Arne-Christian Sigge, Thorsten Spitta

Die Erfassung von Personalaufwand im IT-Bereich

Ein wichtiger Teil der Aufwände eines IT-Bereiches sind Personalaufwände, die man nur dann sinnvoll als Informationsquelle verwenden kann, wenn man sie als Kostenträger-Einzelkosten erfasst. Erfassungssysteme hierfür sind in Großunternehmen verbreitet, jedoch sind die Daten oft unzuverlässig und die Bezugsgröße betriebsspezifisch. Der Beitrag liefert ein Konzept für die manuelle Erfassung von Personalaufwänden und Ereignissen. Er wertet langjährige Erfahrungen mit einem solchen System aus. Die Datengrundlage umfasst die Kontierungen von 170 Personenjahren, die innerhalb von 6½ Jahren von 40 Entwicklern an 5 Standorten erbracht wurden. Der Beitrag fokussiert vor allem auf den Aspekt der Korrektheit der erfassten Daten und den Erfassungsaufwand. Er erläutert Einsatzbedingungen, die systematische Datenverfälschungen durch die Entwickler unwahrscheinlich machen. Es werden beispielhafte Auswertungen für das Projekt- und das Anwendungssystem-Management gezeigt.

Inhaltsübersicht

Inhaltsverzeichnis

1 Einleitung

Viele Methoden verschiedener Disziplinen basieren auf der Kenntnis von Aufwandsdaten: Aufwandsschätzungen, Kostenzuordnungen, Projektmanagement. *The Mythical Man Month* [Brooks 1975] and *Software Engineering Economics* [Boehm 1981] sind noch heute aktuelle Beispiele für grundlegende Bücher, die sich intensiv mit dem Anteil und der Verteilung menschlicher Leistung bei der Softwareentwicklung und Wartung befassen. Auch eine Leistungsverrechnung lässt sich kaum sinnvoll ohne Aufwandsdaten durchführen. Umso verwunderlicher ist es, dass es fast keine publizierten Konzepte gibt, wie man die Aufwände erhebt.

Das hier beschriebene Konzept wurde seit 1987 auf empirischer Basis in einem mittelständischen Konzern mit fünf Standorten entwickelt und implementiert [Spitta 1997]. Beteiligt waren an fünf Standorten insgesamt 40 Entwickler. Hierdurch entstanden 30.000 bis 50.000 Kontierungen pro Jahr, so dass von einer breiten Erfahrungsgrundlage gesprochen werden kann. Die hier gezeigten Daten stammen aus den ersten 6½ Jahren (1987 – 1992). Bis zu die-

sem Zeitpunkt waren 271.077 Stunden Aufwand erfasst, verteilt auf 22 größere und 248 kleinere Projekte sowie 461 Wartungsaktivitäten. Weiterhin waren 106 stornierte Anforderungen ohne erbrachten Aufwand erfasst (s. auch Abbildung 7 am Schluss des Beitrages). Einschließlich der darüber hinaus geleisteten Serviceleistungen enthält die Datenbasis Informationen über 169,4 Personenjahre aus Software-Projekten und -Wartung. Das System ist noch heute (03/2000) in Betrieb.

Aktuelle empirische Untersuchungen zum IT-Controlling [Spitta 1998] zeigen, dass immer noch viel zu wenige Unternehmen solche Daten erfassen. Argumente gegen eine Aufwandserfassung sind die mangelnde Validität und der Erfassungsaufwand. Im Einzelnen:

- Die Grenzen zwischen Entwicklung, Wartung und Service sind nicht scharf genug. Es kommt zu falschen Zuordnungen von Daten. Das ist das *Klassifikationsproblem*.
- Die Daten werden durch die Entwickler selbst erfasst. Dadurch fließt auch menschliches Verhalten in die Daten ein, wodurch diese verfälscht werden können. Die Mitarbeiter müssen motiviert werden, *korrekte* Daten zu erfassen. Wir nennen dies das *Motivationsproblem*.
- Der Aufwand für die Datenerfassung ist hoch. Er beansprucht einen beachtlichen Teil der Arbeitszeit des Entwicklers. Dies ist das *Aufwandsproblem*.

Unser Bericht befasst sich zuerst mit den generellen Anforderungen an die zu erfassenden Daten (Abschnitt 2). Danach werden die Daten im Detail und die Einsatzbedingungen eines solchen Systems mit seinen sozialen Implikationen diskutiert (Abschnitte 3 und 4). Diese werden bei einem rein technik-orientierten Design nur ungenügend beachtet. Der letzte Abschnitt zeigt den mehrfachen Nutzen eines solchen Systems anhand von Beispielen aus Originaldaten. Sie zeigen, dass Aufwandserfassung nicht nur hilfreich für eine Kostenverrechnung und langfristiges IT-Controlling ist, sondern vor allem auch für das Projektmanagement.

