Pflichtenheft

Conways‘s

Game of Life

„Eine universelle Software zur Simulation zellulärer Automaten“

Auftraggeber:

- Hochschule Bochum
- Ansprechpartner: Dipl.-Inform. Christian Düntgen
- Raum: D 3-30

Auftragnehmer:

- Die 5 Kranken Schwestern
- Weder krank noch Frauen
- Definitionsphasenmanager: Jörg Galilee Uwimana
- Architekt (Entwurfsbeauftragter): Felix Reinhardt
- Gruppenschuldiger, Spezifikationsbeauftragter: Alex Chojnatzki
- Implementierungs-Beauftragter: Nicholas Schuran
- Kundenbetreuer, Außenminister, Abnahmebeauftrgter: Diaa El Bathich

Stand: 21. Nov. 2021 - 02:25

Contents

1 Zielbestimmung

1.1 Musskriterien

Das Programm soll dazu dienen, Zelluläre Automaten auf einem 2D orthogonalen Spielfeld darstellen zu können. Dazu werden als Beispiel die Regeln für Conway's Game of Life verwendet. Hierzu sind unbedingt die folgenden Features erforderlich:

M0001	UI	Das Programm muss eine graphische Oberfläche haben.
M0002	Scope	Es soll ein zellulärer Automat mit möglichst großer Freiheit definiert und simuliert werden können.
M0003	Darstellung Spielfeld	Die Darstellung des Zellulären Automaten Erfolgt über eine 2D Matrix aus Quadraten deren Farbe und Helligkeit den Zustand eines Feldes wiedergeben.
M0004	Transitionsregeleditor	Die Transitionsregeln sollen über eine definierte und im Handbuch dokumentierte Syntax (invers Polnische Notation, ggf. auch mathematische Schreibweise) formuliert werden können. Der neue Zustand einer Zelle darf dabei von der Zelle selbst, sowie von den Umliegenden acht benachbarten Zellen abhängen. Ihr Status wird in Variablen bereitgestellt.
M0005	Spielfeldaufbau	Das Spielfeld soll als 2D Array von Integerwerten ausgeführt sein, welche den Zellzustand repräsentieren.
0006	Spielfeldgröße	Die Spielfeldgröße soll vor Simulationsstart vom Benutzer über (Text-)Eingabefelder festgelegt werden können.
M0007	Speichern & Laden	Spielfeldzustand und Transitionsregeln sollen seperat gespeichert und geladen werden können.

M0008	Einfügen	Es sollen Figuren in das Spielfeld eingefügt werden können. Dies soll so geschehen, dass Figuren als Spielstände mit kleinerer Feldgröße als ganzes geladen und eingefügt werden können.
M0009	Navigation	es soll möglich sein, das Spielfeld mit Zoom und Pan verschieden zu betrachten.
M0010	Spielfeldmanipulation	Der Zustand einer Zelle soll durch Mausklick darauf auf einen wählbaren Wert einstellbar sein. Das Wählen des Werts soll durch ein Texteingabefeld auf der Benutzeroberfläche Erfolgten. Details in der Beschreibung der Be–nutzeroberfläche.
M0011	Topologie	Das Randverhalten des Spielfelds soll zwischen begrenztem Rechteck und Torus (Zellen an den Kanten sind mit den ihnen gegenüberliegen zellen benachbart) wählbar sein.
M0012	Automatische Simula–tion	Die Simulationsgeschwindigkeit soll über einen Slider einstellbar sein. Die Simulation soll über einen Button gestartet und unterbrochen wer–den können.
M0013	Manuelle Simulation	Über einen Button soll die nächste Generation berechnet und angezeigt werden können.
M0014	Zufälliger Anfangszus–tand	Der Spielfeldzustand soll zufällig generier–bar sein. Dazu soll einem Zellzustand eine Wahrscheinlichkeit zugewiesen werden können, mit dem Default–Zustand 0, sodass jede Zelle genau einen Zustand erhält.
M0015	Startbedingungen	Beim Programmstart soll ein 80x80 Zellen großes Spielfeld präsentiert werden, auf welches die Spielregeln für Conway's Game of Life verwendet werden.

