Werkzeuge werden gerne und oft diskutiert.

Oft bestimmen die Werkzeuge die Methoden und nicht umgekehrt.

Werkzeuge versprechen erhöhte Produktivität bewirken aber zu oft das Gegenteil. Werkzeuge sind Machtmittel, wenn über Lizenzen oder Know-how der Zugang versperrt ist.

- Wie sehen heute Werkzeugketten für Softwarearchitekten aus?
- Wie erreichen Sie die Nachvollziehbarkeit der Ergebnisse von den Anforderungen bis zur fertigen Software?
- Wie treffen Sie die richtige Wahl der Tools?

Sprecher: Wolfgang Fahl

- Wolfgang Fahl
- Diplom-Informatiker RWTH Aachen

Geschäftsführender Gesellschafter BITPlan GmbH

Wolfgang Fahl ist Diplom-Informatiker und Geschäftsführender Gesellschafter der BITPlan GmbH. Er ist seit 20 Jahren im Software-Engineering tätig.

Softwarequalität liegt ihm besonders am Herzen, seit dem er Software für lebenserhaltende Systeme mitentwickelt hat. Seit 1996 ist Wolfgang Fahl als Berater und Unternehmer aktiv um Organisationen zu unterstützen, Software so zu entwickeln, dass das Ergebnis "passt".

VHIT

VHIT=„Vom Hirn ins Terminal“
ist die bis heute erfolgreichste
„Werkzeugkette“

Solange die Software von dem geschrieben wird, der auch die Anforderungen an die Software festlegt ist der VHIT Ansatz möglich. Das Hauptwerkzeug ist dann das menschliche Hirn. Die Qualität des Ergebnisses hängt vollständig an den Fähigkeiten des „Hackers“ der die Software auf Basis seiner Ideen direkt in Code umsetzt.

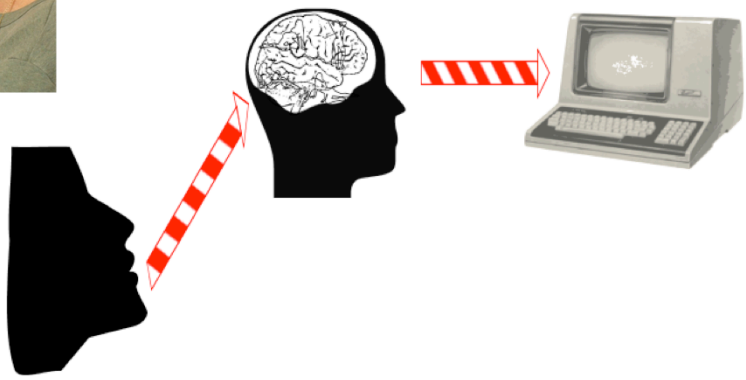
Der einsame Programmierer muss sich mit dem Fachgebiet auskennen, für das die Software erstellt wird. Für Fachgebiete außerhalb der Informatik sind daher nur Leute geeignet die sich Programmierkenntnisse angeeignet haben.

Eine ganze Reihe erfolgreicher kommerzieller Software für die unterschiedlichsten Branchen ist auf diese Weise entstanden. Interessant dabei ist, dass das bessere inhaltliche Verständnis häufig genug die schlechteren Programmierfähigkeiten gegenüber professionellen Informatikern ausgeglichen hat. So ist so manche Branchenlösung ab den 80er Jahren zunächst einmal auf diese Weise entstanden.

Erst mit zunehmendem Erfolg am Markt bestand die Notwendigkeit das Softwareentwicklungsverfahren zu erweitern und zu verbessern.

Zweihirnwerkzeugkette

Sobald jemand die Anforderungen „zuflüstert“ wird die Werkzeugkette länger:



So bald zwei oder mehr Personen an der Softwareentwicklung beteiligt sind, ist nicht mehr sichergestellt, dass die Anforderungen so wie eigentlich gedacht umgesetzt werden. Es kommt das Dilemma der menschlichen Kommunikation zum Tragen.

„Das hab‘ ich nicht gesagt“ und „Du kannst mich einfach nicht verstehen“ sind nicht umsonst beliebte Buchtitel.

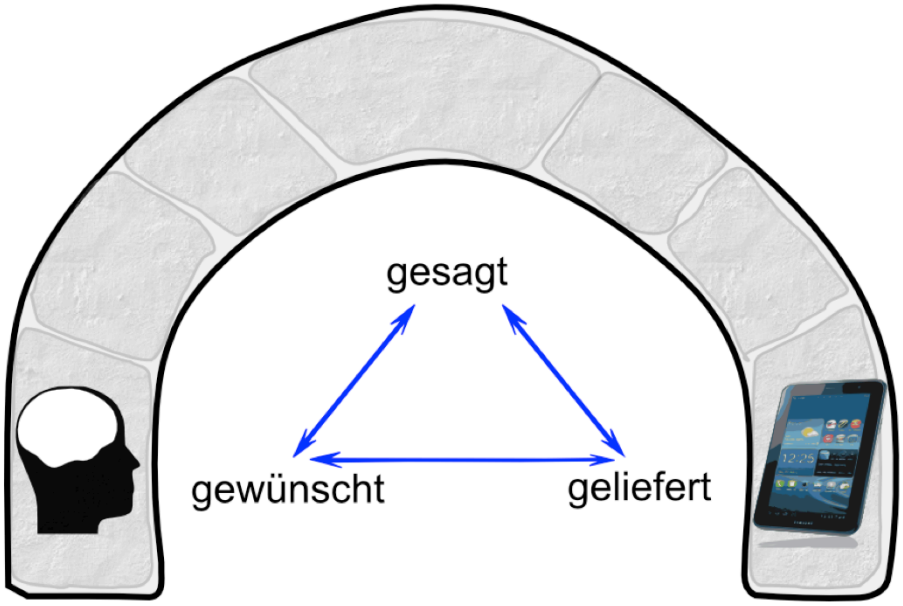
Ab der Zweihirnwerkzeugkette treten alle Probleme auf, die sich aus der Verteilung des Wissens ergeben. Der verbindliche Stand der Software ergibt sich noch immer nur aus dem erstellten Source-code.

