Demands that Stress Patterns

Wie Anforderungen Unternehmenskulturen auf die Probe stellen

Sven Eisenhauer Tobias Koch

8. Januar 2010

Seminar Systems Thinking, Wintersemester 2009/2010
Hochschule Darmstadt
Referent: Herr Prof. Dr. Andelfinger

Inhaltsverzeichnis

1. Reifegradstufen von Entwicklungsprozessen 2
2. Komplexität dynamischer Systeme 3
3. Rückkopplungseffekt zwischen Erfolg und Ambition 4
4. Was dabei hilft, die Kontrolle zu behalten 6
5. Rolle und Einfluss des Kunden 7
6. Systems Thinking 11
Literatur 14

Zusammenfassung

Im ersten Band – *Systems Thinking* – seiner Reihe von Büchern zum Qualitätsmanagement in der Software-Entwicklung, schildert Gerald M. Weinberg wie typische kulturelle Muster in Software-Firmen die Qualität der Entwicklungsprozesse und des Managements widerspiegeln. Unter Betrachtung der inneren und äußeren Anforderungen an Software entwickelnde Organisationen veranschaulicht

er, wie das Verharren in einem solchen Muster die Entwicklung einer Organisation hemmen oder sogar deren Fortbestehen gefährden kann. Gleichzeitig liefert er eine Fülle von Denkanstößen und Vorschläge für Methoden, die dabei helfen können, besagte Firmenkulturen in positiver Weise zu beeinflußen und zu verändern.

Dieser Artikel fasst die Kerngedanken aus dem dritten Teil – *Demands that Stress Patterns* – des Buches zusammen und erläutert anhand konkreter Beispiele aus dem Buch, warum Software-Entwicklung an sich eine hoch-komplexe Angelegenheit ist, wie man der Komplexität Herr werden kann, und welche zusätzlichen äußeren Anforderungen sich beim Projektgeschäft mit Kunden ergeben.

1 Reifegradstufen von Entwicklungsprozessen

In [WEINBERG92] befaßt sich Gerald M. Weinberg mit den Problemen und Herausforderungen, denen sich Software produzierende Organisationen früher oder später stellen müssen. Seinen Analysen legt er „kulturelle Muster“ zugrunde, die in ihrer Beschreibung in etwa dem Reifegrad von Entwicklungsprozessen nach dem Capability Maturity Modell (CMM) [CMMI06] entsprechen. Für die weiteren Ausführungen in diesem Artikel sind davon die Stufen 1 bis 3 von Bedeutung:

Stufe 1 - initial:
Nach Weinberg, der das entsprechende kulturelle Muster „variabel“ nennt, gibt es auf dieser Stufe „kein Konzept von Management als Entwicklungswerkzeug“. Nach dem CMM sind Prozesse auf dieser Reifegradsstufe zufällig und chaotisch und die beherbergende Organisation trifft keine Maßnahmen, um die Entstehung stabiler Prozesse zu fördern. Der Erfolg von Projekten auf dieser Stufe hängt in erster Linie vom Engagement und den Fähigkeiten der beteiligten Personen ab.

Stufe 2 - geregelt:
Auf dieser Stufe werden Prozesse laut CMM nach gefestigten Richtlinien durchgeführt. Qualität und Performanz werden ansatzweise messbar gemacht. Weinberg nennt das entsprechende kulturelle Muster „routiniert“, weil seiner Beobachtung nach Prozesse in diesem Muster nicht hinterfragt sondern blind angewendet werden. Als charakteristisch für dieses Muster nennt er das Konzept des Supermanagers, dem der gesamte Projekterfolg zugeschrieben wird.

Stufe 3 - definiert:
Weinberg nennt das kulturelle Muster, das dieser Entwicklungsstufe entspricht, „steuernd“, weil die Manager auf dieser Stufe vorausschauend und unter Berücksichtigung der ihnen bekannten, aktuellen Entwicklungen entscheiden. Nach CMM sind die Prozesse auf dieser Stufe „genau [...] verstanden und werden anhand von Standards, Arbeitsverfahren, Werkzeugen und Methoden beschrieben“, mit der Zielsetzung diese in der gesamten Organisation zu verheinheitlichen.

Stufe 4 - quantitiv geregelt:
Laut CCM sind Prozesse auf Stufe 4 definerte Prozesse (Stufe 3), die „mit Hilfe statistischer und anderer quantitativer Methoden kontrolliert werden“. Weinberg nennt das kulturelle Muster, das dieser Reifegradstufe entspricht, „voraussschauend“, was bedeuten soll, dass die Wahl und Konzeption neuer Prozesse und Methoden unter Berücksichtigung vorausgehender Erfahrungen erfolgt.

Stufe 5 - optimierend:
Wie der Name schon sagt, sind Prozesse auf dieser Stufe nach CMM quantitativ geregelte Prozesse, deren Performanz ständig verbessert wird. Weinberg beruft sich bei der Beschreibung dieser Stufe auf Watt S. Humphrey, nach dessen Aussage auf dieser Stufe die Qualität des Managements an sich eine zentrale Rolle spielen soll.

Nach Weinbergs eigenen Beobachtungen befinden sich die meisten Unternehmen im Software-Geschäft auf den Stufen 1 oder 2, wenige auf Stufe 3. Die Stufen 4 und 5 sind dagegen selten und auch nur ansatzweise anzutreffen.

