

Ehemalige Veröffentlichungsscheckliste (angepasst):

1. Vespucci Projekt vorbereiten:

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Zu testendes Vespucci-Projekt in Eclipse importieren.	O	O
Den Eintrag „de.tud.cs.st.vespucci/model/vespucci.genmodel“ öffnen und auf dem obersten Knoten im Rechtsklick-Menü „Generate all“ wählen.	O	O
Den Eintrag „de.tud.cs.st.vespucci/model/vespucci.gmfgen“ öffnen und auf dem obersten Knoten im Rechtsklick-Menü „Generate diagram code“ wählen (Grund dieser Maßnahme ist, dass kein Code im Projekt existiert, der nach dem generieren modifiziert wurde und kein „generate not“ Statement hat). ACHTUNG: Im Projekt „de.tud.cs.st.vespucci.diagram“ wird ein leerer „icons“-Ordner angelegt. Dieser ist vor dem Start zu löschen.	O	O
Anmerkungen:		

2. Erste Schritte zum GUI-Test

2.1 Vorhandenes JAVA-Projekt importieren

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das Projekt „GUI_Test_Projekt“ in Eclipse importieren.	O	O
Die Vespucci-Diagramme (*.sad) können geöffnet werden.	O	O

2.2 Vespucci-Diagramme erzeugen

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das neue Vespucci-Diagramm „ReleaseDiagram.sad“ kann per Rechtsklick auf den Ordner „Architecture“ per „New → Other → Software Architecture Constraints Diagram“ erzeugt werden.	O	O
Das neue Vespucci-Diagramm „RäleüseD&\$2iagröm.sad“ kann über das Eclipse-Menü „File → New → Other → Software Architecture Constraints Diagram“ erzeugt werden.	O	O
Alle geöffneten Vespucci-Diagramme schließen.	O	O

3. GUI-Tests

3.1 Sonderzeichenprüfung

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das Diagramm „RäleüseD&\$2iagröm.sad“ öffnen.	O	O
Über den Paletten-Menüpunkt <i>Ensemble</i> ein Ensemble auf der Hauptebene(weiße Diagrammfläche) erzeugen und mit dem Namen „View\s“ versehen.	O	O
Über den Paletten-Menüpunkt <i>Ensemble</i> ein Ensemble „Cört\$%!älßür“ auf der Hauptebene erzeugen.	O	O
Im Ensemble „View\s“ über den Paletten-Menüpunkt <i>Ensemble</i> zwei Ensembles „Flashcärd“ und „Utilities“ erzeugen.	O	O
<i>Outgoing-Dependency</i> aus der Palette wählen und von „View\s“ nach „Cört\$%!älßür“ ziehen.	O	O
Ensemble „Cört\$%!älßür“ wählen und über den angezeigten Pfeil, der vom Ensemble weg zeigt mit gedrückter Maustaste selektieren und auf dem Ensemble „Flashcärd“ loslassen. Im angezeigten Popup Menüpunkt die <i>Outgoing-Dependency</i> auswählen.	O	O
Das Ensemble „Flashcärd“ in das Ensemble „Cört\$%!älßür“ verschieben.	O	O
Vespucci-Diagramm „RäleüseD&\$2iagröm.sad“ speichern und schließen.	O	O
Vespucci-Diagramm „RäleüseD&\$2iagröm.sad“ öffnen.	O	O
Vespucci-Diagramm „RäleüseD&\$2iagröm.sad“ prüfen ob alle erstellten Ensembles noch mit den eingegebenen Namen und allen Abhängigkeiten vorhanden sind.	O	O
Vespucci Diagramm „RäleüseD&\$2iagröm.sad“ speichern und schließen.	O	O

3.2 Vespucci-Diagramme: Erstellung und Bearbeitung

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Vespucci-Diagramm „ReleaseDiagram.sad“ öffnen.	O	O
Über den Paletten-Menüpunkt <i>Ensemble</i> ein Ensemble auf der Hauptebene(weiße Diagrammfläche) erzeugen und mit dem Namen „Views“ versehen.	O	O
Prüfen, dass im Outlineview genau die Elemente angezeigt werden, die auch im Vespucci-Diagramm enthalten sind. Weiterhin muss eine eventuell im Vespucci-Diagramm vorhandene Hierarchie auch im Outlineview erkennbar sein. Diese Prüfung ist nach jedem Testfall durchzuführen.	O	O
Das Package „de.tud.cs.se.flashcards.model“ via Drag and Drop als neues Ensemble einfügen. Das neuentstandene Ensemble ist so selektiert,	O	O