Erst beim Wechsel des Programmierers zeigt sich, ob und wie dieser Ansatz langfristig trägt.

Auftragssoftware entsteht noch heute sehr häufig nach dem Zweihirnverfahren.

BITPlan

Die Brücke



Worum geht es bei der Qualität von Software?

- Wie gut soll das gesagte mit dem gewünschten übereinstimmen?
- Wie gut ist die Übereinstimmung von Gewünschtem und Gesagtem möglich?

- Wie gut soll das Gelieferte mit dem Gesagten übereinstimmen?
- Wie gut ist die Übereinstimmung von Geliefertem und Gesagten möglich?

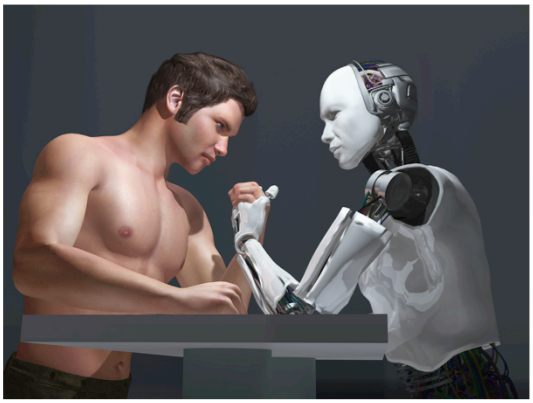
- Wie gut soll das Gelieferte mit dem Gewünschten übereinstimmen?
- Wie gut ist die Übereinstimmung von Geliefertem und Gewünschten möglich?

Auf der linken Seite der Brücke geht es darum herauszufinden was die gewünschte Software leisten soll und wie sie es leisten soll.

Auf der rechten Seite der Brücke geht es um die handwerkliche Umsetzung des „Gesagten“ in Bits und Bytes die dann schließlich als Software geliefert werden.

Mit zunehmender Komplexität der Software und der zugehörigen Entwicklungsorganisationen wird es erforderlich die Aufgaben auf mehr Köpfe zu verteilen und der Bedarf an Methoden und Werkzeugen steigt.

Mensch oder Werkzeug?



- Benutzbarkeit
- Erreichbarkeit
- Nachvollziehbarkeit
- Zusammenarbeit
- Geschwindigkeit
- Zuverlässigkeit
- Kosten

…

Wie soll der Weg über die Brücke verlaufen? Nachdem sich heraus gestellt hat, dass es nicht so einfach ist, Software so zu erstellen, dass die Beteiligten mit dem Ergebnis zufrieden sind gab es einige Vorschläge:

- Prozessverbesserung
- Methodeneinsatz
- Werkzeugeinsatz

die jeweils das Ziel verfolgten die Softwareentwicklung zu verbessern. Die typischen Qualitätsattribute des Endproduktes kann man auch auf den Web über die Brücke anwenden.

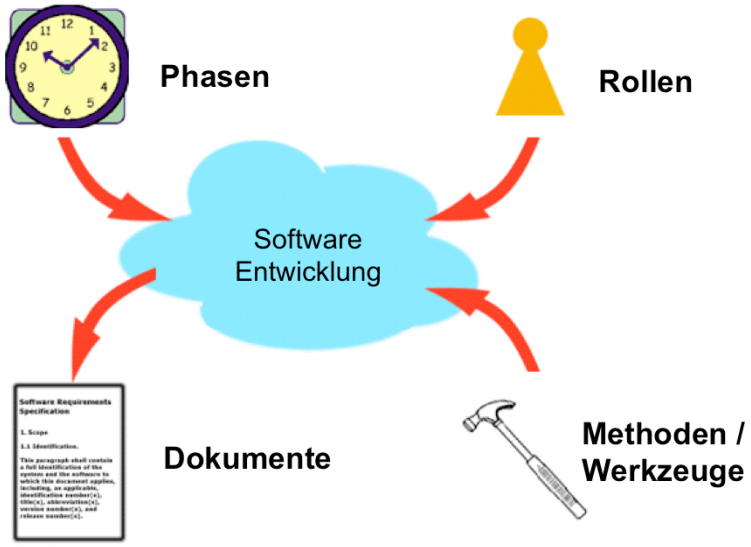
Im agilen Manifest findet sich dazu die Aussage:

"… We have come to value… Individuals and interactions over processes and tools" – " … That is, while there is value in the items on the right, we value the items on the left more. …"

Siehe http://agilemanifesto.org

Die plakative Darstellung „Mensch oder Werkzeug" oder wie oben dargestellt sogar „Mensch gegen Werkzeug" halte ich persönlich für übertrieben.
Auch die Behauptung des englischen Wikipedia „CASE"-Artikels:
„CASE tools were at their peak in the early 1990s" teile ich nicht.