2 Komplexität dynamischer Systeme

Wenn man dynamische Systeme mit Hilfe mathematischer Gleichungssysteme modelliert, dann steigt der benötigte Rechenaufwand zur Simulation eines solchen Systems in Abhängkeit von der Größe oder Komplexität des Systems nicht-linear an. Dieser Zusammenhang wird in Abbildung 1 veranschaulicht: Steigt die Komplexität des Systems, dann nimmt sowohl die Anzahl der Gleichungen, die man benötigt, um das System zu modellieren, als auch die Anzahl der Parameter zu, wodurch der Berechnungsaufwand insgesamt exponentiell ansteigt.

Weinberg nennt diesen Sachverhalt „The Square Law of Computation“. Nach Weinberg handelt es sich dabei um eine sogenannte „natürliche Dynamik“, die sich wie das Gravitationsgesetz jedem menschlichen Einfluss entzieht. Im Gegensatz dazu nennt Weinberg eine Dynamik, die sich aus menschlichen Entscheidungen und Handlungen entwickelt, eine „Interventionsdynamik“.

Zieht man als Beispiele für dynamische Systeme die Spiele Tic-Tac-Toe und Schach heran, dann kann man folgende Feststellungen treffen:

- Bei beiden Spielen handelt es sich um abgeschlossene Systeme mit exakt definierten möglichen Zustandsübergängen.
- Tic-Tac-Toe hat aufgrund des kleinen Spielfelds eine so geringe Komplexität, dass der gesamte Spielbaum innerhalb von Sekundenbruchteilen berechnet werden kann. Sogar Hühner können darauf trainiert werden, fehlerfrei Tic-Tac-Toe zu spielen.

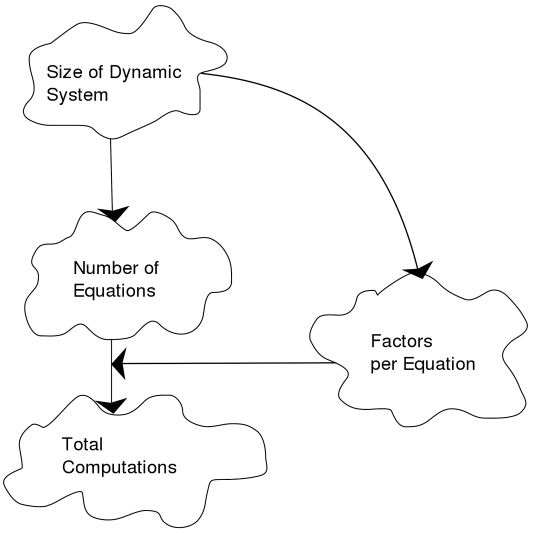

Abbildung 1: Zusammenhang zwischen der Komplexität eines dynamischen Systems und dem benötigten Berechnungsaufwand zur Kontrolle des Systems. Quelle: [WEINBERG92], Seite 131.

- Schach ist aufgrund des größeren Feldes und der durchschnittlich mehr als 30 Entscheidungsmöglichkeiten pro Zug derart komplex, dass selbst ein Supercomputer den vollständigen Spielbaum nicht in endlicher Zeit berechnen kann.

Wenn man Management als Spiel betrachtet, bei dem es darum geht, die Kontrolle nicht zu verlieren, dann ist sofort klar, dass dieses Spiel noch weitaus komplexer ist als Schach, weil es sich

a) um kein abgeschlossenes System handelt,

b) die Zustandsübergänge nicht genau definierbar sind und

c) die Anzahl der Wahlmöglichkeiten pro Zug unermesslich groß ist.

Da es Menschen gibt, die trotz der Komplexität des Spiels sehr erfolgreich Schach spielen, äußert sich Weinberg optimistisch, dass auch Management als „Game of Control“ beherrschbar ist, wenn man sich der richtigen Denkweisen und Arbeitsmethoden bedient.

3 Rückkopplungseffekt zwischen Erfolg und Ambition

Da unsere intellektuellen Kapazitäten von Natur aus begrenzt sind, stößt ab einer gewissen Komplexität der Problemstellung jeder Mensch an seine Grenzen, insofern dass sich Anforderungen und mögliche Lösungsansätze ohne besondere Methodik nicht mehr direkt erschließen.

3. Rückkopplungseffekt zwischen Erfolg und Ambition

Da der Mensch aber von Natur aus ehrgeizig ist und dazu tendiert, sich immer höhere Ziele zu stecken, ist es nur eine Frage der Zeit, bis das zu lösende Problem einen Umfang annimmt, der ohne besondere Maßnahmen nicht mehr zu bewältigen ist. Dieser Zusammenhang ist in Abbildung 2 dargestellt: Der initiale Erfolg wird zunächst von der intellektuellen Kapazität des Leistungserbringers bestimmt. Mit zunehmender Erfolgsrate steigen aber auch die Ambitionen, sich komplizierteren Problemen zu nähern, die Komplexität der Lösung steigt an und wirkt sich schließlich negativ auf die Erfolgsrate aus. Im Zusammenhang mit Software-Entwicklung schlägt Weinberg vor, dass dieser Rückkopplungseffekt mit den Methoden des Software-Engineerings beherrschbar gemacht werden kann.

