RSE mögliche Vorgehen zur Kategorisierung von Software

ein Beitrag zur Diskussion

Prof. Bernhard Rumpe
Software Engineering
RWTH Aachen

http://www.se-rwth.de/

SE Software Engineering | RWTH AACHEN UNIVERSITY

Informatics: Facts and Figures

	RWTH	Informatics
professors	547	39
staff (FTE)	9.496	292
students	45.377	4.577
students in the first core semester	10.866	1.017

State: 09/2018 resp. 06/21 + 10/23

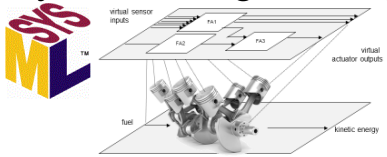
CHE – Ranking: in allen Kategorien in der Spitzengruppe

WiWo – Ranking: Seit Jahren zu den 3 Top-Unis

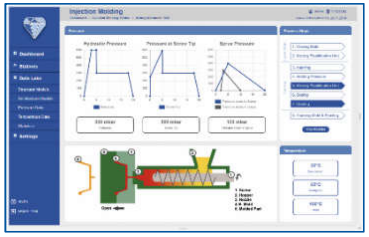
Guide2Research: 54.Platz von 650 der stärksten Unis weltweit

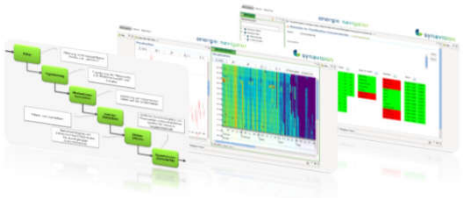
Software Engineering: Our Mission is to improve Software and Systems Development

Model Based Systems Engineering



Digital Twin Cockpits





Energy Efficiency e.g. in Buildings

Contracts, Regulations, Laws, Requirements

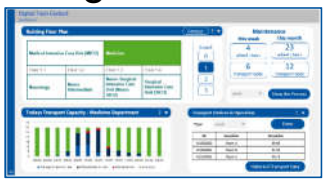
Languages, methods, concepts, tools and infrastructures for

- better and faster agile development,
- resulting in high quality products.

Research Software

BETTER SOFTWARE BETTER RESEARCH

Information Systems: Management Cockpits

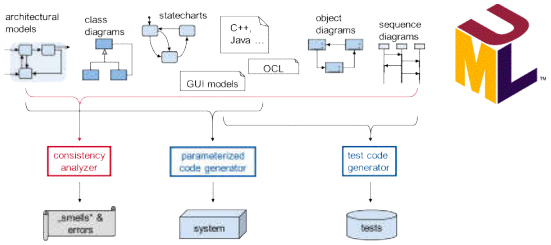


Systems Verification

Software Languages, DSLs, LowCode

Monti Core

Architecture, Agility, Deployment, Design, Tests, Management



SE Software Engineering | RWTH AACHEN UNIVERSITY

Why do we define the categorization?

- Possible use cases:
- As part for a forthcoming set of guidelines for research software development
- To give stakeholders (especially developers and their group/chair/department/institute leaders, but also legal) a better understanding, what kind of software they develop and what measures to be taken
- To produce recognition for scientific developers
- For better assessment of foreign software when deciding to use/import call it
- To negotiate/apply for project based or permanent funding in university/research unit/funding agency
  - Software needs maintenance to stay functional, categorization can highlight importance of software for research and other tasks at university / institute
- In RSE Research, to provide a framework for classifying research objects (e.g., in software corpus analyses)
- more ?

Mögliche Kriterien für eine Katalogisierung

- Je nach Sichtweise lassen sich unterschiedliche Kriterien für eine Kategorisierunganlegen.
- Wir halten eine Auflistung der Kriterien für sinnvoll, entlang derer Kategorisiert werden kann
- Denn: Was wollen wir eigentlich mit den Kategorien machen?
  - Z.B.
    - SE-Methodik festlegen?
    - Test-Überdeckung
    - Zu nutzende Tools (git, etc.)

Kriterium 1: Art der Software

- Forschungssoftware,
- Kommerziell einsetzbare Software,
- Unterstützungs-Software
  (Compiler, Editor, Operating System, etc.)

- Forschungssoftware zeichnet sich aus durch:
  - von Forschenden entwickelt und genutzt wird,
  - zumindest primär keiner kommerziellen Verwertbarkeit zugänglich ist.
- Stattdessen wird Forschungssoftware entweder hausintern, in einer Community oder als Open Source genutzt.
- Nicht alle Software-Pakete sind eindeutig zuzuordnen, viele aber schon.
- Hinweis für die Hochschulleitung: Kommerzialisierbar im Sinne eines verkaufbaren Produktes mit Gewinnerzielungsabsicht ist sehr wenig Forschungssoftware.