BITPlan **Software-Entwicklungsprozess**



Es sind viele Ideen entwickelt worden, wie Software möglichst wohlorganisiert entwickelt werden kann. Die meisten Ideen bedienen sich der oben genannten Begriffe:

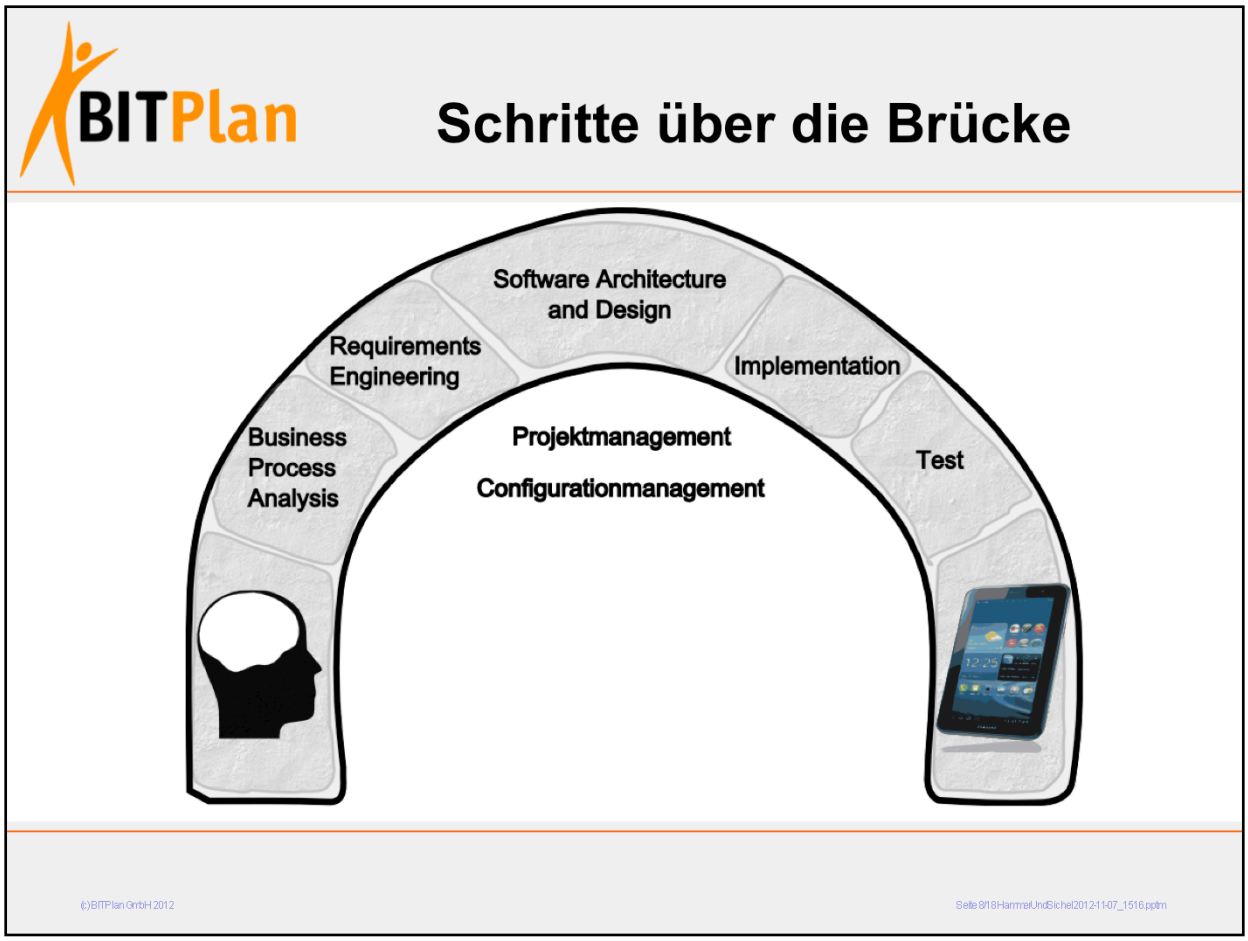
- Phasen: zur zeitlichen Einteilung und Festlegung der Reihenfolge von Aktivitäten
- Rollen: zur Aufteilung der Arbeitsblöcke auf die Beteiligten
- Dokumente: zur Festlegung der erforderlichen Ergebnistypen
- Methoden/Werkzeuge: zur systematischen und (teil-)automatisierten Bearbeitung

Der zugehörige Wikipedia-Artikel

http://de.wikipedia.org/wiki/Liste_von_Softwareentwicklungsprozessen

zählt über 30 verschiedene Vorgehensmodelle für die Softwareentwicklung auf.

Entsprechend ist die Liste möglicher Methoden und Werkzeuge noch erheblich länger.



Das oben gezeigte Beispiel zeigt fünf Schritte über die Brücke mit zwei unterstützenden Querschnittsfunktionen. Insgesamt sind hier 7 Rollen zu besetzen. Für alle diese Rollen und Funktionen gibt es spezielle Methoden und Werkzeuge.

Schritte über die Brücke

Methoden:
BPA, de Bono, Five Why's, MoSCoW, PESTLE, HEPTALYSIS, MOST, SCRS, SWOT, VPEC-T, …

Standards:
BPMN, EPC/EPK, …

Werkzeuge:
Abacus, Adonis, ARIS, Cameo, ezBPR, Fujitsu ABPD, iGrafx, Abacus, Pegasystems, Corporate Synergy, Visio, Innovator, …

Hersteller:
IDS Scheer, IBM, Metastorm, iGrafx, Mega, Casewise, Microsoft, EMC, Sybase, Lombardi, Business Genetics, Savvion, Tibco Software, Ultimus, Sparx Systems

Als Beispiel sei nur einmal der „erste Baustein“ der Brücke (ohne Anspruch auf Vollständigkeit) untersucht.

