Color Ramp
VISUALISIERUNG

VON FELICITAS GRUNENBERG UND MARY-ANNE KOCKEL
FÜR ZHDK & UNIVERSITÄT ZÜRICH, 2007

Ziel des Hochschulprojektes war es eine Applikation zur Visualisierung von großen Mengen Quellcodes mit verschiedenen Parameter zu entwickeln. Der Fokus von Color Ramp liegt im Einsatz von intelligenten Filtern. Die Filterergebnisse können verglichen und anschliessen als Vorlage gespeichert werden.

Konzeption & Entwurf
Mary-Anne Kockel → www.paka.me
Felicitas Grunenberg

Programmierung
Stefan Holm

Dokumentation
Mary-Anne Kockel

SOFTWARE VISUALISIERUNG 2.0

COLOR RAMP

MODUL
Applied Interaction I

KURS
Software Visualisierung

JAHR
Herbstsemester 2007/08
5. Hochschulsemester

STUDENTEN
Felicitas Grunenberg (IAD)
Mary-Anne Kockel (IAD)
Stefan Holm (UZH)

DOZENTEN
Prof. Jürgen Späth, Prof. Dr. Harald Gall
Sandro Boccuzzo, Beat Fluri

FRAGE-
STELLUNG

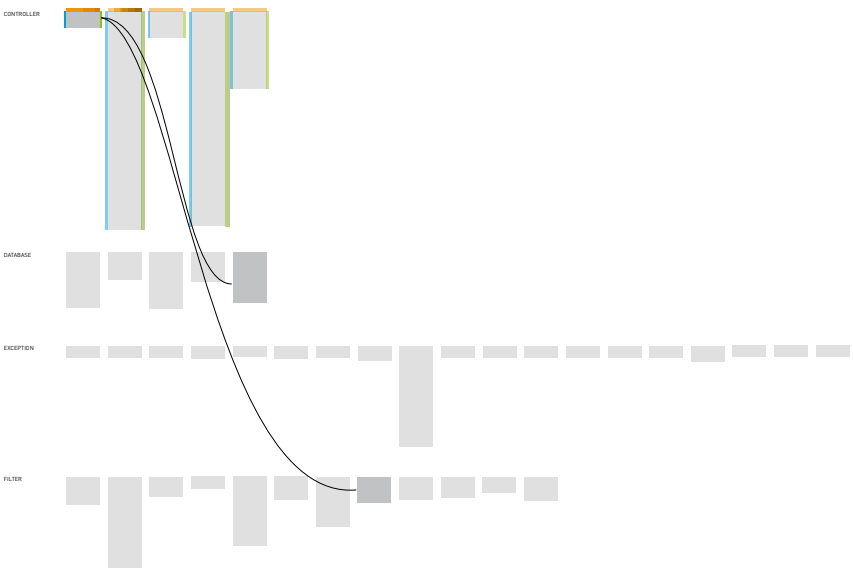
Software visualisieren.

Die Softwarevisualisierung ist eine spezielle Art der Informationsvisualisierung und lässt sind in verschiedenen Bereichen wiederfinden. Der Programmcode einer Software wird nicht wie bisher in Textform dargestellt, sondern in Grafiken und Diagrammen. Dafür ist eine statistische Erfassung von Quellcode-Parametern (Metriken) notwendig. Mit Hilfe dieser Metriken können die Quelltextkomponenten exakt beschrieben werden.

Eine Softwarevisualisierung wendet sich in erster Linie an Softwareentwickler, um Hierarchien abzubilden, Fehler aufzudecken, Komplexitäten zu verringern, Verknüpfungen zwischen einzelnen Komponenten aufzuzeigen, Metriken der Programmcodeteile zu vergleichen, beteiligte Entwickler und deren Anteil zu erforschen, sowie die zeitliche Entwicklung des Systems darzustellen.

Auf Grund der vermehrten Nachfrage nach Anwendungen für Softwarevisualisierung hat sich die Universität Zürich an uns Interaction DesignerInnen gewandt, um neue Ansätzen zu finden. Das Projekt basiert auf einer ersten Analyse[1] aus dem Sommersemester 2007. Auf Basis dieser Erkenntnisse haben wir ein neues Konzept entwickelt.

1 Modell vom SS 2007



ANALYSE

Zielgruppe

Im einfachsten Fall gibt es zwei Zielgruppen. Zum einen die Projektleiter, die für die Organisation zuständig sind und zum anderen die Programmierer, die die eigentliche Kodierung leisten.

Die verschiedenen Zielgruppen haben bestimmte Prioritäten, die in der Softwarevisualisierung dargestellt werden müssen. Für die Projektleiter sind das die Fehler, die Komplexität sowie die Anzahl der Quellkomponenten und Entwickler. Der Fokus der Entwickler liegt auf der Komplexität, den Aufrufen der Komponenten untereinander, sowie der hierarchischen und logischen Struktur.

Wir haben unseren Schwerpunkt auf die Projektleitersicht und das Auffinden von fehlerhaften Programmteilen gelegt. Auf 2. Ebene versuchten wir weitere Prioritäten des Projektleiters und Entwicklers zu berücksichtigen und in unser Modell zu integrieren.

METRIKEN	Developer	PL	
Size • Lines of Code • Number of Func.	X	*	
Komplexität	X	*	
• Reports (Bug)	X	X	
Developer ①		X	
Versionen	X	X	historische Verfolgung
Calls • Incoming • Outgoing	X X		
Package	X	X	Lokalisierung
Verbung (Parent child)	X		

Struktur

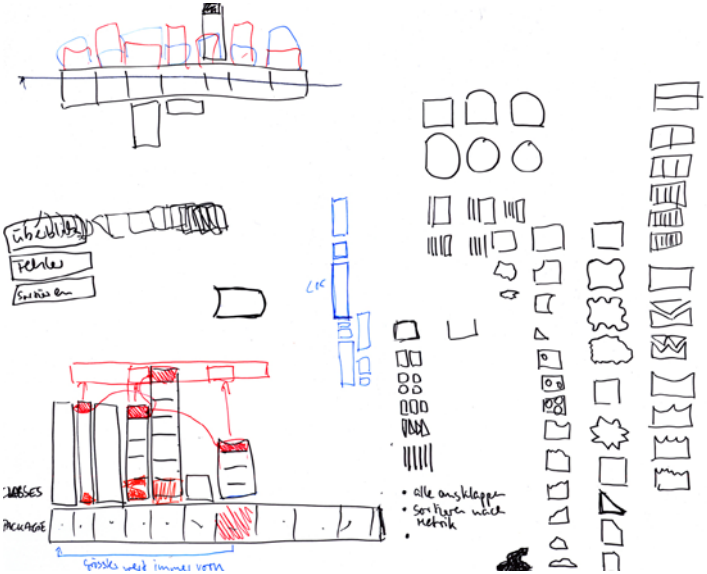
Für unsere Inspirationen habe wir Strukturen aus der Natur, der Architektur und der Textilkunst entnommen. Als Interaktionsmodelle für Gewichtung und Auswahl erschienen uns die Rechenmaschine und der Fächer als geeignet. Die Form- und Strukturanalyse von möglichen Darstellungsarten basiert auf der Hervorhebung von Besonderheiten im Speziellen von Fehlern.

Wie kann ein Fehler im System erkenntlich gemacht werden, sowie dessen Grad oder Gewichtung?

Mit Hilfe des Morphologischen Kastens haben wir mit verschiedenen Formen und deren Transformation experimentiert. Dabei sind wir zu dem Schluss gekommen, dass der Wechsel von eckigen zu runden Formen optisch eindeutig und von der Ferne gut zu erkennen ist.

Zusätzlich müssen wir uns die folgenden Fragen stellen:
Wie können die weiteren Metriken neben den Fehlern aussehen?
Wie kann ein System gefiltert und durchsucht werden?
Wie lassen sich Hierarchien und Verknüpfungen darstellen?
Wie intuitiv kann der Umgang mit der gesamten Softwarevisualisierungsapplikation gestaltet werden? (Human Engineering)

1 Stofffetzen mit Hervorhebungen[2]
2 Schal mit rechteckigem Raster[2]

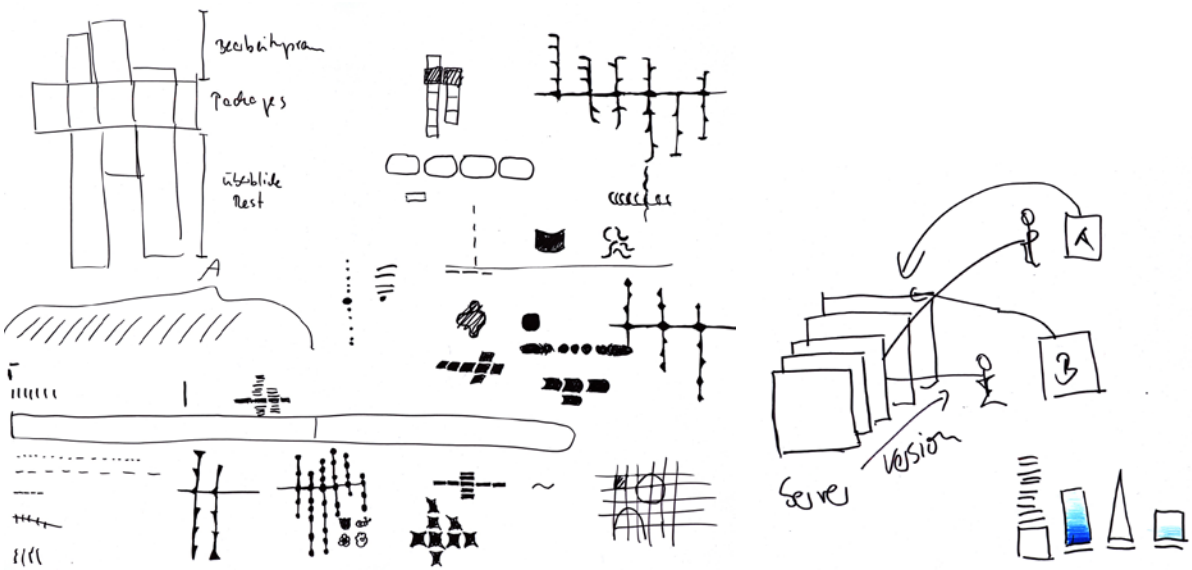

PACKAGE
• alle anschlagen
• sortieren nach metrik

1 Strukturstudien

2 Morphologische Transformation

3 Hierarchie- und Gruppierungsstudien

4 Beispiel für Versionen einer Software



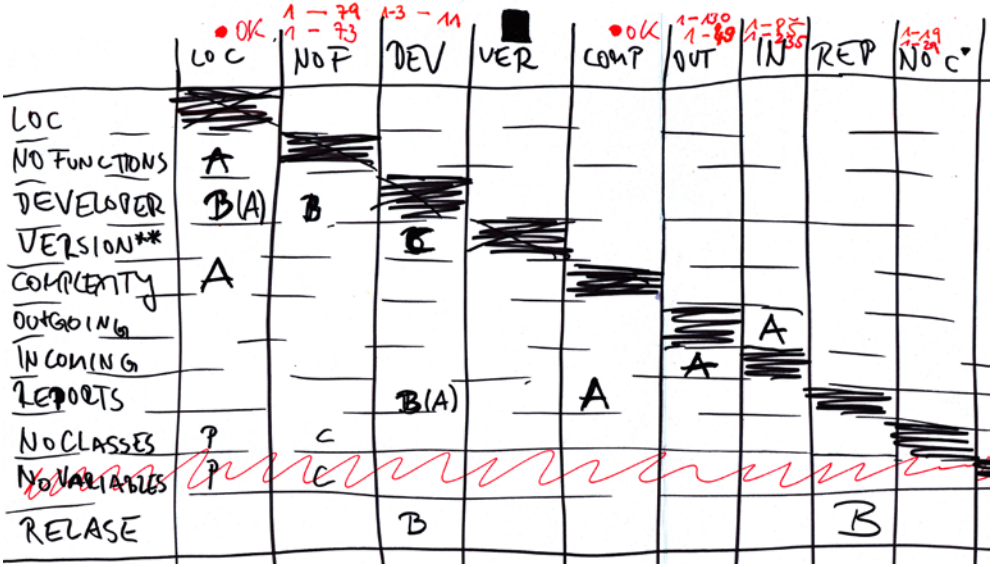
Metriken

Ein Softwaresystem besteht aus verschiedenen Quellcode-Parametern, die im Fachjargon Metriken genannt werden. Mit Hilfe der Metriken, die durch Zahlenwerte repräsentiert werden, lassen sich verschiedene Eigenschaften erkennen und Aussagen über Programmkomponenten machen.