M0016	Nachbarschaftswahl	Die Verwendung von sowohl Moore als auch Neumann Nachbarschaft muss ermöglicht werden. Am Rand einer endlichen Fläche können nicht alle Nachbarn existieren und werden auch nicht gezählt.
M0017	Numerische Anzeige des Zellzustandes	Der genaue Wert einer Zelle muss irgendwie anzeigbar sein.

1.2 Wunschkriterien

W0001	Undo	Es sollen Eingaben rückgängig gemacht werden können.
W0002	Regeleditor	Eingabe der Regeln in für Menschen gut lesbarer Mathematischer Schreibweise, mit Grundrechenarten und logischen Operationen
W0003	Performance	Multithreading parallelisierbarer Prozesse
W0004	Farbanpassung	Wenn möglich soll die Farbe eines Zustands durch den Benutzer einstellbar sein.
W0005	Wahl der Nachbarschaft	Es wurde gewünscht, zwischen der Von–Neumann–Nachbarschaft und Moore–Nachbarschaft wählen zu können. Hierzu sei angemerkt dass dies nichts anderes erfordert, als zwei verschiedene Transitionsregeln mitzuliefern, welche entsprechend benannt sind und dann geladen werden können. Es kann genauso ein Transitionsregel–Ausdruck für Von–Neumann Nachbarschaft wie für Moore–Nachbarschaft gebaut werden, indem die entsprechenden diagonalliegenden Zellen berücksichtigt werden oder eben nicht. Daher ist für dieses Feature keine Ergänzung in der Programmarchitektur erforderlich.

2 Produkt–Einsatz

2.1 Anwendungsbereich

Das Programm soll dazu dienen, Zelluläre Automaten mit recht großer Freiheit bauen zu können. Ob es sich dann um Game of Life, einen Waldbrandsimulator handelt, ist dann außen vor.

2.2 Zielgruppen

Die Verwendung dieses Programms für Conway's Game of life ist einfach, da die Spielregeln mitgeliefert werden. Dies kann von allen interessierten ausprobiert werden, da die Manipulation des Spielfelds zum ausprobieren einlädt.

Leider ist es nicht möglich, den Regeleditor intuitiv bedienbar zu gestalten, da es für eine effiziente Verarbeitung notwendig ist, den Zustand einer Zelle in der nächsten Generation als Mathematische Funktion der Zustönde der Nachbarzellen darzustellen. Aus diesem Grund gibt es zwar einen Leitfaden, um Mathematische Funktionen mit den Umliegenden Zellen als Ausgangsdaten zu erstellen, es ist jedoch nicht einfach, dies zu tun. Deal with it.

2.3 Produktumgebung

2.3.1 Softwareanforderungen

- Ein "Java Runtime Envrionment" der Version 1.8.x oder neuer. Ältere Versionen werden nicht getestet.
- Betriebssystem, was in der Lage ist, besagte JRE auszuführen.

2.3.2 Hardwareanforderungen

- Ein Computer aus diesem Jahrtausend mit einer Prozessorarchitektur für die eine JRE verfügbar ist. Dual–Core oder besser empfohlen, Dienstalter nicht über 1,6 Dekaden.
- Farbmonitor mit ausreichend Nutzfläche mindestens 80 ∗80 Pixel. Empfohlen wird ein HD Ready Display mit 720∗1280 Pixeln.
- Maus die mit Links.–/ Rechtsklicktasten und einem Mausrad bestückt ist.
- Tastatur

2.4 Betriebsbedingungen

- Schreib– und Leserechte für die Speicherstände.
- Verfügbarer Speicherplatz (großzügigerweise werden 500 MB Festplattenspeicher empfohlen).
- Arbeitsspeicher angepasst an die Feldgröße (Standardtkonfiguration benötigt 128 MB).

