Introduktion till vetenskaplig metodik

Johan Åberg

Innehåll

- Forskarvärlden
- Viktiga begrepp
- Referenshantering
- Den vetenskapliga rapporten
- Vetenskaplig diskussion

Forskarvärlden

- Forskare mäts i antal publikationer och antal citeringar
- Publikationer
  - Tidskrifter
  - Konferenser
  - Workshops
- Referee-granskning
- Olika publikationer har olika status

Call for papers (cfp)

ICSE 2014 Call for Papers – Research Papers

Goal and scope

ICSE is the premier forum for researchers to present and discuss the most recent innovations, trends, outcomes, experiences, and challenges in the field of software engineering.

We invite high quality submissions of research papers describing original and unpublished results, pertaining to all aspects of software engineering and particularly topics relevant to today's emerging practices and realities. We encourage all types of work, and especially encourage papers that assess the state of the art in the field, its research trajectory, and core assumptions that may or may not hold in the future.

ICSE is a selective conference, but welcomes innovative ideas that are well presented, timely, and have high likely impact, even if the findings are preliminary or not yet (fully) evaluated. Naturally, all submissions must position themselves within the existing literature, describe the relevance of the results to certain software engineering goals, and include a clear motivation and presentation of the work.

New this year

To guide the authors in preparing their submissions and to establish a consistent set of expectations in the review process, all authors are asked, as part of the online submission process, to self-identify their papers with one or more of the following categories:

- *Analytical*: A paper in which the main contribution relies on new algorithms or mathematical theory. Examples include new bug prediction techniques, model transformations, algorithms for dynamic and static analysis, and reliability analysis. Such a contribution must be evaluated with a convincing analysis of the algorithmic details, whether through a proof, complexity analysis, or run-time analysis, among others and depending on the objectives.
- *Empirical*: A paper in which the main contribution is the empirical study of a software engineering technology or phenomenon. This includes controlled experiments, case studies, and surveys of professionals reporting qualitative or quantitative data and analysis results. Such a contribution will be judged on its study design, appropriateness and correctness of its analysis, and threats to validity. Replications are welcome.
- *Technological*: A paper in which the main contribution is of a technical nature. This includes novel tools, modeling languages, infrastructures, and other technologies. Such a contribution does not necessarily need to be evaluated with humans. However, clear arguments, backed up by evidence as appropriate, must show how and why the technology is beneficial, whether it is in automating or supporting some user task, refining our modeling capabilities, improving some key system property, etc.
- *Methodological*: A paper in which the main contribution is a coherent system of broad principles and practices to interpret or solve a problem. This includes novel requirements elicitation methods, process models, design methods, development approaches, programming paradigms, and other methodologies. The authors should provide convincing arguments, with commensurate experiences, why a new method is needed and what the benefits of the proposed method are.
- *Perspectives*: A paper in which the main contribution is a novel perspective on the field as a whole, or part thereof. This includes assessments of the current state of the art and achievements, systematic literature reviews, framing of an important problem, forward-looking thought pieces, connections to other disciplines, and historical perspectives. Such a contribution must, in a highly convincing manner, clearly articulate the vision, novelty, and potential impact.

All papers are full papers, and papers may belong to more than one category. Note that papers from any research area can fall into any of these categories, as the categories are constructed surrounding methodological approaches, not research topics (e.g., one could write an analytical paper on a new analysis technique, an empirical paper that compares a broad range of such techniques, a technological paper that makes an analysis technique practically feasible and available, or a perspectives paper that reviews the state of the art and lays out a roadmap of analysis techniques for the future).

Evaluation

Submissions that are not in compliance with the required submission format or that are out of the scope of the conference will be rejected without reviewing.

Submitted papers must comply with ACM plagiarism policy and procedures. Papers submitted to ICSE 2014 must not have been published elsewhere and must not be under review or submitted for review elsewhere while under consideration for ICSE 2014.

All submissions that meet the criteria and fit the scope of the conference will be reviewed by at least two members of the Program Committee. Submissions will be evaluated on the basis of originality, evaluation, soundness, importance of contribution, quality of presentation, and appropriate comparison to related work.

