# Reputationsmaße und h-Index

Durchgängiges Beispiel für Software-Qualität

Prof. Dr. Kurt Schneider

Kurt.Schneider@Inf.Uni-Hannover.de

FG Software Engineering Leibniz Universität Hannover





## Was lernt man wie am besten?

## Wieso Sie diese Übungen machen sollten

- Software-Qualität entsteht, indem
  - Konstruktion
  - Prüfung
  - kritische Bewertung von Software ineinandergreifen.
- Vorlesung:
  - Prinzip verstehen: Aspekte isoliert besprechen
- Übungen
  - einzelne Aspekte üben, Aufgaben nicht zu groß
- Mehr ist möglich: in Vorlesung, Übung und Eigeninitiative
  - Echtes Beispiel von vorne bis hinten durchgehen
  - Sie entscheiden, wie stark sie sich engagieren



# Die Anwendung: Reputationssysteme

#### • Eine Anwendung, die Sie vermutlich noch nicht kennen

- Ähnliches kennen Sie aber aus eigenem Umfeld (Internet)
- Nicht besonders komplizierte Aufgabe
- Mag Ihnen nicht extrem interessant vorkommen
- Aber sie verlangt sehr verschiedene Techniken
- Fazit: Genau wie in echten Projekten.

#### • Aufgabe:

- Alle Schritte aus Q-Sicht systematisch bearbeiten
- Reale Aufgabe, echte Herausforderung ernst genommen
- Freiwillig, mit hohem Übungswert.



# Der gute Ruf

- Facebook "likes"
  - Beitrag hat gefallen
  - Viele likes: gutes Votum
  - Autor wird geschätzt







## Foren: positive Rückmeldung

- Mehr Aufmerksamkeit
- Nächster Level oder Rang

#### Amazon-Rezension: hilfreich?

– Viele hilfreiche: Top-Rezensent

TOP 500 REZENSENT
HALL OF FAME REZENSENT
REAL NAME





## Der Ruf in der Wissenschaft

- Qualitätsmodelle für wissenschaftliche Arbeit?
  - Qualität von wiss. Beiträgen messen
  - "Beiträge" sind Artikel,sogenannte "Papers" oder "Papiere",
    - Wo Forschungsergebnisse publiziert werden.
  - Metriken für die Beiträge eines Autors
    - ACHTUNG: nicht der Inhalt wird gemessen!
    - Auch nicht direkt die Qualität des Inhalt
    - Sondern: ob andere sie gut fanden
    - "h-Index" (nach Hirsch) zählt Papiere, und wie oft sie zitiert werden

#### Calculating BPEL Test Coverage through Instrumentation

Daniel Lubke, Leif Singer Leibniz Universität Hannover Software Engineering Group Welfengarten 1, D-30167 Hannover, Germany {daniel luebke, leif singer}@inf.uni-hannover.de Alex Salnikow
Leibniz Universität Hannover
Software Engineering Group
Welfengarten 1, D-30167 Hannover, Germany
alex salnikow@se.uni-hannover.de

#### Abstract

Assessing the quality of tests for BPEL processes is a difficult task in projects following SOA principles.

agrees take to project positioning over promoting to Since templetient resting can lead to impresses defects that can be extremely costly to complex and mission critical environments, this problem needs to be addressed. By using formally defined test metrics that can be eval-

By using formally defined ten metrics that can be evaluated anomatically finesting an extension to the BPELUnit tenting framework, teners are able to assess whicher their white box tens cover all important areas of a BPEL process. This leads to better tens and thus to better BPEL pro-

This leads to bester tens and thus to bester BPEL processes because tessers can improve their test cases by knowting which important areas of the BPEL process have not been seered yes.