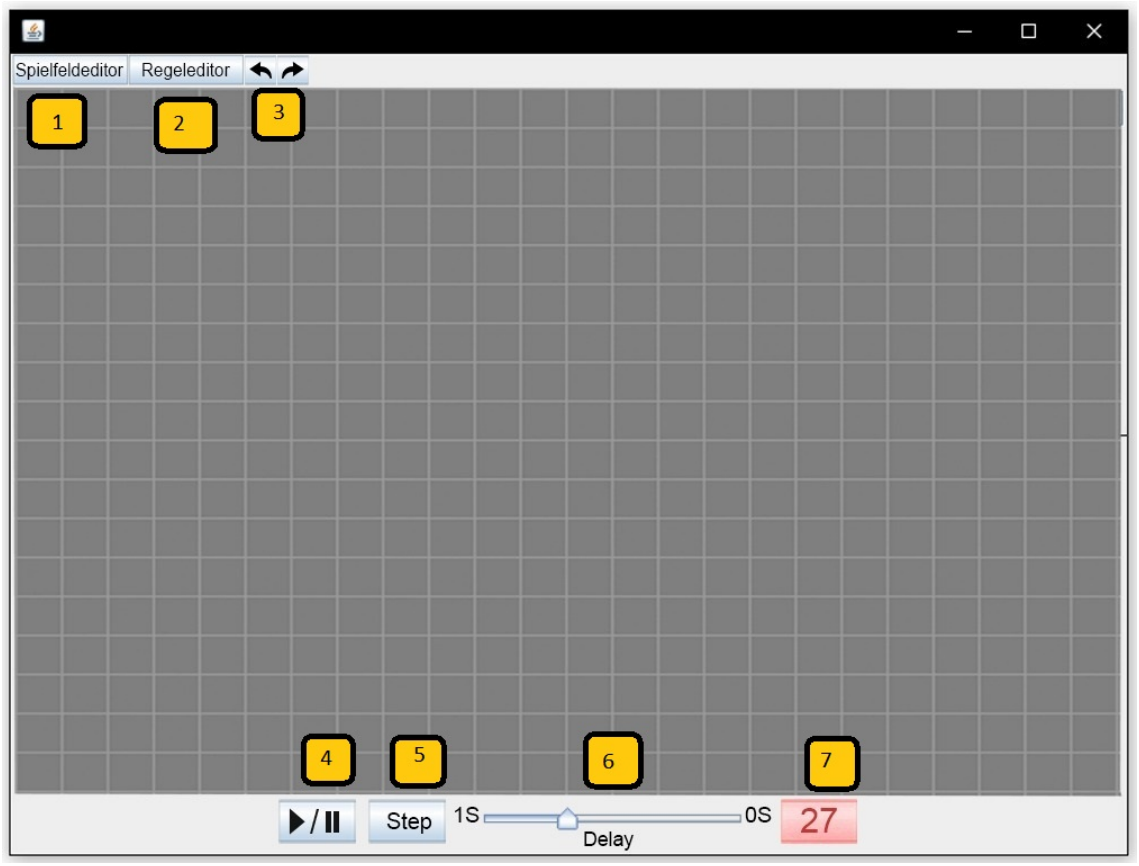
3 Produktfunktionen

3.1 Funktionale Anforderungen

3.1.1 Benutzeroberfläche

Nach dem Start wird folgende Oberfläche als Standard erscheinen. Im folgenden werden die (numerierten) UI–Elemente erläutert.

Hauptbenutzerfläche



AF–01	Spielfeldeditor	Mausklick auf den Button Spielfeldeditor öffnet das Dropdownmenü ”Spielfeldeditor”.
AF–02	Regeleditor	Mausklick auf den ”Regeleditor– Button ” öffnet das Dropdownmenü ”Regeleditor”
AF–03	Undo/Redo	Mausklick auf ”undo” macht die letzte Eingabe des Spielers rückgängig. ”Redo” stellt sie wieder her (Hochoptional)
AF–04	Simulation starten / un–terbrechen	Mausklick auf den Button schaltet die automatische Simulation an oder aus.

AF–05	Stepover	Mauklick auf den ”STEP– Button ” führt genau einen Simulationsschritt aus.
AF–06	Delay–Slider	Mit diesem Slider kann die Verzögerung zwischen zwei Generationen zwischen 1 und 0 sekunden stufenlos ausgewählt werden.
AF–07	Zellmodifikation	In diesem Textfeld kann (nur int) der Wert festgelegt werden, auf den eine Zelle gesetzt werden soll, falls man mit der Maus darauf klickt.

