familienmediation“ konnte eine abnehmende Tendenz des Suchinteresses seit dem Jahr 2004 festgestellt werden. Darüber hinaus zeigt sich ein potentieller Interessenschwerpunkt in 5 Bundesländern. (siehe Abbildung 1)

5.1 ML Studio der Fa. Microsoft

Das im Folgenden beschriebene Experiment wurde mit Hilfe des ML Studios und der in diesem Rahmen angebotenen Module und Komponenten erarbeitet. Mit Hilfe dieser visuellen Entwicklungsumgebung für ML-Lösungen lassen sich u.a. Sentimentanalysen, Qualitative Inhaltsanalysen oder auch Clusteranalysen durchführen. Die aufgesetzten Experimente dienten der Erprobung einer allgemeinen Vorgehensweise und Identifikation potentieller Problembereiche.

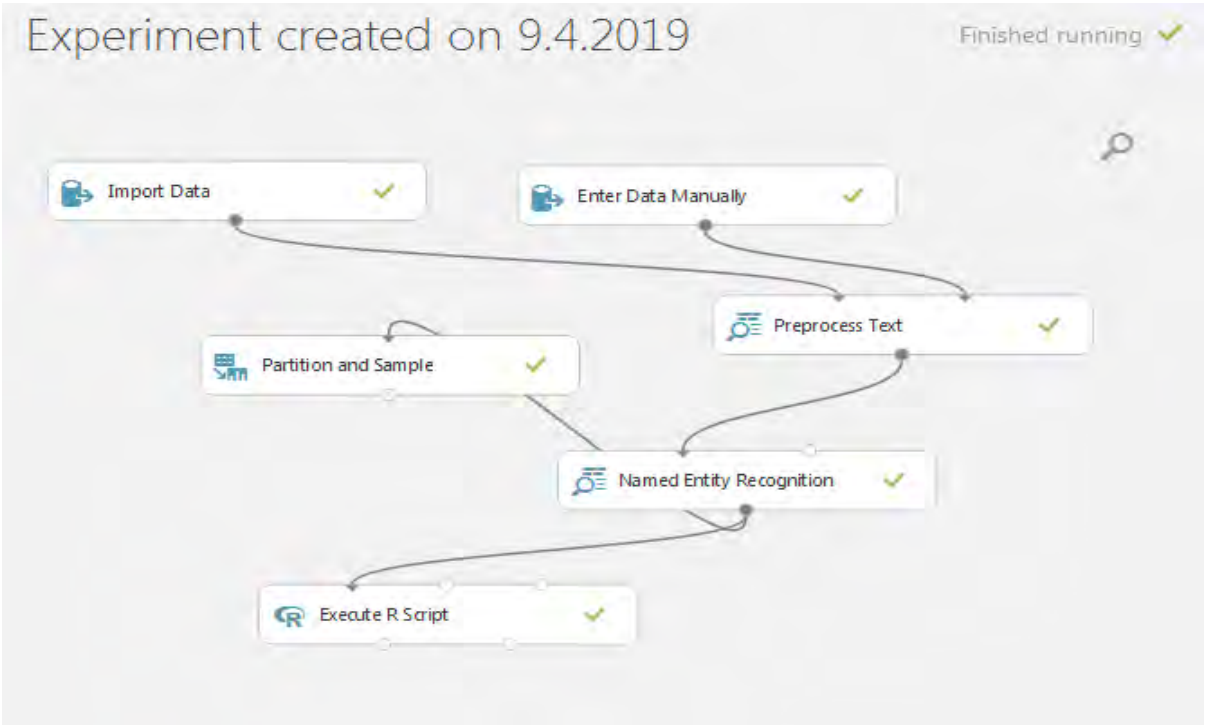


Abbildung 2: *Beispiel einer inhaltsorientierten URL-Analyse erarbeitet mit: https://studio.azureml.net am 09.04.2019*

Das in Abbildung 2 dargestellte Experiment greift auf eine Webseite (URL) mit Hilfe des *Import Data Moduls* zu und liest entsprechende HTML-Inhalte (in diesem Fall exakt eine URL) ein. Diese werden dann einer Vorverarbeitung (Modul - *Preprocess Text*) unterzogen, wobei hier vielfältige Bereinigungen vorgenommen werden. So werden zum Beispiel Sätze erkannt, Sonderzeichen entfernt oder auch Text auf eine festzulegende Sprache (im Beispiel deutsch) eingeschränkt.

Die manuelle Dateneingabe (*Enter Data Manually*) dient der Angabe von so genannten „Stop-Wörtern“. Auch diese werden aus der Datensammlung entfernt. Mit Hilfe des Moduls *Named Entity Recognition* wird schließlich eine semantische Klassifikation (Spaltenbildung) der Datensammlung vorgenommen. Die korrespondierende Visualisierung kann in Abbildung 3 nachvollzogen werden. Ein weiteres Modul wurde auf der Grundlage eines vordefinierten R-Scripts (*Execute R Scrip*) zur Generierung einer Wortwolke verwendet.

Getestet wurde das Experiment unter anderem mit der folgenden URL des Auftragsgebers zum Forschungsprojekt: https://www.bafm-mediation.de

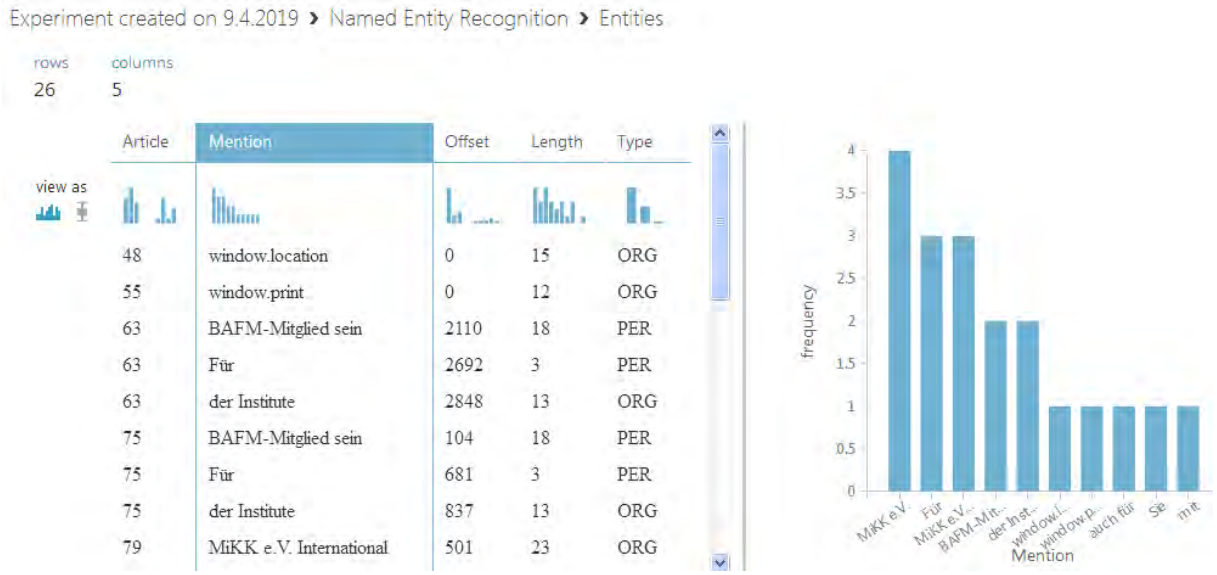


Abbildung 3: *Ergebnis einer durchgeführten URL-Analyse*

Grundsätzlich sind die Möglichkeiten einer visuellen Erstellung von Web Scraping Projekten positiv zu bewerten, da diese bei allen Beteiligten zu einem schnelleren Erkenntnisgewinn führen. Dennoch konnte die aktuelle Lösung des ML-Studios im Zusammenhang mit den speziellen Anforderungen des Projekts nicht überzeugen. Im Detail ergaben sich die folgenden Problemstellungen:

- Probleme bzw. Laufzeitfehler im Umgang mit http-Redirects zur Weiterleitung von URL-Aufrufen.
- Unzureichende Performanceeigenschaften auf PC-basierten Geräten, insbesondere in Bezug auf die Ausführung der Modelle.
- Lange Round-Trip Zeiten, d.h. Konfigurationen am Modell führten zu langen Wartezeiten auf die Modellergebnisse.
- Fehleranfälligkeiten beim Umgang mit großen Datenmengen, die über das Internet gezogen werden.
- Begrenzte Möglichkeiten zur projektspezifischen Anpassung der vordefinierten Module.
- Begrenzte Möglichkeiten zur algorithmischen Modellsteuerung und Automatisierung neuer Rahmenbedingungen bei der Analyse.

