Technische Universität Ilmenau
Fakultät für Informatik und Automatisierung
Fachgebiet: Prozessinformatik

Betreuer: Dipl.–Inf. Detlef Streitferdt

Hauptseminar
Wintersemester 2002 / 2003
zum Thema

Untersuchung von Kostenanalyseverfahren bei der Softwareentwicklung

Bearbeiter: Martin Lange
Termin: 27. Januar 2003

Gliederung

1. EINLEITUNG 4

2. BEGRIFFSKLÄRUNG 4

3. GRUNDLAGEN 5

3.1 Einflussfaktoren der Aufwandsschätzung 6

3.2 Metriken zur Bestimmung des Softwareumfangs 8

4. METHODEN DER AUFWANDSSCHÄTZUNG 9

4.1 Algorithmische Methoden 9

4.2 Vergleichsmethoden 10

4.3 Kennzahlenmethoden 11

4.4 Expertenbefragungen 11

5. AUSGEWÄHLTE SCHÄTZVERFAHREN 13

5.1 Verfahren nach Surböck 13

5.2 Bottom–up-Verfahren nach Aron 14

5.3 Produktivitätsverfahren von Walston und Felix 14

5.4 „Business Objectives“ Verfahren 15

5.5 Function-Point–Verfahren 15

5.6 Data–Point–Verfahren 17

5.7 Object–Point–Verfahren nach Sneed 18

5.8 COCOMO–Verfahren 19

6. BEWERTUNG VON SCHÄTZVERFAHREN 21

6.1 Anforderungen an Schätzverfahren 21

6.2 Übersicht der ausgewählten Schätzverfahren 22

7. ZUSAMMENFASSUNG UND AUSBLICK 23

LITERATURVERZEICHNIS 24

ABKÜRZUNGSVERZEICHNIS 25

ABBILDUNGSVERZEICHNIS **26**

ANHANG A – VERFAHREN NACH SURBÖCK **27**

ANHANG B – BOTTOM–UP–VERFAHREN NACH ARON **29**

ANHANG C – FUNCTION–POINT–VERFAHREN **30**

ANHANG D - DATA–POINT–VERFAHREN **33**

1. Einleitung

Problemstellung

Bei der Softwareentwicklung sind Produktumfang, Projektdauer, Kosten und Produktqualität die wichtigsten Ziele. Oft werden nicht alle oder sogar keines dieser Ziele erreicht. Grund hierfür sind zu ungenaue Aufwandsschätzungen, denn in vielen Fällen ist der tatsächliche Aufwand, z. B. für einen bestimmten Produktumfang, höher, als man vorher geschätzt hat. Dies bedeutet in der Praxis, dass man dem Kunden keinen festen Preis für die Entwicklung einer Software nennen kann. Um dieser Problematik zu begegnen, versucht man Methoden und Verfahren zur Aufwandsschätzung zu entwickeln, welche eine solide Basis für die Kostenanalyse schaffen sollen. Mit Hilfe dieser Werkzeuge will man die Kosten besser analysieren und kalkulieren können, um möglichst genaue Schätzungen zu erhalten und dem Kunden frühzeitig einen Preis für das Softwareprodukt nennen zu können.

Zielstellung

Ziel dieses Dokuments ist es, einen Überblick über die Methoden der Aufwandsschätzung zu geben und ausgewählte, auf diese Methoden basierende, Verfahren vorzustellen. Anschließend sollen Anforderungen an ein Schätzverfahren aufgestellt werden. Ergebnis soll eine Tabelle sein, die eine Zusammenfassung der ausgewählten Verfahren liefert, wobei auf verwendete Methoden, Umfangsmetriken und benötigte Informationen eingegangen werden soll.

Weg zur Problemlösung

Zunächst wird ein linearer Zusammenhang zwischen Kosten und dem Aufwand bei der Softwareentwicklung unterstellt. Aufgrund dieser Annahme wird untersucht, welche Faktoren einen wesentlichen Einfluss auf den benötigten Aufwand haben und welche Metriken existieren, um den Umfang einer Software vergleichen zu können. Anschließend werden zunächst Methoden, und später auf diese Methoden basierende Verfahren der Aufwandsschätzung vorgestellt. Anschließend erfolgt eine Auflistung von Anforderungen an die Schätzverfahren. Darauf aufbauend kann ein Überblick über die Verfahren gegeben und auf die für das jeweilige Verfahren benötigten Informationen eingegangen werden.

2. Begriffsklärung

Um im Folgenden Verfahren und Methoden der Aufwandsschätzung vorstellen zu können, müssen diese Begriffe zunächst definiert werden. Da der Aufwand meist in Personentagen oder Personenmonaten angegeben wird, empfiehlt es sich diese Maßeinheiten zu erläutern. Außerdem soll definiert werden, was in diesem Dokument unter dem Begriff „Kosten“ verstanden wird und auf damit verbundenen Auswirkungen eingegangen werden.

Methode

Der Begriff kann wie folgt definiert werden: „Methoden sind planmäßig angewandte, begründete Vorgehensweisen zur Erreichung von festgelegten Zielen (i.a. im Rahmen festgelegter Prinzipien).“[1]

Verfahren

Im Gegensatz zur Methode sind Verfahren konkreter. „Sie beschreiben einen festdefinierten Weg zur Lösung eines bestimmten Problems.“[2] Verfahren zeichnen sich durch die Verwendung von Kennzahlen aus.

Kosten

Im Rahmen dieses Dokuments wird unterstellt, dass sich die Softwareentwicklungskosten einzig aus den Personalkosten zusammensetzen. Somit ergibt sich ein linearer Zusammenhang zwischen dem Aufwand (z. B. in Personenmonaten) und den Softwareentwicklungskosten.

Aufwandsschätzung

„Die Aufwandsschätzung bildet im Rahmen des Managements von IV – Projekten die Basis der Kapazitäts-, der Termin- und der Kostenplanung.“[3] Der Aufwand wird dabei meist in Personentagen oder Personenmonaten angegeben und kann unter Berücksichtigung der vorherigen Definition direkt in Kosten umgerechnet werden.

Personentag, Personenmonat

Mit einem Personentag ist die Leistung gemeint, die eine Person innerhalb eines Tages erbringt. Ein Personenmonat ist demzufolge die Leistung innerhalb eines Monats. Die Kosten für den Aufwand von einem Personenmonat sind somit gleich den durchschnittlichen Kosten für einen Mitarbeiter je Monat.

3. Grundlagen

Die Softwareentwicklungskosten werden, unter Berücksichtigung der vorhergehenden Definitionen, durch den zu erbringenden Aufwand bestimmt. Sie lassen sich berechnen durch die benötigten Personenmonate, mal dem Kostensatz je Person und Monat. Die Aufgabe der Kostenanalyse besteht nun darin, Einflussfaktoren zu bestimmen, welche den benötigten Aufwand entscheidend beeinflussen. Einer dieser Einflussfaktoren ist der Softwareumfang. Um diesen vergleichbar zu machen und ihn somit als Grundlage für Berechnungen verwenden zu können, existieren verschiedene Metriken, sogenannte Umfangsmetriken, welche im nachfolgenden erläutert werden sollen. Die Verfahren zur Aufwandsschätzung kann man somit hinsichtlich der berücksichtigten Einflussfaktoren und der verwendeten Umfangsmetrik unterscheiden.

[1] [HeMeFr92, S. 54]
[2] [Knöll91, S. 18]
[3] [Noth01, S. 54]

3.1 Einflussfaktoren der Aufwandsschätzung

Die entscheidenden Einflussfaktoren für den benötigten Aufwand (Kosten), sind Quantität, Qualität, Projektdauer und Produktivität.[4] Wobei Qualität, Quantität und Produktivität maximiert, sowie die Projektdauer minimiert werden sollen. Das zusammenwirken dieser Faktoren wird durch das Teufelsquadrat (siehe Abbildung) anschaulich dargestellt. Sobald man an einer der Ecken zieht, wird mindestens eine andere gestaucht. Dies bedeutet, um ein Ziel besser erreichen zu können, wird mindestens ein anderes Ziel schlechter erreicht.

