

2 Einführung in die Function-Point-Analyse (FPA)

Ursprünglich einmal »nur« zur Verbesserung der Aufwandsschätzung von Softwareprojekten gedacht, findet die Function-Point-Analyse (FPA) heute Anwendung im gesamten Lebenszyklus von Software. Sie wird in Projektplanung und -management eingesetzt, aber auch im Controlling und in der kaufmännischen Bewertung von IT-Projekten und -Anwendungen. Dieses Kapitel gibt eine Übersicht über Grundprinzipien und Anwendung der Function-Point-Analyse.

Um den fachlichen Funktionsumfang einer IT-Anwendung quantitativ bestimmen zu können, zerlegt die Function-Point-Analyse diese in ihre Elementarfunktionen oder, wie es dort heißt, Elementarprozesse (*elementary processes*). Dabei kommen drei wesentliche Prinzipien zur Anwendung:

- *Anwendersicht*: Die Bewertung erfolgt aus der Sicht der Anwender eines Systems. Allgemeiner könnte man auch sagen, aus Sicht der Geschäftsprozesse. Der Begriff »Anwender« ist in der FPA nicht auf menschliche Bediener eines Systems beschränkt, sondern umfasst auch andere Systeme, die mit dem betrachteten System interagieren.
- *Atomarität*: Ein Elementarprozess ist die aus Anwendersicht kleinste sinnvolle und in sich abgeschlossene Aktivität, die mit dem System durchführbar ist.
- *Einmaligkeit*: Jeder Elementarprozess wird nur einmal gewertet, unabhängig davon, an wie vielen Stellen oder wie häufig er innerhalb der Anwendung genutzt wird. Dazu gibt es eine Reihe von Kriterien, die die Unterscheidung »ähnlicher« Elementarprozesse erlauben. Das wichtigste Kriterium ist ein Unterschied in der Verarbeitungslogik. An der Oberfläche ähnliche Elementarprozesse, die z. B. eine gemeinsame Bildschirmmaske verwenden, werden so durchaus unterscheidbar.

Die Anwendung dieser drei Prinzipien stellt die wesentlichen Merkmale der Function-Point-Analyse sicher: Eindeutigkeit und Nachvollziehbarkeit.

Die Function-Point-Analyse erlaubt aber nicht nur die Bewertung einer IT-Anwendung, sondern auch der Projekte, die zu deren Entstehung und Erweite-

rung führen. Im letzten Abschnitt dieses Kapitels beschreiben wir deshalb den Einsatz der FPA im Lebenszyklus von Softwareprojekten.

2.1 Anwendungsgebiete von A bis Z

Die Anwendungsgebiete der FPA reichen von A wie Aufwandsschätzung bis Z wie Zieldefinitionen.

Heute versteht man Function Points über die Aufwandsschätzung hinaus als generelles Maß für die fachlich-funktionale Größe eines Softwareprogramms. Die Einsatzgebiete für diese Messgröße sind damit z. B.

- die Bewertung von Projektangeboten externer Lieferanten,
- Wertbestimmung bestehender Softwareprogramme,
- Kosten- und Risikoabschätzungen von IT-Vorhaben,
- Projektcontrolling und
- Benchmarking von Projekten.

Die Herausforderung in der Aufwandsschätzung besteht darin, den Aufwand für ein Projekt möglichst früh sicher vorhersagen zu können. Möglichst früh, das heißt: Die fachlichen Anforderungen sind bekannt, mehr oder weniger detailliert, aber nicht die Art und Weise der technischen Umsetzung.

Diese Aufgabenstellung ähnelt also in etwa der, die Baukosten für ein Einfamilienhaus abzuschätzen, wobei bekannt ist, welche Familie in dem Haus wohnen soll, wie viele Erwachsene und Kinder und welche Ansprüche diese haben. Wie kann man die Baukosten vernünftig abschätzen, ohne dass schon ein detaillierter Bauplan vorliegt? Die Lösung liegt in einer Herangehensweise in zwei Schritten:

1. Wie viele Quadratmeter Wohnfläche braucht die Familie?
2. Wie viel kostet ein Quadratmeter Wohnfläche der geforderten Ausstattung und Lage?

Übertragen auf die Aufwandsschätzung eines Softwareprojekts heißt dies: Wie viele Function Points stecken hinter den fachlichen Anforderungen? Was kostet die Entwicklung eines Function Points in der geforderten Qualität und Performance?

Doch hat man erst einmal ein funktionales Leistungsmaß für die Aufwandsschätzung definiert, so liegt es natürlich nahe, auch das Ergebnis des Softwareprojekts, also das fertige Softwaresystem, mit diesem Maß zu bewerten. Damit eröffnet sich für die Function-Point-Analyse noch eine Reihe weiterer Anwendungsmöglichkeiten.

Die gelieferte Leistung eines Projekts lässt sich über Function Points definieren. Damit hat die Function-Point-Analyse im IT-Projektcontrolling eine besondere Bedeutung. Sie ist die Bezugsgröße bzw. das Leistungsmaß, das in Relation zu verbrauchten Ressourcen – Aufwand, Kosten und Zeit – gesetzt wird.

Ein zentrales Element strategischer Steuerung ist das Benchmarking. Der Begriff Benchmarking, abgeleitet aus dem englischen *benchmark* mit der Bedeutung »Referenzpunkt«, geht auf Robert C. Camp⁴ zurück. Er hat Benchmarking als Managementverfahren begründet. Kurz gefasst, lässt sich dies mit dem Satz »Lernen durch Vergleichen« beschreiben. Durch den Vergleich der eigenen Leistungsfähigkeit mit der anderer Unternehmen sollen Organisationen Verbesserungspotenziale und »*Best Practices*« erkennen. Vergleichen kann man nur, wenn man über ein gemeinsames Maß verfügt: Für Softwaresysteme und -projekte liefert dies die Function-Point-Analyse.

Ein weiteres Anwendungsgebiet liegt in Ausschreibungen und in der Auftragsvergabe von Softwareprojekten. Das klassische Vorgehen kennt zwei Alternativen: Entweder sind die funktionalen Anforderungen (auch als Pflichtenheft bezeichnet) genau bekannt und bilden damit die Ausschreibungsgrundlage oder die Erfassung und Beschreibung der funktionalen Anforderungen sind Teil des Projekts selbst.

