In: Liskowsky, R.; Velichkovsky, B.M.; Wünschmann, W. (Hrsg.): Proceedings der Fachtagung Software-Ergonomie 1997, S. 211-220

Kokonstruktive Weiterentwicklung eines Groupwareproduktes

Das Beispiel der Reimplementierung eines Suchtools

Helge Kahler und Volker Wulf

Projektbereich Software-Ergonomie und CSCW

Institut für Informatik III, Universität Bonn

Zusammenfassung

Die Entwicklung von Groupware ist durch ein produktorientiertes, von einzelnen Anwendungsfeldern abstrahierendes Vorgehen geprägt. Die dabei entstehende Software erfüllt aber häufig nicht die Anforderungen, die Nutzer in konkreten Anwendungsfeldern an sie richten. Dies soll hier gezeigt werden am Beispiel eines Suchtools, das in einem marktgängigen Groupwareprodukt zur Unterstützung räumlich verteilter telekooperativer Arbeit realisiert war. Die sich aus der unzulänglichen Realisierung dieses Suchtools ergebende kokonstruktive Reimplementierung basierend auf der Programmierschnittstelle des Groupwareproduktes wird beschrieben. Abschließend werden Konsequenzen diskutiert, die sich aus dieser Fallstudie für eine auftragsorientierte Weiterentwicklung von Groupwareprodukten ergeben.

1. Einleitung

Der Anwendungskontext von Groupware ist unter den gegebenen makroökonomischen Rahmenbedingungen durch Differenzierung und Dynamik gekennzeichnet. Organisationen und ihre Untereinheiten differenzieren sich aus, um so besser auf sich rasch wandelnde Markterfordernisse reagieren zu können. Aus diesen Merkmalen des Anwendungskontextes ergeben sich zwangsläufig Konsequenzen für den Software-Entwicklungsprozeß bei Groupware.

Aus der Differenziertheit verschiedener Anwendungsfelder zwischen und innerhalb von Organisationen ergibt sich die Notwendigkeit, Nutzer in den Software-Entwicklungsprozeß aktiv einzubeziehen. Nur sie kennen ihren Anwendungskontext hinreichend genau, um entsprechende Anforderungen an einen ihre kooperative Arbeit unterstützenden Artefakt zu formulieren (vgl. [8]). Ein solches Vorgehen soll im folgenden als kokonstruktives Vorgehen (zwischen Entwicklern und Nutzern) bezeichnet werden. Im Gegensatz zur skandinavischen Tradition der partizipativen Systementwicklung, bei der die emanzipatorische Wirkung von Nutzerbeteiligung im Vordergrund stand (vgl. [1]), wollen wir hier unter dem Begriff kokonstruktiver Entwicklung auch Ansätze subsummieren, die lediglich auf die funktionale Wirkung von Benutzerbeteiligung abstellen (vgl. [2]). Im Gegensatz zum Mainstream der US-amerikanischen Diskussion zum Thema "Usability-Engineering" (z.B. [13]) beschränken wir uns beim kokonstruktiven Vorgehen nicht ausschließlich auf die Gestaltung der Ein-/Ausgabe- und Dialogaspekte der Benutzungsschnittstelle gemäß dem IFIP-Modell. Vielmehr wird auch die Gestaltung der Funktionalität und die organisatorische Einbettung der Anwendungen thematisiert (vgl. [4]).

Aus der Dynamik des Anwendungskontextes läßt sich die Notwendigkeit zu einem evolutionären Software-Entwicklungsprozeß ableiten. Da sich die Arbeitsbedingungen während der Nutzung verändern, muß die Möglichkeit bestehen, die Software – wenn nötig – entsprechend den veränderten Anforderungen weiterzuentwickeln.

Um zu untersuchen, inwiefern Möglichkeiten zu einer kokonstruktiven und evolutionären Entwicklung von Groupware bestehen, soll auf Grudins Unterscheidung in Produkt- und Auftragsentwicklung zurückgegriffen werden (vgl. [9, S. 60ff]). In der Produktentwicklung werden aufeinanderfolgende Versionen einer Software für eine Vielzahl von Anwendungsfelder entwickelt. Die anwendende Organisation entscheidet sich in der Regel für ein "fertiges" Produkt. Die Funktionalität eines solchen Produktes wird vom Management und der Marketingabteilung des Herstellers vorgegeben und von den Entwicklern ausgearbeitet. Obwohl es je nach Produkt verschiedene Kommunikationsforen zwischen Hersteller- und Anwenderorganisation gibt (z. B. Anwenderarbeitskreise), findet in der Regel keine direkte Beteiligung von Nutzern im Entwicklungsprozeß statt. Demgegenüber wird bei der Auftragsentwicklung eine Software für genau ein Anwendungsfeld erstellt. Da die künftigen Nutzer bei Beginn der Entwicklung bekannt sind, ergibt sich die Funktionalität der Anwendung aus den Anforderungen des Anwendungsfeldes. Auch wenn dabei in vielen Fällen die Chance zur Etablierung eines kokonstruktiven Vorgehens nicht genutzt wird, so sind die Bedingungen hierfür in der Auftragsentwicklung viel günstiger als in der Produktentwicklung. Ein evolutionäres Moment ist in der Produktentwicklung durch die fortlaufende Erstellung neuer Versionen gegeben, es stellt sich für die Nutzer allerdings die Frage, inwiefern die neuen Versionen die Anforderungen aus den jeweiligen Anwendungsfeldern erfüllen. Bei der Auftragsentwicklung erfordert die Weiterentwicklung häufig hohen Aufwand, so daß dort ein evolutionäres Vorgehen nicht die Regel ist.