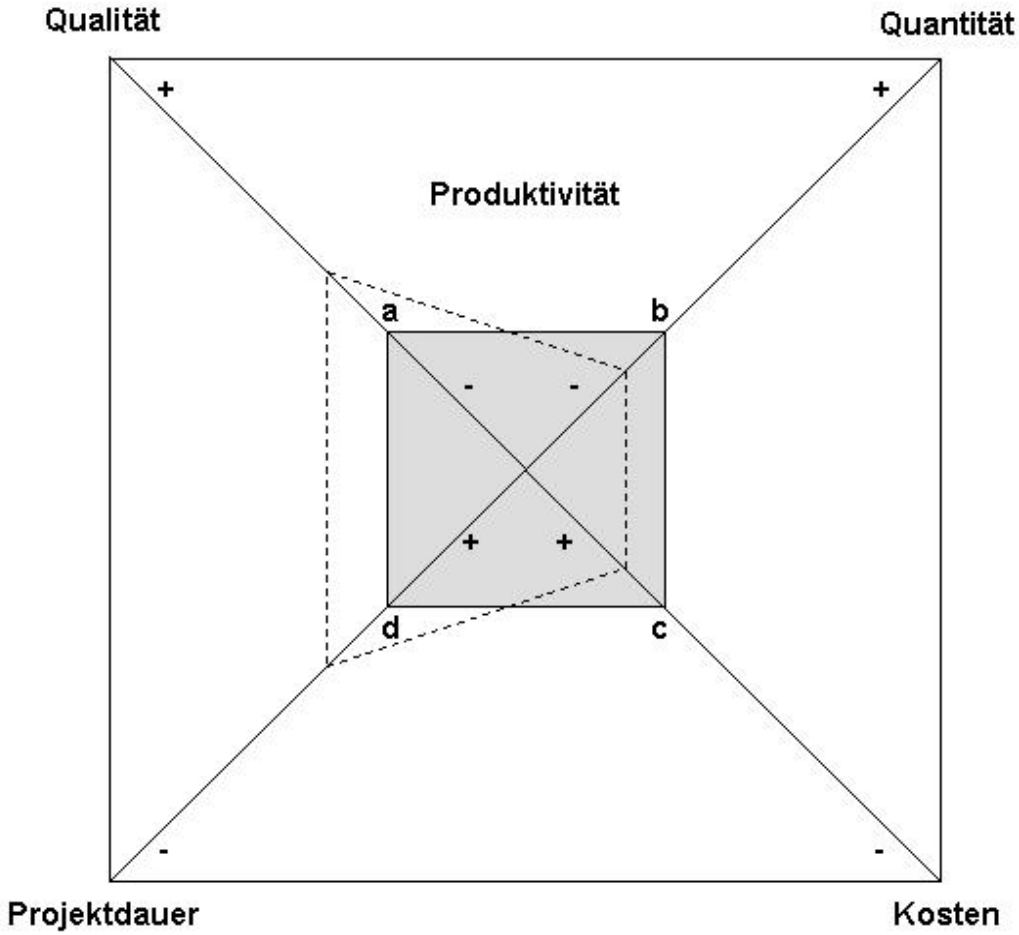

Abbildung 3.1 / 1 Teufelsquadrat [5]

Quantität

Die Größe einer Software wird als das wichtigste Merkmal angesehen. Sie wird in einer bestimmten Maßeinheit angegeben, die wohl gebräuchlichste ist Lines of Code. Aber es existieren auch andere Umfangsmetriken, welche zunehmend an Bedeutung gewinnen. Im nächsten Kapitel werden eine Reihe von Umfangsmetriken zur Bestimmung der Softwaregröße vorgestellt. Der Quantitätsfaktor wird oft durch fest definierte funktionale Zusammenhänge in Aufwand (z. B. in Personenmonate) umgerechnet. Wie das Teufels-

[4] für die folgenden Absätze vgl. [Knöll91, S. 23 – 35]
[5] vgl. [Knöll91, S. 25]

quadrat aber anschaulich zeigt, müssen zusätzlich die Qualität, die Projektdauer und die Produktivität berücksichtigt werden.

Qualität

Unter Qualität versteht man diejenige Beschaffenheit einer Software, die sie für eine spätere Verwendung geeignet macht.[6] Der Qualitätsfaktor ist dabei ein Produkt aus Effizienz, Effektivität und Wartbarkeit. Die Effizienz setzt sich aus Zeiteffizienz (Antwortzeit) und Raumeffizienz (Speicherauslastung) zusammen. Die Effektivität wird bestimmt durch Zuverlässigkeit, Sicherheit und Benutzerfreundlichkeit. Der Faktor Wartbarkeit hängt ab von der Ausbaufähigkeit, Übertragbarkeit und Wartungsfreundlichkeit.
Eine Berechnung des Qualitätsfaktors erfolgt meistens durch die Einteilung der Faktoren in vorher definierte Kategorien. Diesen Kategorien sind in einer Tabelle konkrete Werte zugeordnet, welche durch Analysen abgeschlossener Projekte gewonnen werden und von den verwendeten Verfahren abhängen.
Um die Qualitätsanforderungen in der Aufwandsschätzung zu berücksichtigen, wird die Quantität mit diesem Qualitätsfaktor multipliziert.

Projektdauer

Der Zusammenhang zwischen Aufwand und Projektdauer ist nicht linear, wie man zunächst annehmen könnte, sondern der Aufwand steigt mit verringerter Projektdauer. Begründen kann man dies damit, dass bei einer Verkürzung der Projektdauer mehr Mitarbeiter an dem Projekt teilnehmen müssen, weil die benötigten Personentage in kürzerer Zeit erbracht werden sollen. Da aber teilweise eine Unteilbarkeit von Aufgaben vorliegt und der Kommunikationsaufwand steigt, sinkt die Produktivität der Mitarbeiter und somit steigt der Aufwand und somit auch die Kosten.
Für die Bestimmung der optimalen Projektdauer und Mitarbeiterzahl wird oft folgende Fausformel verwendet:

[7]

Optimale Projektdauer in Monaten = Optimale Mitarbeiterzahl – $(\text{Aufwand in PM})^{1/2}$

Nach Sneed kann die optimale Projektdauer in Abhängigkeit von der Art der zu entwickelten Software wie folgt berechnet werden:

[7]

Batchsysteme:	Optimale Projektdauer = $2{,}5 \bullet (\text{Aufwand in PM})^{0{,}38}$
Onlinesysteme:	Optimale Projektdauer = $2{,}5 \bullet (\text{Aufwand in PM})^{0{,}35}$
Realtimesysteme:	Optimale Projektdauer = $2{,}5 \bullet (\text{Aufwand in PM})^{0{,}32}$

Produktivität

Produktivität ist das Verhältnis von Quantität und Qualität zum Aufwand.[8] Eine Produktivitätssteigerung bedeutet daher in der gleichen Zeit eine bessere oder umfangreichere Software zu produzieren. Die Produktivität kann durch den produzierten Softwareumfang in Abhängigkeit von der Zeit (z. B. Lines of Code / Zeit) angegeben werden.

Wie bereits gesagt, werden die Kosten bei der Softwareerstellung durch den Aufwand bestimmt. Dieser hängt ab von der Quantität, Qualität, Projektdauer und Produktivität.

---

[6] vgl. [Knöll91, S. 30]
[7] vgl. [Knöll91, S. 32]
[8] vgl. [Litke96, S. 24]

Die Verfahren, die den Aufwand bestimmen bzw. schätzen sollen müssen demnach, um denen, in diesem Kapitel gewonnenen Erkenntnissen gerecht zu werden, alle vier genannten Einflussfaktoren berücksichtigen.

3.2 Metriken zur Bestimmung des Softwareumfangs

Wie bereits erwähnt, ist die Quantität der bedeutendste Faktor bei der Kostenschätzung eines Softwareprojekts.[9] Um sie vergleichen zu können, drückt man sie auf Basis von Maßeinheiten, sogenannten Metriken, aus. Da sie sich auf die Größe einer Software beziehen, werden sie als Umfangsmetriken bezeichnet. Es existieren aber auch andere Metriken, wie Prozessmetriken oder Ressourcenmetriken, welche z. B. das Organisationsniveau oder das Kommunikationsniveau quantifizieren.
Bevor eine Auswahl an Umfangsmetriken vorgestellt werden, soll explizit darauf hingewiesen werden, dass kein linearer Zusammenhang zwischen der Ausprägung einer Umfangsmetrik (wie z. B. Lines of Code) und dem Personalaufwand besteht. Es ist vielmehr so, dass der Zusammenhang durch eine Funktionskurve mit steigendem Anstieg beschrieben wird. Der Grund hierfür ist die steigende Komplexität bei z. B. einer erhöhten Anzahl Lines of Code und somit einhergehend ein höherer Aufwand je Einheit. Die in der Praxis bedeutendsten Umfangsmetriken sind Lines of Code und Function Points.

Lines of Code (LOC)

Entspricht der Anzahl Programmzeilen der entwickelten Software, wobei Kommentare und Leerzeilen nicht mitgerechnet werden. Obwohl diese Art der Messung abhängig ist von der verwendeten Programmiersprache, ist sie die meist genutzte Umfangsmetrik.

Function Points

Diese Metrik bezieht sich auf die Anzahl und die Komplexität der zu implementierenden Funktionen. Eine bestimmte Anzahl Function Points drückt demnach aus, dass z. B. eine große Anzahl einfacher Funktionen oder eine geringe Anzahl komplexer Funktionen zu programmieren sind.

Kilos Delivered Source Instructions (KDSI)

Ist die Anzahl von Programminstruktionen, die erzeugt werden müssen. Jobcontrollsprache, Formatanweisungen und Datendeklarationen werden berücksichtigt, Kommentare und unmodifizierte Utility – Software hingegen nicht.[10] Es wird nur die implementierte Software angerechnet, nicht geliefert Unterstützungssoftware (z. B. Treiber) wird exkludiert (Delivered). Ein KDSI entspricht demnach 1000 erzeugten Programminstruktionen.

Data Points

Hier erfolgt eine Bestimmung des Softwareumfangs auf Grundlage der Datenmenge.[11] Die Größe der Software wird also nicht aus der Anzahl Lines of Code abgeleitet, sondern aus den betroffenen Objekten und der Summe der darin enthaltenen Datenelemente. Diese Umfangsmetrik ist somit eine Antwort auf die daten- und objektbezogene Softwareentwicklung.

