# Software-Entwicklung als Lernprozeß

**Die Forderung nach Beteiligung (Partizipation) bei der Gestaltung und Einführung von DV-Systemen ist nicht neu. Sowohl hinsichtlich der erfolgreichen als auch der enttäuschenden Durchführung partizipativer Projekte haben wir in der Bundesrepublik mittlerweile vielfältige Einsichten sammeln können. Ernüchternd ist dabei, zu erkennen, wie wenig sich der Arbeitsalltag aufgrund einmalig durchgeführter Beteiligungsprojekte tatsächlich geändert hat.**

Erfreulich ist dagegen, daß sich im Bereich des Managements und der Softwareentwickler zunehmend die Einsicht durchsetzt, daß durch eine qualifizierte Mitwirkung und Mitbestimmung der Arbeitnehmer sich Entwicklungs- und Produktionsprozesse effektiver und verläßlicher gestalten lassen. Schließlich sollte bei der Abwägung der Erfolge und Mißerfolge noch berücksichtigt werden, daß auch gescheiterte Projekte unsere Erfahrungen hinsichtlich der Grenzen und Möglichkeiten von Partizipation bereichert haben.

Eine dieser Erfahrungen ist, daß die Komplexität der vielfältig ineinandergreifenden sozialen Prozesse weit größer ist als meist angenommen und demzufolge die Planbarkeit und Steuerbarkeit der Prozeßverläufe oft nicht den Erwartungen und Anforderungen der Beteiligten entspricht. Ein entscheidender Grund hierfür ist in

der Tatsache zu suchen, daß eine partizipative Systementwicklung vor allem auch ein kooperativer Lernprozeß ist. Lernprozesse sind jedoch ihrer Natur nach nicht vorhersehbar; Erkenntisse oder Einsichten lassen sich weder vorschreiben noch verordnen. Ohne die Berücksichtigung der drei B's, Bereitschaft, Befähigung und Befugnis können Beteiligungsprojekte daher nicht erfolgreich durchgeführt werden.

Kooperative Lernprozesse erfordern in einem durch unterschiedliche Interessen geprägten betrieblichen Konflikt- und Spannungsfeld ein hohes Maß an Offenheit von allen Beteiligten. Diese Offenheit ist nicht allein eine Frage der Willensbildung und Entscheidungsfindung sowie der Bereitschaft, Verantwortung zu übertragen, sondern sie erfordert auch geeignete Methoden, Techniken und Verfahren, die bezüglich der Projektgestaltung genügend Spielraum lassen und Revisionen ermöglichen. Genau an dieser Stelle hat sich in den letzten Jahren einiges getan, was neue Chancen und Perspektiven eröffnen könnte.

Nachfolgend will ich diesen Trend zu mehr Offenheit und die damit verbundenen Möglichkeiten und Probleme kurz skizzieren. Dabei geht es mir nicht darum, einen Überblick zum Stand der Kunst in der partizipativen Systementwicklung zu geben.[1] Vielmehr will ich versuchen, einen bestimmten Gedankengang deutlich zu machen, der meines Erachtens für die Thematik von besonderer Bedeutung ist: Entwicklung und Einsatz von DV-Systemen basieren in hohem Maße auf sozialen Lernprozessen. Eine Einsicht, deren Konsequenzen sich langsam in den Köpfen von Entwicklern, Managern und Beschäftigten durchzusetzen beginnen.

## Softwaretechnik und Systementwicklung

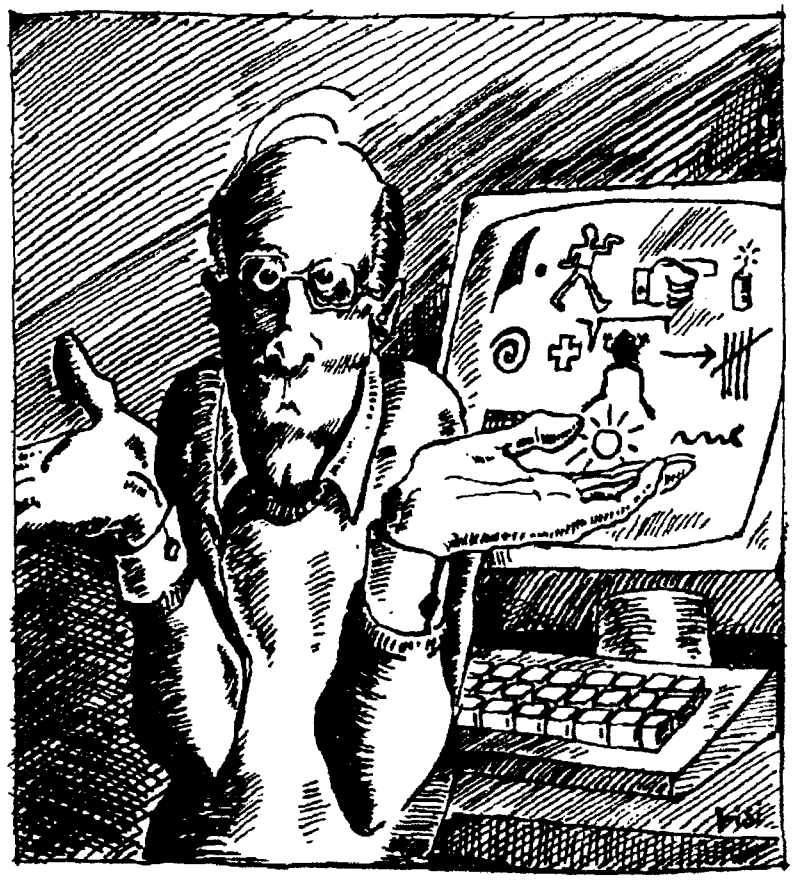
Seit 1968 gibt es den Begriff *Softwaretechnik* (engl. Software Engineering) und etwa seit Mitte der siebziger Jahre hat sich das gleichnamige Fachgebiet in der Informatik etabliert. Der Begriff Software Engineering ist auf einer internationalen Konferenz der NATO geprägt worden, nicht als Benennung eines Tatbestandes, sondern als Ausdruck der Hoffnung und Betonung der Notwendigkeit, daß auch die Entwicklung von Software nach ingenieurwissenschaftlichen Methoden und Verfahren erfolgen sollte.[2]

