



Leitfaden für die Erstellung von Softwareentwicklungswerkzeugen für die Kooperation von Menschen mit und ohne Sehschädigung

Version 1.0

Claudia Loitsch

Karin Müller

Juni 2018

Projektpartner:



Gefördert durch:



INHALTSVERZEICHNIS

1	EINLEITUNG	4
2	GRUNDLAGEN - MENSCHEN MIT SEHBEINTRÄCHTIGUNG	5
2.1	Wann liegt eine Sehbehinderung oder Blindheit vor?	5
2.2	Prävalenz von Sehbehinderung oder Blindheit	7
2.3	Ursachen von Sehbehinderungen oder Blindheit	7
2.4	Leben mit einer Sehbehinderung oder Blindheit	8
2.5	Wie Menschen mit Sehbehinderung digitale Geräte und Informationen nutzen	8
2.6	Zusammenarbeit von Menschen mit und ohne Seheinschränkungen	11
3	IDES - BARRIEREFREIE BEDIENUNG VON ECLIPSE	15
3.1	Navigation im Quellcode-Editor	15
3.2	Erkennen von Programmierfehlern und Debuggen	16
3.3	Überblick über die wichtigsten Tastaturkürzel	17
3.3.1	Eclipse Shortcuts zur allgemeinen Navigation in der Anwendung	18
3.3.2	Eclipse Shortcuts zur Dateiverwaltung	18
3.3.3	Eclipse Shortcuts zur Navigation im Editor	18
3.3.4	Eclipse Shortcuts zur Programmierung	18
3.3.5	Eclipse Shortcuts zur Versionsverwaltung mit SVN	19
4	FALLSTUDIE - GEMEINSAM SOFTWARE ENTWICKELN MIT UML	20
4.1	Erfahrungsbericht eines blinden Softwareentwicklers zum Einsatz von UML	20
4.1.1	Wie wird UML eingesetzt und welche Diagramme werden verwendet?	20
4.1.2	Was mit den UML Diagrammen passiert?	21
4.1.3	Welche Barrieren entstehen durch fehlende barrierefreie Kooperationswerkzeuge?	21
4.2	Das Cooperate Modellierungswerkzeug	22
4.2.1	Der grundlegende Ansatz	22
4.2.2	Editierunterstützung	22
4.2.3	Weitere Orientierungsfunktionen	24
4.2.4	Synchronisation	25
4.2.5	Versionskontrolle	25
4.2.6	Unterstützung von Diskussionen über Diagramme	26
5	ZUSAMMENFASSUNG	27
6	QUELLEN	28

1 EINLEITUNG

Die UN-Konvention sichert Menschen mit Behinderung das Recht auf Arbeit in einem offenen, inklusiven und zugänglichen Arbeitsumfeld zu. Menschen mit Sehschädigung, die in der Softwareentwicklung arbeiten, stehen jedoch oft vor dem Problem, dass sie in ihrem beruflichen Umfeld nur eingeschränkt mit ihrem sehenden Kollegium zusammenarbeiten können. Ein häufiger Grund hierfür sind unzugängliche Programmierumgebungen (IDEs) oder Kooperationswerkzeuge (Versionskontrolle wie SVN oder Git), die in der Softwareentwicklung eingesetzt werden.

Mit diesem Thema setzte sich das vom Bundesministerium für Arbeit und Soziales geförderte dreijährige Cooperate-Projekt¹ auseinander. Die erste Aufgabe des Cooperate-Projektes war, ein Kooperationswerkzeug² zu entwerfen, das je nach Arbeitsweise die graphische Beschreibungssprache UML visuell oder textuell darstellt und ein Lesen und Bearbeiten von Diagrammen ermöglicht. Die zweite Aufgabe bestand darin, ein Trainingskonzept und Schulungsunterlagen³ zu entwerfen, das für das Selbststudium und für die Weiterbildung geeignet ist, damit Menschen mit Sehschädigung einen leichteren Zugang zu der graphischen Beschreibungssprache UML und zu Programmierumgebungen wie Eclipse erhalten. Die dritte Aufgabe des Cooperate-Projektes war, die Übertragbarkeit der Ergebnisse auf die Entwicklung neuer Werkzeuge sicherzustellen. Das Ergebnis ist dieser Leitfaden. Er beschreibt den entwickelten Lösungsansatz des Cooperate-Projektes und soll dadurch ein Ausgangspunkt für die Entwicklung anderer Kooperationswerkzeuge dienen, welche die Zusammenarbeit in Diversity Teams adressieren.

Der vorliegende Leitfaden gibt Hinweise zur Entwicklung von barrierefreien inklusiven Kooperationswerkzeugen – also softwarebasierte Lösungen, welche die Zusammenarbeit von Menschen mit unterschiedlichen, insbesondere visuellen, Fähigkeiten unterstützen. Die Zielgruppe dieses Leitfadens sind Personen, die sich mit der Entwicklung von Modellierungswerkzeugen beschäftigen. Sie sollen durch diesen Leitfaden in die Lage versetzt werden, die neuen Lösungsansätze auf ihre Werkzeuge zu übertragen und deren Barrierefreiheit herzustellen.

Der Leitfaden gliedert sich in drei Teile. Zunächst werden einige *grundlegende Informationen zum Thema Sehschädigung* gegeben und wichtige Anforderungen von Menschen einer Sehschädigung an die Arbeit mit digitalen Informationen und Kommunikationstechnologien beschrieben. Außerdem gehen wir im ersten Teil des Leitfadens auf die Zusammenarbeit zwischen Menschen mit und ohne Sehschädigung ein. Der zweite Teil des Leitfadens beschreibt dann konkret die *Barrierefreiheit von Softwareentwicklungswerkzeugen am Beispiel von Eclipse*. Dies ist wichtig, um aktuelle Barrieren von sehbehinderten Menschen im Umgang mit Programmierumgebungen zu verdeutlichen und auch Arbeitstechniken zu beschreiben. Der dritte Teil hat die *barrierefreie Softwareentwicklung mit UML* zum Fokus und beschreibt das Konzept des entwickelten Kooperationswerkzeuges.

¹ <https://www.cooperate-project.de/index.php/de/>

² <https://www.cooperate-project.de/index.php/de/downloadmenu/entwicklung>

³ <https://www.cooperate-project.de/index.php/de/downloadmenu/schulungsmenu>

2 GRUNDLAGEN - MENSCHEN MIT SEHBEINTRÄCHTIGUNG

In der menschlichen Sinneswahrnehmung werden rund 80 Prozent aller Informationen visuell verarbeitet [1]. Menschen mit einer Sehschädigung sind durch die Dominanz des Visuellen oft mit Barrieren konfrontiert, wobei das Spektrum möglicher Probleme und Hürden sehr vielfältig ist und von der Art und dem Grad der Einschränkung sowie vom Kontext abhängt, wie in Abbildung 1 dargestellt.

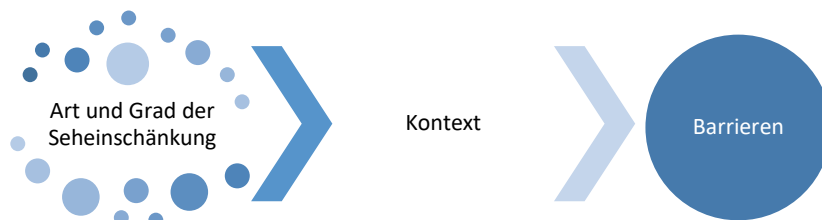


Abbildung 1. Barrieren, Art und Grad der Sehschädigung, und Kontext hängen zusammen.

In diesem Teil des Leitfadens geben wir einen kurzen Überblick über die unterschiedlichen Sehschädigungen. Wir versuchen dabei so allgemein wie nötig und konkret wie möglich zu bleiben, um die Übertragbarkeit zu gewährleisten.

2.1 Wann liegt eine Sehbehinderung oder Blindheit vor?

Im allgemeinen Sprachgebrauch begegnen uns viele Begriffe, welche eine reduzierte Fähigkeit des Sehens beschreiben. Gängige Bezeichnungen sind zum Beispiel Sehbeeinträchtigung, Sehschädigung, hochgradige Sehbehinderung, Sehschwäche und Blindheit.

In Deutschland ist die Bezeichnung und Einordnung im Sozialrecht geregelt. Grundlage für die Klassifizierung, ob jemand sehbehindert ist oder nicht, ist der *Visus*, also die *Sehschärfe*. Konkret gilt in Deutschland folgendes:

Eine Person gilt vor dem Gesetz als sehbehindert, wenn die Sehschärfe kleiner oder gleich 30% ist und dieser Zustand anhaltend, d.h. länger als 6 Monate besteht, sowie nicht korrigierbar ist.

Sehbehinderte Menschen können in Teilen visuelle Informationen wahrnehmen. Was genau visuell wahrgenommen wird variiert jedoch je nach Art und Grad der Einschränkung. In Deutschland gibt es im Wesentlichen drei Einstufungen [4], welche in folgender Tabelle zusammengefasst sind:

Hinweis: Behinderung ist per Definition heute nicht mehr nur als medizinisch dauerhafte Diagnose zu verstehen. Die ICF [6] definiert Behinderung als negative Relation zwischen gesundheitlichen und persönlichen Merkmalen sowie Umweltfaktoren. Obwohl dies schwer zu messen ist, bedeutet es, dass jeder im Laufe des Lebens eine Behinderung erfahren kann – dauerhaft, temporär oder situativ.

Tabelle 1. Klassifizierung Sehbehinderung/Blindheit in Deutschland nach Sehschärfe

Klassifizierung	Sehschärfe (Visus)
wesentlich sehbehindert	< 1/10 (10%)
hochgradig sehbehindert	Sehschärfe auf beiden Augen < 1/20 (5 %)
Blind	Sehschärfe auf dem besseren Auge < 1/50 (2 %)

Grundlagen - Menschen mit Sehbeeinträchtigung - Wann liegt eine Sehbehinderung oder Blindheit vor?

Neben dem Visus können auch andere Merkmale bzw. Kombinationen von Merkmalen herangezogen werden, um eine Sehbehinderung zu bestimmen.

Basierend auf der Deutschen Ophthalmologischen Gesellschaft (Gesellschaft für Augenheilkunde) [8] liegt eine Blindheit beispielsweise auch dann vor, wenn zwar normale Sehschärfe besteht, das Gesichtsfeld jedoch nur noch 5 Grad in beiden Richtungen vom Zentrum ausmacht [7].

Die Fähigkeit ein uneingeschränktes Gesichtsfeld zu haben, ist für viele Lebensbereiche wichtig. Wir nehmen dadurch beispielsweise Geschehnisse wahr, die uns helfen zu erkennen, wenn sich uns jemand von der Seite nähert. *Gesichtsfeldausfälle* werden Skotom genannt und sind neben der verminderten Sehfähigkeit charakteristische Symptome verschiedener Augenerkrankungen (siehe Abschnitt 2.3).