ICSE this year will adopt a program board model in order to better process the increasing number of submissions that it has been receiving each year. The Program Board will work with the Program Committee to make the final decisions about which submissions are accepted for presentation at the conference. Detailed instructions have been provided to the Program Board and Program Committee: program board instructions and program committee instructions.

If you and your co-authors have not previously published a research paper at an ICSE conference, you may want to consider the ICSE 2014 mentoring program.

How to submit

All submitted papers must conform to the ICSE 2014 formatting and submission instructions, and must not exceed 10 pages for the main text, inclusive of figures, tables, appendices, etc. References may be included on up to two additional pages. All submissions must be in PDF.

Papers must be submitted through the CyberChair online submission system. Submissions that adhere to the submission and formatting instructions can be made using the CyberChair link provided here: ~~http://cyberchairpro.borbala.net/icsepapers/submit/~~. The deadline for this year's submission has passed.

Papers must be submitted electronically by the stated deadline. The deadline is firm and not negotiable.

Journal citation index

ISI JOURNALS - SOFTWARE ENGINEERING

Journal	12	11	10
1. IEEE Transactions on Software Engineering (TSE)	**2.59**	1.98	2.22
2. Communications of the ACM (CACM)	**2.51**	1.92	2.35
3. IEEE Computer (Comp)	**1.68**	1.47	1.79
4. IEEE Software (SW)	**1.62**	1.51	1.51
5. ACM Transactions on Software Engineering and Methodology (TOSEM)	**1.55**	1.27	1.69
6. Information and Software Technology (IST)	**1.52**	1.25	1.51
7. Automated Software Engineering (ASE)	**1.40**	0.86	0.81
8. SW Maintenance & Evolution - Research & Practice (JSEP, formerly SMERP)	**1.27**	0.84	0.61
9. Software and Systems Modeling (SoSyM)	**1.25**	1.06	1.27
10. Empirical Software Engineering (ESEJ)	**1.18**	1.85	1.78
11. Journal of Systems and Software (JSS)	**1.14**	0.84	1.28
12. Requirements Engineering Journal (REJ)	**1.05**	0.97	0.86
13. Software Testing Verification & Reliability (STVR)	**1.04**	0.96	0.76
14. Software Practice & Experience (SPE)	**1.01**	0.52	0.57
15. Software Quality Journal (SQJ)	**0.85**	0.42	0.75
16. IBM Journal of Research & Development (IBM JRD)	**0.69**	0.72	1.79
17. IET Software (IET SW, was 'IEE Proceedings - Software' pre 2007)	**0.66**	0.33	0.67
18. Software Engineering and Knowledge Engineering (IJSEKE)	**0.30**	0.13	0.25

Computer Science Conference Rank

Source: CORE

Rank A+ | Rank A | Rank B | Rank C

Rank A+ CS conference

Antonym	Name	Rank
AAAI	National Conference of the American Association for Artificial Intelligence	A*
AAMAS	International Conference on Autonomous Agents and Multiagent Systems	A*
ACL	Association of Computational Linguistics	A*
ACMMM	ACM Multimedia	A*
ASPLOS	Architectural Support for Programming Languages and Operating Systems	A*
CAV	Computer Aided Verification	A*
CCS	ACM Conference on Computer and Communications Security	A*
CHI	International Conference on Human Factors in Computing Systems	A*
COLT	Annual Conference on Computational Learning Theory	A*
CRYPTO	Advances in Cryptology	A*
CSCL	Computer Supported Collaborative Learning	A*
DCC	Data Compression Conference	A*
DSN	IEEE/IFIP International Conference on Dependable Systems	A*
EuroCrypt	International Conference on the Theory and Application of Cryptographic Techniques	A*
FOCS	IEEE Symposium on Foundations of Computer Science	A*
FOGA	Foundations of Genetic Algorithms	A*
HPCA	International Symposium on High Performance Computer Architecture	A*
I3DG	ACM SIGRAPH Interactive 3D Graphics	A*
ICCV	IEEE International Conference on Computer Vision	A*
ICDE	International Conference on Data Engineering	A*
ICDM	IEEE International Conference on Data Mining	A*
ICFP	International Conference on Functional Programming	A*
ICIS	International Conference on Information Systems	A*
ICML	International Conference on Machine Learning	A*
ICSE	International Conference on Software Engineering	A*
IJCAI	International Joint Conference on Artificial Intelligence	A*
IJCAR	International Joint Conference on Automated Reasoning	A*
ISCA	ACM International Symposium on Computer Architecture	A*
ISMAR	IEEE/ACM International Symposium on Mixed and Augmented Reality	A*
ISSAC	International Symposium on Symbolic and Algebraic Computation	A*
ISWC	IEEE International Symposium on Wearable Computing	A*