Wir haben die fünf folgenden Metriken genauer betrachtet. Die Grösse gibt Auskunft über die Anzahl der Quellcodezeilen [Lines of Code] mit der Unterteilung in Anzahl der Funktionen [Number of Functions] und Variablen [Number of Variables]. Die Komplexität sagt aus, wie verschachtelt ein Quellcode ist. Die Metrik Report legt den Fokus auf bereits existierende Fehler. Die Entwickler geben Aufschluss über die Anzahl der beteiligten Programmierer und wieviel sie zum Quellcode beigetragen haben. Die Calls zeigen die Aufrufe von Programmteilen untereinander. Diese werden in Incoming und Outgoing aufgeteilt.

In der Kombination zweier Metriken lassen sich noch weitere Eigenschaften erkennen, die sowohl für den Projektleiter als auch für den Entwickler von Bedeutung sind. Folgende Verknüpfungen sind für den Projektleiter wichtig: Lines of Code und Number of Functions, Lines of Code und Komplexität, Reports und Komplexität, sowie die Incoming und Outgoing Calls.

1 Matrix mit den Kombinationen der wichtigsten Metriken



MODELLE

Modell Feuerstein

In unserem ersten Modell liegt die Priorität in der konkreten Suche nach Fehlern. Diese werden pro Klasse und gruppiert pro Package angezeigt (hierarchische Struktur). Der Fehler verändert die Form einer Klasse und somit auch des Packages. Die Komplexität wird analog, aber spiegelverkehrt angezeigt.

Eine Klasse wird durch ein Rechteck gekennzeichnet. Bei hoher Fehleranfälligkeit und Komplexität verformt sich das Rechteck stufenweise zu einem Kreis. Die Klassen sind in Packages zusammengefasst und bilden so einen Stapel.

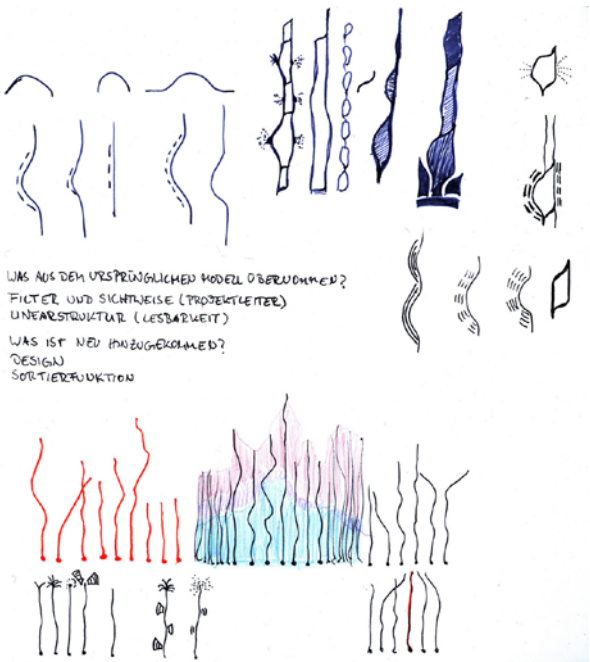
Zur Gewichtung der Fehler und der Komplexität kann ein Filter verwendet werden. Der Stapel wird nach höchster Fehleranzahl bzw. grösste Komplexität umsortiert, so dass die grösste Verformung unten ist.

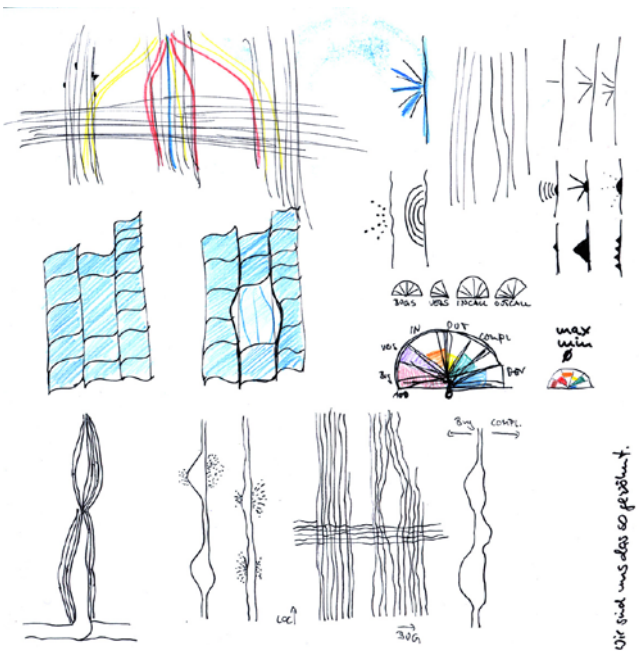
Ein zweiter Filter zeigt die Vererbung (logische Struktur) der Klassen. Dabei werden Rechtecke oder deren Transformation verschieden grau eingefärbt.

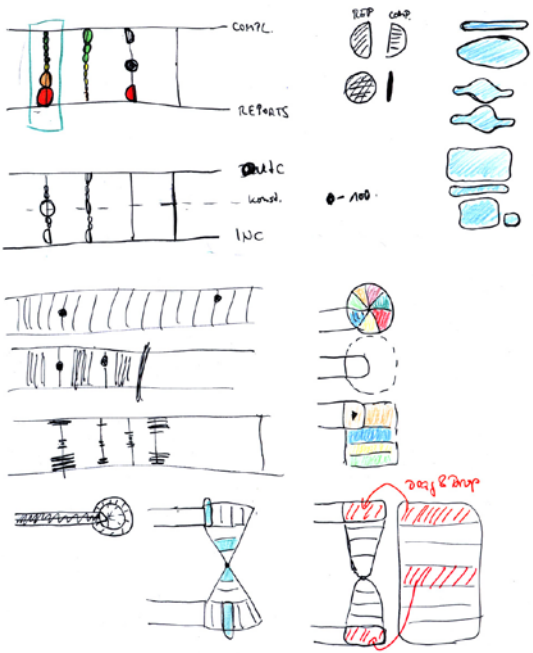
Zum Filtern haben wir ein fächerartiges Interface entwickelt, das die verschiedenen Metriken und deren Intervalle zur Auswahl stellt.

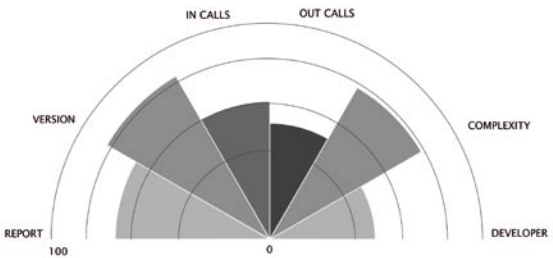
1 Studie über Transformation und Anordnung