Im ersteren Fall besteht der Nachteil darin, dass die Erstellung eines Pflichtenhefts selbst schon erheblichen Aufwand und Kosten erfordert – und spezifisches IT-Know-how, das vielleicht gerade beim Auftraggeber gar nicht vorhanden ist. Im letzteren Fall bereitet die Vergleichbarkeit der Angebote das Problem: Denn jeder Anbieter wird potenziell zu einem eigenen Pflichtenheft kommen. Ein direkter Vergleich ist damit unmöglich.

Die Verwendung von Function Points ist hier eine neue, dritte Alternative: Ein Preis in »Euro pro Function Point« ist aus Sicht der Einkäufer eine kalkulierbare Größe. Ein schöner Nebeneffekt ist dabei, dass Auftraggeber und Auftragnehmer schon zusammenkommen können, bevor die Anforderungen im Detail hundertprozentig feststehen. Function Points werden in diesem Kontext – der Vergabe von Softwareentwicklungsleistungen – also als Vergütungsbasis verwendet.

Auch ein »fertiges« Softwaresystem, eine IT-Anwendung, lässt sich mit Function Points bewerten. Dieser Wert wird z. B. für den Vergleich von Wartungs- und Betriebskosten und -aufwand verwendet. Der Function-Point-Wert spielt auch eine wichtige Rolle im Portfoliomanagement und bei Due-Diligence-Untersuchungen, also dort, wo es darauf ankommt, den betriebswirtschaftlichen Wert eines IT-Systems zu bestimmen⁵.

4. [Camp 1994]

5. Gelegentlich wird der Nutzen eines IT-Systems mit seinem Wert gleichgesetzt. Dies ist jedoch methodisch fragwürdig. Der Nutzen oder auch Nutzwert ist eine subjektive Größe, die die Tauglichkeit eines Vorhabens bestimmt (vgl. [Stickel et al. 1997], S. 488). Für eine unternehmensübergreifende Bewertung ist dagegen eher auf den Substanzwert eines Systems abzustellen (vgl. [Wöhe 1990], S. 800). Falsch ist eine Gleichsetzung des Substanzwerts sowohl mit den Projektkosten als auch mit dem Nutzwert.

2.2 Wie alles anfang

»Am Anfang war die IBM-Kurve ...«, so könnte dieses Kapitel beginnen. Welcher Informatiker kennt sie nicht: 1985 publizierte IBM⁶ auch in Deutschland die als »IBM-Kurve« bekannte Grafik, die für 54 Projekte den Zusammenhang von Aufwand und Function Points darstellte.

Bereits einige Jahre vorher hatte der IBM-Softwareexperte Allan J. Albrecht begonnen, das Problem der Aufwandsschätzung für Softwareprojekte zu lösen: Wie kann ich den Aufwand vorhersagen, wenn ich zwar weiß, was die Software am Ende leisten soll, aber noch nicht sagen kann, wie der Weg dorthin aussieht?

Folgerichtig erkannte Albrecht, dass nur eine Kennzahl, die die »Leistung« der Software quantitativ beschreiben kann, dieses Problem lösen würde. Aber wie müsste solch eine Kennzahl definiert sein? Wie ließe sie sich ermitteln?

Die Antwort lag in einem, aus Albrechts Sicht, grundsätzlichen Perspektivenwechsel: Er betrachtete ein Softwaresystem nicht mehr aus Sicht des Programmierers (der vor allem die einzelnen Bausteine – sprich: Programmieranweisungen – sieht), sondern aus der Sicht des Anwenders (der die Software nutzt und also ganz unmittelbar ihre Leistung erfährt).

Die Wahrnehmung der Methodik bei den Anwendern folgte dann der für alle neuen Technologien bekannten »Hype-Kurve« (vgl. Abb. 2–1). Die Veröffentlichung der IBM-Kurve hatte die Erwartung geweckt, nun endlich über ein einfaches und gleichzeitig zuverlässiges Verfahren zur Aufwandsschätzung zu verfügen. Anfang der 90er Jahre gab es deshalb in fast allen deutschen Großunternehmen den Versuch, das Verfahren zur Aufwandsschätzung einzusetzen. Rückblickend lässt sich eine Reihe von Gründen finden, warum die meisten dieser Versuche zum Scheitern verurteilt waren:

Da ist zunächst die Unerfahrenheit der Anwender mit der Methodik zu nennen. Die sichere und effiziente Durchführung einer Function-Point-Analyse erfordert eine gründliche Ausbildung und praktische Erfahrung. Über beides verfügten die »Pioniere« nicht. Aus dieser Zeit stammt deshalb der irrige Glaube, die Function-Point-Analyse sei aufwändig und liefere keine belastbaren Ergebnisse.

Auf der anderen Seite waren die Erwartungen an die Ergebnisse des Verfahrens unrealistisch hoch gesetzt. Heute wissen wir, dass eines der zentralen Probleme der frühen Projektaufwandsschätzung vor allem in einer lückenhaften oder unklaren Definition der fachlichen Anforderungen liegt. Dieses Problem kann aber auch mit der Function-Point-Analyse nicht gelöst werden.

Aber selbst da, wo die Anforderungen vollständig mit Function Points beschrieben waren, fehlten die für eine Aufwandsschätzung notwendigen differenzierten Erfahrungsdaten. Die in der historischen IBM-Erfahrungskurve ver-

6. [IBM 1985]

öffentlicherten Function-Point-Aufwandsrelationen erwiesen sich für die meisten Unternehmen als unzutreffend.

Kein Wunder also, dass die ersten Anwender sich zunächst enttäuscht wieder von dem Verfahren abwandten.

