Software-Ergonomie.

Seit die ersten mechanischen Webstühle menschliche Arbeit unterstützen oder ersetzen, werden immer mehr Maschinen in der Arbeitswelt eingesetzt. Bis vor einigen Jahrzehnten wurden nur mechanische Tätigkeiten von Maschinen durchgeführt, seitdem übernehmen Computer immer mehr auch das Planen, Steuern, Verwalten und Berechnen.
Bei dem Einsatz von Maschinen herrschte lange Zeit ein ***tayloristisches Konzept*** („bürokratisch-borniert“) vor, nachdem die Arbeit des Einzelnen immer mehr spezialisiert und auf wenige Tätigkeiten beschränkt wurde und Entscheidungen und qualifizierte Tätigkeiten nur von Wenigen getroffen wurden. Dies hat sich als Irrweg erwiesen, denn es gibt nicht den „einen besten Weg, etwas zu tun“, jeder hat seine Eigenarten und seinen Stil und bringt die besten Arbeitsergebnisse, wenn er dies entfalten kann. Die Fehlerrate, Qualität und Produktivität hatte sich durch durch Taylorismus oft verschlechtert, vor allem aber ist für den Arbeitenden unbefriedigend und ungesund, wenn seine Handlungsfreiheit zu sehr eingeschränkt ist. Es gibt Untersuchungen, nach denen Arbeiter, die nur manuelle Tätigkeiten mit geringem Handlungsspielraum verrichten, mehr an Herzkrankheiten leiden und früher sterben, als solche, die auch über ihre Arbeit mit entscheiden können.
So sollte auch durch den Einsatz von Computern die Verantwortung und Entscheidungsfreiheit des Anwenders nicht eingeschränkt werden. So forderte das Tavistock-Institut in seinem „***soziotechnischen Ansatz***“, daß eine Balance zwischen dem sozialen System und der Technik bestehen soll. Insbesondere plädiert das Institut für eine weitgehende Autonomie der Arbeitsgruppen, sie sollten sich weitestgehend selbst steuern, Vorgesetzte sollten nur das Zusammenspiel -die dynamische Interaktion- der Gruppen koordinieren.

Die Erfahrung zeigt, daß eine anfängliche Furcht vor ***technologischem Determinismus***, d.h. daß die Technik die Organisation beherrscht, unbegründet war. Nach wie vor wird entscheidet die Organisation über den Einsatz von Technik und über ihre Struktur. Allerdings kann dieser Einsatz die Strukturen durchaus verändern, so können durch neue Informationskanäle und damit verbundenen Entscheidungen Machtstrukturen verändert werden. Die Technik kann zu erhöhter Zentralisation, aber auch zum Gegenteil führen, je nachdem wie sie eingesetzt wird, also z.B. zentrale Datenbanken mit beschränktem oder allgemeinem Zugriff, Telearbeit, ...
Die Veränderungen geschehen allerdings meist langsam und evolutionär, so daß Mitarbeiter Gelegenheit haben sich der Technik anzupassen -oder die Technik sich, und Wissen und Qualifikationen nicht verloren gehen.
Probleme gibt allerdings oft wegen hohen Kontrollmöglichkeiten durch Computereinsatz und dem dadurch bedingten Datenschutz.
Zu beachten ist auch, daß der Einsatz von Computern Arbeit nicht nur rationeller macht, sondern den Aufwand auch steigen läßt, weil die Ansprüche steigen. So werden heute hohe Anforderungen an das Layout eines Schreiben, an Graphiken und Statistiken gestellt, die die ursprüngliche Arbeitsersparnis oft zunichte machen und sogar zu Mehrarbeit führen.

Ergonomie

Arbeit darf nicht gesundheitsschädlich oder gefährlich sein, dies schreibt das Arbeitsschutzgesetz vor. Dabei ist nicht nur an die körperliche Gesundheit gedacht, sondern auch an das geistige und seelische Wohlbefinden, das nicht gestört oder gefährdet, sondern gefördert werden sollte.
In unserer heutigen Arbeitswelt spielen Computer eine sehr große Rolle, sehr viele Arbeitnehmer sitzen den ganzen Tag davor. Die körperlichen Anforderungen bzw. Schädigungsmöglichkeiten werden von Sitzhaltung, Tastatur und Mausbedienung und Bildschirmeigenschaften bestimmt. Die Anforderungen an die Psyche werden dagegen im Wesentlichen von der eingesetzten Software geprägt, diese soll den Grundsätzen der Ergonomie entsprechen.