Trotz der hier diskutierten Probleme, in der Produktentwicklung auf die Differenziertheit und Dynamik des Anwendungskontextes eingehen zu können, ist Groupwareentwicklung bisher im wesentlichen Produktentwicklung (vgl. [10, S. 94ff]). Vor dem Hintergrund der sich aus der Organisationsumwelt ergebenden Anforderungen stellt sich allerdings die Frage, ob dies in Zukunft so bleiben kann. Da die Produktentwicklung den Anwendern den offensichtlichen Vorteil bietet, mit geringem Aufwand schnell über einsatzbereite Funktionalität zu verfügen, ist zu überlegen, wie ein Produktentwicklungsprozeß durch auftragsorientierte Elemente anzureichern ist. Möglichkeiten hierzu sind bei Produkten sowohl gegeben, wenn deren Funktionalität durch Eingriffe in den Programmcode verändert werden kann, als auch dann, wenn deren Funktionalität durch Einbindung externen Programmcodes mittels einer Programmierschnittstelle modifiziert werden kann. Die Groupware LinkWorks der Firma DEC ist ein Produkt, das beide Formen der Modifikation des Programmcodes ermöglicht.

Bisher finden sich in der einschlägigen Literatur wenig Studien, die kokonstruktives Vorgehen bei der Entwicklung von Groupware thematisieren (z.B. [3]; [5]). Berichte über die kokonstruktive Weiterentwicklung von bestehenden Groupwareprodukten sind bisher kaum veröffentlicht. Dies hat seinen primären Grund darin, daß in Forschungsprojekten Software in der Regel neuentwickelt wird. Nichtsdestotrotz ist dies für den erfolgreichen Einsatz von Groupware in betrieblichen Anwendungskontexten eine wichtige Fragestellung.

In diesem Aufsatz soll über die kokonstruktive Reimplementierung des Suchtools in LinkWorks berichtet werden. Dazu wollen wir im ersten Schritt zeigen, welche Schwächen das ursprünglich in LinkWorks implementierte Suchtool aufwies. Wir werden dann unsere Vorgehensweise beschreiben und darstellen, in welcher Weise sich das kokonstruktiv entwickelte Suchtool von dem ursprünglichen Design unterscheidet. Es wird diskutiert, inwiefern diese Reimplementierung durch die von LinkWorks vorgegebenen Modifikationsmöglichkeiten beeinflußt wurde. Abschließend werden die Vor- und Nachteile des hier gewählten Vorgehens diskutiert und untersucht, auf welche Weise die Gestaltung und die Entwicklungsmethodik des zu Grunde liegenden Groupwareproduktes verändert werden muß, um die Verwendung in einem spezifischen Anwendungskontext zu erleichtern.

2. Probleme des Originalsuchtools in POLITeam

Der Projektbereich Softwareergonomie und CSCW des Instituts für Informatik III der Universität Bonn ist Konsortialpartner des von Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie geförderten POLITeam-Projektes. Dieses Projekt beschäftigt sich mit der Unterstützung von Telekooperation in großen räumlich verteilten Organisationen, wobei u.a. die Metapher von "gemeinsamen Schreibtischen" verwandt wird, auf die mehrere mit der POLITeam-Groupware arbeitende Benutzer Zugriff haben können. Das im Rahmen des POLI-Team-Projektes genutzte Groupwareprodukt LinkWorks der Firma DEC enthielt ein Suchtool, das es erlaubte, prinzipiell jedes dem System bekannte Objekt auf allen "elektronischen Schreib-tischen" zu suchen.

Beim Originalsuchtool konnte der Suchende eine Vielzahl von Suchkriterien im Eingabefenster angeben, darunter die Objektklasse (z. B. ”Text” oder ”Ordner”), das Datum der letzten Änderung und den Namen des Erstellers. Die entsprechend der Kriterien spezifizierten Objekte wurden dann im gesamten System gesucht und im Suchergebnisfenster des Aktivators, also derjenigen Person, die die Suche initiiert hat, angezeigt. Wählte der Suchende bestimmte Objekte zur Übernahme auf seinen Schreibtisch aus, so wurden für diese Dokumente Verweise erzeugt, so daß der Suchende auf sie zugreifen konnte. Um zu vermeiden, daß dem Aktivator private oder geheime Dokumente anderer Nutzer auf diese Weise zugänglich würden, hatten die Entwickler vorgesehen, daß für jedes Objekt mittels eines Attributes festgelegt werden konnte, ob es durch das Suchtool auffindbar sein oder ihm verborgen bleiben sollte. Dieses Attribut konnte nicht von den Nutzern direkt manipuliert werden, sondern war Bestandteil von komplexeren Zugriffsprofilen. In den Anwendungsfeldern von POLITeam wurden nach Absprache mit den Nutzern drei verschiedene Zugriffsprofile definiert. Wollte ein Nutzer sicher sein, daß sein Objekt anderen bei ihrer Suche verborgen blieb, mußte er diesem das Zugriffsprofil ”privat” geben, wohingegen die Zugriffsprofile ”zur Information” und ”öffentlich” ein Objekt mit dem Suchtool auffindbar machten. Da LinkWorks erlaubte, Dokumente mit dem Zugriffsprofil "privat" zu verschicken, dies aber dazu führte, daß die Dokumente beim Empfänger weder bearbeitet, gelesen oder auch nur zurückversandt werden konnten, wurde den Nutzern im Rahmen der Schulung zur Vermeidung von Fehlern empfohlen, lediglich für ausschließlich zum privaten Gebrauch bestimmte Dokumente das Zugriffsprofil "privat" zu vergeben. Insofern hätte der Gebrauch des Suchtools nahezu alle Dokumente dem Suchenden zugänglich gemacht.