---

[9] für die folgenden Absätze vgl. [LeFa90]
[10] für folgenden Absatz vgl. [Litke96, S. 143]
[11] für folgenden Absatz vgl. [Litke96, S. 55 – 56]

Object Points

Es wird der Umfang einer Software aus Objektsicht betrachtet. Ausgehend von Modellen aus der objektorientierten Programmierung erfolgt die Bewertung der Softwaregröße auf Basis von den zu implementierenden Klassen, Nachrichten und Prozessen. Die Anzahl Object Points drückt demnach deren Anzahl und Komplexität aus.

4. Methoden der Aufwandsschätzung

Im Folgenden soll ein Überblick über die Methoden der Aufwandsschätzung gegeben werden. Dieser kann nicht als Basis zur Klassifikation der Schätzverfahren angesehen werden, sondern er soll vielmehr darstellen, welche grundsätzlichen Herangehensweisen es bei der Schätzung von Aufwänden bei der Softwareentwicklung gibt. Grund für die mangelnde Eignung zur Verfahrensklassifikation ist, dass die Verfahren zumeist auf mehreren Methoden basieren und somit nicht eindeutig einer Methode zugeordnet werden können.

4.1 Algorithmische Methoden

Algorithmische Methoden bedienen sich stets einer Formel um den Aufwand in Abhängigkeit von bestimmten Einflussfaktoren darzustellen.[12] Die Struktur der Formel, die Wahl der Konstanten und die Gewichtung der Einflussgrößen, werden mit Hilfe von Korrelationsanalysen bestimmt. Durch die Berücksichtigung vieler Faktoren sind diese Methoden aufwändig, liefern aber gute Ergebnisse. Bei den algorithmischen Methoden kann man zwischen der Gewichtungsmethode und der Methode parametrischer Schätzgleichungen unterscheiden.

Gewichtungsmethode

Bei dieser Methode wird ein System von Faktoren gebildet, welche den Aufwand einer Softwareentwicklung signifikant beeinflussen.[13] Die Faktoren sind entweder subjektiver (z. B. Qualität des Personals) oder objektiver (z. B. verwendete Programmiersprache) Art. Je nach Ausprägung wird dem Faktor ein Wert zugeordnet. Mit Hilfe von mathematischen Verknüpfungen ergeben die Faktorwerte dann den Gesamtaufwand.
Die algorithmische Verarbeitung ist skeptisch zu betrachten. Grund hierfür sind die meist dimensionslosen Werte, die als Schätzgrundlage verwendet werden und die mathematischen Formeln zur Berechnung des Gesamtaufwands, deren Struktur auf empirischen und somit nicht immer nachvollziehbaren Daten beruhen.

Methode parametrischer Schätzgleichungen

Hier werden Korrelationsanalysen durchgeführt, welche eine Aussage darüber liefern sollen, welche Faktoren einen hohen Einfluss auf den Gesamtaufwand haben. Aus den Faktoren mit den höchsten Korrelationen zum Gesamtaufwand wird eine Gleichung gebildet, je nach Einfluss werden die Faktoren mit einem Koeffizienten gewichtet.[14] Je homogener und umfangreicher die Datenbasis ist, umso genauer sind die Ergebnisse dieser Methode.

[12] für den folgenden Absatz vgl. [Mühl01]
[13] für die folgenden Absätze vgl. [Litke95, S. 120 - 121]
[14] für den folgenden Absatz vgl. [Knöll91, S. 37 - 38]

Bei sich stark verändernden Bedingungen und somit Veränderungen der Einflüsse einzelner Faktoren, ist die Schätzung ungenau, da die Daten aus vergangenen Projekten nicht mehr aussagekräftig für die Gegenwart sind.
Diese Vorgehensweise ist auch unter dem Begriff „Faktorenverfahren" bekannt.

4.2 Vergleichsmethoden

Bei dieser Klasse von Methoden wird versucht ein Bezug zwischen vergangenen Softwareentwicklungen und der aktuellen zu schaffen.[15] Je übereinstimmender die Projekte sind, um so genauer sind die Schätzungen, die man erhält. Aufgrund der eher oberflächlichen, nur wenig in die Tiefe gehenden Vergleiche von Projekten, können diese Methoden in der Frühphase einer Softwareentwicklung eingesetzt werden, sind aber deshalb auch recht ungenau. Zu den Vergleichsmethoden gehören die Analogiemethode und die Relationsmethode.

Analogiemethode

Das aktuelle Projekt wird mit bereits abgeschlossenen Projekten verglichen.[16] Dabei wird die Übereinstimmung von Ähnlichkeitskriterien überprüft, die einen relevanten Einfluss auf den Gesamtaufwand haben. Bei der Softwareentwicklungen kommen als Ähnlichkeitskriterien z. B. Anwendungsgebiete, Programmiersprachen, Projektumfang, Personalqualität und der Schwierigkeitsgrad des Projekts in Betracht.[17] Zur Bestimmung des Aufwands werden die Projekte herangezogen, welche dem zu schätzenden am ähnlichsten sind. Sind die Projekte als gleich anzusehen wird der Aufwand komplett übernommen. Weil dies jedoch praktisch kaum der Fall sein wird, muss der Schätzer den Aufwand entsprechend anpassen. Hierin liegt auch der Nachteil dieser Methode. Sie ist zu stark abhängig von der Erfahrung, dem Wissen und dem Geschick des Schätzers und dadurch weder objektiv noch nachvollziehbar. Um diese Nachteile zu minimieren, wird bei einigen Schätzverfahren die Mustererkennungstheorie eingesetzt, um den Projektvergleich zu formalisieren.

Relationsmethode

Die Relationsmethode ist der Analogiemethode recht ähnlich.[18] Sie unterscheidet sich im wesentlichen durch den Versuch, den Einfluss des Schätzers auf das Ergebnis durch einen formalisierten Ablauf zu verringern. Dies geschieht bei vielen Verfahren dadurch, das die Ähnlichkeitskriterien als Indizes vorliegen. Wie bereits gesagt, kann ein Ähnlichkeitskriterium z. B. die Personalqualität sein, welche, auf die Softwareentwicklung bezogen, u.a. von der Programmiererfahrung abhängt. Wenn man nun folgende Indizes zu Grunde legt:

1 Jahr Erfahrung = 130
3 Jahre Erfahrung = 100

erhöht sich der Projektaufwand um 30%, wenn man anstatt Programmierern mit drei Jahren Erfahrung, nur welche mit einem Jahr Erfahrung zur Verfügung hat. Wenn man diese Indizes konsequent verwendet, beschränkt sich der Entscheidungsspielraum des Schätzers auf die Auswahl der Vergleichsprojekte und die Bewertung der Ähnlichkeitskri-

[15] für den folgenden Absatz vgl. [Mühl01]
[16] für den folgenden Absatz vgl. [Litke95, S. 119 – 120]
[17] vgl. [Knöll91, S. 36]
[18] für den folgenden Absatz vgl. [Knöll91, S. 37]

terien (in meinem Beispiel bedeutet dies, dass er festlegen muss, wie viel Jahre Erfahrung die Programmierer haben).[19]

4.3 Kennzahlenmethoden

Auch bei dieser Klasse von Methoden werden bereits abgeschlossene Projekte als Grundlage für die Aufwandsschätzung verwendet.[20] Jedoch geht es hierbei nicht um den direkten Vergleich von Projekten, sondern um die Gewinnung von aussagekräftigen Kennzahlen, die für die Aufwandsschätzung verwendet werden können. Bei den Kennzahlenmethoden kann man zwischen der Prozentsatzmethode und der Multiplikatormethode unterscheiden. Sie können in der Frühphase einer Softwareentwicklung eingesetzt werden.

Prozentsatzmethode

Hier werden die bereits abgeschlossenen Projekte dazu genutzt, die durchschnittliche Verteilung der Aufwände auf die Phasen der Softwareentwicklung zu analysieren.[21] Ausgehend von diesen Daten kann man den Gesamtaufwand auf zwei Arten prognostizieren. Erstens, indem eine Phase des Projekts abgeschlossen wird und man ausgehend von dem Aufwand dieser Phase, auf den Gesamtaufwand schließt. Hierbei wird allerdings nicht berücksichtigt, dass sich Fehler in der abgeschlossenen Phase erhöhend auf den Aufwand für die folgenden Phasen auswirken. Zweitens, man schätzt den Teilaufwand für eine Phase detailliert und berechnet davon ausgehend den Gesamtaufwand. Dabei liegt das Problem darin, dass sich Fehler bei der Berechnung des Teilaufwands multiplizieren.

Multiplikatormethode

Die zu entwickelnde Software wird in Teilprodukte zerlegt, z. B. in Module.[22] Der Gesamtaufwand ergibt sich aus der Multiplikation der Anzahl der Teilprodukte mit dem durchschnittlichen Aufwand für ein Teilprodukt. Der Aufwand je Teilprodukt wird aus vergangenen Projekten gewonnen.
Problematisch ist dabei die Bestimmung der Teilproduktanzahl, welche subjektiv erfolgt. Außerdem wird ein linearer Zusammenhang zwischen dem Gesamtaufwand und der Anzahl der Teilprodukte unterstellt, der so nicht existiert, da der Aufwand mit einer steigenden Zahl von Teilprodukten überproportional steigt. Diese Methode ist auch als „Aufwand–pro–Einheit-Methode“ bekannt.

4.4 Expertenbefragungen

Bei fehlen von Daten aus vergangenen Projekten oder bei einer nicht Vergleichbarkeit der vergangenen Projekte mit dem aktuellen (betreten von „Neuland“), ist die Expertenschätzung ein adäquates Mittel zur Aufwandsschätzung.[23] Voraussetzung hiefür ist, dass Experten vorhanden sind, die über ausreichend Erfahrung verfügen. Bei der Expertenbefragung haben sich verschiedene Ansätze herausgebildet, die im Folgenden beschrieben werden.