Darüber hinaus gibt es verschiedene Arten von *Farbsehstörungen*, welche jedoch per Gesetz nicht als Sehbehinderung, sondern nur als Beeinträchtigung geführt werden. Zu unterscheiden ist hier, ob man bestimmte Farben nicht sehen oder nicht unterscheiden kann. Farbenblindheit beschreibt

den Zustand, wenn ein Mensch gar keine Farben sehen kann. Am häufigsten ist jedoch die Rot-Grün-Schwäche, bei der man die Farben rot und grün nur schwer voneinander unterscheiden kann [10]. Es gibt auch die Rot-Grün-Blindheit, bei der ein Mensch beide Farben überhaupt nicht voneinander unterscheiden kann. Weitere Farbsehstörungen sind beispielsweise die Blauschwäche- bzw. Blindheit.

Zur Feststellung einer Farbsehstörung werden oft Sehtest-Bilder verwendet, beispielsweise die Serie der Ishihara-Farbtafeln⁴. Zwei Beispiele dieser Serie sind in Abbildung 2 dargestellt. Bei diesen klassischen Sehtestbildern werden unterschiedliche Motive in verschiedenfarbigen Punkten dargestellt. Eine entsprechende Farbsehstörung kann dadurch erkannt werden, wenn die Motive nicht deutlich oder gar nicht erkannt werden.

Gesichtsfeld: Generell bezeichnet das Gesichtsfeld den Bereich, den ein Mensch ohne Kopf- bzw. Augenbewegung sieht. Beide Augen zusammengenommen (binokulares Gesichtsfeld), hat das Gesichtsfeld eine Ausdehnung von etwa 180 Grad in der Horizontalen und 60 Grad im Vertikalen. Das Blickfeld bezeichnet den Bereich des Sehens der durch Bewegung der Augen gekennzeichnet ist. Er ist etwa 40 Grad größer.

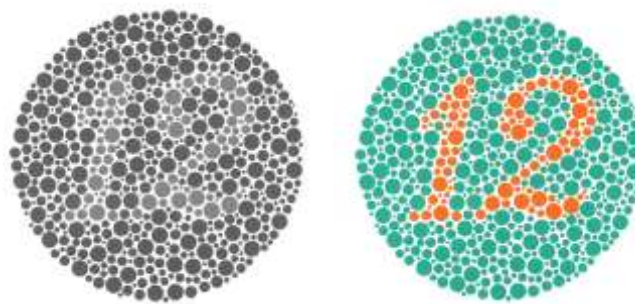


Abbildung 2. Sehtest „Türkis-Orange“ zur Feststellung einer Rot-Grün-Schwäche [11]. Rechts: Bild zeigt ein Zahlenmotiv in der Mitte. Menschen ohne Farbsehschwäche können die Ziffer 12 gut wahrnehmen. Link: Bild zeigt eine mögliche Wahrnehmung des Bildes durch Menschen mit einer Rot-Grün-Schwäche.

⁴ <http://www.spektrum.de/lexikon/psychologie/ishihara-farbtafeln/7481>

2.2 Prävalenz von Sehbehinderung oder Blindheit


Weltweit sind ca. 253 Millionen Menschen von einer Sehbehinderung betroffen, wobei ca. 36 Millionen blind sind und der größte Teil (217 Millionen) wesentlich bzw. starke Sehschädigungen haben [12]. Wichtig ist auch zu sagen, dass die Prävalenz einer Sehbehinderung mit dem Alter zunimmt. Durch die älter werdende Bevölkerung werden somit auch Sehschädigungen bzw. Sehbehinderungen zunehmen. In Deutschland gelten ca. 1 Millionen Menschen als sehbehindert und ca. 160.000 als blind [4].

2.3 Ursachen von Sehbehinderungen oder Blindheit

Ein Überblick über häufige Ursachen für eine Sehbehinderung oder Blindheit ist in folgender Tabelle dargestellt:

Tabelle 1 Zusammenfassung von möglichen Ursachen die zu einer Sehbehinderung oder Blindheit führen können. Informationen basieren auf [2], [13] und [14].

Ursache	Symptome	Mögliche Simulation (mittlere Stadien) Quelle: Pro Retina
(altersbedingte) Makuladegeneration Netzhauterkrankung	<ul style="list-style-type: none"> • Generelle Verminderung der Sehschärfe • Ausfälle im zentralen Gesichtsfeld • Verzerrte Wahrnehmung von Linien • Geringeres Kontrastempfinden und Farbensehen • Erhöhtes Blendempfinden • Häufigste Ursache für Erblindung 	
Diabetische Netzhauterkrankungen (diabetischen Retinopathie)	<ul style="list-style-type: none"> • Unscharfes und verschwommenes Sehen • Wahrnehmung von vielen schwarzen Punkten („Rußregen“) und Lichtblitzen möglich • Erblindung möglich 	
Retinitis Pigmentosa = erbliche bedingte Netzhauterkrankung	<ul style="list-style-type: none"> • Verringerung der Sehschärfe • Nachtblindheit • Blendempfindlichkeit • Verminderung des Kontrastsehvermögens • Verengung des Gesichtsfeldes (Tunnelblick) • Häufigste Ursache für Sehverlust im mittleren Alter 	
Grüner Star (Glaukom) Erkrankung des Sehnervs	<ul style="list-style-type: none"> • Gesichtsfeldausfälle (Skotome) • Erblindung möglich (wenn unbehandelt) 	
Grauer Star (Katarakt) = Erkrankung der Augenlinse	<ul style="list-style-type: none"> • Linsentrübung/Verschleierung der Sicht • Sehen von Doppelbildern • Verstärkte Lichtempfindlichkeit • Vermindertes Kontrastempfinden/ räumliches Sehen • Zunehmende Kurzsichtigkeit • Verminderte Sehschärfe 	

	<ul style="list-style-type: none">• Erblindung möglich	
Netzhautablösung	<ul style="list-style-type: none">• Lichtblitze am Rand des Gesichtsfeldes (anfänglich)• Wahrnehmung von dichten schwarzen Punkten („Rußregen“)• Erblindung möglich	

2.4 Leben mit einer Sehbehinderung oder Blindheit

Blinde Menschen haben keine oder nur eine sehr eingeschränkte visuelle Sinneswahrnehmung und kompensieren daher die visuelle Informationsverarbeitung in erster Linie durch den Gehör- und den Tastsinn. Da die Aufnahmekapazität von akustischen und haptischen Reizen geringer ist als die von visuellen Reizen, benötigen blinde Menschen oft mehr Zeit, um komplexe Informationen zu verarbeiten und eine Situation zu verstehen.

Per Gesetz werden die Art und der Umfang des sogenannten Nachteilsausgleichs definiert, welcher Mehraufwendungen im Alltag, im Beruf oder die soziale Teilhabe regeln soll. Neben Rehabilitationsmaßnahmen wie Orientierungs- und Mobilitätstrainings, Wahrnehmungstraining und Kompensationsstrategien zählen auch die Auswahl sowie Anpassung von Hilfsmitteln – den sogenannten assistiven Technologien – dazu. Ein klassisches Hilfsmittel blinder Menschen ist der Blindenlangstock. Beim Umgang mit Computertechnologien sind Bildschirmlesegeräte oder Braillezeilen nur zwei Beispiele, welche Menschen mit einer Sehbehinderung oder Blindheit die aktive Teilhabe in der digitalen Informationsgesellschaft ermöglichen. Im nächsten Teil des Leitfadens stellen wir weitere Techniken vor, wie Menschen mit einer Sehschädigung digitale Geräte und Informationen nutzen können.


Sinneskompensation: Interessanterweise ist die Informationsmenge, die von den äußeren Bereichen der Netzhaut geliefert wird bei Menschen, die von Geburt oder Kindheit an taub sind größer. Der seitliche Teil der Netzhaut wird dann stärker genutzt und ist leistungsfähiger. Dies ist wahrscheinlich eine Reaktion des Körpers, um mehr von der Seite mitzubekommen.

2.5 Wie Menschen mit Sehbehinderung digitale Geräte und Informationen nutzen

Um die Zusammenarbeit zwischen Sehenden und Blinden zu erleichtern ist es wichtig, die Arbeitsweise von blinden und sehbehinderten Menschen am Computer zu verstehen. Digitale Informationssysteme sind vorwiegend visuell geprägt. Daher sind auch hier Kompensationsstrategien erforderlich, um den Zugang zu digitalen Informationen überhaupt zu ermöglichen, zu verbessern und Barrieren zu vermeiden. Grundsätzlich sind natürlich die Anforderungen jedes Einzelnen individuell, aber es gibt einige generelle Merkmale, welche die Arbeitsweise von Menschen mit und ohne einem Sehrest beschreiben. Die folgenden zwei Datenblätter fassen auf übersichtliche Weise den Stand der Technik zusammen, wie sehbehinderte bzw. blinde Menschen den Computer oder das Handy nutzen.

Tabelle 2 Überblick über Arbeitstechniken von Menschen mit einer Sehbehinderung am Computer.

	<p>STECKBRIEF ARBEITEN AM COMPUTER MIT EINER VERMINDERTEN SEHFÄHIGKEIT.</p>
<p>KURZBESCHREIBUNG</p>	<p>Texte in Schwarzschrift zu lesen stellt kein Problem dar, sofern der Text geeignet dargestellt ist, das heißt, der Text muss hinreichend <i>vergrößert und/oder formatiert</i> werden können sein.</p> <p>Bei dem sogenannten „Tunnelblick“, hervorgerufen durch periphere Gesichtsfeldausfälle, bevorzugen Menschen eher <i>kleineren Text</i> zur besseren <i>Orientierung</i>.</p> <p>Generell ist bei Gesichtsfeldausfällen bzw. starken Vergrößerungen der <i>fehlende Überblick</i> über die Gesamtdarstellung ein Problem. Unterstützungsfunktionen aber auch die Reduktion sowie Vereinfachung von Informationen sind hier adäquate Anpassungen.</p> <p>Menschen mit einer gestörten Farbwahrnehmung benötigen zudem Anpassung des <i>Kontrastes</i> oder auch eine alternative Repräsentation (e.g. <i>Verbalisierungen</i>) von Informationen, welche ausschließlich durch Farbe dargestellt sind.</p> <p><i>Blendfreie Bildschirme und Umgebungen</i> fördern ebenso das Arbeiten mit dem Computer sowie auch eine gute Akustik um visuelle Einschränkungen besser durch andere Sinne kompensieren zu können.</p>
<p>HILFSMITTEL & ANPASSUNGEN</p>	<p><i>Vergrößerungssoftware</i> in verschiedenen Modi (Linse, Vollbild, oder angedockt).</p> <p><i>Sprachausgabe</i> je nach Bedarf. Die Kombination von Vergrößerung und Sprachausgabe ist geläufig.</p> <p><i>Kontrastreiches Farbschema</i>, meist heller Text auf dunklem Hintergrund.</p> <p>Individuelle <i>Textformatierungen</i>, z.B.: Textgröße, Schriftart, Zeilenabstand.</p>

