Case-Kritik ist keine Rechtfertigung für grossmassstäbliche Software-Basteleien : vom Nutzen umfassender Software-Entwicklungsumgebungen

Autor(en): **Widmer, Rolf**

Objekttyp: **Article**

Zeitschrift: **Bulletin des Schweizerischen Elektrotechnischen Vereins, des Verbandes Schweizerischer Elektrizitätsunternehmen = Bulletin de l'Association Suisse des Electriciens, de l'Association des Entreprises électriques suisses**

Band (Jahr): **83 (1992)**

Heft 17

PDF erstellt am: **10.07.2024**

Persistenter Link: https://doi.org/10.5169/seals-902860

Case-Kritik ist keine Rechtfertigung für grossmassstäbliche Software-Basteleien

Vom Nutzen umfassender Software-Entwicklungsumgebungen

Rolf Widmer

Die Erfahrungen mit Computer-Aided Software Engineering (Case) haben nicht wenige Anwender und Entwickler enttäuscht. Schuld daran sind allerdings nicht die Bemühungen, mit bewährten Ingenieur-Arbeitsweisen die immer teurer werdende Softwareentwicklung und -wartung in den Griff zu bekommen, sondern meist unrealistische Erwartungen, die in Case gesetzt worden sind. Dieser Beitrag zeigt, dass umfassend verstandenes Case (SEU) sinnvoll, ja unabdingbar ist, wenn man die langfristigen Konsequenzen in Betracht zieht.

Les expériences avec le Computer Aided Software Engineering (Case) ont déçu bon nombre d'utilisateurs et de développeurs. Responsables de cet état de chose ne sont pas les efforts déployés pour maîtriser, avec des méthodes de travail d'ingénierie éprouvées, le développement et la maintenance de plus en plus coûteux de logiciels, mais les attentes le plus souvent peu réalistes dans la Case. Cet article montre qu'un Case compris globalement est judicieux à condition d'en considérer les conséquences à long terme.

Adresse des Autors
Rolf Widmer, Dipl.Informatik-Ing. ETH, lic.oec. HSG, GfAI Gruppe für Angewandte Informatik AG, 3037 Herrenschwanden.

Nach der Euphorie der letzten Jahre herrscht zurzeit eine ausgeprägte Ernüchterung, was den Einsatz von Case angeht. Einer der Gründe für ein Versagen der Konzepte liegt in unrealistischen Zielsetzungen, d.h. in der Erwartung eines unerreichbaren Nutzens. Projekte aber, die an unrealistischen Zielen gemessen werden, fallen in einer nachträglichen Manöverkritik natürlich durch. Andere Gründe der Case-Ernüchterung sind fehlende oder ungenügende Voraussetzungen, ein falsches Vorgehen bei der Case-Einführung oder das Ergreifen punktueller Massnahmen anstatt des Einsatzes integraler Lösungen (z.B. nur die Einführung von Werkzeugen ohne die Abstimmung mit den anderen Elementen einer SEU).

Falsche Zielsetzungen sind in zweierlei Hinsicht festzustellen: einerseits wird die Zeit, die verstreicht, bis sich Case-Massnahmen auswirken, oft unterschätzt, und andererseits wird der erzielbare Nutzen oft zu einseitig betrachtet und nur auf die Systementwicklung selbst bezogen. Wirkungen, welche die Unternehmung als Ganzes betreffen, bleiben in der Nutzenbetrachtung – und damit in der Entscheidungssituation – oft unberücksichtigt.

SEU statt Case

Der Begriff Case wird vielfach mit Werkzeugen assoziiert, welche den Entwicklungsprozess unterstützen sollen. Weil aber solche Case-Werkzeuge noch in den Kinderschuhen stecken, wird Case oft als Zukunftsmusik oder sogar als Illusion abgetan. Man vergisst dabei, dass die Werkzeuge nur einen Teil einer umfassenden Software-Entwicklungsumgebung (SEU) bilden. Für den Erfolg einer Software-Entwicklung spielen die Methoden und Vorgehensmodelle sowie vor allem der Mensch eine zentrale Rolle; ohne sie nützen auch die besten Werkzeuge nichts. SEU und Case können, wenn sie in einem weiten Sinn verstanden werden, gleichgesetzt werden. Um zu betonen, dass nicht die Werkzeuge im Vordergrund stehen, wird in diesem Beitrag der neutralere Begriff Software-Entwicklungsumgebung (SEU) verwendet.