Litteratursökning

- Databaser
  - IEEE
    - IEEE explore
  - ACM
    - ACM DL
  - Springer
    - Springer Link
- bibl.liu.se
- Google scholar
  - Vilka andra artiklar har citerat den här artikeln?
  - Funktionen ”Cite”

Söka

- Böcker
- E-bokssamlingar
- E-uppslagsverk och e-ordböcker
- Tidskrifter och bokserier
- Databaser
- Examensarbeten
- Dagstidningar

- Allt om inloggning
- Tillgång till e-resurser
- Villkor för användande/Upphovsrätt

Tidskrifter och bokserier

TIDSKRIFTER

software engineering | Sök ▶

(•) Alla tidskrifter ( ) Endast elektroniska tidskrifter

[ ] Sök på första orden i titeln

Sökning på **software engineering** gav 30 träffar.
T = tryckt, **E** = elektronisk

ACM SIGSOFT Software Engineering Notes
1976- (Endast LiU) **E**

ACM transactions on software engineering and methodology
Vol. 1(1992)- (Endast LiU) **E**

Advances in Engineering Software
Vol. 22(1995-) (Endast LiU) **E**

Advances in engineering software / publ. in assoc. with the International Society for Computational Methods in Engineering
Vol. 1-12 (1979-1990) Bevaras enligt TGV-plan/Li TekNat-biblioteket: Tidskr p (i Bokladan)
(i Bokladan) **T**

Advances in engineering software and workstations.
Vol. 13 (1991) Bevaras enligt TGV-plan/Li TekNat-biblioteket: Tidskr p (i Bokladan)
(i Bokladan) **T**

Advances in engineering software.
KB: Vol. 14-30 (1992-1999) Bevaras enligt TGV-plan/Li TekNat-biblioteket: Tidskr p (i Bokladan)
(i Bokladan) **T**

Advances in Software Engineering
2009- (Fri resurs) **E**

Automated Software Engineering
(1997-) (Endast LiU) **E**

HITTAR INTE DET DU SÖKER?

Om du inte hittar den tidskrift du söker kan artiklar även beställas via fjärrlån.

Gör en artikelbeställning

Fråga oss

VANLIGA FRÅGOR OM TIDSKRIFTER

Hur hittar jag tidskrifters impact factor?

Är tidskriften vetenskaplig? Information i Ulrichsweb

Vilken tidskrift ska jag publicera mig i?

Advances in Engineering Software

Supports Open Access | About this Journal | Sample Issue Online | Submit your Article

Publication History: Formerly known as Advances in Engineering Software and Workstations; Incorporating Computing Systems in Engineering;

New Article Feed

Alert me about new articles

Add to Favorites

< Previous vol/iss | Next vol/iss >

Volume 73, In Progress (July 2014)

articles **1** - **5**

Articles in Press

Open Access articles

- Volume 73 (2014)
  - Volume 73
    In Progress (July 2014)
- Volume 72 (2014)
- Volume 71 (2014)
- Volume 70 (2014)
- Volume 69 (2014)
- Volume 68 (2014)
- Volume 67 (2014)
- Volume 66 (2013)
- Volume 65 (2013)
- Volume 64 (2013)
- Volume 62 (2013)
- Volume 60 (2013)
- Volume 59 (2013)
- Volume 58 (2013)
- Volume 57 (2013)
- Volume 56 (2013)
- Volume 55 (2013)
- Volume 54 (2012)
- Volume 53 (2012)
- Volume 52 (2012)
- Volume 51 (2012)

E-mail articles | Export | PDF downloader | Open all previews

All access types

This issue is In Progress but contains articles that are final and fully citable. For recently accepted articles, see Articles in Press.