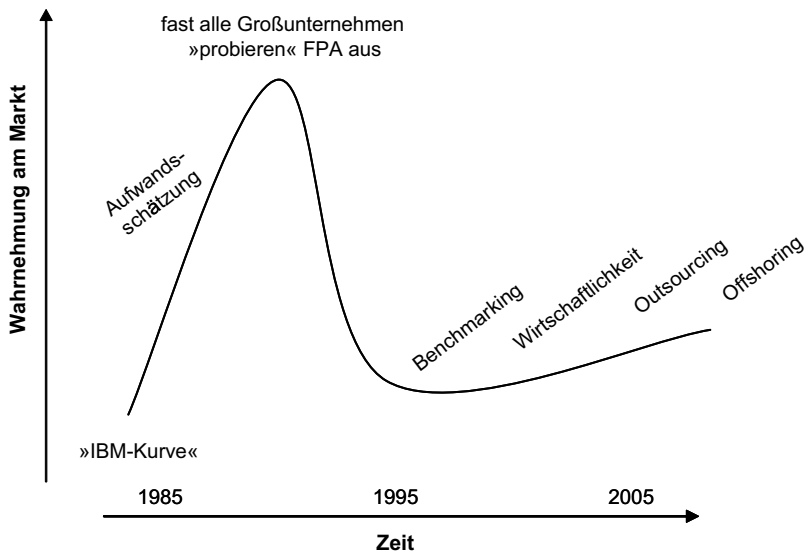


Abb. 2-1 Hype-Kurve der Function-Point-Analyse

Ab etwa Mitte der 90er Jahre gewann ein ganz anderes Motiv für die Verwendung von Function Points an Bedeutung. Bislang hatte man Leistungsvergleiche für Softwareprojekte, auch als Benchmarking bezeichnet, in erster Linie am Umfang des im Projekt erzeugten Quellcodes festgemacht. Das funktionierte so lange sehr gut, wie alle Vergleichsprojekte und -unternehmen mit der gleichen Programmiersprache oder zumindest einer Programmiersprache der gleichen Generation (wie z. B. Cobol und PL/I) arbeiteten. Mit der Verbreitung moderner Sprachen der 4. Generation und leistungsfähiger Werkzeuge und Codegeneratoren verloren Benchmarks auf der Grundlage des Quellcodeumfangs ihre Aussagekraft. So wurden Function Points als über verschiedene Programmiersprachen und Technologien hinweg vergleichbares Leistungsmaß für Softwareprojekte entdeckt.

Heute beobachten wir ein stetiges Wachstum der Bedeutung der Function-Point-Analyse. Im Zuge von Wirtschaftlichkeitsbetrachtungen wird der Function-Point-Wert von Projekten und IT-Anwendungen deren Kosten gegenübergestellt. Im Outsourcing und Offshoring werden die Angebote und Leistungen der Dienstleister mit Function Points bewertet.

Die Aufwandsschätzung ist also heute nur ein Anwendungsgebiet der FPA unter vielen. Die Motive und Triebkräfte für ihren Einsatz kommen heute eher aus kaufmännischen und Controlling-Bereichen.

2.3 Was misst die FPA?

Wenn wir über IT-Systeme sprechen, dann meinen wir in erster Linie solche Systeme, die im betrieblichen Ablauf die Durchführung von Geschäftsprozessen unterstützen. Das kann z. B. die Anlage eines Versicherungsvertrags bei einer Versicherung sein, aber auch der Check-in bei einer Fluglinie.

Ein IT-System unterstützt also die Durchführung eines Geschäftsprozesses. Ist die Durchführung mehr oder weniger vollständig durch das IT-System unterstützt, sagt man auch: Das IT-System bildet den Geschäftsprozess ab. Das heißt, der Geschäftsprozess und das IT-System werden letzten Endes gleichgesetzt. Heute kennen wir in vielen Branchen Geschäftsprozesse, die ohne entsprechende IT-Anwendungen gar nicht mehr existieren könnten.

Die Funktionalität eines IT-Systems lässt sich also daran messen, welcher Anteil eines Geschäftsprozesses unterstützt wird. Beispiel Versicherung: Erfolgt die Berechnung der Prämie bei der Annahme des Versicherungsvertrags automatisch oder manuell? Zudem macht es einen Unterschied, wie »umfangreich« ein Geschäftsprozess ist. Beispiel Check-in: Kann ein einmal zugewiesener Sitzplatz geändert werden? Oder muss die Bordkarte storniert und der Check-in wiederholt werden?

Um diese Unterschiede messbar zu machen, macht die Function-Point-Analyse sozusagen eine Anleihe bei der Geschäftsprozessanalyse: Ein Geschäftsprozess ist definiert als die strukturierte Anordnung einzelner Aktivitäten. Damit lässt sich die »Größe« eines Geschäftsprozesses an der Anzahl und Komplexität der ihn definierenden Aktivitäten festmachen und die funktionale Größe eines IT-Systems entsprechend an der Anzahl und Komplexität der abgebildeten Aktivitäten.

In der Function-Point-Analyse werden Aktivitäten »Elementarprozesse« genannt. Die funktionale Größe eines IT-Systems in Function Points ergibt sich also aus der Anzahl der Elementarprozesse. Diese werden allerdings jeweils einzeln zuvor noch mit einem Punktwert, den »*function points*« gewichtet.

Man sieht also schon: Die FPA misst nicht eine Anwendung »an sich«, sondern deren fachlichen Funktionsumfang, also die Leistung zur Unterstützung der Geschäftsprozesse. Diese Perspektive aus der Sicht der Geschäftsprozesse bezeichnet man auch als das Prinzip der *Anwendersicht*. Es ist eines der zentralen Prinzipien des Verfahrens, und seine strenge Forderung unterscheidet die FPA von den meisten anderen funktionalen Softwaremetriken.

2.4 Elementarprozesse und Datenbestände

Ein Elementarprozess ist die kleinste, aus fachlicher Sicht sinnvolle, in sich abgeschlossene Aktivität, die mit einem IT-System durchgeführt werden kann. Dieses Prinzip kann auch als *Atomaritätsprinzip* bezeichnet werden.

Ein Elementarprozess kann z. B. sein:

- die Erfassung einer Kundenadresse,
- der Ausdruck einer Rechnung,
- die Berechnung eines Tarifs,
- die Übermittlung von Abrechnungsdaten an ein Ex-/Inkassosystem,
- die Anzeige eines Kontostands usw.

Die Funktionen eines IT-Systems werden also in die kleinsten sinnvollen, in sich abgeschlossenen Einheiten zerlegt. Sinnvoll orientiert sich dabei am Standpunkt des Benutzers, also letztlich am Geschäftsprozess und nicht am Systemdesign.

Erfolgt z. B. die Erfassung einer Kundenadresse in einem System über zwei Masken (in der ersten Maske die Postadresse, in der zweiten Maske Zusatzinformationen wie Telefon usw.), so wird dies in der Regel aus Sicht des Benutzers nur einen Elementarprozess darstellen.

Aber auch der umgekehrte Fall ist denkbar: Auf einer Maske werden schon erfasste Kundendaten angezeigt und gleichzeitig kann dort eine weitere Adresse eingegeben werden. Aus Sicht des Benutzers handelt es sich um zwei Elementarprozesse: die Anzeige der gespeicherten Kundendaten und die Erfassung einer Adresse.

Das zweite wesentliche Kriterium für die Identifikation eines Elementarprozesses ist seine »Einmaligkeit«. Ein Elementarprozess gilt dann als einmalig, wenn er durch die ein- oder ausgegebenen Daten oder durch die Verarbeitungslogik unterscheidbar ist. Dieses Kriterium wird auch als *Einmaligkeitsprinzip* bezeichnet.