Begriff und Nutzen umfassender SEU

Moderne Software-Entwicklungsumgebungen (SEU), im englischen Software-Engineering Environments (SEE), sind entgegen aller Case-Skepsis in der Lage, einen wesentlichen Beitrag bei der Kontrolle von Software-Entwicklungsprozessen und damit von Investitionen in Informationssysteme zu leisten. Unter dem Begriff Software-Entwicklungsumgebung, der den etwas abgegriffenen und oft zu wenig umfassend verstandenen Begriff Case ersetzt, wird dabei ein umfassendes System von Werkzeugen, Vorgehensweisen, Methoden und Techniken ebenso wie ein entsprechendes Projekt-Management, die Informatik-Organisation (organisatorische Einbindung der Projekte in die Entwicklungsumgebung) und die technische Infrastruktur (Bild 1) verstanden (und keine blossen computerbasierten Entwicklungswerkzeuge). Unter dem Begriff SEU ist somit eine computerunterstützte Infrastruktur zu verstehen, die technologische, methodische, organisatorische sowie menschliche Elemente umfasst und welche die Software-Entwicklung wirkungsvoll unterstützt. Alle diese Elemente müssen bei der Konzeption und dem Aufbau einer SEU in die Überlegungen einbezogen und aufeinander abgestimmt werden; sie alle beeinflussen den Entwicklungsprozess, dessen

Unterstützung der primäre Zweck einer SEU ist.

Der Nutzen von Software-Entwicklungsumgebungen besteht nicht primär in einer schnell erreichbaren Produktivitätssteigerung, wie sie vielfach von Herstellern von Case-Werkzeugen angepriesen wird, sondern in langfristigen Wirkungen. Die Einführung einer SEU verlangt eine Veränderung der Arbeitskultur, und diese kann nur langsam – im Lerntempo des Menschen – erfolgen. Bis sich die neue Arbeitskultur erfolgreich etabliert hat und der Umgang mit neuen Methoden, Werkzeugen und Denkweisen zur Routine geworden ist, kann es kurz- bis mittelfristig sogar zu einer Produktivitätseinbusse kommen.

Beim Einsatz einer SEU werden zwei Nutzenkategorien unterschieden: Nutzenwirkungen in der Systementwicklung (direkter Nutzen) und Nutzenwirkungen in der Unternehmung als Ganzes (indirekter Nutzen, Tab. I).

Direkter Nutzen

Kontrolle und Lenkung des Entwicklungsprozesses

Ein enormer Vorteil, der sich durch die Einführung einer SEU ergibt, besteht in der Kontrolle des gesamten Entwicklungsprozesses. Die umfassende Strukturierung eines Projekts in Phasen und Arbeitsgegenstände, die klare Festlegung der organisatorischen Verantwortlichkeiten sowie die entsprechende Werkzeugunterstützung erlauben dem Projektleiter eine verbesserte Überwachung und Lenkung des gesamten Entwicklungsprozesses.

Insbesondere erlaubt die klare Definition von Schnittstellen und der durchgängige Einsatz von Werkzeugen eine Arbeitsteilung zwischen den einzelnen Mitgliedern des Entwicklungsteams. Daraus resultiert eine bessere Koordination, so dass der Projektleiter von bestimmten Koordinationsaufgaben entlastet wird. Welche Rolle eine effiziente Lenkung und Kontrolle spielt, kann nicht genug betont werden. Dabei ist nicht der eigentliche Kontrollaspekt der entscheidende Punkt, sondern vielmehr die Massnahmen, die beim Erkennen von Abweichungen des Ist-Zustandes gegenüber dem Soll-Zustand eingeleitet werden. Diese Lenkung (engl. Controlling) ist das Kernstück jedes effektiven Projektmanagements, da nicht nur Abweichungen erkannt (kontrolliert), sondern auch durch Gegenmassnahmen reduziert (gelenkt) werden, womit das Projektergebnis insgesamt verbessert wird.

Die Informatik besitzt in vielen Unternehmungen strategischen Charakter; sie verändert Marktstrukturen, schafft neue (z.B. elektronische) Märkte und beeinflusst die Organisation, die Prozesse und die Produkte der Unternehmungen [1]. Dieser Stellenwert der Informatik spiegelt sich in entsprechend teuren Investitionen wider. Die dafür bereitzustellenden Mittel sind – insbesondere beim heutigen Kostendruck – straff zu kontrollieren und effizient einzusetzen. Ohne klare Konzepte und brauchbare Hilfsmittel ist dies ein Ding der Unmöglichkeit.