Es wird leicht deutlich, dass die Zeit für diesen Vortrag auf keinen Fall ausreicht um die gesamte mögliche Toollandschaft des Softwareengineering bis in dieses Details zu betrachten. Darum soll es hier auch gar nicht gehen.

Werkzeuge und Menschen

- Wir haben kein Tool
- Ich habe ein Tool
  und Du hast keins
- Ich komme mit dem Tool klar
  und Du nicht
- Ich habe eine Lizenz
  und Du hast keine
- Ich habe ein Tool
  und Du hast ein anderes

Egal welche Tools im Einsatz sind ergeben sich aus dem Einsatz oder Nicht-Einsatz von Werkzeugen einige Standardprobleme. Oben ist eine kleine, aber wichtige Auswahl gezeigt.

Werkzeugt machen nur dann Sinn, wenn Sie verfügbar, verstanden und akzeptiert sind. Der Cockburn-Test hat in diesem Sinne nicht nur für Methoden sondern auch für Werkzeuge seine Gültigkeit:

Siehe Alistair Cockburn, Agile Software Development, p 145:

- The project was delivered ..
- The leadership remained intact …
- The people on the project would work the same way again

D.h. mit den Werkzeugen muss wirklich produktiv und nützlich gearbeitet worden sein.

Das Projekt bzw. die Organisation / Gruppe hat den Werkzeugeinsatz „überlebt“ – ist z.B. nicht pleite gegangen.

Die Beteiligten würden beim nächsten mal wieder freiwillig das/ die Werkzeuge einsetzen.

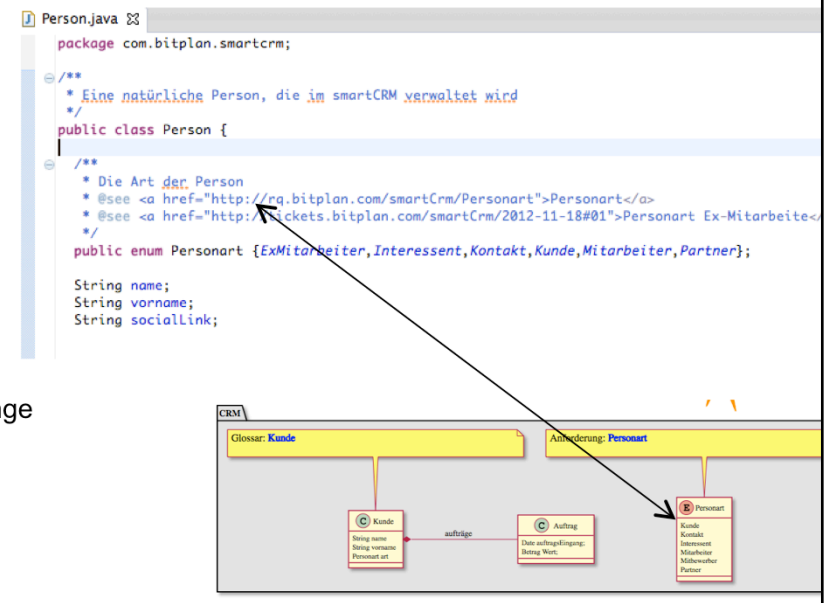
The broken bridge/toolchain

- Verringerung der Produktivität, nutzlos
- Störung der Zusammenarbeit
- Unzusammenhängende Ergebnisse
- Insellösungen, Sackgassen

Werkzeuge können das Gegenteil von dem bewirken was damit beabsichtigt ist. Dieser Effekt tritt bei Werkzeugketten noch stärker auf als bei Einzelwerkzeugen. Diese Negativeffekten gilt es zu vermeiden. Mehr dazu später …

Traceability

- Anforderungen
  - Offene Fragen
- Modell
- Quelltext
- Handbuch
- Testfälle
- Projektplanung
- Änderungen
  - Bug Reports/Change Requests
- Support
  - Support Requests
- ...



Ein wichtiges Kriterium für die Zusammenstellung einer Werkzeugkette ist die Möglichkeit der Nachvollziehbarkeit von Ergebnissen. Vordergründig könnte man meinen, dass es vor allem wichtig ist vorwärts schauend von den Anforderungen zur fertigen Software zu blicken (im Sinne der Brücke also von links nach rechts). Tatsächlich besteht aber in zahlreichen Softwareprojekten das Dilemma nicht zu wissen welche Teil der Software weggelassen werden können wenn eine bestimmte Anforderung entfällt.

Dafür ist es erforderlich auch rückwärts über die Brücke schlüssig nachvollziehen zu können welche **anderen** Anforderungen ebenfalls nach den Bits verlangen die man als Entfernungskandidaten ausgemacht hat. Solange diese Vorwärts-Rückwärts Traceability nicht gegeben ist, wird ein typischer Verlauf von Softwareprojekten sich fortwährend wiederholen: nach einiger Zeit wird die Software als unwartbar erklärt und neu entwickelt. Es ist zu vermuten das eine gute Werkzeugkette helfen könnte diese Situation zu vermeiden.