Neben diesen Problemen der Gestaltung der Funktionalität des Suchtools bestanden auch Probleme hinsichtlich der Dialogaspekte der Benutzungsoberfläche. Durch die Entscheidung, eine Vielzahl von Suchkriterien mittels nur eines Eingabefenster abzufragen, war dieses Eingabefenster des Originalsuchtools überladen und unübersichtlich.

Diese Probleme führten zur Entscheidung, von dem in POLITeam-Projekt üblichen Vorgehen abzuweichen, Groupwareprodukte zunächst einzuführen und die dabei sich zeigenden Probleme für ein späteres Redesign zu nutzen (vgl. [15]). Statt dessen wurde das Suchtool in den Anwendungsfeldern deaktiviert und unmittelbar ein Reimplementierung des Suchtools angestoßen.

3. Vorgehen bei der kokonstruktiven Entwicklung

Gemäß dem für POLITeam gewählten kokonstruktiven und evolutionären Vorgehen erfolgte die Reimplementierung in enger Zusammenarbeit mit den POLITeam-Nutzern (vgl. [6]; [8]). Abb. 1 stellt das dabei verfolgte Vorgehen vor. Wie bereits dargestellt, begann der Entwicklungsprozeß mit der Evaluation des bestehenden Suchtools (vgl. Kap. 2). Außerdem nahmen wir die Ergebnisse der laufenden Diskussion im Bereich menschengerechter Gestaltung von Groupware zur Kenntnis. Von besonderer Bedeutung war die in Voruntersuchungen gewonnene Erkenntnis, daß der Gebrauch einzelner Groupwarefunktionen zu Konflikten zwischen einzelnen Nutzern führen kann (vgl. [14]; [17]).

Basierend hierauf wurde ein Interviewleitfaden bestehend aus insgesamt 29 offenen Fragen entwickelt, mit dem potentielle Nutzer von Groupware zu ihrer bisherigen Suchpraxis befragt wurden. Dabei wurde im ersten Teil des Fragebogens allgemein erhoben, welches grundlegende Suchverhalten bei den Befragten bei papierbasierter oder elektronischer Suche vorherrscht. Hier wurde nach der Art der gesuchten Objekte, dem Suchanlaß und dem Suchvorgang (u.a. exemplarische Beschreibung des Vorgehens, Dauer, verwandte Hilfsmittel wie Post-It-Zettel etc.) gefragt.

Um das Konfliktpotential, das Funktionen des Suchtools anhaftet, aufzudecken, wurden im zweiten Teil des Fragebogens die Rollen einer suchenden Person und derjenigen Person unterschieden, in deren Arbeitsbereich (z. B. auf oder in deren Schreibtisch oder auf deren Festplatte) jemand anderes etwas sucht. Hier wurde u. a. erfragt, in welchen Bereichen von Kollegen gesucht werden darf, ob der Suchende Einblick in Vorgänge erhält, nach denen er nicht gesucht hat, ob eine Suche die Befragten in der Rolle des Betroffenen stört und welche Bedeutung bei der Suche die Position der Beteiligten in der Organisation hat.

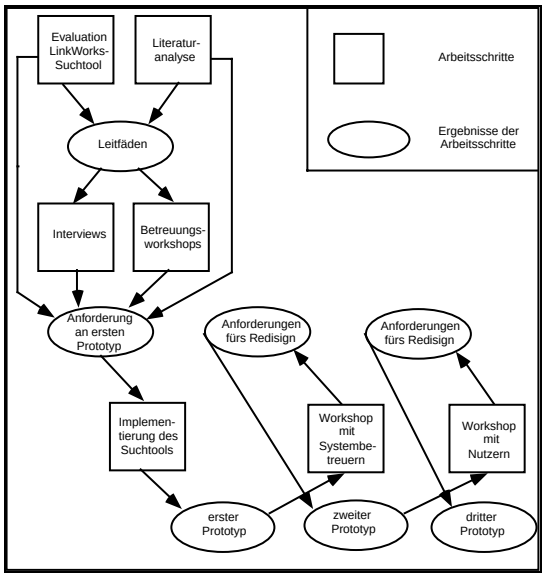

Abb. 1: Vorgehen bei der Reimplementierung des Suchtools

Um nicht ausschließlich die Besonderheiten eines Anwendungsfeldes zu erfragen, wurden Interviews in vier verschiedenen Organisationen aus dem Büro- und Verwaltungsbereichs durchgeführt. In diesen Anwendungsfeldern wurden insgesamt 10 etwa 30- bis 45-minütige Interviews geführt und ausgewertet. Außerdem wurden zwei zur Betreuung von POLITeam-Nutzern durchgeführte Workshops im Hinblick auf sich daraus an das Design des Suchtools ergebende Anforderungen hin beobachtet und ausgewertet. Es zeigte sich dabei, daß insbesondere die Auswertung der Interviews eine reiche Quelle der Inspiration war. Als Ergebnis der Voruntersuchungen ergaben sich die im folgenden Kapitel dargestellten Anforderungen an einen ersten Prototypen.