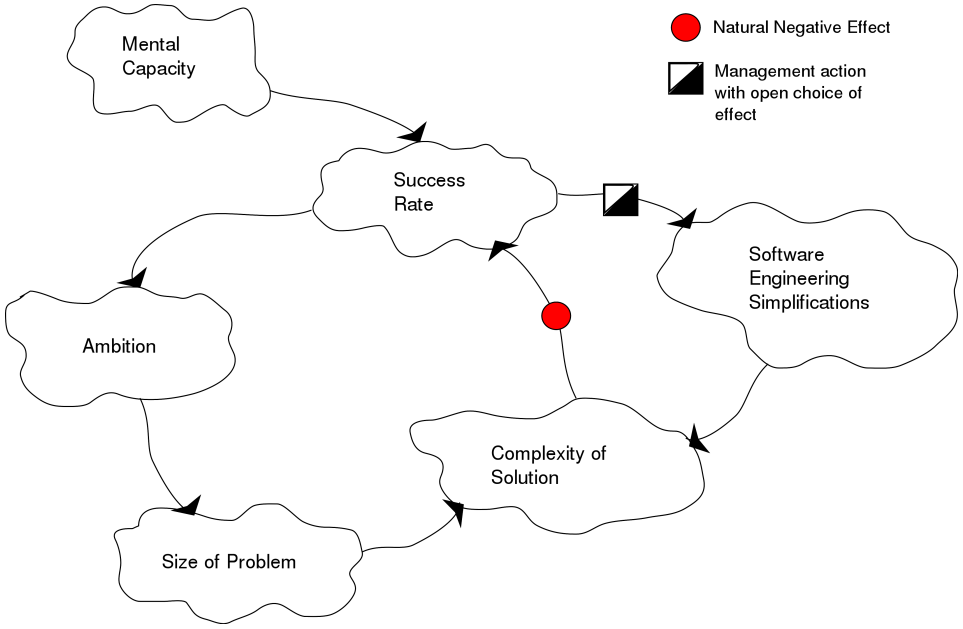

Abbildung 2: Software-Engineering zur Beherrschung des Rückkopplungseffektes zwischen Erfolg und Ambition.
Quelle: [WEINBERG92], Seite 136.

Die Tatsache, dass unsere intellektuelle Kapazität begrenzt ist, wir dazu tendieren, uns immer schwierigere Aufgaben zu suchen, Problemstellungen aber aufgrund des „Square Law of Computation“ schnell Dimensionen annehmen, die diese Kapazität übersteigen, nennt Weinberg „Size/Complexity-Dynamic“.

Diese natürliche Dynamik tritt noch in anderen Facetten zum Vorschein, z.B. in Form der Fault/Location-Dynamic: Mit zunehmendem Unfang eines Software-Quelltextes nimmt sowohl die Anzahl der Fehler als auch die Anzahl der Zeilen, wo man nach den Fehlern suchen muss, zu. Damit ist die Fehlerbehebung in Software-Projekten ein nicht-linearer wachsender Arbeitsaufwand.

Eine weitere Variante ist die People/Interaction-Dynamic, auf die später noch einmal in einem anderen Zusammenhang näher eingegangen wird. Diese Dynamik beschreibt die Tatsache, dass bei zunehmender Anzahl der an einem Projekt beteilgten Personen, die Zahl der Interaktionen zwischen diesen Personen ebenfalls nicht-linear zunimmt.

4 Was dabei hilft, die Kontrolle zu behalten

Zwar können wir durch Training unsere mentalen Fähigkeiten verbessern, allerdings nur sehr marginal. Ohne besondere Methodik können selbst die klügsten Menschen Probleme ab einer gewissen Komplexität nicht mehr lösen, und dass dieser Fall eher früher als später eintritt, dafür sorgen das Square Law of Computation und die Size/Complexity-Dynamic.

Wir benötigen deswegen Methoden und Werkzeuge, die uns dabei helfen, unsere Aufgaben in kleinere Teile herunterzubrechen z.B. durch objektorientierte Modellierung, Verwendung von Mindmapping-Tools und anderer Hilfsmittel. Für die Lösung spezifischer Aufgaben kann die Verwendung einer bestimmten Programmiersprache hilfreich sein. So bietet es sich z.B. an, für das Datamining in Texten Perl zu verwenden, wohingegen die hardwarenahe Programmierung in einem embedded Projekt die Verwendung von C oder C++ nahelegt.

Abbildung 3 zeigt, wie unterschiedlich verschiedene Arbeits- und Entwicklungsmethoden bei zunehmender Problemgröße skalieren könnten. Die Graphik zeigt aber auch, dass jede Methode irgendwann an ihre Grenzen stößt.

Bei der Auswahl der Arbeitswerkzeuge sollten deswegen folgende Faktoren berücksichtigt werden:

- die Art der Aufgabenstellung (z.B. Datamining- oder Datenbankprojekt),
- der Projektumfang,
- und das geschätzte Risiko.

Weinberg erwähnt, dass Manager auf Stufe 3 (definiert) gerne bereit sind, Werkzeuge und Methoden nach unterschiedlichen Kriterien auszuwählen, wohingegen Manager auf den Stufen darunter lieber alle Projekte nach dem selben Schema abwickeln, aus Angst man könne ihnen sonst später eine Fehlentscheidung anlasten. Dass sich gerade dieses Verhalten in vielen Fällen nachteilig auf den Projekterfolg auswirken kann, ist für sie dabei ohne Belang. Sie sehen sich firmenpolitischen Zwängen ausgesetzt, denen sie sich ergeben, z.B. aus Sorge ihre Position oder gar ihre Arbeitsstelle zu verlieren.