	<h2>STECKBRIEF ARBEITEN AM COMPUTER IM FALL VON BLINDHEIT</h2>
<p>KURZBESCHREIBUNG</p>	<p>Zugang zu grafischen Benutzeroberflächen wird blinden Menschen durch <i>auditive oder taktile Informationsdarstellungen</i> ermöglicht.</p> <p>Blinde Menschen arbeiten am Computer ohne Maus. Stattdessen erfolgt die Eingabe am PC über die <i>Tastatur</i>. Am wichtigsten sind hier die Pfeiltasten, der Tabulator und anwendungsspezifische Tastenkombinationen. Mit der Leertaste werden Links, Knöpfe (Buttons) oder Eingabetasten ausgelöst.</p> <p>Auf dem Smartphone gibt es die sogenannte „<i>Explore by touch</i>“ Funktion. Wischt man mit einem Finger über das Display wird das nächste Element fokussiert und angesagt. Ein Doppeltap auf den Bildschirm aktiviert dann das Element.</p> <p>Wichtig ist auch hier das Stichwort <i>Überblick</i> zu nennen. Während sich sehende Menschen rasch einen visuellen Überblick über Situationen verschaffen können, müssen blinde Menschen sich diesen sequentiell erarbeiten. Auf Webseiten kann dies beispielsweise dadurch gelingen, dass man sich die Überschriften, Listen oder Links gezielt ansagen lässt und somit ein <i>mentales Modell</i> der Seite aufbaut.</p> <p>Darüber hinaus müssen für Grafiken <i>alternative Beschreibungen</i> vorliegen damit auch blinde Menschen einen Zugang zu Bildern, Illustrationen oder Diagrammen erhalten.</p> <p>Zeitabhängige Medien wie Videos werden für blinde Menschen durch eine sogenannte <i>Audiodeskription</i> zugänglich. Das heißt, alles was im Video nicht verbal dargestellt wird, wird in einer separaten Audiospur vermittelt.</p>
<p>HILFSMITTEL & ANPASSUNGEN</p>	<p>Blinde Menschen arbeiten am Computer mit einem <i>Screenreader</i>. Dies ist ein Programm, das alle zugänglichen Informationen (Text, Menüs, Listen, etc.) in einen logischen Objektbaum übersetzt und die Informationen durch <i>Sprachausgabe und Braille</i> ausgibt.</p> <p>Üblicherweise ist mit dem Screenreader auch eine Braillezeile gekoppelt. Dadurch wird das, was vorgelesen wird, auch in Punktschrift ausgegeben. Je nach Anwendungskontext wird die verbale oder taktile Informationsdarstellung verwendet. Braille ist zum Beispiel besonders geeignet um <i>Einrückungen oder Formatierungen</i> zu erfassen.</p> <p>Bei der Benutzung eines Screenreaders kann man eine Vielzahl von individuellen Anpassungen vornehmen. Am meisten verbreitet sind die <i>Sprechgeschwindigkeit</i>, die <i>Betonung von Textauszeichnungen</i> (z.B.: Großschreibung) durch individuelle Tonhöhen oder das Ändern der <i>Sprache</i> des Screenreaders.</p>


2.6 Zusammenarbeit von Menschen mit und ohne Sehschädigung

Die Zusammenarbeit zwischen Menschen mit und ohne Sehschädigung erfordert es, dass man in unterschiedlichen Situationen die verschiedenen Arbeitstechniken sowie die Wahrnehmung von Informationen berücksichtigt.

Ein wichtiger Punkt ist die alltägliche Kommunikation. Sehende Menschen verlassen sich auf optische Eindrücke und können leicht darauf reagieren, wenn jemand Kontakt aufnehmen will. Sie erkennen auf einen Blick, ob eine Person beschäftigt ist oder ob sie in einer Gesprächsrunde angesprochen werden. Blinde Menschen können solche nichtverbalen Signale wie die Blickrichtung, das Kopfnicken oder Zeigegesten nicht erfassen. Diese Situation hat im Kontext der Zusammenarbeit Konsequenzen auf die Gesprächskultur. In Teammeetings kann man beispielweise dadurch unterstützen, dass elementare, *nichtverbale Informationen verbalisiert werden*. Dazu zählt auch, dass man eine Person, die man anspricht, beim Namen nennt. Zudem ist es hilfreich, kurz zu erwähnen, wenn eine Person den Raum verlässt. Es ist für einen blinden Menschen unangenehm, eine Person anzusprechen, die bereits den Raum verlassen hat. Außerdem sollte man unerwartete Ereignisse erklären und die Ursache von ungewohnten akustischen Signalen beschreiben, da blinde Menschen besonderes auf akustische Signale wie Schritte, Rascheln, Geräusche, Lärm und Rauschen achten, um sich im Raum zu orientieren und eine Situation einzuschätzen.

Neben kommunikativen Aspekten ist es wichtig, den gleichberechtigten Zugang zu jeglichen visuellen Informationen sicherzustellen. Das bedeutet, dass alle visuellen Materialien auch in adäquater Form für blinde Menschen aufbereitet und zur Verfügung gestellt werden. Sie sollten also Ausdrücke auf Papier, welche sie in einer Teamsitzung verwenden, den blinden Teammitgliedern vorher in digitaler Form zur Verfügung stellen. Spezielle visuelle Informationen sollten zudem in einer *haptischen, textuellen oder auditiven Repräsentation* vorbereitet werden. Folgende Tabellen fassen kurz zusammen, was man bei Meetings, in denen Menschen mit und ohne Sehschädigung sich treffen, beachten sollte.


Tabelle 4. Hinweise, die man in Meetings oder während der Kommunikation mit blinden Menschen beachten sollte.

	STECKBRIEF MEETINGS UND KOMMUNIKATION MIT BLINDEN MENSCHEN
VORBEREITUNG	Bereiten Sie die benötigten Informationen in textueller oder taktiler Darstellung auf. Lassen Sie die Unterlagen der blinden Person rechtzeitig vor dem Meeting zukommen (digital oder analog im Falle von taktilen Ausdrucken).
WÄHREND DES MEETINGS	Geben Sie einen Überblick über die Teilnehmer (Vorstellungsrunde). Dies ist besonders wichtig, wenn es keine festen Plätze gibt oder unbekannte Personen im Raum sind. Dadurch können blinde Teammitglieder andere Personen durch Stimme und Sprechrichtung identifizieren.

Grundlagen - Menschen mit Sehbeeinträchtigung - Zusammenarbeit von Menschen mit und ohne Sehschädigung

	<p>Erläutern sie in einer fremden Umgebung auch die Räumlichkeiten, zum Beispiel, ob das WC gut erreichbar ist, ob die Türen taktil markiert sind oder ob es eventuell Leitlinien gibt.</p> <p>Erklären sie alle nichtverbalen Interaktionen und visuellen Medien. Vermeiden sie visuell-orientierte Angaben wie „hier“ und „dort“. Sagen sie statt dessen, was hier und dort bedeutet, zum Beispiel: „ Auf der Folie ist ein Graph dargestellt, der ... veranschaulicht.“</p> <p>Informieren Sie blinde Teammitglieder, wenn jemand den Raum verlässt oder ihn betritt. Erklären Sie unerwartete Geräusche oder Pausen.</p> <p>Achten sie im direkten Gespräch mit einer blinden Person auch darauf, dass ihre Sprechrichtung der blinden Person zugewandt ist. Signalisieren Sie, wenn Sie sich von der blinden Person abwenden.</p>
E-MAIL	<p>Füllen Sie bei der Kommunikation über E-Mail in jedem Fall die Betreffzeile mit einem sinnvollen Betreff aus.</p> <p>Fügen Sie bei einer Antwortmail Ihren Antworttext immer am Anfang ein. Vermeiden Sie Verschachtelungen durch Zitate aus alten Mails.</p> <p>Achten Sie darauf, nur inhaltlich zusammengehörende Themen in eine Mail zu verpacken und die Inhalte auch kurz und strukturiert zu formulieren.</p> <p>Fügen Sie Links in eine neue Zeile ein und achten Sie auch darauf, dass diese interaktiv als Link formatiert sind.</p> <p>Vermeiden Sie ein „PS“ nach der Grußformel, da dieser leicht überlesen werden kann.</p>

Tabelle 5. Hinweise, die man in Meetings oder während der Kommunikation mit sehbehinderten Menschen beachten sollte.


	<p>STECKBRIEF MEETINGS UND KOMMUNIKATION MIT SEHBEHINDERTEN MENSCHEN</p>
VORBEREITUNG	<p>Bereiten Sie die benötigten Informationen in digitaler Form oder in Großdruck vor. Lassen sie die Unterlagen der sehbehinderten Person vor dem Meeting zukommen (digital oder analog im Falle vom Großdruck)</p>
WÄHREND DES MEETINGS	<p>Verwenden Sie einen hohen Kontrast (Hintergrund-/Schriftfarbe) wenn sie Folienpräsentationen in Meetings nutzen. Lesen Sie alle Informationen vor und springen sie nicht zwischen den Folien hin und her.</p>

	<p>Dunkeln Sie wenn möglich den Raum ab und ermöglichen sie den sehbehinderten Personen einen Sitzplatz mit dem Rücken zum Fenster, um Blendungen zu vermeiden.</p> <p>Türgriffe von Räumen können zudem mit einem auffälligen Klebeband markiert werden, um deren Sichtbarkeit zu erhöhen. Ähnlich verhält es sich mit gefährlichen Stufen.</p> <p>Bei der direkten Kommunikation sollte man beachten, dass nicht-verbale Signale wie zum Beispiel ein Lächeln, Gesten oder ein Blickkontakt oft nicht erkannt werden.</p>
--	---

Neben diesen grundlegenden Aspekten zur Kommunikation gibt es natürlich weitere Situationen, die besondere Beachtung bei der Zusammenarbeit zwischen sehenden und sehingeschränkten Menschen gibt. Beispielweise kann es Situationen geben, in denen blinde Kollegen oder Kolleginnen kurz Hilfe benötigen. Vielleicht steht im Moment keine Arbeitsassistenz zur Verfügung und ihr Arbeitskollege bzw. ihre Arbeitskollegin kommt bei der Arbeit nicht weiter, weil Barrieren in einer Website oder einem Softwareprogramm vorhanden sind.

Da die Zeit meist knapp ist, stellt sich häufig die Frage, wie kann ich schnell und effektiv weiterhelfen? Oft hilft eine kurze Beschreibung. Dies ist jedoch nur möglich, wenn die blinde Person genau sagen kann, was sie von ihnen möchte und wenn sie wissen, wie sie helfen können. Die Art der Hilfe ist natürlich stark von der aktuellen Aufgabe abhängig und kann daher nur bedingt generalisiert werden. Exemplarisch soll jedoch eine mögliche Herangehensweise aufgezeigt werden, wie sie einer blinden Person helfen können, die Unterstützung benötigt. Getrauen sie sich auch der blinden Person zu sagen, wenn der Zeitpunkt für sie nicht passt, sie selbst in Zeitdruck sind oder die Hilfestellung zu lange dauern würde. Die folgende Tabelle zeigt eine mögliche Abfolge bei der gemeinsamen Problemfindung und Lösung auf einer Webseite.

Tabelle 6. Mögliche Abfolge bei der Problemfindung und Lösung auf einer Webseite.

	<p>STECKBRIEF BLINDEN MENSCHEN BEI BARRIEREN AUF EINER WEBSEITE HELFEN</p>
SCHRITT 1	Klären sie, ob die Grundstruktur der Website bekannt ist. Wenn dies nicht der Fall ist, geben sie eine kurze Übersicht über die Struktur der Website. Dabei sind Navigationsleisten, Formulare, Anzahl der Eingabefelder, Knöpfe und Grafiken ohne Beschriftung wichtig.
SCHRITT 2	Wenn sie selbst die Webseite nicht kennen, versuchen sie sich kurz einen Überblick zu verschaffen.
SCHRITT 3	Klären sie, welches Problem aufgetreten ist. Lassen sie sich eine kurze Zusammenfassung geben, was ihr Kollege oder ihre Kollegin macht und welcher Schritt nicht erfolgreich war.