Unter **Ergonomie** versteht man hier, daß die **Eigenschaften des Dialogs, also der Bedienungsoberfläche in Gestaltung und Dynamik an die psychischen und physischen Eigenschaften des Menschen angepaßt sind.**

Das **Arbeitsschutzgesetz** schreibt in **§4** vor, daß der Arbeitgeber zum Arbeitsschutz verpflichtet ist und die zu treffenden Maßnahmen dem Stand der Wissenschaft und Technik zu entsprechen haben. Speziell mit der Software-Ergonomie beschäftigen sich die **DIN EN ISO-**Normen **4291** und **13407**, die auf der Bildschirm-Richtlinie der **EWG 270** von **1990** fußen. In dieser ersten Richtlinie war noch recht oberflächlich festgelegt, daß die Software an die damit zu erledigenden **Tätigkeiten angepaßt** sein muß, daß die Dialoge **vom Nutzer zu steuern** sein müssen und im Format und der Geschwindigkeit **an ihn angepaßt** sein müssen. Das Programm muß über seinen **Zustand** ständig oder auf Verlangen **Auskunft geben** und muß **Fehler erkennen, melden** und diese müssen ohne zu hohen Aufwand **korrigierbar** sein. Deutlich legt diese Richtlinie fest, daß durch das Programm **keine Kontrolle** des Arbeitnehmers stattfinden darf, **von der er nichts weiß**.

Ergonomie-Kriterien

Ausführlicher beschreibt die **ISO 4291** die ergonomischen Anforderungen. Dabei beschäftigen sich die **Teile 1-9** mit den **Hardware**- Anforderungen, also Sitzhaltung, Bildschirm-Eigenschaften, Tastatur, Maus, usw. Die Teile **10** bis **17** beschreiben die Forderungen für die **Software**:

Zunächst muß die Software an die **Aufgabe so angepaßt** sein, daß der Nutzer diese damit **effektiv** und **effizient** erledigen kann. Effektiv heißt, daß man mit der Software zum gewünschten vollständigen Arbeitsergebnis kommen muß und effizient, daß der Aufwand möglichst gering ist. Praktisch heißt dies, daß das Programm keine Angaben verlangen sollte, die es selber wissen oder sich erschließen könnte und daß das Format der Ein-und Ausgaben den Vorlagen entsprechen sollte, so daß keine weitere Umsetzung nötig ist und übliche Werte sollten vorgegeben sein. Natürlich sollen auch Änderungen möglich sei, ohne von vorne anfangen zu müssen.

Das System soll **selbstbeschreibungsfähig** sein, also verständlich sein, ohne ständiges Nachschlagen in einem Handbuch oder besondere Ausbildung. Dazu gehört eine verständliche Terminologie, die der der Arbeitswelt entspricht, verständliche Icons und Hilfefunktionen, auch kontextsensitiv.

Ähnlich verhält es sich mit der geforderten **Erwartungskonformität**. Dies meint, daß Kenntnisse aus dem Programm selbst oder anderen Programmen auf jede Oberfläche übertragbar sein sollte. Also durchgehend gleiche Begriffe für die gleiche Funktion, Verwendung gebräuchlicher Icon (z.B. Windows-Standard), Schaltflächen an der gewöhnten Stelle. Wichtig ist auch eine ***metaphorische Konsistenz***, damit ist die Übereinstimmung von Symbolen und Darstellungen mit der alltäglichen Umwelt.

Natürlich muß ein System **steuerbar** sein, der Mensch soll den Computer beherrschen und nicht umgekehrt. Damit muß der Mensch den Ablauf, Geschwindigkeit und die Richtung des Programms bestimmen können. Er darf vom Programm nicht unter Druck gesetzt werden, ihm darf nichts aufgezwungen werden, was nicht nötig ist. Er sollte auch Formate nach Bedarf einstellen können.