#### 1 Introduction

With the rise of Service-Oriented Architecture (SOA) and the Wis Forico stack of translate, the IT beganness of muny companies are moving to serve, both and feelble architectures. (Web Services are being developed which can be used in so-called compositions in order to realize ratherantee, the processors. Such processors such parallel architectures processes. Such processors due to the heart of an energetise. Therefore, the technical interherantical becomes furnisherantical.

implementation becomes minimen-rivical. The Ws. BPIL (thent BPIL) standard [4] is a language providing the means to implement service compositions. Because BPIL is standardized by OxIS and backed by many software wenders, it is often chosen by enterprises. Before puring BPIL processes and the corresponding Web Services into use, these systems must be served intensively in order to militage the risk imposed by delects due to their important role in the enterprise.

However, Web Services and BPIL both are relatively

However, Web Services and BPEL both are relatively new technologies, and many methods that are available for general software development have either not been ported to SOA yet or SOA-related experience is still missing. Within this paper, we describe twic coverage metrics. BMEL processes and an instrume ration strategy as a way of measuring them. Text coverage in a class of metrics do termining how much of the software has actually been executed by a set of exit cases. It is often used as an indicator of the quality of the sets. Area so which are not being executed by any set at all are more likely to contain errors than those which have been wated.

In the next section of this paper, we will outline retund work from the field of the coverage neasurement and mexica before we will introduce text coverage mercica for BPEL. Afterwards, we introduce too BPEL processes can be instrumented in order to measure the text coverage. The case analy, described in the subsequent section, deduers some experiences we use to highlight implications and achievements of our appears. If Finally, we assumate the implications and achievements of our approach and provide conclusions as well as a possible control.

#### 2 Related Wor

Metrics for measuring software and test quality are common in the field of Software Engineering. Since BPEL is "programming in the large", it makes so noe to adopt these metrics in the new field of SOA as well.

An example of a successfully adopted metric is complexity: McCabe defined Complexity [19] for software programs which has been modified by Cirdoso [8] for usage in BFEL.

For measuring test quality, many test coverage metrics based on different coverage subjects have been proposed and are used in many development projects:

Code Coverage Metrics (like Statement Coverage [6, p. 44]. Branch Coverage [6, p. 45], Conditional Coverage [6, pp. 44] measure how much of the actual program is executed. When achieving 100%, each streement, branch or path is executed at least once. This

Kurt Schneider



## Der Ruf in der Wissenschaft

#### References

- 1. International Standardization Organization. ISO 15408:2007 Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 2, CCMB-2007-09-001, CCMB-2007-09-002 and CCMB-2007-09-003 (September 2007)
- 2. Houmb, S.H., Islam, S., Knauss, E., Jürens, J., Schneider, K.: Eliciting Security Requirements and Tracing them to Design: An Integration of Common Criteria, Heuristics, and UMLsec. Requirements Engineering Journal 15(1), 63–93 (2010)
- 3. Knauss, E., Lübke, D., Meyer, S.: Feedback-Driven Requirements Engineering: The Heuristic Requirements Assistant. In: International Conference on Software Engineering (ICSE 2009), Formal Research Demonstrations Track, Vancouver, Canada, pp. 587–590 (2009)
- 4. Jürjens, J.: Secure Systems Development with UML. Springer, Heidelberg (2005)
- Stellen Sie sich gerne etwas Ähnliches vor für
  - Facebook
  - Amazon
  - Youtube usw.





# Aufgabe 1: Anforderungen klären

Der h-Index ist definiert als:

h-index h eines Autoren A ist :=<sub>DEF</sub>
die höchste (ganze) Zahl z, so dass
A mindestens z Papiere veröffentlicht hat,
von denen jedes mind. z mal zitiert wurde.

- Schreiben Sie diese Definition als Formel eindeutig auf
- Schreiben Sie Software-Anforderungen für Java-Entwickler:
  - Gehen Sie davon aus, dass Sie eine
     (unsortierte) LinkedList von PaperCitations bekommen.
  - Ein Objekt des Typs PaperCitations besteht aus Autorenname, Titel und Anzahl der Zitate für dieses Papier.
  - Sie können sich von jedem PaperCitations die Aspekte: author, title und citations geben lassen.