Die aufgezeigten Probleme im Zusammenhang mit dem ML Studio wurden auch mit Vertretern der Fa. Mircosoft thematisiert. Dementsprechend lassen sich diese durch die originären ML-Ansätze unter Azure lösen.

5 Originäre Java-Web Scraper Entwicklung

5.1 Details zur technischen Implementierung

Ein weiterer Lösungsansatz wurde mit der prototypischen Implementierung eines einfachen Java-Web Scrapers verfolgt. Zur Vermeidung einer „steilen Lernkurve“ wurde bewusst auf den Einsatz von Scraper-Frameworks bzw. Scraper–Cloud-APIs verzichtet und eine native Java-Implementierung verfolgt.

Der Funktionsumfang bezog sich auf das automatische Einlesen vorab zu konfigurierender URLs, die schlüsselwortorientierte Auswertung eingelesener HTML-Dokumente, die Identifikation der Häufigkeiten identifizierter Schlüsselworte und die Ablage der Analysedaten in einem textbasierten Zwischenformat. Für die Visualisierung der Häufigkeiten mit Hilfe von z.B. Tag Clouds oder auch Ableitung von Histogrammen sollten ggf. weitere Werkzeuge hinzugezogen werden. Der Quellcode zum Projekt wurde entsprechend der Abbildung 4 unter GitHub-Verwaltung gestellt. Daher kann dieser von potentiell Interessierten eingesehen und ggf. an neue Anforderungen angepasst werden.

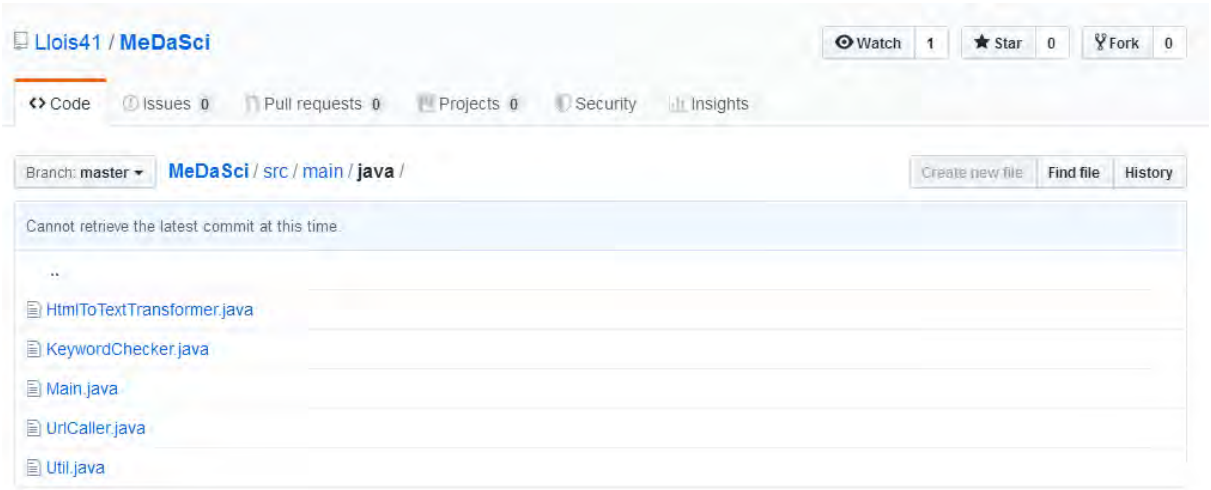


Abbildung 4: *GitHub Repository zum Projekt*

Die konzeptionelle Überlegung bestand zunächst darin, URLs von Mediationsanbietern (d.h. deutschsprachige Webauftritte) automatisiert durchsuchen zu lassen, um einen Überblick über deren angebotene Leistungen zu gewinnen. Aufgrund des Zeitrahmens und der limitiert zur Verfügung stehenden Ressourcen wurde die ursprüngliche Idee, einen funktional umfänglichen Web Scraper zu implementieren, dahingehend angepasst, dass das auf Java basierende Programm lediglich vorab spezifizierte URL-Pfade (d.h. bekannte Linkmechanismen) durchsucht. Das automatische Verfolgen ggf. erst beim Einlesen erkannter Pfade ist so allerdings möglich, was die Einsatzmöglichkeiten eines solchen Systems begrenzt.

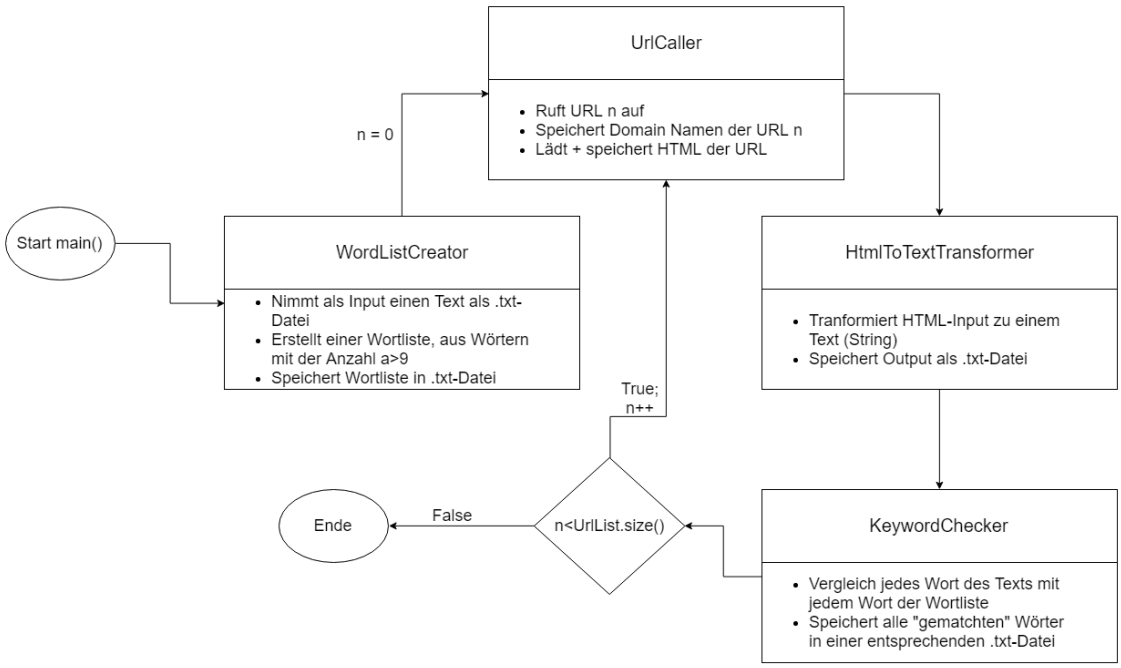


Abbildung 5: *Kernarchitektur des Java-Web Scrapers*

Die Funktionsweise des „Web Scrapers“ ist im Code dokumentiert und kann dementsprechend über das GitHub-Projekt nachvollzogen werden. Im Folgenden sollen die wesentlichen Eigenschaften (funktionale Verantwortlichkeiten der Klassen bzw. Objekte) der Lösung (vgl. Abbildung 5) beschrieben werden.