Anwendungsfall ID	AF–01
AF Name	Spielfeldeditor
Akteur	Benutzer des Programms
Vorbedingung	Programm gestartet, Benutzer lebendig
Auslösendes Ereignis	Mausklick auf den ”Spielfeldeditor–Button”
Nachbedingung Erfolgt	Öffnen des Dropdownmenü ”Spielfeldeditor”
Nachbedingung Fehlschlag	Ausgeben der programminternen Fehlermeldung im Dialogfenster.
Ablauf	Nutzer klickt auf Button und das Dropdown–Menü ”Spielfeldeditor” (SE–0X) öffnet sich.

Anwendungsfall ID	AF–02
AF Name	Regeleditor
Akteur	Benutzer des Programms
Vorbedingung	Programm gestartet, Benutzer lebendig
Auslösendes Ereignis	Mausklick auf den ”Regeleditor–Button”
Nachbedingung Erfolgt	Öffnen des Fensters ”Regeleditor”
Nachbedingung Fehlschlag	Ausgeben der programminternen Fehlermelung im Dialogfenster.
Ablauf	Nutzer klickt auf Button und das Regeleditor–Dropdownmenü öffnet sich.

Anwendungsfall ID	AF–03
AF Name	Undo/Redo
Akteur	Benutzer des Programms
Vorbedingung	Irgendeine Aktion im Programm wurde bereits durchgeführt.
Auslösendes Ereignis	Mausklick auf den Undo– bzw. Redo–Button
Nachbedingung Erfolgt	Undo: Rückgängig machen der zuletzt ausgeführten Aktion. Redo: Wiederherstellen. Hochoptional.
Nachbedingung Fehlschlag	Undo: Fehlermeldung : ”Nichts zurückzusetzen”, Redo: Fehlermeldung: ”nichts wiederherzustellen”.
Ablauf	Nutzer klickt auf Undo, die zuletzt ausgeführte Aktion wird zurückgesetzt. REDO: die zuletzt ausgeführte Aktion wird wiederhergestellt.

Anwendungsfall ID	AF–04
AF Name	Play/Pause
Akteur	Benutzer des Programms
Vorbedingung	Programm gestartet, Benutzer lebendig
Auslösendes Ereignis	Mausklick auf den ”Play/Pause–Button”
Nachbedingung Erfolgt	Umschalten der Simulation zwischen ”Simulation läuft” und ”Pausiert” Anzeige des aktuellen Spielzustands durch Icon oder Farbe. Automatischer fortlauf Zeitdiskreter Aktualisierung aller Zellzustände anhand der Transitionsregeln.
Nachbedingung Fehlschlag	Ausgeben der programminternen Fehlermelung im Dialogfenster.
Ablauf	Simulation gestoppt: User klickt auf Button, Simulation startet, Button zeigt nun ein rotes Quadrat an. Simulation läuft: User klickt auf Button, Simulation stoppt. Button zeigt nun ein rechtsweisendes grünes Dreieck an.

Anwendungsfall ID	AF–05
AF Name	STEPOVER
Akteur	Benutzer des Programms
Vorbedingung	Programm gestartet, Benutzer lebendig und im Vollbesitz seiner Maus
Auslösendes Ereignis	Mausklick auf den ”STEPOVER–Button”
Nachbedingung Erfolgt	Einamlige Zeitdiskrete Aktualisierung aller Zellzustände anhand der Transitionsregeln.
Nachbedingung Fehlschlag	Ausgeben der programminternen Fehlermelung im Dialogfenster.
Ablauf	Nutzer klickt auf Button und der Zelluläre Automat bewegt sich genau einen Simulationsschritt weiter.

Anwendungsfall ID	AF–06
AF Name	Delay–Slider
Akteur	Benutzer des Programms
Vorbedingung	Programm gestartet, Benutzer lebendig
Auslösendes Ereignis	Mausklick und Ziehen auf dem ”Delayslider”
Nachbedingung Erfolgt	Anpassung des Simulationsschritt–Delays zwischen 0 und 5 Sekunden
Nachbedingung Fehlschlag	Ausgeben der programminternen Fehlermelung im Dialogfenster.
Ablauf	... Nicht im Ernst... Sliderbedienungsfähigkeit wird vorausgesetzt. Die Verzögerung ist genau als solche zu verstehen: Sie definiert die Zeit, die das Programm zwischen zwei Spielfeldzustandsiterationen verstreichen lässt. Über den Wertebereich kann bei Bedarf verhandelt werden.