Grundlagen - Menschen mit Sehbeeinträchtigung - **Zusammenarbeit von Menschen** mit und ohne Sehschädigung

SCHRITT 4	<p>Finden sie heraus, wo sich ihr Kollege oder ihre Kollegin mit dem Screenreader befindet, genauer gesagt, wo sich der virtuelle Cursor (Fokus) auf dem Bildschirm befindet. Der virtuelle Cursor wird nur in aktiven Eingabefeldern oder bei markierten Links sichtbar.</p> <p>Wenn sie nicht sehen, wo sich der Fokus befindet, suchen sie gemeinsam nach einem eindeutigen Begriff, an dem sie sich orientieren können. Mit der Suchfunktion kann eine blinde Person nach diesem Begriff suchen.</p> <p>Es gibt die Möglichkeit mit der F5 Taste den Bildschirminhalt zu aktualisieren. Danach wird manchmal der Ausschnitt im Bildschirm angezeigt, wo sich der virtuelle Cursor befindet.</p>
SCHRITT 5	<p>Beschreiben sie kurz die Möglichkeiten, die es gibt. Dazu zählt zu erwähnen, welche Schritte als nächstes möglich sind oder ob es sich um eine nicht beschriftete Grafik handelt. Eventuell kann man beschreiben, welche Aktionen durch das Anklicken der Grafik ausgelöst werden. Versuchen sie, die Möglichkeiten zusammenzufassen.</p> <p>Wenn ein Element nicht über die Tastatur zugänglich ist, fragen sie, ob sie den Schritt mit der Maus durchführen sollen. Nutzen sie die Maus aber nur nach Absprache, da sich sonst der Fokus in einem ganz anderen Bereich befindet und sich die blinde Person danach wieder ganz neu orientieren muss.</p> <p>Ziel einer Hilfestellung sollte immer sein, dass die blinde Person in Zukunft einen Weg findet, diese Schritte selbst auszuführen. Es ist nicht sinnvoll, wichtige Arbeitsschritte aus Zeitgründen einfach rasch selbst zu erledigen, es sei denn, es gibt keine andere Möglichkeit.</p>

3 IDEs - BARRIEREFREIE BEDIENUNG VON ECLIPSE

In diesem Kapitel möchten wir beispielhaft eine integrierte Entwicklungsumgebung vorstellen, um die verschiedenen Aspekte der Barrierefreiheit zu veranschaulichen. Dabei liegt der Fokus auf der Arbeitsweise von Menschen mit Blindheit, die eine integrierte Softwareentwicklungsumgebung nutzen und darauf, welche Barrieren in diesem Zusammenhang oft auftreten. Eclipse haben wir ausgewählt, weil es eine weitverbreitete Entwicklungsumgebung ist, die als Open-Source Projekt über eine aktive Entwickler-Community verfügt.

Prinzipiell ist das Gebiet der Programmierung für Menschen, die auf einen Screenreader angewiesen sind, aufgrund des textuellen Charakters von Quellcode zugänglich. Einfache Softwareprogramme oder Webseiten können bereits mit einem herkömmlichen Texteditor entwickelt werden. Für komplexe Softwareprojekte, in denen mehrerer Entwickelnde programmieren, werden heute jedoch meist integrierte Entwicklungsumgebungen (Integrated Development Environments, kurz: IDE) eingesetzt. IDEs vereinen wichtige Bausteine der Softwareentwicklung wie Editoren, Compiler, Interpreter, Debugger und Versionskontrolle. Dadurch entsteht wiederum eine Funktionskomplexität, welche in erster Linie durch grafische Benutzungsoberflächen, verschachtelte Menüs, verschiedene visuelle Bereiche und Ansichten begegnet wird. Dies bringt Vereinfachungen für sehenden Menschen, schafft aber gleichzeitig Barrieren für seheingeschränkte Menschen. Obwohl Eclipse eine gute Zugänglichkeit aufweist, gibt es verschiedene Barrieren. Am häufigsten ist die fehlende Barrierefreiheit von Softwareerweiterung (Plugins), welche durch Dritte angeboten werden, die fehlende Rückmeldung von Echtzeitinformationen wie Fehlermeldungen sowie das Debugging [16]. Im Folgenden beschreiben wir wichtige Funktionen und gängige Programmierpraxen blinder Softwareentwickelnder, die Eclipse nutzen. Wir beschreiben Workarounds die angewendet werden, wenn eine Barriere vorliegt. Außerdem werden die wichtigsten Tastaturkürzel tabellarisch zusammengefasst, um Eclipse mit der Tastatur zu bedienen.

Die Inhalte dieses Teils des Leitfadens wurden in verschiedenen Arbeiten im Cooperate Projekt, in Diskussionen mit Endnutzern und Befragungen erstellt. Primär liegt diesem Kapitel eine von uns durchgeführte Umfrage zur Barrierefreiheit von Eclipse zugrunde, welche auf der Konferenz AAATE [16] veröffentlicht wurde. Darüber hinaus verweisen wir auf die Cooperate Eclipse Schulung [15], welche im Rahmen des Cooperate Projektes entwickelt wurde. Die Schulung richtet sich an Menschen mit einer Sehschädigung und beschreibt wichtige Einstellungen zur Bedienung von Eclipse mit dem Screenreader und der Braillezeile sowie die wesentlichen Shortcuts um Eclipse mit der Tastatur zu bedienen.

3.1 Navigation im Quellcode-Editor

Sehende Menschen können Strukturen in einem Quellcode leicht visuell erkennen und darin mit der Maus navigieren. Die primäre Arbeitsweise von blinden Softwareentwickelnden besteht jedoch darin, Quellcode zeilenweise im Editor zu lesen. Das zeilenweise Lesen ist jedoch nicht geeignet, um sich schnell einen Überblick zu verschaffen. Wichtig sind daher für Menschen mit Blindheit Funktionen, die das Lesen von umfangreichem Code erleichtern. Dazu gehören Bereiche wie ein Outline, Tastaturbefehle (Shortcuts), Autovervollständigung oder auch Code-Folding.

Eclipse bietet die sogenannte **Outline-View**, um sich im Quellcode zu orientieren und zu navigieren. Es ist eine Art Strukturbaum aller Definitionen und Funktionen in einer Quellcodedatei. Das Outline ist in Eclipse bequem über die Tastatur zu erreichen und zwar mit dem Befehl **Strg + O**. Der Screenreader liest die einzelnen Einträge vor. Nutzende können mit den Pfeiltasten zwischen den Elementen navigieren und mit Enter kann dann zu einem gewünschten Element im Editor gesprungen werden. Diese Funktion trägt entscheidend zu einer besseren Benutzbarkeit von Eclipse bei.

Im Quellcode vieler Programmiersprachen werden Objekte meist mit Klammern umschlossen. Für Sehende sind diese häufig farbig hervorgehoben und helfen, sich einfacher im Quellcode zurecht zu

finden. Eine wichtige Funktion für blinde Menschen ist, mit dem *Shortcut Strg+Shift+P* zwischen öffnenden und schließenden Klammern hin und her zu springen. Auch die *Autovervollständigung* ist über einen Shortcut erreichbar (*Strg+Space+F2*) und zählt zur den elementaren Elementen, um die Barrierefreiheit zu verbessern.

Darüber hinaus hilft das Feature *Code-Folding*, sich in umfangreichem Quellcode zurechtzufinden. Damit ist gemeint, dass man zusammengehörige Bereiche des Quellcodes ein- und ausklappen kann. Über den Shortcut *Strg + Numpad_Divide* kann Code-Folding für alle Methoden aktiviert bzw. deaktiviert werden. Mit *Strg + Numpad_Multiply* können alle Methoden zugeklappt werden und mit *Shift + Strg + Numpad_Divide* können diese wieder ausgeklappt werden.

Wenn sich der Cursor in der Zeile der Methode befindet kann man standardmäßig mit *Strg + Numpad Pluszeichen* die Methode ausklappen und mit *Strg + Numpad Minuszeichen* für zuklappen.

Code-Folding wird jedoch nicht per se von blinden Softwareentwickelnden als nützlich empfunden. Einige empfinden es als schwierig, versteckte Bereiche zu identifizieren, da diese Information leicht übersehen werden kann. Code-Folding kann über den Präferenz Dialog in Eclipse (*window/preferences/java/editor/folding*) ausgeschaltet werden. Generell ist zu empfehlen, neue Funktionen zur Barrierefreiheit konfigurierbar zu machen, um individuelle Bedürfnisse besser adressieren zu können.

3.2 Erkennen von Programmierfehlern und Debuggen

Das Erkennen von Programmcodefehlern, Warnungen und Debuggen ist eine elementare Funktion in der Programmierung. Stand der Technik gängiger IDEs ist es, dem Benutzer bereits beim Schreiben von Quellcode eine Rückmeldung zu geben, an welcher Stelle ein Fehler ist. Dies erfolgt jedoch meist durch visuelles Feedback wie in Abbildung 3 dargestellt. Ein Symbol vor der Zeilennummer markiert, dass in der Zeile 15 ein Fehler ist. Hinweise zu Fehlern sowie Quick fixes werden darüber hinaus angeboten, wenn der Nutzer die Maus über das Fehlericon führt, was wiederum nicht barrierefrei ist.

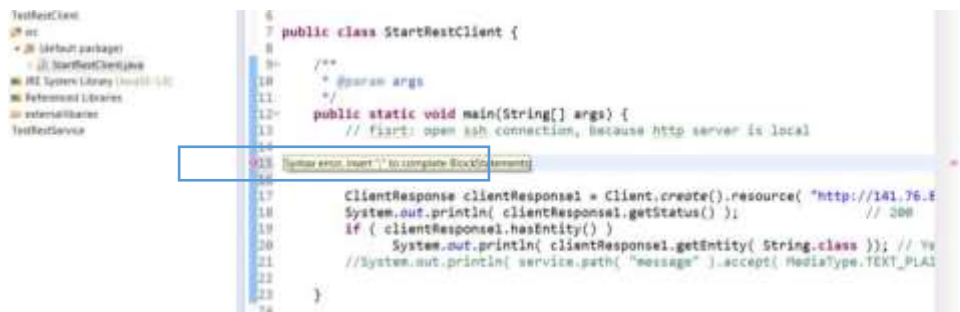


Abbildung 3. Beispiel einer visuellen Darstellung von Fehlern in Eclipse **Fehler! Verweisquelle konnte nicht gefunden werden..**

Screenreader Nutzer sind hier rein auf die Tastaturbedienung angewiesen und können solche visuellen Informationen nicht wahrnehmen. Folgende alternativen Arbeitstechniken sind uns für die Fehlerrückmeldung und das Debugging in Eclipse bekannt. Diese Arbeitstechniken sollten auch von zukünftig zu entwickelnder Software zur Verfügung gestellt werden:

STECKBRIEF FEHLERRÜCKMELDUNG UND DEBUGGING MIT DEM SCREENREADER IN ECLIPSE	
ECHTZEITFEHLER	In Eclipse kann man über die Tastaturkürzel <i>Strg + Punkt</i> bzw. <i>Strg + Komma</i> zwischen Fehlern vor bzw. zurück navigieren. Über <i>Strg + 1</i> erhält man Zugang zu den Quickfixes. Zusätzlich können über die Funktionstaste <i>F2</i> Tooltips aufgerufen werden, welche ergänzende Informationen zum Fehler liefern.

