Grundwissen: Software-Ergonomie

Lothar Bräutigam

Lothar Bräutigam
sovt – Sozialverträgliche Technikgestaltung
Beratung & Seminare
Herdweg 10a,
D-64285 Darmstadt
Fon (06151) 62 602 oder (0172) 61 83 108
Fax (06151) 62 606
http://www.sovt.de
E-Mail: lothar.braeutigam@sovt.de

ergo-online: http://www.sozialnetz-hessen.de/ergo-online
Fachinformationsdienst Arbeit & Gesundheit

Dass Computersoftware nicht nur vom technologischen Aspekt, sondern auch bezüglich der Schnittstelle zum Menschen hin optimiert werden muss, ist eine Binsenweisheit. Dies gilt nicht nur für die Standardsysteme wie Office – die allenthalben eingesetzt werden, sondern ganz besonders auch für die Eigenentwicklungen, die in Kliniken zum Einsatz kommen. Erfahrungen aus der Praxis zeigen, dass es mit der Aufstellung eines Computerarbeitsplatzes und der Einrichtung der Software nicht getan ist: Die Einhaltung der entsprechenden Verordnungen sowie die ergonomische Anpassung stehen im Vordergrund. Pr-InterNet wird in den nächsten Ausgaben in loser Folge über ergonomische Forderungen und Lösungen aus der Praxis berichten. Den Anfang macht in dieser Ausgabe die Software-Ergonomie

Übersicht

- Ziel der Software-Ergonomie ist die Anpassung der Eigenschaften von Software an die psychischen Eigenschaften der damit arbeitenden Menschen.
- Unzureichende software-ergonomische Gestaltung führt zu erhöhten psychischen Belastungen.
- Software-Ergonomie gehört zu den Mindestanforderungen, die die Bildschirmarbeitsverordnung an die Gestaltung von Bildschirmarbeit stellt.
- Die Normenreihe DIN EN ISO 9241, Teile 10–17 enthält konkrete Anforderungen an die ergonomische Gestaltung von Software.

Software-Ergonomie: ein wichtiges Thema für alle, die täglich am Bildschirm arbeiten. PC's mit grafischen Benutzungsoberflächen, insbesondere nach dem Windows-Standard, haben die Bedienung der Programme sicher vereinfacht, die Probleme aber nicht aus der Welt geschafft.

Was ist Software-Ergonomie?

Ähnlich wie bei der Hardware-Ergonomie geht es um die Anpassung von technischen Systemen – hier Software – an menschliches Arbeitshandeln (und nicht umgekehrt!).

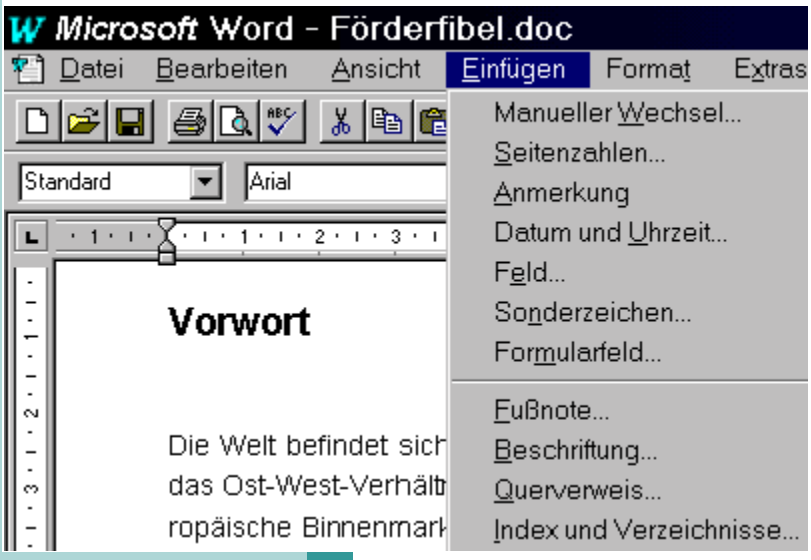
Definition

Ziel der Software-Ergonomie ist die Anpassung der Eigenschaften eines Dialogsystems an die psychischen Eigenschaften der damit arbeitenden Menschen. Laut der offiziellen Definition der internationalen ISO-Norm verwendet Ergonomie wissenschaftliche Erkenntnisse, um Arbeitsaufgaben, Arbeitsumgebungen und Produkte an die physischen und mentalen Fähigkeiten und Grenzen von Menschen anzupassen. Hierbei soll Gesundheit, Sicherheit, Wohlbefinden und Leistungsvermögen verbessert werden.