Im Sinne eines „Normals“ erfolgt zunächst die Erstellung einer Wortliste (*WordListCreator*), wobei eine vorab zu definierende Häufigkeit der berücksichtigten Wörter festgelegt werden kann. Im Projekt wurde zur Erstellung bzw. Plausibilisierung dieser Referenz einzelne Kapitel eines im deutschsprachigen Raum anerkannten Standardwerks zur Mediation verwendet. Nach der Erstellung der Wortliste werden die Liste der URLs eingelesen (*URLCaller*) und die einzelnen URL iterativ aufgerufen. In einem weiteren Schritt erfolgt die Überführung des HTML-Dokuments (*HtmlToTextTransformer*) in ein korrespondierendes Textdokument, wobei entsprechende HTML-Tags entfernt werden. In einem letzten Schritt erfolgt die Identifikation der themenspezifischen Treffer (*KeywortChecker*) unter Berücksichtigung der erstellten Wortliste.

Nach dem Durchlauf des Programms wird in einem Ordner „output“ zu jeder analysierten URL eine .txt-Datei angelegt, welche alle themenspezifischen Wörter enthält. In der zugrunde liegenden Implementierung müssen diese noch händisch in einen Tag-Cloud-Generator[3] eingefügt werden. Eine Idee für eine zukünftige Weiter-/ Neuentwicklung ist die Anbindung einer entsprechenden Lösung per API.

[3] z.B. https://www.jasondavies.com/wordcloud

Dabei entsteht aus jeder .txt-Datei exakt eine Wortwolke, welche grafisch den Inhalt der URL widerspiegelt. Zudem wird aus allen .txt-Dateien am Ende noch eine kommulative Wortwolke erstellt, um ein Gesamtbild der untersuchten Quellen zu gewinnen.

5.2 Fachlicher Test des Web Scrapers

Für den initialen Test und die fachlich orientierte Validation des Web Scrapers wurde auf die Webseiten der beim BAFM e.V. (https://www.bafm-mediation.de) gelisteten 15 Ausbildungsinstitute zurückgegriffen. Im Ergebnis konnten potentielle Themenschwerpunkte der Ausbildungsinstitute, aber auch die sehr uneinheitliche Darstellung des Selbstverständnisses der beteiligten Institute im Rahmen der jeweiligen Webauftritte identifiziert werden. Abbildung 6 enthält die kumulative Darstellung über allen Instituten:

Abbildung ***6****: Cloud-Wolke zu den Themenschwerpunkten der BAFM-Institute*

Darüber hinaus stellte sich die Frage, welche Webauftritte potentieller Mediatoren im deutschsprachigen Diskurs zu berücksichtigen sind. Für eine Auswahl wurde die XING-Gruppe der Wirtschaftsmediatoren herangezogen, die zum Zeitpunkt der Untersuchung ca. 8.500 Mitglieder enthielt. Von diesen nutzen ca. 10% die Möglichkeit zur Angabe eines eigenen Webauftritts bzw. einer dazu korrespondierenden URL. Dementsprechend ergaben sich ca. 850 Einträge, von

denen 20 URLs als Stichprobe ausgewählt wurden. Die Auswahl berücksichtigte jeweils 10 Profile mit Nachnamen am Anfang und am Ende des Alphabets.

Eine weitere Grundlage zur Auswahl von Webauftritten bildeten die „Gelben Seiten“ - https://www.gelbeseiten.de). Zum Zeitpunkt der Analyse fanden sich dort knapp 4000 Kontaktinformationen zur Themenstellung der Mediation. Eine weitere Einschränkung auf den Begriff der Familienmediation ergab 137 potentielle Webauftritte, die zur Auswahl von 20 URLs herangezogen wurden. Entsprechende Mediationsangebote konnten auf verschiedene Branchen entsprechend der aufgezeigten Häufigkeiten skaliert werden. Im Detail wurden Rechtsanwälte (8), Mediatoren (7), Psychologen (2), Familienrechtler (1), Coaches (1) aber auch die Ergetherapie (1) herangezogen. Die Zahlen in den Klammern entsprechen den für die Analyse berücksichtigten Webauftritten.

Auch in Bezug auf Twitter erfolgte eine Analyse unter den Stichwörtern Mediation bzw. Familienmediation. Dafür wurden zwei dort verzeichnete Webauftritte mit Hilfe des entwickelten Web Scrapers analysiert. Auffällig waren die wenigen aktiven Nutzer (ca. 10 im betrachteten Zeitraum), welche thematisch adäquate Tweets produzierten bzw. konsumierten. Ggf. lassen diese Erkenntnisse den Schluss zu, dass es sich bei Mediatoren eher um ältere bzw. gering technikaffine Anbieter handelt.

5.3 Auswahl fachlicher Ergebnisdiskussionen

Innerhalb der Anlage können zwei generierte Cloud-Wolken exemplarisch nachvollzogen werden. Dabei handelt es sich um die jeweils kumulierten Darstellungen über alle URLs durchgeführter Häufigkeitsanalysen. Im Detail wurden mehr als 50 derartige Häufigkeitsanalysen durchgeführt, wobei signifikante Unterschiede in Bezug auf den Informationsgehalt der Ausführungen zum Themengebiet der Mediation bzw. Familienmediation festgestellt werden konnten.

- Es existiert keine strukturierte Darstellung der Mediationsangebote in Deutschland.
- Begrifflichkeiten im Diskurs der Mediation werden diversifiziert und zum Teil widersprüchlich verwendet.
- Mediation wird häufig im Zusammenhang mit juristischen Angeboten genannt („gibt es ggf. auch noch“).
- Der Begriff der Mediation wird für Werbezwecke eingesetzt; kein stringentes Verfolgen ggf. definierter Überschriften.
- Aktuell kann auf eine geringe Reife der Skillprofile entsprechender Berufsgruppen geschlossen werden.

- Für Interessenten sind entsprechende Angebote im Web nur gering vergleichbar (Aufwand/Nutzen).
- Nur wenige Anbieter können „offensichtlich“ ausschließlich von Aufgaben der Mediation leben.
- Erfahrungen zu durchgeführten Mediationen sind im Web nicht existent (Messen des Erfolgs einer Mediation?)

Die durchgeführten Analysen geben primär die Sicht der Anbieter entsprechender Leistungen wieder; potentielle Bewertungen zu durchgeführten Mediationen lassen sich aktuell nur ansatzweise bzw. sehr allgemein auffinden. Aus diesem Sachverhalt lässt sich allerdings eine Anforderung nach mehr Transparenz für dieses Berufsbild bzw. für den Bedarf des Aufbaus von Erfahrungsdatenbanken ableiten. Zu klären sind die damit einhergehenden Rahmenbedingungen.

6 Bewertung des Java Web Scrapers

Die prototypische Implementierung sollte schnell zu Ergebnissen und darauf aufbauend zu fachlichen und technologischen Feedbacks führen. Nicht im primären Fokus waren die Güte der verwendeten Softwarearchitektur des Web Scrapers bzw. inwieweit generische Lösungsansätze verwendet wurden. Dem- entsprechend zeigten sich vielfältige technische Schulden, wie z.B.:

- Die Pfadangaben für Input- und Output-Dateien wurden direkt im Quellcode abgelegt. Bei aufsetzenden Projekten sollen Konfigurationsdateien (z.B. URL-Pfade auslagern) verwendet werden.
- Statistische Auswertungen und grafische Darstellungen wurden mit Hilfe weiterer Werkzeuge realisiert. Vorgesehen ist die Nutzung entsprechender APIs bzw. einer statistischen Programmiersprache wie R.
- Die aktuelle Implementierung kann lediglich explizit angegebene URLs scrapen, d.h. Webseiten können nicht automatisiert „komplett“ durchsucht werden – kein Navigieren in den Pfaden.
- Die Verarbeitung von Umlauten erfolgt aktuell in der Form, dass diese in zwei Buchstaben umgeformt werden (z.B. ß → ss, ä → ae). URLs, die Umlaute enthalten, können daher nicht eingelesen werden.
- Die benötigten Algorithmen zum Natural Language Processing (kurz NLP) können ohne Einsatz entsprechender Bibliotheken bzw. APIs nur bedingt abgebildet werden.
- Die Lemmatisierung bzw. das Stemming als Techniken zum Normalisieren von Texten zur Ableitung von Grundformen der zu verarbeitenden Wörter funktionieren nur rudimentär. Entsprechende Bibliotheken finden sich z.B. unter https://nlp.stanford.edu (Stanford NLP).

