Der dornige Weg zum agilen Team, Teil 5: Was soll heißen „Ich bin fertig“?

Fertig oder nicht fertig, das ist hier die Frage

VON JIRI LUNDAK

Es ist wohl einer der am schwersten zu fassenden Sätze in der Softwareentwicklung. Dabei liest er sich so einfach. Der Satz „Ich bin fertig“ scheint eigentlich keinen Interpretationsspielraum zu bieten. Und doch gibt es in den meisten Teams – und sogar zwischen einzelnen Gliedern dieser Teams – sehr unterschiedliche Ansichten darüber, was es wirklich bedeutet, „fertig“ zu sein. Diese Unterschiede führen bei einem agilen Team oft dazu, dass am Ende einer Iteration sehr unterschiedliche Qualität geliefert wird, was einem Projekt in hohem Maße abträglich ist. Wie kann das verhindert werden?

Wer kennt nicht den Fall des Entwicklers, der bereits seit drei Wochen mit dem Feature bei 95 Prozent Fertigstellungsgrad steht? Oder der das Feature implementiert hat und trotzdem jeden Monat wieder daran arbeitet. Aber es war doch schon fertig?

Aus Erfahrung als Entwickler kann ich sagen: Jeder Entwickler hat seinen eigenen Maßstab, was für ihn „fertig sein“ bedeutet. Für den einen ist es der Zeitpunkt, wenn er das Feature zu Ende programmiert hat, für den anderen, wenn er die Unit-Tests dazu implementiert hat. Und für den dritten, wenn er alle Bugs der ersten Version beseitigt und die Komponente mindestens die Versionsnummer 1.0.1 erreicht hat.

Es gibt nur ein „fertig“

Unter dem Begriff „fertig“ Unterschiedliches zu verstehen, ist sehr gefährlich und wird in einem agilen Team unweigerlich zu Problemen führen [1]. Uneinigkeit in diesem Bereich begünstigt die menschliche Tendenz, so zu tun, als ob im Projekt alles in bester Ordnung wäre, obwohl dies unter Umständen gar nicht der Fall ist. Der Vorspiegelung falscher Tatsachen und dem Verstecken von sonst offensichtlichen Problemen wird auf diese Weise nur Vorschub geleistet.

Es ist deshalb sehr wichtig – gemeinsam im Team – diesen Begriff zu definieren und die einzelnen Team-Mitglieder dahingehend zu sensibilisieren, damit alle unter diesem Begriff dasselbe verstehen. Dies kann z.B. in Form eines Brainstormings im Team geschehen (vgl. Abb. 1 für eine Mindmap, die im Rahmen eines solchen Team-Events entstanden ist).

Wichtig dabei ist, dass die Perspektive des Teams über seine eigenen Grenzen hinausreicht. Fundamental, besonders in einem agilen Projekt, ist das Einbeziehen des Kunden oder eines Kundenvertreters. Wenn das Team cross-funktional ist (d.h. neben Entwicklern und Testern auch mindestens ein Kundenvertreter Teil des Teams ist), sollte dies fast automatisch der Fall sein.

Wer bestimmt, wann es fertig ist?

Wenn es darum geht, den Zeitpunkt festzulegen, wann ein bestimmtes Feature als fertig, als vollständig umgesetzt anzusehen ist, dann gibt es eigentlich nur eine Instanz, die festlegen kann und darf, dass dieser Zustand erreicht worden ist. Es überrascht nicht, dass es sich dabei um den Kunden handelt. Warum?

Nur der Kunde weiß wirklich, was er möchte. Natürlich weiß er es nicht immer im Voraus, sondern muss das Feature „in Aktion“ sehen, bevor er es beurteilen kann. Aber deshalb gehen wir ja auch in Iterationen vor, damit er die geleistete Arbeit so oft wie möglich zu Gesicht bekommt und sein Urteil abgeben kann. Es ist zu beachten, dass in diesem Zusammenhang die Demo des Erreichten am Ende der einzelnen Iteration nicht der einzige Berührungspunkt mit dem Kunden sein sollte. Der Kunde sollte nicht von dem überrascht sein, was er am Ende der Iteration präsen-