Ist das denn nicht selbstverständlich? Mitnichten. Die NATO-Konferenz ist einberufen worden, weil die Entwicklung komplexer militärischer Softwaresysteme in die sogenannte *Software-Krise* geführt hatte: die Systeme waren fehlerhaft und zu teuer, wurden nicht fristgerecht fertig und erfüllten nur unzureichend die gestellten Anforderungen und die in sie gesetzten Erwartungen. Hier galt es, *Verfahren, Methoden* und *Werkzeuge* zu entwickeln, die es erlauben sollten, die Entwicklung von DV-Systemen in Analogie zu traditionellen ingenieurwissenschaftlichen Vorgehensweisen zu gestalten.

Interessant ist, daß bereits in der zweiten Hälfte der 60er Jahre das Problem der Beherrschbarkeit großer Softwaresysteme nicht nur aus dieser entwicklungszentrierten Perspektive heraus erkannt und aufgegriffen worden ist, sondern auch aus der Sicht der Beschäftigten im Hinblick auf Ra-

1 Vgl. dazu Rauterberg (1991)

2 Siehe Naur, Randell (1969); S. 13



tionalisierung und Arbeitsplatzgestaltung. Zur damaligen Zeit begannen Wissenschaftler und Gewerkschaften in Norwegen ein Projekt mit dem Ziel, herauszufinden, welches Wissen Arbeitnehmer und Gewerkschaften benötigen, um ihre Interessen in Bezug auf den schnell wachsenden Einsatz der Informationstechnologie wahren zu können.[3] Doch bevor ich auf einige Aspekte der weiteren skandinavischen Entwicklung zurückkomme, möchte ich zuerst die Entwicklung aus der (software-)technischen Sicht schildern.

## Vom Wirtschaftsgut zum gemeinsamen Lernen

Mit der ingenieurwissenschaftlichen Fundierung der Softwaretechnik wird das Ziel verfolgt, Software zu einem Wirtschaftsgut zu machen. *„Um als Wirtschaftsgut zu gelten," stellt Harry Sneed fest, „muß ein geistiges Gut personenunabhängig, reproduzierbar, übertragbar und meßbar sein"*[4]. Bis heute ist es nicht gelungen, diese Idealvorstellung zu verwirklichen. Im großen und ganzen gilt noch immer der Satz, daß sich die Qualität von Software erst im Einsatz erweist. Ihren sichtbarsten Ausdruck findet diese Tatsache in dem Begriff der Version, der in der Informatik eine wichtige Rolle spielt.

Software ist das einzige technische Produkt, das von vornherein in *Versionen* geplant und ausgeliefert wird. Benutzer von Standard- und Systemsoftware müssen in der Regel die Versionsnummer ihres Produktes kennen. Nicht nur wegen des unterschiedlichen Funktionsumfanges den sie nutzen wollen, sondern auch um zu wissen, mit welchen anderen Programmen und Programmpaketen ihre Software kompatibel ist. Hat man einmal ein bestimmtes Standardpaket erworben, kauft man nicht mehr ein neues Modell zum Neupreis, wie bei einem Auto, sondern man kauft für einen erheblich geringeren Teil des Anschaffungspreises lediglich die Veränderungen und Ergänzungen, die die jeweils neue Version mit sich bringt.

Das ist möglich, weil sich der Baustoff Software gegenüber anderen Werkstoffen dadurch auszeichnet, daß neue Funktionen mit verhältnismäßig wenig Aufwand an Energie hergestellt, angepaßt und verändert werden können. Software ist leicht vervielfältigbar, änderbar und erweiterbar — technisch betrachtet. Diese Flexibilität und Leichtigkeit wird allerdings mit einem Preis an anderer Stelle bezahlt: um Erweiterungen und Anpassungen zuverlässig durchführen zu können, muß man die Konsequenzen solcher Veränderungen erkennen und verstehen können. Die Folge: je flexibler und schneller Software geändert wird, desto mehr ist man bezüglich Entwicklung und Benutzung auf das Wissen in den Köpfen der Menschen angewiesen.

Diese Abhängigkeit hat noch einen anderen Grund, den ich mithilfe des Begriffs *Gestaltungskonflikt* charakterisieren will. Gestaltungskonflikte treten auf, wenn sich nicht alle Anforderungen an ein Produkt oder Verfahren gleichermaßen erfüllen lassen. Jede Entscheidung für etwas, z. B. ein neues effektiveres Verfahren, geht zugleich auf Kosten von etwas Anderem, wie z. B. hoher Umstellungs- und Einarbeitungsaufwand. In solchen Fällen muß man die Frage abwägen, inwieweit bestimmte Anforderungen auf Kosten anderer erfüllt werden sollen.

