Warum findet die Function Point Analysis so wenig Akzeptanz?

1 Heutige Situation

Wer sich, wie der Autor dieses Aufsatzes, intensiv mit der Software Metrik, im allgemeinen und der Function Point Analysis insbesondere beschäftigt, muß feststellen, daß bei den deutschen IT-Fachleuten diese Themen wenig bekannt sind. Bei denen aber, die sich mit ihnen befaßt haben, scheint geringe Akzeptanz, wenn nicht sogar Ablehnung vorhanden zu sein.
Forscht man nach den Ursachen für diese Situation, so stellt man fest, daß diese vielschichtig sind.

Sie liegen

- im allgemeinen menschlichen Verhalten,
- in der Arbeitsweise der Softwareentwickler,
- im betriebswirtschaftlich orientierten Verhalten,
- in den Methoden selbst (Historie),
- in deren Darstellung und heutiger Ausprägung.

2 Allgemein menschliches Verhalten

Die Mitarbeiter der IT-Abteilungen haben seit der Zeit der weißen IBM Kitteln immer noch eine Sonderstellung in den Unternehmen. Auf der einen Seite bilden sie einen Schlüsselbereich im Unternehmen, ohne den nichts mehr läuft und ohne den wesentliche Veränderungen nicht mehr realisiert werden können. Auf der anderen Seite verstehen sich die It-Spezialisten als kreative Mitarbeiter, auf die nicht die allgemeinen Regeln der Personalführung und –organisation angewendet werden können.

In der Software Metrik sehen IT-Mitarbeiter eine Bedrohung ihres „freien“ Arbeitsraumes. Sie befürchten, daß die Software Metrik genutzt wird, um ihre Leistung zu messen. Die bisherigen Berichtswesen der IT-Bereiche sind fast ausschließlich Arbeistnachweise, die projektbezogen den Zeitaufwand des einzelnen Mitarbeiters erfassen. Die dem Zeitaufwand gegenüberstehende erbrachte Arbeit wird nicht ausgewiesen. Dadurch können aus den heutigen Berichtswesen auch keine Leistungsnachweise abgeleitet werden.
Wird mit Hilfe der Software Metrik die Größe einer Software dargestellt, so sind auch Kennzahlen für die Leistung in der Softwareentwicklung darstellbar. Kann der Umfang einer Arbeit einem Mitarbeiter oder einer Gruppe zugeordnet werden, dann kann in der Tat die Produktivität des Mitarbeiters oder der Gruppe beobachtet werden.

Der Zugang zur Software ist für den Außenstehenden bisher auch deswegen schwierig, weil keine allgemein angewandten Maßgrößen vorhanden sind, auf deren Basis Softwareprodukte verglichen werden können. Die Größe einer Software ist für den

Anwender nicht faßbar. Dadurch hat das IT-Personal gegenüber den Anwendern den Vorteil, daß für Außenstehende Preise, Termine u.ä. in Angeboten, Verträgen aber auch bei innerbetrieblichen Leistungsvereinbarungen nicht nachvollziehbar waren. In unserer technischen Welt gilt, was nicht numerisch nachgewiesen werden kann, grenzt ans Mystische oder Künstlerische, und verlangt staunenden Respekt.

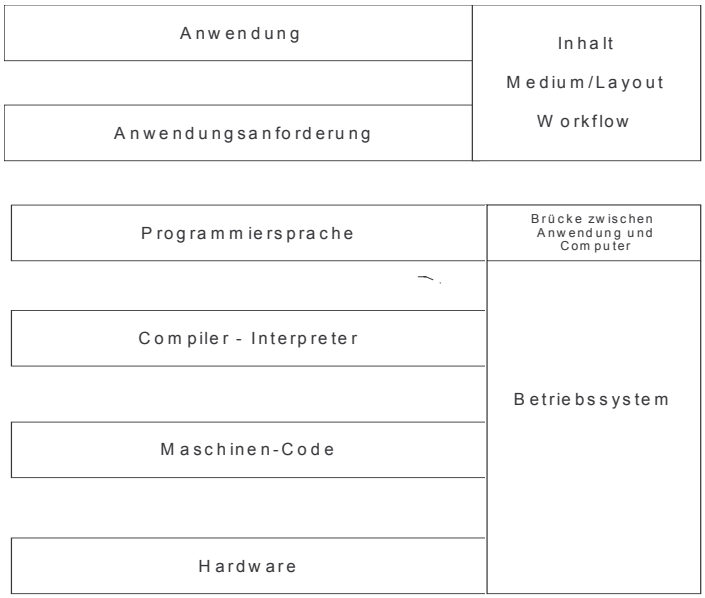
Das Mißtrauen des IT-Personals gegenüber der Software Metrik ist menschlich verständlich. Warum soll man einen Zustand ändern, der mir Ansehen, Respekt, Einfluß und Macht, also nur Vorteile verschafft?
Wenn Argumente gegen die Nutzung von Software Metriken vorgetragen werden, so muß stets die Betroffenheit des IT-Personals an erster Stelle berücksichtigt werden.