2 Versuch markante Stellen hervorzuheben






COMPL.
REPORTS
INC
REP
COMP.
0 - 100

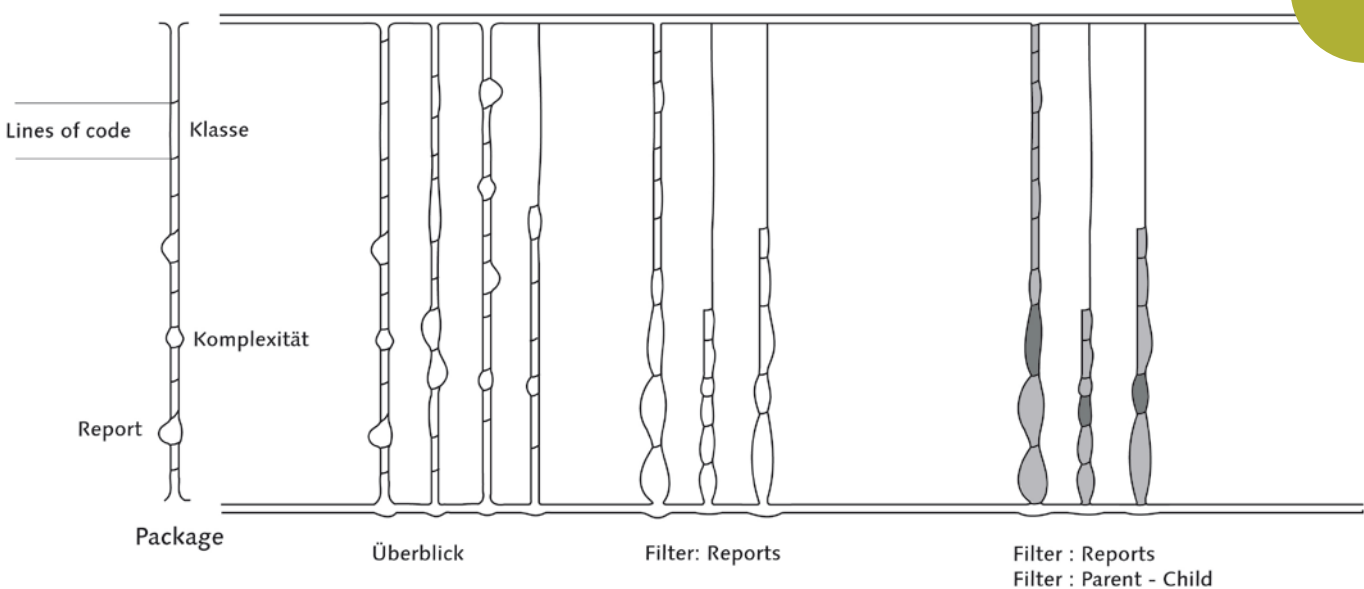

IN CALLS
OUT CALLS
VERSION
COMPLEXITY
REPORT
DEVELOPER
100
0

1 Skizze des Modells Feuerstein mit angedeutetem Workflow

2 Fächerartiges Interface

3 Modell einer Packagestruktur mit den Klassen und Hervorhebungen

4 Anordnung mehrere Packages und Filterfunktion



Modell Pixelflow

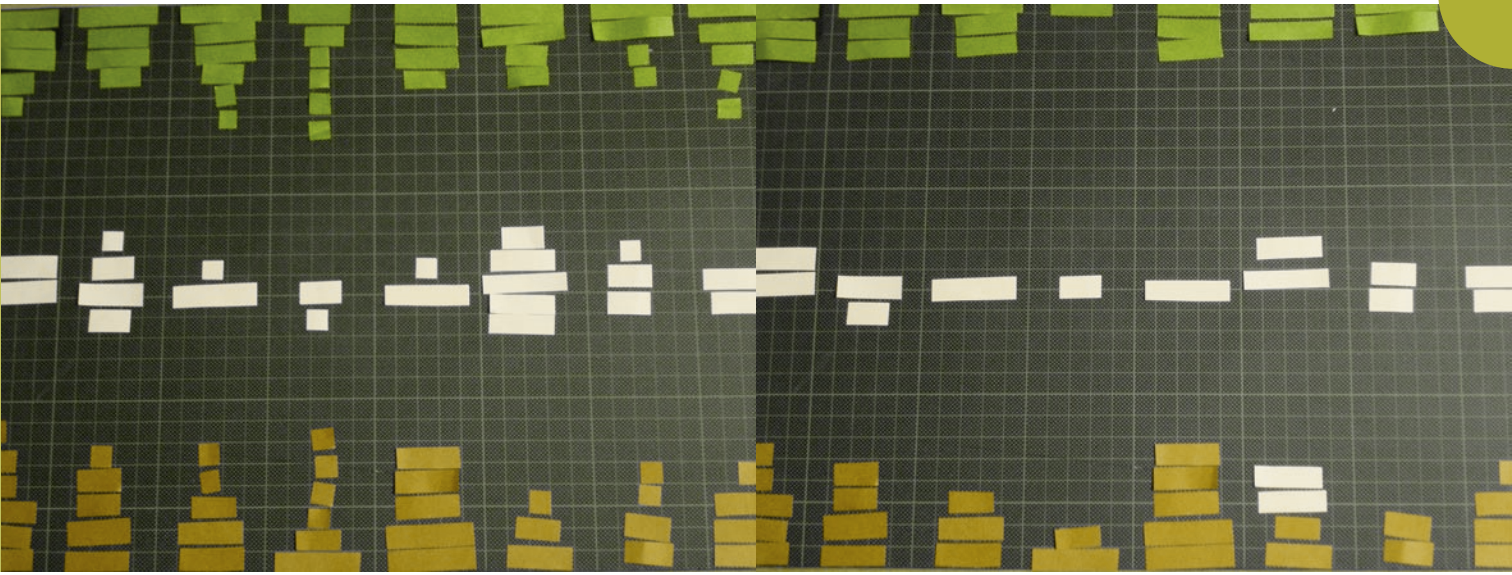
Im zweiten Modell sind die Filterfunktion gegenübergestellt. Das ermöglicht die Kombination zweier Metriken. Die Klassen werden somit verhältnisabhängig dargestellt und einzelne Extreme tendieren in die eine oder andere Metrikrichtung.

Die Klassen sind als Rechtecke dargestellt und zu Packages gruppiert. Neu ist, dass jede Metrik einer Farbe zugeordnet ist. Bei der Anwendung einer Metrik auf die Klassen werden die Rechtecke mit der Farbe eingefärbt. Bei zwei gegenübergestellten Metriken ergibt sich eine Mischfarbe, die das Verhältnis der Metrikwerte wiederspiegelt. Die Extreme, die ein ungleiches Verhältnis haben und damit ein Übergewicht, drängt es nach aussen. Die Metriken ziehen diese Extreme wie Pole an. Die ausgewogenen Klassen mit gleichem Verhältnis befinden sich in der Mitte.

Um sich einen besseren Überblick zu verschaffen gibt es einen Filter, mit dem die Verhältnisstufen der unausgeglichenen Klassen betrachtet werden können.

1 Ansicht aller Klassen und Packages

2 Ansicht nach der Verwendung des Filters



Modell Farbverlauf

Die Haupterkenntnis aus dem Modell Pixelflow ist, dass die eingefärbten Klassen nach Mischfarbe sortiert einen Farbverlauf ergeben. Darauf haben wir das dritte Modell entwickelt. Die Basis des Modells ist ein Farbverlauf, der sich durch zwei gegenübergestellte Metriken ergibt.

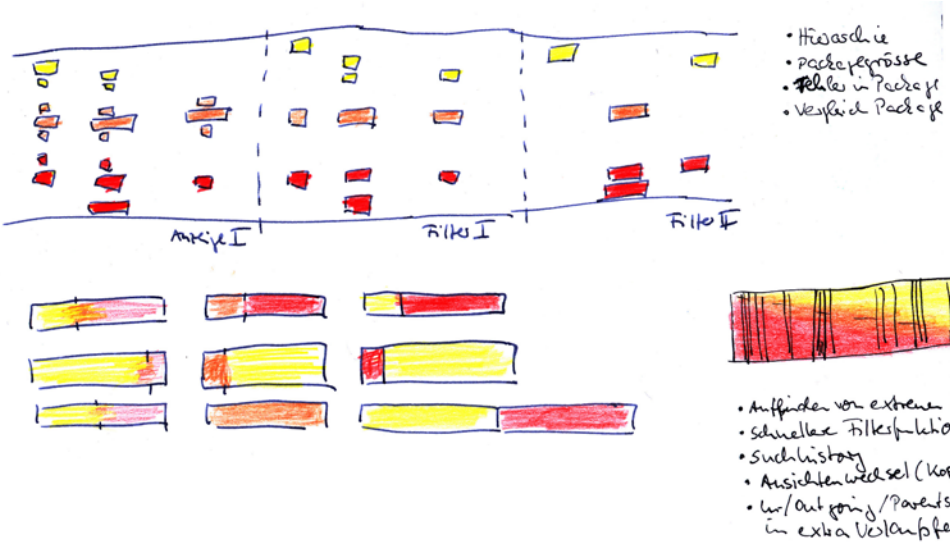
Die Gruppierung der Klassen in Packages ist aufgelöst und die Rechtecke der Klassen sind durch Linien ersetzt. Packages und Klassen werden getrennt betrachtet. Dadurch können zum Beispiel Klassen hierarchieunabhängig für das gesamten Softwaresystem miteinander verglichen werden.

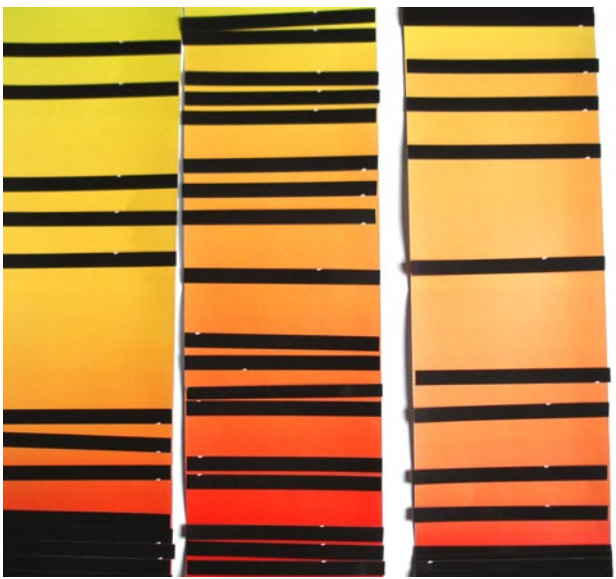
In der Package-Ansicht können über eine konkrete Auswahl die zugehörigen Klassen in einer separaten Ansicht untersucht werden. Ausserdem wird die Vererbung wie auch die Incomings und Outgoings in neuen Farbverläufen angezeigt. Die Verknüpfungen der Packages und Klassen untereinander werden durch Linien sichtbar gemacht.

Mit Hilfe eines Filters für die Verhältnisse und damit den Mischfarben kann ein Intervall im Detail betrachtet werden. Die Klassenrepräsentaten verschieben sich je nach Auswahl an den Rand oder in die Mitte.

1 Weg vom Modell Pixelflow zum Farbverlauf

2 Intervallansicht mit der Verschiebung der Klassen





1 Package-Klassen-Ansicht
2 Intervallansicht im Detail mit verschobenen Klassen
3 Ansicht der Incoming Calls
4 Incoming und Outgoing Calls einer Klasse

COLOR RAMP

Farbcode Metriken

Die Farben für die Metriken des Modells ColorRamp haben wir anhand des Farbkreises von Itten ausgesucht. Unsere Präferenz galt den reinfarbigen und gut unterscheidbaren Farben, um zu gewährleisten dass bei der Zuweisung zu den Metriken die Mischverhältnisse eindeutig hervorgehen.

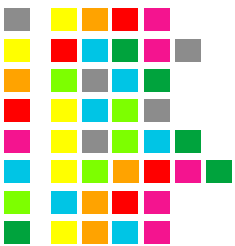
Nach der Zuweisung wurde die Farbauswahl inklusive der wichtigsten Farbverläufe für den Projektleiter anhand der Metrikmatrix überprüft.

Für die Auswahl der Metriken ist ein Interface vorgesehen, das der Assoziation eines Farbfächers entspricht. Die kreisförmige Anordnung setzt keine Prioritäten und ermöglicht eine schnelle Auswahl mit der Maus.

Um den Farbfächer gibt es einen zweiten, mit dem die Hierarchieebenen eingestellt werden. Durch das stufenweise Aufziehen des äusseren Fächers werden die tieferen Hierarchien dargestellt. Im gleichen Aussenfächer kann ausserdem zwischen den Ansichten von Klassen und Packages gewechselt werden.