Der dornige Weg zum agilen Team – die Serie

- Warum agile Teams so schwer zu schaffen und am Leben zu erhalten sind
- Was ist kein agiles Team?
- Warum müssen Builds immer brechen?
- Selbstorganisation – Chaos oder Wunderwaffe?
- **Was soll heißen: „Ich bin fertig“?**
- Metriken: Ballast oder Notwendigkeit?
- Abnahmen: Verhinderer des Projekterfolges?
- Brauchen agile Teams agile Kunden?
- Gedeihen agile Teams nur in agilen Unternehmen?
- Agiles Vorgehen als rotes Tuch
- Nur die Besten sind für das agile Team gut genug
- Cross-funktionale Teams – eine Illusion?
- Wo bleibt der Manager im agilen Team?
- Agile Teams skalieren nicht!
- Wie erhalte ich ein agiles Team am Leben?

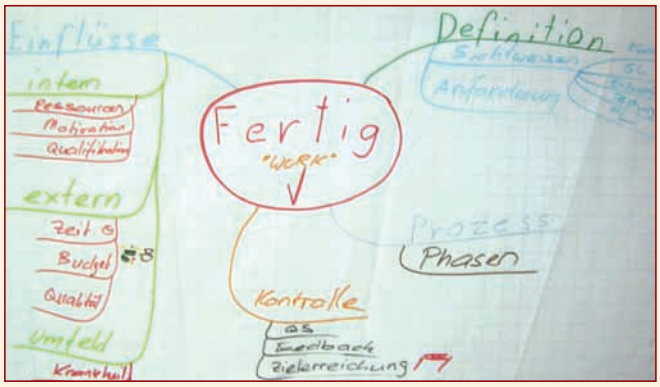

Abb. 1: Der Begriff „fertig“ im Team diskutiert

tiert bekommt. Wenn er jedoch integrierter Bestandteil des Teams ist, dann sollte dies auch nicht zu einem Problem werden.

Ob ein Feature den geschäftlichen Anforderungen des Kunden genügt, ist eine Business-Entscheidung. Aus diesem Grund kann und darf kein Entwickler diese Entscheidung treffen [2] [3]. Selbstverständlich kann man in diesem Zusammenhang Kompromisse eingehen. Die beteiligten Parteien im Team werden gemeinsam festlegen, was für den Zweck des Kunden ein „genügend gutes“ Stück Software darstellt, damit die Anforderung befriedigt ist, ohne den Aufwand für die Implementierung ins Unendliche wachsen zu lassen. Nur so kann „Goldplating“, das Veredeln der Lösung mit nicht notwendigem „Zuckerguss“, verhindert werden. Dazu ist ein anhaltender Dialog innerhalb des cross-funktionalen Teams notwendig.

Hindernisse auf dem Weg zum „fertig sein“

Wie das Foto in Abbildung 1 zeigt, gibt es viele Einflüsse, von denen der Fertigungsstand eines Softwareproduktes abhängt. Dummerweise ist in der Regel keiner dieser Einflüsse konstant. Team-Mitglieder können das Team oder gar die Firma verlassen oder neue Mitarbeiter kommen hinzu.

Die Motivation der Team-Mitglieder ist unterschiedlich. Nicht jedem liegt es, sich einem Iterationsziel verpflichten zu müssen. So kann es z.B. vorkommen, dass sich einzelne Entwickler davor fürchten, dem Kunden am Ende der Iteration etwas präsentieren zu müssen, wenn die Gesamtqualität der Software nicht stimmt. Stellen wir uns die Situation vor, wo ein Entwickler das Ergebnis seiner Arbeit dem Kunden vorstellt und es dabei NullPointerExceptions hagelt, nur weil ein anderer Entwickler bei seiner Code-Integration in letzter Minute nachlässig war. Natürlich stimmt in einem solchen Fall auch etwas im Bereich Qualitätssicherung nicht. Aber diese Erfahrung wird dazu beitragen, dass der betroffene Entwickler – und eventuell auch der Kunde – sein Verständnis von „fertig sein“ revidiert. Der Entwickler wird auf jeden Fall viel vorsichtiger sein, dem Kunden etwas auf das Iterationsende hin zu versprechen.

