Integrierte Organisations- und Technikgestaltung: Zur Rekontextualisierung von IT in Bildungseinrichtungen

Monique Janneck

Fachhochschule Lübeck, Fachbereich Elektrotechnik und Informatik

1. Einleitung

Auch in Schulen, Hochschulen und anderen Bildungseinrichtungen findet sich mittlerweile eine komplexe IT-Infrastruktur, vergleichbar durchaus mit Unternehmen. Neben Office-Software, Internet und E-Mail spielen naturgemäß Programme zur Unterstützung des Lehrens und Lernens (E-Learning, Blended Learning) eine wichtige Rolle. In den letzten Jahren erhalten zudem spezialisierte Informationssysteme wie etwa integrierte Campus-Management-Systeme eine zunehmende Verbreitung.

In diesem Beitrag werden typische Herausforderungen bei der Einführung von Informationstechnologie und Möglichkeiten zum Umgang damit aufgezeigt. Die Erfahrungen und Empfehlungen basieren auf einer Reihe umfangreicher Forschungs- und Evaluationsprojekte, in deren Rahmen Softwareentwicklungs- und -einführungsprojekte in Hochschulen, Schulen, Betrieben und Netzwerken begleitet wurden (vgl. Finck & Janneck 2008; Janneck & Finck 2006; Janneck 2009; Janneck, Adelberger, Fiammingo & Luka 2009; Janneck 2010).

Im folgenden Abschnitt werden Softwareeinführungsprozesse zunächst theoretisch als *Dekontextualisierung* und *Rekontextualisierung* konzeptionalisiert. Anschließend folgt eine Darstellung typischer Phänomene, die bei der Einführung komplexer Informationstechnologie auftreten können. Zur Veranschaulichung dient dabei das Modell der *Leavitt-Raute* (vgl. Leavitt 1965; Oberquelle 1991), das Techniknutzung als *Wechselwirkungen* zwischen Charakteristika der Technik, den Eigenschaften, Anforderungen und Bedürfnissen der sie nutzenden Menschen, ihrer Aufgaben sowie des organisationalen (bzw. sozialen / gesellschaftlichen) Kontextes begreift (*Abb. 1*). Abschließend werden einige „Lessons Learned“ und Empfehlungen für die Gestaltung von Technikeinführungsprozessen formuliert.

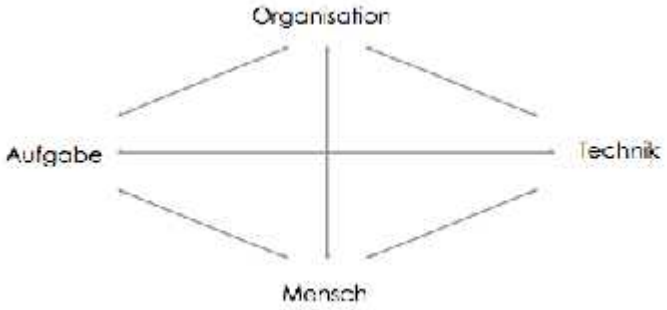

Abb. 1: Wechselwirkungen zwischen technischen und sozialen Faktoren (vgl. Leavitt 1965, Oberquelle 1991)

2. Dekontextualisierung und Rekontextualisierung

Softwareeinführungsprozesse in Organisationen lassen sich als zweifacher „Übersetzungsprozess“ verstehen (vgl. Krause, Rolf, Christ & Simon 2006; Langer, Simon, Gumm & Janneck 2008): Menschliche Handlungen, organisatorische bzw. soziale Strukturen und Prozesse werden aus ihrem Kontext herausgelöst – oder *dekontextualisiert* – um sie in formale Modelle und schließlich Algorithmen zu übersetzen. Schließlich erfolgt eine *Rekontextualisierung*, Rückübertragung der technischen Artefakte in den Ursprungskontext – der nach diesem Prozess meist nicht mehr ganz derselbe ist wie vorher: Technologie – und speziell Informationstechnologie – greift häufig stark in bestehende Arbeitsprozesse ein und führt zu tiefgreifenden und längerfristigen Veränderungen von Abläufen, Gewohnheiten, Anforderungen, Kompetenzen etc. Es liegt in der Natur der Sache, dass diese Veränderungen nicht für alle Beteiligten immer und ausschließlich positiv sind – und daher sind Technikeinführungsprozesse wie andere Organisationsentwicklungsprozesse auch häufig von Widerständen und Konflikten begleitet.