---

[19] vgl. [Litke95, S. 120]
[20] für den folgenden Absatz vgl. [Mühl01]
[21] für den folgenden Absatz vgl. [Litke95, S. 121-122]
[22] für die folgenden Absätze vgl. [Litke95, S. 120]
[23] für die folgenden Absätze vgl. [Mühl01]

Einzelschätzung

Ein einzelner Mitarbeiter, z. B. der Projektleiter, macht aufgrund seiner Erfahrung eine Schätzung hinsichtlich des Gesamtaufwandes. Je nach Wissen und Glück kann dies gute Ergebnisse liefern, aber auch zu vollkommenen Fehleinschätzungen führen.

Mehrfachbefragung

Hier erfolgt eine Schätzung von mehreren Experten, die das Projekt aus möglichst verschiedenen Blickwinkeln betrachten. Durch die Bildung eines arithmetischen Mittelwertes, einem Mittelwert aus Minimal- und Maximalschätzung oder eines Mittelwertes ohne Extremwerte, erhält man dann den Gesamtaufwand. Durch diese Methode verringert man meist die Streubreite und somit auch die Abweichung vom tatsächlichen Aufwand. Problematisch ist diese Methode, wenn Personen beteiligt sind, die über unzureichende Erfahrungen verfügen. Man sollte also nicht die Quantität der Befragten zu Lasten der Qualität erhöhen.

Delphi–Methode

Die Delphi – Methode stellt eine strukturierte Mehrfachbefragung dar, man unterscheidet zwischen der Standard und der Breitband Version. Der Ablauf der Standard–Delphi–Methode ist wie folgt:

1. Der Projektleiter schildert jedem Experten das Projektvorhaben und händigt ihm ein Schätzformular aus.
2. Jeder Experte füllt getrennt das Formular aus. Dabei dürfen Fragen lediglich mit dem Projektleiter besprochen werden. Eine Diskussion zwischen den Experten ist nicht gestattet.
3. Projektleiter analysiert die Angaben. Falls Schätzwerte eines Paketes stark voneinander abweichen, werden diese mit Kommentar auf einem neuen Formular erfasst.
4. Das neue Formular wird erneut zur selbständigen Überarbeitung an die Experten gereicht.
5. Die Schritte 2-4 werden so lange wiederholt, bis die gewünschte Annäherung der Ergebnisse erreicht ist oder der Projektleiter die Ergebnisse akzeptiert.
6. Der Durchschnittswert der letzten Überarbeitung der Ergebnisse aller Aufgabenpakete stellt das endgültige Schätzergebnis dar.

Bei der Breitband–Delphi–Methode erfolgt vor der Wiederholung der Schritte 2 bis 4 eine Diskussion. Der Vorteil dabei ist, dass die verschiedenen Argumente ausgetauscht werden können, nachteilig ist, dass die Anonymität dadurch nicht immer gewahrt werden kann.

Schätzklausur

Bei der Schätzklausur handelt es sich ebenfalls um eine streng systematisierte Vorgehensweise. Sie enthält jedoch gruppendynamische Aspekte, es wird also nicht anonym, sondern in einer Gruppe geschätzt. Die Schätzklausur ist gegliedert in die Vorbereitung, Durchführung und Nachbereitung. Im Folgenden werden die wichtigsten Aufgaben der drei Abschnitte aufgelistet:

Vorbereitung

- Festlegung der Größen der zu schätzenden Projektparameter
- Definition der Projektumgebung

	• Protokollwesen der Schätzklausur
	• Entwurf eines Schätzformulars
Durchführung	• Durchführung der Schätzung (evtl. nur einiger Arbeitspakete und dann Hochrechnung des Gesamtaufwands)
	• Bei großen Differenzen der Einzelschätzwerte erfolgt eine Diskussion, mit Pro und Contra
Nachbereitung	• Erstellung einer groben Projektplanung, zum Nachweis der Machbarkeit des Projekts

Durch die Diskussionen steigt das Verständnis der Experten für das Projekt und die Schätzung lässt sich nach aussen besser vertreten. Allerdings besteht die Gefahr, dass selbstbewusste Experten die anderen dominieren und somit die Ergebnisse verfälschen.

5. Ausgewählte Schätzverfahren

Es gibt mehrere Möglichkeiten Schätzverfahren zu klassifizieren. Eine solche Klassifikation könnte auf Grundlage der verwendeten Methoden, berücksichtigten Einflussfaktoren oder der genutzten Umfangsmetrik durchgeführt werden. Da die Verfahren allerdings oftmals mehrere Methoden verwenden und verschiedene Einflussfaktoren berücksichtigen, wäre eine Klassifikation aufgrund der verwendeten Umfangsmetrik am sinnvollsten. Jedoch ist es nicht Ziel dieses Kapitels alle existierenden Verfahren aufzulisten und zu kategorisieren, sondern einige wenige vorzustellen, um sie im nächsten Kapitel bewerten zu können.

5.1 Verfahren nach Surböck

Dieses Verfahren stützt sich auf die Gewichtungsmethode und Prozentsatzmethode.[24] Es kann in einer frühen Phase der Softwareentwicklung eingesetzt werden. Aus dem Systemplanungs- und Orientierungsaufwand wird auf den Gesamtaufwand geschlossen. Dabei werden sowohl die Komplexität des Systems, als auch die Vertrautheit der Entwickler berücksichtigt. Aus insgesamt 25 Fragen ergeben sich folgende Aufwände und Koeffizienten:[25]

Aufwand für Orientierung	to
Rohaufwand für Systemplanung	ts
Komplexitätskoeffizient	gk
Vertrautheitskoeffizient	ga

Daraus wird der Gesamtaufwand wie folgt berechnet:

[24] für die folgenden Absätze vgl. [Litke96, S. 38 - 40]
[25] siehe Anhang A – Verfahren nach Surböck

[26]

Systemplanungsaufwand (TS) in PT:	TS = ts • (1 + ga / 100 + gk / 100)
Orientierungsaufwand (TO) in PT:	TO = to • (1 + ga /100)
Gesamtaufwand (TT) in PT:	TT = TO + 2.35 • TS

Das Verfahren nach Surböck kann nicht für jedes Unternehmen oder Softwareprojekt unverändert übernommen werden. Es muss vielmehr geprüft werden, welche Faktoren im Unternehmen bekannt sind und sich wiederfinden lassen bzw. an welchen Parametern Veränderungen vorgenommen werden müssen.

5.2 Bottom–up-Verfahren nach Aron

Bei diesem Verfahren wird die Gesamtaufgabe zunächst in Teilaufgaben zergliedert und anschließend die schwierigste Teilaufgabe als Stichprobe ausgewählt.[27] Für diese Stichprobe wird der Aufwand geschätzt und davon ausgehend der Gesamtaufwand bestimmt. Somit basiert das Verfahren nach Aron auf der Multiplikator- und der Prozentsatzmethode. Bei der Schätzung der Stichprobe müssen folgende sieben Teilschritte monatlich wiederholt werden:[28]

1. Schätzung der Anzahl Anweisungen
2. Schätzung der Programmschwierigkeit (nach Anzahl der Schnittstellen) und Projektdauer (um den Einfluss der Lernkurve auf die Produktivität zu erfassen)
3. Bestimmung der Personenmonate für die Programmierphase
4. Anpassung der Schätzung bei Programmierung in einer höheren Programmiersprache
5. Bestimmung der notwendigen Personenmonate für die gesamte Softwareentwicklung
6. Überprüfen der Ergebnisse unter den Gesichtspunkten der Fähigkeit des Entwicklungspersonals und des Innovationsgrades der Entwicklungsaufgabe
7. Umsetzen der Kalkulationsergebnisse in einen Softwareentwicklungsplan (Übertragung der ermittelten Werte in Netzpläne und Aktivitätsdiagramme)

Auch hier müssen die Faktoren, die den einzelnen Berechnungen zugrunde liegen, kritisch betrachtet und gegebenenfalls angepasst werden. Positiv hervorzuheben ist jedoch, dass der Effekt der Lernkurve in diesem Verfahren berücksichtigt wird.