Nach dessen Implementierung führten wir zwei weitere Workshops mit dem Ziel durch, das Design der Prototypen weiterzuentwickeln. Dazu wurde ein lauffähiger Prototyp an Hand eines typischen Anwendungsszenarions den Workshop-Teilnehmern vorgestellt. Im ersten Workshop setzte sich die Teilnehmerschaft aus Anwendungsentwicklern und Nutzerbetreuern des POLITeam-Projektes zusammen. In diesem Workshop wurden insbesondere Verbesserungen der Handhabung und der Bedienung des Suchtools eingefordert. Am zweiten Workshop nahmen neben den Entwicklern des Suchtools drei Mitarbeiter aus einem der in der ersten Interviewphase bereits involvierten Anwendungsfelder teil. Die wichtigsten der in den Workshops für die Weiterentwicklung des Suchtools geäußerten Ideen sind in Kap. 6 dargestellt.

4. Anforderungen an den ersten Prototypen

Aus den Problemen mit dem Originalsuchtool, den Ergebnissen der Interviews und den Erfahrungen aus den Betreuungsworkshops ergaben sich die folgenden Anforderungen an das Design des ersten Prototypen.

Differenzierung verschiedener Suchbereiche – Durch die rollenorientierte Befragung in den Interviews wurde das Konfliktpotential der Suche außerhalb des eigenen Arbeitsbereichs deutlich. Befragte in der Rolle des Suchenden sprachen sich für eher uneingeschränkten Zugriff auf den gesamten Suchraum aus, während sie in der Rolle des von der Suche Betroffenen eher die Befürchtung äußerten, daß private elektronische Bereiche in nicht nachvollziehbarer Weise durchsucht werden. Im vortechnischen Zustand ließ sich in den meisten Anwendungsfeldern ein differenzierter Umgang mit verschiedenen Suchbereichen feststellen ("Die Büros werden grundsätzlich nicht abgeschlossen, die Kollegen haben in der Regel freien Zugang. Eigentlich kann überall gesucht werden, Schubladen werden allerdings als privat angesehen. Da sucht normalerweise keiner drin"). Andere Befragte unterschieden zwischen ”erlaubter und unerlaubter Suche

zum Zweck einer Überprüfung" und erwähnten bei einer Suche in den Büros Anderer existierende "Tabuzonen, die von der Stellung des Kollegen, aber auch seinem Charakter und seinem persönlichen Typ abhängen" (vgl. [12, S. 63ff]). Aus diesen Gründen sahen wir für den ersten Prototypen die Möglichkeit vor, drei Suchbereiche zu differenzieren: den eigenen Schreibtisch, den einer anderen Person und die Registratur bzw. das Archiv. Dem Aktivator sollte nach der Suche und vor der Übernahme der gefundenen Objekte auf den eigenen Schreibtisch anzeigen werden, wo ein gefundenes Objekt lokalisiert wurde. Die Interessen der Betroffenen sollten dann in einem weiteren Schritt durch unterschiedliche Konfliktregelungsmechanismen – wie z.B. der Unterdrückung der Anzeige gefundener Objekte oder der Anzeige des Suchvorgangs – gewahrt werden (vgl. [17]).

Beschränkung auf wichtige Suchkriterien – Die Interviews ergaben, daß der Eingabedialog des Originalsuchtools einige Suchkriterien enthielt, die für die Arbeit in den Anwendungsfeldern als nicht bedeutsam angesehen wurden. Als Suchkriterien waren unseren Interviewpartnern der Name des Objekts, das Datum der Erstellung und/oder letzten Veränderung, der Autor, der Eigentümer, ein Stichwort/Betreff, das Aktenzeichen und die Objektklasse des Objekts wichtig. Darüber hinaus wurde Möglichkeiten zur Volltextsuche gewünscht, weil Textdokumente laut Aussagen von Befragten ("Ich suche eigentlich nur nach Dokumenten") die mit Abstand meistgesuchten Objekte darstellten (vgl. [12, S. 52ff]).

Einbindung eines Kommunikationskanals – In den Interviews berichteten die Nutzer, daß sie während des Suchens häufig Kollegen befragen, um basierend auf deren Auskünften den Suchraum einschränken zu können. Aussagen wie "Suche ich ein Dokument, daß ich an einen Mitarbeiter ausgeliehen habe, so versuche ich ihn persönlich oder aber per Telefon zu erreichen. Ist der Mitarbeiter nicht anwesend, dann lege ich ihm einen Zettel in seinen Postfach oder, wenn es dringend ist, direkt auf seinen Stuhl" oder Fragen an Kollegen wie "Kannst Du Dich noch an die Bestellung vor ein oder zwei Jahren erinnern, wo zuviel Skonto abgezogen wurde?" können als typisch angesehen werden (vgl. [12, S. 59ff]). Da sich Groupwarenutzer nicht immer zur selben Zeit am selben Ort aufhalten, sollte das Suchtool Möglichkeiten zur Kommunikation während der Suche bereitstellen, also beispielsweise direkten Zugriff auf die Groupware-eigene Mailfunktionalität anbieten. Dabei erschien es sinnvoll, das Eingabefenster zur Spezifikation der Suchkriterien mitverschicken zu können.