Ein anderes Problem, das es außerordentlich schwierig macht, den Zustand des „fertig seins“ zu erreichen, ist das Bearbeiten von möglichst vielen Features zur gleichen Zeit. Dies ist dann der Fall, wenn jedem Entwickler (wie dies oft bei traditionellen Entwicklungsprozessen der Fall ist) ein ganzer Geschäftsprozess zugeteilt wird, den er implementieren soll. Diesen Geschäftsprozess fertig zu stellen dauert geschätzte vier Wochen. Wird die Implementierung dieses Stückes der Applikation in einer Monatsiteration fertig zu stellen sein? Wohl kaum! Und doch kann der Entwickler versucht sein, des lieben Fortschrittsberichts wegen zu behaupten, der Geschäftsprozess sei fertig gestellt, sobald er die Erstimplementierung abgeschlossen hat. Dabei hatte der Tester vielleicht noch gar nicht die Möglichkeit, den Code auf Herz und Nieren zu prüfen, geschweige denn hatte der Kunde die Gelegenheit zu überprüfen, ob das System nun seine Anforderungen erfüllt. Welches Verständnis von „fertig“ war nun in der Schätzung enthalten? Hätte man den Geschäftsprozess nicht lieber auf kleinere Aufgaben aufgeteilt und auf mehrere Schultern verteilt? Im Kasten „Ist das Feature fertig?“ sind einige Fragen zu finden, die helfen können, ein gemeinsames Verständnis von „fertig sein“ zu definieren.

Wie hier gezeigt wurde, müssen selbstorganisierende, agile Teams innerhalb eines klar definierten Containers agieren, damit sie nicht ins Chaos abgleiten. Das klare, gemeinsame Verständnis darüber, was es bedeutet, fertig zu sein, gehört mit zu den Rahmenbedingungen, die geschaffen werden müssen, um dem Team zu helfen, seine Ziele zu erreichen.

Ist das Feature fertig?

- Existieren Tests, die beweisen, dass die Anforderung des Kunden umgesetzt wurde? War der Kunde an der Definition dieser Tests beteiligt und ist er damit einverstanden, dass – wenn die Tests laufen – das Feature damit genügend verifiziert ist?
- Wurde gemeinsam mit dem Kunden festgelegt, wann ein Feature fertig ist?
- Kann der Kundenvertreter, der im Team mitarbeitet, selbst informierte Entscheidungen treffen, wann er ein Feature akzeptiert? Hat er die nötige Autorität oder kann er innerhalb kürzester Frist die Entscheidung herbeiführen?
- Machen die Entwickler oder Tester dem Kunden die Entscheidung über das „fertig sein“ streitig, nach dem Motto: „Der versteht ja doch nichts davon!“?
- Machen wir es dem Kunden so leicht wie möglich, beurteilen zu können, ob ein Feature fertig ist (z.B. Visualisierung von Features, die an der Oberfläche nicht sichtbar sind, wie Batch-Jobs und Hintergrundprozesse etc.)?
- Haben alle Team-Mitglieder von Beginn des Projektes an dasselbe Verständnis von „fertig sein“?

Jiri Lundak ist mit mehr als 20 Jahren Erfahrung ein Urgestein der Softwareentwicklung. Er arbeitet gegenwärtig als Entwicklungsleiter für Löwenfels Partner in der Schweiz. Er entwickelt als Architekt immer noch aktiv Software, hat aber erkannt, dass die meisten Probleme in Entwicklungsprojekten nicht technischer, sondern vielmehr menschlich-sozialer Natur sind. Als praktizierender Certified ScrumMaster lebt er die Rolle eines Vorkämpfers für agiles Verhalten innerhalb der Firmenorganisation. Kontakt: jiri.lundak@loewenfels.ch.

Links & Literatur

[1] Peter Clark: When Is Done Really Done?: www.stickyminds.com/s.asp?F=S7621_COL_2

[2] Johanna Rothman: Release Criteria: Is This Software Done?: www.jrothman.com/Papers/releasecriteria.html

[3] Johanna Rothman: Of Crazy Numbers and Release Criteria: www.jrothman.com/Papers/crazynumbers.html