Es gibt eine Fülle von Gestaltungskonflikten auf jeder Ebene der Systemgestaltung. Beispielsweise wäre es sinnvoll, alle für die Bearbeitung eines Kundenauftrages erforderlichen Daten gleichzeitig auf einem Bildschirm präsent zu haben, um unnötiges Suchen und Blättern zu vermeiden. Werden es aber zu viele, dann wird der Bildschirm unübersichtlich — die erforderlichen Felder sind schwer zu finden, die Fehlerwahrscheinlichkeit steigt. Ähnliche Probleme treten auf, wenn man die Frage beantworten will, ob bzw. bis zu welchem Grad die Datenverarbeitung und -verwaltung dezentral erfolgen soll. Abwägen mit Fingerspitzengefühl ist hier gefragt.

Gestaltungskonflikte sind dadurch gekennzeichnet, daß es fließende Übergänge gibt und somit jede Festlegung eine graduelle Bewertung ist und keine Alles-Oder-Nichts-Entscheidung. Erschwerend ist, daß Entscheidungen bezüglich eines Konfliktes Auswirkungen auf die Bewertung anderer Gestaltungskonflikte haben können. Von daher ist es auch unmöglich, alle diese Entscheidungen und Bewertungen zu Beginn einer Systementwicklung zu erfassen und festzulegen — Revisionen sind unvermeidlich.

Wohl mag es für den einen oder anderen Gestaltungskonflikt Faustregeln und Grenzwerte geben, die eine Beurteilung erleichtern, doch können sie grundsätzlich nicht aufgrund mathematischer oder programmiertechnischer Eigenschaften von Software entschieden werden, da es nicht um Produkteigenschaften geht, sondern um Fragen, die die Einbettung eines Systems in den jeweiligen Anwendungsbereich betreffen. Qualitätsmaßstäbe können folglich nicht in dem Maße wie in anderen Ingenieurbereichen über mathematische oder technisch/physikalische Anforderungen beispielsweise an die Materialbeschaffenheit bestimmt werden. Die Beurteilung der Qualität hängt, neben der technischen Zuverlässigkeit und Korrektheit, weitgehend von der Bewertung der Handhabbarkeit, Erlernbarkeit und der Angemessenheit der Funktionalität im Hinblick auf die Arbeitsaufgaben ab.

Diese Bewertung kann aber von den Entwicklern nicht oder nur sehr unzureichend berücksichtigt werden. Zum einen

3 Siehe die ausführliche Darstellung in Bjerknes (1993)

4 Sneed (1983): S. 15

kennen sie das Anwendungsfeld nur aus der Ferne, zum anderen können sie diese Informationen auch nicht durch Befragungen und Erhebungen ermitteln, weil auch die Anwender noch nicht über das praktische Wissen verfügen können, daß sich erst durch den Einsatz des DV-Systems ausbildet.

Seit Beginn der 80er Jahre gibt es zunehmend Bemühungen, diesem Entwicklungsdilemma durch prozeßorientierte Methoden und Verfahren Rechnung zu tragen. Im Vordergrund steht dabei eine stärkere Orientierung auf den wie es so schön heißt „Faktor Mensch". Es geht darum, sich die geistigen Fähigkeiten des Menschen, die es ihm erlauben situationsspezifisch und flexibel neue oder veränderte Anforderungen zu erkennen und in seine Arbeit einzubeziehen, zunutze zu machen. Zur Unterstützung dieser Fähigkeiten ist es folglich notwendig, Techniken und Verfahren bereitzustellen, die auf schnelle und leichte Anpaßbarkeit und Änderbarkeit zielen. Eine *objektorientierte Systementwicklung*, die es erlaubt, universelle und lokal änderbare Bausteine zu entwickeln, soll diesem Ziel ebenso dienen wie die *Entwicklung von Prototypen*, die darauf zielt, möglichst frühzeitig auswertbare Zwischenergebnisse bereitzustellen.[5]

Mit dieser Orientierung auf die geistigen Fähigkeiten des Menschen und die Erfordernisse, die sich aus dem unmittelbaren Entwicklungsprozeß ergeben, wird anerkannt, daß Softwaresysteme die traditionellen Kriterien für geistige Wirtschaftsgüter nur sehr unzureichend erfüllen. Softwareentwicklung ist eben nicht nur ein technischer Konstruktionsprozeß, sondern muß vor allem als sozialer Lernprozeß begriffen und entsprechend methodisch und organisatorisch unterstützt werden.

## Sozialverträgliche Systemgestaltung

Was sich im Bereich der Softwaretechnik als kleine Trendwende abzuzeichnen beginnt, wird zunehmend auch auf der Seite des Einsatzes und der Benutzung von DV-Systemen erkannt: um mit den Folgen und Möglichkeiten des Einsatzes von DV-Systemen angemessen umgehen zu können, sind auch hier kontinuierliche Lernprozesse unvermeidlich. Gesetze, Rahmenvereinbarungen und Verträge sind unerläßlich, um ein gewisses Maß an Sicherheit und eine Grundlage zur Schlichtung in Konfliktfällen zu schaffen, sie können aber die Vielschichtigkeit des Einsatzes von Informationstechnologie am Arbeitsplatz nicht allgemeinverbindlich regeln. Und, was noch stärker wiegt, je später Anforderungen und Erwartungen an die Technik erkannt und gefordert werden, desto schwieriger, ja sogar unmöglich kann es werden, diese noch mit vertretbarem Aufwand zu berücksichtigen.

