6 Aufwandsschätzung in Software-Projekten

In diesem Kapitel wird die Aufwandsschätzung in Software-Projekten erläutert. Nach dem Durcharbeiten dieses Kapitels werden Sie

- die Programmlänge als wesentlichen, aufwandsbestimmenden Parameter von Softwaresystemen erkennen,
- in der Lage sein, mit Hilfe des COCOMO-Schätzmodells den Aufwand zur Erstellung eines Softwaresystems abzuschätzen,
- wissen, wie man die Besonderheiten individueller Projekte mit Hilfe von Korrektur-Parametern ausdrücken kann,
- wissen, wie man bei bekanntem Aufwand die optimale Teamgröße und die optimale Projektdauer bestimmen kann,
- den Aufwand zur Erstellung eines Programms auf der Basis der zu erstellenden Funktionen nach der Function-Point-Methode abschätzen können.

6.1 Aufwandsschätzung anhand der Programmlänge

Lesen Sie dazu das Kapitel 6.4 des Lehrbuchs.

6.2 Aufwandsschätzung anhand der Funktionalität

Motivation

Die unmittelbare Schätzung des Programmieraufwands für ein neu zu erstellendes oder zu änderndes Programm ist eine schwierige Aufgabe, da der Projektgegenstand ein eher abstrakter Sachverhalt ist. Anschauliche physikalische Größen wie die Kubatur bei einem Bauvorhaben oder das Gewicht einer mechanischen Konstruktion sind bei der Software-Erstellung seltener vorhanden.

Die Cocomo-Methode verwendet als Hilfsmittel für die Abschätzung des Programmieraufwands die Programmlänge. Aber auch diese ist zu Beginn eines Software-Vorhabens nur sehr schwer abschätzbar. Zudem handelt es sich im Wesentlichen um eine einparametrische Schätzung, so dass die Schätzgenauigkeit auch dadurch nur sehr grob ist.

Programme sollen in der Regel eine ganze Reihe von Aufgaben erfüllen, wie z.B. die Eingabe bzw. Erfassung von Daten, die Ablage und Verwaltung von Daten in einer Datenbank, oder die Erzeugung von Berichten oder grafischen Darstellungen. Jede dieser Aufgaben wird durch einzelne Funktionsmodule realisiert. Die Gesamtmenge aller Funktionen – die Funktionalität – bildet

dann den Gesamtumfang eines Programms, so dass sich der Gesamtaufwand zu dessen Erstellung aus der Summe der Einzelaufwände für die Erstellung der Funktionen ergibt.

Die Aufwandsschätzung mit Hilfe der Programmfunktionalität bietet zwei Vorteile. Erstens sind die einzelnen Funktionen für die Nutzer und auch für die Entwickler eine relativ anschauliche Sache, so dass der Aufwand für eine einzelne Funktion leichter und präziser abschätzbar ist. Zweitens wird der Gesamtaufwand additiv ermittelt, was in der Regel zu einer deutlichen Verbesserung der Schätzgenauigkeit führt.

ISO 14143

Es sind verschiedene funktionsbasierte Schätzverfahren bekannt, die sich oft nur in Details unterscheiden. Ein grundlegendes und weit verbreitetes Verfahren basiert auf „Function Points“ und soll nun näher behandelt werden. Die Ursprünge der Function-Point-Analyse liegen in den 1970er Jahren bei IBM. Die Methode wurde stetig weiterentwickelt. Die verschiedenen Varianten wurden im Jahre 2003 unter der Bezeichnung „Functional Size Measurement“ als ISO-Standard etabliert (ISO 14143 und ISO/IEC 20296).

Bei der Function-Point-Analyse (FPA) wird eine Software als ein System betrachtet, das mit seiner Umgebung Daten austauscht. Bei dieser Interaktion mit der Umgebung werden drei Arten von Daten-Interaktionselementen unterschieden: External Inputs (EI), External Outputs (EO) und External Queries (EQ). Zudem speichert die Software Daten intern in Form von Internal Logical Files (ILF).

