*M. Herczeg, W. Prinz, H. Oberquelle (Hrsg.): Mensch & Computer 2002: Vom interaktiven Werkzeug zu kooperativen Arbeits- und Lernwelten. Stuttgart: B. G. Teubner, 2002, S. 235-244.*

# Instruktionen und Usability: Empirische Untersuchung anhand eines Tools für Entwickler

Gerd Jan Tschöpe & Julia Nitschke
Humboldt-Universität zu Berlin, Institut für Psychologie
Lehrstuhl für Ingenieurspsychologie

### Zusammenfassung

In einer empirischen Untersuchung mit 40 Software-Entwicklern wurde der Einfluss zweier Instruktionsformen auf die Erlernung einer komplexen Software für professionelle Anwender untersucht. Eine Gruppe von Entwicklern bekam eine konzeptuelle Instruktion, die die Bildung eines mentalen Modells des Systems unterstützte. Die andere Gruppe wurde mit Hilfe eines minimalen Manuals instruiert, das handlungsrelevantes, prozedurales Wissen vermittelte. Die Entwickler mussten vorgegebene Aufgaben mit der Software bearbeiten. Anhand der aufgezeichneten Logfiledaten wurde die Effizienz der Benutzung und die Orientierung im System erfasst. Es zeigte sich, dass bei der Benutzung von Funktionen, die nicht Bestandteil der Instruktion waren, die konzeptuell instruierte Gruppe eine höhere Effizienz aufwies. Weiterhin zeigte diese Gruppe bei zeitkritischen Aufgaben eine effizientere Orientierung im System.

## 1 Einleitung

Die hohe Komplexität und der große Funktionsumfang professioneller Software-Anwendungen erfordert neben der Optimierung der Screen- und Dialoggestaltung auch den Einsatz von schriftlicher Information über die Benutzung eines Systems. Obwohl es wünschenswert und anzustreben ist, eine Applikation so zu gestalten, dass auf den Gebrauch einer Anleitung weitgehend verzichtet werden kann, ist dies bei sehr umfangreichen Systemen oft kaum zu realisieren (z. B. Preece, 1994; Carroll, 1998; Shneiderman, 1998).

Eine angemessene Instruktion soll einen Erstbenutzer in die Lage versetzen, die effiziente Benutzung der komplexen Software zu erlernen, auftretende Probleme zu lösen sowie Fehler zu vermeiden bzw. begangene Fehler zu diagnostizieren und zu korrigieren. Somit ist die Frage nach einer angemessenen Instruktionsgestaltung eng mit kognitionspsychologischen Modellen des Problemlösens und der Wissensbildung verknüpft. Gleichzeitig ist die Untersuchung der Erlernbarkeit der Benutzung von Software eine klassische Fragestellung der Usability-Forschung (z. B. Nielsen, 1993). In der von uns durchgeführten Studie wurde der Einfluss zwischen unterschiedlichen Instruktionsformen auf die Erlernbarkeit des verwendeten Systems untersucht. Dabei wurden Usability-Kriterien wie Effizienz und Orientierung im System verwendet, um die Hypothesen über die Effekte der durch die Instruktionen moderierten kognitiven Prozesse auf die Erlernbarkeit der Software zu überprüfen.

# 2 Mentale Modelle oder handlungsorientiertes Wissen?

In der durchgeführten Untersuchung wurden Modelle der Kognitionspsychologie zur Wissensbildung auf das Forschungsgebiet der Human Computer Interaction (HCI) angewendet. Dabei lassen sich zwei für die software-ergonomische Praxis besonders relevante Ansätze gegenüber stellen: Die ACT* bzw. ACT-R-Theorie des Erwerbs prozeduralen Wissens durch die Bildung von Produktionssystemen nach Anderson (1983, 1993) und die Theorienfamilie der Konstruktion mentaler Modelle im Sinne von Gentner und Gentner (1983) bzw. Johnson-Laird (1983). Aus diesen grundlegenden Theorien der Kognitionspsychologie lassen sich zwei unterschiedliche Prinzipien der Gestaltung von Instruktionen zur Benutzung von Software ableiten: Das minimale Manual nach Carroll (1998) und die Instruktion durch Metaphern oder konzeptuelle Modelle (z. B. Dutke, 1994).