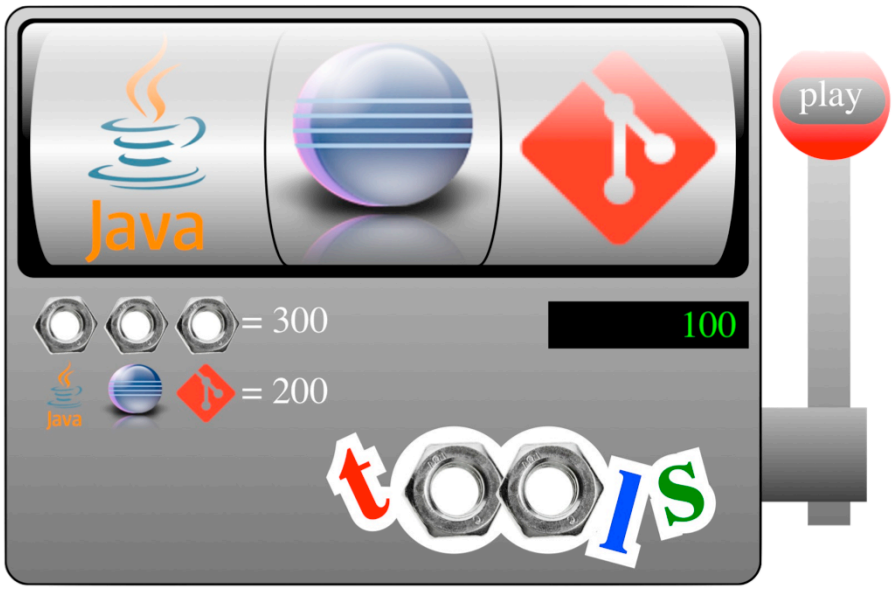
Siehe auch http://en.wikipedia.org/wiki/Traceability

Um die Nachverfolgbarkeit von Software-Architekturdokumenten zu ermöglichen ist es nötig, eindeutige Querverweise zwischen verschiedenen Ergebnissen zu verwenden.

Dazu sollten die Ergebnisse mit eindeutigen Identifiern versehen werden. Ideal ist es wenn dann über ein Internet/Intranet – Hyperlinksystem auf die einzelnen Ergebnisse zugegriffen werden kann.

Oben ist als Beispiel ein Hyperlink vom Sourcecode auf das Requirementsmanagement System gezeigt. (Eclipse - smartGenerator / AKGenerator -> smartRQM)

Werkzeugauswahl

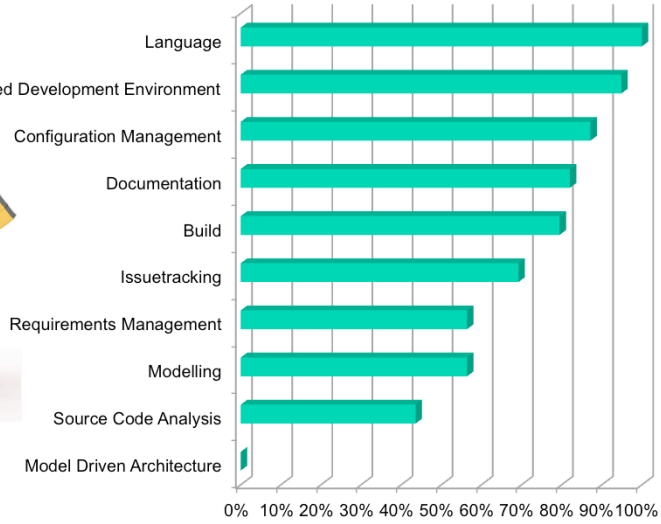


Tatsächlich ist der Markt für Werkzeuge von Oligopolen geprägt.

Die Werkzeugauswahl fällt also leicht, wenn man sich auf die Platzhirsche beschränkt im Sinne von „Nobody ever got fired for buying IBM“.