Transaktions-elemente

External Inputs sind Eingaben an das System. Sie werden dort verarbeitet und führen zum Anlegen neuer oder zum Ändern vorhandener Daten im System. External Queries sind Abfragen vorhandener Daten, die ohne eine weitere Verarbeitung nach außen gegeben werden. Auch die External Outputs erzeugen Ausgaben an die Umgebung, erfordern aber eine Verarbeitung der vorhandenen Daten, so dass EOs komplexer sind als EQs.

Beispiel 6.1 Auftragsverwaltung

Bei einer Software für die Verwaltung von Produktionsaufträgen eines Unternehmens können neue Aufträge eingegeben oder vorhandene verändert (EI), vorhandene Aufträge angezeigt (EQ) oder aus vorhandenen Aufträgen die Maschinenauslastung berechnet (EO) werden. Die intern gespeicherten Datensätze aller Aufträge stellen wiederum ein ILF dar.

Ein zweites ILF bildet die Kundenkartei. Auch hier werden Funktionen zum Anlegen neuer Kundendatensätze (EI) oder deren Ausgabe (EO) benötigt.

Aus vielen Projekten zur Erstellung von Programmen liegen Erfahrungen über den Aufwand zur Realisierung der Transaktionsfunktionen und zum

Aufbau der internen Datensätze vor. Allerdings variiert dieser Aufwand von Projekt zu Projekt teilweise erheblich. Wesentliche Einflussfaktoren sind die verwendete Programmiersprache, aber auch Erfahrungswerte in unterschiedlichen Unternehmen oder Abteilungen.
Aus diesem Grund wird der Aufwand für die Erstellung der Funktionen nicht direkt in Programmzeilen oder in Arbeitsaufwand umgerechnet, sondern in so genannte Funktionspunkte. Es handelt sich hierbei um ganzzahlige Werte im Bereich von 3 bis 15.
Eingaben und Abfragen erfordern ungefähr den gleichen Aufwand, während Ausgaben durch die zusätzliche Verarbeitung mit etwas mehr Aufwand verbunden sind. Noch höher ist der Aufwand für die Handhabung des internen Datenbestands. Bei allen Elementen werden bei einer vollständigen Function-Point-Analyse drei Komplexitätsstufen berücksichtigt. Das Ergebnis ist eine grundlegende Tabelle, die jedem Element abhängig von der Komplexität Funktionspunkte zuordnet (siehe Tabelle 6.1).

Tabelle 6.1 Funktionspunkte der Programmfunktionen

	Komplexität		
Function Points	niedrig	mittel	hoch
External Input (EI)	3	**4**	6
External Output (EO)	4	**5**	7
External Query (EQ)	3	**4**	6
Internal Logical File (ILF)	**7**	10	15

Die Komplexität einer Funktion kann mit Hilfe der Datenelemente bestimmt werden. Stellt man sich ein ILF als eine Tabelle einer Datenbank vor, so entsprechen die Datenelemente den Spalten der Tabelle. In einer objektorientierten Datenbank werden die Datenelemente als Attribute bzw. Merkmale eines Datenobjekts bezeichnet. Je mehr Datenelemente ein ILF enthält, umso höher ist die Komplexität und dementsprechend steigt auch die Zahl der Function Points. In der Praxis wird die Komplexität nur selten berücksichtigt, sondern es wird bei den Transaktionselementen (EI, EO und EQ) eine mittlere und beim Datenelement (ILF) eine niedrige Komplexität angenommen. Zum einen entspricht diese Näherung den in der Praxis am häufigsten vorkommenden Fällen. Zum anderen ermöglicht sie eine deutliche Verringerung des Schätzaufwands.