dass der Ensemble Name direkt editierbar ist. Der Name des Ensembles soll in „Models“ geändert werden.		
Die Klasse „Store.java“ im Package „de.tud.cs.se.flashcards.persistence“ via Drag and Drop als neues Ensemble einfügen. Das neuentstandene Ensemble ist so selektiert, dass der Ensemble Name direkt editierbar ist. Der Name des Ensembles soll in „Store“ geändert werden.	O	O
Über den Paletten-Menüpunkt <i>Ensemble</i> ein Ensemble „Controller“ auf der Hauptebene erzeugen.	O	O
Im Ensemble „Views“ über den Paletten-Menüpunkt <i>Ensemble</i> zwei Ensembles „Flashcard“ und „Utilities“ erzeugen.	O	O
<i>Outgoing-Dependency</i> aus der Palette wählen und von „Views“ nach „Models“ ziehen.	O	O
Ensemble „Controller“ wählen und über den angezeigten Pfeil, der vom Ensemble weg zeigt mit gedrückter Maustaste selektieren und auf dem Ensemble „Models“ loslassen. Im angezeigten Popup Menüpunkt die <i>Outgoing-Dependency</i> auswählen.	O	O
Über den Paletten-Menüpunkt <i>Empty Ensemble</i> ein Empty Ensemble mit dem Namen „BorderFactory“ auf der Hauptebene erzeugen.	O	O
Das Empty Ensemble „BorderFactory“ in das Ensemble „Views“ verschieben.	O	O
<i>Outgoing-Dependency</i> aus der Palette wählen und von „Models“ nach „BorderFactory“ ziehen.	O	O
<i>Expected-Dependency</i> aus der Palette wählen und von „Store“ nach „Controller“ ziehen.	O	O
<i>Expected-Dependency</i> aus der Palette wählen und von „Store“ nach „Views“ ziehen.	O	O
Ensemble „Store“ selektieren. Es erscheinen zwei Pfeile. Den Pfeil der vom Ensemble weg zeigt mit gedrückter Maustaste selektieren und über einer leeren Stelle loslassen. Über den Popup Menüpunkt <i>Create Expected To → New Element: Ensemble</i> ein neues Ensemble mit dem Namen „Dummy“ erstellen.	O	O
Prüfe, dass der Outline View die Elemente des Workspace korrekt anzeigt.	O	O
Alle „Flash*.java“-Klassen im Package „de.tud.cs.se.flashcards.ui“ via Drag and Drop gemeinsam auf das Ensemble „Flashcard“ ziehen.	O	O
Klasse „Utilities.java“ im Package „de.tud.cs.se.flashcards.ui“ via Drag and Drop auf das Empty Ensemble „BorderFactory“ ziehen. Drop ist nicht möglich und es wird das entsprechende Maus-Icon angezeigt.	O	O
Klasse „Utilities.java“ im Package „de.tud.cs.se.flashcards.ui“ via Drag and Drop auf Ensemble „Utilities“ ziehen.	O	O
Empty Ensemble „BorderFactory“ wählen und über einen der angezeigten Pfeil eine Note Attachment erstellen.	O	O
<i>In- and Out-Dependency</i> aus der Palette wählen und von „Utilities“	O	O

nach „Flashcard“ ziehen.		
<i>In- and Out-Dependency</i> aus der Palette wählen und von „Views“ nach „Flashcard“ ziehen.	O	O
<i>Expected Dependency</i> aus der Palette wählen und von „Utilities“ nach „Controller“ ziehen.	O	O
Das Empty Ensemble „BorderFactory“ in „BorderFactories“ umbenennen.	O	O
Vespucci-Diagramm „ReleaseDiagram.sad“ speichern und schließen.	O	O
Vespucci-Diagramm „ReleaseDiagram.sad“ öffnen.	O	O
Vespucci Diagramm „ReleaseDiagram.sad“ speichern und schließen.	O	O
Sicherstellen, dass bei den oben durchgeführten Drop-Operationen kein Move (Datei verschwindet aus dem Projekt-Explorer) durchgeführt wurde.	O	O

3.3 Anstoßen der Prologgenerierung

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Eintrag zum Generieren (<i>Vespucci: Generate Architectural Model</i>) findet sich nur im Kontextmenü von „*.sad“-Dateien.	O	O
Aktualisieren des Package-Explorers (Nach dem Generieren einer neuen „*.pl“-Datei wird diese direkt im Package-Explorer angezeigt).	O	O

3.4 Verhalten der Rote-Linien-Erzeugung überprüfen

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das Diagramm „RoteLinien.sad“ öffnen	O	O
Jede Kombination von eingeklappt und nicht eingeklappt der Ensembles ausprobieren. Die Dependencies werden immer richtig angezeigt.	O	O
Alle Ensembles aufklappen und dann das Ensemble „Layer1“ einklappen.	O	O
Rote Linien von „Ebene2“ nach „Layer1“ löschen → Die dazugehörigen Abhängigkeiten bleiben auch nach Expansion von „Layer1“ gelöscht.	O	O
Alle Ensembles aufklappen und bei „Ebene3“ ein „Note Attachment“ einfügen und in dieses einen beliebigen Text schreiben. Wenn das „Ebene2“ eingeklappt wird, müssen alle Verbindungen von und zu „Ebene2“ weiterhin angezeigt werden.	O	O
Die Schritte alle mittels <i>undo</i> rückgängig machen und per <i>redo</i> wiederherstellen.	O	O

3.5 Query-Tab im Properties View

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das Vespucci-Diagramm „QueryTab.sad“ öffnen und den Properties View selektieren.	O	O
Alle enthaltenen Anweisungen befolgen und erfolgreich durchführen.	O	O
Anmerkungen:		