3 Arbeitsweise in der Softwareentwicklung

Der Bereich der Softwareentwicklung leidet unter zwei wesentlichen negativen Einflußfaktoren:

1. der schnellen Entwicklung in der Hard- und Software führt dazu, daß sich keine stabile Arbeitsmethoden bilden und
2. der Zeitdruck, unter dem die Softwareentwickler arbeiten – oft selbstverschuldet durch ungenaue Planungsunterlagen -, verführt zu einem nicht ingenieurmäßigen arbeiten.

In den 40 Jahren der Datenverarbeitung jagte eine technische Verbesserung die vorhergehende. Ende der 60-er Jahre hörte man bereits auf, den Generationenbegriff in der Datenverarbeitung zu strapazieren. (1. Generation = IBM 421, 2. Generation = 1401, 3. Generation IBM/360) Die sich rapide verändernde Technik bewirkte Veränderungen in den Programmiersprachen und –techniken. Diese Veränderungen waren vorwiegend Ergänzungen im Hinblick auf die neue Technik. Auf der Anwendungsseite wirkte sich dies so aus, daß die Komplexität, Integration und Geschwindigkeit der Verarbeitung sich erhöhte und die Medien (Lochkarte, Papier, Workstation, DFÜ etc.) sich veränderten bzw. erweiterten.
Die Änderungen auf der Seite der Anwendung sind im Vergleich zur Technik gering. Der Inhalt einer Rechung oder eines Kontoauszuges veränderte sich weniger als die Art ihrer Darstellung oder ihres Mediums. Die ist z.B. deutlich bei einer Umstellung auf das Internet zu sehen. Der Inhalt bleibt konstant, das Medium und das Layout ändert sich.



Dennoch muß festgestellt werden, dass die Änderungen in der Technik sich nicht nur im techniknahen Bereich auswirkte, sondern auch auf die Anwenderanforderung. In der IT ist zu beobachten, daß die Tendenz besteht, mit jeder technischen Veränderung auch die Methodik der Softwareentwicklung von Grund auf zu ändern. Damit wird auch erwartet, daß eine funktionsorientierte Meßmethode permanenten Änderungen unterliegt. In Analogie dazu müßte man sagen, dass jede neue Autogeneration eine neue Straßenverkehrsordnung und eine neue Maßeinheit für die Entfernung nach sich ziehen müßte.

Bei dem Argument „wegen Termindruck keine Zeit für eine fundierte Aufwandschätzung mit Software Metriken, muß man die Frage nach der Henne und dem Ei stellen. Entsteht der Termindruck dadurch, daß ohne fundierte Aufwandschätzung nicht ermittelt werden kann, ob Arbeitsmenge und zur Verfügung stehende Zeit realistisch sind? Oder kann man sich wegen des Zeitdrucks nicht die Zeit nehmen, einen realistischen Zeitplan aufzustellen.
Die allgemeine Erfahrung in der Softwareentwicklung ist, daß mit der technischen Realisierung bereits begonnen wird, ehe eine vom Auftraggeber detaillierte und als vom Auftraggeber verbindlich akzeptierte Aufgabenstellung vorliegt. Die Mehrzahl der Projektleiter ist sich bewußt, daß durch permanente Änderungen zusätzlicher Zeitdruck geschaffen wird oder daß das Produkt bei Einhaltung des geforderten Termins nur unvollständig ausgeliefert werden kann.

Die rasante technische Entwicklung und der extreme Termindruck werden häufig als Argumente gegen eine sinnvolle Nutzung der Software Metrik bzw. der Function Point Analysis vorgebracht.

4 Betriebswirtschaftliches Verhalten

Ein wesentliches Merkmal des Softwarebereiches ist, daß sich dort über Jahrzehnte ein Kostenbewußtsein entwickelt hat, daß sich von dem in fast allen anderen Bereichen der Unternehmen oder Verwaltungen merklich unterscheidet. Ein Kostenmanagement im Sinne betriebswirtschaflicher Arbeitsweise gibt es nicht.