Während Software-Engineering-Vorgehensmodelle vielfältige Methoden für die Dekontextualisierungsphase zur Verfügung stellen (Modellierung, Formalisierung), steht die Rekontextualisierung weniger im Fokus (vgl. Gumm & Janneck 2007). Nichtsdestotrotz kann die Gestaltung der Einführungsphase entscheidend für den Projekterfolg sein, und auch ein qualitativ hochwertiges Produkt kann scheitern, wenn die Wiedereinbettung in den Nutzungskontext misslingt.

Konflikte im Zusammenhang mit der Rekontextualisierung von Software können einigen zentralen Faktoren zugeschrieben werden (vgl. Janneck et al. 2009, Janneck 2009, Janneck 2010):

Restriktionen durch Formalisierung. Formalisierung menschlicher Handlungen durch Softwareeinsatz ist häufig mit Standardisierung verbunden. Dies ist oft durchaus gewollt, um (Arbeits-) Prozesse zu vereinheitlichen, effizienter und transparenter zu machen. Standardisierung kann aber auch Flexibilität und Kreativität einschränken. Um die Grenzen der Formalisierung zu beschreiben, sprechen Krause et al. (2006) von *vorläufigen* und *notwendigen Formalisierungslücken*: Während vorläufige Lücken Prozesse beschreiben, die einer Automatisierung prinzipiell zugänglich sind, bezeichnen notwendige Formalisierungslücken Aktivitäten und Prozesse, die einer hohen Flexibilität bedürfen und nur sehr vorsichtig oder gar nicht automatisiert werden sollten.

Vergegenständlichung (verborgener Strukturen). Modellierung bedeutet Bestandsaufnahme: Existierende Strukturen, Prozesse und Beziehungen werden beleuchtet, auch solche, die bislang eher in einer Grauzone angesiedelt waren – wie informelle Absprachen, liebgewonnene Gewohnheiten, ungeschriebene Regeln etc. Konflikte, die daraus entstehen, werden zwar nicht von der Technologie ausgelöst, aber diese trägt dazu bei, sie aufzudecken (Finck & Janneck 2008).

Schaffung neuer Strukturen. Neue Technologie zeigt aber nicht nur Bestehendes auf, sie bringt auch fast unvermeidlich neue Strukturen und Routinen mit sich (Finck & Janneck 2008). Folglich müssen sich die Betroffenen anpassen und ihre Gewohnheiten und Handlungen verändern, oder sie erleben Veränderungen hinsichtlich ihres Status oder ihrer Position – mit positiven oder negativen Folgen für unterschiedliche Akteure.

Rekontextualisierung in einem neuen Kontext. Gerade bei Standardsoftware „von der Stange" finden Dekontextualisierung und Rekontextualisierung unter verschiedenen Rahmenbedingungen statt: Die Software wird für einen abstrakten oder idealisierten Nutzungszweck entwickelt, der von der tatsächlichen Nutzung stark abweichen kann. Entsprechend schwierig kann es für die Nutzer sein, zugrunde liegende Gestaltungsprinzipien zu verstehen und sie in Beziehung zu ihren eigenen Aufgaben und Interessen zu setzen.

Angst vor Veränderung. Jenseits der konkreten Änderungen, die sich durch die Einführung neuer Technologien ergeben, kann diese – wie andere organisationale Veränderungsprozesse auch – bei den Beteiligten Ängste auslösen, die – begründet oder nicht – die Technologieaneignung und damit verbundene Organisationsentwicklung behindern oder sogar boykottieren können.

3. Typische Phänomene in Softwareeinführungsprozessen

In den folgenden Abschnitten werden einige typische Phänomene geschildert, die in Softwareentwicklungs- und -einführungsprozessen in ganz unterschiedlichen Kontexten beobachtet wurden (vgl. Finck & Janneck 2008; Janneck & Finck 2006; Janneck 2009; Janneck et al. 2009; Janneck 2010).

3.1 Technikzentrierung

Mit Technikzentrierung ist gemeint, dass Arbeits- und Organisationsstrukturen an die Software angepasst werden, anstatt umgekehrt die Anforderungen und Erfordernisse der Nutzer in den Vordergrund zu stellen und die IT danach auszurichten. Dies kann durchaus gewollt sein – wenn etwa durch die Einführung einer Verwaltungssoftware Prozesse vereinheitlich werden sollen. Werden bestehende Softwaresysteme eingekauft, die gar nicht oder nur geringfügig angepasst werden können, sind Kompromisse meist unvermeidlich. Doch ob gewollt oder ungewollt: Als negative Begleiterscheinung drohen gut eingespielte Abläufe, „Best Practices" und Erfolgsmodelle verloren zu gehen. Zudem sind solche organisatorischen Anpassungen häufig mit hohem Aufwand für die Beteiligten verbunden und engen deren Handlungsspielraum ein – und gehen daher wenig überraschend oft mit Widerstand einher.