3.6 Ensemble Verhalten beim Verschieben

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das Diagramm „EnsembleHerausziehenTest.sad“ öffnen.	O	O
Alle enthaltenen Anweisungen befolgen und erfolgreich durchführen.	O	O
Anmerkungen:		
Das Diagramm „EnsembleHereinziehenTest.sad“ öffnen.	O	O
Alle enthaltenen Anweisungen befolgen und erfolgreich durchführen.	O	O
Anmerkungen:		

3.7 Outline View

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das Diagramm „OutlineView.sad“ öffnen.	O	O
Das Outline View muss in komplett ausgeklapptem Zustand so aussehen:	O	O

<p>Sämtliche Elemente im Outlineview nacheinander selektieren. Es muss das entsprechende Element im Vespucchi-Diagramm selektiert werden. Außerdem muss das im Outlineview selektierte Element selektiert bleiben.</p>	O	O
<p>Sämtliche Elemente im Vespucchi-Diagramm nacheinander selektieren. Es muss das dazugehörige Element im Outlineview selektiert werden. Bei Dependencies muss nur einer der Dependency Pfeile im Outlineview selektiert werden.</p>	O	O
<p>Den Namen des Ensembles „Kind“ in „Sohn“ umbenennen. Nach der Namensänderung muss diese direkt im Outlineview zu sehen sein.</p>	O	O
<p>Im Ensemble „Vater“ ein Ensemble mit dem Namen „Tochter“ erstellen. Direkt nach der Erzeugung des Ensembles muss dieses auch im Outlineview zu sehen sein. Nach der Namensgebung „Tochter“ muss dies auch direkt im Outlineview zu sehen sein.</p>	O	O
<p>Ein Ensemble auf der Hauptebene erstellen. Diese muss direkt nach der Erstellung im Outlineview zu sehen sein.</p>	O	O
<p>Elemente im Vespucchi-Diagramm in folgender Reihenfolge löschen:</p> <ol style="list-style-type: none"> 1. Ensemble „Solo“ 2. Ensemble „Sohn“ 3. Ensemble „Vater“ 4. Empty Ensemble „empty“ 5. Text „Textfeld auf der Hauptebene“ 6. Note „Text zum Kind Ensemble“ <p>Nach jeder Lösch-Operation, dürfen die direkt und evtl. automatisch mitgelöschten Elemente nicht mehr im Outlineview angezeigt werden.</p>	O	O
<p>Jede im vorherigen Test durchgeführte Lösch-Operation einzeln mit <i>undo</i> rückgängig machen. Bei jedem <i>undo</i>-Schritt müssen die wiederhergestellten Elemente wieder im Outlineview angezeigt werden.</p>	O	O

3.8 Drag & Drop im Vespucci-Plugin

3.8.1 Allgemein

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Folgende Elemente im Package-Explorer können per D&D auf den Workspace einer *.sad-Datei gezogen werden: <ul style="list-style-type: none">• src-Ordner aus einem eigenen JAVA-Projekt• *.java-Dateien aus einem eigenen JAVA-Projekt• *.class aus referenzierten Bibliotheken• *.jar aus referenzierten Bibliotheken• sämtliche hierarchisch tiefer liegende Elemente:<ul style="list-style-type: none">◦ sämtliche JAVA-Packages◦ JAVA-Klassen (sowie innere Klassen)◦ JAVA-Konstruktoren◦ JAVA-Methoden◦ JAVA-Felder◦ JAVA-Konstruktoren	O	O
Ein Drop von nicht unterstützten Elementen auf die Hauptebene ist nicht möglich. Es wird das entsprechenden Maus-Icon angezeigt.	O	O
Ein Drop von unterstützten Elementen auf ein Ensemble erweitert die vorhandene Query um ein gültiges Prolog Statement, welches die fallengelassenen Elemente beschreibt.	O	O
Ein Drop von nicht unterstützten Elementen auf ein Ensemble ist nicht möglich. Es wird das entsprechende Maus-Icon angezeigt.	O	O
Ein Drop, der unterstützte und nicht unterstützte Elemente enthält, ist nicht möglich. Es wird das entsprechende Maus-Icon angezeigt.	O	O
Drops, die als Ziel weder ein Ensemble noch die Hauptebene haben, sind nicht möglich. Es wird das entsprechende Maus-Icon angezeigt.	O	O

3.8.2 D&D aus dem src-Verzeichnis

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das Package „de.tud.cs.se.flashcards.model“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen. Der Name des Ensembles lautet „de.tud.cs.se.flashcards.model“ und die Query „ package ('de.tud.cs.se.flashcards.model')“.	O	O
Die Datei „de.tud.cs.se.flashcards.model.Flashcard.java“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen. Der Name des Ensembles lautet „de.tud.cs.se.flashcards.model.Flashcard“ und der Inhalt der Query ist „ class_with_members ('de.tud.cs.se.flashcards.model', 'Flashcard')“.	O	O
Die Klasse „de.tud.cs.se.flashcards.model.Flashcard“ per Drag	O	O