5. Realisierungsalternativen

Um ein Suchtool zu implementieren, das diesen Anforderungen gerecht werden sollte, bietet LinkWorks prinzipiell zwei Vorgehensweisen an. Zum einen besteht die Möglichkeit, das Originalsuchtool durch Reprogrammierung der LinkWorks-eigenen Objektklassen zu modifizieren, zum anderen gibt es die Alternative, ein in einer gängigen Programmiersprache entwickeltes Suchtool in LinkWorks einzubinden. Dafür stellt LinkWorks eine spezielle Programmierschnittstelle zur Verfügung (APO für *Application Plus Objects*), die einen Zugang zu Objekten innerhalb von LinkWorks und die Veränderung und Ableitung von Methoden erlaubt. Zur APO gehört unter anderem eine Abfragesprache, die die Suche nach Objekten erlaubt. Für die Modifikation des Originalsuchtools sprachen neben dem geringeren Entwicklungsaufwand auch Performancegründe, da auf Grund der eingeschränkten Möglichkeiten der APO-Abfragesprache abzusehen war, daß mit einem extern angebundenen Suchtool die Geschwindigkeit des Ori-

ginalsuchtools nicht zu erreichen war. Gegen die Modifikation des Originalsuchtools sprach die Tatsache, daß über die Programmierung von Objektklassen und die Änderung der Menüstruktur von LinkWorks lediglich Eingriffe bei der Behandlung der Suchergebnisse möglich gewesen wären. Dagegen ließen sich der Eingabedialog und der Suchvorgang selbst nicht ändern. Dies hätte insbesondere zur Folge gehabt, daß die Eingabemaske nicht hätte modifiziert werden können und daß eine Unterteilung der gefundenen Objekte nach den Orten, an denen sie gefunden wurden, *vor* der Übernahme in den Suchergebnisordner nicht möglich gewesen wären. Somit hätten wichtige der erhobenen Anforderungen nicht realisiert werden können.

Unter Berücksichtigung dieser Einschränkungen bei der Modifikation des Originalsuchtools entschieden wir uns für die Realisierung einer externen Lösung, die über die LinkWorks Programmierschnittstelle an die LinkWorkseigene Datenbank angebunden wurde. Da bei den LinkWorks Anwendungspartnern fast ausschließlich Microsoft Windows Clients im Einsatz sind, entschieden wir uns zudem, auf eine aufwendige plattformübergreifende Lösung zu verzichten und realisierten das prototypische Suchtool in Microsoft Visual Basic for Windows. Bei der Realisierung ergab sich gegenüber dem Originalsuchtool eine Performanceeinbuße um den Faktor drei bis fünf je nach Suchanwendung für das noch nicht optimierte externe Suchtool. Diese Performanceschwäche läßt sich auf die Unzulänglichkeit der APO zurückführen, die keinen direkten Zugriff auf *alle* im Orginalsuchtool verwandte Funktionalität zuläßt. So stehen beispielsweise Systemattribute wie "Betreff" und "Schlüsselwörter" zwar dem Orginalsuchtool als Suchkriterien zur Verfügung, werden aber von der APO-Abfragesprache nicht unterstützt, was eine laufzeitkritische Nachprogrammierung der Funktionalität in Visual Basic erfordert.

Neben dem eigentlichen Suchvorgang erfolgt bei der Verwendung des externen Suchtools innerhalb von LinkWorks noch die Bereichsüberprüfung, die für jedes gefundene Objekt feststellt, ob es auf dem Schreibtisch der suchenden Person, einer anderen Person oder im Archiv / in der Registratur zu finden ist. Diese Bereichsüberprüfung ist zur Zeit noch zeitintensiv, da die Information kein Attribut des gefundenen Objekts ist.

6. Neues Suchtool

Das nach Abwägung der Realisierungsalternativen erstellte neue Suchtool erlaubt die Umsetzung der oben genannten Designanforderungen. Nach dem Start der Suche öffnet sich der Eingabedialog, der gemäß den Maßgaben der Voruntersuchung neu gestaltet ist (s. Abb. 2).

Abb. 2: Eingabedialog des Suchtools

Im Eingabedialog lassen sich die Suchkriterien für gesuchte Objekte spezifizieren. Die Suche verläuft um so schneller, je stärker die Suchkriterien, die bei einer Suchanfrage mit einem logischen UND verknüpft werden, den Suchraum einschränken. Grundlegende objektbezogene Suchkriterien sind die Objektklasse, die sich mit Hilfe einer Listbox auswählen läßt, und der Objektname, bei dessen Angabe auch Platzhalter verwandt werden können. Im Bereich *Inhalt* des Eingabedialogs läßt sich ein objekteigenes Betreff sowie ein Suchtext spezifizieren, der dann bei einer Volltextsuche gefunden werden kann. Im Bereich *Datum* kann als Erstellungs- und Änderungsdatum sowohl ein Zeitpunkt (Tag) als auch ein Zeitraum eingegeben werden. Der Bereich *Anwender* schließlich ermöglicht eine Einschränkung der Suche insofern, als daß bei Kenntnis der Person, die das Objekt erstellt oder geändert hat oder sein Eigentümer ist, sich diese noch angeben läßt. Im zweiten Workshop mit Nutzern von POLITeam wurde vorgeschlagen, die Eingabe für den Bereich *Anwender* gegenüber dem hier vorgestellten zweiten Prototypen noch dahingehend zu ändern, daß hier die Angabe einer Person auch ohne ihr Verhältnis zum Objekt reicht, da oft nicht klar sei, ob diese Person Ersteller, Änderer oder Eigentümer des Objekts ist.