- Ggf. benötigt der autorisierte Zugriff auf einzusetzende URLs die Einbindung von Zertifikaten im Zusammenhang mit den auf der Client-Seite eingesetzten Java-Laufzeitumgebungen.

Beim Web Scraping sind Fragen der Datensicherheit, des Datenschutzes und des Urheberrechts zu berücksichtigen bzw. zu gewährleisten. Die durchgeführten Analysen erfolgten ausschließlich unter Verwendung frei zugänglicher Datenelemente. Ebenso wurden die durch den Webseitenbetreiber eingesetzter Sicherheitsmechanismen respektiert. Ein interessanter Beitrag zum Urheberrecht findet sich unter [Klawonn 2020] ebenso wie das folgende Zitat:

> „Im Regelfall ist Web Scraping für die empirische Forschung rechtlich zulässig. Die Nutzungsbedingungen, die häufig verwendet werden, ändern daran nichts. Anders sieht es mit technischen Sperren aus, die nicht umgangen werden dürfen."

7 Zusammenfassung und Ausblick

Entscheidend für den Projekterfolg waren die zyklisch durchgeführten Abstimmungen mit dem Auftraggeber bzw. der zum Projektabschluss öffentlich durchgeführte Workshop (ca. 30 Teilnehmer). Mit Hilfe dieser Abstimmungen konnten die erzielten Zwischen- und Endergebnisse aus der fachlichen Perspektive diskutiert und soweit wie möglich interpretiert werden. Darüber hinaus war es möglich, die Implementierung des Web Scrapers sukzessive an die Projekterfordernisse anzupassen. Das eher pragmatisch gewählte Vorgehen unter der Verwendung einfacher Instrumente zum Web Scraping gewährleistete die Möglichkeit, Analyseergebnisse schnell aber auch nachhaltig bereitstellen zu können. So konnten ein grundlegendes Verständnis bei allen Projektbeteiligten herausgearbeitet und weiterführende Analyseanforderungen identifiziert werden. Zur Umsetzung der neuen Anforderungen bedarf es allerdings des Einsatzes von Sraping-Frameworks bzw. Scraping-Cloud-APIs. In diesem Zusammenhang laufen aktuell vertiefende Forschungsprojekte bei den Autoren.

8 Quellen

[Broucke 2018] vanden Broucke, S.; Baesens, B.: Practical Web Scraping for Data Science - Best Practices and Examples with Python, Springer Science+Business Media, New York, 2018

[Crawlers 2021] 50 Best Open Source Web Crawlers, https://prowebscraper.com/blog/50-best-open-source-web-crawlers, letzter Abruf: 09. Februar 2021

[Geekflare 2020] 11 Beliebte Cloud-basierte Web Scraping-Lösungen, Geekflare Editorial, November 27, 2020

[Hentschel 2015] Hentschel, J.; Neumann, R.; Schmietendorf, A.: Mehrwertpotentiale von Big Data Lösungen – Kreativität entscheidet über die Ergebnisse, in SQ Magazin 37 – Themenschwerpunkt „BIG DATA – Wem nützt der Datenberg?“, Arbeitskreis Software-Qualität und -Fortbildung (ASQF), S. 14-15, Dezember 2015

[Hentschel 2016] Hentschel, J.; Schmietendorf, A.; Dumke, R.: Big Data benefits for the Software Measurement Community, in Proc. International Workshop on Software Measurement (IWSM) and the International Conference on Software Process and Product Measurement (Mensura), IEEE Computer Society Conference Publishing Services (CPS), S.108-114, October 2016

[Klawonn 2020] Klawonn; T.: Urheberrecht - Grenzen des "Web Scrapings", Forschung und Lehre, https://www.forschung-und-lehre.de/recht/grenzen-des-web-scrapings-2421, Januar 2020, letzter Abruf: 09. Februar 2021

[Landers 2016] Landers, R.; Brusso, R.; Cavanaugh, K.; Collmus, A. B.: A Primer on Theory-Driven Web Scraping: Automatic Extraction of Big Data From the Internet for Use in Psychological Research, Psychological Methods, 21, S. 475-492, 2016

9 Anlage Exemplarische Auswertungen

Abbildung 7: *Häufigkeiten identifizierte Begriffe (Gesamtauswertung XING)*

Abbildung 8: *Häufigkeiten identifizierte Begriffe (Gesamtauswertung Gelbe Seiten)*

10 Dank

Die technische Implementierung des hier beschriebenen Java Web Scrapers verantwortete im Projekt Herr Luis-Raphael Ramer (zum Zeitpunkt des Projekts Studierender an der HWR Berlin). Für seine ausgezeichnete und kreative Auseinandersetzung mit den Projektanforderungen sei ihm an diese Stelle noch einmal ausdrücklich gedankt!

Andreas Schmietendorf

ESAPI 2020 – 4. Workshop Evaluation of Service-APIs

Shaker-Verlag, Aachen, November 2020, ISBN 978-3-8440-7515-1

Das vorliegende Buch fasst die insgesamt 11 Beiträge und Diskussionen des 4. Workshop zur Bewertung von service-basierten APIs zusammen und ist in der Buchreihe der Schriften zu modernen Integrations-architekturen erschienen.

Hartenstein/Nadobny/Schmidt/ Schmietendorf:

Sicherheits- und Compliance Management

Logos-Verlag, Berlin, 2020
ISBN 978-3-8525-5086-8

This book describes approaches and techniques for implementing Web APIs keeping security-related requirements. The API management involves analytical and constructive approaches for quality assurance during the development. The DevOps approach was considered in the context of business processes.

Software Metrics: A Complete Guide - 2021 Edition

Gerardus Blokdyk and Publishers, 2021
ISBN 978-1-8674-9201-6

This book summarizes essential software project and management metrics and their application to practical and industrial areas and examples.

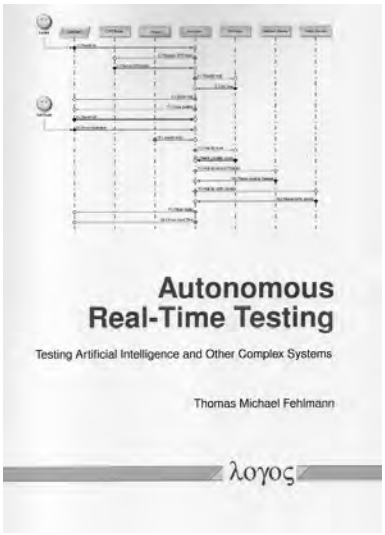
Schmietendorf, A.:

Enterprise Computing Conference 2020

Köln, März 2020

Shaker Verlag, Aachen, 2020,
ISBN 978-3-8440-7320-1

Dieses Buch beinhaltet die Beiträge zur ECC-Konferenz 2020 zur Thematik „Enterprise Transformation“ vor allem in relevanten Anwendungsbereichen.