Research Articles

Söka och använda
Böcker
E-bokssamlingar
E-uppslagsverk och e-ordböcker
Tidskrifter och bokserier
Databaser
Doktorsavhandlingar
Examensarbeten
Dagstidningar
Allt om inloggning
Tillgång till e-resurser
Villkor för användande/Upphovsrätt

Databaser

Visa alla ämnen | A - Ö

ACM digital library

Tidskrifter och konferenser inom området IT och programmering i fulltext från Association for Computing Machinery (ACM). 1985-

Varianttitel
ACM portal

Sök i **ACM digital library**
Extern tillgång endast anställda och studenter vid LiU

FLER DATABASER INOM SAMMA ÄMNE

Datavetenskap | **Allmänt: Ekonomi** |

Google software quality

Scholar About 1,110,000 results (0.05 sec)

Articles
Case law
My library

Any time
Since 2014
Since 2013
Since 2010
Custom range...

Sort by relevance
Sort by date

include patents
include citations

Create alert

[BOOK] **Software quality**: theory and management
A Gillies - 2011 - books.google.com
The need for **quality** in **software** should be self-evident. Our dependence upon computers in all spheres of life is continually interesting. It is therefore essential that computers operate reliably and effectively. There are those who still scoff at the need for **quality** and dismiss it **...**
Cited by 231 Related articles All 6 versions Cite Save

Don't touch my code!: examining the effects of ownership on **software quality**
C Bird, N Nagappan, B Murphy, H Gall... - ... Foundations of **software** ..., 2011 - dl.acm.org
Abstract Ownership is a key aspect of large-scale **software** development. We examine the relationship between different ownership measures and **software** failures in two large **software** projects: Windows Vista and Windows 7. We find that in all cases, measures of **...**
Cited by 40 Related articles All 14 versions Cite Save

Standardized code **quality** benchmarking for improving **software** maintainability
R Baggen, JP Correia, K Schill, J Visser - **Software Quality** Journal, 2012 - Springer
Abstract We provide an overview of the approach developed by the **Software** Improvement Group for code analysis and **quality** consulting focused on **software** maintainability. The approach uses a standardized measurement model based on the ISO/IEC 9126 definition **...**
Cited by 39 Related articles All 12 versions Cite Save

[CITATION] **Software Quality** Prediction Method with Hybrid Applying Principal Components Analysis and Wavelet Neural Network and Genetic Algorithm
C Zhong, Q Hu, F Yang, M Yin - JDCTA: International Journal of Digital Content ..., 2011
Cited by 10 Related articles Cite Save

[BOOK] **Software** modeling and design: UML, use cases, patterns, and **software** architectures
H Gomaa - 2011 - dl.acm.org
... The author describes architectural patterns for various architectures, such as broker, discovery, and transaction patterns for service-oriented architectures, and addresses **software quality** attributes including maintainability, modifiability, testability, traceability, scalability **...**
Cited by 29 Related articles All 2 versions Cite Save

Scholar

About 40 results (0.04 sec)

All citations

Articles

Case law

My library

Any time

Since 2014

Since 2013

Since 2010

Custom range...

Sort by relevance

Sort by date

include citations

Create alert

Don't touch my code!: examining the effects of ownership on software quality

Search within citing articles

Explaining software defects using topic models

TH Chen, SW Thomas, M Nagappan... - ... (MSR), 2012 9th ..., 2012 - ieeexplore.ieee.org

Abstract—Researchers have proposed various metrics based on measurable aspects of the source code entities (eg, methods, classes, files, or modules) and the social structure of a software project in an effort to explain the relationships between software development ...