2 Ziele und Anforderungen

Ein System zur Aufwandserfassung muss für ein mehrdimensionales Zielsystem ausgelegt sein, um den Aufwand der Datenerfassung zu rechtfertigen. Es muss das Informationsmanagement unterstützen, das sowohl technische als auch ökonomische Elemente enthält. Bei den Zielen handelt es sich im Detail um Übersichten für die Unternehmensführung über:

- Investitionen in Software,
- Struktur, Kosten und Trends von Wartungsaufwand [Boehm & Papaccio 1988]
- Projektmanagement, insbesondere das Controlling externer Kapazitäten [Page-Jones 1991]
- Kostenzuordnung von Benutzeranforderungen
- Soll/Ist-Abweichungen und andere Fragen des IT-Controlling.

IT-Controlling bezieht sich nicht nur auf ein Geschäftsjahr. Die Daten müssen über deutlich längere Zeiträume hinweg betrachtet werden können. Daher muss die Klassifikation der Aufwandsdaten über einen längeren Zeitraum hinweg stabil bleiben. Das System muss *Ereignisse* und *Zeiten* (Aufwände) aller Systeme für alle Leistungsklassen erfassen können. Die Anforderungen müssen vier Ziele unterstützen.

1. **Projektmanagement**
    - Status jedes Projektes zu jedem Zeitpunkt
    - Aufwand jedes Projektes für jede Periode

- Unterbrochene Projekte je Periode (kein Aufwand)
- Fehlerarten je System während der Einführungsphase, etc.

2. Wartungs- und Anwendungsmanagement

- Anzahl und Verteilung von Benutzeranforderungen
- Aufwandsverteilung abgeschlossener Wartungsaktivitäten
- Aufwand für den Betrieb von Standardsoftware, etc.

3. Kostenverrechnung

- Aufwand und Kosten je Kostenstelle
- Externer vs. interner Personalaufwand, etc.

4. Support

- PC-Service pro Abteilung
- Fehler in Netzwerk und PC-Software, etc.

3 Zu erfassende Daten

In diesem Abschnitt wird die generelle Klassifikation für eine Implementierung diskutiert, die als Datenmodell in Form von *Objekttypen, Attributen* und *Wertebereichen* dargestellt werden kann [Spitta 1997].

Wir unterstellen das Modell eines dynamischen Lebenszyklus für Software, das allgemein als *‚State of the Art'* akzeptiert ist (s. Abb. 1). Die Implikationen aus diesem Modell für die zu erfassenden Daten sind:

- Sie beschreiben den Lebenszyklus eines *Systems* oder *Teilsystems*, welches sich in den Stadien *Entwicklung* oder *Wartung* befinden kann.

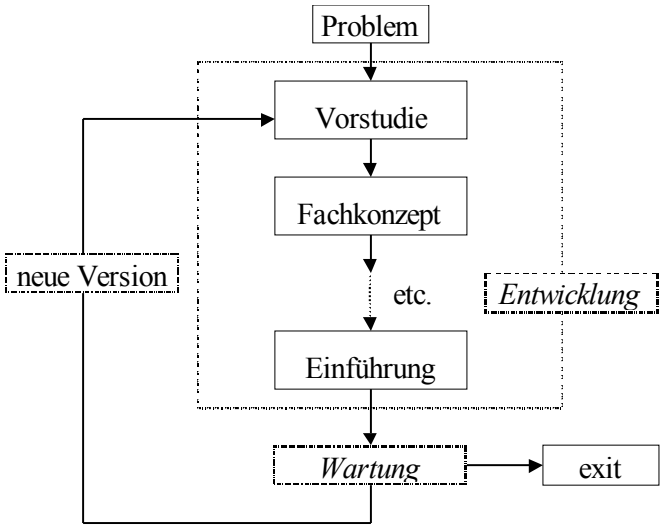


Abb. 1: Zyklisches Vorgehensmodell für Software

- Entwicklung ist eine Folge von *Versionen* (oder *Releases*) [Spitta 1989]. Der Entwicklung einer Version folgt eine Betriebsperiode mit Wartungsaktivitäten. In dieser Betriebsperiode werden Anforderungen für die nächste Version gesammelt. Der Vorgang zur Erstellung einer ersten oder neuen Version ist ein *Projekt,* das einem *System* zugeordnet wird. Während Versionen immer strikt sequenziell aufeinander folgen, kann es sinnvoll sein, Projekte in Teilprojekte zu zerlegen, die parallel durchgeführt werden können.