Anwendungsfall ID	AF–07
AF Name	Zellmodifikation
Akteur	Benutzer des Programms
Vorbedingung	Programm gestartet, Benutzer lebendig und mit einem Alkoholpegel < 5 %
Auslösendes Ereignis	Mausklick auf das ”Zustandstextfeld” oder auf das ”Spielfeld”
Nachbedingung Erfolgt	Textfeld: User kann neuen Edit–Zielzustand angeben und mit Enter bestätigen. Spielfeld: Zustand der angeklickten Zelle wird auf den Zustand im Textfeld gesetzt.
Nachbedingung Fehlschlag	Textfeld: Bei Eingabe einer Zeichenfolge welche keinen signed Integer repräsentiert: Fehlermeldung und bisherigen Zustand beibehalten.
Ablauf	Textfeld: Nutzer klickt auf Textfeld. Nutzer gibt ein, welcher Zielzustand gewünscht ist. Nutzer bestätigt mit Enter. Spielfeld: Nutzer klickt beliebige Zelle an. Zustand der Zelle wird überschrieben durch Zustand im Textfeld Bei Abbruch wird der vorherige Wert weiterverwendet. Hinweis: Die Alkoholprüfung wird beim Programm–start durch bestätigung eines Pop–Ups durchgeführt ”Dieses Programm darf nur von Personen mit einem Alkoholpegel $< 5\%$ verwendet werden. Durch klick auf OK bestätigen Sie, dass sie diese Bedingung erfüllen”

Nachtrag Dialogfenster:

Dialogfenster lassen sich innerhalb des Pragrammes nicht den Fokus entnehmen,
bis die Interaktion abgeschlossen ist.

Regeleditor

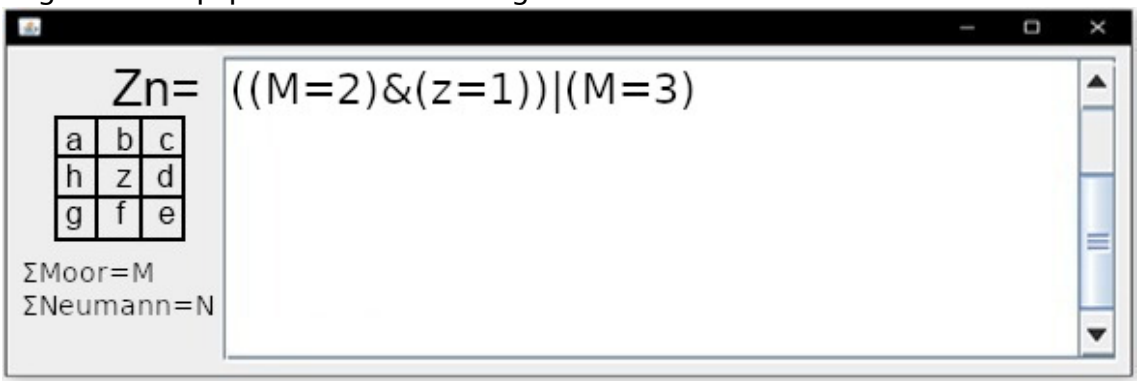
RF–01	Laden	Ruft den Filechooser zum Laden eines anderen Regelausdrucks auf
RF–02	Speichern	Ruft den Java–Swing–Filechooser zum Speichern des aktuellen Regelausdrucks auf.
RF–03	Topologiewechsler	Auswahlschalter für das Spielfeldrandverhalten.
RF–04	Regel Bearbeiten	Ruft das Popup–Fenster zum Regelausdruck bear–beiten auf.

Anwendungsfall ID	RF–01
AF Name	Laden
Akteur	Benutzer des Programms
Vorbedingung	Programm gestartet, Benutzer lebendig, Dropdown–menü ”Regeleditor” auswählen
Auslösendes Ereignis	Mausklick auf den ”Laden– Button”
Nachbedingung Erfolgt	Öffnen des Java–Swing–Fensters mit FileChooser zum öffnen einer Regeldatei
Nachbedingung Fehlschlag	Fehlermeldung ”Laden Fehlgeschlagen”, ggf. mit Ursache ”fehlerhafter Ausdruck” oder ”zugriffsfehler” und Rückkehr zur Haupt–Oberfläche. Beibehalten der bisherigen Regeln.
Ablauf	Nutzer klickt auf Button und öffnet das Fenster zum laden einer Regeldatei. Durch auswählen und Bestätigen durch klick auf ”Öffnen” wird die aktuell aktive Regel durch die geladene ersetzt. Prüfung mittels Algorithmus, ob die Regel der Syntax entspricht. Filechooser gibt Dateiformat .txt vor.