<p>and Drop auf die weiße Fläche des Diagramms ziehen.</p> <p>Der Name des Ensembles lautet</p> <p>„de.tud.cs.se.flashcards.model.Flashcard“ und der Inhalt der Query ist „class_with_members (<code>'de.tud.cs.se.flashcards.model'</code>, <code>'Flashcard'</code>)“.</p>		
<p>Das Feld „de.tud.cs.se.flashcards.model.Flashcard.answer“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.</p> <p>Der Name des Ensembles lautet</p> <p>„de.tud.cs.se.flashcards.model.Flashcard.answer“ und der Inhalt der Query ist „field(<code>'de.tud.cs.se.flashcards.model'</code>, <code>'Flashcard'</code>, <code>'answer'</code>, <code>'java.lang.String'</code>)“.</p>	O	O
<p>Den Konstruktor „de.tud.cs.se.flashcards.model.Flashcard.Flashcard()“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen. Der Name des Ensembles lautet</p> <p>„de.tud.cs.se.flashcards.model.Flashcard.Flashcard“ und der Inhalt der Query ist „method(<code>'de.tud.cs.se.flashcards.model'</code>, <code>'Flashcard'</code>, <code>'<init>'</code>, <code>'void'</code>, <code>[]</code>)“.</p>	O	O
<p>Die Methode „de.tud.cs.se.flashcards.model.Flashcard.removeObserver“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.</p> <p>Der Name des Ensembles lautet</p> <p>„de.tud.cs.se.flashcards.model.Flashcard.removeObserver“ und der Inhalt der Query ist „method(<code>'de.tud.cs.se.flashcards.model'</code>, <code>'Flashcard'</code>, <code>'removeObserver'</code>, <code>'void'</code>, [<code>'de.tud.cs.se.flashcards.model.FlashcardObserver'</code>])“.</p>	O	O
<p>Die Methode der Klasse</p> <p>„de.tud.cs.se.flashcards.model.Flashcard.getRemembered“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.</p> <p>Der Name des Ensembles lautet</p> <p>„de.tud.cs.se.flashcards.model.Flashcard.getRemembered“ und der Inhalt der Query ist „method(<code>'de.tud.cs.se.flashcards.model'</code>, <code>'Flashcard'</code>, <code>'getRemembered'</code>, <code>'java.util.Date'</code>, <code>[]</code>)“.</p>	O	O
<p>Das Package „de.tud.cs.se.flashcards.model“,</p> <p>die Klasse „FlashcardSeriesFilter“,</p> <p>die Methode „FlashcardSeriesFilter“ (Konstruktor),</p> <p>die Methode „removeCards“,</p> <p>und das Feld „searchTerm“ gemeinsam selektieren und per Drag and Drop auf eine weiße Fläche des Diagramms ziehen.</p> <p>Der Name des Ensembles darf „de.tud.cs.se.flashcards.model“ lauten. Die Query ist eine „or-Permutation“ von:</p> <p>„package(<code>'de.tud.cs.se.flashcards.model'</code>)</p> <p>or</p> <p>method(<code>'de.tud.cs.se.flashcards.model'</code>, <code>'FlashcardSeriesFilter'</code>, <code>'<init>'</code>, <code>'void'</code>, [<code>'de.tud.cs.se.flashcards.model.FlashcardSeries'</code>])</p> <p>or</p> <p>method(<code>'de.tud.cs.se.flashcards.model'</code>, <code>'FlashcardSeriesFilter'</code>, <code>'removeCards'</code>, <code>'void'</code>, [<code>'int'</code>])</p> <p>or</p> <p>field(<code>'de.tud.cs.se.flashcards.model'</code>, <code>'FlashcardSeriesFilter'</code>, <code>'searchTerm'</code>, <code>'java.lang.String'</code>)“</p>	O	O

<pre>or class_with_members('de.tud.cs.se.flashcards.model','FlashcardSeriesFilter')"</pre>		
<p>Die Innere Klasse</p> <pre>„de.tud.cs.se.flashcards.model.Flashcard.InnerEvenIterator“</pre> <p>per Drag and Drop auf die weiße Fläche des Diagramms ziehen. Der Name des Ensemble lautet</p> <pre>„de.tud.cs.se.flashcards.model.Flashcard\$InnerEvenIterator“</pre> <p>und der Inhalt der Query ist</p> <pre>„class_with_members('de.tud.cs.se.flashcards.model', 'Flashcard\$InnerEvenIterator')“.</pre>	O	O

3.8.3 D&D aus einer JAR-Datei

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
<p>Die dnsns.jar aus dem „JRE System Library“-Bibliotheken Ordner per Drag and Drop auf die weiße Fläche des Diagramms ziehen. Der Name des Ensembles lautet „dnsns.jar“. Die Query ist eine „or-Permutation“ von:</p> <pre>„package('sun') or package('sun.net') or package('sun.net.spi') or package('sun.net.spi.nameservice.dns') or package('sun.net.spi.nameservice')“.</pre>	O	O
<p>Aus dem dnsns.jar das Package „sun.net.spi.nameservice.dns“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen. Der Name des Ensembles lautet „sun.net.spi.nameservice.dns“ und der Inhalt der Query ist</p> <pre>„package('sun.net.spi.nameservice.dns')“.</pre>	O	O
<p>Aus JAR-Dateien können alle Typen unterhalb der packages ebenfalls auf das Diagramm bzw. ein Ensemble gezogen werden. Die Namen der entstandenen Ensembles sowie deren Queries werden analog vergeben.</p>	O	O