Schlüsselwörter
Software-Ergonomie
Bildschirmarbeitsplatz
Software-Entwicklung
Software-Prototyping
EDV

Zur Relevanz

Unzureichende Software-Gestaltung führt zu erhöhten psychischen Belastungen, zu Kopfschmerzen und Augenflimmern, zu Streß und Zeitdruck und bei längerer Dauer auch zu körperlichen Beschwerden. Aus diesem Grund gehört die Software-Ergonomie jetzt auch zu den rechtsverbindlichen Mindestanforderungen, die bei Bildschirmarbeitsplätzen eingehalten werden müssen. Die gesetzliche Grundlage hierfür bildet die neue Bildschirmarbeitsverordnung, die am 20.12.1996 in Kraft getreten ist.



Woran muß sich Software- Gestaltung orientieren?

- Art und Weise menschlicher Informationsverarbeitung wie Kurzzeitgedächnis, Metaphern, Farbwahrnehmung, ...
- Aufgaben der BenutzerIn die mit Software-Unterstützung verrichtet werden sollen.

Die Gestaltungs- Kriterien

Leitlinien zur Gestaltung der Software, und zwar von Benutzeroberfläche, Zeichenanordnung, Farben, Menüs, Masken und Dialogen sind in der internationalen Normreihe DIN EN ISO 9241 festgelegt.

Wichtig ist dabei die Norm DIN EN ISO 9241, Teil 10. Sie legt „Grundsätze der Dialoggestaltung“ fest:

- Aufgabenangemessenheit
- Selbstbeschreibungsfähigkeit
- Erwartungskonformität
- Steuerbarkeit
- Fehlertoleranz
- Individualisierbarkeit
- Lernförderlichkeit

Beispiel

Herr Mayer sitzt vor seinem PC. Der meldet plötzlich: “*Unknown error # 101. Process aborted. Core dumped.*” Auch nach eingehender Beratschlagung mit seinem Kollegen ist nichts mehr zu machen. Die einzige Lösung, die bleibt: Strom aus, Strom an. Die eingegebenen Daten der letzten halben Stunde sind allerdings verloren, Überstunden drohen mal wieder.

Prototyping

Viele Software-Entwicklungsprojekte scheitern, weil sie nach einem ungeeigneten Verfahrensmodell und ohne ausreichende Beteiligung der Anwender abgewickelt werden.

Prototyping ist ein (iteratives) Verfahren der Software-Entwicklung, bei dem die wesentlichen Schritte mehrfach durchlaufen werden, bis die gewünschte Produktqualität erreicht ist.

Ein wesentlicher Grundsatz ist die Einbeziehung der späteren Anwender. Anwender testen Zwischenergebnisse der Programmentwicklung in Form von Prototypen und erarbeiten Verbesserungsvorschläge, die in den folgenden Projektphasen das Programm verbessern helfen. Im Prototypingverfahren lassen sich die üblichen Kommunikationsprobleme zwischen Software-Entwicklern und späteren Programmnutzern weitgehend vermeiden. Durch Prototyping entwickelte Programme zeichnen sich in der Regel durch bessere Funktionalität und höhere software-ergonomische Qualität aus.

Der Problemfall Software-Ergonomie und die Ursachen

Software wird häufig mit fachlichen Mängeln entwickelt, und ihre Bedienung ist oft viel zu kompliziert. Die Anwender sind die leidtragenden: sie müssen tagtäglich mit software-ergonomisch mangelhaften Programmen arbeiten. Und diese Resultate der Software-Entwicklung stellen sich ein, obwohl viele Personen, auch Anwender und Fachexperten, beteiligt werden. Man sitzt in vielen Meetings zusammen und bespricht die fachlichen Anforderungen, setzt sie dann in technische Spezifikationen um, produziert dabei umfangreiche Werke (dicke Ordner), bevor sie das Programmierteam in langer Arbeit in das neue Programm umsetzt.