1 Auswahl der Farben und Verlaufmuster

2 Modelle für das Fächer-Interface



Lines Of Code 255- 255-0

Number Of Functions 255-0-153

Complexity 0-153-51

Reports 255-0-0

Developer 0-204-255

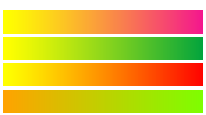
Version 0-102-153

Incoming 255-163-0

Outgoing 125-255-0

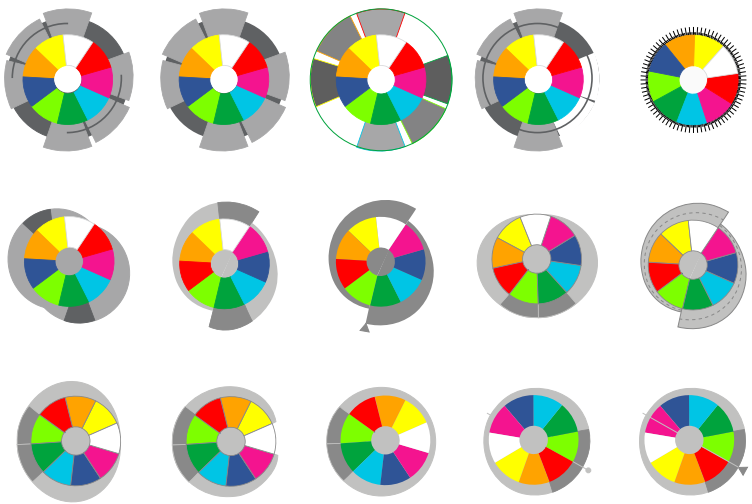
Blank 255-255-255

Kategory A



Kategory B

Kategory C



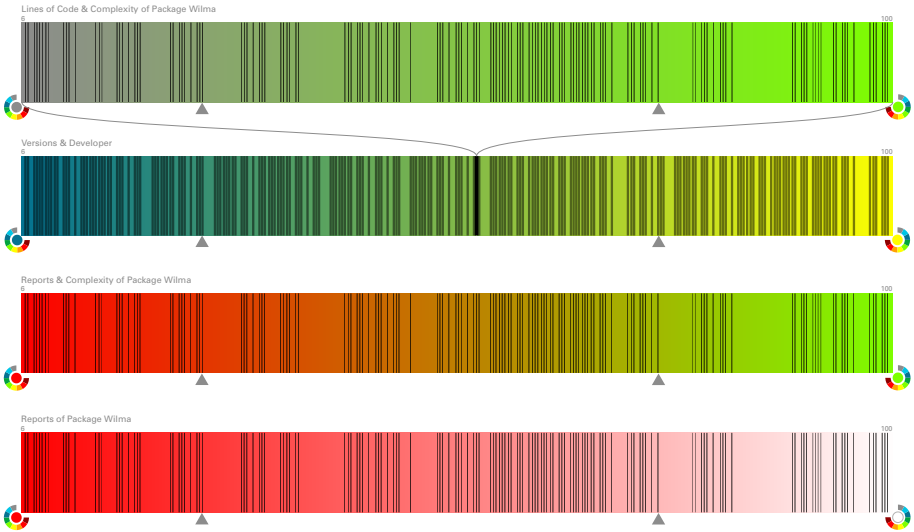
Programmanalyse

Die Farbverläufe geben Auskunft über die einzelnen Programmkomponenten (Klasse/Package/Methode) und deren Verhältnis zueinander. Die Darstellung und Analyse von Klassen, Packages und Methoden erfolgt analog. Aus diesem Grund wird in den weiteren Beschreibungen nur noch von Klassen gesprochen.

Die Klassen werden als Aussparung des Farbverlaufs mit grauen Linien dargestellt. Zu Beginn existiert allerdings noch kein Farbverlauf, nur ein farbneutraler Balken ist erkennbar. Die Farben der Metriken müssen links und rechts dem Balken zugewiesen werden. Nach Definition der Metriken platzieren sich die Klassen im entstandenen Farbverlauf entsprechend ihres Verhältnisses. Der Balken kann auch nur mit einer Metrik befüllt werden.

Eine Auswahl mehrerer Klassen, ersichtlich durch schwarze Linien und deren Namen, ermöglicht die Weiterverfolgung im Arbeitsprozess.

1 Anzeige von verschiedenen Metriken mit Farbverlauf



Programmstruktur

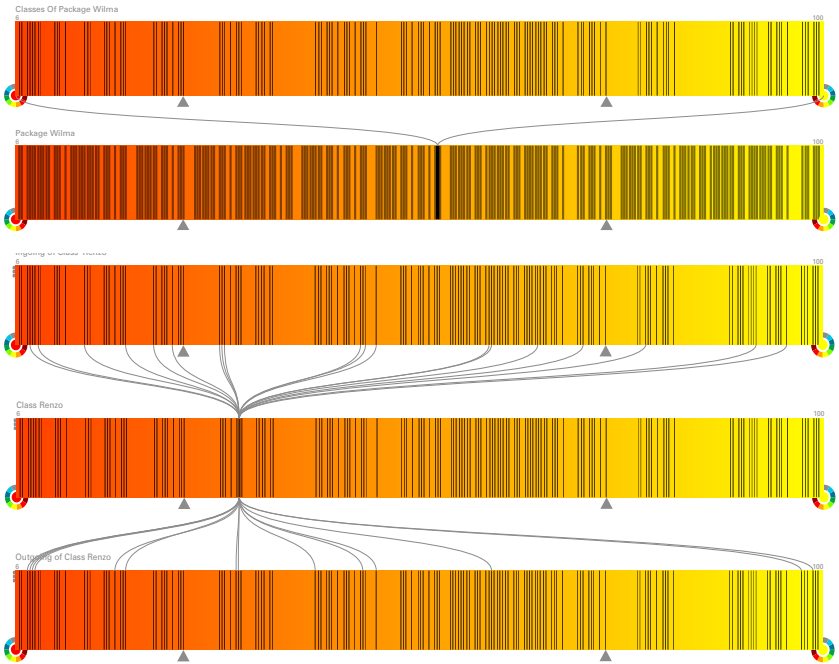
Neben der Programmanalyse wird auch die Struktur abgebildet. Packages werden mit dickeren grauen Linien dargestellt. Mit der Auswahl eines Packages zeigen sich die zugehörigen Klassen in einem neuen Balken mit gleichem Farbverlauf.

Die Aufrufe der Klassen untereinander und damit die Incomings und Outgoings werden ebenfalls über die Auswahl einer Klasse aktiviert. Dabei werden zwei neue Balken mit identischem Farbverlauf erstellt. Im oberen Balken zeigen sich die Klassen mit den Incoming Aufrufen und im unteren Balken die Klassen mit den Outgoing Calls. Zusätzlich werden die Verknüpfungen mit Linien zwischen den Balken sichtbar gemacht.

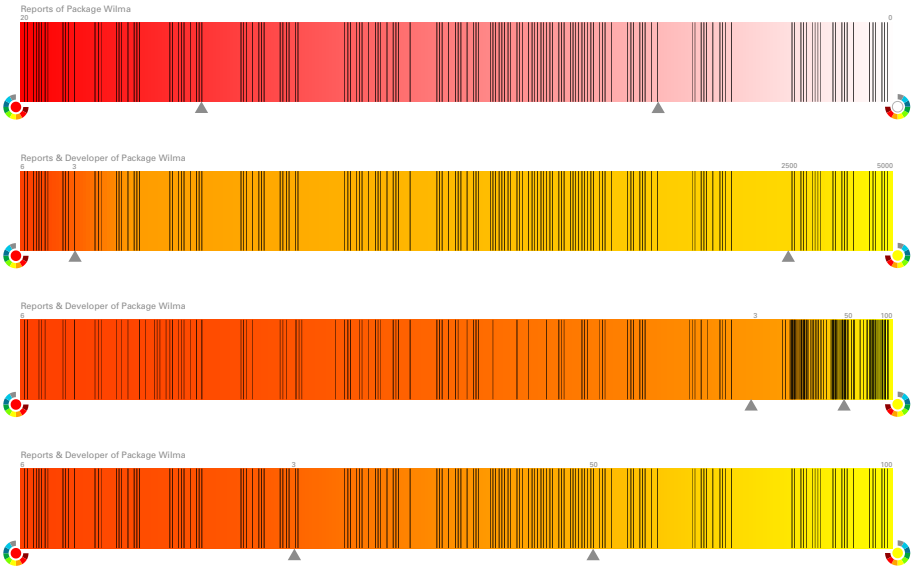
Um Teilbereiche im Farbverlauf zu untersuchen, kann mit Hilfe der grauen Pfeile das Intervall vergrössert und verkleinert werden. Dabei schieben sich die betroffenen Klassen in die Mitte und die Restlichen an den Rand.

1 Ansicht der Klassen und Packages

2 Incomings und Outgoings einer Klasse



1 Verschiedene Intervallbereiche



Programmfragen

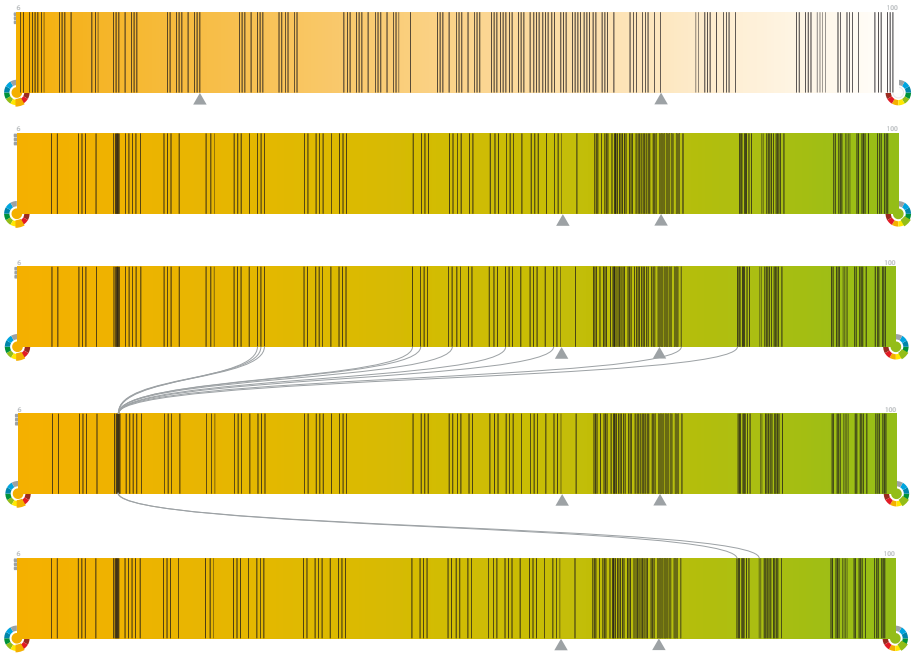
Wir haben versucht zwei wichtige Fragen aus der Sicht des Projektleiters mit der ColorRamp durchzuspielen.

Welche Klassen sind aktiv und welche passiv?

Für die Beantwortung dieser Frage empfiehlt es sich die Incomings und Outgoings zu betrachten. Aktive Klassen weisen viele Incoming sowie Outgoings oder nur viele Outgoings auf. Hingegen passive Klassen wenige Incomings sowie Outgoings oder nur Incomings besitzen. Es gibt verschiedene Möglichkeiten das mit ColorRamp herauszufinden. A: Die Metriken Incoming und Outging werden gegenübergestellt. B: Die Metriken werden einzeln betrachtet. C: Es werden einzelne Klassen ausgewählt, um deren Incoming- sowie Outgoing-Verknüpfungen festzustellen.