Positive Grundhaltung

Für einen Außenstehenden mag dieses Verhalten dann absurd erscheinen, weil sich ihm die sozio-politischen Zusammenhänge nicht direkt erschließen. Aus der Sicht des Betrof-

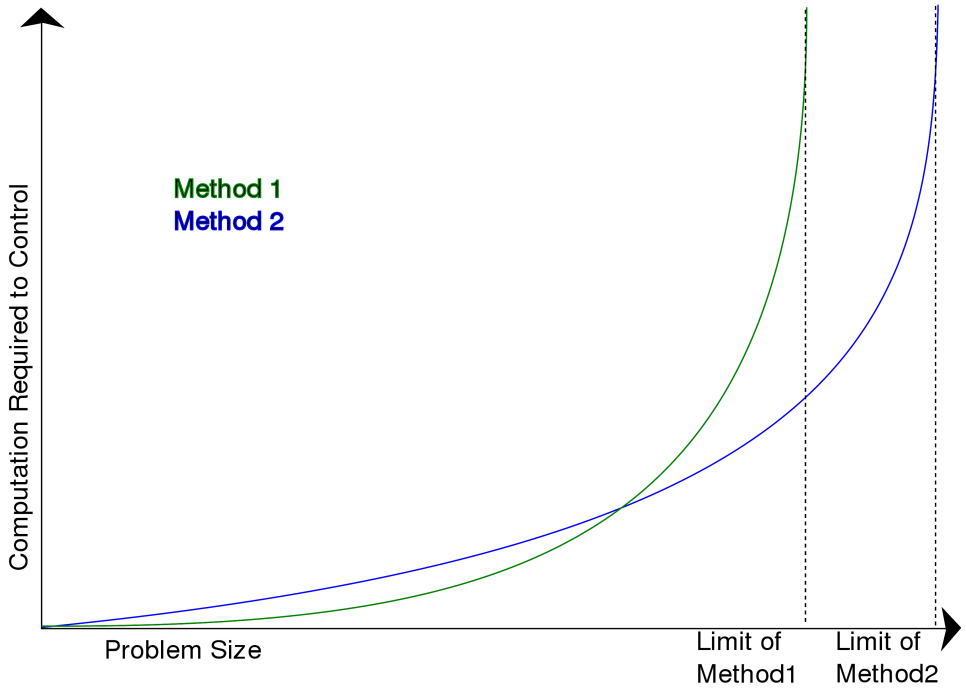

Abbildung 3: Skalierbarkeit und Limitierung verschiedener Arbeitsmethoden bei zunehmendem Problemumfang.
Quelle: [WEINBERG92], Seite 148.

fenen ist es aber durchaus plausibel. Weinberg plädiert deswegen allgemein für einen verständnisvollen Umgang mit Personen in einem solchen Umfeld. Denn prinzipiell sei davon auszugehen, dass jeder Mensch zunächst einmal gute Absichten hat und positiv auf sein Umfeld einwirken möchte. Da jeder die Welt ein bisschen anders sieht, und deshalb zu einer Einschätzung der Lage kommen kann, die von der eigenen divergiert, postuliert Weinberg:

> *Egal wie es erscheint, jeder versucht hilfreich zu sein.*

Weiterhin solle man nicht versuchen, den Mitarbeitern alte Gewohnheiten zu verbieten, sondern ihnen neue, effektivere anzutrainieren, bzw. sie zu ermutigen sich neue Denkweisen zu erschließen, um Konditionierungen auf alte Verhaltensmuster zu überwinden.

5 Rolle und Einfluss des Kunden

Organisationen sind keine abgeschlossenen Gebilde sondern erhalten ihre Daseinsberechtigung oftmals erst durch ihre Interaktion mit der Außenwelt. Einer der wichtigesten Außenkontakte von Organisationen sind ihre Kunden, denn diese generieren den Umsatz,

der für jedes Unternehmen wichtig ist. Mit zunehmendem Erfolg eines Unternehmens steigt die Anzahl der Kunden.

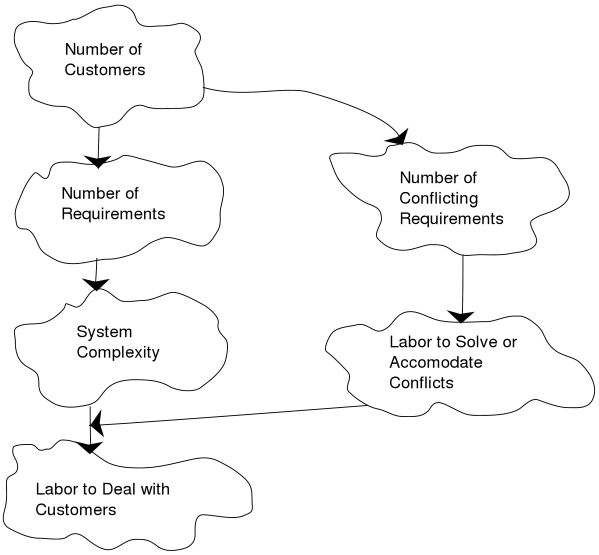

Abbildung 4: Bei zunehmender Anzahl der Kunden, nimmt der nötige Arbeitsaufwand zur Berücksichtigung aller Kundenanforderungen nicht-linear zu. Quelle: [WEINBERG92], Seite 161.