## 2.1 Prozedurale vs. konzeptuelle Instruktion

Nach dem Modell von Anderson (1983) wird prozedurales Wissen, also „knowledge about how to do things“ (Anderson, 1983; S. 215) erworben, indem zunächst deklaratives Wissen in Form sogenannter Produktionsregeln gebildet wird. Diese *condition-action pairs* werden dem Modell nach in einer zweiten Phase der Wissenskompilierung zu aufgabespezifischen Regeln verdichtet, die in der dritten Phase dann automatisiert werden und das prozedurale Wissen bilden. Daraus folgt, dass das Erlernen einer Software erleichtert wird, wenn die Instruktion zur Benutzung dieser Software die Bildung eines Produktionssystems unterstützt. Dies kann zum Beispiel durch die Vorgabe von Produktionsregeln oder Beispiellösungen geschehen, was nach Anderson (1993) zu einer Reduktion erforderlicher Vergleichsoperationen und somit zu einer Entlastung des Arbeitsgedächtnisses führt.

Diese Theorie bietet ein tragfähiges Fundament für den eher pragmatischen Ansatz zur Gestaltung von Instruktion in Form minimaler Manuale nach Carroll (1990; 1998). Das minimale Manual zeichnet sich im Gegensatz zu einer herkömmlichen Bedienungsanleitung durch eine Reduktion des zu lesenden Materials, eine aufgabenorientierte Instruktion und einem Fokus auf Fehlererkennung und -behebung aus. Der Benutzer wird in seiner Eigenaktivität durch eine möglichst große inhaltliche Unabhängigkeit der einzelnen Abschnitte des Manuals unterstützt. Auf eine einleitende Explikation des Systems wird zu Gunsten einer handlungsorientierten Instruktion, die sich an den Aufgaben des Benutzers orientiert, verzichtet. In einem umfassenden Überblick führen McCreary und Carroll (1998) eine Vielzahl empirischer Untersuchungen auf, in denen nach diesen Heuristiken gestaltete minimale Manuale positive Effekte auf die Erlernbarkeit unterschiedlichster Applikationen hatten. Dies äußerte sich u.a. in einer kürzeren Lernphase, in einer geringeren Anzahl der begangenen Fehler und in einer effizienteren Bearbeitung von im Anschluss an die Lernphase gestellten Aufgaben.

Das minimale Manual vermittelt prozedurales Wissen, das *während der Benutzung* zur Verfügung gestellt wird, und verzichtet auf eine vorangestellte Darstellung des Systemaufbaus. Der Ansatz der Instruktion durch ein minimales Manual wird im Folgenden als *prozedurale Instruktion* bezeichnet.

Der Ansatz der konzeptuellen Instruktion verfolgt ein anderes Ziel. Die Instruktion soll hier vor allem die Bildung eines adäquaten mentalen Modells der Software fördern. Mentale Modelle sind bildhaft-anschauliche Abbildungen von Objekten, Sachverhalten oder Phänomenen, die der Erklä-

rung, dem Erkenntnisgewinn oder der Vorhersage dienen (Dutke, 1994). Mentale Modelle erlauben die kognitive Simulation von Systemzuständen und erleichtern das Verständnis funktioneller Zusammenhänge (Gentner und Gentner, 1983).

Daraus folgt für die Gestaltung einer Software-Instruktion, dass sie den Prozess der mentalen Modellbildung möglichst gut unterstützen sollte. Zum einen, weil Personen offenbar ohnehin dazu tendieren, ein solches Modell zu konstruieren (Johnson-Laird, 1983), zum anderen, weil die Fähigkeit zur Lösung von Problemen bei der Benutzung einer Applikation von der Qualität des mentalen Modells beeinflusst wird (z. B. Kieras und Bovair; 1984), weil mentale Modelle in einem Prozess der kognitiven Simulation manipuliert werden können. Mentale Modelle weisen eine mehrdimensionale Qualität auf, und erleichtern dadurch die Repräsentation räumlicher Beziehungen (Johnson-Laird, 1983). Die Bildung eines mentalen Modells kann durch Analogiebildung anhand einer Metapher oder eines vorgegebenen konzeptuellen Modells (sensu Jonassen, 1997) gefördert werden. Auch hier gibt es zahlreiche empirische Belege für die Wirksamkeit dieser Instruktionsform (vg. hierzu Sasse, 1997).