	<p>Alternativ kann auch über die Problems View durch die einzelnen Fehler navigiert werden. Über <i>Strg + Shift + F7</i> kann man zwischen Bereichen wechseln und so in die Problems View gelangen.</p>
DEBUGGING	<p>In Eclipse kann der Debugger über den Shortcut <i>F11</i> aktiviert werden. Innerhalb des Debuggers kann man dann mit <i>F5</i> in eine Funktion hineingehen. Mit <i>F6</i> kann man zur nächsten Zeile navigieren und mit <i>F7</i> kann man wieder aus einer Funktion herausgehen. Mit <i>F8</i> kann man zum nächsten Haltepunkt (Breakpoint) springen.</p> <p>Zwischen Breakpoints kann mit <i>Strg + Shift + B</i> hin und her geschaltet werden. Standardmäßig haben nicht gefangene Exceptions in Eclipse Breakpoints.</p> <p>Mit <i>Strg + Alt + B</i> werden alle Breakpoints überspringen und <i>F2</i> beendet das Debugging.</p> <p>Eine hilfreichere Vorgehensweise neben Breakpoints ist es, an eine beliebige Zeile im Quellcode zu gehen und dann das Tastenkürzel <i>Strg + R</i> anzuwenden. Dadurch terminiert die Anwendung an der entsprechenden Stelle.</p>
WORKAROUNDS	<p>Debugging ist für Nutzende von Screenreadern durch die sequentielle Informationswahrnehmung nicht trivial und erfordert häufig Workarounds. Um beispielsweise mitzubekommen, dass die Ausführung unterbrochen wurde, muss die Debug-View durchsucht werden.</p> <p>Alternativ werden häufig Statements wie <code>System.out.println()</code> verwendet bzw. Logger oder Watchpoints definiert, um Variablen zu beobachten. Das Logging wird generell von blinden Menschen als effektiv bewertet.</p>

Die Tastaturbedienung, mit der Programmierfehler erkannt werden können, macht den nicht-visuellen Zugang prinzipiell möglich. Diese Variante ist aber nicht unbedingt effizient. In unseren Befragungen und Diskussionen mit Anwendenden wurden folgende Verbesserungswünsche am häufigsten benannt:

- Das Vorlesen von Fehlern sollte direkt beim Schreiben erfolgen. Eine mögliche Benennung dieses Features ist *announce errors*.
- Die spezielle *Markierung von Fehlern auf der Braillezeile* wurde als wertvolle Erweiterung identifiziert.
- Ebenso wurde die Sonifikation von Fehlertypen (z.B. durch Earcons) als mögliche Erweiterung und Verbesserung der nicht-visuellen Arbeitsweise identifiziert. Beispielsweise könnte ein konfigurierbarer Ton auf Fehler aufmerksam machen, wenn man mit den Pfeiltasten durch die Codezeilen navigiert.

3.3 Überblick über die wichtigsten Tastaturkürzel

Wie bereits erwähnt ist Eclipse dank der Tastaturbedienung gut bedienbar. Die folgenden Tabellen, welche auch Bestandteil der Eclipse Schulung sind [15] **Fehler! Verweisquelle konnte nicht gefunden werden.** geben einen Überblick über wichtige Tastaturbefehle von Eclipse.

3.3.1 Eclipse Shortcuts zur allgemeinen Navigation in der Anwendung

Liste aller aktiven Shortcuts in Eclipse anzeigen	Strg + Shift + L
Wechsel zwischen geöffneten Tabs/Dateien im Editor	Strg + F6
Offene Sichten (Views) vorwärts durchlaufen	Strg + F7
Offene Sichten (Views) rückwärts durchlaufen	Strg + Shift + F7
Offene Perspektiven wechseln	Strg + F8
View öffnen (auch nicht sichtbare)	Alt + Shift + Q
Geöffneter Dialoge schließen	Escape
Fokus auf das Menü setzen	F10
Kontextmenü zeigen (rechter Mausklick)	Shift + F10

3.3.2 Eclipse Shortcuts zur Dateiverwaltung

Dialog öffnen, um neue Dateien zu erstellen	Strg + N
Kontextmenü öffnen, um eine neue Datei zu erstellen	Alt+ Shift + N
Aktuelle Datei speichern	Strg + S
Alle Dateien speichern	Strg + Shift + S
Aktuell aktive Datei schließen	Strg + W
Alle geöffneten Dateien im Editor schließen	Strg + Shift + W
Dateieigenschaften öffnen	Alt + Enter

3.3.3 Eclipse Shortcuts zur Navigation im Editor

Fokus auf den Editor setzen	F12
Liste aller geöffneten Dateien im Editor anzeigen, Navigation mit Pfeiltasten, Datei auswählen mit Enter	Strg + E
Editor maximieren (oder andere aktive View), erneute Betätigung minimiert das Fenster wieder	Strg + M
Editor-Fenster-Options-Menü öffnen	Alt + Minus
Anzeigemenü von verfügbaren Features für den linken, vertikalen Bereich; Breakpoints, Bookmarks, Zeilennummern, etc.) zeigen	Strg + F10
Zur nächsten Methode springen	Strg + Shift + Pfeil hoch/runter
Zur eingegebenen Zeilennummer im sich öffnenden Fenster springen. Um Zeilennummern ein-/auszublenden, Strg + F10 und „Show Line Numbers“ wählen	Strg + L
Zur zuletzt editierten Position springen	Strg + Q
Zur öffnenden/schließenden Klammer einer selektierten Klammer springen (Klammer ist markiert oder Cursor ist direkt davor oder dahinter positioniert)	Strg + Shift + P
Aktuelle Methode expandieren oder minimieren	Strg + NumPad Plus/Minus

3.3.4 Eclipse Shortcuts zur Programmierung

Code Outline anzeigen	Strg + O
Informationen zu Klassen, Methoden oder Variablen als ToOLTIPtext öffnen	F2
Deklaration öffnen: Springt zur Deklaration der selektierten Klasse, Methode oder Parameter	F3
Typhierarchie Fenster des selektierten Items öffnen	F4
Fehler vorwärts durchlaufen	Strg + Punkt
Fehler rückwärts durchlaufen	Strg + Komma
Quick-Fix: Fehlerbehebung aufrufen, Pfeiltasten für Navigation	Strg + 1
Nach Typ suchen (Klasse, Interface, ...)	Strg + Shift + T
Umbenennung des selektierten Elements und aller Referenzen	Alt + Shift + R
Applikation speichern und ausführen (Run)	Strg + F11

IDEs - Barrierefreie Bedienung von Eclipse - Überblick über die wichtigsten Tastaturkürzel

Debug	F11
Im Debugger: In Funktion reingehen	F5
Im Debugger: Nächster Schritt (Zeile für Zeile)	F6
Im Debugger: Aus Funktion rausgehen	F7
Im Debugger: Zum nächsten Haltepunkt springen	F8
Den Inhaltsassistenten öffnen, z.B. zeigt verfügbare Methoden oder Feldnamen an	Strg + Leertaste

3.3.5 Eclipse Shortcuts zur Versionsverwaltung mit SVN

Mit Repository synchronisieren	Strg + Alt + S
Eigenschaften anzeigen	Strg + Alt + T
Update	Strg + Alt + U
Update zu Revision	Strg + Alt + D
Commit	Strg + Alt + C
Merge	Strg + Alt + E
Zu svn:ignore hinzufügen	Strg + Alt + I

4 FALLSTUDIE - GEMEINSAM SOFTWARE ENTWICKELN MIT UML

Dieser Leitfaden behandelt bisher die Grundlagen zum Thema Sehschädigung, stellte allgemeine Arbeitstechniken von Menschen mit Sehschädigung am Computer vor, diskutierte die Zusammenarbeit zwischen Menschen mit und ohne Sehschädigung (Kapitel 2) und stellte speziell die Barrierefreiheit der Softwareentwicklungsumgebung Eclipse vor (Kapitel 3). In diesem Kapitel werden die konkreten Lösungen beschrieben, welche im Cooperate-Projekt entwickelt wurden. Die entwickelte Lösung ist eine Erweiterung für Eclipse und adressiert Probleme, welche speziell bei der gemeinsamen Modellierung mit grafischen Beschreibungssprache zwischen Entwicklern mit und ohne Sehschädigung auftreten können. Die vorgestellten Lösungen sollen die Zusammenarbeit verbessern und erleichtern.

Als Beispiel haben wir die gemeinsame Arbeit an Softwareprojekten mit der Unified Modeling Language™ (UML™) ausgewählt. UML ist eine standardisierte grafisch-repräsentierte Sprache zur Spezifikation, Visualisierung, Konstruktion und Dokumentation von Modellen beispielsweise für Softwaresysteme. UML Diagramme werden häufig dafür eingesetzt, um über verschiedene Aspekte einer Software zu diskutieren, Anforderungen aufzunehmen oder Versionen zu dokumentieren. Die intuitive grafische Darstellung von UML erleichtert die Kommunikation zwischen den verschiedenen Akteuren in der Softwareentwicklung, weil informatisches Hintergrundwissen nicht unbedingt notwendig ist, um die dargestellten Prozesse zu verstehen. Gleichmaßen erlaubt UML aber auch programmtechnische Details abzubilden oder Programmcode aus UML Modellen heraus zu generieren. Damit ist die UML prinzipiell eine gute Kommunikationsschnittstelle zwischen Menschen mit und ohne technischem Hintergrund. Allerdings erzeugt die UML auch Barrieren, da Diagramme in erster Linie grafisch dargestellt werden und somit für Menschen mit Sehbehinderung unter Umständen nicht zugänglich sind. Für eine reibungslose Zusammenarbeit in Entwicklungsteams, die aus Menschen mit und ohne Sehschädigung bestehen, sind daher Lösungen und barrierefreie Werkzeuge notwendig, welche verschiedene Sichtweisen auf ein UML Modell erlauben und somit die Zusammenarbeit in sogenannten Diversity-Teams ermöglichen.

Eine Einführung in die grafische Beschreibungssprache UML wurde im Projekt Cooperate entwickelt und kann über die Projektwebseite heruntergeladen werden [22]. Im Folgenden gehen wir speziell auf den Einsatz von UML in der Softwareentwicklung ein.

4.1 Erfahrungsbericht eines blinden Softwareentwicklers zum Einsatz von UML

Im Rahmen des Cooperate Projekts haben wir uns zunächst damit beschäftigt, wann und in welcher Weise UML von Menschen mit und ohne Sehschädigung bei der gemeinsamen Softwareentwicklung eingesetzt wird. Die im Folgenden zusammengefassten Ergebnisse von Befragungen beziehen sich auf einen blinden Softwareentwickler eines mittelständischen Unternehmens. In anderen Unternehmen oder Bereichen kann es entsprechend anders aussehen.

