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 von Graph von Ansicht ist nun die Qualitätssicherung an der Reihe. Ziel ist es, Fehler im Programm durch verschiedene Vorgehensweisen zu finden und diese anschließend zu beheben. Ziel ist es auch, durch sowohl zufällige als auch wohl überlegte Testfälle und der Überprüfung dieser auf Korrektheit, die richtige Funktionsweise des Programmes ein Stück weit mehr garantieren zu können. Je mehr Testfälle funktionieren, desto mehr mögliche Fehlerquellen können wahrscheinlicher ausgeschlossen werden.
Da das Programm nach der Implementierung nicht komplett fertig war und es noch einige Stellen gab, an denen aus Zeitgründen nicht weiter gearbeitet werden konnte und diese somit auch teilweise nicht vollständig funktionsfähig waren, wurden diese zuallererst fertig implementiert. Noch zu erledigen waren in diesem Projekt: das korrekte Darstellen von Feldzugriffen (eine bestimmte Teilmenge von Knoten und Kanten des Graphen), dessen Kanten nicht richtig gezeichnet wurden, zeitliche Zusicherungen, welche versprochen wurden (1000 Knoten + 4000 Kanten unter 2 Minuten, 500 Knoten + 2000 Kanten unter einer Minute), ein Programmabsturz bei falschen Kommandozeilenargumenten soll verhindert werden.
Graph von Ansicht ist ein Graphviewer, seine Hauptaufgabe besteht darin, gegebene Graphen zu layouten. Deswegen liegt das Hauptaugenmerk des Testens auf dem Layoutalgorithmus. Die korrekte Funktionsweise des Algorithmus kann aufgrund der hohen Anzahl an möglichen Eingaben nicht garantiert werden, auch bietet das Programm durch eine eingebaute Filterfunktion für Knoten und Kanten eine Vielzahl an Möglichkeiten den Graphen nochmals zu modifizieren bevor der Algorithmus erneut ausgeführt werden kann. Deswegen wurden viele Tests erstellt, die vor allem Randfälle (zum Beispiel ein Graph ohne Knoten oder Kanten, mit einem Selbstzyklus (Kante in sich selbst), etc.) abdecken. Ebenso wird die Benutzbarkeit des Programmes getestet über das Testen des Programmes durch Personen, die sowohl von dem Thema als auch der Funktionsweise des Programmes unwissend sind. Auffälligkeiten hinsichtlich der Handhabung und Navigation in dem Programm werden angepasst, sodass sich auch fachfremde Personen in der Menüführung zurecht finden.

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.
#2	**Fehlersymptom:**	Einige Kanten werden übereinanderliegend gezeichnet.
	Fehlergrund:	Dummy Vertices werden übereinander platziert, da zum einen der Code zum überprüfen von Überlagerung von Kanten fehlerhaft ist und da die Kanten die mehrere Ebenen überdecken nicht immer genügend verschoben werden. Außerdem gibt es Mehrfachzuweisungen von Knoten zu zu begradigenden Abschnitten.
	Fehlerbehebung:	Die fehlerhafte Code zur Überprüfung von Kantenschnitten im Vertex Positioner wurde korrigiert, indem ein fehlerhafter Vergleich berichtigt wurde. Er kann nun auch mit 1 breiten Kanten arbeiten. Die Verschiebung der Kanten wird nun so oft ausgeführt, bis sich Kanten nicht mehr überdecken. Es wurde außerdem ein schwerwiegender Fehler der zu der Mehrfachzuweisung führte behoben. Hier hat ein rekursiver Teilalgorithmus ergebnisse verdoppelt.
#3	**Fehlersymptom:**	Kanten, die von außen in einen FieldAccess hinein oder von einem FieldAccess nach außen gehen, wurden nur bis an die Box des FieldAccesses dran gezeichnet.
	Fehlergrund:	
	Fehlerbehebung:	
#7	**Fehlersymptom:**	Kanten, welche keine Ebene überspannen werden nie begradigt
	Fehlergrund:	Diese Kanten werden beim suchen nach Segmenten(Kantenketten, die im VertexPositioner später begradigt werden) nicht beachtet
	Fehlerbehebung:	Diese Kanten werden nun beim starten des VertexPositioner ebenfalls hinzugefügt. Im Zuge dieser Korrektur wurde es außerdem ermöglicht die Endknoten von Kanten, welche eine oder mehr Ebenen überspringen, zu den aus ihnen enstehenden Segmenten hinzuzufügen.
#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.
#13	**Fehlersymptom:**	Es gibt keine Option zur Einstellung ob eine Abbruchgrenze beim Kreuzungsminimieren in Abhängigkeit von den vorhandenen Kreuzungen gewählt werden soll.
	Fehlergrund:	Diese Option wird in den CrossMinimizer Einstellungen nicht bereitgestellt
	Fehlerbehebung:	Die Option wurde als Checkbox hinzugefügt und wird während der Kreuzungsminimierung beachtet.
#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.
#25	**Fehlersymptom:**	Es wird ein Fehler geworfen, wenn man die ENTR Knoten filtert.
	Fehlergrund:	Der VertexPositioner benötigt mindestens ein Vertex pro Graph.
	Fehlerbehebung:	Der VertexPositioner überprüft nun am Anfang, ob der Graph leer ist und läuft in diesem Fall nicht durch.
#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, die 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.
#45	**Fehlersymptom:**	FieldAccess Boxen werden in der exportierten SVG-Datei nicht angezeigt.
	Fehlergrund:	Die FieldAccess Boxen werden nicht zum serialisierten Graphen hinzugefügt.
	Fehlerbehebung:	Die FieldAccess Boxen werden nun als zusätzliche Knoten im serialisierten Graphen gespeichert.
#46	**Fehlersymptom:**	Exportierte SVG-Datei wird in manchen SVG-Viewer nicht komplett angezeigt.
	Fehlergrund:	Einige SVG-Viewer können die Standard Größe 100% einer SVG-Datei nicht korrekt interpretieren.
	Fehlerbehebung:	Die komplette Größe eines Graphens wird berechnet und als feste Werte in die SVG-Datei geschrieben.