5.3 Produktivitätsverfahren von Walston und Felix

Hierbei handelt es sich um eine Weiterentwicklung der Multiplikatormethode und einer Einbeziehung der Analogiemethode oder des Bottom – up - Verfahrens[29]. Bei der Schätzung erfolgt eine Trennung von Mengen- und Wertgerüst. Zunächst wird aus bereits abgeschlossenen Projekten die durchschnittliche Produktivität bestimmt, indem man den Quotient aus dem Leistungsumfang und dem Personaleinsatz bildet. Bei der Aufwandsschätzung für das aktuelle Projekt wird der neu zu bestimmende Leistungsumfang durch

[26] vgl. [Litke96, S. 40]
[27] für die folgenden Absätze vgl. [Litke96, S. 35 – 36]
[28] siehe Anhang B – Bottom – up – Verfahren nach Aron
[29] für die folgenden Absätze vgl. [Litke96, S. 36 – 37]

den vorher berechneten Produktivitätsgrad geteilt und man erhält so den benötigten Personaleinsatz.
Um den Produktivitätsgrad berechnen zu können wurde eine Analyse bereits abgeschlossener Softwareprojekte durchgeführt. Durch Regressionsanalyse wurden aus 68 untersuchten Einflussfaktoren 29 Variablen bestimmt, die eine hinreichend hohe Korrelation zum Produktivitätsgrad aufweisen. Die folgenden vier Teilschritte müssen durchgeführt werden:

1. Berechnung des Produktivitätsindexes I, wobei
   I = Summe von (wi • xi) für i = 1, ..., 29
   w: Gewicht der Produktivitätsvariablen
   x: Ausprägungsmultiplikator der Variablen i (+1, 0, -1)
2. Ermittlung der voraussichtlichen Produktivität
   Durch die Regressionsgerade, welche aus bereits abgeschlossenen Projekten berechnet wurde, kann der zum Produktivitätsindex I gehörige Produktivitätsgrad berechnet werden.
3. Schätzung des Leistungsumfangs der Softwareentwicklung, entweder durch Analogiemethode oder auf Basis einer repräsentativen Stichprobe (Bottom – up – Verfahren).
4. Der Personaleinsatz wird aus dem Quotienten des Leistungsumfangs (LOC) und des Produktivitätsgrades berechnet.

5.4 „Business Objectives“ Verfahren

Dieses Verfahren wurde von Howard A. Rubin entwickelte.[30] Es orientiert sich an „Business Objectives“ eines Projekts. Im Kern sind 25 Fragen zu beantworten, die an das Verfahren nach Surböck angelehnt sind. Durch Beantwortung dieser Fragen ergeben sich Ausprägungswerte. Diese werden in eine Formel eingesetzt, welche aus einer Untersuchung abgeschlossener Projekte entstanden ist. Ergebnis der Berechnung ist der geschätzte Entwicklungsaufwand. Die Datenbasis umfasst 15.000 Projekte. Zu den angewandten Algorithmen gibt es keine Aussagen, die Schwankungsbreite soll 15 % betragen.

5.5 Function-Point–Verfahren

Das Function–Point–Verfahren wurde 1979 von A. J. Albrecht entwickelt.[31] Es ist aufgrund von guten Schätzergebnissen weit verbreitet. Der Aufwand eines Projekts wird aus Sicht der Anwender geschätzt. Dabei werden je nach Menge und Komplexität der zu entwickelnden Funktionen, eine bestimmte Anzahl Function Points ermittelt, die dann mittels einer Funktionskurve (Produktivitätstabelle) in Personenmonate umgerechnet werden. Die Vorgehensweise bei der Bestimmung des Projektaufwands ist:

1. Function Points zählen
   1.1. Zähltyp festlegen
   1.2. Umfang der Zählung und Systemgrenzen festlegen
   1.3. Ungewichtete Function Points zählen
   1.4. Einflussfaktor ermitteln
   1.5. Gewichtete Fucntion Points errechnen
2. Aufwand aus gewichteten Function Points ableiten

[30] für die folgenden Absätze vgl. [Litke96, S. 40 – 44]
[31] für die folgenden Absätze vgl. [Bund00, S. 188 – 202]

Zähltyp festlegen

Hierbei wird zwischen einem Neuentwicklungs- und einem Weiterentwicklungsprojekt unterschieden. Bei einem Neuentwicklungsprojekt werden alle Function Points gezählt (da die gesamte Funktionalität hinzugefügt werden muss). Bei einem Weiterentwicklungsprojekt werden nur die geänderten, gelöschten und hinzugefügten Funktionen berücksichtigt.

Umfang der Zählung und Systemgrenzen festlegen

Sinn dieser Teilaufgabe ist es, dass zu schätzende Projekt von externen Anwendungen abzugrenzen. Da das gesamte Verfahren aus Sicht des Benutzers konzipiert ist, gilt dies auch für die Festlegung der Systemgrenzen. Es werden im weiteren nur Funktionen berücksichtigt, die innerhalb dieser Systemgrenzen liegen.

Ungewichtete Function Points zählen

Bei der Zählung werden fünf Funktionstypen unterschieden:

- Interne Datenbestände
- Externe Schnittstellen
- Externe Eingabedaten
- Externe Ausgabedaten
- Externe Abfragen

Jeder externen Schnittstelle wird aufgrund ihrer Komplexität eine bestimmte Anzahl ungewichteter Function Points zugeordnet (aus einer fest definierten Tabelle).[32] Die Summe dieser Function Points repräsentiert den, für die Implementierung der externen Schnittstellen benötigten Aufwand. Dies gilt analog für alle Funktionstypen.

Einflussfaktor ermitteln

Die Ermittlung des Einflussfaktors basiert auf 14 allgemeinen Faktoren, die auf einer Skala von 0 (kein Einfluss) bis 5 (starker Einfluss) bewertet werden müssen. Für diese Bewertungen existieren genaue Definition. Die allgemeinen Faktoren sind Datenkommunikation, Verteilte Verarbeitung, Leistungsfähigkeit, Begrenzte Kapazität, Transaktionsrate, Interaktive Dateneingabe, Benutzerfreundlichkeit, Interaktive Änderung, Komplexe Verarbeitung, Wiederverwendbarkeit, Installationshilfen, Betriebshilfen, Mehrfachinstallation und Änderungsfreundlichkeit.
Die Summe aus der Bewertung der 14 allgemeinen Faktoren ergibt den Gesamtfaktor. Aus diesem wird der Einflussfaktor wie folgt berechnet:

$$\text{Einflussfaktor} = (\text{Gesamtfaktor} \bullet 0{,}01) + 0{,}65$$ [33]

Somit ergibt sich, dass der Wert des Einflussfaktors zwischen 0,65 und 1,35 liegt. Durch vergangene Projekte hat sich gezeigt, dass ein Einflussfaktor von 0,95 bis 1,1 typisch für europäische Verhältnisse ist.

[32] siehe Anhang C – Function – Point - Verfahren
[33] vgl. [Bund00, S. 199]

Gewichtete Function Points errechnen

Bei der Berechnung wird zwischen Neuentwicklungsprojekt, Weiterentwicklungsprojekt und Anwendungssystem unterschieden. Bei allen drei Berechnungen ergeben sich die gewichteten Function Points jedoch im Kern aus der Multiplikation von den ungewichteten Function Points und dem Einflussfaktor.

Aufwand aus gewichteten Function Points ableiten

Mittels einer definierten Funktionskurve (Produktivitätstabelle) können nun die gewichteten Function Points in Personenmonate umgerechnet werden und man erhält somit den geschätzten Aufwand.[32]

5.6 Data–Point–Verfahren

Dieses Verfahren basiert nicht auf LOC oder Function Points, sondern, als Antwort auf das objektorientierte Programmieren, auf Data Points.[34] Dies bedeutet, dass der Umfang einer Software nicht an der Anzahl der Funktionen, sondern aufgrund der betroffenen Objekte und der Summe der darin enthaltenen Datenelemente gemessen wird. Demnach stehen hier die Datenobjekte, also die Informationsentitäten und Nachrichten im Mittelpunkt, anstatt der Geschäftsvorfälle. Unter den Informationsentitäten sind logische Sätze und Tabellen der Zieldatenbank, auf die das System zugreifen soll, zu verstehen. Nachrichten sind Bildschirmmasken, Berichte, Datenübergaben und Telegramme an andere Systeme. Informationsentitäten werden durch die Datenanalyse und Nachrichten aus der Kommunikationsanalyse gewonnen, es muss eine vollständige Liste von ihnen vorliegen. Der Vorteil liegt darin, dass Daten- und Kommunikationsanalyse eher durchzuführen sind als eine Funktionsanalyse, gerade bei einem objektorientierten Vorgehensmodell.
Die Informationsobjekte werden in einer Tabelle mit folgendem Aufbau gelistet:[35]

Name, Anzahl Attribute, Anzahl Keys, Integrationsgrad, Nutzung, Änderung in %

Die Nachrichten werden mit folgenden Spalten aufgelistet:[35]

Name, Anzahl Felder, Anzahl Sichten, Komplexitätsgrad, Nutzung, Änderung in %

Nun wird die Summe der Data Points aller Informationsobjekte und Nachrichten gebildet und anschließend mit dem Qualitätsfaktor und dem Einflussfaktor multipliziert.

Umfang in DP = Summe DP aller Objekte • Qualitätsfaktor • Einflussfaktor

[36]

Der Qualitätsfaktor liegt zwischen 0,5 und 1,5. Er hängt ab von den acht Qualitätsmerkmalen Zuverlässigkeit, Sicherheit, Effizienz, Datenunabhängigkeit, Benutzerfreundlichkeit, Übertragbarkeit, Integrität und Wartbarkeit. Durch den Qualitätsfaktor kann die Anzahl der Data Points also um 50 % erhöht bzw. verringert werden.
Der Einflußfaktor berechnet sich ähnlich wie bei dem Function – Point – Verfahren. Allerdings sind es hier 10 allgemeine Faktoren, die auf einer Skala von 1 bis 5 bewertet werden. Die Summe dieser Werte wird von 125 abgezogen und durch 100 dividiert. So-

[34] für die folgenden Absätze vgl. [Litke96, S. 55 – 61]
[35] siehe Anhang D – Data – Point - Verfahren
[36] vgl. [Litke96, S. 60]

mit liegt der Einflussfaktor im Bereich von 0,75 und 1,15. Die 10 allgemeinen Faktoren sind Projektverteilung, Projekterfahrung, Projektkenntnisse, Projektautomation, Rechenbedingungen, Projektunterstützung, Qualitätssicherung, Spezifikations-formalismen, Programmiersprache und Testautomation.
Mittels einer Funktionskurve werden die ermittelten Data Points in Personenmonate umgerechnet.[35]

5.7 Object–Point–Verfahren nach Sneed

Hierbei handelt es sich um ein Metaverfahren, welches ein allgemeines, umfassendes Gerüst zur Entwicklung operativer Schätzverfahren für gegebene Vorgehensmodelle liefert.[37] Spezielle Verfahren auf Grundlage dieses Metaverfahrens stammen u.a. von der Software AG, Henderson – Sellers und Hateras Software.[38] Außerdem existiert ein von Harry Sneed entwickeltes Verfahren, welches im Folgenden vorgestellt werden soll.
Sneed definiert drei Subsysteme, auf denen sein Verfahren basiert.[39] Dies sind das Objektmodel, das Kommunikationsmodel und das Prozessmodel. Auf dieser Grundlage werden die Teilaufwände (Kodierungsaufwand, Integrationsaufwand, Testaufwand) geschätzt.