Die Instruktion anhand einer Metapher findet *vor der Benutzung* statt und vermittelt strukturelle Zusammenhänge des Systems, ohne auf die konkreten Benutzungshandlungen einzugehen. Dieser Ansatz der Instruktion wird im Folgenden als *konzeptuelle Instruktion* bezeichnet.

## 2.2 Was leisten diese Instruktionsformen?

Vertreter der Instruktion durch minimale Manuale wie z. B. Mirel (1998) oder Hackos (1998) erheben genau wie Vertreter einer Instruktion anhand einer Metapher wie z.B. Greenberg et al. (1998) den Anspruch, mit ihrer Methode den Benutzer einer Software bestmöglich in seinem Lernprozess zu unterstützen, und legen empirische Belege für ihre Position vor. Daraus ergibt sich die Frage, ob dieser Anspruch gerechtfertigt ist, wo die Stärken und Schwächen der einzelnen Instruktionsmethoden im direkten Vergleich liegen, und ob sich Effekte der Instruktion auf die Usability einer Software abgrenzen lassen.

Zur Untersuchung dieser Frage wurden die folgenden Hypothesen formuliert, die sich auf den Beginn der Benutzung der Software beziehen, weil es nach einer ausreichend langen Übungszeit zu einer Nivellierung von Performanzunterschieden kommt, die auf die Methode der Instruktion zurückzuführen sind (z. B. Hacker, 1998)

Die *prozedurale Instruktion* durch ein minimales Manual sollte aufgrund ihrer aufgabenorientierten Gestaltung zu einer leichten Verfügbarkeit von Informationen über die erforderlichen Handlungsschritte führen. Das gleiche gilt auch für Informationen zur Fehlererkennung und deren Korrektur bzw. Vermeidung. Dies ermöglicht eine leichte Bildung von Produktionsregeln, die aus einem Ziel und einer Prozedur zur Realisierung des Ziels bestehen, und mit geringem kognitiven Aufwand abgerufen werden können (Anderson, 1993). Bei der Aufgabenbearbeitung entfallen dadurch unspezifische „trial-and-error“ Prozeduren sowie umfangreiche Such- und Vergleichsprozesse von Inhalten des Langzeitgedächtnisses nach möglichen Lösungen. Deswegen sollte die *prozedurale Instruktion* in Form eines minimalen Manuals eine effiziente Benutzung der Software erleichtern und zu einer Entlastung des Arbeitsgedächtnis führen. Dies gilt jedoch nur für die Benutzung von Funktionen der Software, die Gegenstand der Instruktion sind.

Somit wird als erste Hypothese formuliert, dass die *prozedurale Instruktion* durch ein minimales Manual die Effizienz der Software-Benutzung steigert, wenn die benutzten Funktionen Gegenstand der Instruktion sind.

Die *konzeptuelle Instruktion* durch eine Metapher sollte die Bildung eines adäquaten mentalen Modells fördern, was den Benutzern die Einsicht in funktionelle Zusammenhänge erleichtern sollte. Durch kognitive Manipulation des mentalen Modells, sollten derart instruierte Benutzer besser in der Lage sein, zur Benutzung des Systems erforderliche Prozeduren selbständig abzuleiten, wenn diese nicht Bestandteil der Instruktion sind. Dies sollte zu einer effizienteren Benutzung der Software bei der Bearbeitung von Aufgaben führen, deren Lösungsweg nicht bekannt ist.

Somit wird als zweite Hypothese formuliert, dass die *konzeptuelle Instruktion* durch eine Metapher in Verbindung mit einem konzeptuellen Modell zu einer Steigerung der Effizienz der Software-Benutzung führt, wenn die benutzten Funktionen *nicht* Gegenstand der Instruktion sind.

Aufgrund der guten kognitiven Repräsentation räumlicher Beziehungen durch mentale Modelle, sollten Personen die ein adäquates mentales Modell gebildet haben, gut in der Lage sein, sich in der komplexen Software zu orientieren. Deswegen wird als dritte Hypothese formuliert, dass die *konzeptuelle Instruktion* in Benutzungssituationen, bei denen vor allem die *Orientierung* in der Software von Bedeutung ist, zu einer größeren *Effizienz* führt.