Ähnlich verhält es sich mit der Forderung nach der **Individualisierbarkeit**. Er sollte die Bildschirmmaske nach seinen Bedürfnissen und seinem Geschmack einstellen können und auch zwischen Tastatur und Mausbedienung wählen können. Wichtig ist dies vor allem bei körperlichen Problemen, z.B. soll bei Sehschwäche eine größere Schrift gewählt werden können. Er sollte auch Makros leicht selbst erstellen können, um immer wiederkehrende Bedienungsabläufe

automatisieren zu können.

Das System muß **fehlertolerant** sein, es soll Fehler erkennen, melden und ggf. selbst korrigieren. Diese Korrektur muß aber auch vom Bediener beeinflußbar sein, falls er doch die eingegebenen Werte möchte. Die Fehlerquelle muß auch deutlich benannt werden und entsprechende Hilfe angeboten werden, damit der Nutzer weiß, was er falsch gemacht hat und wie es zu berichtigen ist.

Schließlich soll das Programm **lernförderlich** sein, damit es einfach erlernt werden kann. Hierzu soll über das Systemkonzept informiert werden und ein Lernprogramm vorhanden sein. Das Programm soll so konzipiert sein, daß es auch für Anfänger gut zu bedienen ist, und diese ermuntert werden, immer mehr dazuzulernen und alle Möglichkeiten es Programms zu erkunden. Hierfür könnte ein Programm verschiedene Oberflächen für Anfänger, Fortgeschrittene und Spezialisten anbieten, bei denen das Menü von den nötigsten Funktionen bis zum vollen Angebot erweitert wird.

Evaluation

Wie soll nun bewertet werden, ob eine Software diese Forderungen zur Genüge erfüllt?
Es gibt eine Reihe verschiedener Methoden zur Evaluation der Ergonomie, ein Überblick ist in der **ISO**-Norm **16982** festgehalten.
Am einfachsten ist die **Befragung der Anwender**, am besten mit **standardisierten Fragebögen**. Es gibt hierfür fertig entwickelte Fragebögen, z.B. den Benutzerfragebogen zur ISO 9241, aber oft wird man nicht umhinkomme, für das Programm eigene Fragen zu entwickeln.
Statt einer schriftlichen Befragung ist auch eine **Beobachtung der Nutzer** bei der Arbeit möglich. Werden Beobachter eingesetzt, kann die Gleichmäßigkeit aller Beobachtungen aber kaum garantiert werden, eine nachträgliche Kontrolle ist auch nicht möglich, außerdem dürfte das Verhalten durch die Beobachtung beeinflußt werden. Daher empfiehlt sich der Einsatz von Kameras und Mikrophonen (zur Erfassung ärgerlicher Kommentare), aber dies ist auch nicht frei von Reaktanz, erfordert einen hohen Aufwand und ist auf jeden Fall zustimmungspflichtig.
Ebenfalls zustimmungspflichtig ist der Einsatz von **Loggingprogrammen**, mit denen das Bedien-Verhalten aufgezeichnet und nach Geschwindigkeit, Fehlern, umständlicher Bedienung usw. analysiert wird. Diese Analyse ist aber auch sehr aufwändig und kann nur einen Teil der Ergonomie-Anforderungen erfassen, eine schlechte Lesbarkeit der Maske kann z.B. nicht entlarvt werden.
Am besten haben sich in der Forschung **kontrollierte Experimente** bewährt. Hierbei werden mit repräsentativen Testpersonen typische Aufgaben in einer kontrollierten Umgebung (zur Ausschaltung oder Gleichhaltung von Umgebungseinflüssen) mit verschieden Programmen bearbeitet. Hier lassen sich Fehlerrate Ergebnis, Geschwindigkeit und Zufriedenheit relativ gut und zuverlässig messen.
Die Evaluation muß nicht durch die Nutzer erfolgen, sie kann auch **durch Experten** auf dem Gebiet der SW-Ergonomie beurteilt werden. Diese haben zwar mehr Übung Verstöße gegen die Ergonomie-Kriterien zu entdecken und deren psychologischen Auswirkung zu beurteilen (*Handlungsorientierte Fehleranalyse*), aber sie dürften sie schwerer tun, die Software nach ihrer Eignung für eine spezielle Aufgabe und die Richtigkeit der Fach-Begriffe einzuschätzen, als die Nutzer der SW.
Für eine objektive Evaluation ist es nötig, die Kritierien vor der Bewertung festzulegen. Dabei darf aber nicht der Fehler begangen werden, sich an den Merkmalen der vorhandenen Software zu orientieren, denn dies würde die Evaluation gegen vorhandene Unzulänglichkeiten *immunisieren*.