Thomas M. Fehlmann:

Autonomous Real-Time Testing

Testing Artificial Intelligence and Other Complex Systems
Logos-Verlag, Berlin, 2020
ISBN 978-3-8525-5086-8

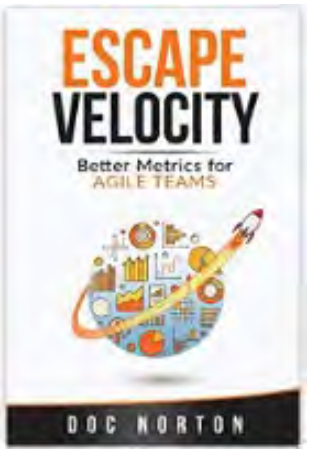
The book explains the theory and the implementation approach for a framework for Autonomous Real-Time Testing (ART) of a software-intense system while in operation. Principles and approaches like Combinatory logic,Analytic Hierarchy Process (AHP) and Quality Function Deployment (QFD) are used for a complex testing approach of real-time systems like automotive solutions, IoT control software and embedded system releases.

Andreas Schmietendorf

Empirische Untersuchungen zum Cloud-Einsatz im KMU-Bereich - eine zusammenfassende Betrachtung

Shaker-Verlag, Aachen, April 2020, ISBN 978-3-8440-7356-0

Das vorliegende Buch reflektiert die Ergebnisse von forschungs- aber auch industrieorientierten Projekten rund um die Themenstellung des Cloud Computings, die durch den Autor initiiert und in den vergangenen 10 Jahren verantwortet bzw. im Rahmen seiner Arbeitsgruppe bearbeitet wurden.



Doc Norton

Escape Velocity: Better Mertrics for Agile Teams

Februar 2020

This book identifies the velocity as the most commonly used metric in agile software delivery. The efficiency of Scrum teams is the main focus. Metrics are considered in general and further measure like lead time, team joy, team performance etc. are proposed especially. The book includes many interesting stories of agile team management.

Elyjoy Muthoni Micheni

Metrtics and Models for Evaluating the Quality and Effectiveness of ERP Software

Juli 2019

This book presents a set of theoretical measurement models and metrics for measuring software size and complexity of large scale enterprise resource planning software based on practical experiences. It focuses on the measurement of usability, service quality, security, interoperability, maintenance and enterprise resource planning.

Joachim Rosberg

Agile Project Management with Azure DevOps: Concepts, Templates, and Metrics

April 2019

This book considers Agile project management to use and customize Microsoft Azure DevOps. The basic process involves the Application Life Cycle Management approach and achieve an overall higher quality output.“

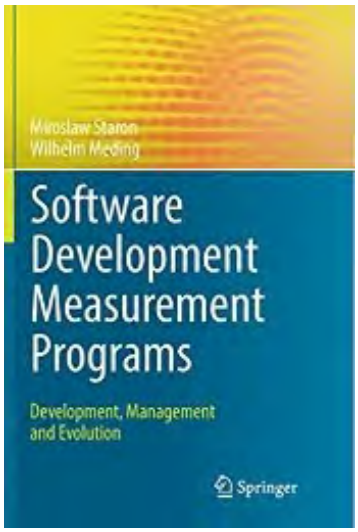
Schmietendorf, A.:

Workshop ESAPI 2019

Dresden, November 2019

Shaker Verlag, Aachen, 2019,
ISBN 978-3-8440-6837-5

Dieses Buch beinhaltet die Beiträge zur ESAPI-Konferenz 2019 zu Sicherheits- und Compliance-aspekten von Web-APIs vor allem in relevanten Anwendungsbereichen.



Miroslaw Staron:

Software Development Measurement Programs

Springer Publ., 2019
ISBN 978-3030063085

This book describes approaches and techniques for implementing software measurement processes from a practical point of view involving toll support, project integration and measurement programs evolution.

Schmietendorf, A.,:

ESAPI 2018

2. Workshop: Evaluation of Service-APIs
8. November 2018, München

Shaker Verlag, Aachen, April 2018, ISBN 978-3-8440-6254-0

The book includes the proceedings of the Evaluation of Service-APIs 2018 Workshop held in Munich in November 2018, which constitute a collection of theoretical studies in the field of measurement and evaluation of service oriented and API technologies.

Gerardus Blokdyk:

Software Measurement the Ultimate Step-By-Step Guide

5starcooks Publ. 2018

This book summarizes some helpful practical experiences about measurement integration in software management processes and their successful implementation.

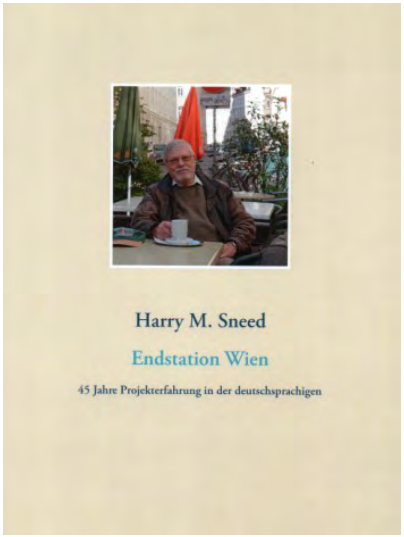
Schmietendorf, A., Nitze, A.:

ESAPI 2018

2. Workshop: API-First/API-Management
19. April, Hamburg

Shaker Verlag, Aachen, April 2018, ISBN 978-3-8440-5927-4

The book includes the proceedings of the API-First/API-Management 2018 Workshop held in Hamburg in April 2018, which constitute a collection of theoretical studies in the field of measurement and evaluation of service oriented and API technologies.



Harry Sneed:

Endstation Wien

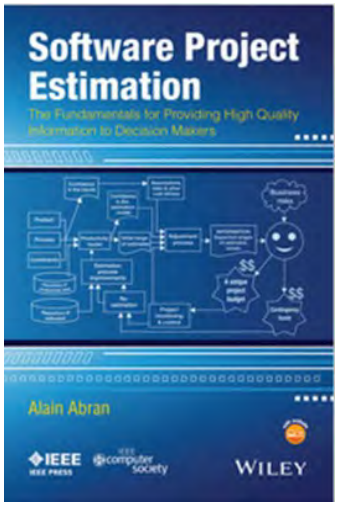
45 Jahre Projekterfahrungen in der deutschsprachigen IT-Welt
BoD Norderstedt, 2017, 328 S.
ISBN 978-3-7448-8364-1

Dieses Buch beschreibt nahezu die gesamte Tätigkeit von Harry Sneed in der IT-Welt, von den Anfängen der Großrechner mit den COBOL und PL/1-Programmen bis hin zu den aktuellen und modernen Ansätzen Service-orientierter Technologien und Systemen. Dieses Buch fasst vor allem die umfangreichen Erfahrungen zu Wartungs-, Migrations- und Testprojekten zusammen, die auch für die Beherrschung aktueller und moderner Software-Anwendungen, von unschätzbarem Wert sind.

Staron, M, Melding, W.:

Proceedings of the IWSM/Mensura 2017

Joined Conference of the 27th International Workshop on Software Measurement (IWSM) and the 12th International Conference on Software Process and Product Measurement (Mensura), ACM 2017, *ISBN 978-1-4503-4853-9*
This proceedings are available at the Computer Science Bibliography of Trier.



Abran, A.:

Software Project Estimation: The Fundamentals for Providing

High Quality Information to Decision Makers

Wiley IEEE Computer Society Press, 2015 (288 pages), ISBN 978-1-118-95408-9

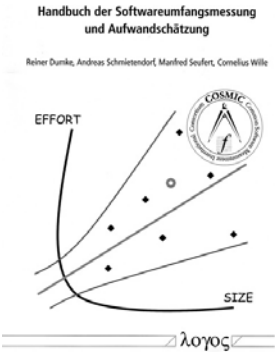
This book introduces theoretical concepts to explain the fundamentals of the design and evaluation of software estimation models. It provides software professionals with vital information on the best software management software out there. End-of-chapter exercises, Over 100 figures illustrating the concepts presented throughout the book, Examples incorporated with industry data.