3 Durchgeführte Tests

3.1 Globale Testfälle aus dem Pflichtenheft

/T010/: Import und Darstellung von einem JOANA Graphen
Anmerkungen: -

/T020/: Öffnen eines JOANA-Methodengraphen
Anmerkungen: -

/T030/: Selektieren mehrerer Knoten und Kanten
Anmerkungen: Da es sich während der Implementierung als wenig nützlich erwiesen hat, wurde das selektieren von Kanten nicht implementiert. Deswegen ist es in diesem Test auch nicht möglich, Kanten zu selektieren.
Es werden außerdem bei mehreren ausgewählten Knoten weder eine Statistik noch eine Information zu diesen angezeigt. Lediglich von einem einzigen Knoten werden Informationen angezeigt.

/T040/: Navigation
Anmerkungen: Das Verschieben des Sichtfeldes geschieht nach der Implementierung nicht per Mittelmaus-klick halten und ziehen, sondern über Strg + Rechts-klick gedrückt halten und ziehen mit der Maus.

/T050/: Constraint zu Knoten eines geladenen Graphen hinzufügen
Anmerkungen: Durch Verschieben des Kriteriums der manuell hinzufügbaren Constraints zu Gruppen in die Wunschkriterien (siehe Pflichtenheft: 4.2 Wunschfunktionen, /FA300/ Constraints hinzufügen), wurde dies in der Implementierungsphase nicht hinzugefügt. Es ist daher hier lediglich möglich, Gruppen zu erstellen (bis einschließlich Punkt 4 des Tests).

/T060/: Filtern von Kanten
Anmerkungen: Die Menüführung zum Erreichen der Filter von Kanten ist "Other->Edit Filter", danach klick auf Edges und dann hinzufügen eines Haken zum filtern dieser Kante, anstatt Ëditieren->Filter anpassenmit anschließendem klick auf Joana und entfernen von Haken.

/T070/: Export von einem geladenen JOANA-Graphen als SVG
Anmerkungen: Die Menüführung zum exportieren des geladenen JOANA-Graphen gechieht über File->Export", anstatt "Datei->Export->SVG". Da nur ein Export-Format zur Auswahl steht, wurde auf den letzten Schritt der geplanten Menüführung verzichtet.

3.2 JUnit Tests

3.3 Hallway Usability Testing

Hier werden unwissende Probanden vor das Programm gesetzt, ihnen die Funktionsweise und Möglichkeiten der Funktionen des Programmes erläutert, sie dann teste lassen und Auffälligkeiten notiert. Auffälligkeiten können sowohl in der Benutzbarkeit des Programmes, also wie die Probanden zurechtkommen, vorkommen, aber auch in der Funktionsweise des Programmen, indem man selbst oder die Probanden mögliche unerwünschte Nebeneffekte oder Fehler im Programm sehen.

Proband 1: Datum 16.08.2016, commit hash: 6b06dd0
Auffälligkeiten: Es traten keine unerwarteten Fehler im Programm auf.
Es wurde vor allem mit collapse gearbeitet, den Menüpunkten zm Filtern, neu importieren, exportieren wurde kaum Aufmerksamkeit geschenkt.

Proband 2: Datum 16.08.2016, commit hash: 6b06dd0
Auffälligkeiten: Auch hier traten keine unerwarteten Fehler im Programm auf.
Verwirrend war zum einen, dass man durch Doppelklick auf einen Methodengraphen in der Strukturansicht diesen öffnen kann, jedoch durch einen Doppelklick auf einen Knoten im Callgraphen diesen dazugehörigen Methodengraphen nicht öffnen konnte, nur mit Rechtsklick->open".
Zum anderen war das Graph verschieben mithilfe der Tastenkombination Strg + Recktsklick ungewöhnlich.
Ebenso unschön wurde das Filtern von Kanten über das checken der checkboxes gefunden, sowie die Unwissenheit über momentan vorhandene Knoten und Kanten im Graphen. Man kann auch Knoten und Kanten filtern, die gar nicht im Graph vorhanden sind.

3.4 Manuelle Tests

3.5 Andere durchgeführte Tests

3.5.1 Überdeckungstests

Es wurden Überdeckungstests sowohl über JUnit-Tests der einzelnen Projekte, als auch über alle Projekte während eines Programmlaufes mithilfe von EclEmma durchgeführt.