Die Wirtschaftlichkeit der Erstellung und Beschaffung einer Software wird an den Ersparnissen beim Anwender gemessen, die dieser durch die Nutzung der Software erzielen wird. Solange der Nutzen größer als die Kosten für seine Beschaffung oder Erstellung ist, wird nicht die Frage gestellt, ob die Kosten nicht verringert werden können. Es ist ein direkter Zusammenhang zwischen Anzahl der Nutzer und den Kosten für die Erstellung/Beschaffung zu beobachten, die nicht dem tatsächlichen Entwicklungsaufwand entsprechen. So sind z.B. die Preise für Standardsoftware nicht kosten- sondern nutzungsorientiert. Dies ist so, als ob der Preis für ein Buch von der Zahl der Leser bestimmt wird.

Erstaunlich ist, daß nur wenige Informatiker zwischen der Größe einer Software und dem Aufwand für deren Erstellung unterscheiden können. Sie messen die Größe einer Software eher in benötigten Mannmonaten als in Maßeinheiten, die sich z.B. an der Funktionalität/Inhalt der Software orientieren.

In der Softwareentwicklung sind bis heute Kostenanalysen, wie sie in anderen technischen Bereichen Standard sind, nicht üblich. Der Verfasser des Aufsatzes stellte in vielen Seminaren fest, daß den Verantwortlichen in der Softwareentwicklung die Kostenfaktoren/-treiber und deren Strukturen nicht gegenwärtig sind. Dies ist ein Zeichen dafür, daß die Kosten hier nicht die gleiche Rolle spielen wie in anderen Produktionsbereichen. Solange aber keine Kostenanalysen gemacht werden, benötigt man auch keine Kennzahlen, die Produkte von der Größe und von der Qualität her vergleichbar machen.
Erst wenn die Frage gestellt wird, ob die Kosten für die Erstellung und Beschaffung einer Software angemessen sind und ob sie nicht gesenkt werden können, ist man gezwungen Software Metriken zu nutzen. Es reicht dann nicht mehr, die Berichtswesen auf die Kosten zu konzentrieren, sondern es sind dann dem Aufwand die Ergebnisse gemessen an Umfang und Qualität gegenüberzustellen.

Betriebswirtschaftliche Arbeitsweise verlangt stabile Basiskennzahlen, auf die alle Auswertungen und Vergleiche aufbauen. Aufgabe der Software Metrik ist es, diese zu liefern. Solange aber kein betriebswirtschaftlich ausgerichtetes Softwaremanagement gefordert wird, solange wird auch die Software Metrik nicht vom Management gefordert werden.

Ein typisches Beispiel für die fehlende betriebswirtschaftliche Denkweise ist, daß der Return of Investment bei der Beschaffung von Softwaretools und der Einführung neuer Arbeitsmethoden selten oder nie beachtet wird. Der Hang zur Modernität und die Lust auf Neues siegt all zu schnell über kritische Kostenbetrachtungen.

5 Verständnis der Methoden

Die im Kapitel 3 beschriebene Situation, die Größe einer Software an dem Aufwand zu messen und auf eine weitergehende Kostenanalyse zu verzichten, führt dazu, daß die Mehrzahl der Projektleiter in der Softwareentwicklung nicht zwischen der Produktgröße und dem Aufwand sauber trennen. Sicherlich sieht jeder zu recht einen Zusammenhang zwischen Produktgröße und Aufwand. Jedoch müssen nicht immer Produkte mit gleichem Umfang auch mit dem gleichen Aufwand erstellt werden. Unterschiedliche Kostentreiber (Erfahrung der Mitarbeiter, Tools etc.) können zu

unterschiedlicher Produktivität in der Entwicklung und damit zu unterschiedlichen Kosten führen. (Beispiel: Ein Sportwagen mit einem guten Fahrer benötigt für die Strecke Köln-München weniger Zeit als ein LKW, sofern alle anderen Bedingungen wie Wetter, Verkehrsdichte etc. gleich sind.)
Sehr viele Veröffentlichungen zur Function Point Analysis standen in der Vergangenheit und stehen auch heute noch unter der Überschrift „Aufwandschätzung". Es ist daher nicht verwunderlich, daß in den Function Points ein direktes Maß für den Aufwand gesehen wird. Diese falsche Interpretation führt zu der Vorstellung, daß Softwareprodukte, die zwar den gleichen Umfang aber unterschiedliche Komplexität haben oder auf unterschiedlichen Plattformen realisiert werden, mit unterschiedlichen Function Points gezählt werden müssen.