Cited by 13 Related articles All 5 versions Cite Save

Dual ecological measures of focus in software development

D Posnett, R D'Souza, P Devanbu... - ... (ICSE), 2013 35th ..., 2013 - ieeexplore.ieee.org

Abstract—Work practices vary among software developers. Some are highly focused on a few artifacts; others make wideranging contributions. Similarly, some artifacts are mostly authored, or "owned", by one or few developers; others have very wide ownership. Focus ...

Cited by 13 Related articles All 8 versions Cite Save

Developing an h-index for OSS developers

A Capiluppi, A Serebrenik... - ... Repositories (MSR), 2012 ..., 2012 - ieeexplore.ieee.org

(OSS) repositories has been used for many practical reasons: detecting community structures; identifying key roles among developers; understanding software quality; predicting the arousal of bugs in large OSS systems, and so on; but also to formulate and ...

Cited by 9 Related articles All 4 versions Cite Save

Who? where? what? examining distributed development in two large open source projects

C Bird, N Nagappan - ...), 2012 9th IEEE Working Conference on, 2012 - ieeexplore.ieee.org

Abstract—To date, a large body of knowledge has been built up around understanding open source software development. However, there is limited research on examining levels of geographic and organizational distribution within open source software projects, despite ...

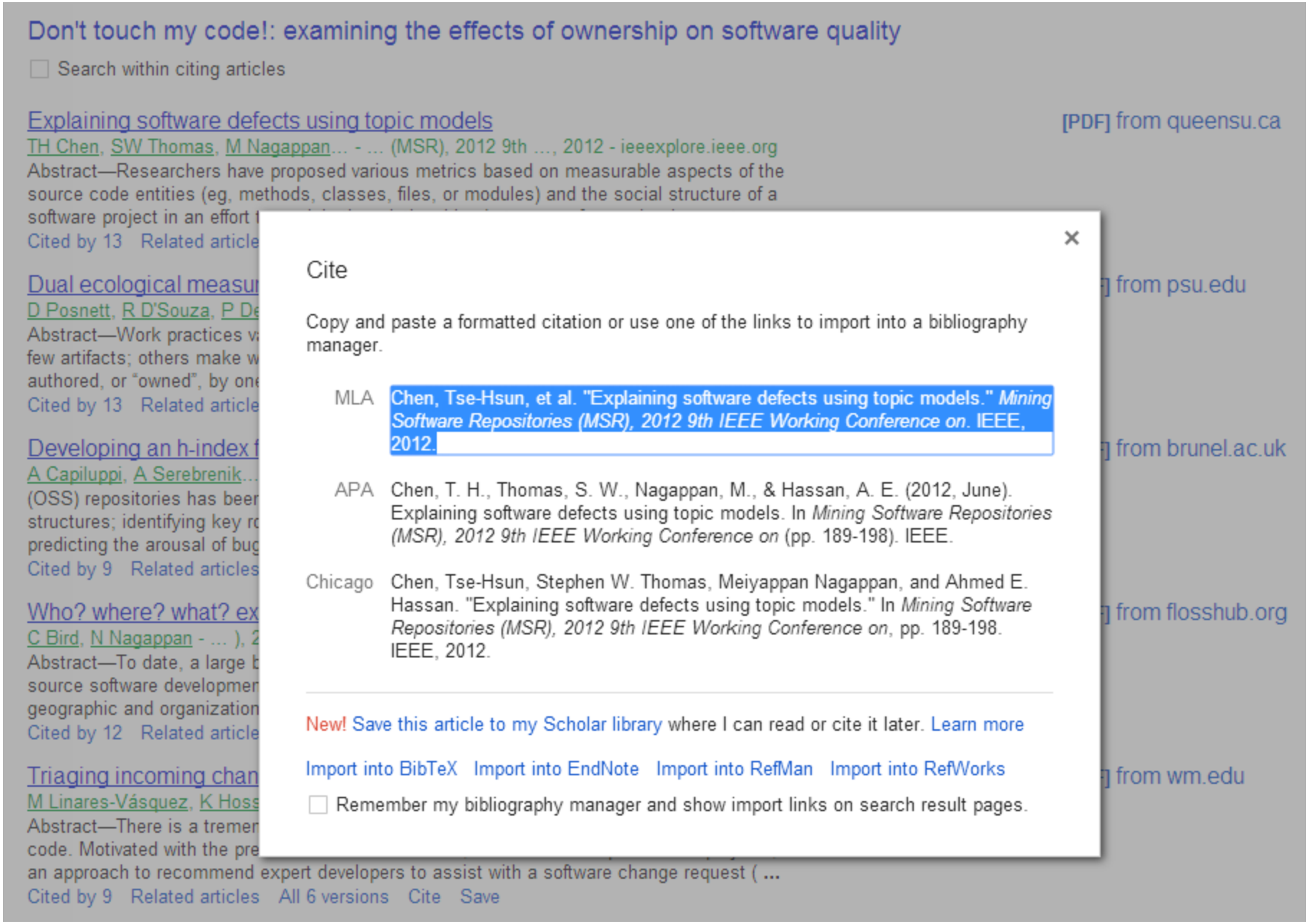
Cited by 12 Related articles All 6 versions Cite Save