- Die zu erfassende Leistung wird in *Arbeitsstunden* gemessen. In einem Projekt können diese den Projektphasen zugeordnet werden, die generell (aber nicht in allen Fällen!) sequenziell verlaufen. Es gibt zwei unterschiedliche Arten von Zuordnungen zu Phasen: Die Phasen eines ganzen Projektes und die Phasen einer einzelnen Aktivität. Beispielsweise kann es während der Projektphase *Einführung* notwendig werden, einige zusätzliche Funktionen zu *spezifizieren*. Beide Arten von Phasen sollten für Zwecke des Projektmanagements erfasst werden können. Der erste Typ von Phase darf nur von einem Projektmanager gesetzt werden, während die Phasenzuordnung von Aktivitäten durch die Entwickler geschieht. Dies vermeidet Klassifikationsfehler.
- In den Phasen gibt es methodische Typen von Aktivitäten (Funktions-/Datendesign, Codierung, Test, etc.), die parallel durchgeführt werden. Das Erfassen dieser Aktivitäten würde zu erheblichen Klassifikationsfehlern führen und auch den Erfassungsaufwand steigern. Obwohl technisch möglich, sollten Aktivitätstypen *nicht* kodiert werden. Die Kodierung kleinster Details konfrontiert die Entwickler mit unlösbaren Klassifikationsproblemen und vergrößert die Fehlerquote, anstatt die Daten genauer zu machen.
- Details sollten als „freier Text“ eingegeben werden, der nicht für Klassifikationen taugt, jedoch wichtige Informationen für Projektmanager enthält.
- In der Wartung sind keine Phasen zu kontieren.
- Ein Projekt oder System ist einer *Kostenstelle* zugeordnet. Aufgrund der heutzutage sehr „flüchtigen“ Unternehmensstrukturen ist die Zuordnung zu einem *AnwendungsSystem* stabiler als zu einer Kostenstelle. Die Klassifikation der Anwendungssysteme sollte nicht firmenspezifisch sein, sich aber auf den speziellen Einsatzbereich der Software beziehen. Abbildung 7 in Abschnitt 5 ist ein Beispiel für den industriellen Bereich. Das ist nicht nur wichtig für ein „Benchmarking“ zwischen Unternehmen, sondern auch für die Stabilität der Daten bei Änderungen in der Unternehmensstruktur. Eine Zwei-Ebenen-Hierarchie der Anwendungssysteme ist notwendig und hinreichend. Mehr als zwei Ebenen erschweren Handhabung und Übersicht.

Als Ergänzung zum Lebenszyklus von Software mit den drei Aufwandsklassen *Entwicklung*, *Wartung* und *Fehler* bezieht sich eine vierte Aufwandsklasse eher auf Hardware oder Office-Software. Dies ist die Klasse *Support*, die nicht in Phasen zu unterteilen und á priori keiner Kostenstelle zuzuordnen ist. Die Festlegung der Kostenstelle erfolgt erst bei Inanspruchnahme einer Leistung. Alle Aufwandsklassen sind *Ereignisse,* bis irgendetwas mit dem Ereignis getan wird (Fehlerkorrektur, Arbeit am Projekt, Erweiterung einer gekauften Software). Ereignisse werden von Managern kontiert, nicht von Entwicklern. Da sie selten auftreten, kann bei ihnen jeder Einzelfall auf Klassifikationsfehler überprüft werden. Von Managern kann angenommen werden, dass sie an korrekten Klassifizierungen interessiert sind, da sie von den Benutzern anhand der verrechneten Leistung überwacht werden. Dies ist die Lösung für das *Motivationsproblem* bei der Erfassung von Ereignissen.

Abbildung 2 zeigt die Eingabemaske der Aufwandserfassung eines Arbeitstages mit authentischen Daten in modernisiertem Design. Es ist zu erkennen, dass nur wenige Felder für eine Kontierung gefüllt werden müssen. Die tabellarische Maske trägt zur Lösung des Motivations- und Aufwandsproblems bei. Entwickler verlangen eine effiziente Datenerfassung, die ausschließlich tastaturgesteuert sein muss.

Legende: grau: Vom System angezeigte Daten. Nach RETURN werden Default-Daten (Kostenstelle und Entwicklungsschritt) angezeigt. Service-Arten: f= Fehler; w= Wartung; s= Support; p= Projekt.

Abb. 2: Erfassungsmaske für einen Tagesbericht

4 Einsatzbedingungen

Da ein Hauptproblem der Datenerfassung die Validität ist, sollte alles formal Überprüfbare auch durch die Implementierung von Integritätsbedingungen geprüft werden. Fehler innerhalb eines normalen Schwankungsintervalls verursachen lediglich eine statistisch erklärbare Varianz. Da pro Mitarbeiterjahr über 1000 Kontierungen anfallen, gilt für diesen Typ von Daten das Gesetz der großen Zahlen. Individuelle Unterschiede kompensieren sich, wenn mehr als fünf Entwickler kontieren.

Es bleibt die Frage nach dem *Motivationsproblem*. Was kann getan werden, um eine systematische Verfälschung der Massendaten durch die Entwickler zu vermeiden?

