Peter Hoppen, Christian Hoppen

Bewertung und Bilanzierung selbst erstellter Software

Auswirkungen des BilMoG und Vorgehensweise bei der Wertbestimmung

Mit dem BilMoG erschließt sich Unternehmen ein Aktivierungswahlrecht für selbst geschaffene immaterielle Wirtschaftsgüter. Aktivierbar sind Aufwendungen, die i.d.R. ab dem 1.1.2010 anfallen und einen einzeln verwertbaren immateriellen Vermögensgegenstand darstellen. Der Beitrag erörtert die gesetzlichen Möglichkeiten nach neuer Gesetzeslage und gibt Praxistipps zur aktivierungsfähigen Gestaltung laufender Projekte. Es wird dargestellt, wie die Werthaltigkeit der Entwicklungsaufwendungen mit Hilfe von Software-Metriken nachgewiesen werden kann und welche Softwareentwicklungsunterlagen geführt werden sollten, um die Basis einer fundierten Bewertung zu schaffen.

I. Neuregelungen des Bilanzrechtsmodernisierungsgesetzes (BilMoG)

1. Bisherige Rechtslage

Nach den bisherigen gesetzlichen Regelungen besteht sowohl handels- als auch steuerrechtlich ein Aktivierungsverbot für selbst geschaffene immaterielle Vermögensgegenstände des Anlagevermögens. Unter diese Kategorie ist auch selbst erstellte Software zu subsumieren, soweit diese dauernd dem Betrieb zu dienen bestimmt ist (Individualsoftware zur Eigennutzung). Hiervon zu unterscheiden ist die auftragsbezogene Softwareentwicklung, die bereits nach bisheriger Rechtslage als Vorratsvermögen (unfertige Leistungen) im Umlaufvermögen zu aktivieren ist.

Eine Aktivierungsmöglichkeit für selbst erstellte Software des Anlagevermögens bestand bisher lediglich im Rahmen einer Bilanzierungshilfe bei Ingangsetzung oder Erweiterung des Geschäftsbetriebes (§269 HGB a.F.).

2. Neuerungen durch das BilMoG

Im Rahmen der Annäherung der deutschen handelsrechtlichen Bilanzierungsregelungen an die internationale Rechnungslegung wurde §248 Abs. 2 ff. HGB neu gefasst.

Hiernach besteht nunmehr ein Wahlrecht, wonach selbst geschaffene immaterielle Vermögensgegenstände des Anlagevermögens als Aktivposten in die Bilanz aufgenommen werden dürfen.

Ein Anwendungsfall dieses neu geschaffenen Aktivierungswahlrechtes stellt selbst erstellte Software dar, die im Unternehmen selbst genutzt oder auch im Wege der Lizenzvergabe an entsprechende User zur Nutzung überlassen wird.

Gemäß Art.66 Abs.3 EGHGB finden die neuen gesetzlichen Regelungen erstmalig in Einzel- und Konzernabschlüssen für Geschäftsjahre Anwendung, die nach dem 31. Dezember 2009 beginnen. Der Bilanzierende kann die Neuregelungen des BilMoG jedoch bereits auf nach dem 31. Dezember 2008 beginnende Geschäftsjahre anwenden, sofern er diese Option vollständig hinsichtlich sämtlicher gesetzlicher Neuregelungen des BilMoG ausübt.

3. Bewertungsmaßstab, Abgrenzung zwischen Forschungs- und Entwicklungskosten

Als Bewertungsmaßstab der Aktivierung selbst geschaffener immaterieller Vermögensgegenstände, und somit auch selbst erstellter Software, sind nach allgemeinen Grundsätzen die Herstellungskosten anzusetzen.

Gemäß § 255 Abs.2a HGB n.F. sind als Herstellungskosten bei selbst geschaffenen immateriellen Vermögensgegenständen die bei der Entwicklung anfallenden Aufwendungen anzusetzen, die durch den Verbrauch von Gütern und die Inanspruchnahme von Diensten für die Herstellung eines Vermögensgegenstandes, seine Erweiterung oder für eine über seinen ursprünglichen Zustand hinausgehende wesentliche Verbesserung entstehen. Hierzu zählen neben den entsprechenden Einzelkosten im Wesentlichen auch angemessene Teile der Gemeinkosten.

Die aktivierbaren Entwicklungskosten sind von Forschungskosten abzugrenzen, die grundsätzlich einer Aktivierung nicht zugänglich sind. Für die Ermittlung der Herstellungskosten ist die Erstellung immaterieller Vermögensgegenstände somit in eine Forschungs- und eine Entwicklungsphase zu unterteilen[2]. Die Forschungsphase ist durch die Suche nach neuen wissenschaftlichen oder technischen Erkenntnissen oder Erfahrungen allgemeiner Art, über deren technische Verwertbarkeit und wirtschaftliche Erfolgsaussichten grundsätzlich noch keine Aussage gemacht werden kann, charakterisiert. Die Entwicklungsphase umfasst dagegen die Anwendung von Forschungsergebnissen oder von anderem Wissen für die Neuentwicklung von Gütern oder Verfahren oder die Weiterentwicklung von Gütern oder Verfahren mittels wesentlicher Änderungen.