## 3 Methode: Benutzer-Test zum Vergleich der Instruktionen

Die Hypothesen wurden in einem Benutzer-Test anhand zwei verschiedener Instruktionen für das Software-Tool GUIDEAS überprüft. GUIDEAS ist ein Tool zur Konzipierung von Assistenzsystemen, das gegenwärtig im Rahmen des EMBASSI Projekts (BMBF Fkz. 01IL904I) entwickelt wird. Das Akronym steht für *Guidance for Developing Assistance*. GUIDEAS richtet sich an Entwickler, die Geräte oder Softwareanwendungen mit einer Assistenzfunktion ausstatten wollen. Der Benutzer des Geräts oder der Software soll durch die Assistenzfunktion bei der Mensch-Maschine Interaktion unterstützt werden. Mit GUIDEAS lässt sich die Gestaltung dieser Assistenzfunktion konzipieren.

Am Anfang der Benutzung von GUIDEAS wird der Entwickler durch eine Anforderungsanalyse geleitet, in deren Rahmen GUIDEAS vom Entwickler Angaben über die Benutzer seiner Anwendung, die Aufgaben die sie damit ausführen, die situativen Bedingungen und die Ziele, die mit der Assistenzfunktion verfolgt werden, abfragt. Auf Basis dieser Angaben generiert das in GUIDEAS implementierte Expertensystem unter Verwendung von gewichteten Wenn-Dann-Regeln einen Vorschlag zur Gestaltung der Assistenzfunktion.

In der Anforderungsanalyse werden zum Teil sehr spezifische Angaben abgefragt, deswegen ist es möglich, dass der Entwickler nicht alle erforderlichen Daten in die Anforderungsanalyse einspeisen kann. Für diesen Fall stellt GUIDEAS ein Modul mit Methoden zur Verfügung. Diese können vom Entwickler angewendet werden, um die fehlenden Daten zu erheben. Die so erhobenen Daten werden dann gleichfalls in die Anforderungsanalyse eingegeben. Sollte das System trotz einer vollständigen Anforderungsanalyse nicht in der Lage sein, einen angemessenen Assistenzvorschlag zu generieren, besteht die Möglichkeit, mit Hilfe eines weiteren Moduls den Assistenzbedarf selbständig zu ermitteln. Die dort zur Verfügung gestellten Methoden dienen nicht der Vervollständigung der Anforderungsanalyse, sondern der eigenständigen Ermittlung eines Assistenzkonzepts durch den Entwickler.

Zur Überprüfung der Hypothesen über den Einfluss der Instruktion auf die Benutzung der Software wurde von den Autoren der Untersuchung eine prozedurale und eine konzeptuelle Instruktion für GUIDEAS entwickelt.

Zur Gestaltung der prozeduralen Instruktion wurde anhand der von Carroll (1998) vorgeschlagenen Heuristiken ein minimales Manual zur Benutzung der Software entwickelt. Um eine unmittelbare Möglichkeit zum Handeln zu bieten, wurde das Manual direkt in die Benutzeroberfläche integriert. Dadurch sollte gewährleistet werden, dass die jeweils zur Benutzung des Programms erforderlichen Informationen leicht verfügbar sind. Die Bereiche der Software, die das minimale Manual enthielten, werden im Folgenden als *Info-Screens* bezeichnet.

Zur Gestaltung der *konzeptuellen Instruktion* wurde die Metapher eines Besuchs in einem medizinisch betreuten Fitnessstudio verwendet. Dabei bestanden Analogiebeziehungen zwischen dem zeitlichen Ablauf des geschilderten Besuchs und dem Workflow der Software, zwischen der Funktion der Räume des Studios und den Arbeitsbereichen der Software, sowie zwischen Diagnose und Anforderungsanalyse bzw. Trainingsplan und Assistenzvorschlag. Um eine korrekte Übertragung der funktionellen Eigenschaften und der strukturellen Beziehungen sicherzustellen, wurde zusätzlich zu der Metapher ein konzeptuelles Modell in Form eines Flowchart-Diagramms eingesetzt.