Triaging incoming change requests: Bug or commit history, or code authorship?

M Linares-Vásquez, K Hossen, H Dang... - ... (ICSM), 2012 28th ..., 2012 - ieeexplore.ieee.org

Abstract—There is a tremendous wealth of code authorship information available in source code. Motivated with the presence of this information, in a number of open source projects, an approach to recommend expert developers to assist with a software change request (...

Cited by 9 Related articles All 6 versions Cite Save


Don't touch my code!: examining the effects of ownership on software quality
Search within citing articles
Explaining software defects using topic models
[PDF] from queensu.ca
TH Chen, SW Thomas, M Nagappan... - ... (MSR), 2012 9th ..., 2012 - ieeexplore.ieee.org
Abstract—Researchers have proposed various metrics based on measurable aspects of the source code entities (eg, methods, classes, files, or modules) and the social structure of a software project in an effort
Cited by 13 Related article
Dual ecological measur
from psu.edu
D Posnett, R D'Souza, P De
Abstract—Work practices v
few artifacts; others make w
authored, or "owned", by on
Cited by 13 Related article
Developing an h-index f
from brunel.ac.uk
A Capiluppi, A Serebrenik...
(OSS) repositories has beer
structures; identifying key r
predicting the arousal of bug
Cited by 9 Related articles
Who? where? what? ex
from flosshub.org
C Bird, N Nagappan - ...), 2
Abstract—To date, a large b
source software developmer
geographic and organization
Cited by 12 Related article
Triaging incoming chan
from wm.edu
M Linares-Vásquez, K Hoss
Abstract—There is a tremer
code. Motivated with the pre
an approach to recommend expert developers to assist with a software change request (...
Cited by 9 Related articles All 6 versions Cite Save
Cite
Copy and paste a formatted citation or use one of the links to import into a bibliography manager.
MLA Chen, Tse-Hsun, et al. "Explaining software defects using topic models." Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on. IEEE, 2012.
APA Chen, T. H., Thomas, S. W., Nagappan, M., & Hassan, A. E. (2012, June). Explaining software defects using topic models. In Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on (pp. 189-198). IEEE.
Chicago Chen, Tse-Hsun, Stephen W. Thomas, Meiyappan Nagappan, and Ahmed E. Hassan. "Explaining software defects using topic models." In Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on, pp. 189-198. IEEE, 2012.
New! Save this article to my Scholar library where I can read or cite it later. Learn more
Import into BibTeX Import into EndNote Import into RefMan Import into RefWorks
Remember my bibliography manager and show import links on search result pages.