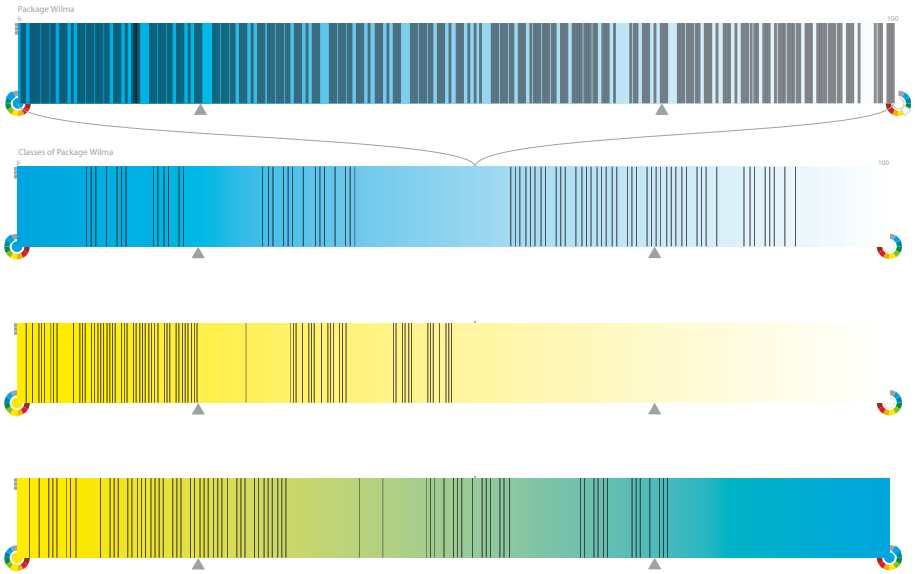
Wie fehleranfällig ist ein Package?

Eine Variante zur Analyse ist das Verhältnis der Metriken Developer und Lines of Code zu vergleichen. Viele Entwickler und viele Lines of Code deuten auf ein fehleranfälliges Package hin, da der Verantwortungsbereich nicht genügend definiert ist. Eine Klasse mit wenig Entwicklern ist wahrscheinlich gut strukturiert und somit weniger fehleranfällig.


6
100
6
100
6
100
6
100
6
100

1 Welche Klassen sind aktiv und welche passiv?

2 Wie fehleranfällig ist ein Package?



Farbcode PH-Streifen

Wird der Farbverlauf mit dem Farbstreifen invertiert, kann jeder Linie ein exakter Farbwert zugewiesen werden. Der Farbwert gibt das Mischverhältnis einer oder zweier Metriken an. Dieser Farbwert kann für den späteren Vergleich gespeichert werden.

Das Verfahren ähnelt der Bestimmung von Säuren und Basen mit Hilfe von PH-Streifen. Die Farbwerte einzelnen Klassen können abgetragen und miteinander verglichen werden.

Ausserdem ist es interessant, die Farbveränderung einer Klasse über eine bestimmte Zeit zu verfolgen, da die Programmteile sowohl vom Gesamtsystem als auch von der Zeit abhängig sind.

Das heisst, eine Klasse und damit ihr Farbwert wird sich immer verändern, ob nun etwas an der Klasse selbst (direkt) oder am System (indirekt) geändert wird.

1 Invertierter Farbverlauf mit Linien ergeben PH-Streifen

UMSETZUNG

Screendesign

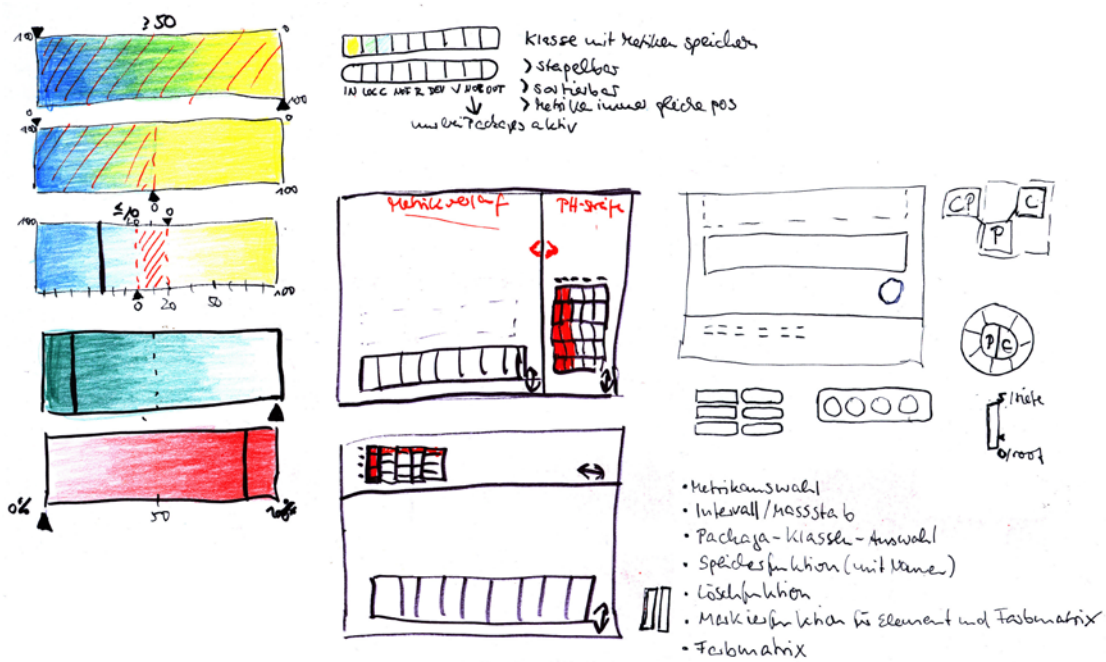
Ausgehend vom Modell ColorRamp gibt es zwei wichtige Bereiche. Einen Bereich für die Farbverläufe und einen für die PH-Streifen.

Der Bereich mit den Farbverläufen kann als Aktionsfeld verstanden werden. Hier findet die Metrikenauswahl für die Klassen, Packages und Methoden statt. Ausserdem werden die Hierarchien, Verknüpfungen und Intervallausschnitte angezeigt.

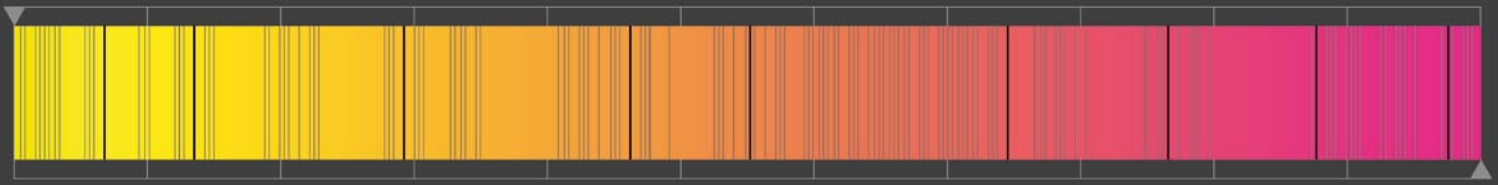
Im 2. Bereich werden die PH-Streifen abgetragen. Die Speicherung erfolgt in Stapeln. Direkt untereinander ist immer die gleiche Klasse mit den verschiedenen Mischungen sichtbar.

Die beiden Bereiche sind dunkelgrau, um den Kontrast zu den Farben zu verstärken. Eine weisse Linie bildet eine optische Grenze zwischen dem Aktionsfeld oben und der PH-Streifen-Zone unten.

1 Farbverläufe und mögliche Anordnungen der Bereiche



1 Screendesign mit Aktionsbereich und PH-Streifen-Zone



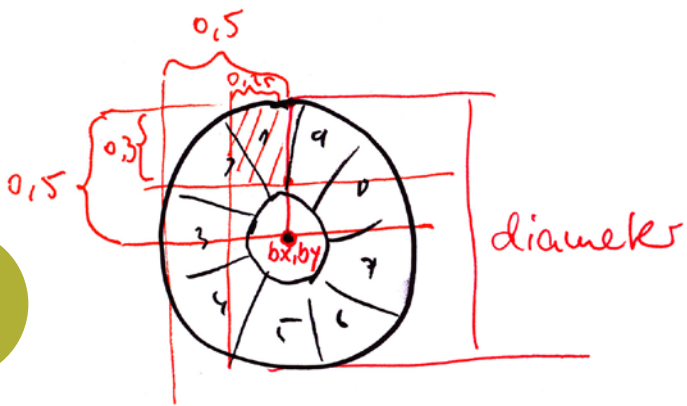
Interface

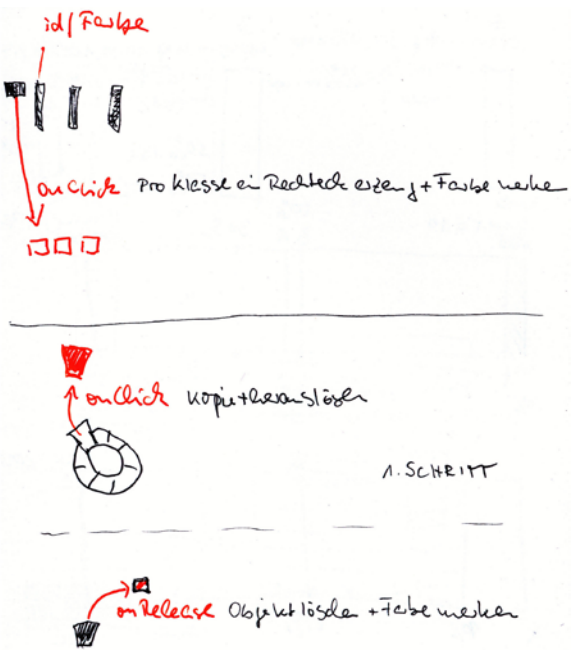
Die Auswahl der Metriken erfolgt durch ein kreisförmiges Tool, das gestalterisch und funktional an einen Farbfächer erinnert.

Mit der Maus wird eine Farbe, die eine Metrik repräsentiert, aus dem Fächer ausgewählt und in einen der beiden Balkenränder gezogen. Auf diese Weise entsteht ein Farbverlauf.

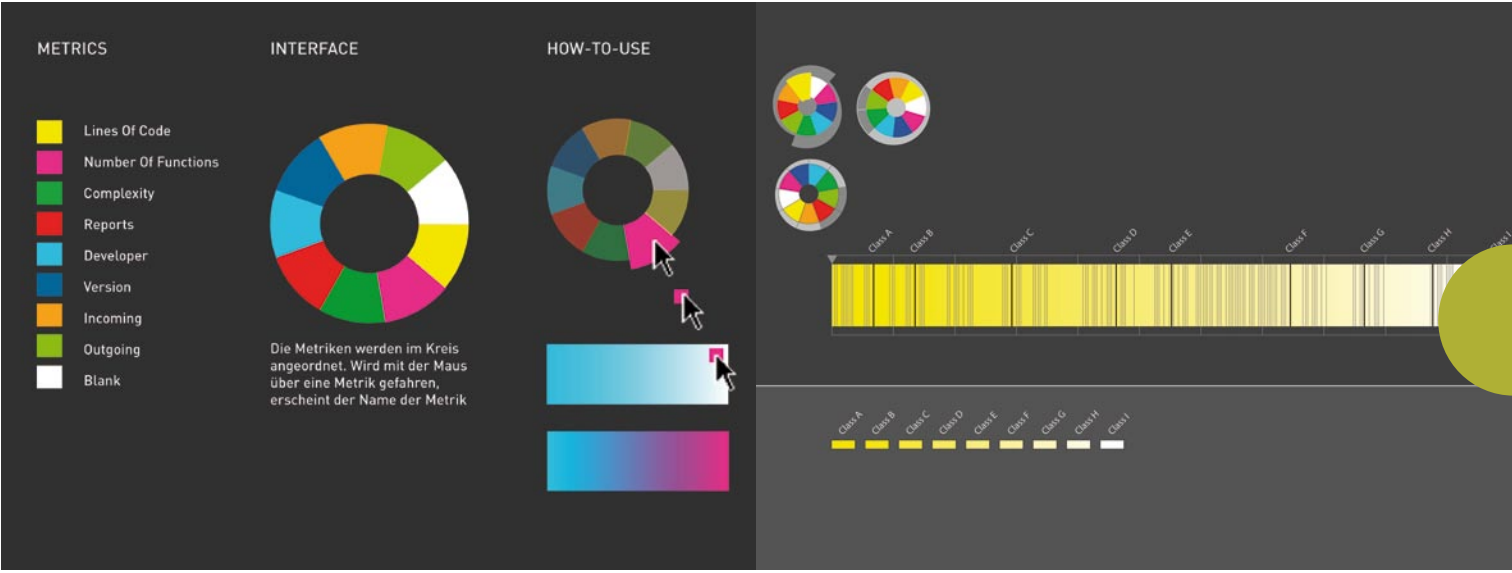
Das Tool ist frei verschiebbar und kann über die Taste C (Controller) ein- und ausgeblendet werden. Über die Taste H (Help) kann in eine Kurzanleitung der Applikation eingesehen werden.