Die Software-Programmierung ist aber in der Regel als Entwicklungsleistung anzusehen, da hier zielgerichtet und mit entsprechenden Werkzeugen ein konkretes immaterielles Wirtschaftsgut erarbeitet wird.

Weitere Voraussetzung für die Aktivierung von Entwicklungskosten für selbst geschaffene immaterielle Vermögensgegenstände ist die Qualifizierung als Vermögensgegenstand. Erst wenn ein Vermögensgegenstand entstanden ist, dürfen die entsprechenden Entwicklungskosten aktiviert werden. Ein Vermögensgegenstand ist dann entstan-

▷ Dr. Ing. Peter Hoppen ist Diplom-Informatiker und in Köln als öffentlich bestellter und vereidigter IT-Sachverständiger mit Schwerpunkt Organisation und Systemanalyse tätig.

▷ Christian Hoppen ist Wirtschaftsprüfer und Steuerberater in der Kanzlei Flick Gocke Schaumburg und Geschäftsführer der Flick Gocke Schaumburg GmbH Wirtschaftsprüfungsgesellschaft in Bonn.

[2] Vgl. Hoppen/Husemann/Schmidt, Das neue HGB-Bilanzrecht, 2009, S.63 ff.

den, wenn mit hinreichender Wahrscheinlichkeit eine selbständige Verwertbarkeit sowie eine selbständige Bewertbarkeit gegeben ist und der Bilanzierende die Verfügungsmacht über diesen Vermögensgegenstand hat.

Sofern im Rahmen der Folgebewertung selbst erstellter Software Erhaltungsaufwendungen anfallen, so sind diese nicht aktivierungsfähig. Lediglich nachträgliche Herstellungskosten, die zu einer wesentlichen Verbesserung der Software über den ursprünglichen Zustand hinaus führen, sind einer weiteren Aktivierung zugänglich, sofern mit der Entwicklung der ursprünglichen Software grundsätzlich frühestens im Jahr der erstmaligen Anwendung der gesetzlichen Neuregelungen durch das BilMoG begonnen wurde.

4. Praktische Anwendungsfälle der Neuregelungen

a) Vorhandene Software

Gemäß Art.66 Abs.7 EGHGB dürfen Entwicklungskosten für selbst erstellte Software nur aktiviert werden, sofern mit der Entwicklung in dem Geschäftsjahr begonnen wird, das nach dem 31. Dezember 2009 beginnt (optional: nach dem 31. Dezember 2008 s.o.).

Dies bedeutet sowohl ein Aktivierungsverbot für bereits existierende selbst erstellte Software als auch für nachträgliche Herstellungskosten die im Zusammenhang mit bereits in der Vergangenheit selbst erstellter Software stehen.

b) Zukünftige Projekte

Bei der Entwicklung zukünftiger selbst zu erstellender Software ist wegen der stringenten Stichtagsbetrachtung des Beginns der Entwicklungsphase strikt darauf zu achten, dass die Entwicklungsphase erst in dem Jahr der erstmaligen Anwendung der gesetzlichen Neuregelungen (in der Regel 2010) begonnen wird. Der Beginn der Entwicklungsphase ist entsprechend dokumentieren.

Grundsätzlich ist zur Abgrenzung der aktivierbaren Entwicklungskosten von den nicht aktivierungsfähigen Aufwendungen ein leistungsfähiges F&E-Controlling einzurichten, das unzweifelhaft die für die Ermittlung der aktivierbaren Herstellungskosten relevanten Informationen zur Verfügung stellt.

c) Laufende Projekte

Hinsichtlich laufender Projekte gelten grundsätzlich die gleichen gesetzlichen Regelungen, wie für bereits vorhandene Software (vgl. 4.a).

Es besteht jedoch die Möglichkeit, die Entwicklungsphase zu modularisieren und laufende Projekte in bereits begonnene und noch nicht begonnene Teilprojekte aufzusplitten. In dieser Konstellation sind noch nicht begonnene Teilprojekte als eigenständige Projekte zu qualifizieren. Insoweit sind die noch nicht begonnenen Teilprojekte aktivierungsfähig, wenn der Teilprojektbeginn frühestens im Jahr der erstmaligen bilanziellen Anwendung der handelsrechtlichen Neuregelungen nach BilMoG liegt.

Auch in diesem Fall ist ein leistungsfähiges Projekt-Controlling einzurichten.

d) Startup-Situationen / Finanzierungssituationen / Ausgründungen

Bei einer Unternehmensgründung und in jeder weiteren Finanzierungsrunde ist die Bewertung der Aktiva in der Regel unabdingbar, zumindest jedoch empfehlenswert. Es können sich durch die Aktivierung selbst erstellter Software nicht unerhebliche positive Effekte auf die Eigenkapitalquote, Finanzierungskonditionen und Financial Covenants ergeben.

e) Überschuldungs- / Insolvenzsituationen

Keine Änderungen hinsichtlich des bilanziellen Ansatzes selbst erstellter Software durch das BilMoG ergeben sich gegenüber der bisherigen Rechtslage in Bezug auf einen Vermögensstatus im Rahmen einer möglichen Überschuldungssituation. Im Falle einer drohenden oder vorliegenden Insolvenz sind die Gegenstände der Insolvenzmasse und somit auch selbst erstellte Software zu erfassen.