Es ist sicherlich nicht möglich und auch nicht sinnvoll, jeden Entwickler zu überwachen. Der einzige Weg, eine systematische Verfälschung zu vermeiden, ist sicherzustellen, dass der Entwickler kein Motiv hat, die Daten zu manipulieren. Hier hilft nur eine vertrauensvolle Arbeitsatmosphäre. Es gibt vier Faktoren, die ein solches Klima schaffen, das eine Verfälschung unwahrscheinlich macht.

1. **Eigenverantwortung**: Jeder Mitarbeiter bucht seine eigenen Daten online. Keine andere Person kann persönliche Daten anlegen oder verändern, vor allem nicht der Vorgesetzte. Die Datenbank ist abgeschottet und erlaubt nur Zugriffe durch die Buchungssoftware.
2. **Transparenz**: Jeder Mitarbeiter darf (nur online!) alle Kontierungen ansehen. Das fördert eine offene Atmosphäre zwischen Entwicklern, Managern und Teams. Nicht nur der einfache Mitarbeiter, sondern auch das mittlere Management erfasst die Aufwände.
3. **Keine Überwachung**: Auf keinen Fall darf eine persönliche Bewertung der Entwickler aus den Daten abgeleitet werden, weder eine Bewertung der Effizienz noch gemachter Fehler. Da IT-Personal mit den technischen Möglichkeiten eines solchen Systems vertraut ist und die Auswertungsmöglichkeiten kennt, ist dies ein besonders wichtiger Punkt bei der Nutzung einer Aufwandsdatenbank. Wenn die Mitarbeiter bemerken, dass sie anhand dieser Daten bewertet werden, ist es wahrscheinlich, dass sie die Daten verfälschen. Die Daten können sogar dazu benutzt werden, die Arbeitsplätze der Mitarbeiter zu schützen, etwa bei Fragen der Art: *„Was tun diese vielen teuren Leute denn eigentlich den ganzen Tag?“* In diesem Zusammenhang kann ein Soll/Ist Vergleich problematisch sein. Misst

man Mitarbeiter an der Varianz ihrer Planabweichung, so werden sie versuchen, geringe Abweichungen zum Plan zu erreichen, also Istdaten falsch zu klassifizieren.

4. **Aufklärung**: Neue Mitarbeiter und externe Entwickler müssen über die Ziele und Regeln des Systems informiert werden. Das sollte am besten anhand von Berichten geschenen (s. Abschnitt 5). Alle buchenden Mitarbeiter müssen darüber informiert sein, was mit ihren Daten geschieht.

Die verbleibenden zwei Punkte sind ebenfalls nützlich für die Korrektheit und Verwendbarkeit der Daten, sie beeinflussen jedoch die Motivation weniger:

5. **Vollständigkeit**: Ein Mitarbeiter sollte entweder für alle seine Aktivitäten Daten erfassen oder für gar keine. Entwickler, Support-Mitarbeiter und Projektmanager gehören zur ersten Klasse. Operator, Netzwerkspezialisten etc. zur zweiten. Dadurch wird eine Verschiebung von Aufwand zwischen Aktivitäten oder Aufwandsklassen oder das Weglassen von Aufwand verhindert. Dieser Punkt ist zentral bei der Lösung des *Klassifikationsproblems*.
6. **Aktualität**: Der letzte Tag der Datenerfassung ist der letzte Arbeitstag in der Woche. Nur aktuelle Daten entsprechen dem tatsächlichen Geschehen.

5 Beispiele aus empirischen Daten

Die folgenden Beispiele aus Originaldaten zeigen, dass ein System zur Aufwandserfassung für mehr als ein Ziel nützlich ist, als da wären: *Projektmanagement* (Steuern und Berichten), *Qualitäts-Engineering* (Wartungsaufwand und Versionsentwicklung) und *Informationsmanagement*. So löst die mehrfache Nutzung der Daten das Aufwandsproblem.

5.1 Projektmanagement

Abbildung 3 zeigt den Aufwand, der an drei Arbeitstagen für ein Projekt angefallen ist. Das Projekt befindet sich in Phase 5 (Implementierung). Trotzdem gibt es verschiedene Aktivitäten in Phasen < 5 (z.B. 3= Spezifikation). Nur einer der vier Entwickler (D) hat an einem Tag (02.04.1992) ausschließlich für dieses Projekt gearbeitet (Fettdruck). Dieser Report gibt dem Projektleiter den Hinweis – nicht mehr – die zusätzlichen Spezifikationen zu überprüfen. Sind diese Aktivitäten a) notwendig b) abgesegnet oder sind sie c) „subversiv“ [Welz & Ortmann 1992]? Ein zweiter Blick zeigt, dass Entwickler B anscheinend bereits an Erweiterungen arbeitet, obwohl sich das System noch im Test befindet (grau unterlegt).

