Notizen

Mit diesem Lieferanten sollten wir die Verträge so gestalten, dass wir im Verlauf der Entwicklung gemeinsam herausfinden können, wie das Zulieferteil genau aussehen soll:

Am Ende dieses Abschnitts möchte ich noch ein kleines Problem mit großen Auswirkungen ansprechen, das nichts mit der Beschaffungsstrategie zu tun hat. Es geht um die Liegezeiten im Beschaffungsprozess. Recht häufig machen die internen Liegezeiten einen signifikanten Anteil an der Zeit zwischen der Bestellung durch die Entwicklung und der Ankunft der Teile aus. Weißt du, wie die Liegezeiten in eurer Einkaufsabteilung aussehen? Natürlich gibt es auch Liegezeiten bei deinen Teams, zum Beispiel bevor die Bestellung ausgelöst wird oder bei der Beantwortung von Rückfragen durch den Einkauf; diese Themen könnt ihr aber eigenständig in den Griff bekommen. Die Einkaufsabteilung ist hingegen oft räumlich, organisatorisch und in puncto »Awareness« weit von der Entwicklung entfernt. Deshalb ist ein Abgleich zwischen den Organisationseinheiten und den einheitlichen Zielen des Topmanagements unerlässlich. Im Idealfall schafft der Einkauf Transparenz durch Backlogs und Kanban-Boards und kann die (hoffentlich) vorgegebenen Prioritäten zwischen den Entwicklungsvorhaben optimal in seine tägliche Arbeit integrieren.

Falls du wirklich den Einkauf mit Kanban beglücken möchtest, findest du mehr Infos im blauen Buch ab Seite 135.

3.2 Das Inkrement

Im vorhergehenden Kapitel habe ich die Verkürzung von Zykluszeiten in der Entwicklung durch die Veränderung mancher Engineering-Praktiken besprochen. Jetzt möchte ich auf das »Inkrement« eingehen. Das Inkrement ist ein essenzieller Bestandteil der agilen Entwicklung und ist vermutlich auch jener Aspekt, der auf den ersten Blick Scrum und physische Produkte schlecht vereinbar erscheinen lässt.

3.2.1 Bedeutung

Warum überhaupt »iterativ-inkrementell« entwickeln? Das Inkrement ist Dreh- und Angelpunkt in Scrum. Es dient der Fortschrittsmessung, der Analyse der Machbarkeit, der Validierung von Anforderungen und manchmal kann es sogar schon vom Kunden verwendet werden. Das Inkrement schafft die maximal mögliche Transparenz über die aktuelle Situation und ist die Basis für die Anpassung der weiteren Pläne.

Versuche dir mehr Klarheit über den Nutzen eines Inkrements in deinem Kontext zu verschaffen. Lies dir die möglichen Nutzen des Inkrements in der folgenden Auflistung durch und versuche, diese für dich zu priorisieren:

- Fortschrittsmessung: Wo stehen wir gerade? Was funktioniert? Was funktioniert noch nicht?
- Rückmeldung zur Machbarkeit: Ist die angestrebte Lösung zuverlässig, produzierbar usw.?
- Rückmeldung zu den Anforderungen: Ist es das, was der Kunde will?
- Früher Kundennutzen: Kann der Kunde die Inkremente schon sinnvoll für sich einsetzen?

Unsere Vorteile durch das Inkrement:

1. _____

2. _____

3. _____

4. _____

Die für dich wichtigen Aspekte eines Inkrements können sich im Lauf der Zeit verändern Am Beginn des Entwicklungsvorhabens sind vielleicht andere Rückmeldungen wichtiger als später, wenn es schon in Richtung Serienproduktion geht.

Mehr Informationen zum Nutzen eines Inkrements findest du im roten Buch ab Seite 79.