3.9 Diagramm-Validierung

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
<p>„DerivedTest.sad“ muss den Fehler „Queries of non leaf Ensembles must be derived“ anzeigen. Die Query des zu dem Fehler gehörenden Ensembles muss auf <i>derived</i> gesetzt werden. Danach muss das Diagramm gespeichert werden. Nach dem Speichern darf der Fehler nicht mehr angezeigt werden.</p>	O	O
<p>„EmptyAndSameNameTest.sad“ muss die Fehler „Ensemble name must be set“ und „Ensemble must have unique name!“ enthalten. Die Namen der Ensembles im Vespucci-Diagramm so anpassen, dass alle Ensembles einen Namen haben und kein Name doppelt vorkommt. Das so modifizierte Vespucci-Diagramm speichern. Nach dem Speichern darf kein Fehler mehr angezeigt werden.</p>	O	O

3.10 Connection-Assistant-Mechanismus

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das Diagramm „Connection-Assistant-Mechanisms.sad“ öffnen.	O	O
Über die Popup Pfeile (mit der Maus über das Ensemble hovern) am Ensemble „1“ <u>Outgoing dependencies</u> mit dem Ensemble „1“ <u>als Source</u> zu den Target Ensembles „1.1“, „1.2“, „1.3“, „1.4“ anlegen.	O	O
Über die Popup Pfeile (mit der Maus über das Ensemble hovern) am Ensemble <u>Outgoing dependencies</u> mit dem Ensemble „2“ <u>als Source</u> zu den Target Ensembles „2.1“, „2.2“, „2.3“, „2.4“ anlegen.	O	O
Über die Popup Pfeile (mit der Maus über das Ensemble hovern) am Ensemble <u>Incoming dependencies</u> mit dem Ensemble „1“ <u>als Target</u> von den Source Ensembles „1.1“, „1.2“, „1.3“, „1.4“ anlegen.	O	O
Über die Popup Pfeile (mit der Maus über das Ensemble hovern) am Ensemble <u>Incoming dependencies</u> mit dem Ensemble „2“ <u>als Target</u> von den Source Ensembles „2.1“, „2.2“, „2.3“, „2.4“ anlegen.	O	O

3.11 Kopieren und Einfügen

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das Vespucci-Diagramm „Copy1.sad“ öffnen.	O	O
Das Ensemble „e1“ Kopieren und in „Copy1.sad“ einfügen.	O	O
Die Ensemble „e2“ und Ensemble „e3“ selektieren und zusammen Kopieren. Beide Ensembles in „Copy1.sad“ einfügen. Es müssen neben den beiden Ensembles auch die zwei dependencies zwischen den Ensembles mit Kopiert worden sein.	O	O
Das Ensemble „e4“ Kopieren und wider in „Copy1.sad“ einfügen. Die Kopie des Ensembles „e4“ muss einen dependency zum Ensemble „xxxx“ haben.	O	O
Alle Kopien umbenennen und sicher stellen das mit der Umbenennung eines Ensembles nie ein weiter Ensemble umbenannt wird.	O	O
Das Vespucci-Diagramm „Copy1.sad“ speichern.	O	O
Das Vespucci-Diagramm „Copy2.sad“ öffnen.	O	O
Die Ensembles „e1“, „e2“, „e3“ und „e4“ im Diagramm „Copy1.sad“ Selektieren und Kopierten. Anschließend in „Copy2.sad“ einfügen. In „Copy2.sad“ müssen jetzt 4 Ensembles und zwischen den Ensembles „e2“ und „e3“ zwei dependencies enthalten sein.	O	O
Das Ensemble „e3“ in „Copy2.sad“ und die Ensembles „e4“ und „xxxx“ in „Copy1.sad“ löschen. Beide Diagramme jeweils sofort speichern, schließen und wieder öffnen. Das Ensemble „e3“ in „Copy1.sad“ und das Ensemble „e4“ in „Copy2.sad“ müssen noch enthalten sein.	O	O

Erweiterte Veröffentlichungsscheckliste:

1. Palette

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Erstelle ein neues Vespucci Diagramm „extendedTest.sad“.	O	O
Der Titel der Palette für die Dependency-Pfeile heißt „Constraints“.	O	O

2. Syntax-Highlighting - Vorbereitung

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Erstelle ein neues Ensemble.	O	O
Rechtsklick auf das Ensemble, dann „Show Properties View → Query“.	O	O

2.1 Klammernzählung

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Gib folgende Query ein: „(DD)“	O	O
Platziere das Caret des Mauszeigers rechts neben die öffnende Klammer . Eine rechteckige Umrandung erscheint um die schließende Klammer.	O	O
Platziere das Caret des Mauszeigers rechts neben die schließende Klammer Eine rechteckige Umrandung erscheint um die öffnende Klammer.	O	O
Teste dieses Prinzip mit den folgenden Queries analog:	O	O
„(first bracket has no closing partner bracket())“	O	O
„(second closing bracket has no opening partner bracket())“	O	O
„(gotnone(got)(got)((got)got)((gotboth))“	O	O
„('String with (brackets)')“ - Ignorierung von „()“ in Strings.	O	O