Please remember:

Dumke, R., Schmietendorf, A., Seufert, M., Wille, C.:

Handbuch der Softwareumfangsmessung und Aufwandschätzung

Logos Verlag, Berlin, 2014 (570 Seiten), ISBN 978-3-8325-3784-5



This book shows an overview about the current software size measurement and estimation approaches and methods. The essential part in this book gives a complete description of the ***COSMIC measurement method****, their application for different systems like embedded and business software and their use for cost and effort estimation based on this modern ISO size measurement standard.*

Software Measurement & Data Analysis Addressed Conferences

January 2021

BigDataSE 2021:	**14th IEEE International Conference on Big Data Science and Engineering** December 29, 2020 – January 1, 2021, Guangzhou, China see: http://www.ieee-trustcom.org/BigDataSE2020/
CPP 2021:	**10th ACM SIGPLAN International Conference on Certified Programs and Proofs** January 17-19, 2021 **as Virtual meeting**, USA see: https://popl21.sigplan.org/home/CPP-2021
SWQD 2022:	**Software Quality Days** **2021 canceled because of Corona** January 18-20, 2022, Vienna, Austria see: https://www.software-quality-days.com/
SOFSEM 2021:	**International Conference on Current Trends in Theory and Practice of Informatics** January 25-28, 2021, Bolzano-Bozen, Italy see: http://www.guide2research.com/conference/sofsem-2021

February 2021

ISEC 2021:	**14h Innovation in Software Engineering Conference** February 25-27, 2021, Bhubaneswar, India, **as Virtual meeting** see: https://isoft.acm.org/isec2021
Smart Data Car Data 2021:	**Automobilwoche Konferenz** to be organized: 2021, Munich, Germany see: https://www.automobilwoche-konferenz.de/vormerken.php
ICEASE 2021:	**International Conference on Evaluation and Assessment in Software Engineering** February 22 - 23, 2021, Paris, France see:https://waset.org/evaluation-and-asssessment-in-software-engineering-conference-in-february-2021-in-paris
ICQAST 2021:	**International Conference on Quality Assurance and Software Testing** February 15 – 16, London, United Kingdom see: https://waset.org/quality-assurance-conference-and-software-testing-conference-in-february-2021-in-london
SE 2021:	**Software Engineering** February 22 – 24, 2020, Braunschweig, Germany see: https://se-2021.tu-bs.de/
ICBDM 2021:	**International Conference on Big Data in Management** February 26 – March 1, 2021, Shenzhen, China see: https://www.icbdm.org

March 2021

ASQ 2021:	**Lean and Six Sigma Conference** March 1 - 4, 2021, **as Virtual meeting**, USA see: https://asq.org/conferences/six-sigma/
ICWS 2021:	**International Conference on Web Services** March 25 - 26, 2021, Madrid, Spain see: https://waset.org/web-services-conference-in-march-2021-in-madrid
ICSA 2021:	**IEEE International Conference on Software Architecture** March 22 -26, 2021, **as Virtual meeting,** Stuttgart, Germany **see:** https://icsa-conferences.org/2020/index.html
Programming 2021:	**Programming 2021** March 22 - 26, 2021, **as Virtual meeting**, United Kingdom see: https://2021.programming-conference.org/
ICDSE 2021:	**International Conference on Data and Security Engineering** March 22 – 23, 2021, Istanbul, Turkey sdee: https://waset.org/data-and-security-engineering-conference-in-march-2021-in-istanbul

April 2021

ETAPS 2021:	**European Join Conference on Theory & Practice of Software** March 27 – April 1, 2021, **as Virtual meeting** see: https://etaps.org2021/fase
ODSC East 2021:	**Open Data Science Conference East** March 30 – April 1, 2021, **as Virtual meeting,** USA **see:** https://www.odsc.com/boston/
A-MOST 2021	**Advances in Model-Based Software Testing** April 12, 2021, **as Virtual meeting**, see: https://icst2021.icmc.usp.br/home/a-most-2021
ICST 2021:	**IEEE International Conference on Software Testing, Verification & Validation** April 12 - 16, 2021, **as Virtual meeting**, see: http://icst2021.icmc.usp.br
REFSQ 2021:	**International Working Conference on Requirements Engineering: Foundation for Software Quality** April 12 - 15, 2021, **as Virtual meeting** see: https://2021.refsq.org/
SOFTENG 2021:	**International Conference on Advances and Trends in Software Engineering** April 18 - 20, 2021, Porto, Portugal see: https://www.iaria.org/conferences2021/ProgramSOFTENG21.html
ICPE 2021:	**ACM/SPEC International Conference on Performance Engineering** April 19 – 23, 2021, Rennes, France see: https://icpe2021.spec.org/

FASE 2021:	**International Conference on Fundamental Approaches to Software Engineering** March 27 – April 1, 2021, **as Virtual meeting** see: https://www.etaps.org/2021/fase
ICGC 2021:	**International Conference on Grid and Clouds** April 19 – 20, 2021, Paris, France see: https://waset.org/grids-and-clouds-conference-in-april-2021-in-paris
FG-GI Workshop 2021:	**Datenökonomie – Wie schaffe ich Wert mit meinen Daten?** April 23, 2021, **as Virtual meeting** see: https://fg-metriken.gi.de/
STAREAST 2021:	**Software Testing Analysis & Review Conference** April 25 - 30, 2021, Orlando, FL, USA see: http://stareast.techwell.com/
ENASE 2021:	**16th International Conference on Evaluation of Novel Approaches to Software Engineering** April 26 - 27, 2021, **as Virtual meeting** see: http://www.enase.org/

May 2021

ICDMCC 2021:	**International Conference on Data Mining and Cloud Computing** May 3 – 4, 2021, Singapore, Singapore see: https://waset.org/data-mining-and-cloud-computing-conference-in-may-2021-in-singapore
CIbSE 2021:	**Iberoamerican Conference on Software Engineering** May 4 - 8, 2021, San Jose, Costa Rica see: http://cbiseconference.org
ICESDIS 2021:	**International Conference on E-Science and Data Intensive Science** May 13 – 14, 2021, Amsterdam, Netherlands see: https://waset.org/e-science-and-data-intensive-science-conference-in-may-2021-in-amsterdam
IoT 2021:	**International online Conference on Internet of Things and its Application** May 19 – 20, 2021, **as Virtual meeting,** see: http://iot2021.ui.ac.ir/en/
ASQ 2021:	**World Conference on Quality and Improvement** May 24 - 28, 2021, **as Virtual meeting,** USA see: https://asq.org/conferences/wcqi
ICWE 2021:	**International Conference on Web Engineering** May 18 - 21, 2021, Biarritz, France see: https://icwe2021.webengineering.org/
ICPC 2021:	**International Conference on Program Comprehension** May 28 - 29, 2021, **as Virtual meeting,** see: http://conf.researchr.org/home/icpc-2021

SEAMS 2021: **International Symposium on Software Engineering for Adaptive and Self-Managing Systems**
May 23 - 24, 2021, **as Virtual meeting,** Madrid, Spain
see: https://conf.researchr.org/home/seams-2021

ICSE 2021: **43th International Conference on Software Engineering**
May 23 - 29, 2021, **as Virtual meeting,** Madrid, Spain,
see https://conf.researchr.org/home/icse-2021

ICAMDS 2021: **International Conference on Applied Mathematics and Data Science**
May 29- 31, 2021, Wuhan, China
see: http://www.icamds.com/

OSS 2021: **International Conference on Open Source Systems**
May 12 - 13, 2021, Lahti, Finland
see: https://www.oss2021.org

MSR 2021: **Conference on Mining Software Repositories**
May 23 - 24, 2021, Madrid, Spain
see: https://2021.msrconf.org/

WTMC 2021: **6th International Workshop on Traffic Measurements for Cybersecurity**
May 27, 2021, **as Virtual meeting**,
see: https://wtmc.info

June 2021

IMMM 2021: **International Conference on Advances in Information Mining and Management**
May 30 – June 03, 2021, Valencia, Spain
see: https://www.iaria.org/conferences2021/IMMM21.html

ICIOT 2021: **International Conference on Internet of Things**
June 3 - 4, 2021, New York, USA
see: https://waset.org/internet-of-things-conference-in-june-2021-in-new-york