Um die Wirkung der beiden Instruktionsformen auf die Effizienz der Benutzung von GUIDEAS zu untersuchen, wurden zwei Versuchsgruppen gebildet, die jeweils aus zwanzig Versuchspersonen bestanden. In jeder Gruppe wurde eine der beiden Instruktionen verwendet. Nach einer zehnminütigen Explorationsphase wurden den Versuchspersonen folgende vier Aufgaben in schriftlicher Form vorgelegt:

1. Erarbeitung eines Assistenzvorschlags für ein in der Aufgabe spezifiziertes System (ASS)
2. Auswahl einer Methode zur Datenerhebung für die Anforderungsanalyse (METH)
3. Selbständige Ermittlung des Assistenzbedarfs (SELB)
4. Orientierung im System (ORIE)

Die ersten beiden Aufgaben (ASS und METH) erforderten die Benutzung von Elementen der Software, die Bestandteil der Instruktion waren. Deswegen wurde angenommen, dass sie von der Gruppe *prozedurale Instruktion* schneller bearbeitet werden. Die dritte Aufgabe (SELB) erforderte die eigenständige Erarbeitung eines Lösungsweges. Die vierte Aufgabe (ORIE) stellte vor allem Anforderungen an die Orientierungsfähigkeit der Versuchspersonen im Programm. Sie bestand aus elf Unteraufgaben, zu deren Beantwortung unterschiedliche Bereiche der Anwendung aufgesucht werden mussten. Die Vpn wurden aufgefordert die Aufgaben so schnell wie möglich zu bearbeiten. Für diese Aufgaben wurde angenommen, dass sie von der *Gruppe konzeptuelle Instruktion* schneller bearbeitet werden

Die Effizienz der Benutzung von GUIDEAS wurde durch die Zeit operationalisiert, welche die Versuchspersonen zur Aufgabenbearbeitung benötigten, und die vom System in Form von Logfiles gemessen wurde.

Die Untersuchung wurde mit Software-Entwicklern durchgeführt. Dazu wurden per e-mail Softwarefirmen angeschrieben. Die so rekrutierte Stichprobe bestand aus 40 Personen, die im Mittel 33 Jahre alt waren (S = 5.68); die jüngste Vp war 23, die älteste 49 Jahre alt. Die Stichprobe bestand aus drei Frauen und 37 Männern. Alle Personen waren beruflich mit der Konzeption von Benutzerschnittstellen befasst, 37 Personen darüber hinaus auch mit der Programmierung technischer Systeme einschließlich der Benutzerschnittstelle.

# 4 Ergebnisse

## 4.1 Wurde das Manual benutzt?

Um zu überprüfen, ob die Probanden das minimale Manual auch tatsächlich gelesen haben, wurden zunächst die Info-Zeiten (Verweildauer auf den aufgerufenen Info-Screens) während des gesamten Tests betrachtet.

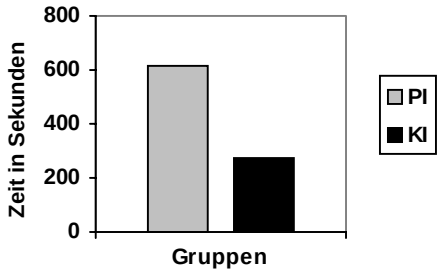


*Abbildung 1: Verweildauer auf den Info-Screens*

Es zeigte sich, dass die Personen der *prozeduralen Instruktion* hochsignifikant länger auf den Info-Screens verweilten ($AM_{PV}$ = 615.80 Sek., S = 98.54; $AM_{KV}$ = 275.25 Sek., S = 48.03, df = 38, p = 0.001**) als die Probanden der *konzeptuellen Instruktion*. Da der einzige Unterschied zwischen Info-Screens beider Gruppen im minimalen Manual bestand, ist es plausibel anzunehmen, dass die bedeutsam längeren Info-Zeiten der Gruppe *prozedurale Instruktion* auf das Lesen des minimalen Manuals zurückzuführen sind.

## 4.2 Der Einfluss der Instruktion auf die Effizienz

Zur Auswertung der Lösungszeiten wurden in beiden Gruppen die Info-Zeiten von den Lösungszeiten subtrahiert. Dies war erforderlich, um die Gruppe *prozedurale Instruktion* gegenüber der Gruppe *konzeptuelle Instruktion* nicht zu benachteiligen, der die Instruktion wie in Abs. 2.1 beschrieben, *vor* der Benutzung der Software dargeboten wurde.