Im Regelfall ist die Unterscheidbarkeit eines Elementarprozesses aufgrund der eingegebenen oder ausgegebenen Daten offensichtlich. Schwieriger kann es werden, wenn verschiedene Berechnungsalgorithmen verwendet werden. Ein Beispiel wäre etwa die Berechnung eines Beitrags einer Versicherung, die, abhängig von einem Stichtag für den Vertragsabschluss, nach verschiedenen Algorithmen durchgeführt wird. Das Atomaritätsprinzip kommt analog auch bei der Bewertung der Datenbestände zur Anwendung. Ein Datenbestand ist dabei als eine für den Benutzer erkennbare logische Gruppe von Daten definiert. Wie bei den Elementarprozessen steht dabei wieder die Sicht des Benutzers bzw. Geschäftsprozesses im Vordergrund.

Für den gesamten Function-Point-Wert eines Systems spielen die Datenbestände eine nur geringe Rolle. Sie tragen zum gesamten Function-Point-Wert eines IT-Systems in der Regel zwischen etwa 10 % und 20 % bei. Aus diesem

Grund liegt der Fokus bei einer Function-Point-Analyse immer auf den Elementarprozessen.

2.5 Woher kommt der Name »Function Points«?

Zu Albrechts Zeiten diente die »Informationstechnologie« vor allem der Speicherung und Verwaltung von Daten, was ja auch in der deutschsprachigen Bezeichnung »Elektronische Datenverarbeitung« zum Ausdruck kommt. So lag es nahe, den Leistungsumfang eines IT-Systems an der Anzahl und der Komplexität der verwalteten Daten und der Anzahl der Möglichkeiten, diese Daten in das System einzugeben und von ihm wieder ausgegeben zu bekommen, zu messen. Die Grundkonstrukte der Function-Point-Analyse waren also nach Albrecht bereits »Datenbestände«, »Eingaben« und »Ausgaben«.

In der Albrecht'schen Terminologie wird der Begriff *function* dabei sowohl im Zusammenhang mit den Ein- und Ausgaben (*transactional functions*) als auch für die Datenbestände (*data functions*) verwendet. Der Begriff hat also nichts mit der »Funktion« im Sinne eines Unterprogramms zu tun und auch nichts mit dem Begriff »funktional«. Aus heutiger Sicht müssen wir diese Bezeichnung wohl einfach als historisch gewachsen hinnehmen, auch wenn wir über die dadurch manchmal ausgelösten Missverständnisse nicht glücklich sind.

2.6 Der funktionale Baum

Obwohl dieses Vorgehen in der Function-Point-Analyse nicht vorgeschrieben wird, erfolgt in der Praxis die Identifikation der Elementarprozesse häufig über eine funktionale Dekomposition, deren Ergebnis als Grafik eines funktionalen Baums (kurz auch Funktionsbaum) dargestellt wird. Die Blätter dieses Baums repräsentieren die Elementarprozesse (vgl. Abb. 2–2).

Das Vorgehen einer funktionalen Dekomposition liefert in seiner Konsequenz zwangsläufig die Elementarprozesse, wobei auf die richtige Anwendung des Atomaritätsprinzips als »Abbruchkriterium« für die Zerlegung geachtet werden muss.

Die Funktionsbaum-Darstellung hat darüber hinaus einen wichtigen Vorteil: Sie erlaubt eine auch für einen »ungeübten« Betrachter übersichtliche und nachvollziehbare Auflistung der identifizierten Elementarprozesse.

Diese Darstellung wird deshalb auch von verschiedenen Werkzeugen zur Dokumentation von Function-Point-Analysen, wie z. B. der Function Point WORKBENCH™, unterstützt.

Bei all seinen Beschränkungen ist der im Zuge einer Function-Point-Analyse erstellte funktionale Baum häufig die erste strukturierte Dokumentation fachlicher Anforderungen und hat allein deshalb wesentliche Bedeutung.

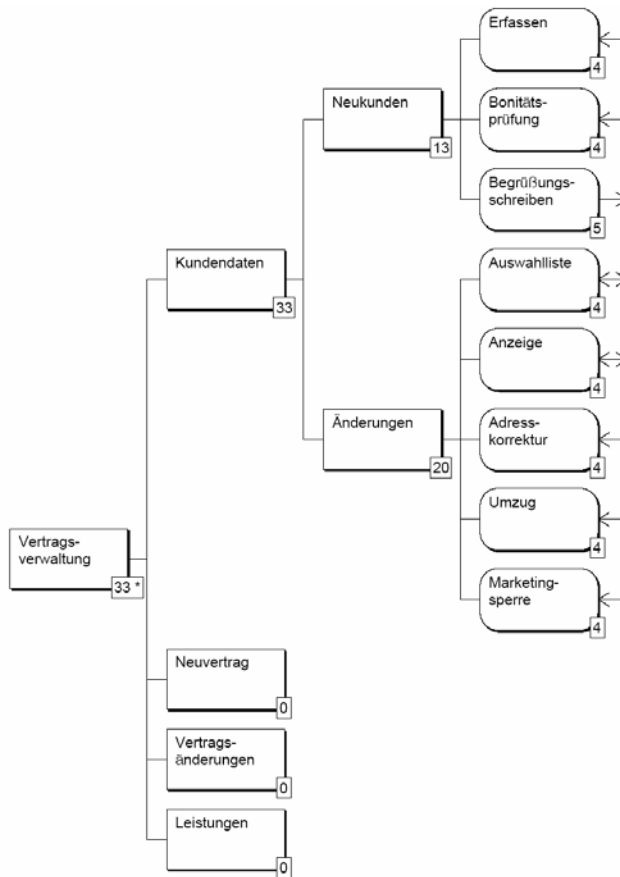


Abb. 2-2 Beispiel für einen funktionalen Baum (erzeugt mit der Function Point WORKBENCH™)

2.7 Bewertung von IT-Systemen

Aus dem oben Beschriebenen wird schon deutlich, dass die Function-Point-Analyse zunächst ein Verfahren zur Messung des fachlichen Leistungsumfangs von IT-Systemen ist.

Dabei sollte man sich immer wieder deutlich machen, dass die Definition der Datenbestände und Elementarprozesse aus Sicht der Anwender bzw. der Geschäftsprozesse erfolgt. Nebenbei, dies ist der Grund, warum allgemeine Function-Point-Angaben für Standardsoftware wenig sinnvoll sind. Standardsoftware kann mit der Function-Point-Analyse nur im Kontext ihrer konkreten Verwendung in einem Unternehmen bewertet werden.