Prinzipiell kannst du also folgende Arten von Inkrementen unterscheiden:

- Das Produkt als Inkrement
- Modellbasierte Inkremente (abgeschwächte Reifegrade oder Abstraktionen des Produkts, dazu gleich mehr)
- Ideelle Inkremente (Inkremente, die z. B. Wissen aufbauen oder Werte festigen)

3.2.2 Reifegrade

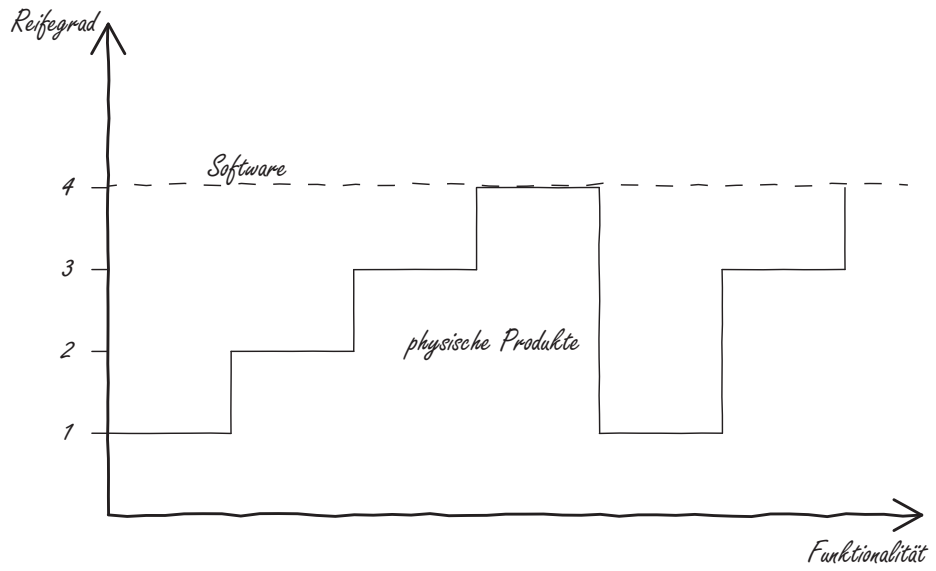
Wenn es um die Verkürzung der Zykluszeiten in der Entwicklung geht, ist ein naheliegender Ansatz, nicht immer das komplette Produkt, sondern einzelne Artefakte im Entwicklungsprozess als Inkrement zu sehen, wie Zeichnungen, Konzepte, Analysen usw. Wenn wir die eigentliche Forderung von Scrum, ein Inkrement zu erstellen, das potenziell produktiv eingesetzt werden könnte, als ein Ende einer möglichen Skala sehen, wären Zeichnungen & Co. als Inkremente am anderen Ende der Skala angesiedelt. Zeichnungen oder Konzepte sind aus meiner Sicht keine brauchbaren Inkremente, diese Beispiele sind viel eher unbestätigte Theorien. Du kannst das Beispiel einer Zeichnung mit den vier Nutzen des Inkrements aus dem vorigen Abschnitt vergleichen, um zu sehen, dass dies zwar ein einfacher Ansatz wäre, aber nicht dem Inkrement-Gedanken dient. Viel mehr würde es sich dann um klassisches Projektmanagement handeln und Scrum würde zu einer »Methode« degradiert, mit der man die Aufgaben im Projekt verwaltet.

Es gilt also, den Inkrement-Gedanken so gut wie möglich beizubehalten und trotzdem eine Vereinfachung gegenüber der – in der Software einfacher umzusetzenden – Maximalforderung zu erreichen. Der Schlüssel dazu ist, zwar immer das ganze Produkt als Inkrement zu sehen, aber in der Reife des Produkts flexibel zu bleiben. Das ist natürlich ein Verstoß gegen den Scrum-Gedanken, denn Inkremente mit reduzierter Reife sind eben nicht potenziell auslieferbar. Der Ansatz ist jedoch, diesen einen Nutzen zu opfern, um die anderen drei Nutzen möglichst gut mitnehmen zu können.