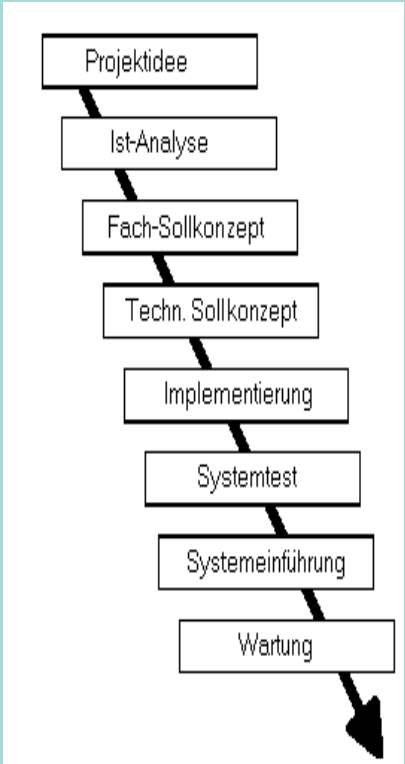


Abb. 1: Phasenmodell

Was sind die Ursachen?

Für die häufig schlechte Qualität der entwickelten Programme sind in der Regel nicht die Projektmitarbeiter verantwortlich, sondern es gibt dafür strukturelle Gründe. Auch heute wird Software noch oft nach dem sog. linearen Phasenmodell entwickelt (Abb. 1).

Eine Projektphase folgt der anderen ohne Wiederholungen. Die Schnittstellen zwischen den Projektphasen bilden in der Regel dicke Papiere, wie Pflichtenheft oder fachliches Feinkonzept, die das Ergebnis der abgeschlossenen und den Auftrag für die folgende Projektphase enthalten.

Problem: keine geplanten Rückkopplungen

Bei komplexeren Entwicklungsvorhaben zeigt sich dieses Modell als nicht mehr zeitgerecht und überfordert: Die Anforderungen an neue Programmsysteme sind heute oft so umfangreich und komplex, daß sie zu Beginn gar nicht vollständig und detailliert bestimmt werden können.

Beim linearen Phasenmodell gibt es keine geplanten Rückkopplungen oder Korrekturschleifen zwischen den einzelnen Phasen. Eine Revidierbarkeit oder Rückholbarkeit von getroffenen Entscheidungen ist im Modell nicht vorgesehen.

Fehler, die schon bei der Anforderungsdefinition entstehen, werden in den späteren Projektphasen nicht erkannt bzw. nicht grundsätzlich behoben. Sie haben aber die weitreichendsten Folgen: das entwickelte Programm wird nicht mehr qualitativ gut, sondern allenfalls „lauffähig“.

Die Zwischenprodukte, die die Übergänge zwischen den einzelnen Projektphasen markieren, sind meist sehr umfangreiche, stark formalisierte Dokumente. Sie sind von den späteren Anwendern nur schwer zu prüfen, da sie ihre Abstraktionsfähigkeit überfordern (insbesondere die Dokumente fachliches und DV-technisches Feinkonzept).

Da die Implementierungsphase nicht durch Zwischentests der Anwender unterbrochen wird, ist das neue Programm für sie vor der Fertigstellung nicht prüfbar. Die software-ergonomische Qualität liegt weitgehend in der Verantwortung der Programmierer, und da liegt sie falsch. Denn sie kennen die spätere Nutzungssituation meist nur ungenügend. So kommt es zu Mängeln vor allem beim software-ergonomischen Kriterium der Aufgabenangemessenheit (vgl.DIN EN ISO 9241, Teil 10).

Problem: Anwenderbeteiligung

Das grundlegende Problem besteht in der unzureichenden Anwenderbeteiligung innerhalb des linearen Phasenmodells. Die Anforderungen an das zu entwickelnde Programm werden zwar zu Beginn unter Beteiligung der Anwender definiert. Die Umsetzung wird von ihnen aber nicht während der Projektlaufzeit, sondern erst nach Abschluß der Programmierarbeiten überprüft.

Bis zum Ende der Phase „Implementierung" findet Anwenderbeteiligung im linearen Phasenmodell nur anhand von Papieren statt, die das zu entwickelnde Programm beschreiben.