Die Bewertung einer Anwendung ist immer eine Momentaufnahme, die den Zustand eines Systems zu einem bestimmten Zeitpunkt beschreibt. Da der Messgegenstand zu diesem Zeitpunkt existiert und fassbar ist (so wie z. B. die kleine

Anwendung Outlook-Adressbuch, die jeder Windows-Nutzer auf seinem PC vorfindet), ist grundsätzlich eine präzise Messung möglich.

Um diese durchführen zu können, brauchen wir ein tiefes Verständnis des Systems. Allerdings nicht darüber, *wie* es aufgebaut ist und in seinem Inneren funktioniert, sondern vielmehr darüber, *was* es für den Anwender bzw. für den Geschäftsprozess leistet. Das »Was« ist wichtig für die Function-Point-Analyse, nicht das »Wie«. Wir sollten zur Analyse einer Anwendung also den Anwender befragen und nicht den Entwickler oder Softwaredesigner.

Die Analyse einer Anwendung, ein *application count*, kann Aufschlüsse über die von einem System gelieferte Funktionalität geben. Der ermittelte Wert kann Bezugsgröße z. B. für die Einschätzung von Wartungskosten und Fehlerhäufigkeiten sein. Ein *application count* dient aber auch als Basis für die Bewertung von Erweiterungsprojekten und wird deshalb häufig auch als »Baseline« bezeichnet.

Nicht immer wird es notwendig sein, eine präzise und detaillierte Messung für eine Anwendung durchzuführen. Kann ein gewisser Fehler in der Bestimmung akzeptiert werden, gibt es entsprechende Näherungsverfahren, die die Angabe eines Function-Point-Werts innerhalb einer recht gut bekannten Unsicherheit angeben. Mit diesen Näherungsverfahren lassen sich Aufwand und Kosten für eine Analyse reduzieren, allerdings auch um den Preis einer verringerten Nachvollziehbarkeit und Präzision.

2.8 Bewertung von IT-Projekten

Wenn wir den Aufwand für ein Projekt mit Hilfe von Function Points abschätzen wollen oder die Projektleistung anhand von Function Points mit anderen Projekten vergleichen wollen, müssen wir das Verfahren sinnvoll auf die Projektsituation übertragen können. Dazu müssen wir die Aufgaben und das Vorgehen des Projekts verstehen. Handelt es sich um die Neuentwicklung eines Systems »auf der grünen Wiese«? Oder wird die Funktionalität eines Systems weiterentwickelt? Oder wird eine Standardsoftware in die Umgebung des Kunden integriert und an die Erfordernisse des Unternehmens angepasst?

Bei einer Neuentwicklung wird man das Endergebnis unmittelbar mit der Leistung des Projekts identifizieren. Handelt es sich dagegen um eine Weiterentwicklung, so wird man wohl nur daran interessiert sein, den aktuellen Beitrag des Projekts zu bewerten.

Noch interessanter ist der Fall der Integration einer Standardsoftware: Will man das Projekt im Sinne einer *Make-or-buy*-Entscheidung betrachten, so sollte das gesamte Projektergebnis mit Function Points bewertet werden. Will man hingegen die Softwareentwicklungsproduktivität im Projekt messen, so sollten nur die tatsächlichen Integrations- und Anpassungsaktivitäten des Projekts berücksichtigt werden.

Die Function-Point-Analyse enthält eine Reihe von Regeln für die Bewertung von Neuentwicklungs- und Erweiterungsprojekten (*development projects* und *enhancement projects*). Darüber hinaus muss immer die konkrete Fragestellung und Zielsetzung beim Einsatz der FPA für die Bewertung von Projekten berücksichtigt werden.

Ist das Projekt abgeschlossen, so lassen sich die durch das Projekt implementierten fachlichen Funktionen präzise beschreiben. Damit lässt sich auch der Function-Point-Wert in einer Nachbetrachtung genau messen. Für Aufwandschätzungen ist dagegen immer ein Planungsstand die Grundlage, der mehr oder weniger konkretisiert ist, beispielsweise ein Pflichtenheft oder vielleicht nur eine noch gröbere Beschreibung der Anforderungen. Eine »Messung« der zugehörigen Function Points ist hier aufgrund der Informationslücken noch nicht möglich. Trotzdem lassen sich – entsprechende Erfahrung vorausgesetzt – recht zuverlässige Prognosen abgeben.

Die häufig in der Literatur zu findende Aussage, eine Function-Point-Analyse sei frühestens mit Abschluss eines Fachkonzepts möglich, ist hingegen so nicht richtig. Eine wirklich sichere Messung lässt sich natürlich erst nach Projektabschluss durchführen, denn erst dann steht unzweifelhaft und tatsächlich die Leistung des Projekts fest. Zum Zwecke der Aufwandsschätzung reichen jedoch Prognosen aus, die naturgemäß einer umso größeren Streuung unterliegen, je früher sie durchgeführt werden.

2.9 FPA im Projektzyklus

Einer der großen Vorteile der Function-Point-Analyse liegt in ihrer Verwendung über den gesamten Lebenszyklus eines Softwareprojekts. So lässt sich mit ihr die Entwicklung des Anforderungsumfangs von der ersten Projektidee über die verschiedenen Konzept- und Designphasen bis hin zum fertigen Produkt verfolgen. Wir wollen im Folgenden die Anwendung der Function-Point-Analyse im Verlaufe eines Softwareprojekts übersichtsartig darstellen.

2.9.1 Anforderungsanalyse und -beschreibung

Die Beschreibung und Analyse fachlicher Anforderungen in einer sowohl für die Anwender als auch für die Softwareentwickler gemeinsamen Sprache ist und bleibt ein fundamentales Problem. Die Anwender wissen, was sie wollen, können es aber nicht adäquat in Software-Terminologie übersetzen. Die Softwareentwickler verstehen oder glauben manchmal auch nur zu verstehen, was die Anwender wollen, können es aber nicht in einer für diese verständlichen Sprache beschreiben. Die Missverständnisse lassen sich wegen des Fehlens einer gemeinsamen Sprache nicht aufklären.

Auch die in den letzten Jahren entwickelte *Unified Modeling Language* (UML) hat dieses Problem nicht lösen können. Der Spagat zwischen anwenderverständlicher Darstellung der Anforderungen und dem Anspruch, gleichzeitig die Grundlage für eine quasi automatisierte Umsetzung der Anforderungen in ein IT-System zu sein, ist zugunsten Letzterem ausgegangen. Die UML selbst ist so formalisiert und zugleich mächtig, dass ein »normaler« Anwender sie nicht mehr ohne zeitaufwändige Ausbildung verstehen kann.