Die richtige Einordnung der Function Point Analysis kann nur dann erwartet werden, wenn im Softwaremanagement die Begriffe der betriebswirtschaftlichen Kostenanalyse bekannt und beherrscht werden.

6 Die Mängel in den heutigen Varianten der Function Point Analysis

Ein Verfahren, das in sich selbst nicht stimmend ist, liefert selbst Argumente für eine Akzeptanzverweigerung.

Die Function Point Analysis unterscheidet sich von anderen Software Metriken dadurch, daß sie nicht die Software an Programmen, Strukturen oder ähnlichem mißt sondern an der Realität, die in der Software abgebildet wird. Genau genommen wird bei der Function Point Analysis nicht das Softwareprodukt gemessen, sondern die in der Software abgebildete Realität. Dies wird in der Literatur zur Function Point Analysis als „Funktionalität der Software aus der Sicht des Anwenders" bezeichnet.

Die Mängel bei allen Varianten der Function Point Analysis sind,

- Sie geben keine Definition für die Funktionalität.
- Es fehlt eine genormte Darstellung der Funktionalität.
- Die historische Entwicklung ist nicht zielgerichtet.
- Sie vermischen funktionsabhängige Meßergebnisse mit anderen.
- Die Veröffentlichungen zur Function Point Analysis bedienen sich einer Sprache, die mehr softwaretechnisch- als funktionsorientiert ist.
- In den Veröffentlichungen wird nicht konsequent zwischen Produktgröße und Aufwand unterschieden..

6.1 Fehlende Definition für die Funktionalität

Es ist ertaunlich, daß sich in allen Veröffentlichungen zur Function Point Analysis keine Definition der Funktionalität einer Software befindet. Selbst in der ISO/IEC 14143/1. die die grundlegenden Prinzipien einer funktionalen Software Metrik beschreibt und die seit 1998 verbindlich ist, wird keine Definition gegeben. Sie gibt lediglich Hinweise auf Dokumentationsunterlagen, die der Messung zu Grunde zu

legen sind. Ferner sagt sie, welche Faktoren bei der Messung nicht beachtet werden dürfen: Qualität, technische und methodische Realisierung etc.

Da Messen Vergleichen bedeutet, müssen zunächst die zu messenden Gegenstände vergleichbar gemacht oder dargestellt werden, damit man mit einem einheitlichen Maß zu vergleichbaren Ergebnissen kommt. Für eine funktionsorientierte Software Metrik bedeutet dies, daß man sich primär auf eine einheitliche Definition der Funktionalität einer Software einigen muß.

6.2 Fehlende genormte Darstellung der Funktionalität.

Alle bekannten Function Point Methoden unterscheiden sich unter anderem dadurch, daß sie andere Darstellungen der Funktionalität verlangen, um die Function Points entsprechend ihrer Methodik ermitteln zu können. Verallgemeinert wird von der Anwendersicht gesprochen, aus der die Grundlagen für die Zählung der Function Points zu erstellen sind. Welche Daten in der Zählgrundlage enthalten sein müssen, und wie diese darzustellen sind, ist in keiner Norm, auch nicht für die einzelne Methode, definiert. So verlangt zum Beispiel die IFPUG-Gruppe eine nach logischen Gruppen getrennte Betrachtung von Eingabe-, Ausgaben- und Bestands- Daten. Die britischen Anhänger der MK II Methode differenzieren dagegen in ihren Zählgrundlagen nur nach Eingabe-, Ausgabe- und Bestands- Daten, ohne logische Gruppen zu bilden.

Grundsätzlich kann sich jedes Unternehmen oder jede Verwaltung eine eigene Norm setzen. Es bzw. sie hat dann jedoch den Nachteil, daß Vergleiche mit anderen Unternehmen hinsichtlich Produktivität und Qualität im Softwarebereich nicht sinnvoll stattfinden können.

Tatsache ist, daß in den wenigsten Unternehmen keine durchgehend einheitliche Darstellung der Funktionalität über längere Zeiträume zu finden ist. Normiertes Messen basiert aber immer auf einer einheitlichen Darstellung der Meßgegenstände..