ServiceNr: P475 AnwSyst: L.LGB KostenSt: 800

Auszug aus den Kontierungen

Datum	UserID	Aufwand [h]	Entw Schr	Tätigkeit
04.02.92	A	1,0	3	Abstimmen Aktivitäten mit PL (=Projektleiter)
	B	1,0	3	Abstimm Aktivitäten LGB 1.HJ
	B	2,5	5	Erstellen automat. Umlagerungen
	C	1,0	5	Pflege Aktivitätenplan
	C	5,0	5	Sitzung Geschäftsleitung
	D	**1,5**	**5**	**LGB2942P Liste getestet**
	D	**3,0**	**5**	**LGB2105P (online) überarbeitet**
	D	**3,0**	**5**	**Listenaufstellung aktualisiert**
05.02.92	B	1,0	5	Erweit Lagerstellen
	B	3,5	5	Neuprogr LG-Entnahme-Beleg
	D	3,0	5	Überarb LGB2105P (online) u. Test
06.02.92	A	0,4	5	Änderung Prog QBE750A
	B	1,1	3	Erweitern autom. Umlagerungen
	B	1,5	5	Erweit Druck WB-Schein fremde Auftr-Nr
	D	3,0	5	LGB2105P getestet und in TEST gestellt
	D	1,0	5	LGB2927P getestet: Charge fehlt

Abb. 3: Kontierungen des Projektes Lagerbestandsführung für drei Tage

Abbildung 4 zeigt den aggregierten Aufwand externer Mitarbeiter in den großen Projekten über fünf Quartale. Die Sicht der Buchhaltung würde sich auf die Kostenstruktur konzentrieren: *„Externe Kosten sind relativ konstant (auf einem hohen Level)“*. Aus Sicht des Projektmanagements sieht die Situation anders aus: Das Projekt *Vertrieb* steht kurz vor der Einführung. Ein Anstieg im letzten Quartal ist normal. Das Projekt *Produktion* befindet sich auf einem idealen Weg: Es befindet sich seit I/91 in der Einführungsphase. Der Aufwand sinkt kontinuierlich. Das Projekt *Logistik* hingegen zeigt eine alarmierende Entwicklung. Die Kurve scheint einmal mehr das Brook'sche Gesetz zu beweisen: *„Adding people to a late project makes it later.“*

Aufwand [Std] je Anwendungsgebiet

	Quartal						*Projekte*	
AnwGebiet	IV/90	I/91	II/91	III/91	IV/91	I/92	:	nicht in der Grafik
Vertrieb	456	1557	1611	1682	1292	1745		Aufträge, Versand
Logistik	2070	1787	1533	2258	2548	2786		Bestände, Artikeldaten
Produktion	2511	2102	1916	1664	1362	839		PPS-System
Informatik	421	441	227	97	195	136		90/91: PC-Vernetzung
Admin.	107	82	195	150	76	127		überw iegend SAP
Wartung*	90	29	78	12	50	180		*= ad hoc - Wartung