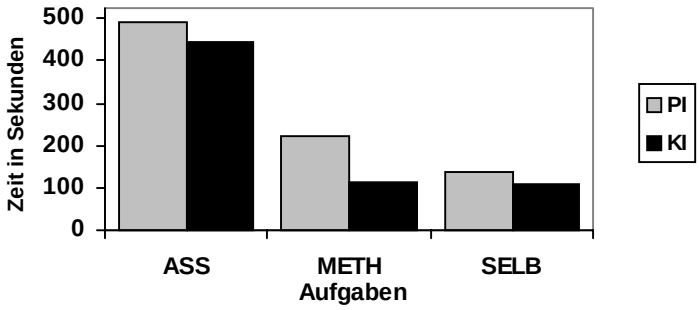


*Abbildung 2: Durchschnittliche Lösungszeiten ohne Info-Zeiten*

Ein Vergleich der Mittelwerte mit einem t-Test zeigt, dass es bei der ersten Aufgabe (ASS) keinen statistisch bedeutsamen Unterschied zwischen den Gruppen gibt. Bei dem Mittelwertsvergleich der zweiten Aufgabe (METH) wurde ein nonparametrisches Verfahren eingesetzt, da nicht alle Personen die Aufgabe gelöst haben, was zu ungleichen Stichprobengrößen führte ($n_{PV} = 17$, $n_{KV} = 19$). Die Kennwerte der Mittelwertsvergleiche werden in Tabelle 1 dargestellt.

*Tabelle 1: Vergleich der Lösungszeiten in Sekunden*

| Aufgabe | M (PI) | SD (PI) | M (KI) | SD (KI) | Kennwerte | | |
|---|---|---|---|---|---|---|---|
| ASS | 492.80 | 166.81 | 443.65 | 134.72 | df = 38 | t = 1.025 | p = 0.156 |
| METH | 224.12 | 124.77 | 111.68 | 35.03 | U = 64 | Z = -3.22 | p = 0.001** |
| SELB | 138.30 | 65.14 | 107.65 | 51.72 | df = 25 | t = 3.951 | p = 0.094 |

Es zeigten sich hochsignifikant kürzere Lösungszeiten für die Probanden der Gruppe *konzeptuelle Instruktion* in der zweiten Aufgabe (METH). Bei der dritten Aufgabe (SELB) besteht ein Trend der Gruppe *konzeptuelle Instruktion* zu kürzeren Lösungszeiten.

## 4.3 Der Einfluss der Instruktion auf die Orientierung

Bei der Orientierungs-Aufgabe (ORIE), mussten die Versuchspersonen nacheinander in elf Unteraufgaben verschiedene Arbeitsbereiche des Programms aufsuchen. Dabei wurde die Anzahl der richtigen Lösungen über alle elf Unteraufgaben pro Versuchsperson zu einem Summenscore zusammengefasst, der maximal erreichbare Wert beträgt elf. In der Abbildung 3 werden die Mittelwerte dieses Summenscores für jede Gruppe dargestellt. Die Abbildung 4 zeigt die durchschnittliche Lösungszeit pro Unteraufgabe als Gruppenmittelwert.

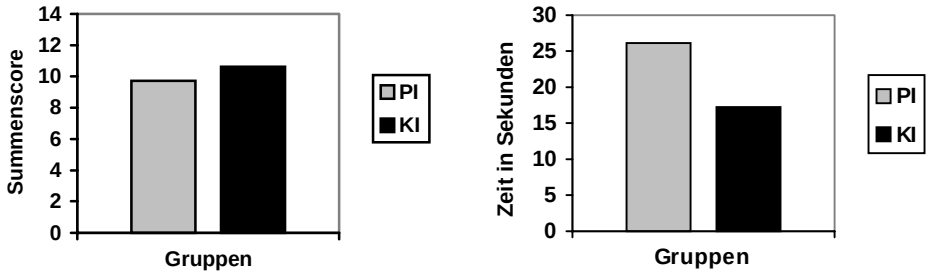


*Abbildung 3 und 4: Gruppenvergleich Summenscore Lösungen und Lösungszeiten, Aufgabe ORIE*

Der Mittelwert dieses Summenscores ist bei der prozeduralen Instruktion in der Aufgabe ORIE niedriger ($M_{PI} = 9.75$, SD = 1.21) als bei der konzeptuellen Instruktion ($M_{KI} = 10.65$, SD = 0.59). Der U-Test von Mann-Whitney ergibt einen Prüfwert von $Z_{ORIE} = -1.35$. Das entspricht einer einseitigen Irrtumswahrscheinlichkeit von p = 0.09. Somit besteht ein Trend dahingehend, dass die Versuchspersonen der Gruppe *konzeptuelle Instruktion* durchschnittlich mehr richtige Lösungen bei der Orientierungsaufgabe aufweisen als die Probanden der Gruppe *prozedurale Instruktion*. Gleichzeitig zeigen sich kürzere Lösungszeiten bei der Gruppe konzeptuelle Instruktion ($M_{PI} = 25.99$ Sek., SD = 8.46; $M_{KI} = 17.16$, SD = 5.11). Dieser Unterschied zu Gunsten der konzeptuelle Instruktion ist statistisch hochsignifikant (df = 32, t = 1.351, p = 0.001**).