Kriterium 2: Nutzerkreis

- Instituts-intern,
- Universitäts*-intern,
- In einem Konsortialprojekt,
- in einer Forschungscommunity,
- Open Source,
- Individualanfertigung für einen Auftraggeber

- Abhängig vom Nutzer*kreis ist offensichtlich unterschiedlich zu regeln:
  - Qualität
  - Installationsmechanismen
  - Dokumentationsleistung
  - Schulungsunterlagen

- Uni* = Uni + Hochschule + Forschungseinrichtung
- Nutzer* = kurz für Nutzer:innen

Kriterium 3: Entwicklerinnen-Gruppe

- Projekt-intern,
- Instituts-intern,
- Universitäts-intern,
- In einem Konsortialprojekt,
- in einer Forschungscommunity,
- Open Source

- Abhängig vom Entwicklerkreis ist offensichtlich unterschiedlich zu regeln:
  - Entwicklungsprozess
  - Tool-Infrastruktur
  - IP
  - On Boarding
  - Community Building

Kriterium 4: Beitrag der eigenen Universität

- Federführend,
- Co-Federführend,
- substantiell beitragend,
- unterstützend
- Passiv nutzend

- Abhängig davon kann die Durchsetzbarkeit von hausinternen Leitlinien ggf. nicht gewährleistet werden
- Bei “federführend” helfen gemeinsame Leitlinien
- IP-Thematik zu regeln (Uni?, Verein?)

Kriterium 5: Kritikalität

- (Wie groß sind Risiken bei Software-Versagen)?
  - Safety-critical
  - Business-critical
  - Mission-critical
  - Society-critical
- Beispiel: Medizinische Forschungssoftware, Steuerungen von Quantencomputern
- Hier gibt es klare Kategorisierungen im Bereich Safety und Security, sowie Risiko-Abschätzung & Managment
- → siehe dort

Kriterium 6: Qualität

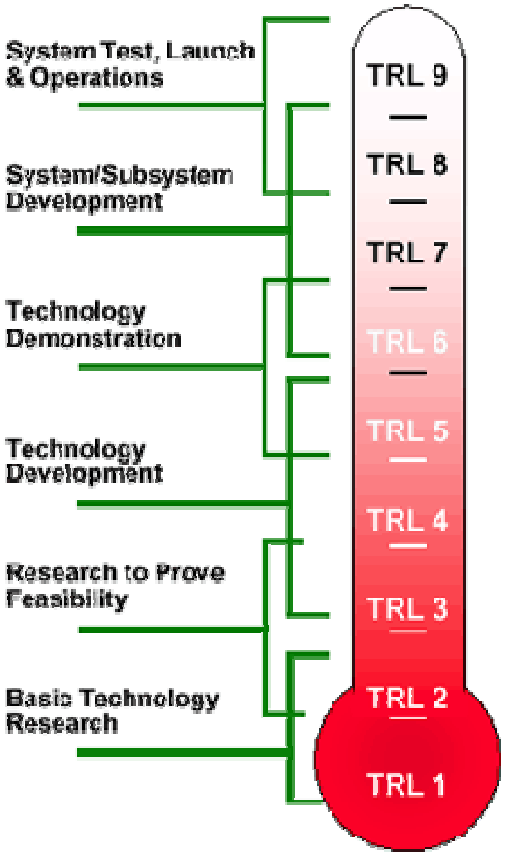
- Wie gut ist die Software einsetzbar?
- Wie gut erfüllt sie ihre Aufgaben?

- Hier gibt es klare Messverfahren im Bereich Software Engineering, aber aufwändig
-

  → siehe dort

- TRL

  Technology Readiness Levels



Zusammenfassung und Herausforderungen

- Kriterium 1: Art der Software
- Kriterium 2: Nutzerkreis
- Kriterium 3: Entwicklerinnen-Gruppe
- Kriterium 4: Beitrag der eigenen Universität
- Kriterium 5: Kritikalität
- Kriterium 6: Qualität
- Ggf. weitere?

- 1) Zu viele Kriterien, zu feingranulare Kategorien?
  - Aber: Nur 5 “Anwendungs-klassen” ist eher zu kompakt
- 2) div. Kulturen → div. Kriterienrelevanz
- 3) Ist-Kategorien vs. Soll-Kategorien
  (beides ist sinnvoll)
- 4) SE traditionell adressiert den Prozess nicht das Produkt (CMMI, Spice, …)
- 5) How about an „RSE Maturity Model“?