Berechnung Class Points

Ausgehend von dem Objektmodell wird der Kodierungsaufwand geschätzt, wobei für jede Klasse die Anzahl der Attribute, Relationen, Methoden und der Neuheitsgrad bestimmt wird. Eingesetzt in die folgende Formel und für alle Klassen aufsummiert, ergibt sich daraus die Anzahl der Class Points.

Class Points = ((Attribute) + (Relationen • 2) + (Methoden • 3)) • Neuheitsgrad	[40]

Berechnung Message Points

Für die Berechnung des Integrationsaufwands wird das Kommunikationsmodel verwendet und als Metrik die Message Points. Für jede Nachricht wird die Anzahl der Parameter, der Quellen und der Ziele bestimmt, außerdem die Komplexität (niedrig = 0.75, normal = 1, hoch = 1.25) und der Neuheitsgrad. Diese Werte werden in die folgende Gleichung eingesetzt und über alle Nachrichten aufsummiert

Message Points = ((Parameter) + (Quellen • 2) + (Ziele • 2)) • Komplexität • Neuheitsgrad	[40]

Berechnung Process Points

Hier wird das Prozessmodel, auf dessen Grundlage der Testaufwand berechnet werden soll, verwendet. Für jeden Prozess wird der Typ (multiplier = 6, batch = 2, online = 4, realtime = 8), die Anzahl der Varianten und der Komplexitätsgrad (siehe Berechnung Message Points) bestimmt. Die erhaltenen Werte werden in die nachfolgende Gleichung eingesetzt und über alle Prozesse aufsummiert.

---

[37] vgl. [Litke96, S. 61]
[38] für diesen und den nächsten Satz vgl. [Bund00, S. 211]
[39] für die folgenden Absätze vgl. [Henr97]
[40] vgl. [Henr97]

Process Points = (Prozesstyp + Varianten) • Komplexität	[40]

Berechnung ungewichteter Object Points

Die ungewichteten Object Points resultieren aus der Summe der Class Points, Message Points und Process Points.

Object Points = Class Points + Message Points + Process Points	[40]

Berechnung gewichteter Object Points

Hier werden zusätzliche Einflüsse auf den Aufwand durch den Qualitätsfaktor und den Projektfaktor berücksichtigt. Der Qualitätsfaktor beinhaltet z. B. Zuverlässig-keit, Effizienz und Portabilität. Der Projektfaktor berücksichtigt Projektattribute wie z. B. den technischen Support, die Zuverlässigkeit des Netzwerks und die eingesetzten Methoden.

gewichtete Object Points = Object Points • Qualitätsfaktor • Projektfaktor	[40]

Nach Sneed entspricht ein Object Point einem Aufwand von 0,25 Personentagen.

5.8 COCOMO–Verfahren

Das Akronym COCOMO steht für Constructive Cost Model.[41] Bei diesem Verfahren existieren verschiedene Versionen mit zunehmendem Detaillierungsgrad, das Basic COCOMO, Intermediate COCOMO und Detailed COCOMO, sie werden im Folgenden beschrieben.

Basic COCOMO

Dieses Modell soll frühzeitige Schätzungen ermöglichen. Als Input werden nur die KDSI benötigt, allerdings erhält man dadurch nur eine grobe Schätzung. Bei den Schätzgleichungen unterscheidet man zwischen drei Arten von Projekten:

[42]

Organic	unabhängig relativ kleine Teams in vertrauter Umgebung stabile Entwicklungsumgebung geringer Zeitdruck	Aufwand [PM] = $2{,}4 \bullet KDSI^{1{,}05}$
Semidetached	abhängig, eingebettet Systemumgebung verändert sich stark hoher Termindruck	Aufwand [PM] = $3{,}0 \bullet KDSI^{1{,}12}$
Embedded	halb unabhängig alles zwischen Organic und Embedded	Aufwand [PM] = $3{,}6 \bullet KDSI^{1{,}2}$

[41] für die folgenden Absätze vgl. [Litke96, S. 142 – 173]
[42] vgl. [Litke96, S. 172]

Um den Aufwand eines Projektes zu schätzen, muss man die Art des Projekts anhand der, in der Tabelle stehenden Kriterien festlegen. Anschließend bestimmt man die KDSI und setzt sie in die entsprechende Gleichung ein. Dadurch erhält man den Gesamtaufwand in Personenmonaten (PM).

Intermediate COCOMO

Beim Intermediate COCOMO handelt es sich um eine kompatible Erweiterung des Basic COCOMO. Diese Zwischenstufe soll in einer späteren Phase des Projekts eingesetzt werden, da es mehr Input benötigt, dafür liefert es aber auch genauere Schätzungen. Der wesentliche Unterschied besteht darin, dass die Aufwandswerte mit einem Einflussfaktor multipliziert werden, der sich aus der Bewertung von 15 Kostenfaktoren ergibt. Jeden Kostenfaktor ordnet man einer Kategorie zu (sehr niedrig, niedrig, nominell, hoch, sehr hoch, extrem hoch), wobei nicht immer alle Kategorien zur Auswahl stehen und ließt dann aus einer Tabelle den entsprechenden Multiplikatorwert für jeden Kostenfaktor ab. Anschließend wird das Produkt aus allen Multiplikatorwerten gebildet und man erhält so den Einflussfaktor. Die 15 Kostenfaktoren können in 4 Kategorien eingeordnet werden:

- **Produktattribute** Software – Zuverläsigkeit, Datenbankgröße, Produktkomplexität
- **Computerattribute** Exekutionsbeschränkungen, Hauptspeicherbeschränkungen. Unbeständigkeit der virtuellen Maschine, Turnaroundzeit des Rechners
- **Personalattribute** Analysator – Fähigkeiten, Programmier – Fähigkeiten, Anwendung – Erfahrung, Erfahrung mit der virtuellen Maschine, Erfahrung mit der Programmiersprache
- **Projektattribute** moderne Programmierpraktiken, Verwendung von Software – Werkzeugen, erforderlicher Entwicklungs - Zeitrahmen

Um den Aufwand in Personenmonaten zu berechnen, müssen die KDSI und der Einflussfaktor in folgenden Gleichungen, je nach Projektart eingesetzt werden:

42

Organic	Aufwand [PM] = 3,2 • KDSI ^{1,05} • Einfluss
Semidetached	Aufwand [PM] = 3,0 • KDSI ^{1,12} • Einfluss
Embedded	Aufwand [PM] = 2,8 • KDSI ^{1,2} • Einfluss

Detailed COCOMO

Um eine noch genauere Schätzung zu erhalten, wird das Software – Produkt in drei Ebenen gegliedert, die System-, die Subsystem- und die Modulebene. Außerdem finden phasensensitive Einflussfaktoren Einsatz, welche für jede Phase die Auswirkungen der Kostenfaktoren widerspiegeln.

6. Bewertung von Schätzverfahren

6.1 Anforderungen an Schätzverfahren

Die Bewertungskriterien für Schätzverfahren lassen sich in drei Gruppen einteilen.[43] Diese sind Ergebnisqualität, Projektsteuerung und Benutzerfreundlichkeit.

Ergebnisqualität

Unter diesem Begriff wird zusammengefasst:

- **Genauigkeit** Das Ergebnis der Schätzung sollte möglichst nah am tatsächlichen Aufwand liegen.
- **Nachvollziehbarkeit** Jeder Prüfer der Schätzung muss wissen, wie man zu dem Ergebnis kommt.
- **Bewertbarkeit** Das Ergebnis muss in Geldeinheiten umzurechen sein.
- **Einflussabdeckung** Es sind nur Einflussfaktoren relevant, die quantitativ oder qualitativ beurteilt werden können.
- **Parameterzahl** Es sind Parameter zu vermeiden, die keinen nennenswerten Einfluss auf das Ergebnis haben.
- **Objektivität** Dazu müssen Kriterien vermieden werden, die auf subjektiven Einschätzungen beruhen.
- **Stabilität** Schätzungen mit gleichen Eingabedaten müssen dieselben Resultate erzielen.
- **Fehlerlokalisierung** Ist die Eigenschaft eines Verfahrens eine falsche Bewertung eines Einflusses zu erkennen und zu nennen.
- **Anpassung** Das Verfahren sollte sich an jede Entwicklungsumgebung und jedes Entwicklungsprojekt anpassen können.
- **Adaptivität** Verfahren muss selbständig auf veränderte Bedingungen reagieren (Lernfähigkeit).