An dieser Stelle sei nur kurz darauf hingewiesen, dass eine Überschuldung nach derzeitiger, zeitlich befristeter Rechtslage, zu keiner Insolvenzantragspflicht führt, soweit für das betroffene Unternehmen eine positive Fortführungsprognose besteht.

5. Maßstäbe der Ermittlung eines angemessenen Wertes

Die Wertermittlung kann sich am Substanz- oder am Ertragswert einer Software orientieren. Der Substanzwert bemisst sich anhand der für die Herstellung einer Software erforderlichen finanziellen Ressourcen, während sich der Ertragswert hingegen anhand der marktseitigen langfristig erzielbaren zukünftigen Erfolgsbeiträge ermittelt.

Eine seriöse Software-Bewertung kann sich weder ausschließlich auf Marktdaten, noch ausschließlich auf Angaben zur Technik und Entwicklungsmannschaft abstützen. Hier ist ein interdisziplinäres Vorgehen von Wirtschaftsfachleuten und Informatikern erforderlich. Vielfach sind die in einer Softwareentwicklung steckenden Entwicklungsaufwendungen und Qualitätsrisiken nur bei detaillierter und sachverständiger Betrachtung beurteilbar.

Entsprechend der kodifizierten Ermittlungsnormen der Wertansätze bilanzierungsfähiger immaterieller Vermögensgegenstände ist zunächst auf die Herstellungskosten abzustellen. Dieser Wertansatz entspricht somit auch etwaigen Wiederbeschaffungskosten.

Im Rahmen der Folgebewertung sind die aktivierten Herstellungskosten um planmäßige Abschreibungen über die betriebsgewöhnliche Nutzungsdauer der selbst erstellten Software zu reduzieren.

Die tatsächlichen bzw. fortgeführten Herstellungskosten können jedoch nicht ohne weiteres angesetzt werden, sofern dem Vermögensgegenstand am Bilanzstichtag ein niedrigerer Wert beizulegen ist (z.B. aufgrund ineffektiver Softwareentwicklung). Insoweit ist der Wertansatz zu verifizieren und ggf. einer außerplanmäßigen Abschreibung zu unterziehen.

Hierzu stehen verschiedene Wertermittlungsmethoden zu Verfügung (siehe auch IDW S 5). Bei der Verifikation sind die tatsächlich angefallenen Kosten (i.d.R. Lohnkosten im eigenen Unternehmen nebst Gemeinkosten oder Unterbeauftragung von Subunternehmern) einem objektivierten Wert der selbst erstellten Software gegenüberzustellen. Dabei kann es sich a) um den Wert handeln, der nach dem Umfang der Softwareentwicklung üblicherwei-

se zur Entwicklung hätte angesetzt werden müssen, oder, falls es sich um eine Software handelt, die am Markt auch Dritten gegenüber zur Nutzung angeboten werden kann, b) um den Marktwert.

Die Wertermittlung kann sich insoweit nach vergleichbaren Transaktionen (Kaufpreis), Ableitungen eines Ertragswertes anhand von in der Zukunft erzielbaren Lizenzeinnahmen oder aus einem Reproduktionswert (Cost Approach) ableiten. Bei einer selbst erstellten Individualsoftware ist der Cost-Approach häufig das einzig praktikable Verfahren[3]. Bei der Ermittlung eines Reproduktionswertes sind die jeweils verwandte spezifische Technologie, ihre technische Aktualität sowie die Funktionalität und die jeweiligen Kostenkomponenten einer Reproduktion (Programmieraufwand) zu berücksichtigen.

Neben der Notwendigkeit, die Entwicklung aufzuzeichnen, darzulegen und nachzuweisen, ergeben sich für den Wirtschaftsprüfer häufig faktische Schwierigkeiten bei der Ermittlung des der Entwicklung zu Grunde liegenden angemessenen Zeit- und Wertgerüstes. Im Wesentlichen ist dies auf die Komplexität und Vielschichtigkeit der Informationsbasis zurückzuführen. Hierzu bieten sich aus technischer Sicht die nachfolgend beschriebenen systematischen Verfahren an.

II. Verfahren der Informatik zur Bewertung selbst erstellter Software

Die Informatik verfügt über etablierte Verfahren zur Abschätzung von Entwicklungsaufwendungen und Beurteilung von Projektrisiken[4]. Solche *Software-Metriken* können helfen, die Entwicklungskosten einer Software im Vorfeld abzuschätzen oder auch rückwirkend anhand der erzielten Arbeitsergebnisse (entwickelter Quellcode) zu plausibilisieren.