Technikzentrierung kann jedoch ebenso auftreten, wenn Software speziell für einen Anwendungskontext entwickelt wird. Sie äußert sich dann häufig als „Featuritis“ – der Implementierung von Funktionen um des technisch Machbaren willen (als Beispiel hierfür wurden in einer Studie zur Einführung von Campus-Management-Systemen verschiedene Kontrollfunktionen etwa zur Protokollierung der Anwesenheit von Studierenden in Seminaren genannt, die in der Praxis dazu führten, dass solche Kontrollen tatsächlich vermehrt durchgeführt wurden, wenngleich dies in den entsprechenden Studien- und Prüfungsordnungen keinesfalls so vorgesehen war). Solche „Technologie auf Vorrat“, die auf keiner konkreten Anforderung fußt, erzeugt die dazugehörige Nutzungssituation gewissermaßen selbst. Dabei sei angemerkt, dass dies natürlich nicht zwangsweise negative Folgen haben muss, sondern ebenso positiv wirken kann im Sinne von neuen, kreativen Impulsen. Nichtsdestotrotz ist die Entwicklung „technology driven“ (*Abb. 2*).

Abb. 2: Technikzentrierung

Technikdominanz kann sich auch in *mangelndem Einbezug* von Nutzern in den Entwicklungsprozess äußern: Softwareentwickler geben dann einseitig Ziele und Funktionalitäten vor, anstatt den Entwicklungsprozess in Wechselwirkung mit den (späteren) Nutzern zu gestalten.

Ein etwas anders gelagerter Fall von Technikzentrierung liegt vor, wenn der IT eine *Sündenbockfunktion* seitens der Organisation zugeschoben wird: Probleme und Schwierigkeiten, die in einem Veränderungsprozess auftreten, werden dann der IT zugeschrieben, auch wenn sie eigentlich individuell, organisatorisch oder sozial bedingt sind. Im Fallbeispiel einer Groupwareeinführung war beispielsweise zu beobachten, dass Teilnehmer nicht eingehaltene Absprachen oder Termine häufig mit schwer nachvollziehbaren technischen Schwierigkeiten begründeten (Finck & Janneck 2008). Auch hierbei handelt es sich um ein häufig zu beobachtendes Phänomen.

3.2 Kontextblindheit

Die Kehrseite einer starken Technikzentrierung ist meist die Vernachlässigung der Arbeits- und Organisationsgestaltung im Rahmen eines IT-Einführungsprozesses (*Abb. 3*). Wie oben beschrieben, geht Technikeinführung fast zwangsläufig mit – ggf. gravierenden – Veränderungen von Arbeitsprozessen einher. Grundlage eines erfolgreichen Veränderungsprozesses ist daher, diese Veränderungen von Anfang an mitzudenken und mitzugestalten, um unerwünschte Folgen zu minimieren: Ein Softwareeinführungsprozess sollte in diesem Sinne immer auch ein aktiver Organisationsentwicklungsprozess sein.

In der Realität ist häufig das Gegenteil der Fall: Dass mit der IT- auch eine Organisationsentwicklung verbunden ist, wird nicht oder erst zu spät gesehen. Dementsprechend stehen für klassische Change-Management-Maßnahmen wenig oder gar keine Ressourcen zur Verfügung.

Ein weiteres Symptom der Kontextblindheit ist auch hier eine geringe Partizipation der Betroffenen, die an der Gestaltung ihrer Arbeitssituation folglich ebenso wenig beteiligt werden wie an der Gestaltung der Software. Die Grundvoraussetzung und erste Stufe der Beteiligung ist dabei eine transparente Kommunikation und Information im und über den Veränderungsprozess. Erfolgt diese nicht, drohen wiederum Unmut und Widerstände der Betroffenen. In der Fallstudie eines Campus-Management-Einführungsprozesses beispielsweise war ein großer Teil der aufgetretenen Probleme in einer mangelhaften Kommunikation begründet. Dabei führten nicht nur versickernde oder fehlende Informationen zu Problemen, sondern insbesondere auch die Art und Weise, in der Informationen zum Prozess kommuniziert wurden und die von den Beteiligten als anmaßend und wenig wertschätzend empfunden wurde (Janneck et al. 2009; Janneck 2010).