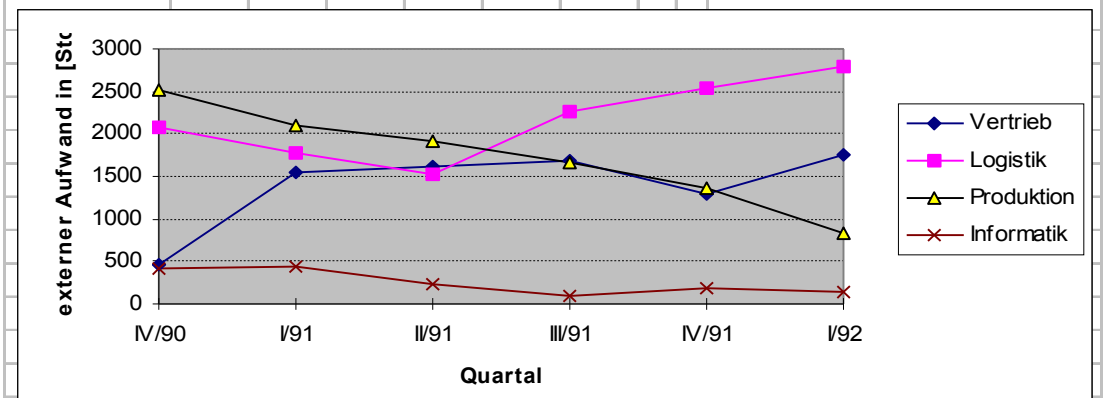


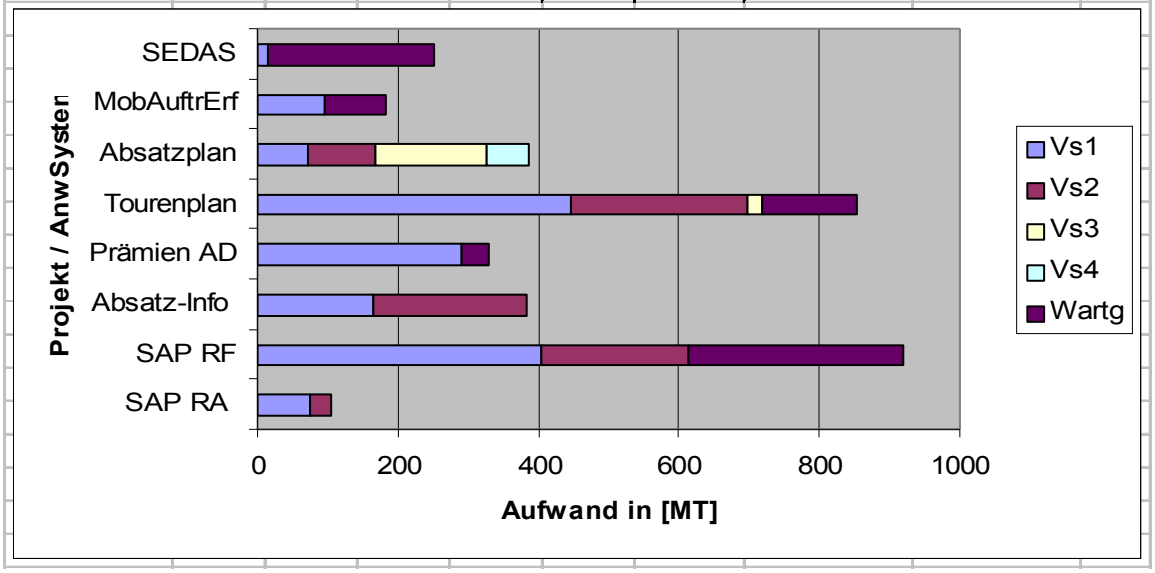
Abb. 4: Personaleinsatz Externe über 5 Quartale je Anwendungsgebiet

5.2 Qualitätsüberwachung

Wartungsaufwand verrät mehr über Qualität als Entwicklungsaufwand. Abbildung 5 zeigt die Entwicklung einiger Systeme und deren kumulierten Wartungsaufwand. Um die Werte richtig interpretieren zu können, muss man die Zeitverteilung eines Projektes und den organisatorischen Kontext der Daten kennen. Als Beispiel die Interpretationen für drei Systeme in Abbildung 5:

SEDAS ist ein Vorläufer des internationalen Standards EDIFACT für den Handel. Das System wurde unter hohem Druck eines wichtigen Kunden 1987 ‚quick and dirty' eingeführt. Der relativ große Wartungsaufwand beweist, dass ein Qualitätsproblem vorliegen muss, obwohl es sich um ein relativ simples Batch-Interface handelt. Die *Mobile Auftragserfassung* dagegen ist ein ambitioniertes und komplexes System für 140 Vertreter, das fast gleichzeitig eingeführt wurde. Der Wartungsaufwand (ca. 70% des Entwicklungsaufwandes) in 5½ Jahren scheint ein Zeichen für ausreichende Qualität zu sein. Die *Absatzplanung* wurde evolutionär entwickelt. Die vier Versionen ohne stabile Wartung können entweder als Indikator für eine sehr „kreative“ Fachabteilung oder „kreative“ Entwickler sein, die jeder technologischen Modeerscheinung folgen. Ab Version 3 wurde auf ein OLAP-System „gesetzt“.

Aufwand [MT] pro Version u. Wartung							
Projekt	Vs_1	Vs_2	Vs_3	Vs_4	Wartg	seit	1 MitarbeiterTag = 8 Stunden
SAP RA	74	30			0	Feb 92	0 = Wartung nicht erfaßt
SAP RF	404	211			304	Okt 91	Wartung für RF und RA
Absatz-Info	165	217			0	Jul 91	
Prämien AD	291				39	Jul 89	
Tourenplan	447	250	21		134	Dez 89	
Absatzplan	72	97	157	61	0	Jun 87	Vs_4 zum Stichtag noch nicht beendet
MobAuftrErf	97				87	Aug 87	
SEDAS	16				234	Okt 87	



Abkürzungen: Datentransfer nach dem Standard SEDAS, Mobile Auftragserfassung, Prämiensystem Außendienst, Finanz- und Anlagenbuchhaltung des Systems SAP R/2

Abb. 5: Entwicklungs- und Wartungsaufwand mittelgroßer Projekte 1.1.1987 bis 30.6.1992

5.3 Informationsmanagement und IT-Controlling

Die Datenbank erlaubt auch eine langfristige Betrachtung der Ressource Software. Abbildung 6 zeigt einen Report über Projekte des Vertriebsbereiches. In diesem Fall dient ein Attribut *strategisch relevant?* dazu, große und wichtige Projekte von kleineren Projekten zu unterscheiden. Der Leser wird auch abgebrochene Projekte entdecken. Der analoge Report für Wartung oder Projekte mit der „Phasennummer“ 0 würde die ‚Pipeline‘ der Projektwünsche zeigen, für die noch kein Aufwand entstanden ist.

