Leichtgewichtige Traceability im agilen Entwicklungsprozess am Beispiel von Scrum

Elke Bouillon[1], Baris Güldali[2], Andrea Herrmann[3], Thorsten Keuler[4], Daniel Moldt[5], Matthias Riebisch[6]
[1]Technische Universität Ilmenau, elke.bouillon@tu-ilmenau.de
[2]s-lab – Software Quality Lab/Universität Paderborn, bguldali@s-lab.upb.de
[3]Freie Software Engineering Trainerin und Forscherin, herrmann@herrmann-ehrlich.de
[4]Fraunhofer IESE, Thorsten.keuler@iese.fraunhofer.de
[5]Universität Hamburg, moldt@informatik.uni-hamburg.de
[6]Universität Hamburg, riebisch@informatik.uni-hamburg.de

Motivation

Einer der wichtigsten Erfolgsfaktoren der agilen Softwareentwicklung ist die schnelle und unkomplizierte Verteilung von Informationen. Dabei reicht das Spektrum der Informationsverteilung von einfachen Dokumenten über Wikis, Videos und Telefonaten bis hin zum "Face-to-Face" Gespräch. In der traditionellen Softwareentwicklung lag der Fokus typischerweise auf einer dokument-basierten Erfassung und Verteilung von Informationen.

Explizite Dokumentation beruht auf der Annahme, dass sich die festgehaltenen Informationen nur in bestimmtem Maße ändern. Da dieser Umstand insbesondere in der Softwareentwicklung nicht automatisch gegeben ist, hat man dies im Kontext der agilen Vorgehensweisen als ein Kernproblem von schwergewichtigen Prozessen identifiziert. Als Konsequenz dazu wird in der agilen Software-Entwicklung der Umfang der Dokumentation minimiert. Dies spiegelt sich im Manifest für Agile Software-Entwicklung wider (s. agilemanifesto.org): Obwohl die umfassende Dokumentation als wichtig erachtet wird, wird der Wert funktionierender Software höher eingeschätzt. Nach den zwölf inizialen Prinzipien hinter dem Agilen Manifest gilt die direkte persönliche Übermittlung als effizientestes und effektivstes Verfahren.

Funktioniert diese Kompensation besonders gut solange die Entwickler möglichst nah zusammen arbeiten, so ergeben sich jedoch Herausforderungen beim Versuch, agile Entwicklung in mehreren Teams und Standorten zu realisieren. Insbesondere fehlt es in der aktuellen Praxis an Techniken, welche die Dokumentation und Traceability der Entwicklungs-artefakte optimal unterstützen. Somit kann es in der Praxis vorkommen, dass Anforderungen oder Entwurfsentscheidungen dem Rest des Teams nicht mitgeteilt werden und diese Wissenslücke die weiteren Implementierungs- und Testaktivitäten negativ beeinflusst.

Unsere Hypothese ist es, dass es sinnvoll ist, Traceability in einem agilen Projekt zu unterstützen. Dies ist nach unserer Meinung notwendig, sobald ein Projekt eine Größe überschritten hat, bei der noch jedes Teammitglied das gesamte System überschauen kann oder wo ein persönlicher Kontakt, z.B. aufgrund der Laufzeit eines Projektes und der räumlichen, organisatorischen oder zeitlichen Verfügbarkeit von Personen, nicht mehr gegeben ist. Traceability-Informationen unterstützen den Projekterfolg durch

- eine Überprüfung der Abdeckung (beispielsweise der User Stories durch Code und Testfälle),
- das Konservieren von Entscheidungen,
- das Unterstützen von Entscheidungsfindungen sowie
- die Weitergabe von Wissen, z.B. bei der Einarbeitung neuer Teammitglieder.

Der Arbeitskreis „Traceability" der GI-Fachgruppe „Architektur" arbeitet aktuell an einem Ansatz für die Integration von Traceability in die agile Entwicklung. Der zu entwickelnde Ansatz soll leichtgewichtig sein und keine / möglichst wenige neue Artefakte in die agile Entwicklung einbringen, sondern die vorhandenen Artefakte nutzen und verbinden. Der Ansatz soll unabhängig von einem konkreten agilen Vorgehen sein, auch wenn wir ihn am Beispiel von Scrum entwickeln und darstellen. Wir stellen hier den aktuellen Stand unserer Überlegungen dar und die weitere geplante Arbeit.

Lösung: Konzept und Werkzeugunterstützung

Wie in Abbildung 1 dargestellt, sieht der Scrum-Prozess folgende Artefakte zur Erfassung der Anforderungen und zur Dokumentation der Entwicklungsvorgänge vor: Produkt-Backlog (PB), Sprint-Backlog (SB), Sprint-Tasks (ST). Die Entwickler setzen die Implementierungsaufgaben um, die in STs festgehalten werden, und checken ihren Code (Daily Results – DR) täglich ins Code-Repository ein. Im Laufe des Sprints werden die DRs zu funktionellen Inkrementen (I) zusammengefügt. Die Inkremente aus mehreren Sprints werden nach und nach integriert und ergeben am Ende des Projektes das fertige Produkt (P).