2.2 Keyword-Highlighting

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Gib folgende Query ein: "package class_with_members class method field or derived Dies wird nicht gehighlighted, 'package in String-Apostrophen auch nicht'"	O	O
Alle fettgedruckten Wörter werden im Query-Tab gemäß den Einstellungen der Eclipse-Preferences hervorgehoben.	O	O

3. Ensemble-Beschreibung

3.1 Konsistenz bei Sonderzeichen und Minimierung

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Rechtsklick auf das Ensemble, dann „Show Properties View“.	O	O
Wähle Unterpunkt „Description“.	O	O
Gib folgende Description ein: „mä\$Sig gü/e Säö/\%“.	O	O
Minimiere die Description, speichere das Diagramm und schließe es.	O	O
Nach erneutem Öffnen und Expandieren ist die Description unverändert.	O	O

3.2 Konsistenz bei *nested ensembles*

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Erstelle ein Ensemble mit Namen „inner“.	O	O
Die Description „nested one“ durch Doppelklick (1. Klick: Ensemble fokussieren, 2. Klick auf das Description-Compartment) direkt eingeben.	O	O
Minimiere die Description von „inner“ und minimiere anschließend das Compartment von „testDescr“, speichere das Diagramm und schließe es.	O	O
Öffne es wieder, expandiere das Compartment von „testDescr“, expandiere die Description und bestätige, dass diese unverändert ist.	O	O

3.3 Multilining

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Erstelle ein neues Ensemble „LineBreak“ mit der Description „Diese Beschreibung soll absichtlich länger sein als die Standardgröße.“	O	O
Die Description darf nicht am Ende der Zeile abgeschnitten sein und das Ensemble muss sich der Länge der Description anpassen.	O	O
Minimiere und expandiere die Description und bestätige, dass sich selbige nicht geändert hat und korrekt angezeigt wird.	O	O
Öffne das PropertiesView von „LineBreak“ und füge als Description „This \n is \n Sparta!“ ein und überprüfe ob die Zeilenumbrüche auf der Hauptebene des Editors ebenfalls angezeigt werden. (\n = Zeilenumbruch)	O	O
Minimiere und expandiere die Description und bestätige, dass sich selbige nicht geändert hat und korrekt angezeigt wird.	O	O

Klicke auf das Description-Compartment des Ensembles „Line-Break“ auf der Hauptebene, füge wieder „This \n is \n Sparta!“ ein und drücke STRG+ENTER für einen manuellen Zeilenumbruch.	O	O
Minimiere und expandiere die Description und bestätige, dass sich selbige nicht geändert hat und korrekt angezeigt wird.	O	O

4. Mitgelieferte Dateien

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Im Ordner "Doc" des Packages "de.tud.cs.st.vespucci.diagram" sind die folgenden Dateien enthalten: <ul style="list-style-type: none"> • AutosizeUndoTestfaelle.xlsx • FunktionDerTracedatei.pdf • GUI_Test_Projekt.zip • GUI-Checkliste.odt • GUI-Checkliste.pdf • Verisoning HowTo.pdf • Verisoning HowTo.tex 	O	O

5. Versionierung von älteren *.sad-Dateien

5.1 Konvertierung

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Wähle im Ordner „old sad-files“ die Datei „iReview.sad“.	O	O
Beim Versuch des Öffnens durch Doppelklick erscheint ein PopUp-Menü.	O	O
Mit Klick auf „Yes“ wird eine fehlerfreie Konvertierung durchgeführt.	O	O
Im neuen „iReview.sad“ werden die nun auch Description-Felder korrekt und nach den Vorstellungen von Punkt 3 dieser Checkliste angezeigt.	O	O
Wähle im Ordner „old sad-files“ die Datei „Architecture.sad“.	O	O
Rechtsklick, dann „Vespucci“ und „Update to current version“.	O	O
Mit Klick auf „Yes“ wird eine fehlerfreie Konvertierung durchgeführt.	O	O
Im neuen „iReview.sad“ werden die nun auch Description-Felder korrekt und nach den Vorstellungen von Punkt 3 dieser Checkliste angezeigt.	O	O
Bei bereits konvertierten *.sad-Dateien erscheint der Menüpunkt „Update to current version“ nicht mehr.	O	O