4.1.1 Wie wird UML eingesetzt und welche Diagramme werden verwendet?

Wie und welche UML Diagramme eingesetzt werden, ist zuweilen abhängig von der Rolle eines Entwicklers im Team. Während Entwickler UML Diagramme zur Veranschaulichung und zur Diskussion von komplexen Sachverhalten im Team nutzen, verwenden andere Beteiligte UML Diagramme zur Kommunikation mit Auftraggebern.

Generell kann man aber sagen, dass UML Diagramme stärker in der Anforderungserhebung als in der Entwurfsphase eingesetzt werden. Bei der Anforderungserhebung dienen meist Anwendungsfalldiagramme oder Aktivitätsdiagramme in graphischer Form zur Abstimmung mit dem Kunden. Paket- und Kompositionsdiagramme werden darüber hinaus meist dafür verwendet, einen Überblick über die Struktur des Gesamtsystems zu vermitteln. Sequenz- und Zustandsdiagramme werden weniger zur Diskussion, dafür mehr für die Dokumentation genutzt.

Beim Feinentwurf von Software werden UML Diagramme eher selten genutzt, weil beispielsweise Klassendiagramme schon recht nah am Quelltext sind und sich gezeigt hat, dass das Verhältnis Aufwand zu Nutzen bei feingranularen Modellierungen nicht ausgewogen ist. Konkrete Realisierungsideen werden dann oft verbal diskutiert oder verschriftlicht. Insbesondere in der agilen Entwicklung wird eine intensive Entwurfsphase eher vermieden.

4.1.2 Was passiert mit den UML Diagrammen?

Häufig werden UML Diagramme in der Anforderungsphase skizzenhaft mit Hilfe von grafischen Editoren erstellt, um eine rasche Diskussionsgrundlage zu schaffen. Dies erfolgt oft auch losgelöst von der im Unternehmen verwendeten Entwicklungsumgebung, da es unter Umständen schneller geht, ein Diagramm beispielsweise in PowerPoint zu skizzieren. Welches Tool verwendet wird, ist jedoch sehr unterschiedlich und auch davon abhängig, welche Rolle ein Entwickler in einem Projekt inne hat.

Werden UML Diagramme für die Kommunikation und Diskussion verwendet werden, dann werden diese nicht immer überarbeitet, falls sich etwas im Entwurfs- bzw. Entwicklungsprozess ändert. In der Qualitätssicherung kann es sich jedoch anders verhalten, weil Diagramme zum Beispiel für die Architekturdokumentation erstellt und auch versioniert werden müssen.

4.1.3 Welche Barrieren entstehen durch nicht barrierefreie Kooperationswerkzeuge?

Die offenkundige Barriere, die Menschen mit einer Sehschädigung im Zusammenhang mit UML Diagrammen erfahren, ist die eindimensionale und grafische Darstellung. Das führt zur Exklusion von Menschen mit einer Sehbehinderung.

Prinzipiell existieren derzeit zwei Möglichkeiten, UML Diagramme für Menschen mit Sehschädigungen zugänglich zu machen. Entweder werden UML Diagramme textuell beschrieben oder taktil aufbereitet und ausgedruckt.

Taktile Ausdrücke eignen sich in erster Linie zum Erlernen von UML und damit fürs Studium bzw. für die Ausbildung. In der Berufspraxis steht meist keine Hardware, d.h. kein taktiler Drucker zur Verfügung. Außerdem fehlen oft auch die Ressourcen und Kenntnisse zum Aufbereiten oder Erstellen von taktilen Grafiken. Zusätzlich erfordert Brailleschrift viel Platz, wodurch Diagramme auf mehrere Blätter verteilt werden müssen und umständlich mit Legenden gearbeitet werden muss.

Textuelle UML Beschreibungen können von blinden Menschen gut genutzt werden, haben aber natürlich auch einige Nachteile. Sie bieten viel Raum für Interpretationen, sind teilweise unpräzise und informell. Eine stärkere Anbindung an die UML Spezifikation wäre hier hilfreich. Außerdem arbeiten sehende Menschen gerne mit graphischen Darstellungen, was hier zu einem Medienbruch bei der Zusammenarbeit führt und die Diskussion erschwert.

Die Arbeitsweise von Menschen mit Blindheit ist oft nicht in den Workflow einer Softwarefirma integriert und ermöglicht auch nur eine unidirektionale Nutzung bzw. Erstellung von grafischen UML Diagrammen. Das heißt, dass basierend auf der textuellen Darstellung eine grafische Darstellung erzeugt wird, welche dann wiederum dem Kunden zur Verfügung gestellt werden kann. Diese kann eine blinde Person beispielsweise mit einem barrierefrei-bediensbaren Werkzeug wie z.B. PlantUML selbst erzeugen, jedoch gibt es kaum Kontrolle über das konkrete Layout. Aus einer Grafik kann aktuell nur durch manuelle Übersetzung einer sehenden Person eine textuelle Darstellung erzeugt werden. Dieser Umstand hindert Menschen mit einer Sehschädigung an einer unabhängigen und effizienten Arbeitsweise.

Zudem sind natürlich die in diesem Leitfaden bereits erwähnten Arbeits- und Kommunikationsaspekte relevant. Das heißt, wenn man über die grafische Version der UML Diagramme diskutiert, müssten die UML Diagramme entsprechend taktil aufbereitet sein, damit blinde Menschen sich auf eine Diskussion vorbereiten bzw. an dieser aktiv teilnehmen können. Exklusion kommt also auch daher, dass man nur eindimensional über Informationen spricht.

Fallstudie - Gemeinsam Software entwickeln mit UML - Erfahrungsbericht eines blinden Softwareentwicklers zum Einsatz von UML

Ein Kooperationswerkzeug für die UML-basierten Softwareentwicklung sollte daher mehrere Sichten auf ein und dasselbe Modell erlauben und unterstützen, dass jede Person ein Diagramm in der bevorzugten Darstellung bearbeiten oder lesen kann. Wie so ein Werkzeug aussehen kann, beschreiben wir im nächsten Abschnitt.

4.2 Das Cooperate-Modellierungswerkzeug

Das Cooperate-Modellierungswerkzeug (im folgenden kurz Kooperationswerkzeug genannt) ist eine Erweiterung (Plugin) für Eclipse, welches das gemeinsame Modellieren von UML in Diversity Teams ermöglicht. Es berücksichtigt die Arbeitsweisen von Menschen mit und ohne Sehschädigung in besonderem Maße. Viele Grundlagen zum Verständnis und der Arbeitsweise von Menschen mit Sehschädigung wurden bereits in den vorausgegangenen Kapiteln erläutert. Das Kooperationswerkzeug unterstützt diese Techniken. In diesem Teil des Leitfadens stellen wir die neuen Konzepte vor, welche das gemeinsame Modellieren verbessern.

4.2.1 Der grundlegende Ansatz

Das Kooperationswerkzeug kann auf verschiedenen Arbeitsplätzen installiert werden und ermöglicht es jedem Teammitglied für das Erstellen und das Bearbeiten von UML-Diagramm entweder einen grafischen oder einen textuellen Editor zu nutzen. Dabei werden UML-Modelle auf einem zentralen Modellspeicher automatisch angelegt und konsistent gehalten. Erfolgen Änderungen an Diagrammen, wird der Modellspeicher aktualisiert, sodass die einzelnen Arbeitsplätze dann die jeweils aktuellste Version abrufen, was dort wiederum für Konsistenz sorgt. Abbildung 4 verdeutlicht das Konzept.

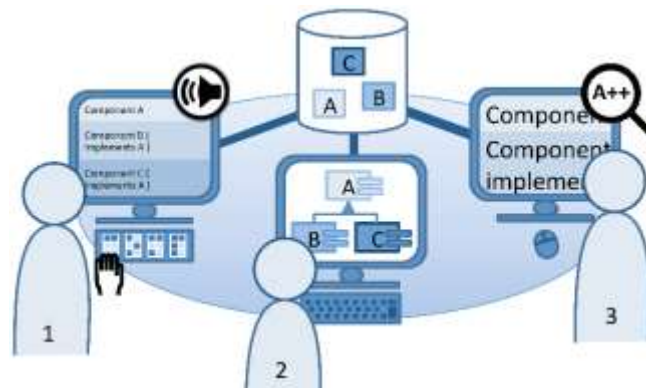


Abbildung 4. Aufbau des Kooperationswerkzeugs im Diversity Team bestehend aus blinden Menschen (1), Menschen ohne Sehschädigung (2) und Menschen mit Sehbehinderung (3). Dabei nutzen alle einen gemeinsamen Modellspeicher (oben Mitte), auf dem das Modell gespeichert wird.

4.2.2 Editierunterstützung

Die wesentliche Funktion, welche das gemeinsame Erstellen und Bearbeiten von UML Diagrammen verbessert, ist die Integration von einem grafischen sowie einem textuellen Editor. Der grafische Editor umfasst Standardfunktionen, die man auch von anderen Werkzeugen zur Erstellung von Diagrammen her kennt. Im Kooperationswerkzeug wurde Papyrus⁵ als grafischer Editor verwendet. Abbildung 5 zeigt ein Bildschirmfoto des graphischen Editors Papyrus.