Und das muß scheitern:

Die Interessenlage von DV-Experten unterscheidet sich von der der Anwender. Während diese an Programmen interessiert sind, die einfach zu bedienen sind und ihnen die Arbeit erleichtern, ist für Informatiker eher eine elegante und einfache Programmierung interessant. Im herkömmlich organisierten Entwicklungsprozeß haben die DV-Experten die besseren Chancen sich durchzusetzen.

Anwender und DV-Experten sprechen keine gemeinsame Sprache. Man versteht nur wenig vom Fachgebiet und der Begriffswelt des anderen. Informatiker verklausulieren ihre Sprache häufig mit Anglismen und informationstechnischen Fachbegriffen und haben aufgrund von Termindruck meist wenig Zeit zur Auseinandersetzung mit Anwendern.

Dieses Kommunikationsproblem wird dadurch verschärft bzw. unlösbar, daß es keine anschaulichen Zwischenprodukte gibt, auf die sich beide Seiten – Anwender und DV-Experten – beziehen könnten. Formalisierte Dokumente (wie ein DV-technisches Feinkonzept) sind dazu ungeeignet, sie sind zu komplex, weisen keinen Bezug zum Erfahrungswissen der Anwender auf und sind einer Überprüfung, ob das zu erstellende Produkt den Arbeitsprozeß unterstützen kann, nicht zugänglich.

Notwendig: Kooperative Systemgestaltung

Gleichberechtigte Systemgestaltung durch DV-Experten und Anwender

Um diese Fehlentwicklungen zu vermeiden, muß die Anwenderbeteiligung integraler Bestandteil der Software-Entwicklung sein: von Beginn an, gleichberechtigt und während des gesamten Prozesses. Eine solche Form der Beteiligung wird als kooperative Systemgestaltung bezeichnet.

Prozeßorientierte Perspektive betonen

Das zu entwickelnde Programm entsteht innerhalb eines Arbeits-, Lern- und Kommunikationsprozesses aller Beteiligter, der sich über alle Projektphasen erstreckt. Die Qualität des Endprodukts wird zu einem Großteil von der Art und Weise, wie diese Prozesse vollzogen werden, bestimmt.

Es ist allerdings mehr erforderlich als die Anwender nur zu beteiligen. Notwendig ist ein geeigneter methodischer Rahmen, der Beteiligung unterstützt und die Lern- und Kommunikationsmöglichkeiten zwischen Anwendern und Entwicklern fördert.

Prototyping oder Versionenkonzept

Die Fehler des linearen Phasenmodells vermeidet ein rückgekoppeltes, iteratives Verfahrensmodell, in dem sich die Phasen Analyse, Planung/Entwurf, Implementierung und Systemeinführung sowie produktive Nutzung (Test) zyklisch wiederholen. Die Weiterentwicklung der Anforderungen und die Bewertung der geleisteten Arbeit erfolgt so permanent während eines kontinuierlich laufenden Entwicklungsprozesses unter Einbeziehung der Anwender.

Auch die im April 1998 verabschiedete Norm DIN EN ISO 13407 (benutzerorientierte Gestaltung interaktiver Systeme) beschreibt eine solche Vorgehensweise anhand von Prototypen.

In jedem Projektzyklus wird eine lauffähige Programmversion, ein Prototyp, spezifiziert, entwickelt und getestet. Der Test sollte in der konkreten Arbeitssituation am normalen Arbeitsplatz erfolgen. Dadurch werden die Anwender in die Lage versetzt, die bisherigen Anforderungen und ihre programmtechnische Realisierung dahingehend zu überprüfen, ob ihre Arbeit damit effektiv und effizient unterstützt wird. Aus jeder Testphase ergibt sich somit eine überarbeitete und präzisierte Anforderungsdefinition für die folgende Programmversion, die durch Weiterentwicklung aus der vorhergehenden entsteht. Prototypen, die später durch ein völlig neu programmiertes Endprodukt ersetzt werden, sollten vermieden werden. Sie können allerdings ihre Berechtigung haben, wenn verschiedene Gestaltungsalternativen überprüft werden sollen.

Anstelle dicker Projektpapiere stellt der Prototyp ein anschauliches und erfahrbares Zwischenprodukt dar. In jeder Projektphase wird die bisher entwikkelte Programmversion korrigiert, verfeinert, weiterentwickelt, bis ein für die Anwender akzeptables Produkt entwickelt wurde.