Anwendungsfall ID	RF–02
AF Name	Speichern
Akteur	Benutzer des Programms
Vorbedingung	Programm gestartet, Benutzer lebendig, Dropdown–menü ”Regeleditor” auswählen
Auslösendes Ereignis	Mausklick auf den ”Speichern– Button”
Nachbedingung Erfolgt	Öffnen des Java–Swing–Fensters mit FileChooser zum speichern einer Regeldatei als plain text string.
Nachbedingung Fehlschlag	Fehlermeldung ”Fehler beim Speichern” und Rückkehr zur Haupt–Oberfläche.
Ablauf	Nutzer klickt auf Button und öffnet das Fenster zum Speichern einer Regeldatei. Durch auswählen und Bestätigen durch klick auf ”Öffnen” wird die aktuell aktive Regel an besagter Stelle im Dateiformat .txt gespeichert.

Anwendungsfall ID	RF–03
AF Name	Topologie
Akteur	Benutzer des Programms
Vorbedingung	Programm gestartet, Benutzer lebendig, Dropdown–menü ”Regeleditor” auswählen
Auslösendes Ereignis	Mausklick auf den ”Topologie– Radio– Button”
Nachbedingung Erfolgt	Setzen der Spielfeldkantenbehandlung auf Torus oder Beschränkt, je nach Wunsch.
Nachbedingung Fehlschlag	Ausgeben der programminternen Fehlermeldung im Dialogfenster.
Ablauf	Durch Klick auf ”Standard” wird das Spielfeld als endliches Spielfeld behandelt, an den Kanten wer–den alle nachbarzellen als ”Zustand 0” angenom–men. Durch klick auf ”Torus” werden die Zellen an den Kanten die Zellen an gegenüberliegenden Kan–ten als Nachbarn behandeln.

Regeleditor Popup–Fenster mit Texteingabe:



Anwendungsfall ID	RF–04
AF Name	Regel Bearbeiten
Akteur	Benutzer des Programms
Vorbedingung	Programm gestartet, Benutzer lebendig, Dropdownmenü ”Regeleditor” ausgewählt.
Auslösendes Ereignis	Mausklick auf den ”Regel Bearbeiten”–Button im Regeleditor–Dropdownmenü
Nachbedingung Erfolgt	Öffnen des Popupfensters ”Regeleditor” (siehe oben).
Nachbedingung Fehlschlag	Ausgeben der programminternen Fehlermeldung im Dialogfenster.
Ablauf	Öffnen des Übergangsregel–Editors. Beim Öffnen steht im Textfeld die zurzeit verwendete Transitionsregel. Durch Tastatureingabe kann der String im Textfeld verändert werden, womit die Transitionsregel angepasst wird. Ferner gibt es ein Bild links, welches als Hilfestellung die Variablen angibt, welche die Zellzustände von Nachbarzellen angeben. Auf die Weise kann der Zustand der akuell betrachteten Zelle für die nächste Iteration auf Basis ihrer Nachbarn berechnet werden

Spielfeldeditor Dropdown

SE–01	Laden	Ruft Filechooser auf, worüber ein bereits gespeichertes Spielfeld geladen wird.
SE–02	Einfügen	Einen kleineren Spielfeldzustand in das Aktuelle wird an einer beliebigen Stelle eingefügt.
SE–03	Speichern	Aktueller Zustand des Spielfeldes wird in einer Datei gesichert.
SE–04	Größe	Dimensionen: Das erste Eintragskästchen gibt die Breite an, das Zweite die Höhe des gewünschten Spielfeldes.
SE–05	Anwenden	Das aktuelle Spielfeld wird auf die gewünschten Dimensionen gebracht.
SE–06	Zufallsgenerator	Werkzeug um das Spielfeld mit Zufälligen werten zu füllen.
SE–07	Clear	Generiert ein leeres Spielfeld.