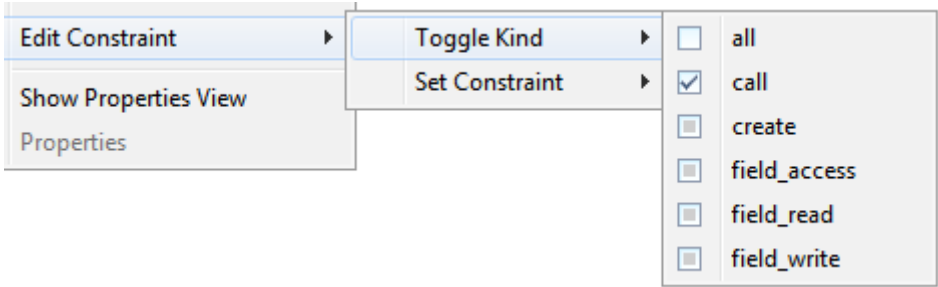
5.2 Korrektes Backup

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Das Backup heißt jeweils „<oldname>.v0.sad“.	O	O
Beim Versuch des Öffnens durch Doppelklick erscheint ein PopUp-Menü.	O	O
Klick auf „No“ wirft keine Fehler und führt auch keine Aktion aus.	O	O
Das Diagramm wird ohne Probleme geöffnet.	O	O
Wird ein Backup nochmals konvertiert so erhält dessen Backup einen Zeitstempel, der sich bei wiederholtem Konvertieren nicht stackt, sondern lediglich ändert.	O	O

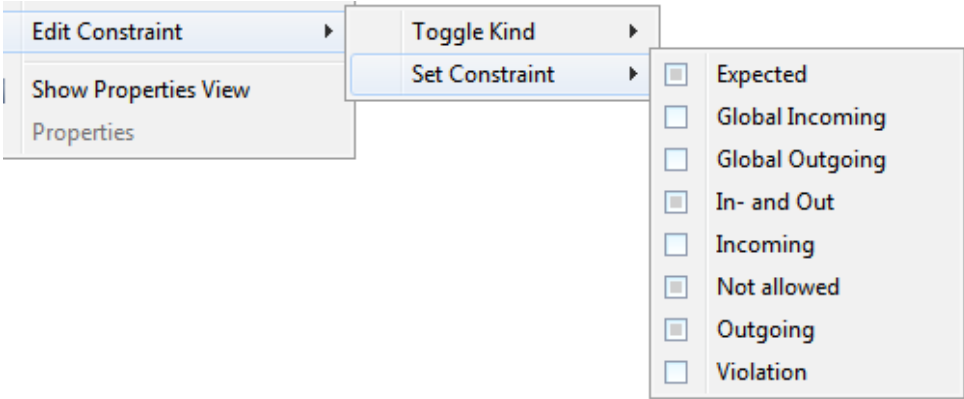
6. Kontextmenü - Vorbereitung

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Öffne die Datei „DependBeschrConstr.sad“.	O	O

6.1 Toggling Dependency Kinds

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Rechtsklick auf eine freie Stelle, dann „Select“ und „All Connectors“. Alle Constraints werden markiert.	O	O
Rechtsklick auf ein beliebiges Constraint, dann „Edit Constraint“ und „Toggle Kind“. Das Menü muss wie folgt aussehen: 	O	O
Klick auf „create“ setzt auf allen Constraints ebenfalls ein „create“.	O	O
Klick auf „call“ entfernt auf allen Constraint das „call“.	O	O
Klick auf „all“ setzt auf allen Constraints ebenfalls ein „all“.	O	O
Mache alle Änderungen rückgängig.	O	O

6.2 Typänderung

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Rechtsklick auf eine freie Stelle, dann „Select“ und „All Connectors“. Alle Constraints werden markiert.	O	O
Rechtsklick auf ein beliebiges Constraint, dann „Edit Constraint“ und „Set Constraint“. Das Menü muss wie folgt aussehen: 	O	O
Klick auf „Incoming“ wandelt alle Constraints entsprechend um.	O	O
Klick auf „Global Outgoing“ wandelt alle Constraints entsprechend um.	O	O
Mache alle Änderungen rückgängig.	O	O

7. Properties View

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Öffne ein beliebiges sad-File oder erstelle ein neues.	O	O
Es gibt einen Tab „Overview“, in dem alle Eigenschaften eines selektierten Objektes angezeigt werden.	O	O
Anmerkungen:		

8. Autosize + Undo

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Wähle die Datei „autosizeUndo.sad“ und öffne sie.	O	O
Rechtsklick auf das Ensemble, dann „Format → Autosize“.	O	O
Das Ensemble passt sich der Länge des Namens und der Description an.	O	O
Führe „Undo“ und „Redo“ aus. Es hat sich nichts geändert.	O	O

9. Keine Constraints zu Textfeldern.

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Erstelle oder öffne eine beliebige *.sad-Datei.	O	O
Erstelle nacheinander ein Ensemble, eine Attached Note zum Ensemble, eine Note, ein Text-Objekt und ein zweites Ensemble..	O	O
Es besteht keine Möglichkeit zu den Nicht-Ensembles von den Ensembles ausgehend eine Constraint zu definieren.	O	O

10. Validierung von Dependencies

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Wähle die Datei „correctDependencies.sad“ und öffne sie.	O	O
Im „Problems View“ darf kein Fehler erscheinen.	O	O
Wähle die Datei „incorrectDependencies.sad“ und öffne sie.	O	O
Im „Problems View“ erscheint der Fehler: „Dependency kind is invalid.“	O	O

11. Prolog Facts

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Wähle die Datei „PrologTest.sad“ und öffne sie.	O	O
Falls die Datei „PrologTest.pl“ noch nicht existiert wähle <i>Vespucci</i> → <i>Generate Architectural Model</i> im Kontextmenü von „PrologTest.sad“.	O	O
Die letzte Zeile von „PrologTest.pl“ lautet: global_incoming('PrologTest.sad', 1, 'Zugreifer', [], 'Sohn', [], [all]).	O	O