Der erste Prototyp

Wichtig beim Projektbeginn ist, daß nicht bereits das fertige Endprodukt in allen Einzelheiten festgelegt wird und daß dies von allen Projektbeteiligten verstanden und akzeptiert wird.

Die Erstellung des ersten Prototypen kann nach dem gewohnten Phasenkonzept abgewickelt werden. Repräsentativ ausgewählte Anwender begleiten dabei den gesamten Entwicklungsprozeß innerhalb einer Projektgruppe. Vorab-Ergebnisse werden sobald wie möglich vorgestellt und mit den beteiligten Anwendern diskutiert. Bei diesen Zwischentests werden in der Regel bereits eine Vielzahl von Änderungswünschen, Ergänzungen und Verbesserungsvorschlägen geäußert. Diese können sofort implementiert werden oder aber im nächsten Prototyp realisiert werden.

Die Spezifikation und Realisierung jedes Prototypen kann als (finanziell) eigenständiges Teilprojekt erfolgen. Es muß allerdings schon zu Projektbeginn die Erstellung zweier oder mehrerer Prototypen einkalkuliert werden.

Mehrere Prototypen auf dem Weg zum Ergebnis

Nach Realisation des ersten Prototypen wird er in einem Feldtest, d.h. in der Realumgebung begutachtet. Als Resultat ergibt sich die Spezifikation des zweiten Prototypen, und der Prozeß beginnt von neuem. Dieser Zyklus sollte in der Regel nicht mehr als drei oder viermal durchlaufen werden, jedoch solange dauern, bis ein befriedigendes Ergebnis erreicht ist.

Die einzelnen Verfahrensschritte beim Prototyping werden in einem gesonderten Dokument beschrieben.

Mangelnde Termintreue und höhere Kosten beim Prototyping?

Die häufigsten Gegenargumente

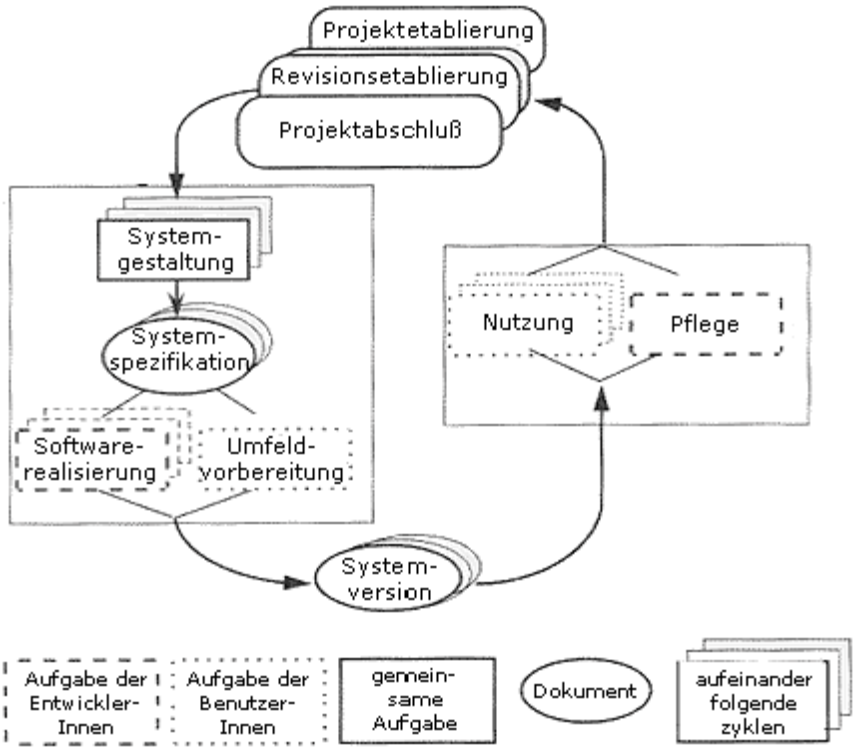
Höhere, nicht planbare Kosten und längere, ebenfalls nicht planbare Entwicklungszeiten, das sind die häufigsten Gegenargumente, die dem Prototyping vorgehalten werden. Darauf komme es aber an, insbesondere in der öffentlichen Verwaltung, wo Software-Projekte nach sog. BVB-Verträgen auf der Basis von Festpreisen abgewickelt werden müssen.

