Agile Software-Entwicklungs-Methodik

sei nur über ihren Zweck definiert
(andere Definitionen erwiesen sich als zu gefährlich)

Zweck ist:

Software muss die am Tag ihrer Auslieferung gewünschten Anforderungen erfüllen (nicht schon überholte). Sie muss zudem langfristig gut wartbar sein.

Gebhard Greiter, 2011

Agile Methodik entstand als Reaktion auf die Erkenntnis, dass es heute nicht mehr zielführend ist, Software so zu entwickeln, dass

- zunächst Anforderungen definiert werden,
- die — sobald der Auftraggeber sie als zutreffend und ausreichend akzeptiert hat — über längere Zeit als ***nicht mehr in Frage zu stellende*** Definition dessen gelten, was zu implementieren ist.

Um bekannter zu machen, dass sie diesem Prinzip nicht mehr folgen wollen, haben die Erfinder und Befürworter von Extreme Programming 2001 das Agile Manifesto publiziert.

In Formulierung des Manifests machten sie aber den Fehler, zur Lösung des Problems einen Weg zu skizzieren, der vielen Entwicklern willkommene Ausrede dafür war, fast alles zu vergessen, was bis dahin propagierte Vorgehensmodelle an notwendiger Spezifikation gefordert hatten: schriftliche Dokumentation aller Anforderungen und ganz besonders auch genaue Dokumentation wichtiger Schnittstellen und ihrer Leistung.

Mehr und mehr hat man dann — versehentlich — eben diesen in vieler Hinsicht zu wenig durchdachten Weg als die eigentliche Definition Agiler Methodik gesehen und dabei völlig ignoriert, dass man so nur bis Ende der Implementierungsphase denkt: ***Unverzichtbare Anforderungen wartungstechnischer Art für die erst dann beginnende wichtigste Phase im Lebenszyklus der Software werden so weder diskutiert noch berücksichtigt* (!)**.

Von zu wenig nachdrücklichen Warnungen Einzelner einmal abgesehen hat erst Gartner auf diesen *folgenreichen, wirklich schweren* Fehler deutlich genug hingewiesen (das allerdings erst nachdem Agilisten schon 10 Jahre lang ihren vermeintlich besseren Prozess propagiert hatten — sie finden ihn heute noch gut).

Leider ignoriert die Diskussion um Agile zudem noch ein anderes Problem:

Wer nämlich fordert (was sinnvoll ist), dass die Anforderungsdefinition auch noch während der gesamten Implementierungsphase abänderbar sein muss, müsste dann auch deutlich machen, dass diese neue Regel es praktisch unmöglich macht, Entwicklungsprojekte zum Festpreis abzuwickeln. Diese unangenehme Wahrheit aber

- will kein Auftraggeber hören und
- wollen, aus der Furcht heraus, so ein Projekt dann gar nicht erst zu bekommen, auch die Auftragnehmer (zunächst jedenfalls) lieber *nicht* diskutieren.

Dass solche Vogel-Strauß-Politik nicht lange gut gehen kann, wird allzu sehr verdrängt. Frage also:

Ist es nicht wirklich höchste Zeit,
wieder professioneller zu argumentieren und zu arbeiten?

Best Practice Agile wäre guter Einstiegspunkt hierfür: Es muss erreicht werden, dass die Bereitschaft der Auftragnehmer, jederzeit Änderung der Anforderungen zu akzeptieren, mit als Leistung zählt, die ein Honorar wert ist. Es explizit in Rechnung stellen zu können, muss selbstverständlich werden (ist es aber nach derzeitigem EU-Ausschreibungsrecht noch lange nicht — ein wirklich großes Problem).

Die eben geführte Diskussion zeigt deutlich, dass man Agile Methodik *ganz grundsätzlich nur über ihren Zweck* definieren sollte (keinesfalls aber über einen bestimmten Weg hin zu diesem Zweck).

Wer diese Regel missachtet gerät in Gefahr, sich methodisch im Kreis zu bewegen und/oder zu vergessen, dass es bessere Wege geben könnte.

References and the New Definition of Agile

Three observations have shown us that we should no longer see the Agile Manifesto as the ultimate definition of Agile:

- In 2006 already John Rusk wrote:

  "Agile development is hard to define, because most people define it by giving examples. For instance they give a specific description of Extreme Programming (XP), instead of defining agile development in general. We've ended up with a widespread misconception that agility is about XP techniques like Pair Programming and Test-Driven Development (TDD). …

  "Defining agility by describing XP is like defining democracy by describing America. Such a "definition" obscures the underlying concept with details of the chosen example."

- James and Suzanne Robertson (senior consultants at Cutter Consortium) even dare to ask "Is Agile Shortchanging the Business?".

  What they tell us about the concerns a growing number of their clients have may well be seen as a proving the fact that what Agile Methods promise to achieve in terms of value is by far not clear enough.

  Even worse:

- At the Gartner BPM Summit (in March 2011) analyst David Norton told us:

  "CIOs should be aware that although agile software development is cheaper in the short term, the costs are often hidden in the long term maintenance. Agile development is often perceived to be a cost-effective solution, as small changes are typically added in to respond to a dynamic environment. This compares with traditional approaches where development may be done on a large scale."

  However, Norton said that because the initial costs are relatively small, CIOs often forget there are costs in the long term:

  "I was speaking to a CIO about agile development and I asked him how he felt about it in his organisation. His response was, 'I wish I had never heard of it'," said Norton. "Agile had solved his problems of productivity and not being

  responsive to the business, but two years into it and he had hundreds of different solutions that duplicate code and requirements, and his asset reuse had taken a nose dive," he added.

  "This is one of the main things we need to think about – how do we use and leverage the benefits of agile? But also, how does it adapt in the long term?"

  Norton went on to describe some "horrific figures", which suggest that "Only eight per cent of capital and operating agile development costs are in the initial delivery of applications. The remaining 92 per cent is found in maintaining the development over the next 10 to 15 years".

  "This is a something that CIOs need to be aware of and is something we haven't really focused on in the agile community," said Norton.

The problem definitely is that the Agile Manifesto is a far too specific definition of Agile: It is a definition of principles assumed to be helpful but not a definition of the goal to achieve.

From now on we better focus on the following (new) definitions:

Software Developer's Definition of Agile:

Agility means to have a process in place that will allow us (and urge us)
to react on changing business requirements as soon as possible:
— Accept that the project's goal is a moving target —

Project Manager's Definition of Agile:

Agile Project Management means to have a process in place
that is to maximize team efficiency,
user satisfaction,
and maintainability of the product independent of specific persons.

Only this can reduce TCO, satisfy the User, and also bring about commercial success for the developers.

To accept that the project's goal is a moving target means:

- Owner and representative future users of the software (users/owner) should always be able to see what you are going to implement or have already implemented.
- As soon as you learn or suspect that users/owner might not be happy which a specific feature you plan to implement (or have already implemented), you are to speak to them in order to clarify what they really need or want:
  - — Update the requirements specification accordingly;
  - — then go to implement this new version of the requirements.

Suggestions to re-focus Agile in this sense are Agile Principles 2011 and the SST (Specify, Subcontract, Test) process for software development.

WARNING:

To accept that the project's goal is a moving target is going to cost money.

Clients need to understand that the developer's promise to promptly react on updated requirements is a value well worth to be payed for.

Author's homepage: http://greiterweb.de/spw/