Beispiel 6.2 Auftragsverwaltung

Für die Auftragsverwaltung wird zunächst der genaue Funktionsumfang ermittelt. Daraus können dann die Funktionspunkte bestimmt und der Gesamtwert berechnet werden. Die folgende Tabelle zeigt einige exempla-

rische Funktionen, den zugehörigen Typ und die daraus durch die oben beschriebene Näherung ermittelten Funktionspunkte.

Funktion	Typ	FP
Kundendatenbank handhaben	ILF	7
Neuen Kundendatensatz anlegen	EI	4
Vorhandenen Kundendatensatz ändern	EI	4
Kunden anzeigen (mit Filteroptionen)	EO	4
Umsatz-Zeitverlauf für einen Kunden bestimmen	EQ	5
… weitere Funktionen der Kundendatenbank	…	13
Auftragsliste handhaben	ILF	7
Neuen Auftrag anlegen	EI	4
Auftrag stornieren	EI	4
Vorhandene Aufträge (gefiltert) ausgeben	EO	4
Maschinenauslastung berechnen	EQ	5
… weitere Funktionen der Auftragsliste	…	39
Summe		**100**

Der geplante Funktionsumfang für dieses Programm umfasst also 156 Funktionspunkte.

Im nächsten Schritt muss aus den Funktionspunkten der erforderliche Arbeitsaufwand ermittelt werden. In der Regel erfolgt dies indirekt, indem die Funktionspunkte in erforderliche Programmzeilen (Lines of Code LOC) umgerechnet werden. Der Umrechnungsfaktor ist zum einen von der Programmiersprache, zum anderen aber auch von der jeweiligen Aufgabenstellung und vom Projektteam abhängig. Bei objektorientierten Programmiersprachen wie C++, C# oder Java liegen mittlere Werte bei ca. 50 Programmzeilen pro Funktionspunkt. Allerdings schwanken die dokumentierten Erfahrungswerte bis zu einen Faktor 2,5 nach unten (20 LOC/FP) und nach oben (125 LOC/FP). Aus diesem Grund sollte in jedem Unternehmen eine eigene Erfahrungssammlung in Software-Projekten erfolgen, damit die Programmlänge mit besserer Genauigkeit geschätzt werden kann.

Verschiedene Veröffentlichungen machen Vorschläge für die direkte Umrechnung der Funktionspunkte in den erforderlichen Arbeitsaufwand. Auch hier ist die Bandbreite relativ groß. Im Mittel geht man von einem Aufwand von sechs Personenmonaten (PM) pro 100 Funktionspunkten aus. Dieser Wert wiederum kann um einen Faktor 2 nach unten (3 PM/100FP) und oben (12 PM/100FP) schwanken.

Kostentreiber

Jede Abschätzung mit Hilfe von Funktionspunkten und allgemeingültigen Faktoren stellt nur eine grobe Abschätzung für ein konkretes Software-Projekt dar, da die Berechnung und die verwendeten Parameter aus der Mittelung

vieler Projekte entstanden sind. Um die Genauigkeit der Abschätzung für ein konkretes Projekt zu verbessern, wurden zahlreiche Vorschläge veröffentlicht, die eine Anpassung an individuelle Projektgegebenheiten ermöglichen. Ein bekanntes Verfahren hierfür sind so genannte Kostentreiber. Sie werden gehandhabt wie die Korrekturfaktoren bei Cocomo.
Jeder Kostentreiber wird durch einen ganzzahligen Faktor ausgedrückt, der zwischen 0 und 5 liegen kann. Die Kostentreiber werden dann gewichtet aufsummiert und zu einem Korrekturfaktor verrechnet, der zwischen 0,65 und 1,35 liegen kann. Mit diesem Faktor werden dann aus den ursprünglichen Funktionspunkten (FP) die adjustierten Funktionspunkte (AFP) berechnet.