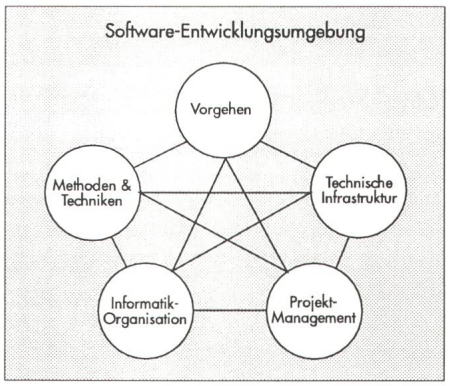

Bild 1 Elemente einer umfassenden Software-Entwicklungsumgebung

Verbesserung des Wartungsproblems

Die Übernahme bestehender, oft überalterter Anwendungen in moderne SEU stellt heute ein viel diskutiertes Thema dar, das unter dem Stichwort Re-Engineering abgehandelt wird. Ein bedrängendes Problem ist, dass das in vielen Informationssystemen eingepflanzte Wissen über Datenstrukturen und Abläufe oft nur in den Köpfen der Mitarbeiter gespeichert ist und im Laufe der Zeit verloren geht. Erweiterungen, Anpassungen an neue Anforderungen oder Korrekturen können deswegen sehr zeitintensiv sein. Die Folge davon ist, dass die Wartungskosten bis 70% (in Extremfällen sogar noch mehr) der gesamten Lebenszykluskosten ausmachen. Diese Altlasten werden für die Unternehmungen zu einem zunehmenden Problem, da sie das Budget belasten und Investitionen in neue Anwendungen verhindern.

Der Einsatz von SEU kann diesbezüglich mittel- bis langfristig eine Verbesserung erzielen. Eine Überführung der bestehenden Anwendungen in eine auf modernen Grundsätzen beruhende SEU und eine anschliessende Neuimplementierung nach den Grundsätzen des modernen Software Engineerings (Stichwort Re-Engineering) erlauben einerseits, eine nicht oder nur teilweise vorhandene Dokumentation neu zu erstellen, und andererseits, Anwendungsänderungen nicht mehr direkt im Code, sondern auf der konzeptionellen Ebene vorzunehmen.[1] Die Vorteile liegen auf der Hand:

- Änderungen können auf der konzeptionellen Ebene sehr viel schneller und qualitativ besser vorgenommen werden als direkt im Code. Untersuchungen haben gezeigt, dass im Wartungsprozess bis zu 50% der Zeit für das Verstehen des Codes eingesetzt wird [2]. Diese Zeit kann offensichtlich wesentlich verkürzt werden, da das Verstehen von Konzepten auf einem hohem Abstraktionsniveau einfacher ist als das Verstehen von unstrukturiertem und undurchsichtigem Code.
- Da jede Änderung computerunterstützt erfolgt, bleibt auch die Dokumentation auf dem neusten Stand und verhindert damit, dass Know-how verloren geht.
- Da das erworbene Know-how weiterhin verwendbar ist, können zu einem späteren Zeitpunkt Migrationen auf neue Technologien vorgenommen werden, ohne dass das geschäftliche Wissen wieder von Grund auf neu erarbeitet werden muss. Die getätigten Investitionen werden so besser geschützt.
- Die Verbesserung des Wartungsproblems wiederum hat weitere positive Folgen. So kann der Anwendungs-Backlog[2] reduziert werden, da weniger Zeit für Wartungsarbeiten aufgewendet werden muss. Die Durchlaufzeit für neue Anwendungen wird somit, wie weiter unten gezeigt, entsprechend reduziert.

[1] Obwohl ein grosser Teil des Re-Engineering manuell durchgeführt werden muss, ist Re-Engineering oft die einzige Lösung, um existierende Systeme nachdokumentieren und in neue SEU überführen zu können.

[2] Bestand von geforderten, aber noch nicht realisierten Anwendungen.

Qualität der Produkte

Durch den Einsatz von ingenieurmässigen Methoden in der Software-Entwicklung, die bis zum heutigen Tag noch zu stark auf Intuition basiert, kann selbstverständlich auch die Produktqualität wesentlich verbessert werden. Qualität darf nicht erst in der Implementierungsphase zum Thema werden, sie muss bereits in den ersten Projektphasen zentrales Element sein. Durch eine frühe Anwendung von Qualitätssicherungsmassnahmen, wie dies in der industriellen Fertigung schon lange der Fall ist, lassen sich die Qualitätskosten unter Beibehaltung, ja sogar Steigerung der Produktqualität minimieren. Unter Qualitätskosten werden hier die Kosten für Fehlererkennung und Fehlerbehebung verstanden; im weiteren müssen auch jene Kosten dazu gezählt werden, die aus negativen Folgen von Fehlern in Systemen entstehen, zum Beispiel in Form von Haftungsschäden. Auf diesen letzten Punkt wird weiter unten noch genauer eingegangen werden.