Abb. 3: Kontextblindheit

3.3 Der Faktor Mensch

Auch in IT-Entwicklungs- und Einführungsprozessen ist der Mensch letztlich der entscheidende Faktor. Die beteiligten Personen und späteren Nutzer der IT bringen meist sehr unterschiedliche Voraussetzungen, Kenntnisse, Kompetenzen, Einstellungen und Vorlieben mit, die für eine entsprechende Dynamik und Konflikte sorgen können.

Zudem gehen die bereits mehrfach erwähnten Veränderungen, die im Zuge einer IT-Einführung zu erwarten sind, auch an den Beteiligten nicht spurlos vorbei. Veränderung von Arbeitsabläufen und Strukturen führt häufig auch zu veränderten *Rollen und Positionen*, die mit mehr oder weniger Prestige verbunden sein können (*Abb. 4*). Auch informelle (Rollen-) Zuschreibungen können sich verändern: So war in mehreren Fallstudien zu beobachten, dass technisch besonders versierte Personen im Zuge einer Softwareeinführung zu gefragten Experten wurden, während andere ihre vorherige zentrale Position verloren, da sie sich beispielsweise in stärker virtualisierten Kommunikationssituationen nicht mehr soviel Gehör verschaffen konnten (vgl. Finck & Janneck 2008). Mit anderen Worten: Macht- und Einflussstrukturen verschieben sich, und wiederum gibt es dabei naturgemäß Gewinner und Verlierer. Dass hierbei auch sehr emotionale Reaktionen auftreten können, verwundert nicht. Werden solche Aspekte nicht beachtet oder unter den Teppich gekehrt, birgt dies vielfältigen sozialen Sprengstoff.

Abb. 4: Der Faktor Mensch

3.4 Neue IT – und dann?

Abschließend soll ein Phänomen thematisiert werden, das mit „Lost in Organizational Hyperspace“ umschrieben werden könnte (*Abb. 5*). Dahinter verbirgt sich die Beobachtung, dass IT-Einführungsprozesse selten gut strukturiert und langfristig geplant sind (Gründe hierfür wurden weiter oben bereits angeführt; insbesondere die Vernachlässigung von Organisationsentwicklungsmaßnahmen ist hier zu nennen).

D.h., häufig wird das „Go live“, der Produktivstart der neuen Software, intensiv geplant und auch betreut (mit anfänglichen Schulungsmaßnahmen, Informationskampagnen usw.), mittel- und längerfristige Betreuungskonzepte sind jedoch unterentwickelt. Dabei können IT-Einführungsprozesse sehr langwierig sein, bis – gerade bei großen Systemen – die gesamte Organisation durchdrungen ist. Auch manche Schwierigkeiten stellen sich erst nach längerer Nutzungsdauer heraus (wenn beispielsweise etablierte und gut eingespielte Abläufe sich nicht mit der neuen IT „vertragen“), erfordern dann aber intensive Beachtung. In solchen späteren Projektphasen sind jedoch oft kaum noch Ressourcen verfügbar.

Eine Folge schlecht strukturierter und kurzfristig gedachter IT-Einführungsprozesse können Nutzungsboykotte oder nicht intendierte Nutzungsweisen („kreativer Missbrauch“) sein. Während Letztere häufig produktive Impulse setzen, kann ein (häufig stillschweigender) Boykott das Scheitern eines IT-Projektes bedingen. Hierfür gibt es zahlreiche Beispiele insbesondere aus dem Bereich des Wissensmanagements: Etliche Unternehmen investieren in teure Intranetsysteme, die ungenutzt verkümmern (vgl. Riege 2005).

In der bereits erwähnten Fallstudie eines Campus-Management-Einführungsprozesses wurde die einführende Hochschule mit einem „Supertanker“ verglichen (Janneck et al. 2009). Dieses Bild weist auf die Trägheit hin, mit der gerade große Organisationen auf Veränderungsprozesse reagieren. Entsprechend sollten IT-Einführungen langfristig geplant werden – weit über den Starttermin hinaus.

Abb. 5: Neue IT – und dann?