Stand: 31.03.1992

Serv. -Nr	Thema	Auf-wand	Entw Schr
	strategische Projekte		
154	Absatz-Info Artikel/Reisender	714	+
281	Tourenplanung	1558	+
340	Kundenaufträge/Versand	11300	5
348	Integr. Altsyst. LebensmittelH.	6477	+
579	Faktura Lebensmittelhandel neu	1259	+
680	*Grüner Punkt*	69	3
*** **Vertrieb**		**21308**	
260	Festlegen Artikelstruktur	1777	+
323	Sanieren Artikeldaten	1071	+
371	Artikel/Stückl./Arbeitspläne	19031	6
*** **Grunddaten**		**21879**	
	sonstige Projekte		
301	Ablösen Faktura Fachhandel	65	+
335	*Deckungsbeiträge*	97	!
355	*Versandänderung Sparte B*	158	+
361	Warenanforderung grob	742	+
439	Sanieren Faktura Altsystem	140	!
518	*Änd. Gebiets-/Bezirks-Schlüssel*	18	!
*** **Vertrieb**		**1220**	
374	Integr. Schlüsselverzeichnisse	280	+
405	Farbbündel Dreierpacks	21	!
*** **Grunddaten**		**301**	

Legende: *kursiv*: reaktives ~, untersstr.: nachträgliches Projekt.
+: in Betrieb; !: abgebrochen; <zahl>: Phasen-Nr.

Abb. 6: Projekte aus einem Anwendungsbereich

Die Liste zeigt Aspekte des strategischen Informationsmanagements. In Ergänzung zu geplanten Projekten ergeben sich unvorhergesehene Probleme in strategischen Projekten: *Nachträgliche* Teilprojekte. So sind etwa die Artikelstammdaten sanierungsbedürftig (ServiceNr 323) und müssen neu spezifiziert werden (ServiceNr. 260) um das strategische Projekt 371 nicht zu gefährden. Auf der anderen Seite gibt es *reaktive* Projekte wie „Grüner Punkt“ (ServiceNr. 680), die durch neue Gesetze oder wichtige Kunden erzwungen werden.

Abbildung 7 zeigt eine Aggregation der kompletten Datenbank für die ersten 6½ Jahre als Verteilung von Service-Ereignissen über die Anwendungsstruktur. Die Klassifikation der Anwendungssysteme war über diesen langen Zeitraum stabil geblieben, obwohl sich die Organisationsstrukuren des Unternehmens zweimal vollständig verändert hatten. Dies hatte jedesmal zu einer Veränderung der Firmen- und Kostenstellenstrukturen geführt. Man sieht, dass auch eine Ereignis-Sicht ein wichtiges Hilfsmittel für das Projekt- und Informationsmanagement ist. Hierzu sollen zwei Auffälligkeiten diskutiert werden: Die große Anzahl abgebrochener Aktivitäten und die Anzahl gleichzeitig in Arbeit befindlicher Projekte in verschiedenen Anwendungsgebieten. Für die zweite Frage wurden die beiden schattierten Zahlen in Unter-Tabellen verfeinert.

Die hohe Zahl abgebrochener Aktivitäten ist vermutlich ein Kennzeichen mittelständischer Unternehmen. Dort gibt es eine Kultur der flexiblen Reaktionen auf den Markt. Dies ist eigentlich eine Stärke des Mittelständlers. Sie führt jedoch *auch* zu spontanen Anforderungen im IT-Bereich mit kurzfristigen Prioritätsänderungen. Wird dieser Prozess nicht überwacht, kann es zu einer Verschwendung von IT-Kapazitäten und zu einer Gefährdung strategischer Projekte führen.

Eine Verfeinerung der Fälle *in Arbeit* der Gebiete *Produktion* und *Vertrieb* (schattiert) zeigt interessante Hintergründe. Die Produktion hat gemäß der Detailsicht 25 Projekte in Arbeit, 3 in der Einführung und 3 in der ‚pipeline'. Das dürfte ein Anzeichen für ein chaotisches Projektmanagement sein.

Anzahl IsService-Fälle in 6,5 Jahren: 01.01.1987 - 30.06.1993													
SvArt	*F + S + W*				*Projekte*				*gesamt*				Sum
AnwGebiet	s	!	i.A.	"+"	s	!	i.A.	"+"	s	!	i.A.	"+"	
Admin.	12	4	6	64	4	4	8	25	16	8	14	89	127
Informatik	5	0	9	48	4	2	5	23	9	2	14	71	96
Logistik	7	1	7	48	11	15	19	24	18	16	26	72	132
Produktion	3	1	6	26	6	2	31	19	9	3	37	45	94
Vertrieb	31	20	19	202	23	14	28	51	54	34	47	253	388
gesamt	58	26	47	388	48	37	91	142	106	63	138	530	**837**