Dieser Umstand kann eine Organisation aber auch sehr schnell an die Grenzen ihrer Leistungsfähigkeit bringen. Mit der steigenden Zahl von Kunden steigen auch deren Anforderungen, was wiederum zu komplexeren Systemen führt. Diese steigende Anzahl von Kunden führt außerdem noch zu Konflikten unter den Anforderungen. Zur Lösung der Konflikte ist ein zusätzlicher Arbeitsaufwand nötig. Dieser und die gestiegene Systemkomplexität steigern den Gesamtaufwand zur Bearbeitung der Kundenbeziehungen.

Der vergrößerte Arbeitsaufwand durch komplexere Systeme und mehr Kundenanforderungen können es erfordern, dass sich eine Organisation auf ein kulturelles Muster höherer Ebene umstellt. Dies liegt darin begründet, dass mit einem kulturellen Muster höherer Stufe mehr Arbeit bewältigen lässt. Ohne diese Anpassung des Musters könnte eine Organisation irgendwann den Arbeitsaufwand nicht mehr bewältigen und würde scheitern.

Ebenso stellt die Verringerung der Anzahl der bestehenden Kunden keine Option dar. Stattdessen versuchen Organisationen höherer Ebene den Umgang mit Kunden besser zu organisieren. Weinberg beschreibt dazu die Etablierung von Stellvertretern (Surrogates). Diese sollen zwischen den Kunden und der Entwicklungsorganisation platziert werden, wodurch sie in der Lage sind, den Informationsfluss zwischen diesen beiden zu filtern und zu regulieren. Dadurch verringert sich aus Sicht der Entwicklungsorganisation die Anzahl der effektiven Kunden, mit der sie interagiert.

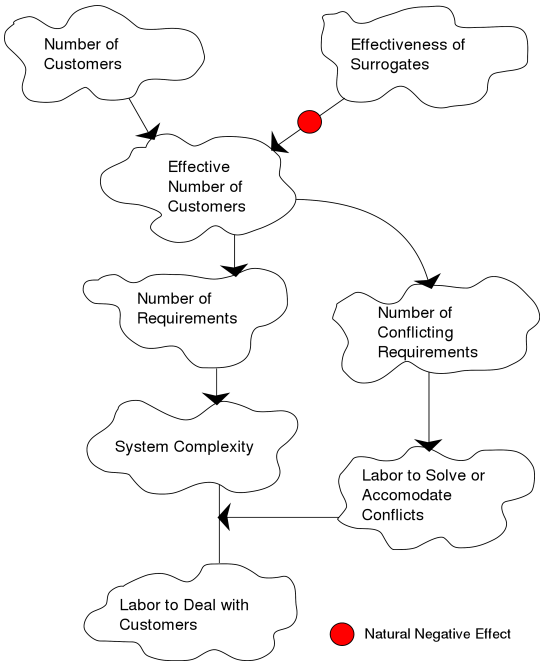

Abbildung 5: Wie der Einsatz eines Surrogates die effektive Anzahl der Kunden und den mit den Kunden verknüpften Arbeitsaufwand reduzieren kann. Quelle: [WEINBERG92], Seite 167.

Somit bilden diese Stellvertreter einen natürlichen negativen Effekt auf die Anzahl der effektiven Kunden. Dabei spielt allerdings die Effektivität dieser Stellvertreter eine entscheidende Rolle, da sie sehr großen Einfluss auf die Entwicklungsorganisation besitzen.

Diese Filterung durch Stellvertreter erlangt ihre Bedeutung durch die oftmals unterschätzte Auswirkung von Arbeitsbrechungen. Weinberg bedient sich zum Beleg einiger Zahlen von DeMarco und Lister. Danach beträgt der Arbeitsausfall eines Entwicklers für ein Telefonat von 5 Minuten insgesamt 20 Minuten. Zu der reinen Gesprächszeit muss noch eine Wiedereinarbeitungszeit von 15 Minuten addiert werden. Kann nun durch den effektiven Einsatz von Stellvertretern die Anzahl der Arbeitsunterbrechungen eines Entwicklers reduziert werden, so führt dies zu einer Steigerung seiner Produktivität, womit er mehr Arbeit bewältigen kann.

Besonders deutlich wird die Wichtigkeit effektiver Stellvertreter bei der Betrachtung von Besprechungen. Dort multipliziert sich die Summe aus Unterbrechungs- und Wiedereinarbeitungszeit noch mit der Anzahl der Teilnehmer. Diese Zeit ist verschwendete Arbeitszeit. Mit einer steigenden Anzahl von Kunden steigt die Anzahl von Besprechungen und die Anzahl der Unterbrechungen dieser. Somit steigt die verschwendete Arbeitszeit nicht linear an.