In dieser Hinsicht hat sich auch im skandinavischen Raum im Rahmen der verschiedenen Ansätze und Projekte für eine soziotechnische Systementwicklung allmählich eine Verschiebung der Handlungs- und Forschungsschwerpunkte ergeben. Dazu ein paar Beispiele.

Bei dem bereits erwähnten Projekt der norwegischen Eisen- und Metallarbeiter in den Jahren 1971—73 ging es noch vorrangig um die Einschätzung der globalen Entwicklung der Informationstechnologie und die daraus resultierenden Konsequenzen für Arbeitnehmer im Hinblick auf Ausbildung und soziale Sicherung der Zukunft. Bei dem 1975 vom Schwedischen Gewerkschaftsbund geförderten Projekt DEMOS stand dann weniger die globale Entwicklung im Vordergrund als vielmehr die Frage, welche Verhandlungsstrategien insbesondere unter Einbeziehung regionaler und lokaler Einrichtungen der Gewerkschaften erforderlich und förderlich sind, um die Mitbestimmungsmöglichkeiten effektiv durchsetzen und nutzen zu können. In der Folge enstand, daß *Schwedische Zentrum zur Erforschung des Arbeitslebens*, das zu je einem Drittel vom Staat, der Industrie und den Gewerkschaften finanziert wurde. Bemerkenswert ist, daß die wissenschaftlichen Probleme, die es hier mit Wissenschaftlern zu bearbeiten galt vorrangig als sozialwissenschaftliche Probleme gesehen wurden; Informatiker und Systementwickler wurden nicht oder nur am Rande einbezogen.

Obwohl bei dem 1977 zusammen mit dem dänischen Gewerkschaftsbund durchgeführten Projekt DUE sich dieses Verhältnis zugunsten der Beteiligung von Technikern besserte, standen im Vordergrund Fragen der Arbeitsgestaltung und die Zusammenarbeit mit den Beschäftigten und Betriebsräten. Die Notwendigkeit der Berücksichtigung arbeitswissenschaftlicher Aspekte bei der Entwicklung von DV-Systemen spiegelte sich in der Folge in einer Begriffsverschiebung wieder: Statt von Software-Entwicklung sprach man jetzt von Systementwicklung, um deutlich zu machen, daß es für eine angemessene Gestaltung nicht ausreichend ist, nur (software-) technische Gesichtspunkte zu betrachten.

Erst bei dem wiederum in Schweden ab 1981 durchgeführten Projekt UTOPIA standen zum erstem Mal technische Fragestellungen im Zentrum des Interesses. Um nicht immer nur der neuen technischen Entwicklung hinterherlaufen zu müssen, was nur nachträgliche und damit in ihrer Wirksamkeit sehr begrenzte Korrekturen ermöglichte, wollte man mit diesem Projekt für das Druckereigewerbe in enger Zusammenarbeit mit Druckern und Setzern frühzeitig Anforderungen an die Gestaltung von DV-Systemen und DV-gestützten Arbeitsplätzen erarbeiten. Obwohl in vielerlei Hinsicht anregend und erfolg-

5 Siehe z.B. Mehl, Reisin, Schmidt, Wolf (1988)

reich, ist das DEMOS-Projekt durch das hohe Innovationstempo der Informationstechnik gewissermaßen überrollt worden.

Doch noch ein anderes Defizit, das mit dem Anfang bis Mitte der achtziger Jahre in Dänemark durchgeführten Projekt MARS systematisch untersucht wurde, zeichnete sich hier bereits ab: der Mangel an systematisch anwendbaren Techniken und Methoden für eine partizipative Systementwicklung. Basierend auf den vielfältigen Erfahrungen, die bei der Durchführung partizipativer Projekte bis dahin nicht nur in Skandinavien gesammelt worden sind, setzte sich die Einsicht durch, daß für das Scheitern und die Mißerfolge nicht allein die Interessenkonflikte maßgeblich sind, die aus dem Grundwiderspruch von Kapital und Arbeit resultieren, sondern eine Fülle weiterer Aspekte, die bisher nicht oder nur unzureichend berücksichtigt worden sind. Um die Gründe für diese Diskrepanz offenzulegen, wurde die eigene Praxis der Systementwicklung selbst zum Gegenstand der Forschung gemacht.[6]

Bezogen auf methodische und organisatorische Ansätze zur partizipativen Systementwicklung entspricht diese Situation in etwa auch dem Stand der Kunst bei uns in der Bundesrepublik. Daß die Lernprozesse bezüglich der Entwicklung und des Einsatzes von DV-Systemen bei weitem noch nicht abgeschlossen sind, sollte jedoch nicht resignierend nur als Defizit gewertet, sondern als programmatische Herausforderung für die Zukunft betrachtet werden.