⁵ <https://www.eclipse.org/papyrus/>

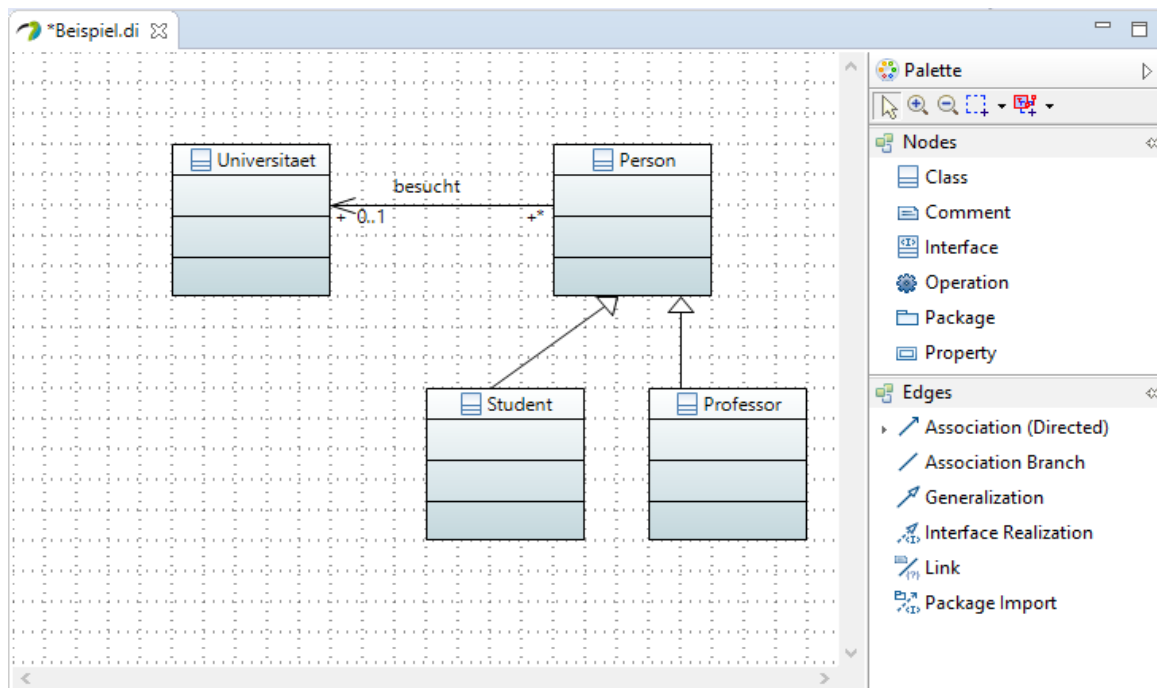


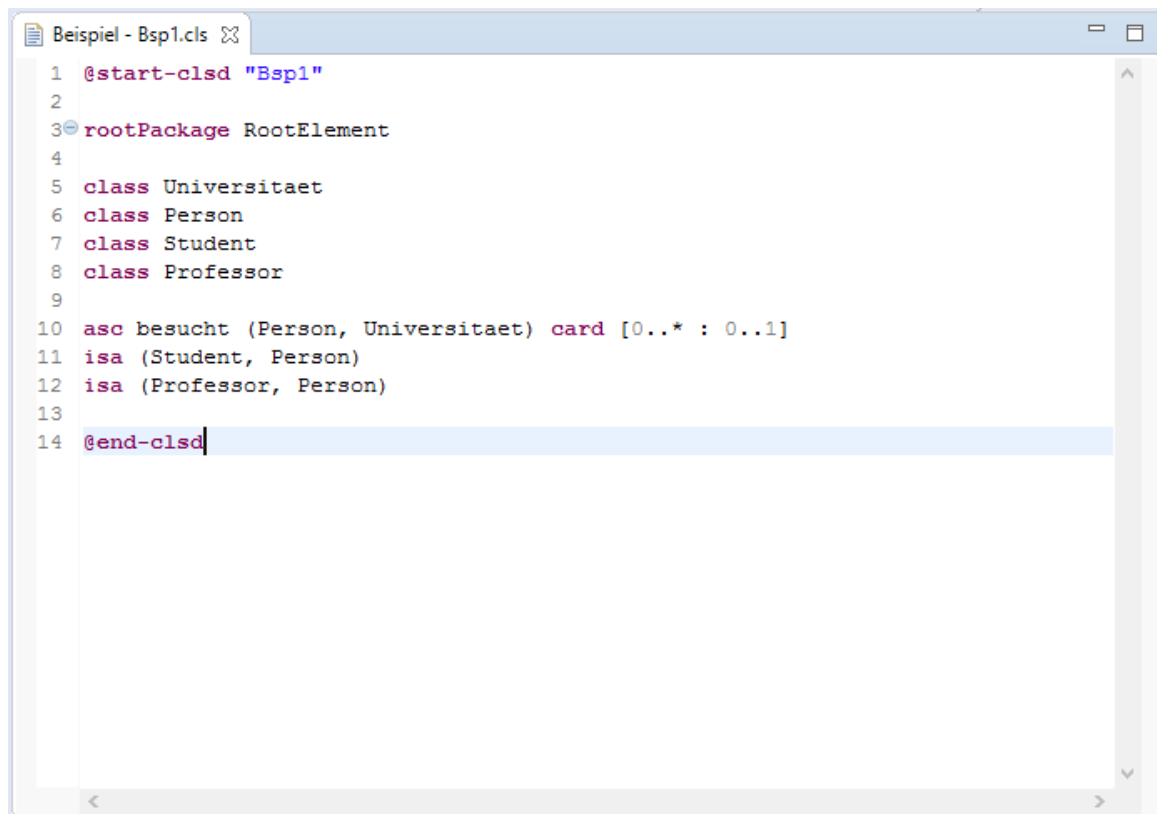
Abbildung 5. Grafischer Editor (Papyrus) des Kooperationswerkzeuges.

Der textuelle Editor hingegen wurde eigens im Projekt Cooperate entwickelt, um das Erstellen sowie das Editieren von UML-Diagrammen in textueller Form zu ermöglichen und effizienter zu gestalten. Dazu wurde zunächst eine neue UML-Syntax entwickelt, welche die Arbeitstechniken von blinden Menschen berücksichtigt. Die Beschreibung der UML4all-Syntax wird auf der eigens dafür angelegten Webseite⁶ gepflegt. Die UML4all Syntax wurde evaluiert und unterstützt die Arbeitsweise von Menschen die einen Screenreader nutzen [23]. Details über die vorausgegangenen Analysen sowie Richtlinien, welche bei der Entwicklung der Syntax angewendet wurden, finden sich in [17] und [18]. Ein Überblick über andere textuelle UML Notationen wird in [19] nachgelesen werden.

Neben der UML4ALL-Syntax wurden Konzepte aus der Programmierung in den textuellen Editor integriert, um das sequentielle Erstellen sowie Lesen von UML Diagrammen in textueller Form zu unterstützen. Dazu zählen die automatische Vervollständigung von Quelltext, Fehlerbenachrichtigung und Refaktorisierungen. Die Auto-Vervollständigung bezieht sich hier sowohl auf Schlüsselwörter, als auch auf Querverweise. Ein Nutzer muss dadurch nicht alle Details der textuellen UML Notation auswendig wissen, was eine enorme Erleichterung zu bisherigen Verfahren darstellt. Abbildung 6 zeigt einen Screenshot des im Cooperate Projekt entwickelten barrierefreien Texteditors.

⁶ <http://www.uml4all.net/de/index.html>

Fallstudie - Gemeinsam Software entwickeln mit UML - Erfahrungsbericht eines blinden Softwareentwicklers zum Einsatz von UML



```
1 @start-clsd "Bsp1"
2
3 rootPackage RootElement
4
5 class Universitaet
6 class Person
7 class Student
8 class Professor
9
10 asc besucht (Person, Universitaet) card [0..* : 0..1]
11 isa (Student, Person)
12 isa (Professor, Person)
13
14 @end-clsd
```

Abbildung 6. Textueller Editor des Kooperationswerkzeuges.

Die Fehlerbenachrichtigungen erfolgen in Echtzeit und ermöglichen es, direkt zur betroffenen Stelle im Texteditor zu springen. Um auch eine blinde Person, die mit einem Screenreader arbeitet, über Fehler zu informieren, wurde eine akustische Benachrichtigung integriert, sobald sich der Cursor in einer Zeile mit einem Fehler befindet. Diese Funktion ergänzt den visuellen Hinweis, der in vielen Editoren eingesetzt wird.

Mittels Refaktorisierungen können komplexe Änderungen in einem einfachen Schritt durchgeführt werden. Beispielsweise kann ein Element umbenannt werden, wodurch auch alle referenzierende Elemente automatisch angepasst werden.

Für das Editieren bestehender Diagramme bietet der Editor Unterstützung beim Explorieren und Verstehen des existierenden Diagramms. Dazu ist es möglich, von Querverweisen direkt zur Definition des Elements zu springen. Die Darstellung kann über das Ausblenden von Quelltextbestandteilen das Querlesen unterstützen.

4.2.3 Weitere Orientierungsfunktionen

Neben der reinen Editorfunktion ermöglichen hierarchische Darstellungen von Diagrammen alternative Orientierungsfunktionen. Dabei werden Diagramme nach UML Elementen (z.B. Pakete, Klassen, Methoden, etc.) geordnet und angezeigt. Nutzer können Teile des Diagrammbaums einklappen um irrelevante Infos auszublenden. Diese Darstellung ist besonders für blinde Menschen geeignet und auch weit verbreitet. Man sollte jedoch bei der Darstellung darauf achten, dass man unterschiedliche Elementtypen im Baum nicht ausschließlich über visuelle Elemente (z.B. Icons) kenntlich macht.

Im Kooperationswerkzeug nutzen wir die sogenannte Outline View von Eclipse um Diagramme als Baum darzustellen. Dabei wurde berücksichtigt, dass man von Elementen im Baum direkt zur Position im textuellen Editor springen kann. Abbildung 7 zeigt ein Bildschirmfoto eines Outline Views.

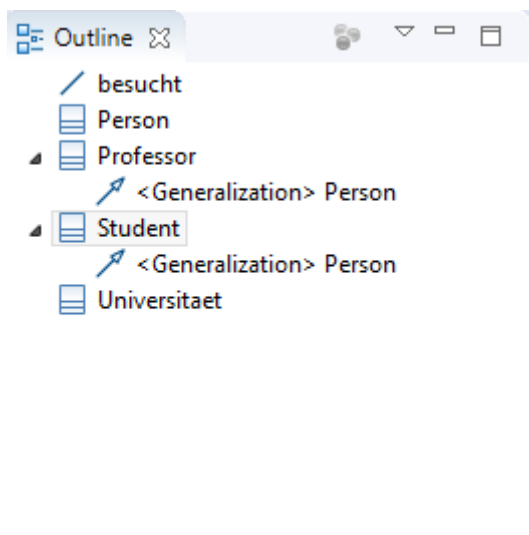


Abbildung 7. Outline View in Eclipse, welche ein UML Diagramm als Baum darstellt.

4.2.4 Synchronisation

Wie eingangs bereits erwähnt, arbeiten alle Teammitglieder auf einem zentralen Repository. Diese wesentliche Funktionalität macht Echtzeitkollaboration erst möglich, da UML-Diagramme in den verschiedenen Darstellungen (grafisch, textuell) nie widersprüchliche Informationen abbilden sollen.

Die Synchronisation zwischen textueller und grafischer Darstellung erfolgt über bidirektionale und inkrementelle Transformationen. Bidirektional bedeutet, dass eine Änderung in beide Richtungen (Text zu Grafik bzw. Grafik zu Text) übertragen werden kann. Inkrementell heißt in diesem Kontext, dass nur die von der Änderung betroffenen Bestandteile geändert werden. Letzteres ist vor allem von Bedeutung, wenn nicht alle Informationen in beiden Notationen vorhanden sind, da diese sonst bei vollständig neuem Erzeugen verlorengehen würden. Ein prominentes Beispiel dafür sind Layoutinformationen. Diese werden nicht zwischen verschiedenen Notationen synchronisiert, sollen aber auch nach Änderungen erhalten bleiben. Die technische Umsetzung der bidirektionalen und inkrementellen Modelltransformationen ist in [21] beschrieben.

4.2.5 Versionskontrolle

Eine weitere elementare Funktion bei großen Softwareprojekten, an denen mehrere Entwickler arbeiten, ist die Versionskontrolle.

Die integrierten Funktionen zur Versionskontrolle von IDEs weisen derzeit noch keine gute Barrierefreiheit auf, weshalb blinde Softwareentwickler oft kommandozeilen-basierte Lösungen verwenden. Gängige Darstellung von Änderungen ist das sogenannte *side-by-side diff*. Diese Anzeige kann schnell unübersichtlich werden, unabhängig davon, ob man eine Sehbehinderung hat oder nicht. Allerdings werden zum Hervorheben von Änderungen meistens Farbkodierungen verwendet, was für Menschen mit einer Sehschädigung problematisch ist.

In das Kooperationswerkzeug wurde daher ein eigens entwickeltes Konzept integriert, welches Änderungen an einem UML-Diagramm leichter zugänglich macht. Dafür kombinieren wir eine hierarchische Darstellung der Diagramme mit einer Tabelle, welche nur relevante Informationen über die Änderungen am Diagramm enthält (Abbildung 8).