Oberhalb der Eingabefelder für die Suchkriterien im Eingabedialog befinden sich noch zwei Buttons und ein Anzeigeelement. Einer der Buttons aktiviert den Eingabedialog für Kurznachrichten von POLITeam und leistet so einen Beitrag zu Kommunikation mit Kollegen bezüglich der Suche. Für eine geplante Einbindung von Audio- und Videotechnik in POLITeam ist hier auch die Öffnung eines entsprechenden Kommunikationskanals zu einer Person denkbar, die zur Suche angesprochen werden soll. Der andere Button ermöglicht den Abbruch des Suchvorgangs. Das Anzeigeelement gibt Auskunft über den Fortschritt bei der Suche.

Nach einer erfolgreichen Suche und Überprüfung, auf welchem Schreibtisch ein gefundenes Objekt liegt, werden die gefundenen Objekte in den drei Gruppen ”eigener Schreibtisch”, Schreibtisch von Kollegen”, ”Archiv” angezeigt (s. Abb. 3). Von dort aus können sie ausgewählt und in ein Ergebnisfenster auf dem elektronischen Schreibtisch übernommen werden. Diese Unterteilung in die drei Bereiche dient der ersten Orientierung des Suchenden darüber, wann die

Übernahme welcher Objekte eventuell die Interessen Anderer berührt (vgl. Kap. 4). Hinsichtlich des Umgangs mit Konflikten bezüglich des Suchens äußerten die Nutzer im zweiten Workshop, daß sie als von einer Suche Betroffene für einzelne Dokumente individuell festlegen wollten, ob diese von anderen Nutzern gefunden werden können. Außerdem wünschten sie sich eine Anzeige dann, wenn Andere auf ihrem elektronischen Schreibtisch suchen. Eine weitere von den Nutzern geäußerte Idee betraf die Art, in der bei Anderen gefundene Objekte dem Suchenden zugänglich gemacht werden. Im Orginalsuchtool ebenso wie im ersten Prototypen führte eine erfolgreiche Suche dazu, daß das beim Suchenden ein Verweis auf das entsprechenden Objekt erzeugt wird, so daß der Suchende darauf Zugriff hat, ohne daß andere Personen davon wissen. Um das Risiko unerwünschter Manipulationen am Original auszuschließen, verlangten die Nutzer, daß im Normalfall dem Suchenden lediglich eine Kopie des Objektes bereitzustellen ist und nur dann, wenn dieser explizit zugestimmt hat, ein Verweis zu erstellen ist.

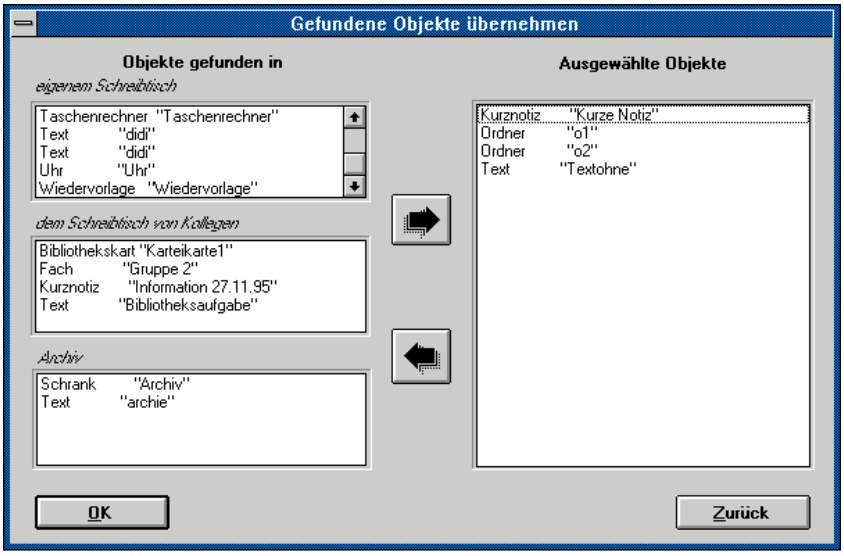

Abb. 3: Übernahmefenster des Suchtools

Die ausgewählten Objekte werden in einen Suchergebnisordner übernommen, von dem aus mit ihnen wie mit jedem anderen LinkWorks Objekt gearbeitet werden kann.

7. Zusammenfassung und Ausblick

In der hier beschriebenen Fallstudie wurden Schwächen eines produktorientierten Entwicklungsansatzes für Groupware deutlich. Die Gestaltung des ursprünglichen Suchtools deutet darauf hin, daß die Entwickler die bei Groupware im Vergleich zu einzelplatz-orientierten Systemen zu berücksichtigenden Probleme nur unzureichend wahrgenommen hatten. Für die hier involvierten Anwendungsfelder ließ sich durch interview- und prototypbasierte Kokonstruktion ein die Nutzeranforderungen befriedigenderes Design erzielen. Der im weiteren geplante Einsatz der ersten Version des neuentwickelten Suchtools muß allerdings noch erweisen, inwiefern damit alle augenblicklichen Gebrauchsanforderungen des im Workshop repräsentierten Anwendungsfeldes abgedeckt sind. Dabei ist insbesondere auf Einflüsse zu achten, die sich aus der Dynamik dieses Anwendungskontextes ergeben. Darüber hinaus ist zu untersuchen, inwiefern auch weitere – über

die bisher einbezogenen – Anwendungsfelder die existierenden Gestaltungsanforderungen teilen oder ob und inwieweit diese zu modifizieren sind. Insofern stellt die präsentierte Version des Prototypen lediglich ein "Zwischenprodukt" eines kokonstruktiven und evolutionären Softwareentwicklungsprozesses dar.