Eine zusätzliche Verbesserung der Qualität ergibt sich durch die Bereitstellung einer Kommunikationsplattform in Form von Modellen und Werkzeugen, die eine kooperative Software-Entwicklung zwischen dem Anwender und dem Informatiker zulassen. Der frühe Einbezug der Anwender, zum Beispiel für die Mitarbeit bei der Modellierung des Geschäftssystems, stellt sicher, dass diese sich mit dem Endprodukt identifizieren können und verhindert damit kostspielige Korrekturmassnahmen.

Auswirkungen auf die Unternehmung

Der Informatikbereich ist kein isolierter Bereich, sondern ist mit anderen Unternehmensbereichen verknüpft und steht mit diesen in vielfältigen Beziehungen. Veränderungen im Informatikbereich rufen Veränderungen in der Unternehmung als Ganzes hervor, die ihrerseits auf die Wettbewerbsfähigkeit einen entsprechenden Einfluss haben können.

Flexibilität

Infolge des verbesserten Entwicklungs- und Wartungsprozesses können Änderungen an den bestehenden Informationssystemen schneller und einfacher durchgeführt werden. Solche Änderungen können notwendig sein, wenn das Umfeld ändert (z.B. die gesetzlichen Rahmenbedingungen) und das Geschäft sich den veränderten Verhältnissen anpassen muss. Dies ist heute, wo die Zeit ein relevanter Wettbewerbsfaktor ist, von entscheidender Bedeutung. Gleichermassen wichtig ist, dass sich alle Unternehmensbereiche möglichst gleich schnell anpassen können; es gilt das Gesetz der Kette, die nur so stark ist wie ihr schwächstes Glied. Die Informationssysteme, welche die Unternehmung mit den notwendigen Informationen versorgen und oft strategischen Charakter haben, müssen schnell an neue Bedingungen angepasst werden können.

Qualität

Qualität aus unternehmerischer Sicht wird heute oft unter dem Schlagwort des Total Quality Managements (TQM) verstanden. TQM stellt den Versuch dar, die eigene Markstellung durch gesteigerte Qualität und Kundenzufriedenheit zu verbessern, wobei alle Mitarbeiter und Unternehmensbereiche involviert werden [3]. Qualität bezieht sich freilich nicht nur auf die Produktqualität, sondern auf jede Befriedigung von Kundenbedürfnissen, also auch auf die Prozessqualität, die ausdrückt, auf welche Weise der Kunde bedient wird. Dabei ist es irrelevant, ob es sich um «echte» oder um interne Kunden handelt (d.h. Bezüger der erbrachten Leistung innerhalb der Unternehmung). Nur eine in allen Bereichen erbrachte Qualität führt letztendlich zu einer langfristigen und umfassenden Kundenzufriedenheit.

TQM bedingt, dass die qualitativen Zielsetzungen für jeden Bereich der Unternehmung gelten, also auch für den Informatikbereich im allgemeinen und die Systementwicklung im speziellen. TQM wird in zwei Punkten vom Einsatz einer SEU betroffen: einerseits bewirkt die Verbesserung des Entwicklungsprozesses selbst sowie dessen effektivere Lenkung eine Zunahme der Prozessqualität, andererseits wird die Produktqualität infolge kooperativer Entwicklung und verbesserten Methoden und Hilfsmitteln gesteigert. Beide Ansätze führen schlussendlich zu einem zufriedeneren Kunden, der das gewünschte Produkt in der gesetzten Frist zu den budgetierten Kosten erhält.

