Softwareentwicklung als Prozeß der Arbeitsstrukturierung[1]

Friedrich Weltz, Veronika Lullies, Rolf G. Ortmann
SPG - Sozialwissenschaftliche Projektgruppe, München

Zusammenfassung

Das Verständnis des besonderen Charakters von Softwareprojekten wie der Anforderungen, mit denen sich deren Gestaltung auseinanderzusetzen hat, erschließt sich nicht zuletzt aus dem Anwendungsbezug der entwickelten Produkte. Softwareentwicklung heißt, technisches Potential für die Bewältigung von Aufgaben verfügbar zu machen. Software liefert jedoch nicht nur technische Lösungen, sie strukturiert zugleich das Feld, in dem sie Anwendung findet. Dort, wo die Aufgaben von Menschen wahrgenommen werden - und dies ist ja noch immer zum weitaus größten Teil der Fall - heißt dies: Softwareentwicklung ist zugleich Technikgestaltung und Arbeitsstrukturierung.

Der Einsatz von Software kann zu Veränderungen in der Ausführung der Arbeit, ja selbst in der Definition der Arbeitsaufgaben führen. Die Entwicklung benutzer- und aufgabenorientierter Software ist also Teil eines Prozesses, in den sowohl softwarebezogene wie aufgabenbezogene Aspekte und Aktivitäten eingehen und aufeinander abgestimmt werden müssen. Sie kann gesehen werden als ein wechselseitiger Anpassungsprozeß zwischen dem software-technischem Potential und der Strukturierung der Arbeitsaufgaben.

Die optimale Lösung erfordert also Anpassung von beiden Seiten: von softwaretechnischer Seite, die möglichst maßgeschneiderte Berücksichtigung der Anforderungen,

---

[1] Der Beitrag entstand im Rahmen des Projektes IPAS (Interdisziplinäres Projekt zur Arbeitssituation in der Softwareentwicklung). Das Projekt besteht aus einem sozialwissenschaftlichen Teilprojekt (SPG - Sozialwissenschaftliche Projektgruppe, München, Projektleiter Prof. F. Weltz), einem Teilprojekt Informatik (Universität Marburg, Projektleitung Prof. W. Hesse) und einem arbeitspsychologischen Teilprojekt (Universität München, Projektleitung Prof. M. Freese). Das Projekt wird vom Bundesministerium für Forschung und Technologie (BMFT) im Rahmen des Programmes "Arbeit und Technik" gefördert.

die sich aus der Arbeitsaufgabe ergeben; auf der Seite der Gestaltung der Arbeitsaufgabe die Berücksichtigung des technischen Unterstützungspotentials.

Dort, wo Software Arbeit im kooperativen Zusammenhang unterstützt - und dies ist faktisch bei allen betrieblichen Anwendungen der Fall - hat ihr Einsatz meist über den einzelnen Arbeitsplatz hinausreichende Auswirkungen. Sie kann Arbeitsabläufe, die Arbeitsteilungs- und Kompetenzstrukturen verändern. Software ist in diesem Fall nicht nur ein Arbeitsmittel, sondern ein Organisationsinstrument. Ihrer Entwicklung wächst damit eine betriebspolitische Signifikanz zu, d.h. sie berührt in ihrem Anwendungsfeld bestehende Einfluß- und Interessenverteilungen.

Dies alles ist natürlich von Projekt zu Projekt in sehr unterschiedlichem Maße relevant, je nach den möglichen Auswirkungen, den der Einsatz der Software im Anwendungsfeld haben kann. Tendenziell gilt: In dem Maße, wie die Verfahren der Computerunterstützung zugleich umfassender wie auch flexibler nutzbar werden, gewinnt die Anwendungssituation als Gestaltungsdimension an Bedeutung und muß im Prozeß der Softwareentwicklung mit berücksichtigt werden.

Diese arbeitsstrukturierende und betriebspolitische Bedeutung von Software hat nun ihrerseits Konsequenzen für den Prozeß ihrer Entwicklung selbst: Sie stellt Anforderungen an dessen Gestaltung, sie definiert deren Rahmenbedingungen. Anforderungen ergeben sich etwa bezüglich der Konsensbildung, d.h. des Ausgleichs divergierender Interessen, die mit der Entwicklung und dem Einsatz der Software verbunden sind. Die Notwendigkeit zu solchem Interessenausgleich ergibt sich, wie bei allen Entwicklungsprozessen, aus den Wünschen, Zielvorstellungen und Erfahrungen, die Entwickler einerseits, Auftraggeber bzw. Nutzer andererseits in den Entwicklungsprozeß einbringen. Bei Softwareentwicklung stehen tendenziell technikimmanente Aspekte im Vordergrund (etwa geringe Beanspruchung von Speicherkapazität, "Kompatibilität", "professionelle" Qualitätskriterien, "Eleganz" der Programme), bei den Nutzern dagegen aufgaben- und arbeitsbezogene Aspekte (etwa Erleichterung bei der Aufgabenerledigung, Belastungsreduzierung, Erweiterung oder Einschränkung des Gestaltungsspielraumes).