Fallstudie - Gemeinsam Software entwickeln mit UML - Erfahrungsbericht eines blinden Softwareentwicklers zum Einsatz von UML

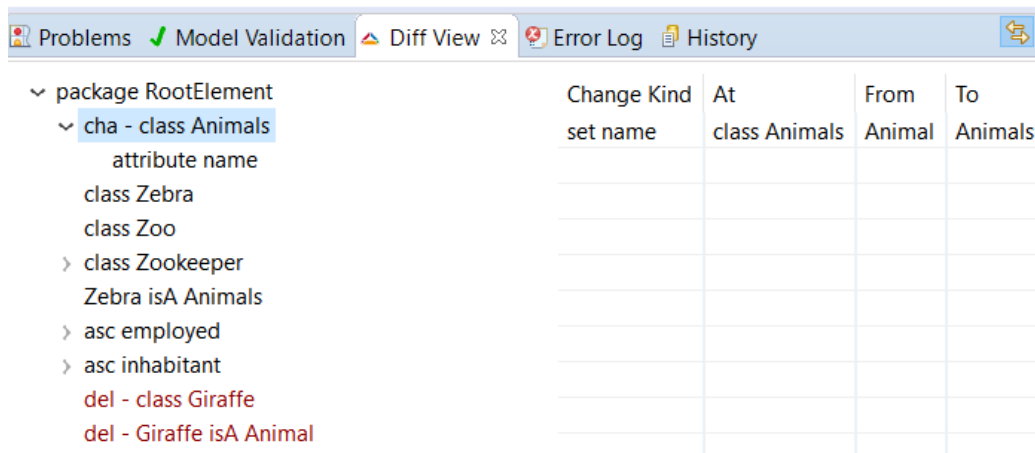


Abbildung 8. Screenshot der Diff-View. Links ist das Diagramm als Baum dargestellt. Änderungen an Elementen sind mit den Abkürzungen cha für change, add für added und del für deleted markiert.

4.2.6 Unterstützung von Diskussionen über Diagramme

Wie in Kapitel 2 beschrieben, erfordert die Kommunikation und Diskussion in Diversity-Teams die Vorbereitung von barrierefreien Materialien und die Aufmerksamkeit bei der Gesprächsführung. Bei Diskussionen über UML Diagramme neigt man auf Grund der visuellen Elemente auch schnell zu nicht-verbalen Gesten oder Sätzen wie „wie sie dort sehen“. Die Folge davon ist, dass Personen mit Blindheit Diskussion nur schwer folgen können.

Um Personen mit Blindheit einen alternativen Zugang zu dieser nicht-verbalen Geste zu ermöglichen, haben wir im Kooperationswerkzeug eine virtuelle Zeigegeste integriert. Das heißt, wenn sich ein Teammitglied in der Focus-View befindet und ein bestimmtes Element fokussiert, werden die anderen Teilnehmer an ihrem eigenen Arbeitsplatz darüber informiert und können ebenso zu dem Element springen und dieses während der Diskussion näher erkunden. Abbildung 9 zeigt ein Bildschirmfoto, wie eine solche Zeigegeste im Editor realisiert wurde.

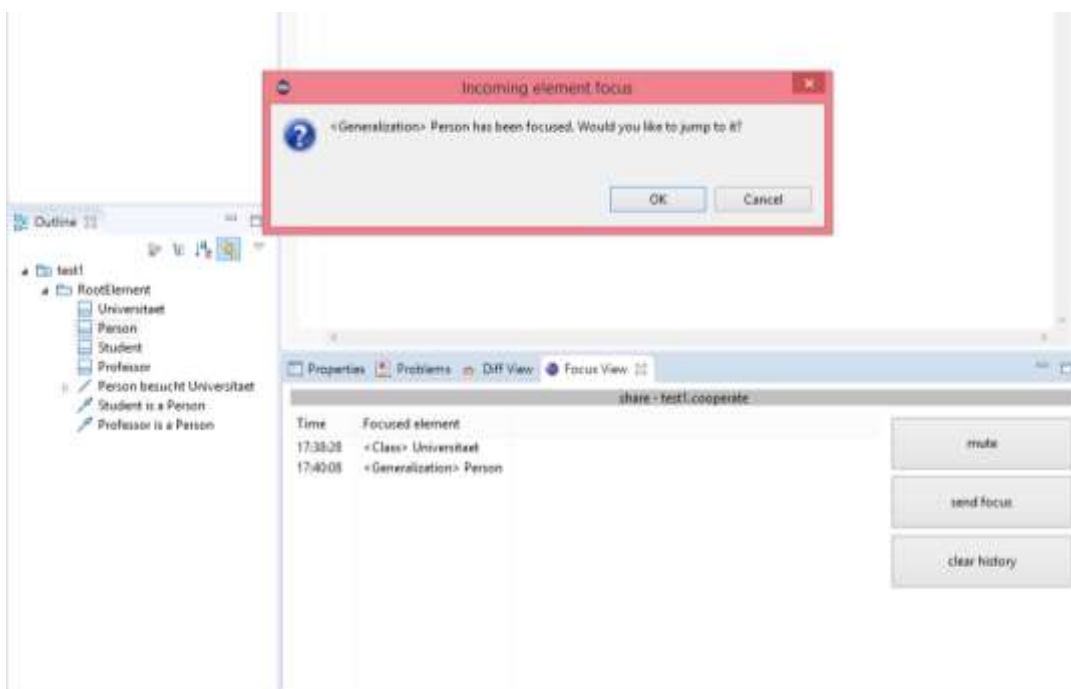


Abbildung 9. Focus View des Cooperate Werkzeuges zeigt ein Event, dass auf ein Element „gezeigt“ (geklickt) wurde.

5 ZUSAMMENFASSUNG

Damit Menschen mit und ohne Behinderung im Bereich der Softwareentwicklung zusammenarbeiten können müssen Werkzeuge (z.B. IDEs) und Methoden (z.B. grafische Beschreibungssprachen) barrierefrei sein und die besonderen Aspekte bei der Zusammenarbeit berücksichtigen. Es gibt heute bereits Programmierumgebungen wie Eclipse, die zu weiten Teilen barrierefrei sind. Allerdings macht die Zusammenarbeit im Arbeit erforderlich, spezielle Funktionen bereitzustellen.

In diesem Leitfaden haben wir die Ergebnisse des Cooperate-Projektes zusammengefasst und geben Hinweise für die Entwicklung von barrierefreien, inklusiven Kooperationswerkzeugen – also softwarebasierten Lösungen, welche die Zusammenarbeit von Menschen mit unterschiedlichen, insbesondere visuellen, Fähigkeiten unterstützen. Neben konkreten Lösungen und Anforderungen an Werkzeuge, die das gemeinsame Modellieren von Software mit UML zu Ziel haben, erklären wir auch grundlegende Informationen zum Thema Sehschädigung und beschreiben Anforderungen von Menschen einer Sehschädigung an die Arbeit mit digitalen Informationen und Kommunikationstechnologien. Diese Informationen sind die Grundlage um die Anforderungen von Menschen mit Sehschädigungen zu verstehen und neue Kooperationswerkzeuge zu entwickeln.

6 QUELLEN

- [1] <http://corporate-senses.com/sensorische-reize/>
- [2] <http://www.dbsv.org/augenkrankheiten.html>
- [3] <https://joeclark.org/book/sashay/serialization/Chapter03.html>
- [4] <http://www.bsvsb.org/index.php/definition-sehbehindert.html>
- [5] <http://www.who.int/mediacentre/factsheets/fs282/en/>
- [6] World Health Organization, "How to use the ICF: a practical manual for using the International Classification of Functioning, Disability and Health (ICF)," 2013.
- [7] <http://www.bsv-nordrhein.de/allgemeine-informationen/wer-ist-sehbehindert-bzw-blind.html>
- [8] <http://www.dog.org/>
- [9] <https://www.blickcheck.de/auge/funktion/gesichtsfeld/>
- [10] Portal der Augenmedizin, Farbenfehlsichtigkeit, Farbenblindheit: <http://www.portal-der-augenmedizin.de/augenkrankheiten/farbenblindheit/farbenfehlsichtigkeit-farbenblindheit.html>
- [11] Beispiele Sehtestbilder: <https://www.sehtestbilder.de/>
- [12] WHO, Fact Sheet: <http://www.who.int/mediacentre/factsheets/fs282/en/>
- [13] Blickcheck, Wie kommt es zur Sehbehinderung? – Ursachen: <https://www.blickcheck.de/auge/krankheiten-und-sehstoerungen/sehstoerungen/sehbehinderung/>
- [14] Pro Retina, Sehstörungen: Simulator von PRO RETINA und BKK: <http://www.pro-retina.de/simulation>
- [15] Cooperate Eclipse Schulung für Blinde und sehbehinderter Menschen. Online verfügbar: <https://www.cooperate-project.de/eclipseschulung/Eclipse.pdf>
- [16] PETRAUSCH, Vanessa; LOITSCH, Claudia. Accessibility Analysis of the Eclipse IDE for Users with Visual Impairment. AAATE'17. IOS press, 2017. Online verfügbar: https://www.cooperate-project.de/images/AAATE_WEB.pdf
- [17] MÜLLER, Karin; PETRAUSCH, Vanessa; JAWOREK, Gerhard; HENSS Jörg; SEIFERMANN, Stephan, LOITSCH Claudia; STIEFELHAGEN, Rainer. UML4ALL: Gemeinsam in Diversity Teams Software modellieren. In: Informatik Spektrum (2017). <https://doi.org/10.1007/s00287-017-1073-y>
- [18] PETRAUSCH, Vanessa; SEIFERMANN, Stephan; MÜLLER, Karin. Guidelines for Accessible Textual UML Modeling Notations. In: International Conference on Computers Helping People with Special Needs. Springer International Publishing, 2016. S. 67-74. <https://www.cooperate-project.de/images/icchp2016.pdf>
- [19] SEIFERMANN, Stephan; GROENDA, Henning. Survey on Textual Notations for the Unified Modeling Language. Proceedings of the 4th International Conference on Model-Driven Engineering and Software Development, 2016, S. 20-31. <https://www.cooperate-project.de/images/publications/MODELSWARD2016.pdf>
- [20] SEIFERMANN, Stephan; GROENDA, Henning. Towards Collaboration on Accessible UML Models. Mensch und Computer 2015–Workshopband, 2015, S. 411-417. <https://www.cooperate-project.de/images/AI4VIP.PDF>
- [21] SEIFERMANN, Stephan; HENSS, Jörg. Comparison of QVT-O and Henshin-TGG for Synchronization of Concrete Syntax Models. In Proceedings of the 6th International Workshop on Bidirectional Transformations (Bx 2017), Romina Eramo and Michael Johnson, editors, Uppsala, Sweden, 2017, volume 1827 of CEUR Workshop Proceedings, pages 6-14. CEUR-WS.org. 2017. <http://ceur-ws.org/Vol-1827/paper5.pdf>
- [22] Vanessa Petrausch: Einführung in die grafische Beschreibungssprache UML: https://www.cooperate-project.de/umlschulung/1_1_UMLSchulung_v2.pdf

- [23] Claudia Loitsch, Karin Müller, Gerhard Jaworek, Stephan Seifermann, Jörg Henß, Sebastian Krach and Rainer Stiefelhagen: UML4ALL Syntax – A Textual Notation for UML Diagrams, akzeptiert für die Veröffentlichung auf der ICCHP 2018