Auch die Function-Point-Analyse mit der durch sie vorgegebenen strukturierten funktionalen Dekomposition fachlicher Anforderungen, die in einem funktionalen Hierarchiebaum resultiert, bietet hier keine vollständig befriedigende Lösung. Im praktischen Einsatz des Verfahrens werden jedoch zwei wertvolle Ergebnisse erreicht:

- Der funktionale Hierarchiebaum mit der Darstellung der Elementarprozesse liefert eine grobe Systemübersicht. Diese ist statisch und beschreibt nur das »Was« (d. h. das, was das System leisten soll) und nicht das »Wie«, stellt aber gerade damit den zu einem frühen Projektzeitpunkt erreichbaren größten gemeinsamen Nenner dar. Der Hierarchiebaum und das Prinzip der Elementarprozesse sind auch einem nicht vorbelasteten Anwender leicht verständlich zu machen.
- Gleichzeitig ist das Verfahren hinreichend genau definiert. So kann bereits zu einem frühen Projektzeitpunkt eine quantitative Aussage über den fachlichen Funktionsumfang aus dem funktionalen Hierarchiebaum abgeleitet werden.

Ist es Ihnen aufgefallen? In der Anforderungsanalyse werden Function Points nicht gemessen. Vielmehr dient die FPA, ergänzt um das Konzept des funktionalen Baums, einer ersten Grobbeschreibung der fachlichen Anforderungen.

2.9.2 Aufwandsschätzung

Die Aufwandsschätzung war ja die Ausgangsmotivation für die Entwicklung der FPA. Tatsächlich spielt dieser Anwendungsbereich heute jedoch nur noch eine vergleichsweise geringe Rolle in der praktischen Anwendung des Verfahrens.

Basierend auf ihren Vorgehensmodellen, Entwicklungsprozessen und technischen Entwicklungsumgebungen haben heute zahlreiche Unternehmen, insbesondere in der IT-Dienstleistungsbranche, eigene Verfahren für sichere und nachvollziehbare Aufwandsschätzungen entwickelt. Grob kann man sagen, dass diese proprietären Verfahren der FPA etwa ab dem Vorliegen eines Fachkonzepts hinsichtlich der Aussagegenauigkeit überlegen sein können.

Ihre Berechtigung hat die FPA für die Aufwandsschätzung jedoch immer noch aus drei Gründen:

- Zu sehr frühen Projektzeitpunkten stellt die FPA einen Rahmen für eine strukturierte Grobbeschreibung der Anforderungen zur Verfügung.
- Auf der FPA basierende Aufwandsschätzungen sind nachvollziehbar, auch außerhalb des eigenen Unternehmens.
- Im Zusammenwirken mit Projektcontrolling und Benchmarking liefern sie eine konsistente Messgröße für die Projektleistung über den gesamten Projektlebenszyklus.

Wichtig für einen zielführenden Einsatz der FPA zur Aufwandsschätzung ist vor allem die Erkenntnis, dass sie selbst kein Aufwandsschätzverfahren ist. Auch wenn dies bis heute in der Presse und leider auch in der Fachliteratur irrigerweise häufig so dargestellt wird: Für die Ermittlung einer Aufwandsschätzung sind noch wesentliche weitere Informationen notwendig, die von der FPA gar nicht geliefert werden können⁷.

Ein gutes Beispiel für das Zusammenwirken eines Aufwandsschätzmodells mit der FPA ist das COCOMO-Verfahren, das weiter hinten im Buch ausführlich behandelt wird. Dieses Verfahren verwendet den FP-Wert als Größenmaß für die fachlichen Anforderungen, der zu erwartende Projektaufwand wird dann unter Hinzuziehung 23 weiterer Parameter ermittelt.

2.9.3 Auftragsvergabe

Soll die Entwicklung des IT-Systems durch einen Dienstleister erfolgen, gilt es, den einerseits günstigsten Anbieter zu finden, andererseits aber auch eine zuverlässige Lieferung und hohe Qualität des Systems sicherzustellen. Auch im Prozess der Auftragsvergabe findet die Function-Point-Analyse deshalb Anwendung.

Zunächst ermöglicht eine Function-Point-Analyse eine Präzisierung der Anforderungsdefinition zwischen Auftraggeber und -nehmer. Hierzu gibt es verschiedene Vorgehensweisen:

Auftraggeber und -nehmer erstellen auf der Basis des Anforderungsdokuments unabhängig voneinander Function-Point-Analysen. Die Ergebnisse werden verglichen, eventuelle Abweichungen geben Hinweise auf zusätzlichen Klärungsbedarf in der Anforderungsdefinition.

Etwas direkter wäre der Weg, dass der Auftragnehmer direkt mit dem Auftraggeber die Function-Point-Analyse durchführt. Die dritte Alternative schließlich ist die Erstellung von Function-Point-Analysen der von den Auftragnehmern vorgelegten Angebote durch unabhängige Gutachter.

Liegen Angebote mehrerer Auftragnehmer vor, so lässt sich neben dem jeweiligen Gesamtpreis nun auch ein »Preis pro Function Point« vergleichen. Damit

7. Tatsächlich findet sich auch in der FPA ein Ansatz, diese weiteren Einflussfaktoren schon in den FP-Wert einfließen zu lassen. Dieser ist in Abschnitt 3.9 beschrieben. Dieser Ansatz findet jedoch aus den dort beschriebenen Gründen in der Praxis keine Anwendung.

kann nicht nur das »billigste«, sondern auch das »günstigste« Angebot ermittelt werden.

In Situationen, in denen die Einholung von Vergleichsangeboten nicht möglich ist, stellt die Function-Point-Analyse schließlich die einzige Möglichkeit dar, ein Projektangebot hinsichtlich seiner Marktgerechtigkeit zu bewerten.

Schließlich gibt es noch die Fälle, in denen eine Beauftragung bereits stattfinden soll, obwohl die Anforderungen noch nicht vollständig feststehen. Hier gibt es nur die Alternative, nach *time and material*, also dem tatsächlichen Aufwand, abzurechnen oder aber nach *unit costs*, also einem Stückpreis pro geliefertem Function Point. Wir glauben, dass die zweite Alternative letztlich die für beide Seiten vorteilhafteste Abrechnungsweise ist. Der Auftraggeber zahlt letzten Endes nur für den tatsächlich erhaltenen Funktionsumfang. Auf der anderen Seite kommen durch den Auftragnehmer erzielte Produktivitätsverbesserungen auch diesem zugute.