In der Praxis am häufigsten angewandt werden die sog. *Function-Point-Analyse* FPA und das *Constructive Cost Model* COCOMO. Aus FPA ist in den letzten Jahren das besser auf objektorientierte Softwareentwicklungen zugeschnittene Verfahren COSMIC-FFP entstanden. Diese Software-Metriken werden nachfolgend näher beschrieben.

1. COCOMO

Boehm entwickelte bereits in den 80er Jahres des vorigen Jahrhunderts ein Schätzverfahren namens COCOMO (*Constructive Cost Model*)[5]. Dieses Verfahren kann angewandt werden, um den Entwicklungsaufwand einer Software zu bestimmen und beruht auf tatsächlichen Projektaufwendungen in großen Softwareprojekten. Das Verfahren wurde im Laufe der Zeit kontinuierlich weiterentwickelt und steht heute als COCOMO II- Modell der University of Southern California (USC) zur Verfügung[6]. Dabei erfolgte eine Anpassung des Modells an die sich verändernden Entwicklungsumgebungen und aktuelle Programmiersprachen. Außerdem aktualisiert die USC regelmäßig die in das Modell einfließenden Parameter, die durch die Beobachtung und Auswertung von Softwareentwicklungsprojekten verfeinert werden[7].

a) Bestimmung des Entwicklungsaufwandes

In dem Verfahren wird der Entwicklungsaufwand einer Software in Personenmonaten *PM* ermittelt. Dieser umfasst in dem Modell nicht nur die reinen Programmierarbeiten, sondern den gesamten Projektaufwand. Er bestimmt sich in Abhängigkeit der Programmgröße *Size* durch die Gleichung

$$PM_{nominal} = A \times (Size)^{B}$$

Gleichung (1)

Der Multiplikator A sowie der Exponent B passen das Modell den spezifischen Projektgegebenheiten an. A ist eine Konstante. Die Bildung des Wertes B reflektiert die Entwicklungsbedingungen, unter der die Software entsteht, s. hierzu die nachfolgenden Ausführungen unter 1.c).

Der so ermittelte nominelle Entwicklungsaufwand $PM_{nominal}$ ist mittels einer Reihe von Kostentreiberfaktoren (*EM, effort multipliers;* insgesamt 17) zu korrigieren. Damit werden Charakteristika der jeweiligen Softwareentwicklung berücksichtigt, die den Aufwand beeinflussen.

$$PM_{adjusted} = PM_{nominal} \times \left(\prod_{i=1}^{17} EM_i \right)$$

Gleichung (2)

Die Kostentreiberfaktoren werden multiplikativ angewandt und führen zu einer Korrektur des Entwicklungsaufwands, wenn die Gegebenheiten von „normalen", typischen Projektgegebenheiten abweichen. Im Detail werden die Kostentreiberfaktoren nachfolgend unter 1.d) erläutert.

Bei der Ermittlung des Entwicklungsaufwandes werden somit folgende Schritte durchgeführt:

- Bestimmung des Quellcodeumfangs (*Size*)
- Bestimmung und Berücksichtigung der Entwicklungsbedingungen
- Bestimmung und Berücksichtigung der Kostentreiberfaktoren

Das Verfahren liefert im Ergebnis eine Abschätzung der gesamten Entwicklungsaufwendungen. Dies umfasst die Phasen bzw. Tätigkeiten Systemanalyse, Systementwurf, Programmierung mit Feinkonzeption und Codierung, Modultest, Systemtest/Integration und Dokumentation.

Da das Verfahren davon ausgeht, dass der Quellcodeumfang abgeschätzt werden kann, ist es erst anwendbar, nachdem die Anforderungsanalyse und Projektierung abgeschlossen wurden. Der von dem Verfahren gelieferte Entwicklungsaufwand berücksichtigt deswegen die Aufwendung für Anforderungsanalyse und Projektierung nicht. Diese sind ggfls. noch hinzuzurechnen.

---

[3] s. Mäder, Ehret, Verwertung selbst erstellter Software im Rahmen der Eigennutzung – Auswirkungen des BilMoG, BRZ 2009 Heft 1

[4] s. K. Jantzen, Verfahren der Aufwandsschätzung für komplexe Softwareprojekte von heute, Informatik Spektrum 1/2008

[5] Barry W. Boehm, Software Engineering Economics, 1981

[6] http://sunset.usc.edu/csse/research/COCOMOII/cocomo_main.html

[7] Eine fortentwickelte und auf die Gegebenheiten zeitgemäßer Softwareentwicklungsprojekte adaptierte Fassung von COCOMO II findet sich in einfach nutzbarer Form auch auf der Internet-Präsenz http://www.siegfried-seibert.de/oldhome/quicksoft/index.htm.

Natürlich kann das Verfahren auch angewandt werden, wenn der Quellcode bereits erstellt wurde und damit der Quellcodeumfang bereits bekannt ist. Es kann dann sehr gut dazu genutzt werden, die tatsächlich angefallenen Entwicklungsaufwendungen des Unternehmens zu plausibilisieren bzw. abzustützen.