Projektsteuerung

Setzt sich zusammen aus den Kriterien:

- **Frühzeitigkeit** Anwendbarkeit des Verfahrens auch in frühen Phasen des Projekts, wenn nur wenige Informationen vorhanden sind.
- **Strukturierung** Kalkulationsergebnis sollte bis auf die Ebene überschaubarer Einzelaktivitäten strukturiert werden.
- **Iterativiät** Möglichkeit das Verfahren während des Entwicklungsprozesses mehrmals einzusetzen und dabei Hinweise auf die Lernfähigkeit zu erhalten.
- **Sensitivitäts-analysen** Verfahren sollte Analysen über den Einfluss von Faktoren zulassen.

[43] für die folgenden Absätze vgl. [Bund00, S. 117 – 120]

Benutzerfreundlichkeit

Wird bestimmt durch:

- **Einsetzbarkeit** Das Verfahren sollte ohne umfangreiche Vorarbeiten einsetzbar sein.
- **Erlernbarkeit** Es sollte leicht erlernbar und beherrschbar sein.
- **Zeitaufwand** Der Zeitaufwand sollte im Verhältnis zur Qualität des Ergebnisses stehen.
- **Rechnerunterstützung** Es soll die Möglichkeit bestehen das Verfahren ohne Komplikationen rechnergestützt einsetzen zu können.
- **Transparenz** Es muss klar sein, wie das Kalkulationsergebnis zustande gekommen ist.

6.2 Übersicht der ausgewählten Schätzverfahren

Verfahren	Metrik Methode EF	EQ	PS	BF	Benötigte Informationen
Verfahren nach Surböck	PM G, P 1, 4	- -	-	+ +	Datenstruktur besteht aus den Antworten auf die 25 Fragen
Bottom up Verfahren nach Aron	KDSI M, P 1, 2, 3, 4	-	0	0	Komponentenanzahl der Stichprobe und des Restsystems durchschnittliche Komponentengröße Programmschwierigkeit und Projektdauer
Produktivitäts-verfahren	LOC M, A 1, 4	-	0	-	Gewichtung der Produktivitätsvariablen Ausprägung der 29 Variablen [+1, 0, -1] Schätzung der LOC
„Business Objectives“ Verfahren	PM G, P 1, 4	- -	-	+	Datenstruktur besteht aus den Antworten auf die 25 Fragen
Function-Point-Verfahren	FP A, G 1, 2, 3, 4	+	+	0	Komplexität der 5 Funktionstypen [low, average, high] Bewertung der 14 Einflussfaktoren [0..5]
Object-Point- Verfahren nach Sneed	OP A, G 1, 2, 3, 4	0	+	0	Für jede Klasse die Anzahl der Attribute, Relationen, Methoden und der Neuheitsgrad Für jede Nachricht die Anzahl der Parameter, Quellen, Ziele, der Komplexitätsgrad und der Neuheitsgrad Für jeden Prozess den Prozesstyp, die An-zahl der Varianten und den Komplexitätsgrad Bewertete Qualitätsfaktoren Bewertete Projektfaktoren

Data-Point-Verfahren	DP A, G 1, 2, 4	0	+	0	Informationsobjekte: Name, Anzahl Attribute, Anzahl Keys, Integrationsgrad, Nutzung, Änderung in % Nachrichtenobjekte: Name, Anzahl Felder, Anzahl Sichten, Komplexitätsgrad, Nutzung, Änderung in % Bewertete 10 Einflussfaktoren [1..5] Bewertete 8 Qualitätsfaktoren
COCOMO (Intermed.)	KDSI G, Ps 1, 2, 3, 4	+	0	0	Art des Projekts Anzahl KDSI Bewertung der 15 Einflussfaktoren (nur bei Intermediate COCOMO)

Legende			
++ = sehr gut	+ = gut	0 = befriedigend	- = mangelhaft - - = schlecht
Metrik = Umfangsmetrik (teilweise Aufwandsmetrik) Methode = zugrundeliegende Schätzmethoden EF = berücksichtigte Einflussfaktoren		G = Gewichtungsmethode Ps = parametrische Schätzgleichung A = Analogiemethode R = Relationsmethode P = Prozentsatzmethode M = Multiplikatormethode	1 = Quantität 2 = Qualität 3 = Projektdauer 4 = Produktivität EQ = Ergebnisqualität PS = Projektsteuerung BF = Benutzerfreundl.

7. Zusammenfassung und Ausblick

Die Softwareentwicklungskosten werden durch die Personalkosten bestimmt, welche vom erforderlichen Aufwand abhängen. Der Aufwand ist wiederum abhängig von dem Umfang der Software, der Qualität, der Projektdauer und der Produktivität. Für die Bestimmung des Umfangs existieren mehrere Metriken, wobei Lines of Code und Function Points die am häufigsten verwendeten sind. Die Methoden der Aufwandsschätzung kann man in Algorithmische Methoden, Vergleichsmethoden und Kennzahlenmethoden einteilen, zusätzlich existiert die Expertenbefragung. Spezielle Verfahren der Aufwandsschätzung werden charakterisiert durch die zugrundeliegenden Methoden, der verwendeten Metrik, der berücksichtigten Einflussfaktoren und der zur Schätzung benötigten Informationen.
Trotz des umfangreichen Angebots an Verfahren mit z.T. guten Schätzwerten, unterscheiden sich die tatsächlichen Kosten von den prognostizierten teilweise erheblich. Dies liegt am unzureichenden Wissen einiger Projektleiter über die Aufwandsschätzung und am Einsatz von Verfahren, welche nur wenige Informationen benötigen, schnell durchgeführt sind, aber schlechte Ergebnisse liefern (z. B. alleinige Anwendung der Analogiemethode). Die Ergebnisse der Schätzverfahren sind demnach stets kritisch zu betrachten.
Bei zukünftigen Projekten sollten mehrere Verfahren berücksichtigt und auf die jeweiligen Projektbedingungen angepasst werden. Verfahren, bei denen Einschätzungen von Experten einen relativ großen Einfluss haben, sollten möglichst nicht eingesetzt werden. Es ist mit einer zunehmenden Automatisierung von Schätzungen zu rechnen, wodurch der Schätzaufwand verringert und die Objektivität erhöht wird.

Literaturverzeichnis

[Bund00] Manfred Bundschuh: Aufwandschätzung von IT – Projekten.
Bonn, 2000

[HeMeFr92] W. Hesse, G. Merbeth, R. Frölich: Software-Entwicklung – Vorgehensmodelle, Projektführung, Produktverwaltung.
Handbuch der Informatik, Band 5.3, Oldenbourg, 1992

[Henr97] Andreas Henrich: Repository Based Software Cost Estimation.
Universität Siegen, 1997
http://citeseer.nj.nec.com/henrich97repository.html,
Abruf am 2003-01-10

[Knöll91] Heinz–Dieter Knöll: Aufwandsschätzung von Software – Projekten in der Praxis: Methoden, Werkzeugeinsatz, Fallbeispiele.
Mannheim – Wien – Zürich, 1991

[LeFa90] Hareton Leung, Zhang Fan: Software Cost Estimation.
The Hong Kong Polytechnic University, 1990
http://citeseer.nj.nec.com/488252.html,
Abruf am 2002-12-23

[Litke95] Hans-Dieter Litke: Projektmanagement: Methoden, Techniken, Verhaltensweisen.
3. Auflage, München - Wien, 1995

[Litke96] Hans-Dieter Litke: DV – Projektmanagement: Zeit und Kosten richtig einschätzen.
München – Wien, 1996

[Mühl01] Andreas Mühlhausen: Überblick: Methoden der Aufwandsschätzung.
Fraunhofer - Institut für Software- und Systemtechnik ISST, 2001
http://www.visek.de/?4768,
Abruf am 2003-01-10

[Noth01} Thomas Noth: Aufwandsschätzung von IV – Projekten.
In: Peter Mertens, Andrea Back (Hrsg.):
Lexikon der Wirtschaftsinformatik.
4. Auflage, Berlin u.a., 2001, S. 54 - 56

Abkürzungsverzeichnis

LOC	Lines of Code
DP	Data Point
FP	Function Point
OP	Object Point
COCOMO	Constructive Cost Model
KDSI	Kilos Delivered Source Instructions
PM	Personenmonat
PT	Personentag

Abbildungsverzeichnis

Abbildung 3.1 / 1 Teufelsquadrat 6

Anhang A – Verfahren nach Surböck

Im Folgenden sind die Fragen und ihre Bewertung aufgelistet:[44]

Orientierung

1. Wieviele Abteilungen sind beteiligt?	• 2
2. Wieviele Filialen, Geschäftsstellen sind durch das neue System betroffen?	• 6
3. Wieviele 100 Mitarbeiter sind durch das System betroffen (aufgerundet)?	
Aufwand für Orientierung	to