## Partizipation: Wunsch und Wirklichkeit

Anhand der auf S. 512 dargestellten Lernzyklen werde ich kurz die bisher geschilderte Entwicklung zusammenfassen. Damit will ich deutlich machen, daß trotz der ernüchternden Erfahrungen in der betrieblichen Alltagspraxis sich ein enormer Bewußtseinswandel vollzogen hat. Bei allen Beteiligten ist die Einsicht gewachsen, daß sich die komplexen Veränderungsprozesse, die mit der Einführung und Veränderung von DV-Systemen einhergehen, nur im Rahmen kooperativer Lernprozesse angemessen bewältigen lassen.

So ergab beispielsweise eine umfangreiche in den USA durchgeführte Studie, daß bezüglich der Kostenverteilung beim PC-Einsatz etwa 70 % auf Schulung, Beratung und Service (40 %) sowie Aufwand zum Erlernen (30 %) entfallen und nur 30 % der Gesamtkosten auf Hardware (20 %) und Software (10 %); gerechnet über einen Zeitraum von fünf Jahren.[7] D. h., die Einpassung der Systeme in einen routinisierten Arbeitsablauf kostet weit mehr als die Beschaffung der Geräte und der Software. Hinzu kommt, daß die langfristigen Kosten dieser Lern- und Anpassungsprozesse weit schwieriger zu ermitteln sind als die Abschreibung der technischen Investitionen.

Entgegen der weitverbreiteten Meinung, daß Kommunikation bei der Entwicklung von DV-Systemen größtenteils unproduktive Arbeit sei, legen neuere Untersuchungen zur Kommunikation der Entwickler untereinander nahe, daß ein höherer Kommunikationsaufwand eine bessere Produktqualität zur Folge hat.[8] Zwar gibt es auch Hinweise darauf, daß die Einbeziehung von mehr Beteiligten in den Entwicklungsprozeß zu einer Verschlechterung der Qualität führen kann, doch kann dies durch entsprechend intensive Kommunikation vermieden bzw. ausgleichen werden.[9]

Bezogen auf die Software-Entwicklung beginnt sich zunehmend die Einsicht durchzusetzen, daß umfangreiche Dokumente (Pflichtenhefte, Spezifikationen) allein nicht als Kommunikationsgrundlage ausreichend sind. Erst mit auswertbaren Zwischenergebnissen, beispielsweise der Simulation kritischer Arbeitsabläufe oder der Gestaltung der Benutzungsoberfläche, hat man eine für alle Beteiligten brauchbare Entscheidungsgrundlage. Doch allein die Entwicklung eines Prototypen beispielsweise ist noch nicht ausreichend. Wichtig ist, wie Prototypen ausgewertet werden und die dabei gewonnenen Einsichten in die Revision einfließen. Hier gibt es zwar erste Erfahrungen doch noch wenig systematisiertes Wissen.[10]

Bei der Frage, wie denn insgesamt der Software-Entwicklungsprozeß methodisch und technisch besser unterstützt werden kann, hat man sich lange Zeit auf die Merkmale und Eigenschaften des zu entwickelnden Produktes bezogen. Mit besseren Werkzeugen, neuen Techniken und Beschreibungssprachen sollten die jeweiligen Probleme beseitigt werden. Eine empirische Überprüfung jedoch, ob die dabei zugrundegelegten Annahmen überhaupt zutreffen, hat es nicht gegeben. Erst in den letzten Jahren gibt es einige Untersuchungen, die die Alltagspraxis der Systementwicklung genauer unter die Lupe nehmen, um herauszufinden, warum die Projekte nicht so laufen wie sie laufen sollten.

Hier liegt m. E. das größte Defizit, denn wie ich abschließend am Beispiel des Verhältnisses von Entwicklern und Benutzern deutlich machen will, müssen wir viele Annahmen revidieren, die wir — meist ohne uns darüber im klaren zu sein — bei der Systementwicklung zugrunde legen.

---

6 Die Ergebnisse sind in einem Lehrbuch zusammengefaßt, das bisher aber leider nur in englischer Sprache vorliegt (Andersen et al. 1990).

7 Siehe Klotz (1990)

8 Pasch (1992)

9 Siehe Brodbeck (1993)

10 Siehe Ortlieb, Holz auf der Heide (1993), Holz auf der Heide, Hacker (1991)

## Lernprozesse und Rollenverhalten

Bereits die bei Informatikern so beliebte Annahme, es komme in erster Linie auf die Entwickler-Benutzer-Kommunikation an, ist bezogen auf die alltägliche Praxis eine unzulässige Vereinfachung, wenn nicht gar irreführend. Sie unterstellt bereits stillschweigend, daß es darauf ankomme, bestimmte Probleme durch die Einführung oder Erweiterung eines DV-Systems zu lösen. Diese Sichtweise, die nicht nur von Entwicklern vertreten wird, stellt bereits eine ungeheure Einschränkung dar, weil es nicht mehr möglich ist, die Natur des Problems selbst und die spezifischen Konsequenzen und Alternativen zu thematisieren. Der Konflikt ist vorprogrammiert:

- Die Entwickler haben bereits einen fest vorgegebenen Auftrag und sind nicht mehr willens oder in der Lage, Anforderungen und Wünsche der Beschäftigten zu berücksichtigen, die sich nicht aus dem vorgegebenen Pflichtenheft ableiten lassen. Umgekehrt erwarten sich die Beschäftigten von der Technik z. T. „Lösungen", die entweder generell oder mit den vorhandenen Ressourcen nicht machbar sind. Gemäß einer kürzlich veröffentlichten Studie scheinen solche widersprüchlichen Erwartungen ein weit größeres Problem darzustellen, als z. B. die vielfach zitierten Kommunikationsprobleme.[11]