## 5 Diskussion der Ergebnisse und Schlussfolgerungen

Zum Erwerb von Wissen über ein technisches System bedarf es einer Interaktion mit dem System. Eine sinnvolle und lehrreiche Interaktion setzt jedoch oft Wissen voraus. Das Lesen einer Anleitung wird aber häufig als lästig empfunden. Die hochsignifikant längeren Verweilzeiten der Probanden der Gruppe *prozedurale Instruktion* auf den Info-Seiten stützen jedoch die Argumentation Carrolls (1998), dass das minimale Manual aufgrund der guten Verfügbarkeit der Informationen und der Möglichkeit, die Inhalte ohne vorgeschriebene Reihenfolge zu lesen, von den Benutzern eines Systems auch tatsächlich verwendet wird. Es kann aufgrund der vorliegenden Daten also davon ausgegangen werden, dass die Probanden der Gruppe *prozedurale Instruktion* das minimale Manual genutzt haben, um die Benutzung von GUIDEAS zu erlernen und die Aufgaben zu bearbeiten. Dies spricht dafür, Benutzern statt einer herkömmlichen Bedienungsanleitung ein minimales Manual zur Verfügung zu stellen, weil konventionelle Anleitungen oft nicht benutzt werden (Carroll, 1990).

In der ersten Hypothese wurde angenommen, dass die *prozedurale Instruktion* eine effiziente Benutzung der Software fördert, wenn die zu bearbeitenden Aufgaben durch die Instruktion angleitet werden. Das Wissen über die erforderlichen Benutzungsschritte ist in diesen Situationen schnell verfügbar, weil es vom minimalen Manual direkt bereitgestellt wird. Diese Hypothese wurde anhand der ersten beiden Aufgaben (ASS und METH) überprüft. Die zur Bearbeitung dieser Aufgaben erforderlichen Handlungsschritte wurden im minimalen Manual beschrieben. Die Ergebnisse bestätigen diese Hypothese nicht: In der ersten Aufgabe (ASS) weisen beide Gruppen

die gleiche Effizienz auf, in der zweiten Aufgabe (METH) ist die Gruppe *konzeptuelle Instruktion* sogar hochsignifikant *effizienter* als die Gruppe prozedurale Instruktion.

In der zweiten Hypothese wurde davon ausgegangen, dass die *konzeptuelle Instruktion* die effiziente Benutzung der Software fördert, wenn Aufgaben bearbeitet werden, die nicht durch die Instruktion abgedeckt werden. Dies wurde mit der besseren mentalen Modellbildung begründet, die derart instruierten Versuchspersonen das Ableiten eigenständiger Lösungswege erleichtert. Diese Hypothese wurde anhand der dritten Aufgabe überprüft (SELB). Um diese Aufgabe erfolgreich zu bearbeiten, mussten Funktionen der Software benutzt werden, die nicht in der Instruktion beschrieben wurden. Hier zeigte sich ein Trend der Gruppe *konzeptuelle Instruktion* zu einer größeren Effizienz der Aufgabenbearbeitung.

In der dritten Hypothese wurde für Versuchspersonen, die mit der *konzeptuellen Instruktion* instruiert wurden eine größere Effizienz angenommen, wenn bei der Bearbeitung der Aufgabe vor allem die Orientierung in der Software von Bedeutung ist. Auch diese Hypothese wurde mit einer Förderung der mentalen Modellbildung begründet, welche die kognitive Repräsentation räumlicher Beziehungen erleichtert. Zur Überprüfung diente die vierte Aufgabe (ORIE), in der die Versuchspersonen nacheinander elf verschiedene Bereiche des Programms aufrufen sollten. Hier zeigte sich in den hochsignifikant kürzeren Lösungszeiten der Gruppe *konzeptuelle Instruktion*, dass diese Form der Instruktion wie angenommen zu einer größeren Effizienz führt. Dieser Befund wird durch den Trend dieser Gruppe zu mehr richtigen Lösungen gestützt. Offenbar erleichtert die *konzeptuelle Instruktion* die Orientierung in dem benutzten Software-System. Dies steht in Kongruenz zu Befunden aus der Literatur, die einen positiven Zusammenhang zwischen der Qualität der mentalen Modellbildung und der kognitiven Repräsentation räumlicher Strukturen berichten (z.B. Johnson-Laird, 1983).

