Herausforderungen der agilen Entwicklung wissenschaftlicher Software

Johannes Link Sascha Fendrich

vorname.nachname@heigit.org

HeiGIT gGmbH

7. Oktober 2022

HeiGIT | HEIDELBERG INSTITUTE FOR GEOINFORMATION TECHNOLOGY

Umfeld

HeiGIT Mission Statement

We provide research and development to support decision making in the field of sustainable mobility and humanitarian aid. We accomplish this through open geoinformation, open methods, open software and close collaboration with our partners.

- HeiGIT ist An-Institut der Uni Heidelberg
- Großer Anteil an projektunabhängiger Grundfinanzierung
- Viele Mitarbeiter sind ehemalige Uni-Angestellte oder Studierende
- Ziel: Forschung in die Praxis zu bringen

Herausforderungen I

- Nicht wirtschaftliche Erfolge, sondern Publikation und Anerkennung in der Forschungsgemeinde zählen
- Beteiligte sehen sich selbst oft nicht als SoftwarenentwicklerInnen, sondern als ForscherInnen
- Pflege von Software über die Laufzeit eines einzelnen Projektes hinaus ist nicht vorgesehen, aber aus Sicht langfristiger Forschung erwünscht
- Der Freiraum, dass sich jemand darum kümmert, muss erst geschaffen werden

Herausforderungen II

- Das System Universität ist eine Bildungseinrichtung und nicht für dauerhafte Karriere vorgesehen
- Für die einzelne WissenschaftlerIn ist eine Stelle ein Schritt zur wissenschaftlichen Qualifizierung, was die Teamarbeit deutlich erschwert
- Neben der Softwareentwicklung haben WissenschaftlerInnen noch viele andere Aufgaben (Forschung, Lehre, Publikation, Konferenzvorträge, Projektanträge, . . .)
- Die Herausforderung von Forschungssoftware liegt stärker in Algorithmen und Verfahren, während Business-Software oft aus einer großen Menge kleinerer Anforderungen besteht

Passen die agilen Standardprozesse?

- Habt ihr Erfahrungen in ähnlichen Kontexten?

Agile Werte leben statt fertige Prozesse implementieren

- People over Processes
- Embrace Change
- Reflect and Improve
- Sustainable Pace
- Working Software over Documentation (?)
- ...

Ideen 1: Teamstruktur

- WissenschaftlerInnen in Software-Team integrieren
- Aus mehreren WissenschaftlerInnen ein Software-Team bilden
- WissenschaftlerInnen als Stakeholder und Input-Geber für den Product-Owner

Ideen 2: Software-Techniken

- Was ist in diesem Kontext eine gute Modularisierung?
  - Ubiquitous language (DDD) orientiert sich an der Forschungsfrage
  - Algorithmen und Datenstrukturen separieren
  - PoC und stabile Teile separieren
  - Unterschiedliche Qualitätslevel (abhängig vom Lebenszyklus eines Features)
- Clean Code
- Dokumentation fokussiert auf häufigen Personalwechsel
- Good Practices (git, statische Analyse, Testautomatisierung, Compiler-Warnungen beachten, ...)
- **Risiko:** Unerfahrene können gute Ansätze überstrapazieren

Ideen 3: Vom Wissenschaftler zum Softwareingenieur

- Spezielle Workshops für WissenschaftlerInnen
- Pairing mit erfahrenen SoftwareentwicklerInnen
- Communities of practice
- **Chance:** WissenschaftlerInnen sind meist durch das Thema inhärent motiviert

Was versuchen wir bereits?

- Agiles Team aus WissenschaftlerInnen und SoftwareentwicklerInnen geschaffen
- Team-Forming als wichtiger Schritt, der Zeit, Diskussion und Lernen benötigt
- Heranführung von WissenschaftlerInnen an Themen der Softwareentwicklung und Agilität
  - Software-Engineering Book-Club
  - Agile immersion camp
- Erste Schritte, die Codebasis anhand der Qualitätskriterien zu refaktorisieren
- Software-Techniken eingeführt
  - Compiler-Warnungen beachten
  - Statische Analyse
  - Automatisierte Tests

Erste Erfahrungen I

- Die meisten Menschen wollen lieber im Team arbeiten, aber
  - Es ist anstrengend, aus einer Gruppe von unabhängig arbeitenden Personen ein Team zu schaffen
  - Der Rollenkonflikt zwischen Einzelkämpfer und Teamplayer ist real und auch nach einem Wechsel in ein Unternehmen erstmal noch da

Erste Erfahrungen II

- WissenschaftlerInnen mit ausreichend Erfahrung in Softwareentwicklung sind bereit in wartbare Software zu investieren
  - EinE WissenschaftlerIn interessiert sich nicht automatisch für Software-Engineering
  - Die Haltung "Wenn es funktioniert, ist es gut genug" wird aus Forschungsprototypen übernommen
- Widerstand gegen Veränderungsvorschläge und Kritik ist genauso groß wie anderswo

Diskussion

- Was könnte man noch versuchen um Agilität im Forschungskontext zu realisieren?

Feedback-Code