Per Mausklick können die Klassen ausgewählt und anschliessend über einen Button abgetragen werden.


0,5
0,15
0,3
0,5
bx,by
diameter


id/Farbe
onClick Pro Klasse ein Rechteck erzeugt + Farbe merken
onClick Kopiertheranlösen
1. SCHRITT
onRelease Objekt lösche + Farbe merken

1 Mathematische Aufteilung eines Kreises
2 Programmierlogik für das Fächer-Interface
3 Kurzanleitung der Applikation
4 Auswahl des geeigneten Interfaces



Workflow

Die Applikation verhält sich in der Anwendung folgendermassen:

1. Beim Öffnen des Programms zeigt sich ein weisser Balken sowie die verfügbaren Packages in alphabetischer Reihenfolge.

2. Dem Balken wird die Metrik Lines of Code (gelb) zugewiesen. Es entsteht ein Farbverlauf von gelb nach weiss und die Packages sortieren sich dem-entsprechend neu.

3. Um Packages für die aktuelle Metrikansicht zu speichern, muss eine Auswahl getroffen werden und mit der Maus in der linken oberen Ecke der Abtrag-Button betätigt werden.

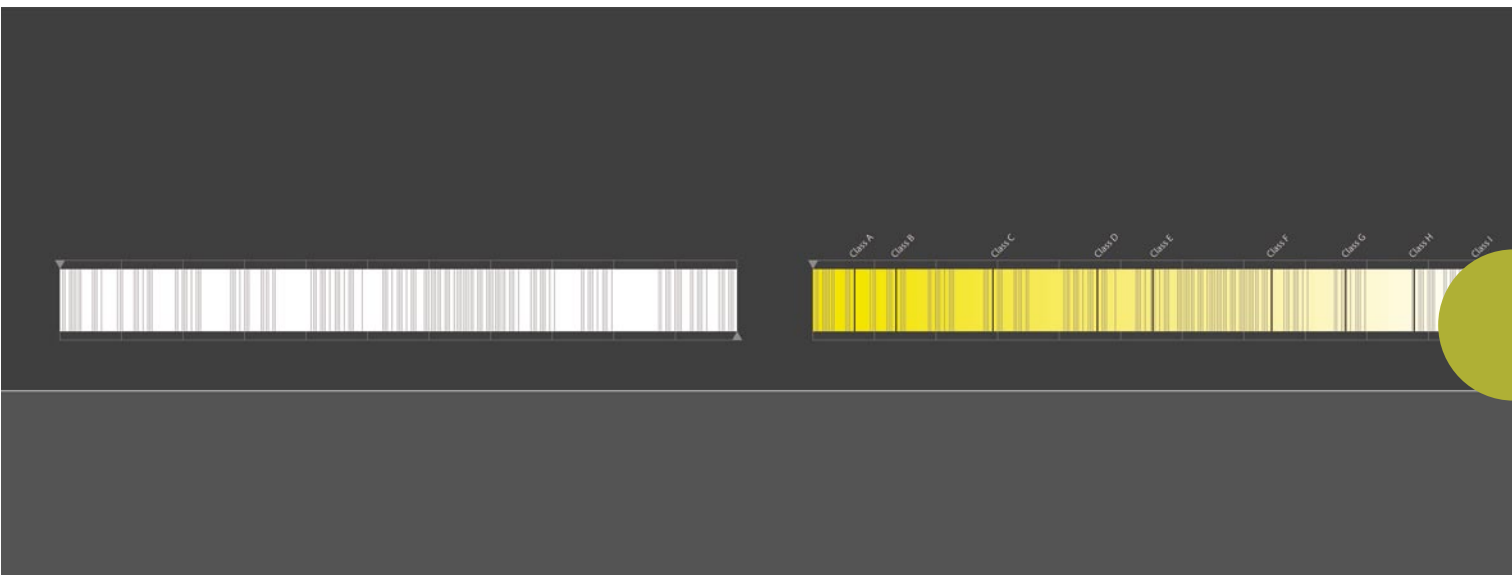
4. Die Farbwerte der ausgewählten Packages werden in den PH-Streifen-Bereich abgetragen.

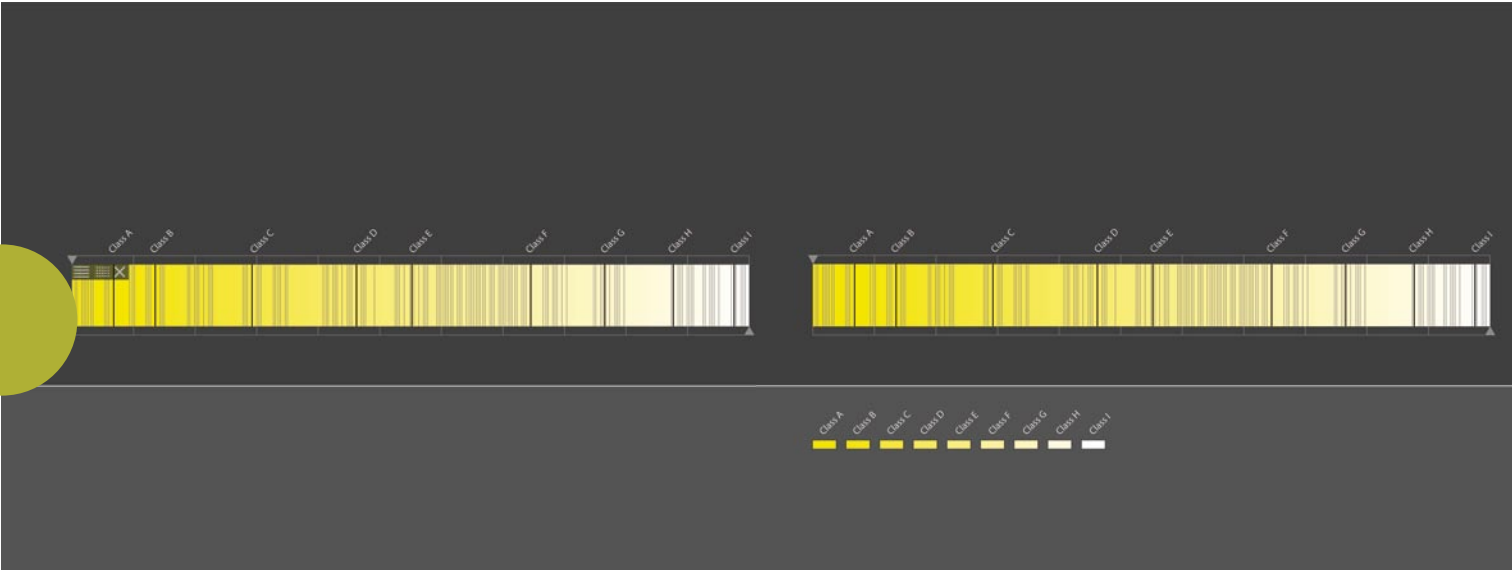
5. Der Farbverlauf kann ebenfalls über einen Speicher-Button (oben links) gesichert werden. Es erscheint eine Kopie des Farbverlaufs unterhalb.

6. Die Analyse kann nach dem beschriebenen Prinzip fortgesetzt werden.

1 Ansicht nach dem Öffnen des Programms

2 Auswahl von Klassen in einer aktiven Metrik



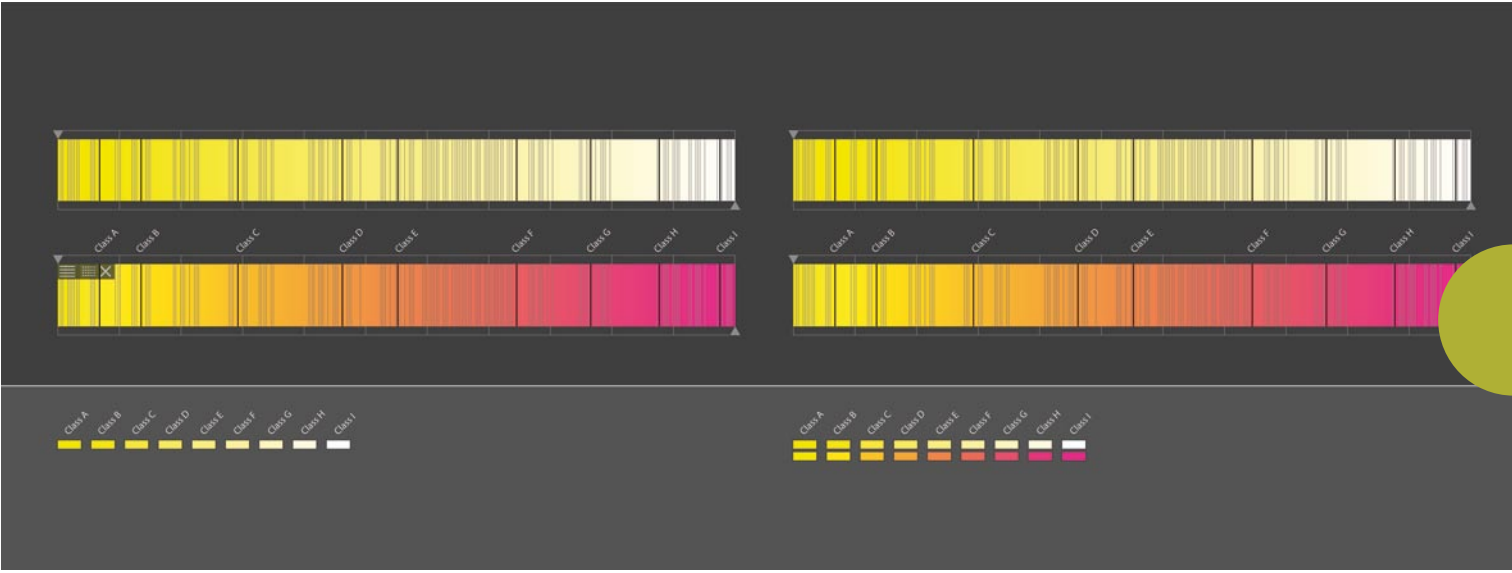

Class A
Class B
Class C
Class D
Class E
Class F
Class G
Class H
Class I
Class A
Class B
Class C
Class D
Class E
Class F
Class G
Class H
Class I
Class A
Class B
Class C
Class D
Class E
Class F
Class G
Class H
Class I

1 Anzeige des Abtrage-Buttons oben links im Farbverlauf

2 PH-Streifen werden im unteren Bereich dargestellt

3 Kopieren des aktuellen Farbverlauf, um mit Kopie weiterzuarbeiten

4 Erneues Abtragen mit einer Metrikenkombination



FAZIT

Produkt

Die Umsetzung der Applikation wurde mit Processing[3] realisiert. Die Testdaten von einer real existierenden Software haben wir von der Universität Zürich gestellt bekommen und in eine SQL[4]-Datenbank eingelesen.