Zusammenfassend lässt sich anhand der Ergebnisse der Untersuchung die Frage nach einer geeigneteren Instruktionsform zum Erlernen einer komplexen Software klar zugunsten der *konzeptuellen Instruktion* beantworten. Diese Methode der Instruktion durch eine Metapher in Verbindung mit einem konzeptuellen Modell erscheint sowohl besser geeignet, die Benutzung von Funktionen der Software zu instruieren, als auch das strukturelle Verständnis der Benutzer zu verbessern. Die konzeptuelle Instruktion führt zu einer besseren Orientierung in der Software und tendenziell auch zu einer effizienteren Benutzung von Funktionen, die nicht durch die Instruktion erklärt werden.

## 6 Literatur

Anderson, J. R. (1983). *The Architecture of Cognition.* Cambridge: Harvard University Press.

Anderson, J. R. (1993). *Rules of the mind.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Carroll, J. M. (1990). *The Nurnberg Funnel. Designing Minimalist Instruction for Practical Computer Skill.* Cambridge, MA: MIT Press.

Carroll, J. M. (1998). *Minimalism beyond the Nurnberg funnel.* Cambridge, MA: MIT Press.

Dutke, S. (1994). *Mentale Modelle: Konstrukte des Wissens und Verstehens.* Göttingen, Stuttgart: Verlag für angewandte Psychologie.

Gentner, D., & Gentner, R. G. (1983). Flowing Waters or Teeming Crowds: Mental Models of Electricity. In D. Gentner & A. L. Stevens (Eds.), *Mental Models* (pp. 99-127). Hillsdale, NJ: Lawrence Erlbaum Associates.

Greenberg, S., & Roseman, M. (1998). *Using a Room Metaphor to Ease Transitions in Groupware. Research report 98/611/02.* Calgary: University of Calgary, Department of Computer Science.

Hacker, W. (1998). *Allgemeine Arbeitspsychologie. Psychische Regulation von Arbeitstätigkeiten.* Bern: Huber.

Hackos, J. T. (1998). Choosing a minimalist approach for Expert Users. In. Carroll, J. M (Ed.), *Minimalism beyond the Nurnberg funnel* (pp. 149-177). Cambridge MA: The MIT Press.

Johnson-Laird, P. N. (1983). *Mental models: Towards a cognitive science of language, inference and consciousness.* Cambridge, MA: Harvard University Press.

Johnson-Laird, P. N., Girotto, V., & Legrenzi, P. (1998). *Mental models: a gentle guide for outsiders.*

Jonassen, D. H. (1995). *Operationalizing Mental Models: Strategies for Assessing Mental Models to Support Meaningful Learning and Design-Supportive Learning Environments.* [WWW-Document] Available URL : http://cica.cica.indiana.edu/cscl95/jonassen.html.

Kieras, D. E.& Bovair, S. (1984). The Role of a mental model in learning to operate a device. *Cognitive science*, *8*, 522-273.

Mirel, B. (1998). Minimalism for complex tasks. In. Carroll, J. M (Ed.), *Minimalism beyond the Nurnberg funnel* (pp. 179-218). Cambridge MA: The MIT Press.

Nielsen, J. (1993). *Usability Engineering.* Cambridge, MA: Academic Press.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. & Carey, T. (1994). *Human-Computer Interaction.* Wokingham: Addison-Wesley Publishing Company.

Sasse, M. A. (1997). *Eliciting and Describing Users' Models of Computer Systems*. [WWW-Document] Available URL: http://www.cs.ucl.ac.uk/staff/angela/thesis/Contents.html.

Shneiderman, B. (1998). *Designing the user interface: Strategies for effective human-computer interaction.* Reading, MA: Addison-Wesley.

## Kontaktinformationen

Gerd Jan Tschöpe & Julia Nitschke
Humboldt-Universität zu Berlin
Lehrstuhl für Ingenieurspsychologie
Oranienburgerstr. 18
10178 Berlin

Email: gerdjan_tschoepe@web.de
julia.nitschke@rz.hu-berlin.de