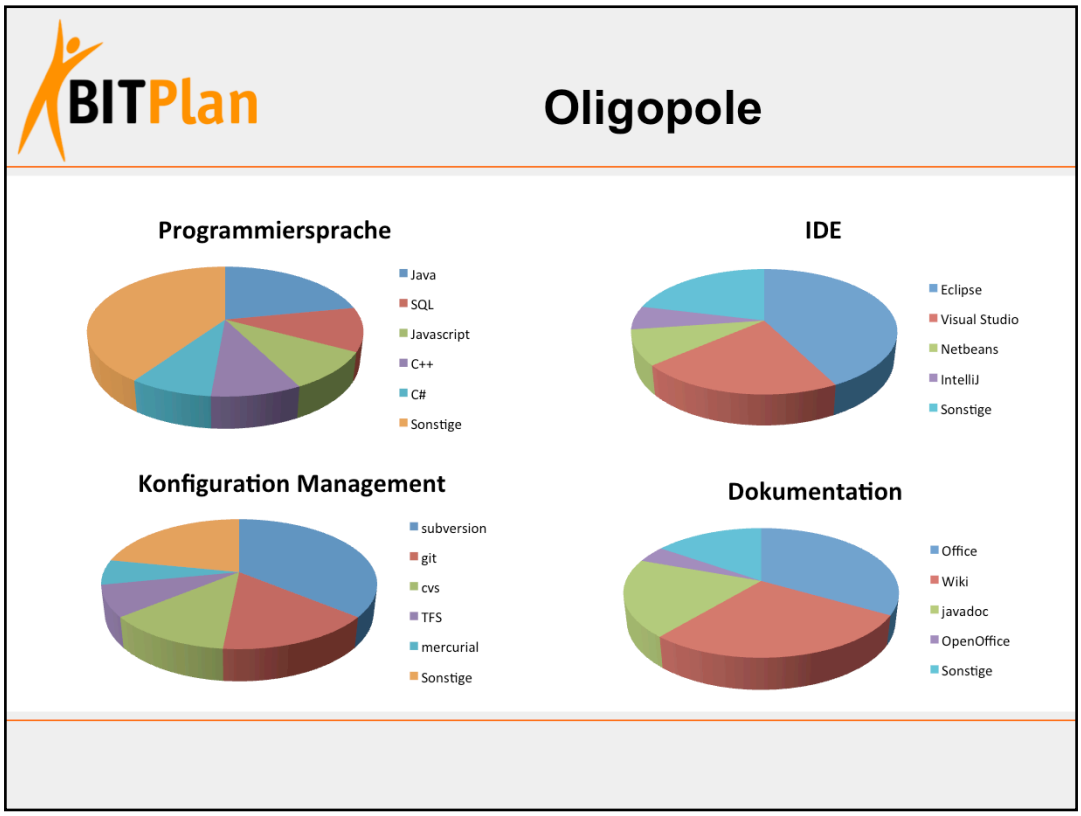
Tatsächlich können sich nur noch ganz große Organisationen leisten eigene Softwareentwicklungsumgebungen zu entwickeln. Das erinnert ein wenig an die Zeit als es klar wurde, dass die Entwicklung und Vermarktung von Betriebssystemen selbst die oben noch so bewunderte IBM scheitern lässt. Mit Eclipse hat IBM bei den IDEs einen entscheidenden Beitrag geleistet. Erstaunlich ist, dass Microsoft mit Office bei den Requirements-Management Tools noch immer führend ist.

Werkzeuge



Werkzeuge nehmen im Softwareentwicklungsprozess eine entscheidende Rolle ein. Manche Werkzeuge wie der Compiler und Editor erscheinen notwendig, andere lediglich mehr oder weniger nützlich. Die obige Statistik zur Nutzung verschiedener Werkzeugarten beruht auf einer Umfrage unter den Teilnehmern von BITPlan CPSA-F Kursteilnehmern. Sie zeigt, dass die Werkzeugketten immer vollständiger werden, auch wenn der Anteil der Organisationen mit erheblichen Lücken in der Werkzeugkette noch ziemlich auffällig ist.

Die Liste der abgefragten Werkzeuge war auf 10 beschränkt insbesondere Testtools wurden nicht abgefragt. Tatsächlich gibt es natürlich erheblich mehr Tools. Insbesondere wenn man jedes Plugin für jeden IDE-Typ und jedes Utiltity hinzuzählt ergibt sich eine äußerst unüberschaubare Anzahl von möglichen Werkzeugen.



In den wichtigsten Werkzeugkategorien gibt es nur wenige Marktbeherrschende Anbieter.

BITPlan

Die „git-Story“

- 2005 CM „Notlösung“ für Linux Kernel Sourcen
- 2008 GitHub: 2.1 million users 3.7 million repositories
- Erfolgsfaktor: komplette(!?) Werkzeugkette

GitHub hat es innerhalb von 4 Jahren (2008-2012) geschafft auf 3.7 Millionen Repositories zu kommen. Zum Vergleich brauchte das nächst größere Portal “SourceForge” für 250.000 Projekte 10 Jahre (2000-2010.

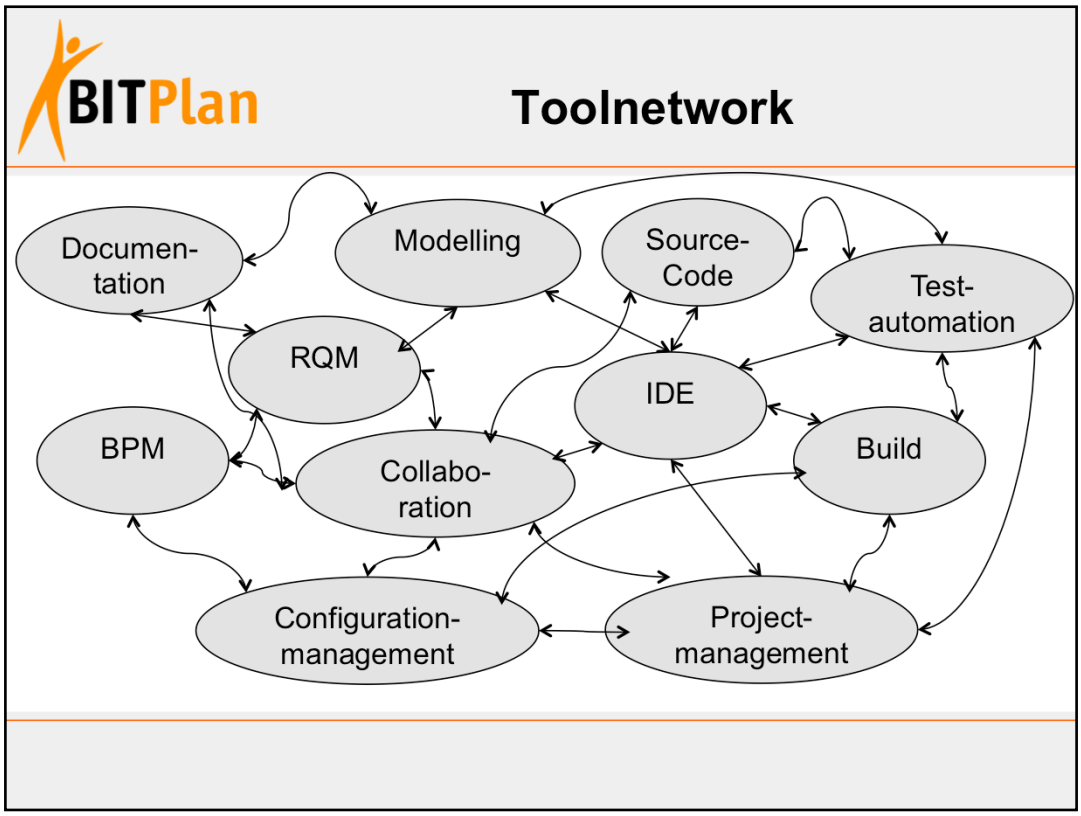
Siehe auch http://en.wikipedia.org/wiki/Git_%28software%29