Eine **summative Evaluation**, d.h. die Beurteilung der fertigen Software ist aber immer der schlechtere Weg. Denn ist die SW bereits fertig, lassen sich gewünschte Änderung oft nur schwer verwirklichen und erfordern hohen Aufwand. Dieser Aufwand kostet Zeit und Geld und schon

daher wird sich der Auftraggeber -also der Arbeitgeber-, damit schwertun und lieber die unergonomische Software in Kauf nehmen und seinen Beschäftigten zumuten.

Softwareentwicklung

Anstatt die Software anhand der Auftrags-Anforderung selbstständig zu entwickeln und fertig den Benutzern zu übergeben -das sogenannte ***lineare Phasenmodell,*** auch *Wasserfallmodell* genannt, sollte die Software von Anfang an unter Einbezug der Nutzer mittels **Prototypen** entwickelt werden.
Dieser ***Prozeß der benutzerorientierten Gestaltung*** ist in der **DIN ISO 13407** beschrieben.
Hierbei muß zunächst einmal der genaue Benutzerkontext festgelegt und verstanden werden. Der Entwickler muß wissen, welche Kenntnisse und Fähigkeiten die Nutzer mitbringen, mit welchen Fachbegriffe sie arbeiten und wie der typische Arbeitsablauf ist. Auch die Umgebung, ob es sich um einen normalen Büroarbeitsplatz handelt oder um ein Terminal, ob der Nutzer in Ruhe arbeiten kann oder unter der Belastung steht Kunden bedienen oder einen Prozeß beobachten zu müssen, sind wichtige Kriterien. Wichtig sind natürlich auch Sicherheitsaspekte und rechtliche Regelungen und auch der soziale und kulturelle Kontext soll beachtet werden.

Nachdem diese Grundlagen für die SW bekannt sind, müssen Regeln für das Erstellen der Prototypen und deren Evaluation und der Kreis der Testpersonen festgelegt werden.
Dann kann mit der schrittweisen Entwicklung der SW begonnen werden. Nach jedem Schritt soll nun das Ergebnis in einem Prototyp dem späteren Nutzer vorgestellt und von diesen beurteilt werden. Diese Urteile sollen dann in die Änderung des Schrittes und die weitere Entwicklung einfließen. Der ideale Prototyp wäre natürlich ein lauffertiges Programm, bzw. eine bedienbare Oberfläche, die die Dynamik des echten Programms aufweist. Dafür dürfte allerdings der Aufwand zu hoch sein. Außerdem verwirft kein Programmierer gerne ein Programm in das er viel Arbeit gesteckt hat und dürfte dann Widerstand gegen geforderte Änderungen leisten.
Daher reicht es aus, das Ergebnis in einer Form zu präsentieren, die die Bildschirmgestaltung, den Dialog und den Programmablauf realistisch wiedergibt und erfahrbar macht, z.B. mit Folien, Power-Point-Repräsentationen oder simulierten Bildschirmmasken.
Wird das Programm auf diese Weise Schritt für Schritt mit den Benutzern entwickelt, wird das fertige Produkt den Regeln für ein ergonomisches Programm entsprechen.

Gegen diese -recht aufwändige- Form der SW-Entwicklung gibt es leider oft Widerstände, da dieser Prozeß länger dauert und teurer ist., als das übliche lineare Modell, vor allem wenn dieses keiner Evaluation unterzogen wird oder nicht danach umgeändert.
Es muß aber bedacht werden, daß der Einsatz unergonomischer Software den Arbeitgeber letztlich teuer kommen kann: Die Arbeit wird schlechter und langsamer erledigt, es kommt zu Fehlern, die manchmal fatale Folgen haben können. Die Mitarbeiter sind über schlechte Software verärgert und unmotiviert und in der Summe kann es durch den hierdurch ausgelösten Streß zu Krankheiten und Fehlzeiten kommen.
Auch für die SW-Entwickler rechnet sich eine sorgfältige Evaluation und Nachbesserung, denn hierdurch steigen die Marktchancen für wirklich gute Software.