Abweichende Interessen können auch bestehen innerhalb der Softwareentwicklung: etwa zwischen hierarchisch übergeordneten und ausführenden Positionen oder zwischen den in den einzelnen Phasen des Entwicklungsprozesses Beschäftigten. Divergenzen können sich aber vor allem ergeben zwischen einzelnen Gruppierungen von Nutzern. Viele Softwareentwicklungen berühren - zumindest potentiell - beste-

hende Arbeits- und Einflußstrukturen in Unternehmen und zwar umso mehr, als der Einsatz von Software zu Veränderungen in den Aufgaben- und Arbeitsstrukturen führt bzw. führen könnte. Allein die Möglichkeit solcher Veränderungen verleiht Softwareprodukten eine betriebspolitische Qualität, d.h. sie werden auf Interessenposition bezogen.

Die Entwicklung angemessener Softwareprodukte setzt die Berücksichtigung dieser unterschiedlichen Interessen voraus. Der Prozeß der Entwicklung von Software muß daher immer auch ein Prozeß der Konsensbildung sein.

Auf dem Hintergrund dieser Interessendivergenzen sind nun wiederum die Lernprozesse zu sehen, die faktisch in jedem Softwareentwicklungsprojekt notwendig sind. Die Entwicklung aufgaben- und nutzungsgerechter Software setzt Wissen voraus:

- bei den Softwareentwicklern vor allem über die Aufgaben, die Bedingungen und Anforderungen der Arbeitssituation, die durch Softwareprodukte zu unterstützen sind;
- bei den Auftraggebern bzw. Nutzern vor allem über das Leistungspotential und die Leistungsbeschränkungen der Software, die sich daraus ergebenden Möglichkeiten und Konsequenzen für die Gestaltung ihrer Aufgaben.

Dieses Wissen ist im Regelfalle nur unvollständig vorhanden, Lernprozesse sind erforderlich. Wesentlich ist dabei, daß diese Lernprozesse meist die Grenzen des eigenen Fachwissens überschreiten müssen. Informationsvermittlung spielt also eine große Rolle: aus dem Anwendungsbereich in den Entwicklungsbereich und umgekehrt sowie im Entwicklungsbereich selbst.

Solche Informationsvermittlung stößt auf strukturelle Schwierigkeiten, etwa das jeweilige Fachwissen und den Fachjargon auf beiden Seiten, Zeitmangel, Unterschätzung der Wissensanfordernisse, vor allem aber, daß Informationsvermittlung häufig interessengesteuert erfolgt.

Im Regelfalle spielen in diesen Lernprozessen die jeweiligen "Kontrahenten" eine zentrale Rolle. Für die Softwareentwickler sind die Auftraggeber bzw. Nutzer wesentliche Informationsquelle, wie umgekehrt sie für diese. Der (potentielle) Informant bringt spezifische Interessen ein, ist Partei. Der Systementwickler mag an bestimmter Hard- oder Softwarekonfiguration interessiert sein, der Nutzer an bestimmten arbeitsorganisatorischen Lösungen (häufig Beibehaltung des Status quo). Dies bedeutet, daß zu den allgemeinen Schwierigkeiten, die in diesen Lernprozessen über-

wunden werden müssen (z.B. die Komplexitäten technischer Details), noch erschwerend hinzukommt, daß die Lernprozesse eingebettet sind in das Feld divergierender Interessen, in das faktisch jede Softwareentwicklung stößt. Diesen Schwierigkeiten gegenüber erweisen sich die formalisierten Instrumente der Wissensvermittlung (Ist-Analysen) meist als inadäquat.

Erschwert werden diese Lernprozesse auch durch die Abstraktheit des Gestaltungsgegenstandes. Darin besteht ein wesentlicher Unterschied der Softwareentwicklung zu anderen Entwicklungsprozessen. Das Produkt, sein Leistungsprofil, die Anforderung, die an es gestellt werden, sind nur in abstrakten Termini beschreibbar. Diese sind nicht nur meist in einer großen Bandbreite interpretierbar, sie haben unter Umständen für einzelne Beteiligte gar keine oder sogar eine falsche Bedeutung.

Natürlich stellt sich dieses Problem in gewisser Weise allen Entwicklungsprozessen in ihren Frühphasen. Anders als im Softwareentwicklungsprozeß allerdings steht meist die Möglichkeit, relativ rasch diese Abstraktheit - etwa durch Skizzen, Vorentwürfe etc. - zu reduzieren und damit auch den Spielraum des Mißverstehens, unterschiedlicher Interpretationen etc. zu verringern.