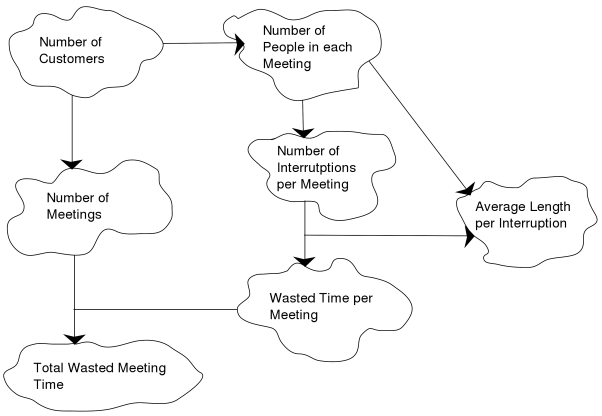

Abbildung 6: Zusammenhang zwischen Anzahl der Kunden und verlorener Zeit durch Unterbrechungen in Meetings.
Quelle: [WEINBERG92], Seite 172.

Mehr Kunden resultieren auch noch in einer anderen Form in Mehrarbeit für eine Softwareentwicklungsorganisation. Je mehr Kunden diese besitzt, desto mehr Kunden setzen ihre Software in unterschiedlichen Konfigurationen ein. Dies verlängert die Reparaturdauer und erhöht die Anzahl der Fehler, die nicht durch Tests gefunden wurden. Dies führt zu mehr Fehlern, die repariert werden müssen.

Dieser Zuwachs an zu reparierenden Fehlern und die längere Reparaturdauer erhöhen den Gesamtarbeitsaufwand zur Fehlerbeseitigung. Organisationen ab Stufe 3 aufwärts, die sich selbst die nötigen Werkzeuge zur Bewältigung ihrer Aufgaben suchen, können hier ihren Aufwand durch den Einsatz von geeigneten Werkzeugen reduzieren. Zum einen kann durch Konfigurationsverwaltung die Zeit pro Reparatur verringert werden, zum anderen reduzieren automatisierte Tests nicht gefundene Fehler.

Ähnlich wächst die Anzahl von unterschiedlichen Versionen einer Software, die eine Organisation unterstützen muss, mit der Anzahl der Kunden. Auch dies resultiert in mehr Arbeitsaufwand für die Organisation bei der Unterstützung und Wartung der vielen Versionen. Stufe 3 Organisationen bedienen sich hier Werkzeuge zur Verwaltung von Versionen. Interessant ist hierbei die Tatsache, dass sich bei vielen Organisationen ein Release-Zyklus von sechs Monaten, also zwei Releases pro Jahr, etabliert hat. Dies resultiert aus dem Versuch des Managements durch längere Release-Zyklen den Aufwand dafür zu drosseln. Andererseits steigt mit der Anzahl der Kunden der Bedarf an Reparaturen, was die längeren Releaseabsichten des Managments ausbalanciert.

All diese Beobachtungen stellen unterschiedliche Varianten der Size/Complexity-Dynamic dar. Ebenso zeigen sie, wie komplex die Wirkungszusammenhänge in einer Organisation sind. Selbst bei einer Beschränkung der Betrachtung auf die Interna einer Organisation, lassen sich die Vorgänge nicht als einfache Ursache-Wirkung-Zusammenhang beschreiben. Eine Ursache wirkt sich an vielen Stellen aus, ein Wirkungssymptom kann

von verschiedenen Ursachen oder deren Kombination hervorgerufen werden.

6 Systems Thinking

An dieser Stelle entsteht der Bedarf für ein besseres Mittel zur Analyse und Darstellung der Zusammenhänge. Weinberg bedient sich hierfür des Systemdenkens, engl. Systems Thinking. Damit lassen sich komplexe Wirkungszusammenhänge analysieren und mittels entsprechender Diagramme veranschaulichen. Desweiteren ermöglichen die so gennanten System Dynamics die Simulation von Wirkungszusammenhängen auf Basis verschiedener Ausgangswerte. Mittels geeigneter Software, wie in unserem Falle Vensim, lassen sich Diagramme erstellen, verschiedene Simualtionen durchführen und die Ergebnisse vergleichen. Dafür stehen Graphen oder auch tabellarische Daten zur Verfügung. Die Herausforderung hierbei besteht in der mathematischen Abbildung der Realität. D.h. die Qualität der Simulation liegt in der möglichst genauen mathematischen Formulierung der Auswirkungen eines Faktors auf die anderen im System.

Im folgenden möchten wir dies an dem von Weinberg vorgestellten Systemdiagramm zu Softwareentwicklungsmethoden ausführen. Wie bereits zuvor dargestellt geht es hierbei um die Erfolgsquote einer Softwareorganisation und welche Rolle dabei die verschiedenen Faktoren wie die begrenzte Kapazität des menschlichen Gehirns, die Ambitionen, die Größe des Problems, die Komplexität der Lösung und die Effektivität von Vereinfachungen durch Softwaretechnik spielen.

Das entwickelte Vensim Modell (siehe Abbildung 7 basiert auf diesen Variablen, die wie folgt pro Simulationsschritt berechnet werden: Die Kapazität des menschlichen Gehirns ist naturgegeben begrenzt und somit im Modell eine Konstante. Die Erfolgsquote hängt davon ab und wird allerdings negativ von der Lösungskomplexität beeinflusst. Diesen Umstand berücksichtigt die Formel

$$\frac{\log\left(\text{Mental Capacity}\right)}{\log\left(\text{Complexity of Solution}\right)}$$

Die Verwendung einer bi-logarithmischen Skala dient hier in erster Linie zur besseren Darstellung beim Plottten. Ansonsten wären die Unterschiede bei unterschiedlichen Parametereinstellungen kaum sichtbar gewesen.