Dem ist entgegen zu halten, daß gerade bei der bisherigen Software-Entwicklung in der Regel weder die veranschlagten Kosten noch die geplanten Zeiten eingehalten wurden.

Es darf aber nicht übersehen werden, daß die Projektbegleitung durch die Anwender und die Tests der Prototypen einen nicht zu unterschätzenden Zeitaufwand verursachen. Aber: jede hier investierte Minute spart auf der anderen Seite Zeit bei der Anwendung des fertigen Produktes (d.h. Arbeitszeit der Anwender) und verringert die Notwendigkeit weiterer Nachbesserungen nach Projektabschluß. Werden wirklich alle Kostenanteile berücksichtigt, so schneidet das Prototyping- Verfahren günstiger als die herkömmliche Software-Entwicklung ab.

Die wesentlichen Rahmenbedingungen

Der Erfolg hat Vorbedingungen In vielen Software- Entwicklungsprojekten wurde bisher der Erfolg kooperativer Systemgestaltung nach dem Prototyping-Verfahren nachgewiesen. Wenn ein solches beteiligungsorientiert ange-



Das zyklische Projektmodell unter Verwendung von Prototypen (hier: Systemversionen). Quelle: C. Floyd u.a., in: Informatik-Spektrum Heft 1, Feb. 1997.

legtes Projekt dennoch scheitert, dann fehlen in der Regel die entsprechenden Voraussetzungen. Dazu gehören vor allem:

- Qualifizierung der beteiligten Anwender (z.B. in Grundlagen der Software-Ergonomie, Teamarbeits- und Kommunikationstechniken)
- Qualifizierung der DV-Entwickler (z.B. Kenntnissse im Anwendungsfeld, Software-Ergonomie, kommunikative Kompetenz)
- Bereitstellung ausreichender Zeitkontingente für alle Beteiligten
- Betonung qualitätsbezogener Erfolgskriterien gegenüber Kriterien wie Kosten- und Termintreue
- Aufbau einer kooperationsförderlichen Projektorganisation für eine gleichberechtigte Zusammenarbeitvon System- und Anwendungsexperten.

Literatur

Ch. Stary, Th. Riesenecker-Caba:.EU-CON II - Software-ergonomische Bewertung und Gestaltung von Bildschirmarbeit Schriftenreihe der Bundesanstalt für Arbeitsschutz und Arbneitsmedizin. Forschung Fb 826 Dortmund, Berlin 1999

Wolfgang Schneider: Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten - Grundsätze der Dialoggestaltung. (Kommentar zu DIN EN ISO 9241-10) Hg.: DIN Deutsches Institut für Normung e.V. Beuth Verlag, Berlin Wien Zürich, 1998 (ISBN 3-410-13832-2)

Ivo Wessel:GUI-Design: Richtlinien zur Gestaltung ergonomischer Windows-Applikationen, Hanser-Verlag, 1998

CH. Rudlof, E. Becker-Töpfer: Software-Ergonomie und Arbeitsplatzgestaltung Düsseldorf 1997 Bezug: Gemeinnützige hbv-Kommunikations-, Bildungs- und Verlagsgesellschaft mbH, Kanzlerstr. 8, 40472 Düsseldorf

Claudia Döbele-Martin, Peter Martin, Richenhagen, Gottfried, Prümper, Jochen: Ergonomie-Prüfer Hg.: TBS Technologieberatungsstelle beim DGB Landesbezirk NRW, Oberhausen 1997

W. Hampe-Neteler: Software auf dem Prüfstand. Ausschlußkriterien zur ergonomischen Prüfung von Büro-Software Bund-Verlag, Reihe HBS-Praxis, Bd. 8 Köln 1994.

Die Vorteile des Prototypings

Software, die in einem zyklischen Projektmodell unter Verwendung von Prototyping und anderen geeigneten Methoden und Techniken der Anwenderbeteiligung entwickelt wurde, zeichnet sich in der Regel durch folgende Vorteile aus:

- höhere software-ergonomische Qualität des Endprodukts,
- bessere Gestaltung von Arbeitsorganisation und Arbeitsbedingungen (durch frühzeitiges Erkennensolcher Auswirkungen)
- geringerer Wartungs- und Weiterentwicklungsaufwand
- höhere Akzeptanz bei den Anwendern,
- geringere Schulungskosten.