Implementierung
Graph von Ansicht

Nicolas Boltz
uweaw@student.kit.edu

Jonas Fehrenbach
urdtk@student.kit.edu

Sven Kummetz
kummetz.sven@gmail.com

Jonas Meier
Meierjonas96@web.de

Lucas Steinmann
ucemp@student.kit.edu

Inhaltsverzeichnis

1 Einleitung

Nach der Implementierung ist nun die Qualitätssicherungsphase an der Reihe, mit eine der wichtigsten Phasen eines Softwareprojektes. Denn hier wird garantiert, dass ein Programm auch sicher das tut, was es soll.
Zuallererst werden hier Programmfunktionalitäten, die aus Gründen mangelnder Zeit nicht mehr bzw. nicht komplett oder korrekt implementiert wurden, hinzugefügt und Fehler behoben. Anschließend gilt es, die in der Implementierungsphase implementierten Funktionalitäten des Programmes auf Herz und Nieren zu testen und sowohl schon bekannte, als auch mögliche Fehler zu beseitigen. Da Graph von Ansicht eine graphische Benutzerschnittstelle bietet, ist es nicht möglich, alle nur denkbaren Szenarien durch zu testen, da innerhalb dieser Benutzerschnittstelle viele Aktionen beliebig oft wiederholt werden können und diese sich gegenseitig beeinflussen.
Vielmehr wird nun darauf gesetzt mit Testfällen, die einerseits einen ordnungsgemäßen Ablauf aber auch andererseits einen möglichen Randfall oder gar einen fehlerhaften Ablauf darstellen können, das Programm zu simulieren und das Ergebnis mit dem erwarteten Wert abzugleichen. Begleitet werden diese Tests von unzähligen manuellen Durchläufen und Testen des Programmes, sowohl durch die Entwickler, als auch durch unwissende Personen. Letzteres dient vor allem zur Verbesserung der Benutzbarkeit des Programmes und zur besseren Abdeckung von Randfällen des Programmes.

2 Bugs und Reparatur

#1	**Fehlersymptom:**	Graphische Hervorhebung von FieldAccessen durch eine farbige Box im Hintergrund wird nicht angezeigt.
	Fehlergrund:	Die Struktur im Hintergrund war leicht fehlerhaft und in der Erstellung der Box gab es einen Bug.
	Fehlerbehebung:	Die Struktur im Hintergrund wurde refactored und der Bug beim erstellen des Hintergrunds wurde behoben.
#8	**Fehlersymptom:**	Kollabierte Knoten in einem FieldAccessGraph verschwinden bei Ihrer Erstellung.
	Fehlergrund:	Bei dem Layouten von MethodGraphen wurden alle FieldAccess betrachtet. Auch FAs die teilweise kollabiert wurden.
	Fehlerbehebung:	Es werden nun nur noch FAs welche vollständig ausgeklappt sind im MethodGraphLayout betrachtet
#9	**Fehlersymptom:**	Beim Öffnen eines Graphens über die Kommandozeile wurde nicht das Default Layout des ausgewählten Workspace übernommen.
	Fehlergrund:	Es wurde immer der LayoutSelectionDialog aufgerufen, obwohl ein workspace angegeben wurde.
	Fehlerbehebung:	Aufruf gelöscht und das Default Layout aus dem Workspace angewendet.
#10	**Fehlersymptom:**	Keine Informationen oder Warnung über falsch eingegebene Kommandozeilenparameter.
	Fehlergrund:	Keine explizite Überprüfung der Eingaben. Es wurde angenommen, dass der User alle Eingaben richtig macht.
	Fehlerbehebung:	Hinzufügen von expliziten Überprüfungen und Anzeigen von Fehlermeldungen mit hilfreicher Information.
#11	**Fehlersymptom:**	Falls das Joana-Workspace für eine generische GraphML-Datei ausgewählt wird, kommt keine Warnung und es passiert nichts.
	Fehlergrund:	Keine Überprüfung ob ein GraphBuilder für diesen Graphentyp existiert.
	Fehlerbehebung:	Überprüfung ob GraphBuilder null ist.
#12	**Fehlersymptom:**	Der Graph springt beim verschieben mit der Maus zu Beginn ein Stück nach unten.
	Fehlergrund:	Die Berechnung der Mausposition wurde die Position im Fenster genutzt.
	Fehlerbehebung:	Nutzen der Mausposition des Mausevents die das Ziehen einleitet.
#17	**Fehlersymptom:**	Das Filtern von Knoten bestimmten Types (z.B. EXPR) in MethodGraphLayout führt zu einer NullPointerException
	Fehlergrund:	Für das Layouten der FAs und des ganzen MethodGraphen wurde der selbe Layoutalgorithmus, mit den selben Constraints gewählt.