Die Ambitionen einer Organisation wachsen mit der Erfolgsquote. Je ambitionierter eine Organisation ist, desto größer sind die Probleme, an die sie sich wagt. Die Komplexität der Lösung eines solchen Problems errechnen wir nach der Formel

$$\frac{\left(\text{size of problem}\right)^2}{\log\left(\text{Software Engineering Simplifications} + 1\right)}$$

Die Software Engineering Simplifications stellen hier auch einen konstanten Wert dar, der pro Simulationslauf verändert wird, um die Auswirkungen vergleichen zu können.

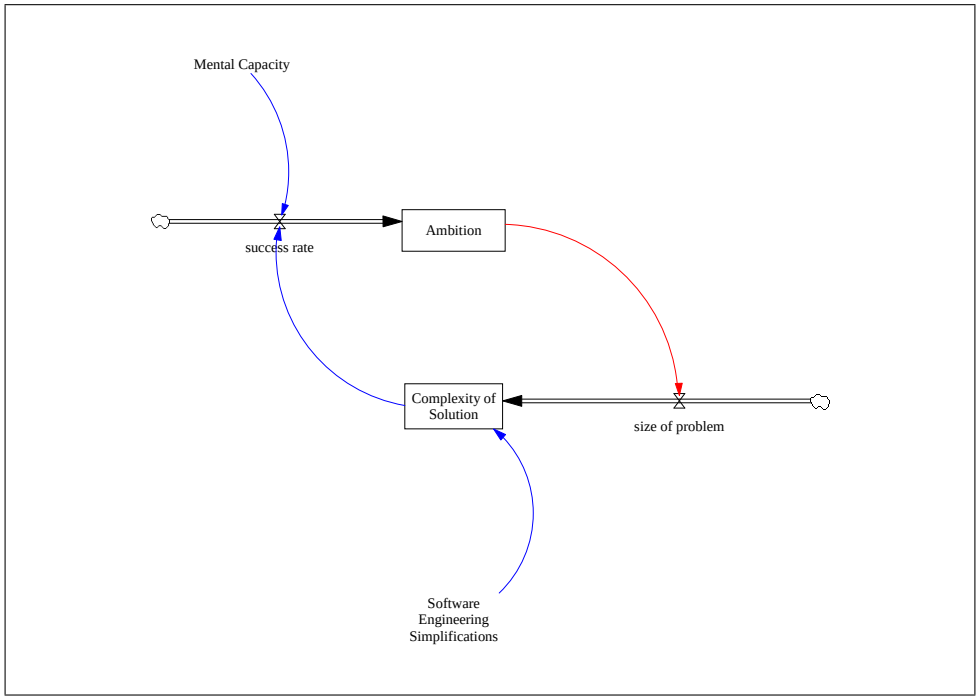

Abbildung 7: Einfaches Vensim-Modell zur Veranschaulichung der Rückkopplung zwischen Ambition, Lösungskomplexität und Projekterfolg.

Die Verwendung des Logarithmus steht an dieser Stelle für die limitierte Skalierbarkeit jeglicher Methoden des Software-Engineerings.

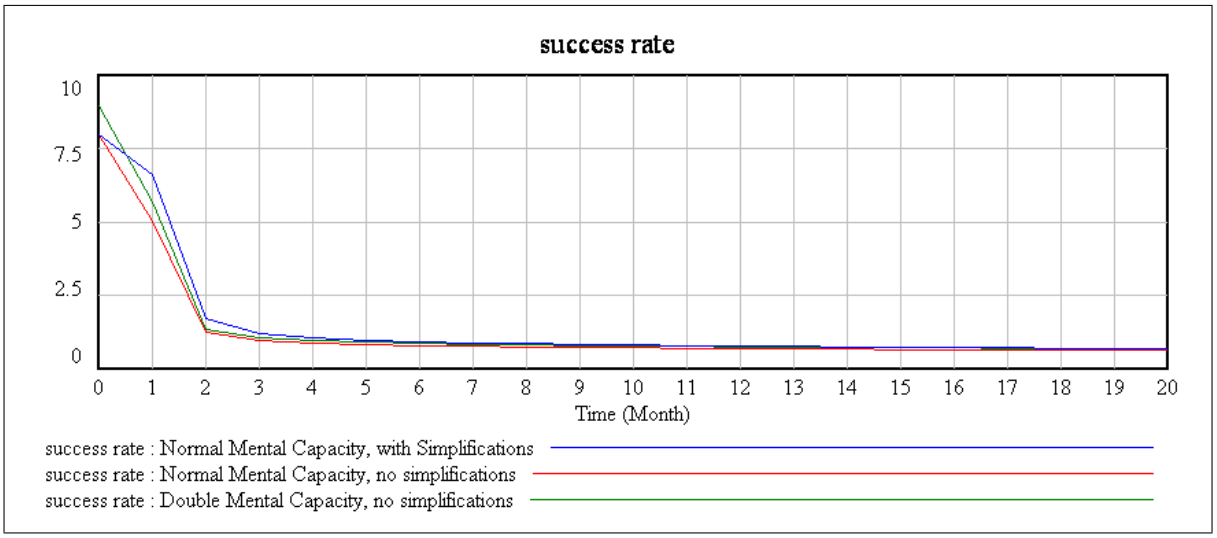

Abbildung 8: Simulation der Erfolgsrate mit verschiedenen Parametern.