Welchen Einfluss ein konsequent durchgeführtes TQM auf die Marktposition haben kann, hat das japanische Beispiel gezeigt. Der Einbezug aller Mitarbeiter, zum Beispiel in Form von Quality Circles oder mittels eines Vorschlagswesens für Verbesserungen, beweisen, dass eine von der gesamten Unternehmung getragene Qualitätsstrategie sehr erfolgreich sein kann. Mit TQM erreicht man sogar, dass die Kosten langfristig sinken, da der vollständige Einbezug sämtlicher Mitarbeiter zu einem Umdenken und damit häufig zu Einsparungen an überflüssigen und bürokratischen Abläufen führt. Während die traditionelle Gemeinkostenwertanalyse nach einem bestimmten Zeitpunkt ihre Wirkung verliert und die Kosten infolge des fehlenden äusseren Drucks wieder zu steigen beginnen, besitzt TQM durch seinen mitarbeiterzentrierten Ansatz langfristige Wirkung. Die Änderung der Kultur, die damit verbunden ist, bewirkt eine Verbesserung von innen heraus und ist daher wesentlich stabiler als ein von aussen aufgelegtes Kosteneinsparungsprogramm (Bild 2).

Eine derart durchdringende Qualität kann allerdings nicht einfach eingekauft werden, sondern sie bedingt eine Verpflichtung jedes Mitarbeiters auf die Qualitätsziele. Bezogen auf den Einsatz einer SEU bedeutet dies die absolute Bereitschaft, in moderne Ansätze zu investieren und diese dann auch anzuwenden. Freilich stellt eine SEU nur einen kleinen Teil des TQM dar, kann aber – sofern richtig eingesetzt – einen nicht unwesentlichen Anteil an Verbesserungen erbringen, die sich wiederum auf die Unternehmung als Ganzes auswirken. Bereits der

Tabelle I Nutzeneffekte einer Software-Entwicklungsumgebung

Direkter Nutzen	• Kontrolle und Lenkung des Entwicklungsprozesses • Verbesserung des Wartungsproblems • Qualität der Anwendungen
Indirekter Nutzen	• Flexibilität • Beitrag zu Qualitätsstrategie • Vermeiden von Opportunitätskosten

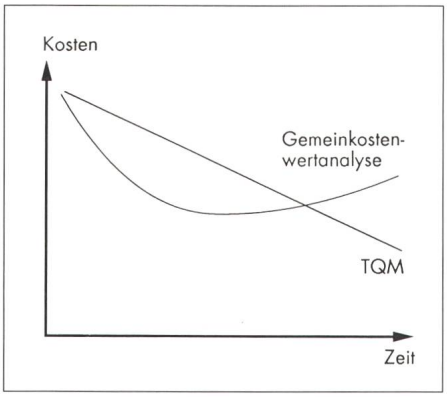

Bild 2 Zu erwartende Kostenverläufe bei Total Quality Management (TQM) und Gemeinkostenwertanalyse

dazu notwendige Kulturwechsel in der Systementwicklung kann schlussendlich zu erhöhter Motivation und damit wiederum zu besseren Leistungen führen.

Nicht zuletzt kommt der Qualität auch aus haftungsrechtlichen Überlegungen Bedeutung zu, besteht doch ein allgemeiner Trend, Hersteller und Dienstleistende für Schäden haftbar zu machen, die auf Qualitätsmängeln von deren Produkten zurückzuführen sind. Haftungsklagen können Unternehmungen in ihrer Überlebensfähigkeit bedrohen. Gerade weil Software-Anwendungen sehr oft den Kunden direkt berühren, zum Beispiel bei der Abwicklung von Zahlungen im Bankgeschäft, darf der Haftungsaspekt nicht vernachlässigt werden.

Vermeiden von Opportunitätskosten

Nicht zu vernachlässigen sind natürlich die Opportunitätskosten, welche die Kosten bzw. Ertragseinbussen umfassen, die entstanden wären, wenn eine SEU nicht eingeführt worden wäre. Sogenannte Mussinvestitionen, wie diese zum Beispiel bei der Einführung von CIM diskutiert werden, sind notwendig, wenn die Unternehmung längerfristig konkurrenzfähig bleiben will. Dazu bedarf es einer ständigen Optimierung aller Prozesse, und zwar sowohl aus Kosten- als auch aus qualitativer Sicht. Denn nur eine Beherrschung von Kosten *und* Qualität kennzeichnet eine erfolgreiche Unternehmung. So ist es auch kein Zufall, dass ein konsequent implementiertes TQM langfristig zu einem Absinken der Kosten führt. Bei der Systementwicklung kann insbesondere die effektivere Kontrolle und Lenkung der Projekte eine derartige Optimierung der Prozesse bewirken. Ein umfassendes Projektcontrolling, das erst durch eine integrale SEU ermöglicht wird, vermag die Kosten, die aus Zielabweichungen entstehen, zu minimieren.