VDA Automotive SYS 2021: **Quality Management for Automotive Software-based Systems and Functionality**
June 6 - 7, 2021, Potsdam, Germany
see: https://vda.de/de/services/veranstaltungen/sys-conference.html

EJC 2021: **International Conference on Information Modeling and Knowledge Bases**
June 7 - 11, 2021, Hamburg, Germany
see: https://ejc.entavis.com

ODSC 2021: **Open Data Science Conference Europe**
June 8 - 10, 2021, **as Virtual meeting**,
see: https://www.ideaconnection.com/conferences/5167-ODSC-Europe-Virtual-Conference-2021.html

SEAI 2021: **International Conference on Software Engineering and Artificial Intelligence**
June 11 – 13, Xiamen, China
see: http://www.seai.org/

XP 2021:
International Conference on Agile Software Development
June 14 - 18, 2021, **as Virtual meeting,**
see: https://www.agilealliance.org/xp2021/

SDS 2021:
Swiss Conference on Data Science
June 9, 2021, Luzern, Switzerland
see: https://www.sds2021.ch/

BigData 2021:
Big Data Conferences
June (January to December) 2021, **mostly as Virtual meeting**,
see: https://conferenceindex.org/conferences/big-data

ECC 2021:
Enterprise Computing Conference (ECC)
June 29 . July 2, 2021, Rotterdam, Netherlands
see: https://ecc21.euca-ecc.org

July 2021

ISSTA 2021:
International Symposium on Software Testing and Analysis
July 12 - 16, 2021, Aarhus, Denmark
see: https://conf.researchr.org/home/issta-2021

BIGDACI 2021:
International Conference on Big Data Analytics, Data Mining and Computational Intelligence
July 20 - 253 2021, **as Virtual meeting**,
see: https://bigdaci.org/

MCCSIS 2021:
Multiconference on Computer Science and Information Systems
July 20 - 23, 2021, **as Virtual meeting**,
see: https://mccsis.org/

ICSOFT 2021:
International Conference on Software Technologies
July 7 - 8, 2021, **as Virtual meeting**,
see: http://www.icsoft.org/

ICML 2021:
International Conference on Machine Learning
July 18 -24, 2021, Vienna, Austria
see: https://icml.cc/Conferences/2021

AGILE 2021:
Annual North American Agile Conference
July 19 – 22, 2021, **as Virtual meeting**,
see: https://www.agilealliance.org/agile2021/

CSCE 2021:
World Congress on Computer Science and Engineering
July 7 - 9 2021, London, UK
see: https://www.iaeng.org/WCE2021

ICESGC 2021:
International Conference on e-Science and Grid Computing
July 12 – 13, 2021, Ottawa, Canada
see: https://waset.org/e-science-and-krid-computing-conference-in-july-2021-in-ottawa

ICNMTBD 2021:
International Conference on New Methods and Tools for Big Data
July 19 – 20. 2021, Toronto, Canada
see: https://waset.org/new-methods-and-tools-for-big-data-conference-in-july-2021-in-toronto

August 2021

IcABCD 2021:
International Conference on Advances in Big Data, Computing and Data Communication System
August 6 - 7, 2020, Durban, South Africa
see: https://icabcd.org/2021/

Big Data 2021:
Big Data Analysis and Data Mining
August 09 - 10 2021, Zurich, Switzerland
see: https://datamining.expertconferences.org/

ICDSE 2021:
International Conference on Data Science and Engineering
August 12 - 14, 2020, Kerala, India
see: http://icdse.in/

BigDataService 2021:
IEEE Big Data Service 2021
August 23 - 26, 2021, **as Virtual meeting**
see: http://www.big-dataservice.net/

QEST 2021:
International Conference on Quantitative Evaluation of SysTems
August 23 - 27, 2021, Paris, France
see: http://www.qest.org/qest2021/

ESEC/FSE 2021:
European Software Engineering Conference and **Symposium on the Foundation of Software Engineering**
August 23 – 27, 2021, Athens, Greece
see: /https://2021.esec-fse.org/

ICGSE 2021:
International Conference on Global Software Engineering
August 30 - 31, 2021, Moscow, Russia
see: https://conf.researchr.org/home/icgse-2020

September 2021

Euromicro DSD/ SEAA 2021:
Software Engineering & Advanced Application Conference
September 1 – 3, 2021, Palermo, Italy
see: https://dsd-seaa2020.unipv.it

EuroAsiaSPI² 2021:
European Systems & Software Process Improvement and Innovation Conference
September 1 - 3, 2021, Krems, Austria
see: https://2021.eurospi.net/index.php/about/eurospi-2016

SCC 2021:
International Conferences on Services Computing
September 5 - 10, 2021, **as Virtual meeting**,
see: https://conferences.computer.org/scc/2021/

CLOUD 2021:
IEEE International Conference on Cloud Computing
September 5 -10, 2021, **as Virtual meeting**,
see: https://conferences.computer.org/cloud/2021/

SERVICES 2021:
IEEE World Congress on Services
September 5 -10, 2021, **as Virtual meeting**, Chicago, USA
see: https://conferences.computer.org/services/2021/

RE 2021:
IEEE International Requirement Engineering Conference
September 20 - 24, 2021, South Bend, USA
see: http://conf.researchr.org/home/re-2021

SEFM 2021: **International Conference on Software Engineering and Formal Methods**
September 2021, **organization in process**
see: https://event.cwi.nl/sefm2020/

October 2021

ESEM 2021: **Conference on Empirical Software Engineering and Measurement**
October 11 - 15, 2021, Bari, Italy
see: https://conf.researchr.org/home/esem-2021

ESAPI 2021: **API Conference 2021**
October , 2021, Berlin, Germany
see: https://blog.hwr-berlin.de/schmietendorf/

data2day 2021: **Konferenz für Big Data, Data Science und Machine Learning**
October , 2021, Heidelberg, Germany
see: https://www.xpobuzz.com/.data2day

ICSEA 2021: **International Conference on Software Engineering Advances**
October 3 - 7, 2021, Barcelona, Spain
see: https://www.iaria.org/conferences2021/ICSEA21.html

ASQT 2021: **Arbeitskonferenz Softwarequalität, Test und Innovation**
October , 2021, Bozen, Austria
see: http://www.asqt.org/

IWSM/Mensura 2021: **Common International Conference on Software Measurement**
October , **organization in process**
see: https://www.iwsm-mensura.org/

November 2021

ICPCC 2021: **Performance Computing and Communications Conference**
November 15 – 16,2021, Jeddah, Saudi Arabia
see: https://waset.org/perfromance-computing-and-communications-conference-in-november-2021-in-jeddah

PROFES 2021: **International Conference on Product Focused Software Process Improvement**
November , 2021, Turin, Italy
see https://www.profes-conferences.org/

ASE 2021: **Automated Software Engineering**
November 15 - 19, 2021, Melbourne, Australia
see: https://conf.researchr.org/home/ase-2021

SERA 2021: **IEEE/ACIS Conference on Software Engineering Research, Management and Applications**
November 30, 2021, Kanazawa, Japan
see: https://acisinternational.org/conferencences/sera-2021

December 2021

IEEE ICDM 2021:	**IEEE International Conference on Data Mining** December 7 - 10 , 2021, Auckland, New Zealand see: https://icdm2021.auckland.ac.nz/
Big Data 2021:	**IEEE International Conference on Big Data** December 15 - 18, 2021, Orlando, USA see: http://bigdataieee.org/BigData2021/
BCD 2021:	**International Conference on Big Data, Cloud Computing, and Data Science** December 14 - 16, 2021, Macao see: https://2020.aconf.cn/conf-177327.html