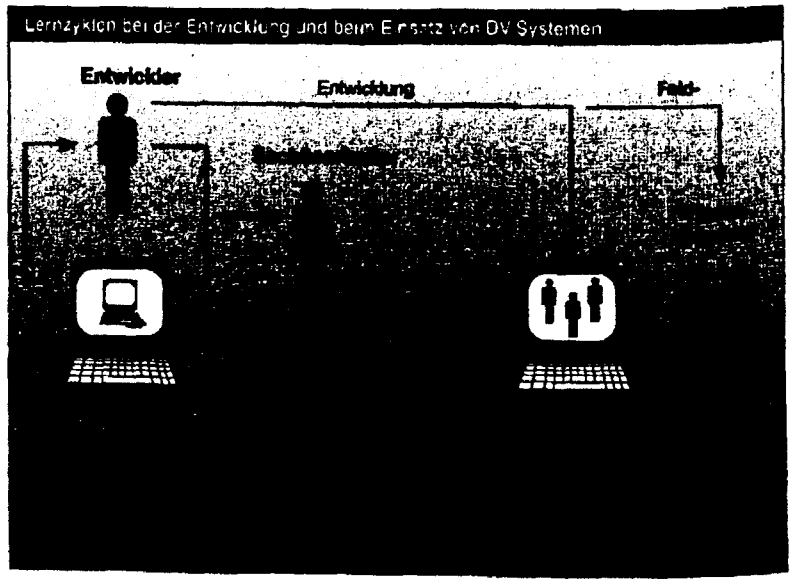
- Sowohl auf Entwicklerseite als auch auf der Seite der Beschäftigten werden grundlegende, den Projektverlauf betreffende Entscheidungen meist nicht von den unmittelbar Beteiligten getroffen, sondern vom mittleren oder höheren Management. Hier dominieren im Allgemeinen Juristen und Betriebswirte ohne spezifische DV-Kenntnisse, die zudem eine große Distanz zum aktuellen Anwendungsproblem haben und die in verteilten Unternehmen und Konzernen noch nicht einmal im anwendenden Betrieb sitzen müssen. Ohne die Einbeziehung von Vertretern dieser Ebene als Akteure in den Gestaltungsprozeß können selbst einvernehmlich erzielte Ergebnisse schnell hinfällig werden.

- In der Regel wird bereits die Tatsache, daß verschiedene Beteiligte und Gruppen halbwegs gleichberechtigt miteinander reden (können) als ausreichend für einen partizipativen Ansatz erachtet. Obwohl es sich um eine offene Lernsituation handelt, die zudem durch Ressourcenknappheit, unsichere Zukunftsaussichten und widersprüchliche Interessenkonstellationen gekennzeichnet ist, was ungeheure Anforderungen an die beteiligten Akteure stellt, findet eine „moderierte Kommunikation" offenbar nicht statt. Sieht man einmal von speziellen Forschungsprojekten ab, ist in Berichten über partizipativ durchgeführte Projekten weder die Rede von der Einbeziehung eines Supervisors (Gesprächspsychotherapeuten, deren Aufgabe die Einleitung und Unterstützung von kritischen Gruppenprozessen ist) oder von Facilitatoren (projektneutrale Person, die darauf achtet, daß der Sachbezug in der Diskussion erhalten bleibt), noch von Mechanismen, wie bestimmte *Funktionelle Rollen* einzelnen Gruppenmitgliedern zugeordnet werden.[12] Wohl aber häufen sich die Berichte, die die mangelnde soziale Kompetenz der beteiligten Akteure beklagen bzw. diese Kompetenz seitens der Ausbildung einklagen.

- Die interdisziplinäre Komponente fehlt bei vielen Partizipationsvorhaben ebenso, wie die Einbeziehung externer Berater. Obwohl bekannt ist, daß sich auch in einem beteiligungsorientierten Vorhaben, eine gewisse Betriebsblindheit einstellt, gibt es kaum systematische Bemühungen, Personen einzubeziehen, die nicht unmittelbar am Vorhaben beteiligt sind und dadurch eine größere Distanz mitbringen, die es ihnen erlaubt Probleme unbefangen zu beobachten. Dies ist um so wichtiger,

11 Siehe Beck (1993)

12 Vgl. Kap. 8 in Pasch (1992). Eine überarbeitete Fassung dieser Arbeit wird im Springer Verlag erscheinen.



als sowohl Entwickler als auch Beschäftigte berechtigterweise dazu tendieren, jede Entscheidung immer auch im Hinblick auf die Konsequenzen für die eigene Arbeit zu beurteilen. Es ist eine Illusion zu glauben, daß Entwickler gleichermaßen ihre eigenen wie auch die Interessen der potentiellen Benutzer im Auge behalten könnten — und zwar auch dann nicht, wenn der Wille dazu vorhanden ist.