	Fehlerbehebung:	Separate Instanzen des SugiyamaAlgorithm für FAs und MethodGraph.
#22	**Fehlersymptom:**	Die alte StrukturView kann nach Import einer neuen Datei nicht zurückgebracht werden.
	Fehlergrund:	Alte Graphen wurden nach einem neuen Import nicht geschlossen, obwohl die Anwendung nur eine einzelne Datei unterstützt.
	Fehlerbehebung:	Bereits offene Graphen werden nach erfolgreichem Import geschlossen.
#23	**Fehlersymptom:**	Ähnlich wie in Bug 22 wird versucht nach einem Import über den noch offenen alten Callgraphen einen Methodengraphen zu öffnen.
	Fehlergrund:	Siehe Bug 22
	Fehlerbehebung:	Siehe Bug 22
#24	**Fehlersymptom:**	Programm stürzt ab, falls der eingegebene Dateipfad über den Kommandozeilenparameter –in keinen Punkt enthält.
	Fehlergrund:	Falls Dateipfad keinen Punkt erhält gibt Java Methode lastIndexOf('.') als Ergebnis -1 zurück. Dieses Ergebniss wird bei substring() als ungültiger Index verwendet was zum Abstürz führt.
	Fehlerbehebung:	Vor Aufruf der Methode substring() wird überprüft ob der Dateipfad einen Punkt enthält. Falls nicht wird eine Fehlermeldung ausgegeben.
#27	**Fehlersymptom:**	Text in exportierter SVG-Datei ist größer als die Textboxen der Knoten.
	Fehlergrund:	Default Textgröße in SVG ist größer als Default Textgröße innerhalb der GUI. Die Größe der Textboxen richten sich jedoch nach der Textgröße der GUI.
	Fehlerbehebung:	Hinzufügen einer expliziten Textgröße zur SVG-Datei, welche der Textgröße der GUI entspricht.
#28	**Fehlersymptom:**	Kanten laufen ggf. aus der Box des Feldzugriffes hinaus.
	Fehlergrund:	Kantenläufe wurden bei der Berechnung der Größe der Box nicht berücksichtigt.
	Fehlerbehebung:	Kantenläufe werden nun bei der Berechnung der Größe der Box berücksichtigt.
#30	**Fehlersymptom:**	Kanten laufen durch Knoten durch und manchmal von unten rein und verbinden sich dann oben auf dem Knoten.
	Fehlergrund:	Der Kantenzeichner ging davon aus, dass Kanten immer von oben nach unten verlaufen, also der source-Knoten sich immer oberhalb des target-Knoten befindet. Durch relative- und absolute-layer-constraints kann es aber vorkommen, dass miteinander verbundene Knoten auf dem selben layer liegen, weswegen die Kanten teilweise unvorhersehbar durch Knoten gezeichnet wurden.
	Fehlerbehebung:	Kanten zwischen Knoten auf dem selben layer werden nun gesondert gezeichnet, und zwar unterhalb des layers.
#31	**Fehlersymptom:**	Beim Zoomen wird nicht vom Mauszeiger weg oder zum Mauszeiger hin gezoomt. Der Fokus und Sichtbereich springt herum.
	Fehlergrund:	Die GraphView wurde durch das Zoomen vergrößert, dadurch gab es von den äußeren Oberflächenelementen, die um die GraphView gelegt waren, Interferenzen, die das Springen und unfokussierte Zoomen ausgelöst haben.

Fehlerbehebung: Die Ansicht des Graphen besteht nun aus 4 ineinander verschachtelten Panes, wobei das innerste die GraphView und das äußerste ein ScrollPane ist. Vergrößert wird beim Zoomen nun das Pane welches direkt um der GraphView liegt. Das Pane darum passt sich per Listener immer an die Größe der beiden inneren Panes an. Dadurch entstehen keinen Interferenzen mehr und anderes ungewolltes Verhalten tritt auch nicht mehr auf.

#40 **Fehlersymptom:** Die Richtung mancher Kanten zwischen zwei Knoten änderte sich manchmal nach neuem importieren oder neuem layouten des Graphen.

Fehlergrund: Im Sugiyama werden notwendigerweise zuallererst Kanten gedreht, die Zykel im Graph verursachen. Der Kantenzeichner im letzten Schritt des Sugiyama hat nun nur die gedrehten Kanten, die sich über ein layer erstrecken in ihrer Richtung angepasst. Kanten,d ie über mehrere layer gingen wurden ignoriert, da diese in einem anderen Set lagen.

Fehlerbehebung: Der Kantenzeichner passt nun durch Berücksichtigung des Sets der sich über mehrere Layer erstreckenden Kanten, von diesen ehemals gedrehte Kanten in ihrer Richtung an.