Inkremente in diesem Sinn könnten also zum Beispiel sein:

- Simulationen
- Mockups
- Wegwerf-Prototypen
- Produkte mit anderen Materialien (z. B. Eval-Boards)
- Produkte mit anderen Fertigungsverfahren/Werkzeugen (z. B. 3D-Druck)
- Produkte mit Komponenten von Vorgängerversionen

Die Reife muss im Zeitverlauf nicht streng monoton ansteigen. Es kann durchaus Sinn ergeben, beim Hinzufügen einer neuen Funktionalität in der Reife wieder nach unten zu gehen. Wichtig ist, zumindest eine grobe Roadmap für die Reife im Zeitverlauf zu machen, insbesondere dann, wenn es bestimmte Kundenmeilensteine gibt, an denen du Inkremente bzw. Muster abliefern musst. In diesem Punkt unterscheidet sich das Vorgehen von jener in der Software-Entwicklung, wo immer mit der maximalen Reife gearbeitet wird (Abbildung nächste Seite).



Nimm dir an dieser Stelle etwas Zeit, um über für dich sinnvolle Reifegrade nachzudenken. Im nächsten Kasten kannst du verschiedene Szenarien durchspielen, um das für dich passende Modell der Reifegrade zu finden. Du musst nicht unbedingt auf fünf Reifegrade kommen, vielleicht ergeben in deinem Fall auch nur zwei oder drei Reifegrade Sinn. Mehr als fünf würde ich persönlich aber nicht verwenden.

Reifegrade unseres Produkts (eventuell mehrere Szenarien):

3.2.3 Modellbasierte Inkremente

Eine andere Sicht auf die im vorherigen Abschnitt beschriebenen Reifegrade ist, dass ein Inkrement ein Modell des eigentlichen Produkts im Sinne des Model Based Systems Engineering (MBSE) darstellt. Falls du dich mit MBSE beschäftigt, kommt dir diese Sicht vielleicht gelegener. Um in die Diskussion einzusteigen, möchte ich zunächst noch einmal kurz darstellen, was ein Modell (nach der allgemeinen Modelltheorie) ausmacht. Ein Modell

- bildet ein Original/die Wirklichkeit ab,
- reduziert/abstrahiert das Original und
- macht dies im Hinblick auf einen bestimmten Nutzen/Zweck.

Bezogen auf unseren Kontext der agilen Produktentwicklung bedeutet dies, dass ein Modell alles ist, was das wirkliche Produkt in irgendeiner Form zu einem bestimmten Zweck abstrahiert. Somit gelten nicht nur digitale Modelle als Modelle, sondern auch physische Prototypen oder Mockups.

Was den Zweck der Modelle betrifft, lässt sich im Systems Engineering prinzipiell zwischen Modellen für die Beschreibung des Problemraums und solchen für die Beschreibung des Lösungsraums unterscheiden. Agile Ansätze wie Scrum bringen bereits bestimmte Modelle mit: So können zum Beispiel Backlogs oder eine User Story als Modelle für den Problemraum angesehen werden. Modelle lassen sich auch dahingehend unterscheiden, ob sie formal oder informal sind, also ob sie nach bestimmten Regeln oder in einer bestimmten Syntax erstellt wurden bzw. vorliegen. Bei den formalen Modellen kann noch unterschieden werden, ob das Modell maschinell interpretierbar bzw. ausführbar ist oder nicht.

Die beschriebene Zweckgebundenheit (nach der Modelltheorie »der Pragmatismus«) eröffnet für agile Produktentwickler:innen ein weites Feld, um Modelle in Scrum einsetzen zu können. Als Impulse habe ich in der folgenden Tabelle ein paar Beispiele aufgeführt:

Modell	**informal (i) formal (f) ausführbar (a)**	**Problem-raum (P) Lösungs-raum (L)**	**Zweck**	**Nutzbar als Inkrement**
Kontext-Diagramm (SysML)	f	P	Verständnis Systemgrenzen	n
Use Case Diagramm	f	P	Verständnis Anwendungs-fälle	n
User Story	i	P	Verständnis Anwendungs-fall/Anforde-rungen	n
Mockup (z. B. Design-Proto-typ)	i	P	Verständnis Anforderun-gen	j