Die Genauigkeit des Schätzverfahrens kann allgemein mit +/- 25% angesetzt werden, insbesondere, wenn das Projekt abgeschlossen ist und der Quellcode-Umfang aus dem Projekt heraus bekannt ist.

Das Verfahren liefert neben dem Entwicklungsaufwand auch Angaben über die erforderliche Anzahl an Personen und die zu erwartende Entwicklungsdauer.

b) Quellcodeumfang

Die Programmgröße *Size* fließt in der Einheit *KDSLOC* (*kilo delivered source lines of code*) als Tausende effektive Programmbefehle in das Modell ein. Dabei sind Kommentarzeilen und Codebestandteile, die nicht zur effektiven Programmfunktionalität beitragen (z.B. Textcode-Fragmente) nicht zu berücksichtigen.

Zunächst ist daher zu bestimmen, welche Programme zum Gesamtsystem gehören. Dabei sind nur die Befehle zu berücksichtigen, die tatsächliche Programmanweisungen darstellen. Kommentierungen bzw. nicht verwendete Programmteile sind nicht heranzuziehen. Grundsätzlich wird dabei von einer strukturierten Programmierung ausgegangen, die in der Regel einen Befehl pro Programmzeile enthält.

Bei einer retrospektiven Betrachtung einer selbst erstellten Software liegt der Quellcode in der Regel vor. Die o.g. Betrachtungen können dann relativ einfach angestellt werden.

Ist die Anzahl der Programmbefehle dagegen (noch) nicht bekannt, so kann sie in einer Hilfsrechnung näherungsweise anhand der Anzahl zu entwickelnder Programmfunktionen oder Prozeduren, sog. *Function Points* ermittelt werden (s. hierzu weiter unten die Verfahren FPA und COSMIC-FFP). Aus der Anzahl der Function Points kann dann – abhängig von der verwendeten Programmiersprache – der zu erwartende Quellcode-Umfang berechnet werden.

c) Entwicklungsbedingungen

Das Entwicklungsprojekt wird bei der Bewertung nach Komplexität und Schwierigkeit kategorisiert. Dabei sind die Merkmale

- Neuartigkeit
- Entwicklungsflexibilität
- Risikovorsorge
- Teamzusammenarbeit
- Prozessreife

zu berücksichtigen.

Zu jedem Merkmal sind in dem COCOMO-Modell mehrere Faktoren W_i vorhanden, deren Ausprägung nach folgender Skala unterschieden wird:

- sehr niedrig
- niedrig
- normal
- hoch
- sehr hoch
- extra hoch

Nicht alle Merkmale sind in allen Stufen vorgesehen; teilweise liegen keine Faktoren für die Stufen sehr niedrig, niedrig, sehr hoch und extra hoch vor.

Die *Neuartigkeit* (W_1) ist zu bewerten an dem Grad der Identifikation der Organisation mit den Produktzielen, der Erfahrung aus vergleichbaren Softwaresystemen, der parallelen Entwicklung neuer Hardware oder neuer Ablauforganisationen sowie des Erfordernisses neuer Algorithmen oder Datenverarbeitungsarchitekturen.

Die *Entwicklungsflexibilität* (W_2) ist zu bewerten nach der Abhängigkeit der zu entwickelnden Software von vorher definierten Einsatzumgebungen, der Abhängigkeit von extern vorgegebenen Schnittstellenspezifikationen und dem Zeitdruck, unter dem das Projekt steht.

Die *Risikovorsorge* (W_3) bestimmt den Grad der Abstimmung der Projektarchitektur auf die Projektrisiken. Sie ist zu bewerten nach der Existenz und Vollständigkeit eines Risk-Management-Plans, der Abstimmung des Projektplans auf diese Risiko-Plan, dem Anteil der konzeptionellen Projektaktivitäten, dem Grad der Erfordernis hochqualifizierter Mitarbeiter, dem Einsatz von Werkzeugen zur Durchsetzung der Projektziele, dem Grad der konzeptionellen Unbestimmtheit in zentralen Bestandteilen der Softwarearchitektur und der Anzahl erkannter kritischer Risikostellen.

Die *Teamzusammenarbeit* (W_4) ist zu beurteilen anhand der Übereinstimmung der persönlichen Ziele und Kulturen aller Beteiligten, der Fähigkeit und Bereitschaft einzelner Beteiligter, die Ziele anderer zu verfolgen, der Erfahrung in der Zusammenarbeit sowie der Fähigkeit, eine gemeinsame Vision und Commitments zu entwickeln.

Die *Prozessreife* (W_5) kann schließlich anhand eines Untermodells (CMM *Capability Maturity Model* des Software Engineering Institutes der USC) ermitteln werden, auf dessen Darstellung hier verzichtet wird, um den Umfang des Beitrags nicht zu sprengen.