Diese Abstraktheit erschwert den Austausch der notwendigen Informationen zwischen Softwareentwicklern und Anwendern und zwar gleichermaßen in beiden Richtungen. Nicht nur ist es für die Anwender schwierig, den besonderen Charakter der Anforderung, die an die Technik gestellt werden, dem Softwareentwickler zu vermitteln, auch umgekehrt ist es schwierig, das Leistungspotential der Technik in vollem Maße anschaulich und nachvollziehbar darzustellen. Vor allem dürfte die Abstraktheit des Gestaltungsgegenstandes die systematische Ermittlung aller Interdependenzen, Besonderheiten und Abhängigkeiten, die bei der jeweiligen Anwendung der Technik zu berücksichtigen sind, erschweren wie auch den Wunsch der Umwandlung von allgemeinen Zielen und Wünschen in präzise, eindeutige Vorgaben für die Softwareentwicklung.

All dies schlägt umso mehr zu Buche, als Softwareprodukte zunehmend komplex werden: durch ihren Umfang, durch die Zahl der zu berücksichtigenden Variablen, durch die Integration von Teilsystemen, durch die Bedeutung der Kompatibilität mit bereits eingesetzter Hard- und Software. Entsprechend wird auch die Durchführung von Entwicklungsprojekten zunehmend komplexer. Zu dieser Komplexität trägt insbesondere auch der Bezug zur Anwendungssituation bei.

Man könnte von einer strukturellen Komplexität sprechen: Softwareimmanente Gegebenheiten, das technische Umfeld, die qualifikatorischen und motivationalen Voraussetzungen bei den Nutzern, die Struktur des Anwendungsfeldes und der Aufgaben müssen gleichermaßen berücksichtigt und aufeinander bezogen werden.

Die Anforderungen, die sich aus dieser strukturellen Komplexität ergeben, werden verstärkt durch die prozessuale Komplexität des Softwareentwicklungsprozesses. Diese ergibt sich vor allem aus der Notwendigkeit zu ständigen Feedbackprozessen und Korrekturschleifen. Es gilt, die im fortschreitenden Entwicklungsprozeß gemachten Erfahrungen auf die vorangegangenen Entwicklungsschritte zurückzubeziehen, was möglicherweise zu "retrospektiven" Korrekturen, etwa in der Definition der Zielvorgaben, führen kann.

Aus dieser Komplexität ergeben sich Anforderungen an die Gestaltung des Softwareentwicklungsprozesses: an die Regelung der Arbeitsteilung, der Kooperationsbezüge und auch ihres Ablaufs. Dies gilt sowohl innerhalb des Entwicklungsbereiches als auch für die Beziehung zum Anwendungsfeld.

Dabei zeichnet sich ein Spannungsverhältnis ab zwischen der Notwendigkeit zu einer zunehmenden Arbeitsteiligkeit in der Organisation der Entwicklungsprozesse und der Notwendigkeit zur Integration. Fragen der Verteilung und Abgrenzung der Zuständigkeiten und Aufgaben sowohl im Entwicklungsbereich wie auch zwischen Entwicklungs- und Anwendungsbereich bekommen zentrales Gewicht. Ein Spannungsverhältnis besteht auch in der prozessualen Dimension zwischen der Notwendigkeit zu verbindlichen Zielsetzungen und Vorgaben und der Notwendigkeit zu ständigen Korrekturen im Laufe des Entwicklungsprozesses.

Fazit

Die Gestaltung von Softwareentwicklungsprojekten unter der Zielsetzung der Entwicklung menschen- und aufgabengerechter Software steht vor der Anforderung, einerseits der effektiven Organisation des technischen Entwicklungsprozesses, andererseits der wechselseitigen Abstimmung von Leistungspotential der Technik und der Möglichkeiten zur Umgestaltung der Arbeit, die sich durch diese eröffnen, gerecht zu werden.

Dies macht Lernprozesse bei Entwicklern und Anwendern erforderlich und setzt Konsensbildung voraus. Diese ihrerseits erfordern Reduzierung der Unbestimmtheit des Gestaltungsgegenstandes wie auch die Bewältigung der strukturellen und prozessualen Komplexität des Entwicklungsvorhabens. Diesen Anforderungen werden die herkömmlichen Konzepte von Softwareergonomie und von Nutzerpartizipation kaum gerecht, die Anforderungen verweisen vielmehr auf ein grundsätzliches Verständnis von Softwareentwicklung als Prozeß der Arbeitsstrukturierung.

In der wissenschaftlichen Literatur wie in der Praxis ist in den letzten Jahren eine zunehmende Hinwendung auf ein solches Verständnis zu erkennen. Vielfach allerdings wird der Prozeß der Entwicklung von Software noch sozusagen eindimensional gesehen: als Gestaltung von Technik. Die Bezeichnung "Softwareengineering" drückt diese Sichtweise sehr präzise aus, es geht um die ingenieursmäßige Optimierung der Entwicklungsprozesse, bei der natürlich soziale Aspekte - "Motivation", "Akzeptanz", "Nutzerpartizipation" - mit berücksichtigt werden, die aber selbst nicht als sozialer und "betriebspolitischer" Prozeß behandelt wird; dabei ist der Gegenstand des Prozesses nicht die Gestaltung von Technik, sondern letztlich von Arbeit.