6.3 Die Darstellung der Function Point Methoden.

Die bisherigen Veröffentlichungen zur Software Metrik und zur Function Point Analysis sind von Informatikern für Informatiker geschrieben worden. Dies spiegelt sich in dem stark technisch orientierten Wortschatz wieder. Was soll z.B. ein Laie sich unter internal und external logical files vorstellen und wie ist dazwischen zu differnzieren? Die Folge ist, daß die Controller und Betriebswirte in den Unternehmen, die die vorliegende Literatur studieren wollen, nur sehr schwer einen Zugang zur Software Metrik finden. Aber auch die Anwender, die Angebotsvergleiche auf der Grundlage des Umfangs der angebotenen Softwareleistung vornehmen wollen, können die vorhandene Literatur zur Software Metrik nicht nutzen. Die Folge ist, daß vorwiegend nur die Informatiker den Zugang zur Software Metrik finden, die jedoch selbst nicht sehr motiviert sind, ihre Aktivitäten meßbar zu machen (s.O.).

6.4 Die historische Erblast.

Es wurde bereits darauf hingewiesen, daß A. Albrecht in den 70er Jahren in den Function Points eine Einheit zur Schätzung des Aufwandes sah.
In den Veröffentlichungen zur Function Point Analysis findet man bis zum Beginn der 90er Jahre keine Anweisungen, die Funktionalität auf der Basis diskreter, auszählbarer Werte zu bestimmen. Wie die nachfolgenden Tabellen zeigen, wurden die Klassifizierungen auf der Grundlage unscharfer Gewichtungen vorgenommen, die dann in eine diskrete Zahl von Function Points transformiert wurden.

Anforderungen	einfach	mittel	komplex
Datenelemttypen	wenige	mehrere	viele
Logische Datengruppen (Bei Bildschirminhalten)	wenige	mehrere	viele
Ansprüche an die Bedienerführung	gering	mittel	hoch
Cusorhandhabung	einfach	mittel	schwierig
Gewicht	3	5	6

Klassifizierung der Eingaben

Anforderungen	einfach	mittel	komplex
Listenspalten	wenige	mehrere	viele
Umsetzen von Datenelementen	einfach	mehrfach	komplex
Dateizugriffe und Verknüpfungen	wenige	mehrere	viele
Anforderungen an die Performance	keine	wenige	mehrere
Gewicht	4	5	7

Klassifizierung der Ausgabendaten

Anforderungen	einfach	mittel	komplex
Datensatzform	eine	mehrere	viele
Performanceeinflüsse	keine	wenige	viele
Wiederanlaufeinflüsse	keine	wenige	viele
DDP-Konzept der Daten	nein	-	ja
Gewicht	7	10	15

Klassifizierung der Datenbestände

Welche Regeln für die Transformationen von einfach, mittel und komplex nach 7,10,15 etc. durch A. Albrecht zu Grunde gelegt wurden, ist nicht nachvollziehbar. Ferner ist aus den Tabellen erkennbar, daß bei Albrecht nicht nur funktionsabhängige Kriterien zur Ermittlung der Function Points herangezogen wurden, wie es ISO/IEC 14143 fordert. Die Kriterien „Ansprüche an die Bedienerführung“ oder „DDP-Konzept der Daten“ sind z.B. eindeutig Merkmale für Qualität bzw. technische Realisierung.

In den nachfolgenden Jahren wurden die unscharfen Kriterien zur Bestimmung der Function Points neu gefaßt. In der IFPUG-Regel werden z.B. datenorientiert Attribute und Relationen als Primärgrößen gezählt. Unverändert bleibt dagegen deren Trans-

formation in die Function Points mit den Gewichtungen, wie sie A. Albrecht vor mehr als 20 Jahren als zweckmäßig für seine Methode zur Aufwandschätzung definierte.

Interne Geschäftsentität	Anzahl Datenelementtypen <= 19	Anzahl Datenelementtypen < 19 und <= 50	Anzahl Datenelementtypen < 50
Anzahl RETs/Beziehungstypen <= 1	7	7	10
Anzahl RETs/Beziehungstypen > 1 und <= 5	7	10	15
Anzahl RETs/Beziehungstypen > 5	10	15	15

Beispiel: Klassifizierung der internen Datenbestände nach IFPUG

In allen Metriken, die heute unter der Bezeichnung Function Point laufen, sind die Function Points eine Sekundärgröße, die nach nicht nachvollziehbaren Regeln aus primären, diskreten und nachzählbaren Einheiten abgeleitet werden.