- Dasselbe gilt für die Beteiligung von anderen Disziplinen. So sehr es zu begrüßen ist, daß sich Informatiker bzw. Softwareentwickler den Problemen der Organisationsentwicklung und Arbeitsgestaltung gegenüber öffnen, so sehr sollte man darauf achten, wo die Grenzen ihrer Kompetenz jeweils überschritten werden. Für die Einbeziehung anderer Disziplinen ist es jedoch nicht ausreichend, wenn man der eigentlichen Projektarbeit z. B. eine Arbeitsplatzanalyse als ein in sich abgeschlossenes Teilprojekt vorausgehen läßt. Vielmehr sollte dies ein integrierter Teil des Gesamtvorhabens sein. Für eine solche Integration fehlt jedoch bisher noch das methodisch/technische Instrumentarium.

- Schließlich gilt es festzustellen, daß der Erfolg eines Beteiligungsvorhabens maßgeblich davon abhängt, inwieweit alle Beteiligten sich das Vorhaben zu eigen machen. Das Zurückziehen auf tradierte Rollen — beispielsweise der Macher: sagt mir was ihr wollt und ich baue das, oder des Stellvertreters: ich vertrete hier nur die Interessen der anderen Mitarbeiter — ist eine zu defensive Haltung, um damit aktiv gestalten zu können. Wie ein Erfahrungsbericht zeigt, bedarf aktive Gestaltung „eines langfristigen strategischen Konzepts, einer Re-Organistion der Betriebsratsarbeit, einer professionellen Unterstützung von außen und v. a. auch einer intensiven Beteiligung der betroffenen Beschäftigten."[13] Hier aber stoßen alle Beteiligten schnell an die Grenzen ihrer eigenen gewachsenen Organisationskultur, die nur sehr schwer zu überwinden ist.

Die vorstehend aufgeführten Punkte zeigen, daß es noch enormer Anstrengungen bedarf, um eine beteiligungsorientierte Systemgestaltung zur Alltagspraxis werden zu lassen. Speziell am Verhältnis von Entwicklern und Beschäftigten wird deutlich, daß es hier nicht allein mit einer weiteren Integration von Methoden und Techniken getan ist. Vielmehr gilt es, das eigene Selbstverständnis neu zu überdenken wie auch die tradierten Organisationskulturen. Ebenso wie es Überlegungen zu neuen Berufsfeldern gibt, muß man sich Gedanken über neue Rollen und Funktionsbereiche machen.

So wäre zu überlegen, ob denn z. B. die Einrichtung einer *betrieblichen Technologiebeauftragten* eine gewisse Kontinuität und Integration ermöglichen könnte, die gegenwärtig weder von einzelnen Beschäftigten noch von Betriebs- und Personalräten auf Dauer gewährleistet werden kann. Außerhalb der Anwenderbetriebe gilt es neue Verbindungen zu schaffen z. B. zwischen externen Beratern und Softwarehäusern sowie neue Formen einer Kooperation zwischen Technikproduzenten, öffentlichen und privaten Forschungseinrichtungen und Technologieberatungsstellen zu entwickeln und zu erproben. Auch wenn manches von dem heute noch sehr utopisch klingen mag, sollte man sich vor Augen halten, daß die Notwendigkeit zur Organisation sozialer Lernprozesse in der Zukunft eher steigen als abnehmen wird. Entscheidend ist dabei nicht so sehr das Verhältnis einzelner Entwickler zu einzelnen Beschäftigten, sondern die Veränderung der Organisationskultur in der dieser Austausch stattfindet.

## Partizipation in der Arbeitswelt von Morgen

Partizipation hat so viele Facetten, graduelle Unterschiede und qualitativ unterschiedliche Realisierungsmöglichkeiten, daß die Frage, ob man denn mit Beteiligung die betrieblichen Verhältnisse grundlegend verbessern könnte, bereits an der Realität vorbeigeht, denn die Veränderungsprozesse finden ohnehin statt. Vielmehr geht es darum, alle Bereiche, die durch die Entwicklung und den Einsatz von Informationstechnologien berührt werden, einer erweiterten Mitwirkung und Mitbestimmung zu öffnen.

Zum einen gilt es, verstärkt Verantwortung auf die Mitarbeiter zu übertragen, da sie diejenigen sind, die über das Wissen und Können verfügen, Ausnahmen, Sonderfälle und unvorhergesehene Ereignisse situationsangemessen und flexibel handhaben zu können. Zum anderen kommt es darauf an, sich stärker der Dynamik der Entwicklungs- und Veränderungsprozesse zu stellen, da viele Ereignisse und Probleme nicht vorhersehbar sind und demzufolge auch nicht nach einem festen Schema aufgrund starrer Vorgaben ordnungsgemäß behandelt werden können.

Unter Stichworten wie Prototyp-Entwicklung, prozeßorientierte Systemgestaltung, oder effektive Steuerung von Veränderungs- und Anpassungsprozessen (Management of Change, Lean Production) gibt es in der Literatur eine Fülle von Beispielen, denen allen die Erkenntnis gemeinsam ist, daß Lernprozesse ein unvermeidbarer, ja sogar produktiver Faktor sind, den es gilt, technisch, methodisch und organisatorisch geeignet zu unterstützen. Hinzu kommt, daß sich mit dem zunehmenden Einsatz von Arbeitsplatzrechnern und Rechnernetzen, sowie softwareseitig mit benutzerprogrammierbaren bzw. -anpaßbaren Anwendungsmöglichkeiten

13 Schwitalla, Wicke (1991), S. 257

(z. B. Sprachen der 4. Generation für Datenbanken oder Tabellenkalkulation) die technisch bedingten Einschränkungen der Handlungsflexibilität enorm verringert haben.[14]

Nur, Verringerung technischer Erschwernisse und wachsende Einsicht in die Notwendigkeit von Lernprozessen sind zwar entscheidende Voraussetzungen für den Erfolg partizipativer Gestaltung, stellen aber für sich allein genommen noch keine ausreichende Grundlage dar.