4. Empfehlungen und Lessons Learned

Aus den oben dargestellten Erfahrungen und Phänomenen lassen sich konkrete Empfehlungen für die Gestaltung von IT-Einführungsprozessen ableiten. Sie gliedern sich in vier Bereiche: *Partizipation*, *Formalisierungslücken*, *Change Management* und *Betreuungsmaßnahmen*.

4.1 Partizipation

Beteiligung ist ein wesentlicher Erfolgsfaktor in Veränderungsprozessen (vgl. Schreyögg 2008) – und dies gilt ebenso für IT-Einführungen. Die folgenden Aspekte gilt es zu beachten:

- *Pull- vs. Push-Strategien*: Unter Push-Strategien sind solche Maßnahmen zu verstehen, die darauf abzielen, die Beteiligten zu aktivieren und „ins Boot zu holen“, während Pull-Strategien Maßnahmen bezeichnen, die Personen in Eigeninitiative anwenden, um sich in den Prozess einzubringen (indem sie z.B. Verbesserungsvorschläge machen, sich aktiv an das Entwicklerteam wenden, ihre Mitarbeit anbieten etc.). In den untersuchten Fallstudien waren Pull-Maßnahmen häufig anzutreffen, und die betreffenden Personen waren damit meist sehr erfolgreich, d.h., es gelang ihnen, sich aktiv in den Einführungsprozess einzubringen. Allerdings handelte es sich dabei um eine stark selektive Gruppe von Personen, die typischerweise sehr interessiert an Technik und auch entsprechend versiert waren, während die „breite Masse“ nicht beteiligt wurde. Dementsprechend sollten Beteiligungsmaßnahmen explizit als Push-Strategien geplant werden.

- *Heterogene Nutzergruppen*: Naturgemäß können gerade in großen Organisationen nicht alle Betroffenen in gleicher Weise und gleich intensiv beteiligt werden. Daher ist es wichtig, eine möglichst repräsentative Auswahl von Nutzern zu treffen, die dann intensiv am Entwicklungs- und Einführungsprozess mitwirken. So kann vermieden werden, dass einseitige Gestaltungsentscheidungen getroffen werden, die bestimmte Nutzergruppen bevor- oder benachteiligen. Insbesondere Gruppen, die dem Prozess kritisch oder passiv gegenüberstehen, sollten Beachtung finden. Dafür ist oft auch notwendig, die *Selbstselektion* von Personen, die (wie oben beschrieben) aus Eigeninitiative ihre Mitarbeit anbieten, zu begrenzen. Der Aufwand, der hierfür nötig ist, sollte nicht unterschätzt werden: Oft ist es schwierig, Personen zur Mitarbeit – die nicht immer honoriert wird oder im Rahmen der regulären Arbeitszeit erfolgen kann – zu motivieren. Die anfänglichen Investitionen in den Prozess zahlen sich jedoch später aus.
- *Niedrigschwellige Angebote:* Um – insbesondere auch passivere, technisch weniger affinere etc. – Personen zur Mitarbeit zu motivieren, sollten die Hürden hierfür möglichst gering sein. Standard-Methoden zur Benutzerbeteiligung wie Workshops, ausführliche Arbeitsplatzbegleitungen und -beobachtungen, Arbeitsanalysen etc. sind typischerweise zeitaufwändig und schrecken möglicherweise ab. Niedrigschwelligere Maßnahmen sind z.B. (kurze) schriftliche Befragungen, die ggf. in regelmäßigen Abständen wiederholt werden, Feedbackmöglichkeiten über E-Mail, Online-Foren etc. Auch zu Interviews erklären sich viele Personen bereit. Aus solch ersten niedrigschwelligeren Kontakten kann dann die Bereitschaft entstehen, sich weitergehend – z.B. an Workshops – zu beteiligen, denn diese tiefgreifenderen Partizipationsmethoden sind selbstverständlich nach wie vor notwendig und sinnvoll.
- *Kommunikation und Information*: Eine Grundlage für Partizipation und deren niedrigschwelligste Maßnahme ist eine umfassende Information der Beteiligten. Information sollte dabei *Bring*- und nicht *Holschuld* sein; d.h. die Beteiligten sollten sich die Informationen über den Prozess nicht ausschließlich selber suchen müssen (über Webseiten, ausliegende Infoblätter etc.), sondern es muss sichergestellt werden, dass ihnen umfassend und rechtzeitig alle Informationen über den Prozess zugehen. Hilfreich dabei ist, *Kommunikationswege* und *-knoten* genau zu planen und zu strukturieren – auch im Zeitalter einfacher elektronischer Kommunikation erreichen Informationen andernfalls erstaunlich häufig nicht ihre Adressaten, wie die Fallstudien zeigen (Janneck et al. 2009, Janneck 2010).