Die Einstufung des Projektes innerhalb der vorgenannten Merkmale kann von einem erfahrenen EDV-Sachverständigen gut vorgenommen werden und bestimmt den Exponenten B aus der o.g. Gleichung (1) zu:

$$B = 0.91 + 0.01 \times \sum W_i$$

Gleichung (3)

Wenn bei der Beurteilung Informationen über die eigentliche Projektentwicklung nicht zur Verfügung stehen, wird in der Regel aus Gründen äußerster Vorsicht von Werten an der unteren möglichen Bandbreite der Bewertung ausgegangen.

d) Kostentreiberfaktoren

Es werden in dem COCOMO-Modell 17 Kostentreiberfaktoren betrachtet, die sich in vier Gruppen zusammenfassen lassen:

- Produkt-Merkmale
- Computer-Merkmale
- Personal-Merkmale
- Projekt-Merkmale

In jeder Gruppe sind mehrere Faktoren vorhanden, deren Ausprägung nach folgender Skala unterschieden wird:

- sehr niedrig
- niedrig
- normal
- hoch
- sehr hoch
- extra hoch

Nicht alle Merkmale sind in allen Stufen vorgesehen; teilweise liegen keine Faktoren für die Stufen sehr niedrig, niedrig, sehr hoch und extra hoch vor. Eine Einstufung als „normal“ hat immer den Wert 1, führt also zu keiner Korrektur des Entwicklungsaufwands.

Folgende *Produkt-Merkmale* werden unterschieden:

- Geforderte Zuverlässigkeit der Software
- Größe der Datenbasis
- Komplexität des Produkts
- Geforderte Wiederverwendbarkeit
- Anforderungsniveau an die Software- Dokumentation

Im Bereich der *Computer-Merkmale* liegen folgende Kriterien vor:

- CPU-Zeit-Beschränkungen
- Hauptspeicher-Beschränkungen
- Änderungshäufigkeit in der Systemumgebung

Die *Personal-Merkmale* berücksichtigen die Qualifikation der im Projekt eingesetzten Mitarbeiter.

- Analyse- Fähigkeiten
- Programmier- Fähigkeiten als Team
- Erfahrung der Programmierer im Anwendungsgebiet
- Erfahrung der Programmierer in der Entwicklungsplattform
- Erfahrung der Programmierer in der Programmiersprache und den eingesetzten Software-Werkzeugen
- Personalfluktuation

Folgende *Projekt-Merkmale* existieren:

- Einsatz moderner Softwarewerkzeuge
- Kommunikationsintensität
- Geforderte Entwicklungszeit

e) Erfüllungsgrad

Ergänzend ist bei einer noch nicht abgeschlossenen Softwareentwicklung zu berücksichtigen, in welchem Umfang die einzelnen bei einer Softwareentwicklung üblicherweise zu durchlaufenden Projektphasen bereits fertiggestellt sind.

In der Regel kann von folgender grober Prozentverteilung des Gesamtaufwandes ausgegangen werden[8]:

Phase	Prozentanteil
Anforderungsanalyse und Projektierung (außerhalb des durch COCOMO bestimmten Entwicklungsaufwandes)	5 %
Produktdesign (Grob- und Systemkonzept)	15 %
Feinkonzept	20 %
Programmierung	25 %
Modul- und Integrationstests	25 %
Einführungs- und Stabilisierungsphase	10 %
Summe	**100 %**

Zur Ermittlung des Entwicklungsaufwandes ist der Fertigstellungsgrad der einzelnen Bereiche zu ermitteln. Es ist daher zu prüfen, inwieweit die Arbeiten in den einzelnen Phasen erbracht wurden. Hier ist jeweils ein Erfüllungsgrad zu bestimmen.

2. Function-Point-Analysis (FPA)

Ein weiteres gebräuchliches Verfahren zur Aufwandsbestimmung ist die Function-Point-Analysis (*FPA*)[9]. Das Verfahren geht zurück auf erste Ansätze von *Albrecht* bei IBM in 1979, wurde aber im Laufe der Jahre in seiner Definition verfeinert und dem Stand der Technik angepasst. Die *International Function-Point User Group* (*IFPUG*[10]) hat weltweit einheitliche Bewertungsrichtlinien veröffentlicht.

Das Verfahren bestimmt den fachlichen Leistungsumfang und geht aus von den Funktionalitäten, die die Softwareentwicklung aus der Sicht des Endanwenders zu erfüllen hat. Diese werden in die kleinsten sinnvollen und mit dem System durchführbaren abgeschlossenen Aktivitäten, sog. *Elementarprozesse*, zerlegt.

Die Elementarprozesse werden dabei nach einem relativ traditionellen Verständnis von Datenverarbeitung in fünf Kategorien eingeteilt und in ihrer Komplexität jeweils nach festgelegten Kriterien als *Low*, *Average* oder *High* bestimmt. Die fünf Kategorien sind:

- External Input – Übernahme von Daten aus anderen Systemen,
- External Output – Übergabe von Daten an andere Systeme,
- External Inquiry – Annahme, Verarbeitung und Beantwortung von Anfragen aus anderen Systemen,
- Internal Logical Files – Aus Anwendersicht identifizierbare Datengruppe, die von der Anwendung verwaltet wird sowie
- External Interface Files – Aus Anwendersicht identifizierbare Datengruppe, die von der Anwendung nicht verwaltet wird, aber für die Verarbeitung relevant ist.