Bei der Reflexion des Vorgehens zeigte sich, daß die Auswertung der interview-orientierten Exploration von nicht technisch unterstützen Suchvorgängen wichtige Designideen für das zu entwickelnde Suchtool erbrachte. Die aus der Voruntersuchung abgeleitete rollenorientierte Form der Befragung deckte das einzelnen Funktionen des Suchtools immanente groupware-spezifische Konfliktpotential auf. Techniken, die dazu beitragen, daß Nutzer im Rahmen einer kokonstruktiven Vorgehensweise sich in verschiedene technisch induzierte Rollen versetzen können, sollten weiterentwickelt werden. Solche Techniken stellen eine groupware-spezifische Erweiterung bisher bekannter Ansätze kokonstruktiver Systementwicklung dar.

Die hier dokumentierte Fallstudie macht deutlich, daß sich die kokonstruktive Weiterentwicklung eines Groupwareproduktes nicht auf die Ein-/Ausgabe- und Dialogschnittstelle beschränken darf. Vielmehr erfordern gerade die Verschiedenheit der potentiellen Anwenderorganisationen und die ihnen inhärente Entwicklungsdynamik, daß auch Möglichkeiten für eine Anpassung der Funktionalität bereitgestellt werden. Hier müssen klassische software-ergonomische Erkenntnisse daraufhin untersucht werden, inwieweit sie Hilfestellung für den Prozeß der Anpassung einer generischen Groupware für eine spezielle Organisation bieten können. Dabei spielen sowohl technische Voraussetzungen der Groupware eine Rolle als auch das methodische Vorgehen bei der Anpassung sowie die Fragen, wer auf welche Weise die Anpassungen vornehmen kann.

Hinsichtlich der Integration des neuentwickelten Suchtools in das Produkt LinkWorks zeigten sich deutliche Grenzen der Möglichkeit, das bestehende Produkt weiterzuentwickeln (vgl. auch [11]). Ähnliche Schwierigkeiten zeigen sich auch bei der Reimplementierung eines den Nutzeranforderungen entsprechenden Ereignisdienstes in LinkWorks (vgl. [7]). Diese Probleme scheinen typisch für augenblicklich auf dem Markt erhältliche Groupwareprodukte zu sein. LinkWorks bietet durch die Möglichkeit der Modifikationen bestimmter Methoden und einer leistungsfähigen Programmierschnittstelle vergleichsweise eher günstige Voraussetzungen.

Es stellt sich abschließend die Frage, welche Hinweise sich aus der hier dokumentierten Fallstudie für den Softwareentwicklungsprozeß bei Groupware gewinnen lassen. Die Fallstudie zeigt, daß ein ausschließlich produktorientierter Entwicklungsprozeß zu unbefriedigenden Ergebnissen führt. Andererseits spricht die Höhe des Aufwandes und die Dauer der Implementierung gegen eine in jedem Anwendungsfeld erfolgende Auftragsentwicklung. Insofern scheint eine modifizierte Kombination der beiden Ansätze ein aussichtsreiches Konzept. Ein "Rohprodukt" könnte produktorientiert allerdings unter erheblich stärkerer Betonung kokonstruktiver Vorgehensweise entwickelt werden. Die Entwickler sollten die zu beteiligenden Anwendungsfelder so auswählen, daß eine möglichst weitgehende Heterogenität der Nutzeranforderungen berücksichtigt wird. Sich dabei widersprechende Anforderungen sollten entweder alternativ umgesetzt werden, so daß bei der Einführung des Produktes im Anwendungskontext eine einfache diesbezügliche Konfiguration vorgenommen werden kann. Ansonsten sollte an solchen Stellen Möglichkeiten zur Reimplementierung in einer der Produktentwicklung zeitlich nachfolgenden Auftragsentwicklung ermöglicht werden. Solche Reimplementierungen werden

bisher durch die vorherrschende "Produktsicht" der Hersteller nicht hinreichend unterstützt. Im Falle des Suchtools bot die Möglichkeiten zur Klassenprogrammierung und die APO-Abfragesprache zu wenig Möglichkeiten zu einer suchzeiteffizienten Implementierung (vgl. Kap. 5). Ein weiteres Problem stellt der Umgang mit zeitlicher Dynamik der Nutzeranforderungen dar. Zwar sieht produktorientiertes Vorgehen eine iterative Produktversionen vor. Augenblicklich bestehen in der Regel nur schwache Informations- und keinerlei Mitbestimmungsmöglichkeiten der Anwender und Nutzer bei der Definition der zukünftigen Gestaltungsziele des Herstellers. Um eine nutzerorientierte Auftragsentwicklung auf Basis eines solchen Produktes über einen längeren Zeitraum betreiben zu können, müßten deshalb die Ziele der Versionsentwicklung Gegenstand intensiver Kommunikation zwischen Hersteller, Auftragsentwicklern und Nutzern werden.