Einführungsprozeß

Ist die Software fertig, muß sie eingeführt werden. Dieser Einführungsprozeß besteht oft leider nur aus einer Mitteilung an die Mitarbeiter, daß diese Software nun benutzt werden muß und ggf. noch einer kurzen Vorstellung.
Diese recht „kompakte“ Form der Einführung kann bei den Mitarbeitern eine Vielzahl von Ängsten

auslösen: Durch neue Software werden **Arbeitsroutinen verändert**, gewohnte Arbeitsprozesse müssen **neu eingeübt** werden, es muß **neu gelernt** werden und vorhandene **Kenntnisse** werden vielleicht sogar **entwertet**. Neben der Befürchtung **erhöhter Belastung** kann dies auch die **soziale Konstellation** unter den Kollegen **verändern**, so kann es passieren, daß „alte Hasen" plötzlich hilflos vor dem Computer sitzen, während junge Kollegen spielend mit diesem Medium umgehen. Dieser **Kompetenzverlust bzw. -gewinn** verändert die Anerkenntnis und die **soziale Reihung**. Schließlich ermöglicht der Einsatz von Computern erhöhte **Kontrolle** der Arbeitszeiten und -abläufe. Dadurch können gewohnte **Freiräume** verloren gehen und totale Kontrolle befürchtet werden. Und wenn die Software **Rationalisierungen** ermöglicht kommt die Angst vor Arbeitsplatzverlust dazu.

Solche Ängste können zu **Widerstand** (Reaktanz), **Passivität** oder auch **Überkonformität** führen. Es kommt zu (absichtlichen) Fehleingaben, die Arbeit wird sehr langsam und absichtlich umständlich ausgeführt (Dienst nach Vorschrift). An allem wird dem Computer die Schuld gegeben. Oder durch Angst vor Kritik wird die Arbeit mit der Software durchgeführt ohne sich über Fehler in der Software oder mangelnde Schulung zu beklagen.

Deshalb soll Software in einem gründlichen Prozeß eingeführt werden. Es beginnt damit, daß die neue Software **frühzeitig angekündigt** wird, bevor dies durch Gerüchte geschieht. Denn die Einführung muß **glaubwürdig** sein, d.h. **objektiv** und **kompetent** und sollte **persönlich** durch die Vorgesetzten erfolgen.
Die erste Vorstellung der neuen SW muß auf dem **Kenntnisstand** der Mitarbeiter aufsetzen und soll sich nicht in programmtechnischen Interna verlieren, sondern sich an den **Aufgaben orientieren**. Die möglichen Ängste und **Befürchtungen** müssen offen **angesprochen**, ernstgenommen und diskutiert werden.
Optimal ist es, wenn die Nutzer in an der Einführung der SW **partizipieren** können, also an Entscheidungen, ob, wann und welche Technik eingeführt wird, mit entscheiden können. Sie sollten auch an den Entscheidungen über die Schulung beteiligt werden.
Durch eine partizipative Einführung kann das **Interesse** an der Technik **geweckt** und **Gefühle** von **Bedrohung** oder **Überforderung vermindert** werden. Ein **aktives Herangehen** vermindert das Gefühle von Hilflosigkeit und Ausgesetztsein und kann eine eine **positive Herausforderung** wecken. So wird denn auch das so oft geforderte **commitment** gefördert.
Je mehr sich die Betroffenen -freiwillig- mit neuer Software beschäftigen, desto mehr Verständnis für **betriebliche Zusammenhänge** wird vorhanden sein und desto bessere und **kreativere Entscheidungen** werden getroffen. Schließlich wird das Engagement für die nötige Schulung und das Training geschaffen.
Der Arbeitgeber kann isch durch eine sorgfältige und partizipative Einführung von Software und Technik weniger Widerstand und ein besseres, schnelleres und sicheres Arbeiten mit dem neuen Medium erwarten. Je wohler sich der Mensch in seiner Arbeitsumgebung fühlt, desto produktiver wird seine Arbeit sein, desto weniger Krankheiten werden auftreten.