Abhängig von der Komplexität werden für jeden Elementarprozess nach einem definierten Verfahren sog. *function points* ermittelt. In der Summe bestimmt sich so der gesamte function-point-Wert der Software-Entwicklung. Der function-point-Wert ist ein Maß für den Aufwand zur Programmierung der Software, er berücksichtigt aber nicht die weiteren Projektaufwendungen, die neben der reinen Programmierung anfallen.

[8] s. CuR 1991 Praetorius u. a., Softwarebewertung, S. 499ff., Boehm, a.a.O.

[9] D. Garmus, D. Herron: Function Point Analysis. Addison-Wesiey (2001)

[10] http://www.ifpug.org/

Ursprünglich wurde dieser Wert in dem FPA-Verfahren anhand von 14 System-Charateristiken korrigiert (man spricht dann von *adjusted function points*). Diese System-Charakteristiken sind aber auf heutige Systeme praktisch nicht mehr anwendbar, weil sie teilweise noch von einer Batch-Verarbeitung ausgehen. Die Bedeutung von FPA liegt daher heute in der systematischen Ermittlung von *unadjusted function points*, die dann als Ausgangspunkt für weitere Berechnungen bspw. die Bestimmung des Quellcode-Umfangs in dem COCOMO-Verfahren, dienen.

3. COSMIC-FFP

Im Jahre 1999 entstand auf der Basis von FPA unter Führung des *Common Software Measurement International Consortium* das Verfahren *COSMIC-FFP*, in dem die speziellen Gegebenheiten objektorientierter Software berücksichtigt werden[11]. Beispielsweise ist es mit dieser Metrik möglich, den Umfang eines Software-Systems, das in der Definitionssprache UML (*Unified Modelling Language*) definiert wurde, abzuschätzen. Das Verfahren wurde 2003 als internationaler Standard definiert[12], der in 2008 als ISO/IEC 19761:2008 an die Version 3 von COSMIC angepasst wurde.

Die Elementarprozesse werden in dem Modell *functional process type* genannt und umfassen nicht nur die aus Anwendersicht geforderten Funktionalitäten sondern auch die aus Entwicklersicht zusätzlich notwendigen systeminternen Prozesse. Dadurch können systeminterne Gegebenheiten und Technologieaspekte besser berücksichtigt werden. Jeder Elementarprozess wird danach untersucht, in welchem Umfang er bestimmte Datenbewegungen – nämlich *Read* bzw. *Write* als Lesen oder Schreiben von im System verwalteten Datenbeständen und *Entry* bzw. *Exit* als Annahme bzw. Abgabe von Daten an externe Systeme oder Anwender – vornimmt. Die Anzahl der Datenbewegungen bestimmt dann die Vergabe von sog. *CFSU*s, *COSMIC functional size units*. Die Summe aller CFSUs ist in dem Modell, ähnlich wie bei FPA, das Maß für den Programmieraufwand der Software und dient dazu, den Quellcode-Umfang abzuschätzen. Die übrigen Charakteristika eines Softwareentwicklungsprojektes (vgl. Kostentreiberfaktoren in COCOMO) werden auch hier nicht betrachtet.

III. Weitere Bewertungsaspekte

1. Personalkosten

Die vorgenannten Modelle liefern im Ergebnis Entwicklungsaufwendungen in Personentagen oder Personenmonaten. Um einen bilanzierungsfähigen Betrag zu erhalten, müssen diese Entwicklungsaufwendungen noch mit den Personalkosten bewertet werden.

Hierzu sind Kostensätze anzusetzen – differenziert nach den einzelnen Projektphasen und unterschiedlichen Qualifikationen – die einem EDV-Sachverständigen in der Regel zugänglich sind, teilweise aber auch abgestützt auf eine breite Basis im Internet zugänglich sind[13].

2. Softwareentwicklungs-Dokumentation

Generell kann davon ausgegangen werden, dass rund 30 % der Gesamtkosten im Lebenszyklus einer Software auf die Entwicklung und 70 % auf die Programmpflege und Weiterentwicklung entfallen. Bei der Wertbestimmung der Software muss deswegen geprüft werden, ob aufgrund der konkreten Realisierung Abstriche bei der Wartbarkeit oder der Weiterentwickelbarkeit vorgenommen werden müssen. Eine wichtige Rolle spielen hier qualitätssichernde Maßnahmen, die dazu führen, dass konzeptionelle Fehler möglichst in frühen Phasen der Softwareentwicklung erkannt und korrigiert werden.