Auf die durch eine umfassende SEU erzielbare Steigerung der Qualität wurde bereits hingewiesen. In diesem Zusammenhang spielt vor allem auch die Tatsache eine wesentliche Rolle, dass für die Vergabe von Entwicklungsaufträgen immer häufiger eine Zertifizierung nach dem ISO 9000-Standard verlangt wird, was ein funktionierendes Qualitätssicherungssystem voraussetzt. Unterlässt eine Unternehmung diese Zertifizierung, so könnte sie längerfristig in einen strategischen Nachteil gelangen, da sie für die Übernahme von Entwicklungsaufträgen nicht mehr zugelassen ist. Dieser Trend kann so weit gehen, dass sogar für interne Entwicklungen diese Standards zur Geltung kommen, zumal diese Leistungen oft bezahlt werden müssen. Der Marktmechanismus betrifft somit nicht nur die Preise, sondern auch die Qualität der erbrachten Leistungen.

Dass ein effizientes Qualitätssicherungssystem eine auf modernen Grundsätzen basierende SEU voraussetzt, ist offensichtlich, da erst der Einbezug aller (d.h auch menschlicher und organisatorischer) relevanten Aspekte in den Entwicklungsprozess eine umfassende und integrale Qualität gewährleisten kann.

Strategische Optionen

In bezug auf die Systementwicklung verfügt jede Unternehmung aus strategischer Sicht über verschiedene Optionen, die sie aus der politischen und der strategischen Warte aus beurteilen muss. So muss sie entscheiden, wie weit sie in eine moderne SEU investieren oder Entwicklungsaufträge im Sinne eines Outsourcings nach aussen vergeben will. In der industriellen Fertigung hat sich diese Erkenntnis schon lange durchgesetzt. Ein Automobilhersteller zum Beispiel ist weder in der Lage noch gewillt, alle Teile und die dazugehörigen Maschinen effizient zu fertigen. Vielmehr werden dafür Unterlieferanten hinzugezogen, die in bezug auf bestimmte Teile oder Verfahren über komparative Vorteile verfügen.

In der Informatik hingegen scheinen solche Überlegungen erst zögernd angestellt zu werden. Meistens werden alle Informatikaktivitäten durch die Unternehmung selbst übernommen, so dass infolge des begrenzten Budgets die knappen Mittel nach dem Giesskannenprinzip verteilt werden müssen. Stärken in Teilbereichen können somit gar nicht aufgebaut werden.

Eine Auslagerungen von gewissen Funktionen an qualifizierte Zulieferanten ist ein modernes Konzept der Organisationslehre, welches die für den verstärkten Wettbewerb notwendige Flexibilität sicherstellen soll [4]. Muss eine Unternehmung sämtliche vor- oder nachgelagerten bzw. unterstützenden Leistungen (wozu auch die Systementwicklung zählt) selbst erbringen, so stehen zu wenig Mittel für den Aufbau eigener nachhaltiger Stärken zur Verfügung; die Unternehmung wird längerfristig ins Hintertreffen geraten. Da aber aus den erwähnten Gründen der Aufbau einer umfassenden SEU für eine effiziente Entwicklung von Informationssystemen unumgänglich ist, muss eine Unternehmung, die noch nicht über eine solche verfügt, in entsprechende Massnahmen investieren oder eben die Entwicklung nach aussen verlagern, da externe Zulieferer Anwendungen (aus der Sicht der gesamten Lebenszykluskosten) billiger und qualitativ besser realisieren können.

Eine solche Entscheidung besitzt weittragende Folgen und muss daher von der Geschäftsleitung selbst gefällt und getragen werden. Dazu bedarf es aber der weitgehenden Einsicht, dass überhaupt etwas unternommen werden muss. Zu viele Firmen begnügen sich mit dem Status quo und zögern, entsprechende Entscheide zu treffen. Dass diese Einsicht sich aber infolge der steigenden Informatikkosten (insbesondere im Bereich der Anwendungsentwicklung) durchsetzen wird, ist anzunehmen.

Literatur

[1] *Porter M.E., Millar V.E.:* How Information Gives you Competitive Advantage. Harvard Business Review, July–August 1985, pp. 149–160.

[2] *Rock-Evans R., Hales, K.:* Reverse Engineering: Markets, Methods and Tools, London, Ovum Ltd., 1(1990), pp. 54.

[3] *Weintraub D.L.:* Implementing Total Quality Management. Arthur D. Little: Prism, First Quarter 1991, pp. 23–35.

[4] *Miles, R.E., Snow, C.C.:* Network Organizations: New Concepts for New Forms. The McKinsey Quarterly, Autumn 1986, pp. 53–66.