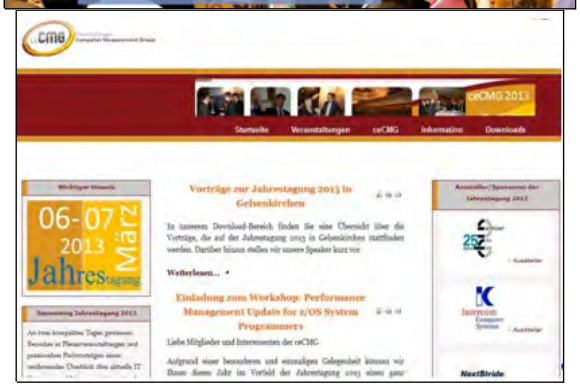
see also:

- http://www.acisinternational.org/newconferences.html
- https://www.acm.org/conferences
- https://www.ieee.org/conferences_events/index.html

COMMUNITIES

Common Software Measurement International Consortium (COSMIC)

http://cosmic-sizing.org



Central Europe Computer Measurement Group (ceCMG)

http://www.cecmg.de

Metrics Association's International Network (MAIN)

http://*www.mai-net.org*

Netherlands Software Metrics users Association (NESMA)

http://www.nesma.org/

GI-Fachgruppe Software-Messung und Bewertung

https://fg-metriken.gi.de/

(Measurement News Online)

Deutschsprachige Anwender-gemeinschaft für Software-Metrik und Aufwandschätzung

http://www.dasma.org

International Software Bench-marking Standard Group (ISBSG)

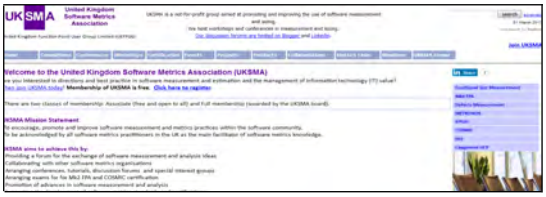
https://www.isbsg.org

Finnish Software Measurement Association (FISMA)

http://www.fisma.fi/in-english/

Asociacion Espanola de Metricas de Software

http://www.aemes.org/



United Kongdom Software Metrics Association (UKSMA)

http://www.uksma.co.uk

Gruppo Utenti Function Point Italia - Italian Software Metrics Association (GUFPI - ISMA)

http://www.gufpi-isma.org

Anwenderkonferenz Software-qualität und Test (ASQT)

http://www.asqt.org

MEASUREMENT SERVICES

Software Measurement Laboratory (SML@b)

http://www.smlab.de

International Function Point Users Group (IFPUG)

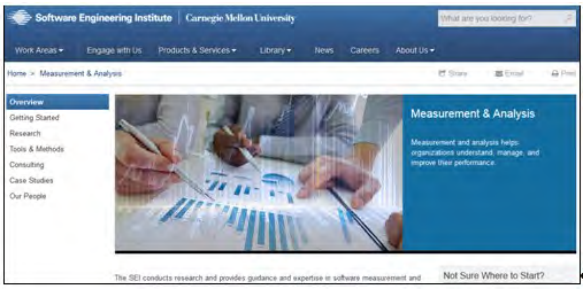
http:www.ifpug.org

Practical Software & Systems Measurement

www.psmsc.com/:

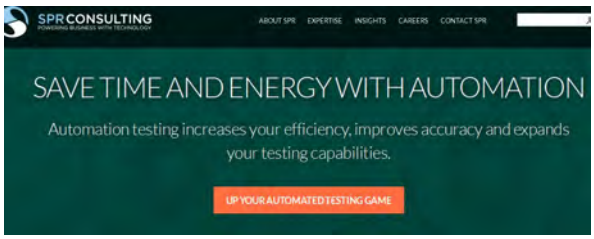
Computer Measurement Group (CMG)

http://www.cmg.org



Software Engineering Institute (SEI)

www.sei.cmu.edu/measurement/



Software Productivity Research (SPR)

http://www.spr.com/

McCabe & Associates

http://www.mccabe.com

	**SQS Gesellschaft für Software-Qualitätssicherung** ***http://www.sqs.de***
	**Quantitative Software Management (QSM)** ***http://www.qsm.com/***
	**Fraunhofer Institute for Experimental Software Engineering (IESE)** **https://www.iese.fraunhofer.de/**
	**National Institute of Standards and Technology (NIST)** ***https://www.nist.gov/el***

SOFTWARE MEASUREMENT INFORMATION

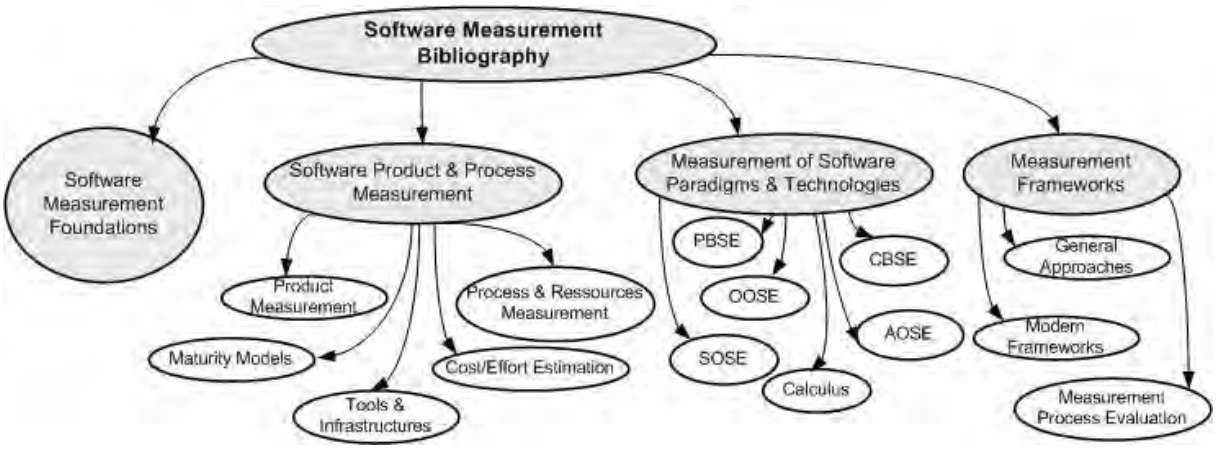
Software Measurement Bibliography

See our overview about software metrics and measurement in the Bibliography at

https://fg-metriken.gi.de/bibliographie/

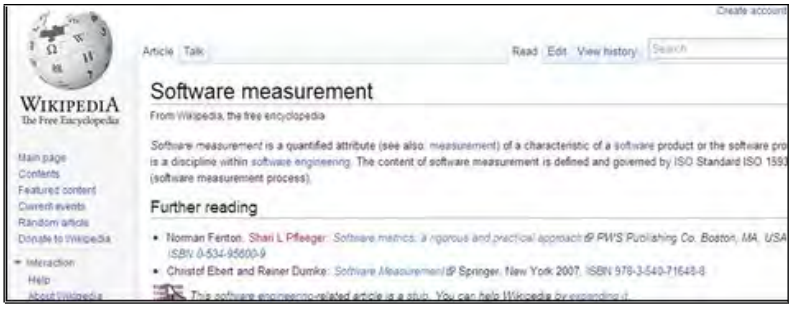
including any hundreds of books and papers

Bibliography Structure:



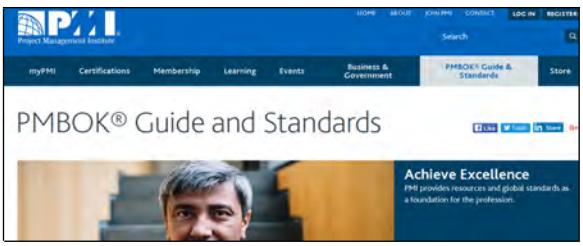
Software Measurement & Wikipedia

Help to qualify the software measurement knowledge and intentions in the world wide web:



Software Engineering Body of Knowledge (SWEBOK)

http://*www.swebok.org*



Project Management Body of Knowledge (PMBOK)

http://www.pmbook.org

SOFTWARE MEASUREMENT NEWS

VOLUME 26 2021 NUMBER 1

CONTENTS

ISSN 1867-9196