Wir betrachten drei Simulationsdurchläufe. Der erste stellt die Referenz mit Standardwerten dar. Beim Zweiten nehmen wir einen höheren Wert für Software Engineering Simplifications an. Der dritte Durchlauf dient zur theoretischen Veranschaulichung der Auswirkungen, wenn es möglich wäre, die Kapazität des menschlichen Gehirns zu verdoppeln. Hierfür greifen wir einen Gedanken von Weinberg auf, der aussagt, dass eine Steigerung der menschlichen Geistesleistung kaum Auswirkungen auf die Erfolgsquote hätte.

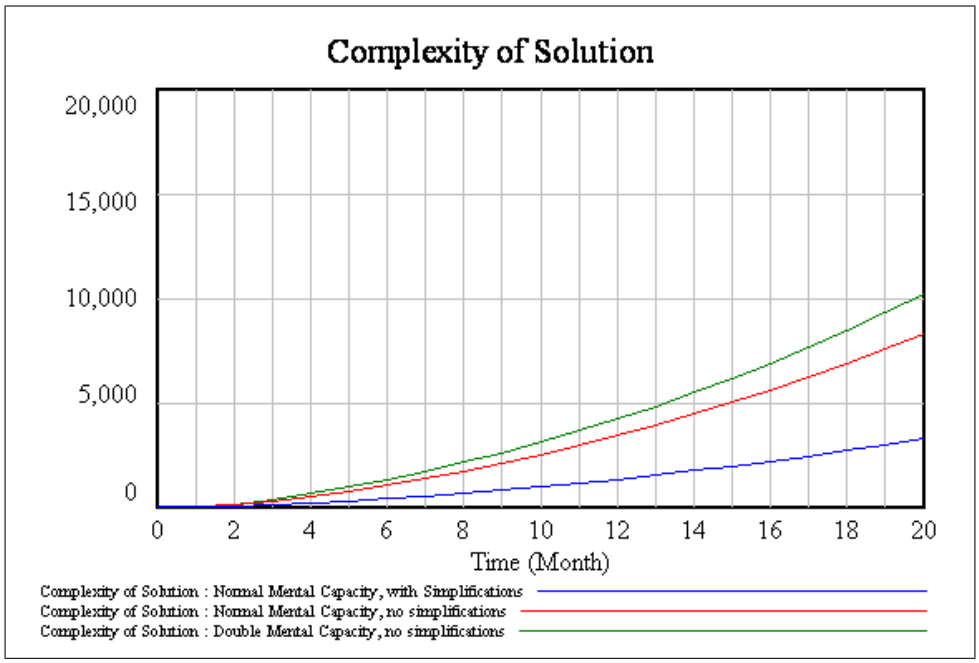

Abbildung 9: Simulation der Lösungskomplexität mit verschiedenden Parametern.

Die Graphen zur Erfolgsquote (siehe Abbildung 8) belegen diese Aussage nach unserer Simulation. Selbst unter Annahme einer verdoppelten Geistesleistung fällt die Kurve nach etwas höherem Beginn sehr schnell ab und nähert sich der Kurve der normalen Geistesleistung. Einzig der Graph in dessen Berechnung die Software Engineering Simplifications verstärkt eingingen fällt zu Beginn deutlich flacher. Er liegt sogar sehr schnell über dem Graphen der doppelten Geistesleistung.

Somit haben wir ein Indiz für die Annahme, dass allein Suche nach immer klügeren Entwicklern eine Organisation nicht voran bringen kann. Selbst, wenn es immer klügere Entwickler gäbe. Einzig die Softwaretechnik kann hier helfen. Dies allerdings auch nur begrenzt, da dieser Graph ebenfalls gegen einen konstanten Wert konvergiert. Diese Beobachtung belegt die Aussage, dass auch die Software keine unendlich großen Probleme lösbar macht. Je besser und effektiver die Softwaretechnik ist, desto weiter kann sie den Abfall der Erfolgsquote nach hinten verschieben. Allerdings wird es immer zu

einem Abfallen der Erfolgsquote kommen. Aufgrund unserer Berechnungsformeln für die Variablen ist der Graph zur Erfolgsquote erst bei genauerem hinsehen aussagekräftig.

Im Gegensatz dazu liefern die Graphen zur Komplexität (siehe Abbildung 9) des Problems ein eindeutiges Bild. Alle drei Graphen steigen stetig, wobei der Graph mit effektiveren Software Engineering Simplifications deutlich unterhalb, der anderen beiden liegt. Dies belegt die Vermutung, dass durch den Einsatz von Methoden der Softwaretechnik, die Komplexität eines Problems deutlich verringert werden kann. Demzufolge ließen sich komplexere Probleme bewältigen, die selbst mit doppelter Gehirnkapazität nicht lösbar wären.

Literatur

[WEINBERG92] Gerald M. Weinberg. *Quality Software Management: Volume 1, Systems Thinking.* Dorset House Publishing, 1992.

[CMMI06] CMMI Product Team. *CMMI for Development, Version 1.2.* Carnegie Mellon Software Engineering Institute, August 2006.
URL: http://www.sei.cmu.edu/reports/06tr008.pdf