Zur Validierung gegen die Implementierungs-vorgaben werden die Codeteile nach jeder Entwicklungsphase getestet, z.B. mittels Continuous Integration oder nach Projektkonvention. Die DRs werden mittels Unit-Testfälle (UT) geprüft. Das Inkrement wird am Ende eines Sprints mittels Integrationstestfällen (IT) und Regressionstestfällen

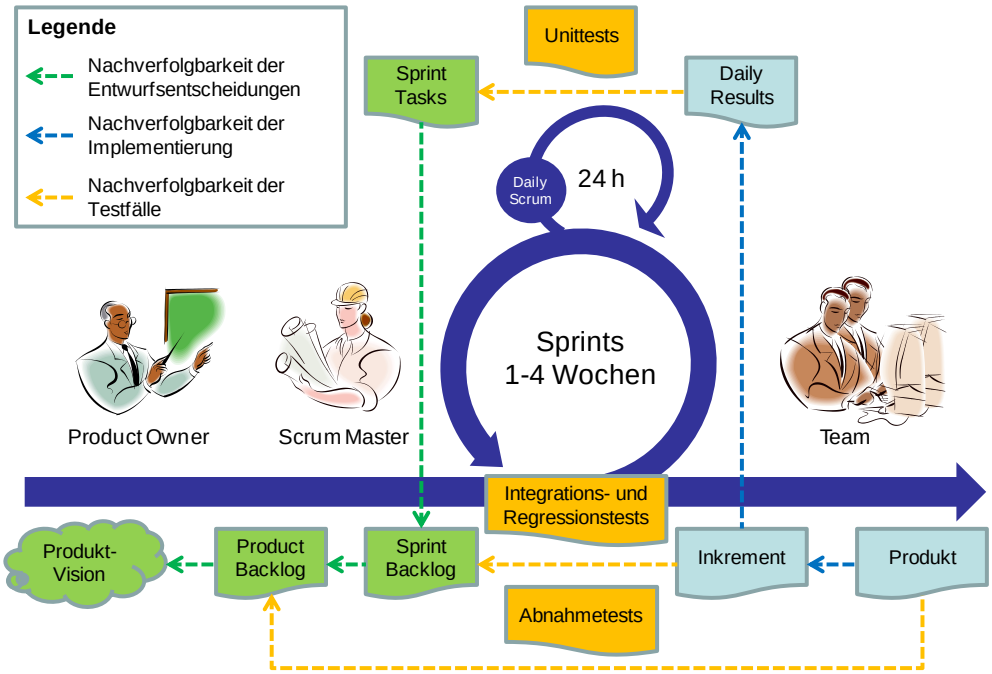

Abbildung 1. Traceability Architekturartefakte im erweiterten Scrum

(RT) geprüft. Die Inkremente aus den Sprints werden am Ende des Projektes zu einem Produkt integriert und mittels Abnahmetests (AT) geprüft.

Für die oben eingeführten Artefakte betrachten wir die Traceability

- der Entwurfsartefakte: PB←SB←ST
- der Implementierung: DR←I←P
- der Testfälle: ST←UT←DR, SB←IT/RT←I, PB←AT←P.

Existierende agile Werkzeuge[1] könnten eingesetzt und angepasst werden, um diese Verknüpfungen darzustellen und die oben aufgeführten Konzepte zu realisieren. Sie sollen dabei die folgenden Aktivitäten der Traceability-Verwaltung unterstützen:

- Erkennen von Traceability-Beziehungen zwischen existierenden Artefakten,
- explizites Erstellen von Traceability-Informationen,
- Verwalten von Traceability-Information,
- Behandeln von geänderten Traceability-Informationen,
  - Erkennen von obsoleten Traceability-Informationen nach Änderungen,
  - Nachführen von Traceability-Informationen nach Änderungen,
- Auswerten von Traceability-Information,
  - (Verständnis und Überblick,
  - Verfolgen von Entscheidungen,
  - Impact Analyse).

Fazit und Ausblick

Mit diesem Positionspapier wurde der Bedarf für Traceability in agilen Entwicklungsprozessen am Beispiel von Scrum gezeigt. Desweiteren wurden die für die Traceability relevanten Entwicklungsartefakte und ihre Beziehungen schematisch erläutert. Auch wenn die vielfältigen Werkzeuge für agile Softwareentwicklung die Traceability-Aktivitäten unterstützen, sehen wir in unserem Arbeitskreis weiteren Untersuchungs- und Evaluierungsbedarf bei der Umsetzung der im Papier dargestellten Konzepte.

Unsere nächsten Schritte in Bezug auf die Traceability für agile Vorgehensmodelle werden sein:

1. Analyse agiler Werkzeuge, wie und wie gut sie bereits die von uns vorgeschlagenen Artefakt-Verknüpfungen sowie Aktivitäten der Traceability-Verwaltung unterstützen,
2. Vorschlag eines praktischen Vorgehens und dessen Werkzeugunterstützung,
3. Diskussion dieses Ansatzes mit mehreren agilen Praktikern,
4. praktische Evaluierung dieses Ansatzes in einem oder mehreren Projekten im Hinblick auf dessen Aufwand und Nutzen.

Wir laden insbesondere Praktiker dazu ein, an der Entwicklung und Bewertung von Empfehlungen mitzuwirken. Nähere Informationen zum Arbeitskreis finden Sie unter http://www.theoinf.tu-ilmenau.de/~riebisch/traceability.

[1] Eine umfangreiche Werkzeugliste befindet sich unter http://www.agilescout.com/best-agile-scrum-tools/