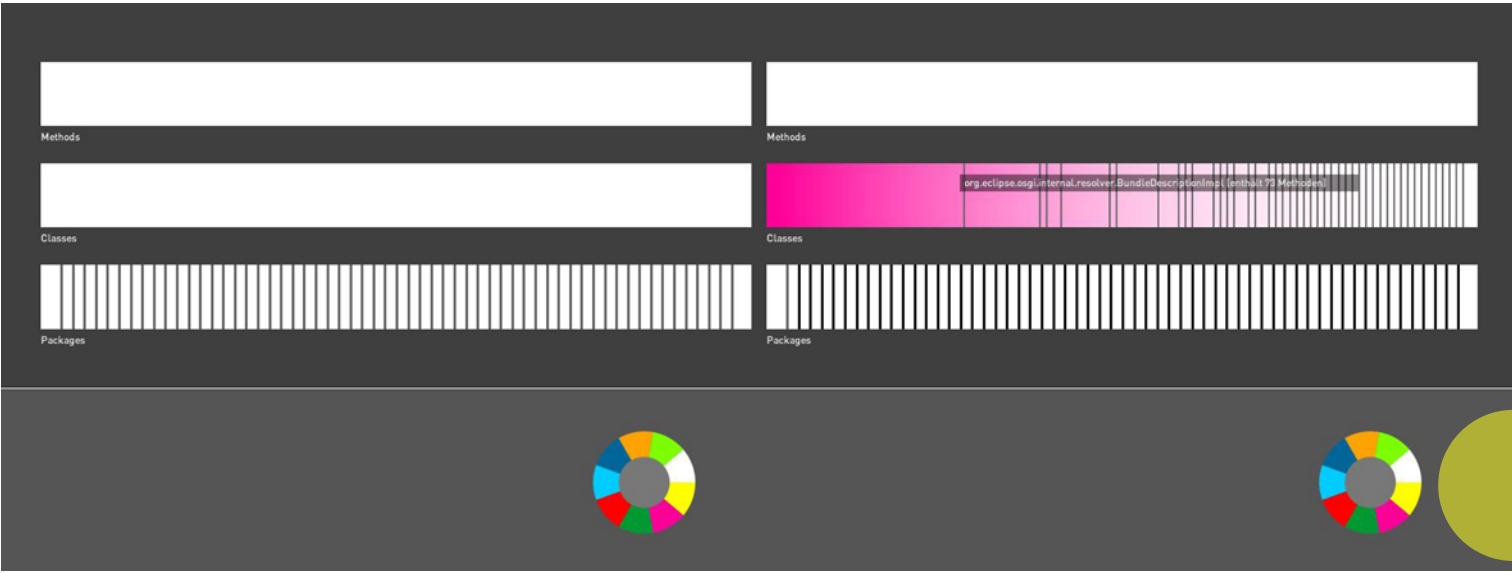
Das Interface gestaltet sich folgendermassen. Im Aktionsbereich zeigen sich beim Start der Applikation drei weisse Balken für die Package-, Klassen- und Methodeansicht. Die Packages werden als Linien alphabetisch sortiert angezeigt. Bei Auswahl eines Packages erscheinen die Klassen im entsprechenden Balken.

Über das Kreisinterface können die Metriken wie bereits beschrieben in die Balken gezogen werden. Dann sortieren sich die Klassen oder die Methoden neu. Wird mit der Maus über ein Package, eine Klasse oder eine Methode gefahren, wird der Name sichtbar. Des Weiteren können Klassen analog zu den Packages ausgewählt und somit die Methoden der Klassen abgebildet werden.

Durch eine Auswahl und die Betätigung des Abtrage-Buttons können die Farbwerte im PH-Streifen-Bereich gespeichert werden. Diese Funktion ist allerdings nur im Prototypstatus verfügbar (Klick-Dummy).

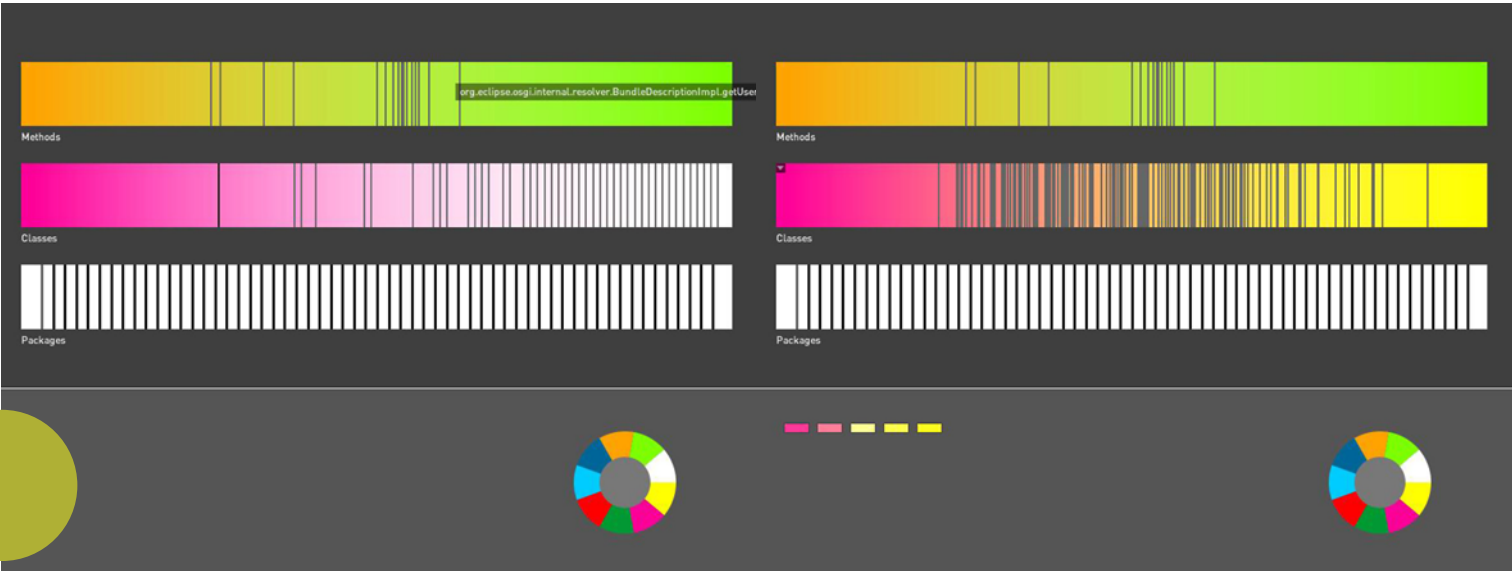
1 Ansicht nach dem Öffnen der Applikation

2 Metrik für die Klassen eines ausgewählten Packages festgelegt



1 Auswahl von Klassen und Anzeige der Methoden

2 Abtragen der ausgewählten Klassen in den PH-Streifen-Bereich



Probleme

Bei der Umsetzung der Applikation sind einige Probleme zum Vorschein gekommen, die Vorher nicht ersichtlich waren.

Die vielen Datenbankabfragen pro Sekunde führen zu einer Verzögerung der einzelnen Animationen im Programm. In diesem Fall sollte eine andere Programmiersprache gewählt werden.

Die Linien der Packages, Klassen und Methoden sind schwer mit der Maus anzuklicken. Die Namen der Programmkomponenten sind zu lang und lassen kaum Rückschlüsse für den Projektleiter zu.

Ein weiteres Problem stellt die Überlagerung der Packages, Klassen und Methoden durch die gleichen Verhältnisse dar. Das heisst, gleiche Verhältnisse können unterschiedliche Werte aufweisen (50:50=80:80=1:1). Eine erweiterte Ansicht des Farbverlaufs würde das Problem lösen.

Die Ansichten der einzelnen Verläufe und PH-Streifen können nicht gespeichert werden. Die Verknüpfungen und ausgewählten Klassen sind nicht ersichtlich. Die Packages können mit Hilfe der Metriken nicht umsortiert werden. Ausserdem sind keine Intervallansichten möglich.

EXKURS

ColorPod

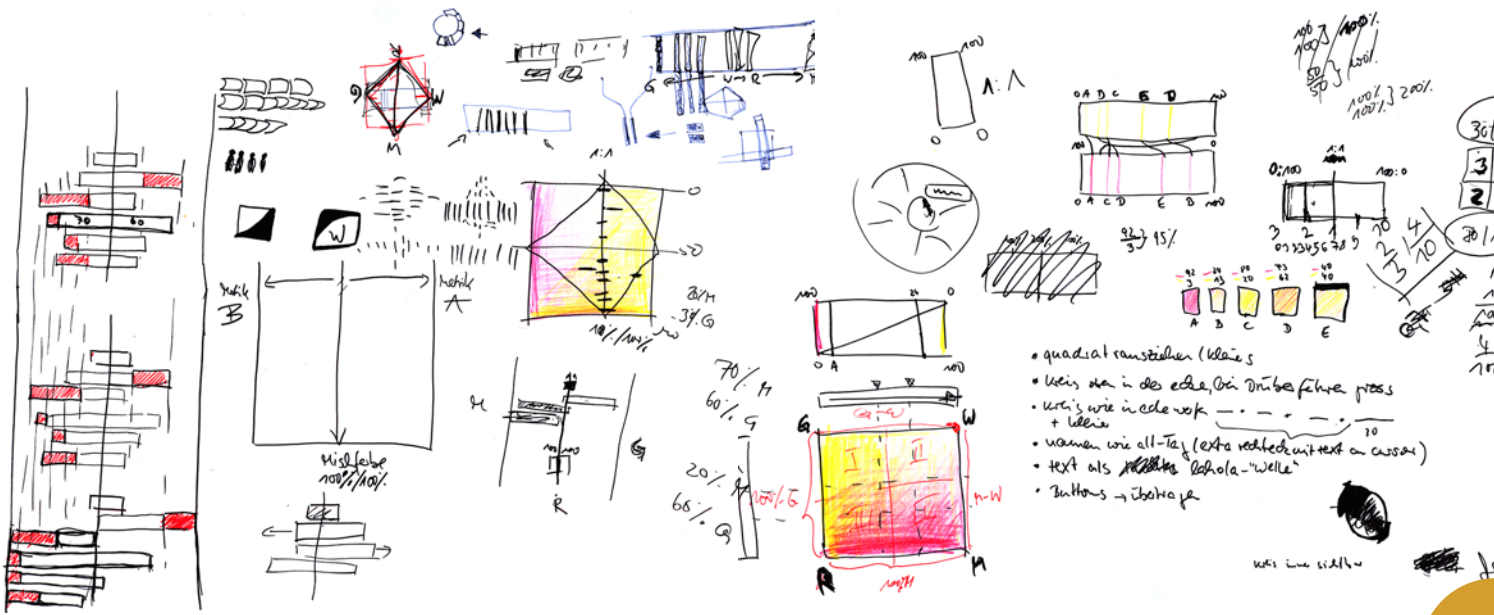
Ein Vorschlag zur Modelloptimierung ist die Darstellung des kompletten Farbbereichs. Auf diese Weise werden die konkreten Mischwerte (im Vergleich zu ColorRamp: Darstellung der Verhältnisse in Prozent) verwendet. Es gibt neben den beiden Metrikfarben zusätzlich den Verlauf nach weiss und den Verlauf zur Volltonmischung der beiden Metrikfarben. Vergleichbar ist diese Ansicht mit einer Seite des CMYK-Farbwürfels.