Vetenskapliga kvalitetsbegrepp

- Validitet
  - Mäter man det man vill mäta?
- Reliabilitet
  - Tillförlitlighet
    - skulle man få samma resultat om man mätte igen, oberoende av vem som skötte mätningen?
- Replikerbarhet
  - Är en studie så väl beskriven och någon kan göra om samma studie?

Metodperspektiv

- Kvalitativ
  - Kvalitativ data
    - Tal, text
  - Fokus
    - Förstå, se olika perspektiv
  - Datainsamling
    - Intervju
    - Fritextsvar i enkät
  - Analys
    - Kategoriseringsmetoder, …
- Kvantitativ
  - Kvantitativ data
    - Siffror
  - Fokus
    - Kvantifiera, jämföra
  - Datainsamling
    - Mätning med instrument
    - Flervalsalternativ
  - Analys
    - Statistiska metoder

Referenshantering

- Tidskrift
  - Nosek, J. T. (1998). The case for collaborative programming. *Communications of the ACM*, *41*(3), pp. 105-108.
- Konferens
  - Plonka, L., Sharp, H., & van der Linden, J. (2012). Disengagement in pair programming: does it matter? In *Proceedings of the 34th International Conference on Software Engineering (ICSE),* pp. 496-506. IEEE.
- Workshop
  - Samma som för konferens
- Avhandling/uppsats
  - Furulund, M.K. (2007). *Empirical Research on Software Effort Estimation Accuracy*, Master Thesis, Department of Informatics, University of Oslo.
- Bok
  - Ford, N. (2008). *The productive programmer*. O'Reilly Media, Inc.

Referenshantering – olika format

[1] E. Arisholm, H. Gallis, T. Dyba, and D. I.K. Sjoberg, "Evaluating pair programming with respect to system complexity and programmer expertise," *IEEE Trans. Softw. Eng.*, vol. 33, pp. 65–86, 2007.

[2] A. Bandura, *Social Foundations of Thought and Action*. Prentice, 1986.

[3] A. Begel and N. Nagappan, "Pair programming: What's in it for me?" in *Proc. 2nd ACM-IEEE Int'l Symp. Empirical Software Engineering and Measurement* (ESEM '08). ACM, 2008, pp. 120–128.

[4] A. Belshee, "Promiscuous pairing and beginner's mind: Embrace inexperience," in *Proc. Agile Development Conference* (ADC '05). IEEE Computer Society, 2005, pp. 125–131.

[5] S. Bryant, P. Romero, and B. du Boulay, "Pair programming and the mysterious role of the navigator," *Int. J. Hum.-Comput. Stud.*, vol. 66, no. 7, pp. 519–529, 2008.

[6] J. Chong, "Social behaviors on XP and non-XP teams: A comparative study," in *Proc. Agile Development Conf.* (ADC '05). IEEE, 2005, pp. 39–48.

[7] J. Chong and T. Hurlbutt, "The social dynamics of pair programming," in *Proc. 29th Int'l Conf. Software Engineering* (ICSE '07). IEEE Computer Society, 2007, pp. 354–363.

Boehm, B., 1984. Software engineering economics. IEEE Transactions on Software Engineering 10, 4–21.

Briand, L.C., Daly, J.W., Wust, J.K., 1999. A unified framework for coupling measurement in object-oriented systems. IEEE Transactions on Software Engineering 25, 91–121.

Brown, R., 2000. Group Processes, second ed. Blackwell Publishers.

Buehler, R., Messervey, D., Griffin, D., 2005. Collaborative planning and prediction: does group discussion affect optimistic biases in time estimation? Organizational Behaviour and Human Decision Processes 97, 47–63.

Cohen, J., 1969. Statistical Power Analysis for the Behavioral Sciences. Academic Press, Inc., New York.

Cohn, M., 2005. Agile Estimating and Planning. Addison-Wesley.

Fairley, D., 2002. Making accurate estimates. IEEE Software 19, 61–63.

Fenton, N.E., 1995. Software Metrics. Thompson Computer Press, London.

Fischer, G.W., 1981. When oracles fail—a comparison of four procedures for aggregating subjective probability forecasts. Organizational Behaviour and Human Performance 28 (August), 96–110.