Historisch ist auch die Differenzierung in unbewertete und bewertete Function Points zu sehen. A. Albrecht benutzte sogenannte Einflußfaktoren, um die zunächst ermittelten Function Points, - für ihn anscheinend identisch mit Aufwand - positiv oder negativ zu korrigieren. Diese Einflußfaktoren wurden und werden durch Anforderungen an die Qualität bzw. der Komplexität der Software, dem Entwicklungsumfeld und der Technik bestimmt. Derartige Einflußfaktoren werden bei allen Function Point Methoden zur Korrektur der unbewerteten Function Points genutzt.

Beispiel:

Einflußfaktoren nach A. Albrecht:

1. Die in der Anwendung zu verarbeitenden Daten werden über Kommunikationseinrichtungen gesendet oder empfangen.
2. Die Verwaltung oder die Verarbeitung der Daten wird nach dem DDP-Konzept durchgeführt.
3. Bezüglich Design Implementierung und Wartung nimmt ein günstiges Antwortzeitverhalten einen hohen Stellenwert ein.
4. Zu Faktor 3 kommt erschwerend hinzu, daß dies auf einem stark belasteten System erreicht werden muß.
5. Design, Implementierung und Wartung werden durch ein hohe Transaktionsrate beeinflußt.
6. Die Dateneingabe erfolgt online.
7. Die Online-Dateneingabe ist dialogorientiert und eine Transaktion wird aus mehreren Verarbeitungsschritten aufgebaut.
8. Die Dateien und Datenbanken werden online gewartet.
9. Es liegt ein komplexe Verarbeitung vor, die viele Verknüpfungen, Entscheidungen, logische und mathematische Gleichungen und Ausnahmeregeln beinhaltet.
10. Bei Entwicklung, Implementierung und Wartung ist eine Wiederverwendung der Programme in anderen Anwendungen zu berücksichtigen.
11. Bei Design und Implementierung sind Konvertierungen zu berücksichtigen.
12. Für die Erleichterung der späteren Bedienung sind bestimmte Prozeduren vorzusehen.
13. Bei Design, Implementierung und Wartung muß berücksichtigt werden, daß die Anwendung in mehreren Funktionen oder lokal getrennten Stationen installiert wird.

14. Bei der Entwicklung der Anwendung wird vorgesehen, dem Benutzer die Bedienung und einen eventuellen Änderungsdienst zu erleichtern.

Diese Umrechnung von unbewerteten in bewertete Function Points widerspricht der Methodik jeder funktionsorientierten Meßmethode wie durch die ISO/IEC 14143 definiert wird. (Nebel auf der Autobahn verlängert nicht die Fahrstrecke, sondern nur die Fahrzeit.)

Es ist zu beobachten, daß die Informatiker, die sich erst mal mit der Function Point Analysis beschäftigen, die historisch gewachsenen Widersprüche schnell als Argument gegen die Software Metrik ins Feld führen, um selber keine Software Metrik anwenden zu müssen.

7 Prognosen

Es ist nicht zu erwarten, daß kurzfristig eine Änderung des Akzeptanzverhaltens zur Software Metrik eintreten wird, solange sich nicht einige Änderungen im IT-Bereich vollziehen.

Die Mitarbeiter in der Softwareentwicklung werden erkennen müssen, daß sie durch die Metrik nicht nur kontrolliert werden, sondern daß sie von ihrer Seite aus z.B. zuverlässige Argumente in die Hand bekommen, um unrealistische Terminsetzungen frühzeitig aufzuzeigen.

Das Management im Softwarebereich darf sich nicht weiterhin ausschließlich aus Informatikern=Insidern rekrutieren. Es muß ein Kostenmanagement mit dem Ziel eingerichtet werden, die Produktivität zu verbessern.

Terminsetzungen dürfen nicht ohne Machbarkeitsstudien vorgenommen werden. Die Machbarkeitsstudien müssen auf Produkt- und Produktivitätskennzahlen basieren.

Die heute gängigen funktionsorientierten Meßmethoden müssen im Hinblick auf die ISO/IEC 14143 überarbeitet werden. Die z.Zt. vorhandenen Widersprüche in den Function Point Anwendungen müssen ausgeräumt werden.

Die Function Point User Groups müssen bereit sein, selbstkritisch das Verfahren zu prüfen, das sie z.Zt. vertreten. Sie sollten dann die methodischen Widersprüche bereinigen, so daß die Verfahren keine Argumente gegen sich selbst liefern.

www.huerten-partner.de