Was die Benutzerseite betrifft, so mögen die bisher gemachten Erfahrungen gerade vor dem Hintergrund hochgesteckter Erwartungen an partizipative Projekte eher enttäuschend sein. In der Tat gibt es auch keinen Grund zum Jubeln, wenn man sich die betriebliche Alltagsrealität vor Augen hält. Genauso verfehlt wäre es jedoch, diese Alltagsrealität nur als Scheitern guter Bemühungen zu werten. Denn was die von mir geschilderten Entwicklungen im Bereich der Softwaretechnik wie auch der soziotechnischen Systemgestaltung gleichermaßen deutlich machen, ist die Tatsache, daß hinter der Ernüchterung ein Netz vielfältiger Erfahrungen und Lernprozesse steckt, die sich positiv nutzen lassen. Voraussetzung dafür ist jedoch, daß wir diese Erfahrungen zum Anlaß nehmen, auch die Annahmen und Erwartung zu überdenken, die unsere Vorstellungen von Partizipation prägen.

Zusammenfassend könnte man festhalten, daß das Verhältnis von Entwicklern zu Benutzern noch nie so offen war wie heute, daß es aber entscheidend darauf ankommt, diese Offenheit realistisch auszugestalten, um zu vermeiden, daß sie aufgrund des jeweils nicht Erreichten allzu schnell in Resignation umschlägt. Lernprozesse zu gestalten heißt, den Pfad der Gewißheiten zu verlassen. Und darüber sollten wir uns klar sein: für keinen der an einer partizipativen Systementwicklung Beteiligten ist dies eine leichte Aufgabe.

**Prof. Dr. Reinhard Keil-Slawik**
**Universität Paderborn**

## Literatur

Andersen, N. E., Kensing, E., Lundin, J., Mathiassen, L., Munk-Madsen, A., Rasbech, M., Sørgaard, P.: *Professional Systems Development. Experience, Ideas and Action.* New York: Prentice Hall, 1990

Beck, A.: Benutzerpartizipation aus Sicht von SW-Entwicklern und Benutzern. In: Rödiger, K.-H. (Hg.): *Software-Ergonomie '93.* Stuttgart: Teubner, 1993

Bjerknes, G.: Systementwicklung in Skandinavien. *Arbeitsrecht im Betrieb,* Nr. 1, 1993, S. 20—25

Brodbeck, E: Warum es sinvoll ist, Kommunikation und Kooperation in Software-Entwicklungsprojekten verstärkt zu kultivieren: Ergebnisse aus einer empirischen Untersuchung. In: Rödiger, K.-H. (Hg.): *Software-Ergonomie '93.* Stuttgart: Teubner, 1993

Holz auf der Heide, B., Hacker, S.: Prototyping in einem Designteam: Vorgehen und Erfahrungen bei einer benutzerorientierten Software-Entwicklung. In: Ackermann, D., Ulich, E. (Hg.): *Software-Ergonomie '91.* Stuttgart: Teubner, 1991

Klotz, U.: Die Wende in der Bürokommunikation. *Office Management,* Teil 1 Heft 6 und Teil 2 Heft 7—8, 1990

Mehl, W.-M., Reisin, E.-M., Schmidt, G., Wolf, G.: Prototyping als Technik im Kontext partizipativer Systementwicklung. In: Brückers, W., Meyer, N. (Hg.): *Arbeit, Technik und berufliche Bildung im Metallbereich.* Köln: vgs, 1988

Naur, P., Randell, B. (Eds.): Software Engineering. Scientific Affairs Division, Nato: Brussels, 1969

Ortlieb, S., Holz auf der Heide, B.: Benutzer bei der Software-Entwicklung angemessen beteiligen — Erfahrungen und Ergebnisse mit verschiedenen Konzepten. In: Rödiger, K.-H. (Hg.): *Software-Ergonomie '93.* Stuttgart: Teubner, 1993

Pasch, J.: Dialogischer Software-Entwurf. Dissertation, Forschungsberichte des FB Informatik, Nr. 92—4, TU Berlin, 1992

Rauterberg, M.: Partizipative Konzepte, Methoden und Techniken zur Optimierung der Softwareentwicklung. In: Brödner, P., Simonis, G., Paul, H. (Hg.): Arbeitsgestaltung und partizipative Systementwicklung. Opladen: Leske & Buderich, 1991

Schmitz, K.: Mitbestimmung durch Eigengestaltung. Arbeitsrecht im Betrieb, Nr. 1, 1993, S. 25—30

Schwitalla, U., Wicke, W.: Beteiligung in einem Versicherungsunternehmen. Ein Fallbeispiel. In: Brödner, P., Simonis, G., Paul, H. (Hg.): *Arbeitsgestaltung und partizipative Systementwicklung.* Opladen: Leske & Buderich, 1991

Sneed, H.: Software-Qualitätssicherung für kommerzielle Anwendungssysteme. Köln-Braunsfeld, 1983

14 vgl. Schmitz (1993)