Wie die **Partizipation** genau aussieht kann man nicht sagen, denn diese muß auf den jeweiligen Fall hin entwickelt werden. Dabei muß der Einführungsprozeß **geplant** sein, aber die Planung soll **nicht** so detailliert und **starr** sein, daß es keinen Spielraum für Ergänzungen oder Verbesserungen mehr gibt. Dabei sollte man sich nicht zu früh entmutigen lassen, denn Partizipation ist ein demokratischer Vorgang und muß eingeübt werden. Natürlich können hierbei Unstimmigkeiten und Kritik auftreten, aber dies ist gewollt, denn verborgene Konflikte können nicht gelöst werden. Sollten sich die Mitarbeiter gegen die Einführung neuer Software sträuben, so kann dies auch an schlechten Erfahrungen mit Software und deren Einführung liegen, gerade hier sollte gezeigt werden, daß es auch anders geht.

Schulung

Alle Anwender einer neuen Software müssen geschult werden. Oft werden zur Kostenreduktion nur wenige geschult, die ihr Wissen dann an den Rest weitergeben sollen. Dies ist aber ein falscher Weg, denn zum einen braucht es zu einer guten Schulung auch gute rhetorische Kenntnisse, zum anderen kann kaum jemand selbst als Lehrer auftreten, wenn er den Stoff selbst erst gerade gelernt hat und darin unvollkommen und unsicher ist und vor allem die Praxiserfahrung fehlt. Außerdem fehlt oft die Zeit, zumal wenn die Schulung neben der normalen Arbeit erfolgen soll.
Wer an der Schulung spart, wird draufzahlen müssen, wenn die Arbeit mit den neuen Arbeitsmitteln nur schlecht ausgeführt werden kann.
Auch die Vorgesetzten sollen geschult werden, auch wenn sie mit der Anwendung nicht selbst umgehen. Aber sie müssen ja die Anwender beurteilen und müssen Verständnis für deren Probleme haben.
Beim Erlernen von Software muß verschiedenes Wissen geschult werden:
Zunächst geht es beim **Funktionswissen** um die grundlegende Funktionsweise der Software mitsamt der Hintergründe. Der Anwender soll kein SW-Spezialist werden, aber eine Vorstellung vom Gesamt-Zusammenhang bekommen. Es muß auch das **Problemwissen** geschult werden, nämlich die Fehlermöglichkeiten und Probleme der Software und dem Umgang damit. Damit hängt auch das **Unterstützungswissen** zusammen, das Wissen wie man sich -oder auch Kollegen- bei Problemen hilft, sei es durch Hilfsfunktionen, Handbücher, Kollegen oder Helplines.
Das **Fachwissen** bettet die Software in die Struktur der Arbeit und der Organisation ein. Hier können sich Arbeitsabläufe, Organisationsstrukturen oder die Kommunikation durch die neue Technik auch ändern.
Alles zusammen mündet schließlich in das **Umsetzungswissen**, mit dem Die Software in der Praxis angewandt werden kann.

Ein guter Trainer muß die Software **aufgabenorientiert lehren**, die Funktionen also anhand der typischen Arbeitsaufgaben und die Vorkenntnisse seiner Schüler erklären. Dazu muß er diese erst einmal kennen und sich einen Überblick durch vorherige Befragungen der Schüler oder deren Vorgesetzten verschaffen. Denkbar ist auch daß die Schüler ihre Aufgaben im Kurs vorstellen.
Wichtig ist, daß die Befehle nicht einzeln ohne Zusammenhang erklärt werden, sondern die **gesamte Struktur** und **innere Logik** des Programms erklärt wird. Der Schüler muß sich ein ***kognitives Modell*** von der Software machen können, dann kann er sich nicht nur die gelernten Befehle besser merken, sondern sich weiteres auch selbst erschließen. Hierzu ist es besser die SW anhand von Arbeitsaufgaben ***explorativ*** zu erkunden, anstatt nur theoretischen Unterricht zu erteilen.
Denn Menschen lernen lieber durch eigenes Probieren, anstatt durch Unterweisung. Handlungen werden durch „***operative Abbildsysteme***“ reguliert, solche ***mentale Modelle*** werden durch exploratives Lernen gebildet. Wichtig ist es auch, daß sich keine falschen Kenntnisse festsetzen, diese müssen rechtzeitig entdeckt und korrigiert werden, was durch praktische Erprobung am besten geschieht.
Daher sollten im Unterricht Fehler nicht sofort korrigiert, sondern zugelassen werden, damit Gelegenheit besteht, den **Umgang mit Fehlern zu erlernen**, womit auch die Angst vor möglichen Fehlern bei der Bedienung abnimmt.
Das Training sollte **teamorientiert** erfolgen, dabei wird die Kompetenz anderen zu helfen oder Hilfe zu suchen gleich mitgeschult. Als optimale Gruppengröße haben sich 6-8 Personen bewährt, mehr als 12 sollen es keinesfalls sein.