Details für das Projektmanagement:						('+' für das letzte Halbjahr)
Produktion	(+)	(-)	"1-6"	"0"	*Ges*	
Wartung	3		3	3	6	
Projekte	4	3	25	3	31	
gesamt	7	3	28	6		
Vertrieb	(+)	(-)	"1-6"	"0"	*Ges*	
Wartung	19		3	16	19	
Projekte	9	2	8	18	28	
gesamt	28	2	11	34		

Legende:	s: storniert, ! : Abbruch
(+):	fertig seit 01.01.93
(-):	in Einführung
"1-6"	in Arbeit (i.A.)
"0"	registriert ('pipeline')

Abb. 7: Anzahl Service-Fälle in 6½ Jahren für 271.077 Entwickler-Stunden

Es ist sehr wahrscheinlich, dass die meisten dieser „Projekte" abgebrochen werden und der in sie investierte Aufwand ohne Nutzen verloren ist. Die Vermutung wird dadurch gestützt, dass im vergangenen Halbjahr nur 4 Projekte fertig geworden waren. Im Gegensatz dazu zeigt der Vertrieb ein diszipliniertes Bild. 19 Wartungsaktivitäten und 9 Projekte wurden im letzten Halbjahr beendet, nur 2 Projekte sind in der Einführung. Es gibt weiterhin eine große ‚pipeline' von 16 Wartungs- und 18 Entwicklungsaktivitäten.

Unsere Erhebungen zum Aufwandsproblem ergaben 1,2 bis maximal 2% des Gesamtaufwandes am Erfassungsaufwand. Der Gesamtaufwand enthält nur produktive Zeiten, da Fehl- und Urlaubszeiten nicht kontiert wurden. Ein Teil dieses Aufwandes ist die Onlineerfassung durch die Entwickler selbst. Dies ist, wie in Abschnitt 4 diskutiert, eine gute Investition in die Korrektheit der Daten. Weiterhin kann angenommen werden, dass die Effekte für Transparenz und Kostenverantwortung höher sind als der Aufwand für die Datenerfassung.

6 Fazit

Es wurde gezeigt, wie man eine universelle Software für die Aufwandserfassung konzipiert, die zuverlässige Daten für technologische, kostenrechnerische und Management-Zwecke liefert. Das technische Design muss kombiniert werden mit geeigneten Einsatzbedingungen. Diese beiden Elemente einer Datenerfassungsumgebung fördern das notwendige Klima des Vertrauens für die Mitarbeiter, damit korrekte Daten entstehen. Da die Personalkosten den weitaus größten Teil der IT-Kosten ausmachen, ist eine Aufwands-Datenbank ein unverzichtbares Mittel für das IT-Controlling. Sie verwirklicht das ‚corporate memory', das Boehm vor 19 Jahren für die IT gefordert hatte [Boehm 1981] und ist unverzichtbare Basis für Planungsprozesse des Controlling. Heute sagt man Erfahrungs- oder „Wissensbasis" dazu.

7 Literatur

[Boehm 1981] Boehm, B.W.: Software Engineering Economics. Prentice Hall, Englewood Cliffs 1981.

[Boehm & Papaccio 1988] Boehm, B.W.; Papaccio, P.N.: Understanding and Controlling Software Costs. IEEE Transactions on Software Engineering 14(1988) 10, 1462-1477.

[Brooks 1975] Brooks, F.P.: The Mythical Man Month, Addison-Wesley, Reading/MA 1975.

[Page-Jones 1991] Page-Jones, M.: Praktisches DV-Projektmanagement. Hanser, München - Wien, 1991.

[Spitta 1989] Spitta, Th.: Software Engineering und Prototyping. Springer, Berlin - Heidelberg et al. 1989.

Spitta 1997] Spitta, Th.: Die Gewinnung korrekter Daten aus manuellen Aufschreibungen: Empirische Ermittlung eines Erfassungssystems. In: Grün, O.; Heinrich, L.J. (Hrsg): Empirische Forschung in der Wirtschaftsinformatik, Springer, Wien - New York 1997, 105-118.

[Spitta 1998] Spitta, Th.: IV-Controlling in mittelständischen Industrieunternehmen - Ergebnisse einer empirischen Studie. Wirtschaftsinformatik 40(1998) H.5, 424-433.

[Welz & Ortmann 1992] *Welz, F; Ortmann, R.:* Das Softwareprojekt - Projektmanagement in der Praxis. Campus, Frankfurt - New York 1992.

Schlagwörter: Aufwandserfassung, Entwicklung, IT-Controlling, Projektmanagement, Wartung, Service

Kurztitel: IT-Aufwandserfassung

Dipl.-Kfm. Arne-Christian Sigge
Prof. Dr.-Ing. Thorsten Spitta
Universität Bielefeld
Fakultät für Wirtschaftswissenschaften
Postfach 10 01 31
33501 Bielefeld
{aSigge|thSpitta}@wiwi.uni-bielefeld.de