2.9.4 Controlling

Im Controlling dient die Function-Point-Analyse dazu, eine Vergleichbarkeit der Leistung zwischen Projekten und Anwendungen herzustellen. Die Überwachung von Budget- und Zeitplaneinhaltung ist in der Regel nicht ausreichend, um die Leistung eines Softwareentwicklungsbereichs oder eines Softwaredienstleisters zu beurteilen.

Interessant aus der Perspektive des IT-Controllers ist hier vor allem die Möglichkeit, die Leistungsmessung ohne Hinzuziehung des Dienstleisters durch die neutrale Bewertung eines außenstehenden Gutachters ermitteln zu können.

Auch in aktuellen Controlling-Standards, wie etwa dem ITIL-Standard⁸, werden Größenmessungen (*size*) der in der Softwareentwicklung gelieferten Produkte verlangt. Insbesondere der ITIL-Standard *Application Management* liefert wertvolle Hinweise auf zu verfolgende Mess- und Ergebnisgrößen.

2.9.5 Benchmarking

Benchmarking ist ein Begriff, der in der Mitte der 1980er Jahre als Managementmethodik entwickelt wurde. Im Kontext der Softwareentwicklung wird der Begriff üblicherweise eingeschränkter verwendet, d.h. mehr seiner üblichen Bedeutung (benchmark = Vergleichswert) entsprechend. Beim Benchmarking geht es also im Grunde einfach darum, die eigene Leistung mit der anderer zu vergleichen.

Die Vergleichbarkeit der Leistung von Softwareprojekten, aber auch von Aktivitäten in der Softwarewartung und -betreuung, wird heute durch die Mes-

8. [Office of Government Commerce 2002]

sung nach dem Function-Point-Standard hergestellt. Diese Standardisierung erlaubt den Vergleich auch über Unternehmensgrenzen hinweg, ja sogar auf einer weltweiten Ebene.

In Benchmarks ermittelte Kennzahlen werden häufig als *key performance indicators*, kurz KPI, bezeichnet. Ein KPI unterscheidet sich von einer Messgröße grundsätzlich dadurch, dass er aus diesen abgeleitet wird. Messgrößen für ein Softwareprojekt wären z. B.:

- Funktionale Größe
- Aufwand
- Kosten
- Dauer
- Fehler nach Freigabe

Daraus lassen sich dann u. a. die folgenden KPIs ableiten:

- Produktivität = Funktionale Größe/Aufwand
- Stückkosten = Kosten/Funktionale Größe
- Liefergeschwindigkeit = Funktionale Größe/Dauer
- Qualität = Funktionale Größe/Fehler nach Freigabe

Zum Teil kann die Ableitung der KPIs natürlich auch komplexer sein. So werden z. B. in den Benchmarks der QuantiMetrics GmbH auch ein Prozesseffizienz-Index und ein Staffing-Index bestimmt. Es liegt auf der Hand, dass die Messgrößen und KPIs für einen Projektbenchmark erst nach Abschluss eines Projekts ermittelt werden können.

Die Auswertung solcher KPIs erlaubt entsprechende Rückschlüsse auf Verbesserungspotenziale. Diese müssen nicht notwendig im einzelnen Projekt liegen, nicht selten ist es die Projektumgebung, die die größten Möglichkeiten für Verbesserungen bietet. Eine ausführliche Behandlung der einzelnen KPIs, ihrer Ableitung und Interpretation kann im Rahmen dieses Buches nicht erfolgen. Dennoch wollen wir übersichtsartig die unserer Erfahrung nach wichtigsten darstellen. Abbildung 2–3 zeigt die typische Darstellung von KPIs für ein Softwareprojekt in einem QuantiMetrics-Projektbenchmark.

Die Darstellung im Kiviat-Diagramm ist dabei so gewählt, dass ein »anstrebenswertes« Ergebnis für den Kunden (dunkle Linie) außerhalb des Bezugswertes der Vergleichsgruppe (graue Linie »Median«) liegt. Zusätzlich ist die Lage der jeweiligen oberen Quartile (Top 25 %) und Dezile (Top 10 %) eingezeichnet⁹. So

9. Eine Ausnahme stellt hierbei der Staffing-Index dar. Da eine entspannte oder hohe Personalausstattung eines Projekts nicht für sich gesehen als erstrebenswert oder vermeidenswert gelten kann, ist hier eine Betrachtung der Quantile wenig sinnvoll. Die Abweichung vom Median nach oben (entspannte Planung) oder unten (hohe Personalausstattung) ist im Zusammenhang mit den in den anderen KPIs erzielten Ergebnissen zu betrachten.

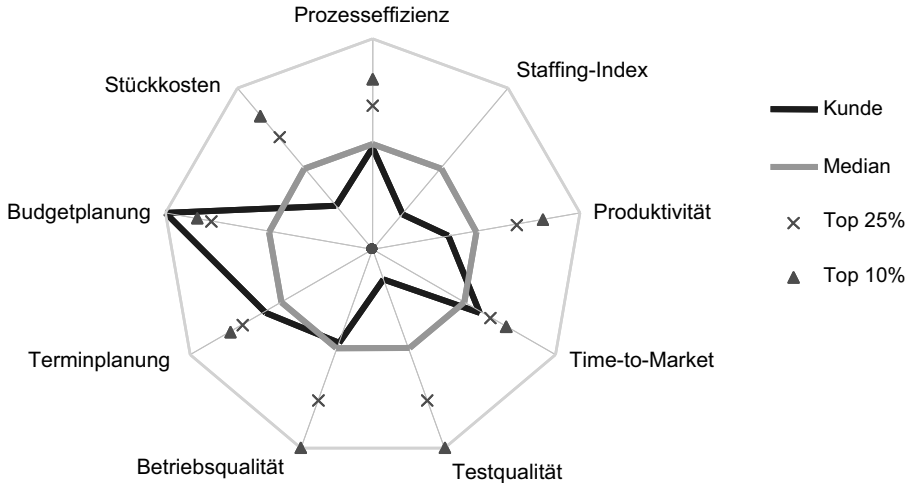


Abb. 2-3 Beispiel einer KPI-Darstellung für ein Softwareprojekt

lässt sich auf einen Blick das erreichte Ergebnis in der Gesamtbetrachtung bewerten. Die hier dargestellten KPIs stehen für:

- *Prozesseffizienz*: Ein von QuantiMetrics berechneter Index, der die erreichte Prozesseffizienz beschreibt.
- *Staffing-Index*: Ein von QuantiMetrics berechneter Index, der die Personalausstattung der Projekte beschreibt. (Werte für eine zeitlich entspannte Personalausstattung liegen außerhalb, Werte für eine hohe Personalausstattung innerhalb der Mediankurve.)
- *Produktivität*: Leistung (FP)¹⁰/Aufwand (PM)¹¹
- *Time-to-Market*: Entwicklungsgeschwindigkeit Leistung (FP)/Zeit (Monat)
- *Testqualität*: Leistung (FP)/Anzahl Fehler, wobei sich diese auf die üblichen systematisierten Teststufen eines Projekts (System-, Integration- und Abnahmetests) bezieht.
- *Betriebsqualität*: Wie Testqualität, die Anzahl der Fehler bezieht sich dabei auf einen bestimmten Zeitraum (üblicherweise 30 Tage) nach Produktionsstart.
- *Terminplanung*: Ein von QuantiMetrics berechneter Index, der die Einhaltung der Terminplanung beschreibt. Dabei werden auch Änderungen im Funktionsumfang gegenüber den Planwerten berücksichtigt.
- *Budgetplanung*: Wie Terminplanung, aber bezogen auf Aufwands- oder Kostenplanungen.

10. FP = Function Points

11. PM = Personenmonat

- *Stückkosten*: Projektkosten normiert auf die Anzahl der gelieferten Function Points.

Ähnlich wie Softwareprojekte lassen sich auch Aktivitäten in der Softwarewartung und -pflege mit Hilfe von Functions Points vergleichbar machen oder »benchmarken«. Ein KPI wäre hier zum Beispiel der Wartungsaufwand pro Function Point oder die Häufigkeit von Fehlern pro Function Point, jeweils in einem bestimmten Zeitintervall. Ein Ergebnisbeispiel ist in Abbildung 2–4 dargestellt.

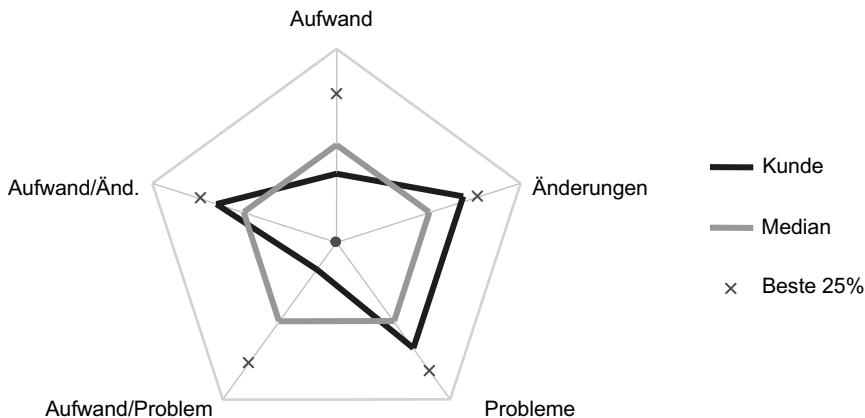


Abb. 2–4 Beispiel einer KPI-Darstellung für Softwarewartungsaktivitäten

Die hier dargestellten KPIs stehen für:

- *Aufwand*: Aufwand für Wartung (PM)/Systemgröße (1000 FP)
- *Änderungen*: Anzahl der Änderungen/Systemgröße (1000 FP)
- *Probleme*: Anzahl der beobachteten Probleme/Systemgröße (1000 FP)
- *Aufwand/Problem*: Aufwand für Problembehebung/Anzahl der Probleme
- *Aufwand/Änderungen*: Aufwand für die Durchführung von Änderungen (funktionaler oder technischer Art)/Anzahl Änderungen

Vergleiche zwischen Projekten oder Anwendungen lassen sich im internen Vergleich innerhalb eines Unternehmens, in einer Zeitreihe oder auch im externen Vergleich gegen andere Unternehmen durchführen. Dabei können einzelne Projekte oder Anwendungen, Stichproben oder auch die Gesamtheit innerhalb des Unternehmens betrachtet werden. Jeder dieser Ansätze liefert andere Erkenntnisse und kann für sich genommen sinnvoll sein.

Immer wieder wird beim Benchmarking die nahe liegende Frage gestellt, wie sich die Vergleichbarkeit der Daten herstellen ließe. Hier müssen genau genommen zwei Aspekte unterschieden werden: Zunächst gilt es sicherzustellen, dass die dem Vergleich zugrunde liegenden Messgrößen in gleicher Weise erhoben

wurden. Dies gilt also für den Function-Point-Wert, aber auch für alle anderen Größen wie Aufwands- und Kostenzahlen. Für die Ermittlung des FP-Werts sollte also auf eine möglichst standardkonforme Vorgehensweise geachtet werden, gerade deshalb bietet sich der IFPUG-Standard (der ausführlich im Kapitel 3 beschrieben wird) als heute weitverbreitetster Standard an. Für die Ermittlung von Aufwands- und Kostenzahlen gibt es heute noch keine einheitlichen Standards, so dass hier im Einzelfall ein Abgleich vorgenommen werden muss.

Ist die einheitliche Erfassung der Messgrößen sichergestellt, bleibt immer noch die Frage, ob zwei ganz verschiedene Projekte oder verschiedene Unternehmen überhaupt vergleichbar sind. Diese Frage ist jedoch falsch gestellt: Ob zwei Objekte miteinander verglichen werden, ist immer eine subjektive Entscheidung des Betrachters, die sich an seinen Zielsetzungen orientiert. Anders gesprochen: Das Management z. B. eines Anwendungsbereichs muss entscheiden, welche Projekte, innerhalb und außerhalb des eigenen Hauses, es miteinander vergleichen will und im Vergleich zu welchen anderen Organisationen eine Benchmark-Analyse durchgeführt werden soll.

Ausschlaggebend sind dabei die konkreten Zielsetzungen und Fragestellungen. Soll z. B. durch den Benchmark die Effizienz einer besonderen Entwicklungsumgebung untersucht werden, so müssen die Projekte mit solchen aus anderen Entwicklungsumgebungen verglichen werden. Geht es dagegen darum, die Effizienz der Entwicklungsprozesse innerhalb der Entwicklungsumgebung zu bewerten, sollte der Vergleich gegen Projekte aus einer gleichen oder ähnlichen Entwicklungsumgebung durchgeführt werden.