Fluri, B., Gall, H., 2006. Classifying change types for qualifying change couplings. In: Proceedings of 14th IEEE International Conference on Program Comprehension, Athens, Greece, June, pp. 14–16.

Furulund, M.K., 2007. Empirical Research on Software Effort Estimation Accuracy, Master Thesis, Department of Informatics, University of Oslo.

Graves, T.L., Mockus, A., 1998. Inferring change effort from configuration management databases. In: Proceedings of the 5th International Symposium on Software Metrics, pp. 267–272.

Grenning, J., 2002. Planning Poker or How to avoid analysis paralysis while release planning.

Grimstad, S., Jørgensen, M., 2006. A framework for the analysis of software cost. In: ISESE 2006, Rio de Janeiro, Brazil, pp. 58–65.

Hanson, R., 1999. Decision markets. IEEE Intelligent Systems 14, 16.

Haugen, N.C., 2006. An empirical study of using planning poker for user story estimation. In: Agile 2006 Conference (Agile'06).

Den vetenskapliga rapporten

- Abstract
  - Kort och koncist, inklusive resultat och slutsatser
- Inledning
  - Väck intresse och motivera
  - Tydliga frågeställningar
- Teori
  - Beskriv relaterad forskning
  - Tematiskt, inte per författare
- Metod
  - Detaljerad beskrivning av studien
    - tänk på replikerbarhet
  - Datainsamling och analys
- Resultat
  - Presentera resultatet
  - Sakligt, utan analys eller kommentar
- Diskussion
  - Hur kan resultaten tolkas?
  - Koppling till relaterad forskning (teorikapitlet)?
  - Metodmässiga risker eller begränsningar?
- Slutsatser
  - Återkoppla till forskningsfrågorna
  - Vad blev svaret?
- Referenser
  - Lista källorna enligt passande format

Vetenskaplig diskussion

- Liknande resultat
  - “The strong tendency of navigators to suggest specifications (i.e., what to click or scroll) to the driver is a testament to how closely partners worked together. Chong et al. [7] also observed pairs (professionals) working very closely together—so close that the partners were practically finishing each other’s sentences. Similar to the Chong pairs, our navigators were so engaged in the task and in tune with the context that they made most of their suggestions at the level of what to click next, rather than higher level strategies.”
- Skilda resultat
  - “Our navigators’ strong tendency to offer ideas for specific actions contrasts with prior findings about the level of abstraction of navigator discourse. In particular, Bryant et al. [5] studied the utterances of professional pairs and coded them based on five levels of abstraction (from lowest to highest). Their study found that navigator discourse was predominantly at a moderate level of abstraction, in which the program was discussed in terms of logical chunks and strategies. However, our navigators’ specific-action suggestions were at a lower level of abstraction than logical chunks and strategies.”

Exempel från artikeln: Jones, D.L., and Fleming, S.D. What use is a backseat driver? A qualitative investigation of pair programming. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing*, pp. 103-110, 15-19 September, 2013.

- Varför skilda resultat?
  - “This difference may be because we looked only at utterances in which navigators offered ideas, but it may also be because of differences between the Bryant pairs and ours. For example, our pairs may have worked more closely together than the Bryant professional programmers. A study of professional pairs by Plonka et al. [23] found that their navigators often had reason to disengage from the driver’s activity, for example, because of interruptions or because they divided up work to be done in parallel with the driver. Our navigators generally did not exhibit such disengagement behavior.”

- Alternativ förklaring
  - “The difference may also be because the Bryant pairs were professionals who had been pair programming for over 6 months. Thus, their pairs were likely already jelled, and as such, had developed their pair communication such that they could converse using higher levels of abstraction. In contrast, our pairs may not have developed the common vernacular necessary for easy communication at higher levels of abstraction.”

Exempel från artikeln: Jones, D.L., and Fleming, S.D. What use is a backseat driver? A qualitative investigation of pair programming. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing*, pp. 103-110, 15-19 September, 2013.