Wie bei der Softwareentwicklung vorgegangen wurde, lässt sich bei der Bewertung am besten anhand einer systematisch geführten Software-Entwicklungsdokumentation nachweisen. Dazu gehören:

- Fachkonzept
- DV-Feinspezifikation
- Quellprogramme mit zugehöriger Dokumentation
- Datenbank- Modelle
- Beschreibung der Testverfahren
- Testdokumentation
- Beschreibung der Abnahmeverfahren
- Abnahmeprotokolle
- Softwaredokumentation (System- und Anwenderhandbuch)
- Darstellung des Vorgehensmodells bei der Entwicklung
- Dokumentation der Entwicklungsumgebung
- Aufstellung der beteiligten Mitarbeiter mit Aufgaben und angefallenem Entwicklungsaufwand

3. Marktfaktoren / Ertragswert

Wird eine selbstentwickelte Software nicht ausschließlich im eigenen Unternehmen eingesetzt, sondern auch Dritten in Lizenz angeboten, so bestimmt sich der Wert der Software nicht mehr aus den Herstellungskosten, die mit den zuvor beschriebenen Verfahren bestimmt werden können, sondern wird eher „*durch ihre Eigenschaft bestimmt, Einnahmeüberschüsse zu erwirtschaften*“ bzw. deren Erwirtschaftung in einem Unternehmen zu ermöglichen. *„Der Barwert der zukünftigen Überschüsse der Einnahmen über die Ausgaben bildet den theoretisch richtigen Wert eines Unternehmens*“ bzw. der Software[14].

Um einen solchen Ertragswert zu bestimmen, sind bei der Bewertung zusätzlich folgende Informationen erforderlich:

- Informationen über die Marktpositionierung
  - Angaben zum Gesamtmarkt
  - Marketingunterlagen

---

[11] http://www.cosmicon.com

[12] ISO/IEC 19761:2003: Software Engineering - COSMIC-FFP - A functional size measurement method

[13] Für Freiberufler bietet die Projektbörse Gulp eine Übersicht der Stundensatzforderungen und tatsächlich gezahlter Stundensätze für unterschiedliche Qualifikationen, http://www.gulp.de/kb/st/stdsaetze/mainstfreib.html

[14] Standard des Fachausschusses für Unternehmensbewertung und Betriebswirtschaft (FAUB) des Instituts der Wirtschaftsprüfer IDW S 1 „Grundsätze zur Durchführung von Unternehmensbewertungen“ sowie IDW S 5 „Grundsätze zur Bewertung immaterieller Vermögensgegenstände“

    - Verzeichnis der Installationsbasis / Referenzen
    - Angaben zu den Wettbewerbern
- Vertriebswege
- Kosten des Vertriebs (direkte oder über Provisionen/Rabatte)
- Umsatzentwicklung und Prognose (3 Jahre)
    - Preise für Lizenzen, Updates und Support sowie Prognose zur Preisentwicklung (3 Jahre)
    - Anzahl der vergebenen Lizenzen, Entwicklung und Prognose (3 Jahre)

Bei der Softwarebewertung kann der Wert der Software dann unter folgenden verschiedenen Aspekten betrachtet werden:

1. tatsächlich angefallene Entwicklungsaufwände
2. Ableitung der Entwicklungsaufwände anhand der Größe des Quellcode, bspw. unter Verwendung des COCOMO-Verfahrens
3. Marktwert (unter Berücksichtigung von möglichen Einnahmen aus Lizenzen, Support, Updates bzw. Ableitung aus vergleichbaren Transaktionen)

In der Summe ergibt dies erfahrungsgemäß eine verlässliche, nachvollziehbare und in sich konsistente Wertbestimmung.

IV. Fazit

Durch die Neuregelungen des Bilanzrechtsmodernisierungsgesetzes (BilMoG) bieten sich dem Unternehmen neue Gestaltungsmöglichkeiten hinsichtlich der Aktivierung selbst erstellter immaterieller Vermögensgegenstände, die aufgrund ihrer wirtschaftlichen Auswirkungen im Rahmen der bilanzpolitischen Gestaltung unbedingt berücksichtigt werden sollten.

Wenngleich die Ermittlung der aktivierungsfähigen Herstellungskosten, die innerhalb der Entwicklungsphase selbst erstellter Software anfallen, gewisse Schwierigkeiten bereiten kann, bietet die Informatik entsprechende Metriken zu deren Ermittlung an, die auch die komplexen bewertungsbeeinflussenden Faktoren, wie z.B die Neuartigkeit und die Komplexität der selbst erstellten Software berücksichtigen.

Insbesondere bei bereits begonnenen Entwicklungsprozessen, deren Entwicklungsphase zeitlich sowohl in den Geltungsbereich des Aktivierungsverbotes als auch des neuen Aktivierungswahlrechtes fallen, ist zu prüfen, ob eine Modularisierung in entsprechende Teilprojekte möglich ist.

Zur sachgerechten Dokumentation und Ermittlung der Herstellungskosten ist ein leistungsfähiges Projekt-Controlling einzurichten, um das Aktivierungswahlrecht nicht aus formellen Gründen einer mangelnden Konkretisierbarkeit der für die Aktivierung notwendigen Voraussetzungen zu gefährden.

Insgesamt ist zu empfehlen, dass der Bilanzierende bereits frühzeitig die Aktivierungsfähigkeit und die zu Grunde liegenden Bewertungsparameter mit einem EDV-Sachverständigen und dem Wirtschaftsprüfer abstimmt.