Insofern stimmen wir mit Sumner und Stolze ([16]) überein, die als Ergebnis einer empirischen Untersuchung der Nutzung einzelplatzorientierter Softwareprodukte dynamisch sich entwickelnde Anforderungen feststellten und zu deren Umsetzung eine kokonstruktive Entwicklung basierend auf den Möglichkeiten von in den Produkten implementierten Programmierschnittstellen vorschlugen. Zur Unterstützung von Auftragsentwicklungen basierend auf Groupwareprodukten glauben wir jedoch, daß bei der Festlegung der Eigenschaften einer Programmierschnittstelle bereits erhebliches Wissen über mögliche Anwendungsfelder vorliegen sollte, um die hier dargestellten Probleme vermeiden zu können. Auf diese Weise könnte aus einer Kombination von produkt- und auftragsorientiertem Vorgehen eine Perspektive für ein nutzergerechtes Software-Engineering bei Groupware erwachsen.

Literaturverzeichnis

[1] Bjerknes, G.; Ehn, P.; Kyng, M.: Computers and Democracy: A Skandinavian Challenge, Avebury 1987

[2] Clement, A.: Computing at Work: Empowering Action by "Low-level Users", in: Communications of the ACM, Vol. 37, No. 1, 1994, S. 53 - 63

[3] Cool, C.; Fish, R.S.; Kraut, R.E.; Lowery, C.M.: Interactive Design of Video Communication Systems, in: CSCW '92. Sharing Perspectives. Proceedings of the Conference on Computer-Supported Cooperative Work, ACM Press, New York 1992, S. 25 - 32

[4] Dzida, W.: Das IFIP-Modell der Benutzerschnittstelle, in: Office Management, Sonderheft Software-Ergonomie, Vol. 31, 1983, S. 6 - 8

[5] Egger, E.; Wagner, I.: Time Management: A Case for CSCW, in: CSCW '92. Sharing Perspectives. Proceedings of the Conference on Computer-Supported Cooperative Work, New York 1992, S. 249 - 256

[6] Floyd, Ch; Reisin, F.-M. and Schmidt, G. 1989: *STEPS* to software development with users, in: Ghezzi, C.; McDermid, J.A. (eds.): ESEC'89 - 2nd European Software Engineering Conference, University of Warwick, Coventry. Lecture Notes in Computer Science No. 387, Heidelberg: Springer, S. 48 - 64

[7] Fuchs, Ludwin; Sohlenkamp, Markus; Genau, Andreas; Kahler, Helge; Pfeifer, Andreas; Wulf, Volker: Transparenz in kooperativen Prozessen: Der Ereignisdienst in POLITeam. In: Krcmar, Helmut; Lewe, Henrik, Schwabe, Gerhard (Hrsg.): Herausforderung Telekooperation (Proceedings of the DCSCW 1996), Springer, Berlin u.a. 1996, S. 3-16

[8] Grønbæk, K.; Kyng, M.; Mogensen, P.: Cooperative Experimental System Development – Cooperative Techniques Beyond Initial Design and Analysis, in: Proceeding of the Third

Decennial Conference: Computers in Context: Joining Forces in Design, Aarhus, DK, August 14 - 18, 1995, S. 20 - 29
[9] Grudin, J.: Interactive Systems: Bridging the Gaps between Developers and Users, in: IEEE Computers, No. 4, 4/1991, S. 59 - 69
[10] Grudin, J.: Groupware and Social Dynamics: Eight Challenges for Developers, in: Communications of the ACM, Vol. 37, No. 1, 1/1994, S. 92 - 105
[11] Kahler, Helge: Developing Groupware with Evolution and Participation - A Case Study. In Proceedings of the Participatory Design Conference 1996, Cambridge, MA, November 1996, S. 173-182
[12] Krüdenscheidt, G.: Partizipative Entwicklung eines Suchtools für Groupware, Diplomarbeit am Institut für Informatik der Universität Bonn, Bonn 1996
[13] Nielsen, J.: Usability Engineering, Academic Press, Boston et al. 1993
[14] Rohde, M.; Pfeifer, A.; Wulf, V.: Konfliktmanagement bei Vorgangsbearbeitungssystemen; in: WIRTSCHAFTSINFORMATIK, 38. Jg., Nr.2, S. 199 - 209
[15] Sohlenkamp, M.; Mambrey, P.; Fuchs, L.; Prinz, W.; Syri, A.; Kolvenbach, S.; Klöckner, K.; Pankoke-Babatz, U.: Unterstützung verteilter Regierungsarbeit mit POLITeam, in: Augsburger, W.; Ludwig, H.; Schwab, K. (Hrsg.): Tagungsband des Workshops "Koordinationsmethoden und Werkzeuge bei der computergestützten kooperativen Arbeit" vom 7.7.95 im Bamberg, Universität Bamberg, Bamberg 1995, S. 2 - 14
[16] Sumner, T; Stolze, M.: Evolution, not Revolution: PD in the Toolbelt Era, in: Proceeding of the Third Decennial Conference: Computers in Context: Joining Forces in Design, Aarhus, DK, August 14 - 18, 1995, S. 30 - 39
[17] Wulf, V.: Konfliktmanagement bei Groupware, Vieweg, Braunschweig 1997 (in Vorbereitung)

Adressen der Autoren

Helge Kahler
Universität Bonn
Institut für Informatik III
Projektbereich Software-Ergonomie und CSCW
Römerstr. 164
53117 Bonn
Email: kahler@informatik.uni-bonn.de

Dr. Volker Wulf
Universität Bonn
Institut für Informatik III
Projektbereich Software-Ergonomie und CSCW
Römerstr. 164
53117 Bonn
Email: volker@informatik.uni-bonn.de