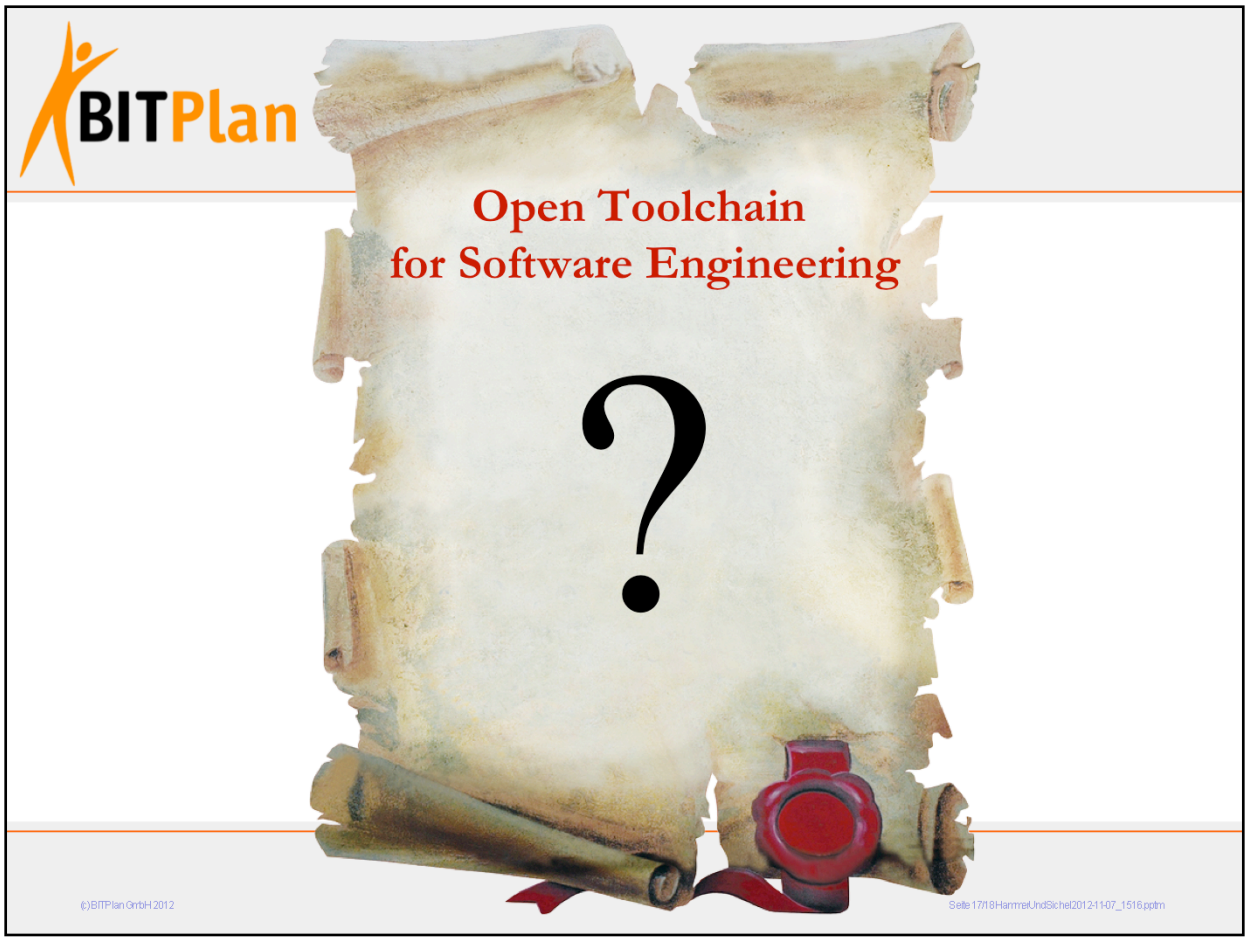
The git „toolchain“

- Configuration Management
- Issue Tracking
- Webhosting
- Social Organization

Der Rest gilt als selbstverständlich

Erstaunlicherweise ist die Sammlung von github angebotener Werkzeuge relativ überschaubar und sogar in gewissem Sinne kleiner als die der Mitbewerber. Dennoch ist der Erfolg von github größer.


BITPlan
Toolnetwork
Documen-
tation
Modelling
Source-
Code
Test-
automation
RQM
IDE
BPM
Collabo-
ration
Build
Configuration-
management
Project-
management



Wie müsste ein Manifest für eine offene Werkzeugkette zum Thema Softwareengineering aussehen?