12. Neue Constraints: Global Incoming, Global Outgoing und Warning

12.1 Erstellung der Constraints

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Vorbereitung: Erstellung eines neuen Vespucci-Diagramms „new.sad“ mit zwei neuen Ensembles „A“ und „B“ und einem Empty Ensemble „C“.	O	O
Alle drei Constraints Global Incoming, Global Outgoing und Violation lassen sich über die Palette von „A“ nach „B“ erstellen.	O	O
Alle drei Constraints Global Incoming, Global Outgoing und Violation lassen sich über den Pfeil zwischen „A“ und „B“ erstellen.	O	O
Alle drei Constraints Global Incoming, Global Outgoing und Violation lassen sich über den Pfeil zwischen „A“ und einem neuem Ensemble „D“ erstellen.	O	O
Es lässt sich keine der drei Constraints über die Palette zu oder von einem Dummy erstellen.	O	O
Es lässt sich keine der drei Constraints über den Pfeil zu oder von einem Dummy erstellen.	O	O
Speichern/ Neu laden/ Undo und Redo/ Delete funktionieren.	O	O
Outlineview zeigt in allen oben beschriebenen Fällen die Constraints korrekt an.	O	O

12.2 Umorientieren / Set Type

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Die in 12.1 erstellten neuen Constraints lassen sich zwischen den Ensembles „A“ und „B“ und einem neuem Ensemble „D“ beliebig umorientieren.	O	O
Keine der Constraints lässt sich zu einem Empty Ensemble umorientieren.	O	O
Die in 12.1 zwischen „A“ und „B“ erstellten Constraints lassen sich über das Popupmenu Edit „Constraints/ Set Type“ beliebig ändern.	O	O
Gleiches ist für den Empty Ensemble „C“ nicht möglich, weil der Menüpunkt „Set Type“ ausgeblendet wird.	O	O
Constraints lassen sich nicht zu Notizen, Textfeldern u.a. umorientieren.	O	O

12.3 Grafische Tests

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Anlegen eines Violation Constraints zwischen „A“ und „B“. Während des Umrorientieren richtet sich das Warndreieck der Neigung des Pfeils entsprechend aus. Bei mehr als 90 Grad Neigung kehrt sich die Richtung um.	O	O

12.4 Prolog

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Vorbereitung: Erstellen folgender Constraints: 1) Global Incoming von „A“ nach „B“ 2) Global Outgoing von „B“ nach „C“ 3) Violation von „C“ nach „A“ Abspeichern. Erzeugte Prologdatei öffnen.	O	O
Es existieren folgende Einträge in dieser Reihenfolge: global_incoming('new.sad',1,'A',[],'B',[all]). global_outgoing('new.sad',2,'B',[],'C',[all]). global_violation('new.sad',3,'C',[],'A',[all]).	O	O

13. Problemsview

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Ein neues Ensemble mit Query „ () “ erstellen.	O	O
Ein neues Ensemble mit Query „ ()) “ erstellen.	O	O
Ein neues Ensemble mit Query „ [] “ erstellen.	O	O
Ein neues Ensemble mit Query „ []] “ erstellen.	O	O
Ein neues Ensemble mit Query „) (“ erstellen.	O	O
Ein neues Ensemble mit Query „] [“ erstellen.	O	O
Ein neues Ensemble mit Query „ (' ') “ erstellen.	O	O
Ein neues Ensemble mit Query „ ('un'voll'ständig'_') “ erstellen.		
Ein neues Ensemble mit Query „ (k(orrek)t) “ erstellen.	O	O
Ein neues Ensemble mit Query „ [k(orrek)t] “ erstellen.	O	O
Ein neues Ensemble mit Query „ (k'o(rr)ek't) “ erstellen.	O	O
Ein neues Ensemble mit Query „ (k(orrek)t) “ erstellen.	O	O
Ein neues Ensemble mit Query ('de.tud.cs.se.flashcards.model', 'FlashcardSeriesFilter', 'void', ['int']) erstellen.	O	O

Das Diagramm abspeichern.	O	O
Im Problemsview sollten bei folgenden Ensembles eine Warnung wegen unvollständigen Prologstrings angezeigt werden: <ul style="list-style-type: none"> • „ (' ') “ • „ ('un'voll'ständig' _ ') “ 	O	O
Im Problemsview sollten bei folgenden Ensembles eine Warnung wegen unvollständiger Klammerung mit '(' und ')' (Parenthesis) angezeigt werden: <ul style="list-style-type: none"> • „ (() “ • „ ()) “ • „) (“ 	O	O
Im Problemsview sollten bei folgenden Ensembles eine Warnung wegen unvollständiger Klammerung mit '[' und ']' (Brackets) angezeigt werden: <ul style="list-style-type: none"> • „ [[] “ • „ []] “ • „] [“ 	O	O

TEMPLATE

<u>Testfall</u>	<u>Windows 64-Bit</u>	<u>Linux 64-Bit</u>
Case1	O	O
Case2	O	O
Case3	O	O
Case4	O	O