Systemplanung

1. Art des Systems	batch 0 online 5 realtime 10
2. Komplexität der Verarbeitung	
- Daten kombinieren, verdichten, umordnen, löschen	einfach 0 mittel 2 kompliziert 3
- Daten aufsuchen (Indextabellen, Adressketten etc.)	einfach 0 mittel 3 kompliziert 5
- Prüfung auf Formate, Plausibilität, Fehlerroutinen	einfach 0 mittel 3 kompliziert 4
- Logik (Rechnen, Vergleiche)	einfach 0 mittel 2 kompliziert 4
- Ansprüche an Optimierung und Sicherung (Checkpoints)	einfach 0 mittel 2 kompliziert 3
3. Behandlung von Ausnahmen	manuell 0 automatisch 10
4. Systembelastung	unkritisch 0 kritisch 5
5. Zahl von Subsystemen	• 5
6. Zahl der Eingaben	• 1.5
7. Zahl der Ausgaben	• 1.5
8. Zahl der Abfragen	• 2
9. Zahl der verwendeten Datenbestände	• 3.5
Rohaufwand für Systemplanung	ts

[44] für dieses Kapitel vgl. [Litke96, S. 38 – 40] ts

Komplexität

1. Systemverfügbarkeit	normal 0 wichtig 4 kritisch 8
2. Backupverfahren	manuell0 automatisch 6
3. Werden bestehende Arbeitsverfahren automatisiert?	ja 0 nein 10
4. existiert bereits ein DV – system	ja 0 nein 6
5. Ist das zu entwickelnde System das erste seiner Art?	ja 10 nein 2
6. Werden neue DV – Verfahren eingesetz?	ja 10 nein 0
7. Wird ein Verbundsystem oder ein Netzwerk entwickelt?	ja 10 nein 0
8. Ist eine spätere Erweiterung auf Online- oder Dialogbetrieb geplant?	ja 10 nein 0
9. geplante Hardwaregröße	klein 0 mittel 4 groß 8
Komplexitätskoeffizient	gk

Vertrautheit

Wie vertraut ist das Team mit der Aufgabenstellung?	völlig neu 30 einiges Wissen15 schon einmal gemacht 0 schon öfter gemacht -20
Haben die zukünftigen Benutzer an der Vorstudie Mitgearbeitet?	nein 20 teilweise 10 intensiv 0
Wo wird gearbeitet?	am gewohnten Platz 0 in derselben Stadt 10 woanders 20
Wird die Zusammenarbeit mit den zukünftigen Benutzern leicht sein?	ja 10 nein 0
Vertrautheitskoeffizient	ga

Bestimmung des Gesamtaufwands

Systemplanungsaufwand in MT:	TS = ts • (1 + ga / 100 + gk / 100)
Orientierungsaufwand in MT:	TO = to • (1 + ga /100)
Gesamtaufwand in MT:	Tt = TO + 2.35 • TS

Anhang B – Bottom–up–Verfahren nach Aron

Folgende sieben Teilschritte müssen monatlich wiederholt werden:[45]

1.	Schätzung der Anzahl Anweisungen	A = K • G K = Komponentenanzahl aus der Stichprobe (detailliert) und dem Testsystem (geschätzt) G = durchschnittliche Komponentengröße (etwa 400 – 1000 Anweisungen)
2.	Schätzung der Programmschwierigkeit nach Anzahl der Schnittstellen und der Projektdauer (um den Einfluss der Lernkurve auf die Produktivität zu erfassen)	leicht = sehr wenige mittel = einige
3.	Bestimmung der Personenmonate für die Programmierphase	PM = A / P P = Produktivitätswert je nach Schierigkeitsgrad (Tabelle)
4.	Anpassung der Schätzung bei Programmierung in einer höheren Programmiersprache	PMP = 2 • PM
5.	Bestimmung der notwendigen Personenmonate für die gesamte Softwareentwicklung	PMS = 2,5 • PMP
6.	Überprüfen der Ergebnisse unter den Gesichtspunkten der Fähigkeit des Entwicklungspersonals und des Innovationsgrades der Entwicklungsaufgabe	
7.	Umsetzen der Kalkulationsergebnisse in einen Softwareentwicklungsplan (Übertragung der ermittelten Werte in Netzpläne und Aktivitätsdiagramme)	

[45] für dieses Kapitel vgl. [Litke96, S. 35 – 36]

Anhang C – Function–Point–Verfahren

Im Folgenden werden Anhaltspunkte für die Kategorisierung der fünf Funktionstypen aufgelistet und anschließend die Produktivitätstabelle:[46]

Interne Datenbestände

	low	**average**	**high**
• Anzahl unterschiedlicher Datenelemente	1 - 20	21 - 40	> 40
• Anzahl Schlüsselbegriffe / Satzarten	1	2	> 2
• Datenbestand vorhanden	ja		nein
• Implementierte Datenbestandstruktur wird verändert	nein	ja	
Function Points	**7**	**10**	**15**

Externe Schnittstellen

Tabellen:

	low	**average**	**high**
• Anzahl unterschiedlicher Datenelemente	1 - 20	21 - 40	> 40
• Anzahl Schlüsselbegriffe / Satzarten	1	2	> 2
Function Points	**5**	**7**	**10**

Read – Only – Dateien:

	low	**average**	**high**
• Anzahl unterschiedlicher Datenelemente	1 - 5	6 - 10	> 10
• Dimensionen	1	2	3
Function Points	**5**	**7**	**10**

Externe Eingabedaten

	low	**average**	**high**
• Anzahl unterschiedlicher Datenelemente	1 - 5	6 - 10	> 10

[46] für dieses Kapitel vgl. [Litke96, S. 46 – 54]

• Eingabeprüfung	formal	formal, logisch	formal, logisch, Zugriff auf DB
• Anspruch an Bedienerführung	gering	normal	hoch
Function Points	**3**	**4**	**6**

Externe Ausgabedaten

	low	**average**	**high**
• Medium	Liste	Liste	Liste, Formular
• Anzahl Spalten	1 - 6	7 - 15	> 15
• Gruppenwechsel	1	2 - 3	> 3
• Datenelemente druckaufbereiten	keine	einige	viele
Function Points	**4**	**5**	**7**

Externe Abfragen

	low	**average**	**high**
• Anzahl unterschiedliche Schlüssel	1	2	> 2
• Anspruch an Bedienerführung	gering	normal	hoch
Function Points	**3**	**4**	**6**

Function – Points – Produktivitätstabelle

Diese Tabelle wurde aufgrund der Erfahrungswerte von IBM entwickelt:

Function – Point	Personen-monate	Function – Point	Personen-monate	Function - Point	Personen-monate
150	5	900	65	2300	205
200	9	950	70	2400	217
250	13	1000	75	2500	229
300	17	1100	84	2600	242
350	21	1200	93	2700	255
400	25	1300	102	2800	269
450	29	1400	111	2900	284
500	33	1500	121	3000	299
550	37	1600	131	3100	315
600	41	1700	141	3200	331
650	45	1800	151	3300	350
700	49	1900	161	3400	369
750	53	2000	171	3500	390
800	57	2100	182	3600	415
850	61	2200	193	3700	441

Anhang D - Data–Point–Verfahren

Die Informationsobjekte werden in einer Tabelle mit folgendem Aufbau gelistet:[47]

Spalten	Bedeutung	Errechnung der Data - Points
Name	Name des Objektes	
Anz. Attrib	Anzahl der Attribute	1 DP / Attribut
Anz. Keys	Anzahl der Primär- und Sekundärschlüssel	4 DP / Key
Integr.grad	Integrationsgrad	niedrig 2 DP mittel 4 DP hoch 8 DP
Nutzung	Eingabe, Ausgabe oder Ein- / Ausgabe	bei zu schreibenden Objekten die Summe der bisherigen DP um 10 % erhöhen
Änderung in %	zu welchem Prozentsatz das Objekt zu ändern ist bei neuen Objekten = 100 %	% der bisher ermittelten DP = Anzahl DP für dieses Objekt

Die Nachrichten werden mit folgendem Aufbau gelistet:

Spalten	Bedeutung	Errechnung der Data - Points
Name	Name der Nachricht	
Anz. Felder	bei Masken alle Felder, die vom Programm oder vom Benutzer gefüllt werden bei Listen alle Kopf- und Fußfelder, alle Datenreihen und alle Gruppenwechselfelder bei Schnittstellen alle Felder	1 DP / Feld
Anz. Sichten	Anzahl der Informationsobjekte, die in der Nachricht vertreten sind	4 DP / Sicht
Kompl.grad	Komplexitätsgrad	niedrig 2 DP mittel 4 DP hoch 8 DP
Nutzung	Eingabe, Ausgabe oder Ein- / Ausgabe	bei Eingabenachrichten werden die bisherigen DP um 10 % erhöht
Änderung in %	zu welchem Prozentsatz das Objekt zu ändern ist bei neuen Objekten = 100 %	Prozentsatz mal bisher ermittelten DP = Anzahl DP für dieses Objekt

[47] für dieses Kapitel vgl. [Litke96, S. 55 – 61]

Data - Point – Produktivitätstabelle

Data - Point	Personen-monate	Data - Point	Personen-monate	Data - Point	Personen-monate
80	2	1440	44	2800	115
160	4	1520	47	2880	120
240	6	1600	50	2960	125
320	8	1680	54	3040	130
400	10	1760	58	3120	135
480	12	1840	62	3200	140
560	14	1920	66	3280	146
640	16	2000	70	3360	152
720	18	2080	74	3440	158
800	20	2160	78	3520	164
880	23	2240	82	3600	170
960	26	2320	86	3680	176
1040	29	2400	90	3760	182
1120	32	2480	95	3840	188
1200	35	2560	100	3920	194
1280	38	2640	105	4000	200
1360	41	2720	110		