Nach der Schulung sind genug Möglichkeiten für **Übungen** wichtig. Im Arbeitsalltag sollten zeitliche Nischen eingerichtet werden, in denen die Kenntnisse ohne Zeitdruck anhand typischer

Aufgaben eingeübt und vertieft werden kann. Dabei soll durchaus ermuntert werden mit der Software zu „spielen“, um Möglichkeiten, Grenzen und Probleme zu erkunden.
Am effektivsten kann Software eingesetzt werden, wenn der Mitarbeiter genug **Handlungs-Spielraum** hat. Wenn möglich sollte dieser erweitert werden.

Neben der allgemeinen Schulung sollten besonders geeignete Kollegen weiter ausgebildet werden, um zu „**lokalen Experten**“ zu werden, die den Kollegen helfen und sie beraten können, bevor ein zentraler Support angerufen werden muß.

Fazit

In der heutigen Arbeitswelt ist die Entwicklung und Evaluation ergonomischer Software äußerst wichtig, um dem Nutzer die Möglichkeit zu geben seine Arbeitsmittel optimal einzusetzen und seine Arbeitskraft und sein kognitives Potential nicht mit dem Ärger an umständlichen, fehlerhaften und unübersichtlichen Programmen zu verschwenden und statt sich statt dessen auf seine eigentlichen Aufgaben zu konzentrieren und diese kreativ zu bewältigen. Dies ist auch im Interesse des Arbeitgebers, da die Produktivität steigt und Konflikte, Krankheiten und Fehlzeitem abnehmen. Es wird geschätzt, daß durch Software, die von Anfang an ergonomisch entwickelt wurde, die Kosten für die Implementierung um das 6-fache, für die Wartung sogar um das 60-fache sinken!
Das Arbeitsschutzgesetz verpflichtet die Arbeitgeber auch auf den Einsatz von ergonomischer Software, da unergonomische Programme in der Summe zu Ärger, Streß und Unzufriedenheit führen und letztlich ein Faktor für Krankheiten sind.
Leider sind in vielen Unternehmen noch immer Programme im Einsatz, die in vielen Punkten gegen die Ergonomie-Kriterien verstoßen, weil die Entwickler, -kommerzielle oder innerbetriebliche- keine Ergonomieschulung haben und keine Evaluation stattfindet.
Auf den Einsatz ergonomischer Software und deren Evaluation zu drängen ist auch ein wichtiges Betätigungsfeld für den Betriebsrat, der laut Betriebsverfassungsgesetz dazu berechtigt und sogar verpflichtet ist.

Links im Internet:

- **Das Arbeitsschutzgesetz**
  :http://www.gesetze-im-internet.de/bundesrecht/arbschg/gesamt.pdf

- **Die Bildschirmschutzverordnung**
  http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31990L0270:DE:HTML

Prüfhandbücher:

- http://www.datech.de/share/files/Pruefhandbuch_ISO_9241.pdf
- http://www.datech.de/share/files/Pruefhandbuch_ISO_13407.pdf

- **Ein Ergonomie-Leitfaden**
  http://www.ergonomie-leitfaden.de/

Allgemeines

- http://www.kommdesign.de/texte/din.htm
- http://asi-www.informatik.uni-hamburg.de/informatik/SE_Evaluation/html/pages/kriter.htm