4.2 Formalisierungslücken

Komplexität schafft Probleme: Fast zwangsläufig nehmen mit der Komplexität eines Softwaresystems Probleme mit der Gebrauchstauglichkeit zu. Systeme einfach und intuitiv bedienbar zu gestalten ist ab einem gewissen Funktionsumfang kaum mehr ohne Einschränkungen möglich. Zudem leidet häufig die Konsistenz der Gestaltung, wenn sehr viele und unterschiedliche Anforderungen und Funktionalitäten in einem Softwaresystem vereinigt werden sollen. Weniger ist häufig mehr – in diesem Sinne kann man auch bei Softwareentwicklungsprojekten den „Mut zur Formalisierungslücke“ propagieren. Das bedeutet, die Grenzen dessen, was ein IT-System leisten soll, genau zu ziehen und überflüssige Funktionalitäten und rein technologiegetriebene Entwicklungen zu vermeiden. Im Folgenden sind einige Aspekte hierzu aufgeführt:

- *Formalisierungslücke: Ausnahmen und Sonderfälle*. In den untersuchten Fallstudien wurden insbesondere bei großen Softwaresystemen, die von heterogenen Personengruppen genutzt wurden, häufig Ausnahmeregeln und Sonderwünsche hinsichtlich bestimmter Funktionalitäten geäußert, die meist nur für einen kleinen bzw. sehr speziellen Anwenderkreis relevant waren. Neben dem erheblichen Implementierungsaufwand, den solche Sonderwünsche mit sich bringen, beeinträchtigen diese in besonderem Maße eine konsistente, transparente Systemgestaltung. Sonderwünsche sind daher kritisch zu hinterfragen: Warum reicht die Standardfunktionalität nicht aus? Stecken möglicherweise organisatorische Probleme oder Konflikte dahinter, wenn bspw. einzelne Gruppierungen, Abteilungen, Fachbereiche etc. sich nicht an einem einheitlichen Verfahren beteiligen wollen oder können? Wurde gar das falsche System gewählt? Auch wenn Ausnahmen nachvollziehbar und gut begründet sind, ist nicht notwendigerweise für jeden Einzelfall eine spezielle Implementierung notwendig. Häufig ist die bessere Lösung, die Standardfunktionalität so flexibel zu gestalten, dass verschiedene Vorgehensweisen unterstützt werden. In jedem Fall hat sich als hilfreich erwiesen, die verschiedenen Nutzergruppen an einen Tisch zu bringen und gemeinsam einen Entwurf zu gestalten, der allen Anforderungen möglichst gut gerecht wird (vgl. Obendorf, Janneck & Finck 2009).
- *Formalisierungslücke: Rollen- und Rechtesysteme*. Sehr häufig weisen gerade E-Learning-Systeme, Groupware, Campus-Management-Systeme etc. überkomplexe und starre Rollen- und Rechtesysteme auf, mit denen (lesende und schreibende) Zugriffsrechte z.T. bis auf kleinste Details modelliert werden. In manchen Fällen ist dies aufgrund datenschutzrechtlicher Bestimmungen unumgänglich. In den Fallstudien zeigte sich allerdings, dass trotz dieser Komplexität die Systemrollen häufig nicht mit der Realität übereinstimmten, was zu vielfältigen Arbeitsverzögerungen (aufgrund mangelnden Zugriffs auf Informationen) oder gerade zu Sicherheitsproblemen führte, weil beispielsweise Kennungen und Passwörter weitergereicht wurden. Auch bei der Gestaltung von Rechtesystemen ist daher vielfach eine einfachere, flexible Lösung vorzuziehen – und solche Systeme bewähren sich auch in der Praxis.
- *Intelligente Schnittstellen statt EWOMS (= Eierlegende Wollmilchsäue)*. Sind Anforderungen begründet und nachvollziehbar zu unterschiedlich, um sie unter einen Hut zu bringen (s.o.), ist darüber nachzudenken, ob diese tatsächlich in einem System abgebildet werden sollten und müssen. Möglicherweise existieren sogar bereits Alt- bzw. Alternativsysteme, die einen bestimmten Bereich abdecken und gut funktionieren. Dann ist

empfehlenswert, anstatt einer aufwändigen Neu- oder Nachimplementierung solche Systeme über intelligente Schnittstellen, die u.a. eine konsistente Datenhaltung sichern, anzubinden.