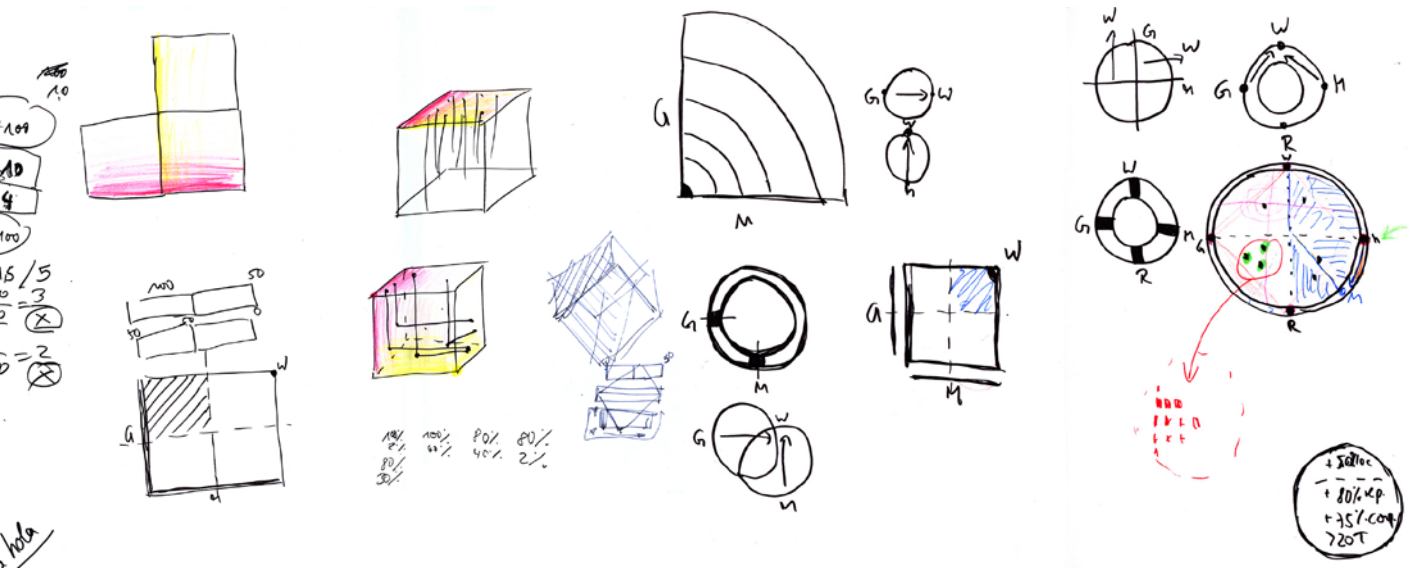
Die Darstellung für den Verlauf lässt sich auch auf einen Kreis übertragen, der sich als Interface im vorangegangenen Modell bewährt hat. Bei Auswahl der Metriken ordnen sich die Klassen innerhalb der Kreisfarbfläche in Form von Quadraten an. Mit Hilfe von Pfeilen kann das Intervall genauer betrachtet werden. Die Auswahl der Klassen im Farbkreis ist ebenfalls möglich und durch Herausziehen können diese als PH-Streifen gespeichert werden. Die PH-Streifen werden gruppiert und extra gespeichert.

Diese Funktion ermöglicht Suchmuster zu erstellen, mit denen betroffene Klassen aus dem Farbkreis herausgefiltert werden können. Der Projektleiter kann sich somit für seine Betrachtungskriterien verschiedene Schablonen anlegen.

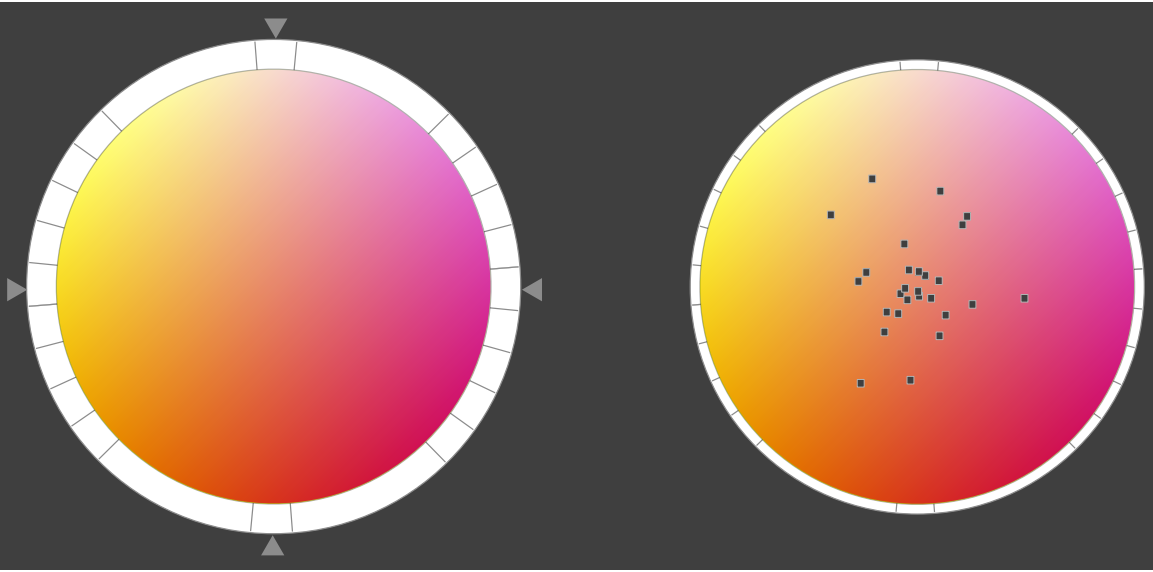
1 Transformation vom Modell ColorRamp zum Modell ColorPod

2 Funktionsweise von ColorPod

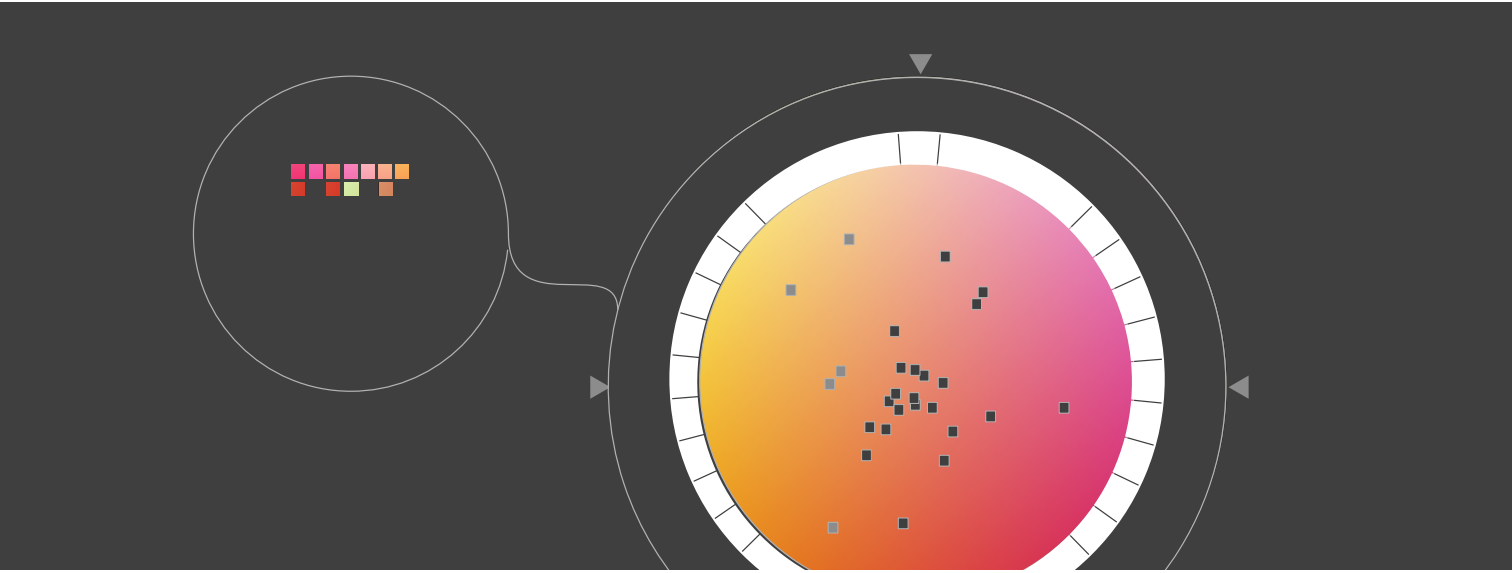



G
M
W
R
+ 80% xp
+ 75% cog
>20T

1 Verschieden Darstellungsversuche des Farbverlaufs
2 Ansicht eines möglichen Interfaces



1 Ansicht für die PH-Streifen im Modell ColorPod



Impressum

QUELLENNACHWEIS

[1] Mary-Anne Kockel, Software Visualisierung, HGK Zürich, 2007
[2] Kazuyoshi Sudo, Boro Boro, NUNO Corporation Tokio, 1997
[3] http://www.processing.org
[4] http://www.mysql.de/

LITERATURVERZEICHNIS

Ira Greenberg, Processing, Springer New York, 2007
IID Japan, Information Design, Birkhäuser Basel, 2005
Khazaeli, Cyrus Dominik, Systemisches Design, Rowohlt Hamburg, 2005
Richard Saul Wurman, Understanding, TED Inc. Newport, 2000
Jonathan Raimes, The Digital Canvas, Ilex Cambridge, 2006

TEXT & GESTALTUNG

Mary-Anne Kockel, Interaction Design, Zürcher Hochschule der Künste
Felicitas Grunenberg, Produktdesign, Bauhaus Universität Weimar