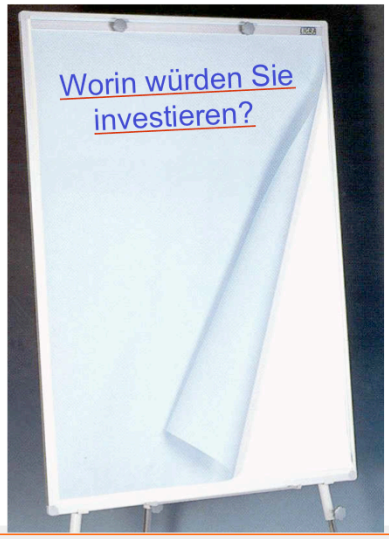
Wichtige Punkte sind aus meiner Sicht:

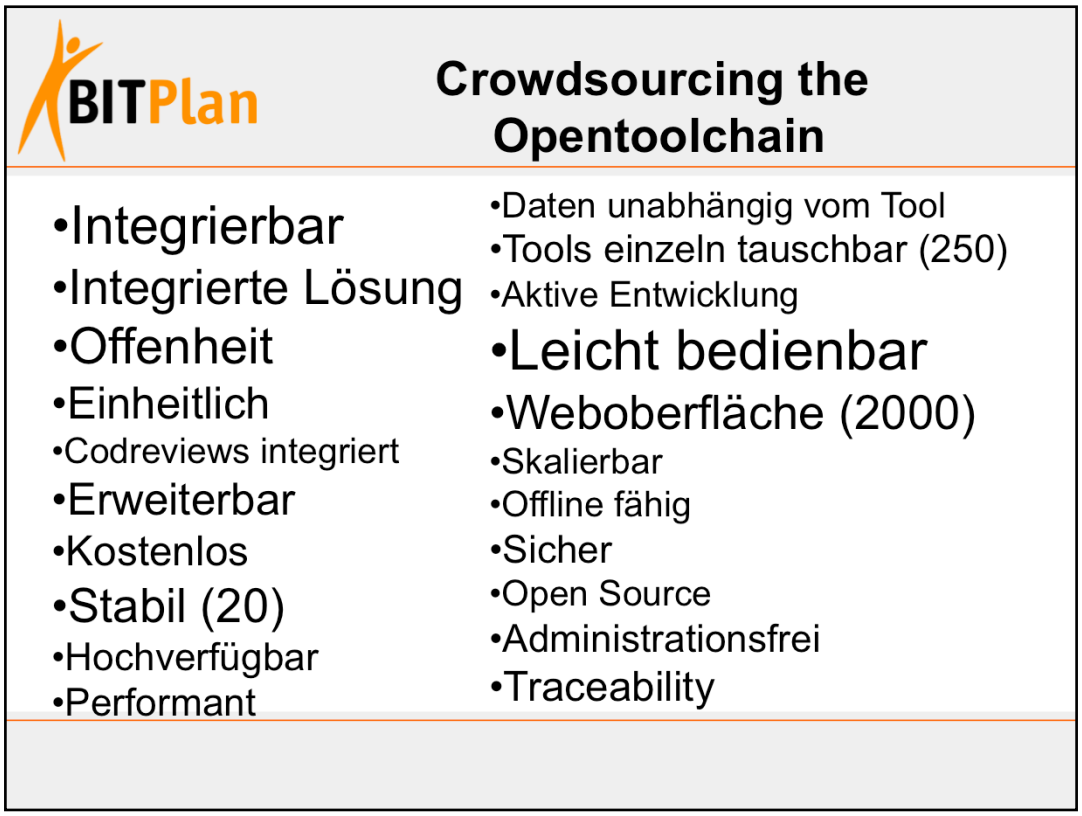
- Werkzeuge müssen verständlich sein, dass bedeutet nicht nur dass sie einfach bedienbar sind sondern dass das zugrunde liegende Konzept keine unnötige Komplexität einführt.
- Werkzeuge müssen die Zusammenarbeit fördern.
- Werkzeuge müssen Ergebnisse liefern die verknüpfbar und damit nachvollziehbar sind. Die Verknüpfungen müssen stabil sein und in zwei Richtungen funktionieren. Dabei gibt es ggf. keine Ursache-Wirkungsrichtung sondern lediglich eine Verknüpfung
- Werkzeuge müssen nützlich sein, d.h. die Produktivität fördern, die Aufwände und Kosten für den Einsatz des Werkzeuges müssen durch den Nutzen bei weitem mit getragen werden.
- Werkzeuge müssen verfügbar und zugänglich sein. D.h. es darf keine technischen oder lizenztechnischen Hindernisse für die Nutzung des Werkzeuges geben. Lokale, Intranet und Internetnutzung müssen gleichberechtigt sein.
- Werkzeuge müssen zusammenwirken – die Verantwortung für das Zusammenwirken liegt dabei bei den Werkzeugherstellern nicht den Werkzeugnutzern.

BITPlan

Crowdsourcing the Opentoolchain

Was für Eigenschaften müsste eine Opentoolchain haben, damit Sie in die zugehörige Organisation investieren würden?





24 Punkt = 1 Stimme

Pro Stimme 2 Punkt Schriftart grösser

Max=

Fragen => Mixed Zone

www.BITPlan.com

Wolfgang.Fahl@BITPlan.com

1-2 Fragen können hier noch direkt beantwortet werden. Ansonsten gibt es das Angebot der “Mixed Zone” zur Vertiefung des Themas.