- *Evolutionäre Entwicklung*. Die Forderung nach einem schrittweisen, evolutionären Entwicklungsprozess, bei dem Zwischenergebnisse und Prototypen frühzeitig mit den Anwendern diskutiert und erprobt werden, ist mittlerweile „Stand der Kunst" in der Softwaretechnik. Ein solches Vorgehen ermöglicht gerade bei großen Entwicklungsprojekten, mit einer Kernfunktionalität zu beginnen, die schrittweise erweitert wird. So können Erfahrungen in den Entwicklungsprozess einfließen und es kann vermieden werden, dass dieselben Fehler wiederholt gemacht werden. Außerdem wird so die Gefahr begrenzt, dass „Technologie auf Vorrat" entwickelt wird, ohne dass konkrete Anforderungen hierfür existieren, wenn klar ist, dass in späteren Ausbaustufen noch die Möglichkeit zur Nachrüstung besteht. Dies gilt auch für Ausnahmen und Sonderregeln, die sich möglicherweise doch als unnötig erweisen, nachdem die Standardfunktionalität gründlich erprobt werden konnte.

4.3 Change Management

IT-Gestaltung bedeutet gleichzeitig immer auch Arbeitsgestaltung und Organisationsgestaltung. Entsprechend sollte eine IT-Einführung auch als *Change-Management-Prozess* geplant und gestaltet werden (vgl. z.B. Doppler & Lauterburg 2002; Lauer 2010). Das bedeutet zuallererst, ein klares *Konzept* zu entwickeln, welche Ziele in organisatorischer Hinsicht mit der IT-Einführung verbunden sind, welche Veränderungen in Arbeitsabläufen, Struktur, Kompetenzen... angestrebt werden und wie diese mittel- und langfristig umgesetzt werden sollen. Es bedeutet aber auch, dass für diesen Organisationsentwicklungsprozess *Ressourcen* eingeplant und bereitgestellt werden müssen, und zwar sowohl in Hinblick auf Personal – interne sowie ggf. externe Personen, die den OE-Prozess planen und begleiten bzw. für ihre Teilnahme an Maßnahmen von anderen Aufgaben freigestellt werden – als auch im Hinblick auf Zeit: Organisationsentwicklungsprozesse sollten längerfristig geplant und begleitet werden.

Es würde den Rahmen dieses Beitrags deutlich sprengen, Change-Management-Prozesse ausführlich zu thematisieren. Abschließend sei daher nur noch ein konkreter Hinweis gegeben: In den meisten Organisationen wird bei der Einführung (größerer) IT-Systeme ganz selbstverständlich ein technischer Support angeboten. Einen *organisatorischen Support* als Pendant hierzu, der die Beteiligten bei der Re-Organisation ihrer Arbeitsabläufe unterstützt, findet man daher sehr selten – dieser wäre jedoch mindestens ebenso wichtig.

4.4 Betreuung

Wie im vorangegangenen Abschnitt schon angeklungen: Betreuung und Support sind wichtige Erfolgskriterien für IT-Einführungsprozesse, und zwar nicht nur in technischer, sondern auch und gerade in *organisatorischer* und ggf. – bei E-Learning- und Blended-Learning-Systemen – auch *didaktischer* Hinsicht. Dies zeigen die Fallstudien deutlich (vgl. Janneck et al. 2009; Janneck 2010; Strauss, Pape, Adam, Klein & Reinecke 2003).

Darüber hinaus bieten Supportangebote eine unkomplizierte und niedrigschwellige Partizipationsmöglichkeit (s.o.): Sie ermöglichen einen „direkten Draht“ zu den Nutzern, wodurch Kommunikationsanlässe geschaffen werden und wertvolles Feedback zu Nutzungserfahrungen, -problemen und Anregungen für die Weiterentwicklung eingeholt werden kann. Daher sollte der Support mit gut geschulten Mitarbeitern besetzt werden und engen Kontakt zum Softwareentwicklungsteam unterhalten.

Auch für den Support sollten v.a. *Push*- statt *Pull*-Maßnahmen (vgl. Abschnitt 4.1) zum Einsatz kommen, d.h., es sollten niedrigschwellige Möglichkeiten zur Kontaktaufnahme angeboten werden, und die Supportmitarbeiter sollten selber regelmäßig den Kontakt zu den Nutzern suchen (z.B. über Online-Fragebögen oder kleine Feedbackformulare, die ggf. direkt in die Software integriert werden können).