Zusammenfassung

Das COCOMO-Modell ist eines der bekanntesten Verfahren zur Schätzung des Arbeitsaufwands in Softwaresystemen. Es basiert in erster Näherung auf der Annahme, dass der Aufwand im Wesentlichen proportional zur Programmlänge ansteigt. Diese wird in Form der Quellcode-Programmzeilen gemessen.
Zusätzliche Einflussparameter, wie z.B. im Team vorhandene Erfahrungen, technologische Anforderungen und Klarheit der Zielformulierung werden in einer verfeinerten Schätzung berücksichtigt. Sie führen dazu, dass der Aufwand etwas stärker als proportional ansteigt.
Mit Hilfe weiterer Parameter lässt sich aus dem Aufwand die optimale Projektdauer und die optimale Anzahl von Projektmitarbeitern bestimmen.
Statt der schwer zu bestimmenden Programmlänge basiert die Aufwandsschätzung mit Hilfe von Funktionspunkten auf den zu realisierenden Datenverarbeitungsfunktionen. Für die verschiedenen Funktionen zur Eingabe, Ausgabe und Abfrage von Daten sowie aus den benötigten Datenelementen werden Punkte vergeben, die aus Erfahrungen mit zurückliegenden Projekten herrühren. Aus der Gesamtzahl der Punkte kann dann direkt oder indirekt über die zu erwartende Programmlänge der Arbeitsaufwand für ein Software-Projekt bestimmt werden.

Übungsaufgaben

Aufgabe 6.1 Aufwandsschätzung für eine Geldautomaten-Software

Bei einem Hersteller von Geldautomaten soll ein neuer Automatentyp inklusive PC-basierter Steuerung entwickelt werden. Da die Steuerung bisher mit einem Mikrocontroller und eigenem Betriebssystem realisiert war, muss die Software komplett neu erstellt werden. Da ein einfacher Austausch möglich sein soll, müssen die neuen Automaten hinsichtlich der Schnittstellen vollständig kompatibel zu den alten sein. Die Hardware der Steuerung wird mit Hilfe eines fertigen CPU-Boards und einem selbst zu entwickelnden Interface-Board realisiert.
Aufgrund der Erfahrungen mit der alten Steuerung konnte die Architektur der Software vorab bestimmt werden. Sie besteht aus 4 Modulen: Benutzerinterface, Interface für Sensoren und Aktoren, Interne Datenverarbeitung und Rechnerschnittstelle. Innerhalb der Module gibt es ca. 160 Funktionen. Anhand des Anforderungsprofils konnte der Code-Umfang der einzelnen Funktionen abgeschätzt werden. Für den gesamten Code-Umfang ergab sich ein Schätzwert von 24.000 Codezeilen.
Bestimmen Sie anhand dieser Angaben den geschätzten Arbeitsaufwand zur Programmerstellung, bestehend aus Entwurf, Codierung, Test, Integration und Dokumentation.
Wie sehen die optimale Projektdauer und die optimale Teamgröße für dieses Projekt aus?

Hilfestellung: Tabelle 6.4 aus dem Lehrbuch.

Aufgabe 6.2 Funktionspunkte für eine Webseitenrealisierung

Für einen Pizza-Lieferservice soll eine Webseite erstellt werden, mit der Rezepte, Kunden und Bestellungen verwaltet werden können. Folgende Aufgabe soll die Webseite für Bestellungen erfüllen: eingeben, stornieren, anzeigen, auflisten. Des Weiteren soll für ein beliebiges Zeitintervall das Bestellaufkommen bestimmt werden. Die Daten für einen Kunden können angelegt, ausgegeben, geändert oder gelöscht werden. Zudem soll das Bestellaufkommen für einen Kunden berechnet werden. Pizza-Rezepte sollen eingegeben, angezeigt und geändert werden können. Für den Import von Rezepten sollen drei gängige Formate unterstützt werden.
Ermitteln Sie anhand dieser Aufgabenbeschreibung die Anzahl der Funktionspunkte.