Zusammenfassend lässt sich sagen: Eine komplexe IT-Infrastruktur ist auch aus Bildungseinrichtungen kaum mehr wegzudenken – und kann deren Alltag deutlich bereichern und erleichtern. Werden Softwareeinführungsprozesse als längerfristige Organisationsentwicklungsprojekte angenommen und entsprechend geplant, lassen sich die geschilderten Schwierigkeiten minimieren und Chancen und Potentiale nutzen.

5. Literatur

Doppler, K. & Lauterburg, C. (2002). *Change Management. Den Unternehmenswandel gestalten*, 10. Aufl. Frankfurt, New York: Campus.

Finck, M., Janneck, M. (2008). Das Unvorhersehbare steuern? Zum Umgang mit der komplexen Dynamik in Technologieaneignungsprozessen. In: Gumm, D., Janneck, M., Langer, R., Simon, E. (Hrsg.), *Mensch, Technik, Ärger? Zur Beherrschbarkeit soziotechnischer Dynamik aus transdisziplinärer Sicht*. Hamburg: Lit-Verlag, S. 85-102.

Gumm, D., Janneck, M. (2007). Requirements Engineering for Software Recontextualization. In: Tiainen, T., Isomäki, H., Korpela, M., Mursu, A., Paakki, M.-K., Pekkola, S. (Eds.), *Proceedings of 30th Information Systems Research Seminar in Scandinavia*, S. 226-243.

Janneck, M. (2009). Recontextualizing Technology in Appropriation Processes. In: Whitworth, B., de Moore, A. (eds): *Handbook of Research on Socio-Technical Design and Social Networking Systems*. Hershey, PA: IGI Global, pp. 153-166.

Janneck, M. (2010). Challenges of Software Recontextualization: Lessons Learned. In: *Proceedings of CHI 2010*, April 2010, Atlanta, GA, USA.

Janneck, M., Adelberger, C., Fiammingo, S., Luka, R., (2009). Von Eisbergen und Supertankern: Topologie eines Campus-Management-Einführungsprozesses. In: Hansen, H. R., Karagiannis, D., Fill, H.-G. (Hrsg.): *Business Services: Konzepte, Technologien, Anwendungen. 9. Internationale Tagung Wirtschaftsinformatik*. Wien: OCG, S. 453-462.

Janneck, M., Finck, M. (2006). Making The Community a Hospitable Place - Identity, Strong Bounds, and Self-Organisation in Web-Based Communities. *International Journal of Web Based Communities*, 2 (4), pp. 458-473.

Krause, D., Rolf, A., Christ, M. & Simon, E. (2006). Wissen, wie alles zusammenhängt. In: *Informatik-Spektrum, 29* (4), S. 263-273.

Langer, R., Simon, E., Gumm, D., Janneck, M. (2008). Soziotechnische Systeme und ihre transdisziplinäre Erforschung – eine Skizze. In: Gumm, D., Janneck, M., Langer, R., Simon, E. (Hrsg.), *Mensch, Technik, Ärger? Zur Beherrschbarkeit soziotechnischer Dynamik aus transdisziplinärer Sicht*. Hamburg: Lit-Verlag, S. 177-205.

Lauer, T. (2010). *Change Management. Grundlagen und Erfolgsfaktoren*. Berlin: Springer.

Leavitt, H.J. (1965). Applied organizational change in industry: structural, technological and humanistic approaches. In March, J. G. (ed), *Handbook of organizations*. Chicago: Rand McNally.

Obendorf, H., Janneck, M., Finck, M. (2009). Inter-Contextual Distributed Participatory Design: Communicating design philosophy and enriching the user experience. *Scandinavian Journal of Information Systems*, 21 (1), pp. 51-76.

Oberquelle, H. (1991) (Hrsg.). *Kooperative Arbeit und Computerunterstützung. Stand und Perspektiven*. Göttingen.

Riege, A. (2005). Three-dozen knowledge-sharing barriers managers must consider. *Journal of knowledge management, 9(3)*, 18-35.

Schreyögg, G. (2008). *Organisation. Grundlagen moderner Organisationsgestaltung*. Wiesbaden: Gabler

Strauss, M., Pape, B., Adam, F., Klein, M., Reinecke, L. (2003). *CommSy-Evaluationsbericht 2003: Softwareunterstützung für selbstständiges und kooperatives Lernen*. Berichte des Fachbereichs Informatik der Universität Hamburg.