Anwendungsfall ID	SE–01
AF Name	Laden
Akteur	Benutzer des Programms
Vorbedingung	Simulation über den ”Play/Pause– Button” pausieren. Hintergrundfarbe des Buttons färbt sich grau.
Auslösendes Ereignis	Mausklick auf den ”Laden– Button”
Nachbedingung Erfolgt	Öffnen des Java– Swing– Filechoosers zum öffnen der Spielfeld .csv Datei. Nach beenden des Auswählens: Rückkehr auf die Hauptoberfläche.
Nachbedingung Fehlschlag	Ausgbe der programminternen Fehlermeldung im Di–alogfenster.
Ablauf	Nutzer klickt auf den Laden–Button und wählt die jeweilige Datei aus, die in das Spiel eingebunden werden soll.

Anwendungsfall ID	SE–02
AF Name	Einfügen
Akteur	Benutzer des Programms
Vorbedingung	Simulation über den ”Play/Pause– Button” pausieren. Hintergrundfarbe des Buttons färbt sich grau.
Auslösendes Ereignis	Mausklick auf den ”Einfügen– Button”
Nachbedingung Erfolgt	Öffnen des Java– Swing– Fensters mit Filechooser zum öffnen einer Datei um ein kleineren Spielzustand per ”Drag and Drop” in das aktuelle Spiel einzufügen.
Nachbedingung Fehlschlag	Ausgbe der programminternen Fehlermeldung im Di–alogfenster.
Ablauf	Nutzer klickt auf den ”Einfügen–Button” und wählt aus dem Pop–Up–Fenster den jeweiligen Spielzu–stand aus, der in das Spielfeld eingebunden wer–den soll. Anschließend Rückkehr zur Haupto–berfläche, auf welcher der Nutzer den einzufügenden Spielstand nun per linksklick an gewünschter Stelle einfügen kann.

3.2 Transitionsregeleditor: Heimlicher Star der ganzen Show

Die Forderung, die Transitionsregeln des Spiels zur Laufzeit (nicht zur Simulationszeit) verändern zu können, benötigt einen Interpreter, weil Code nicht wirklich nachkompiliert werden kann. Da die Zellzustände als Integer–Variablen gespeichert sind, ist es leicht, Arithmetik mit ihnen zu betreiben. Vergleichsoperationen können dadurch eingesetzt werden, dass ”wahr” wie ”1” und ”falsch” wie ”0” behandelt wird. Ferner ist es zwingend notwendig, das Verwenden von Klammern zu erlauben. Es ist Nützlich, den Ausdruck in gewohnter Mathematischer Schreibweise angeben zu können, dies für jede Zelle auszuwerten ist jedoch ressourcenintensiv und daher ungeeignet. Denselben Ausdruck in invers polnischer Notation anzugeben verkürzt die Berechnungsdauer enorm, weil sie jetzt proportional zur Länge des invers polnisch aufgeschriebenen Ausdrucks ist, statt dass darüber hinaus jedes mal der Ausdruck rekursiv geparst werden muss. Ob es in der vorgegebenen Zeit gelingt, einen Übersetzer zu schreiben ist fraglich. Daher werden hier Beispiele für mathematische und Polnische Notation angegeben.

Erforderlich ist es dafür, die Zellzustände der Nachbarzellen in Variablen bereitzustellen, welche in der Syntax verwendet werden können, zur Vereinfachung wird für Moore–Nachbarschaft und Von–Neumann–Nachbarschaft zusätzlich die Summe der betreffenden Variablen geliefert, um Schreibaufwand zu sparen. Aus Zeit– und Lohngründen ist es nicht vorgesehen, die Verwendung von eigenen Variablen zu ermöglichen.

Die zur Verfügung stehenden Variablen und Operatoren:

Symbol	Name	Beschreibung
+	Addition	Addiert die Zahlen links und rechts des Operators
–	Subtraktion	addiert den linken Wert mit dem negativen des rechten Wertes
*	multiplikation	multipliziert den linken Wert mit dem rechten Wert
/	division	dividiert den linken durch den rechten Wert. Achtung: Zellzustände sind integer, daher wird die Division wie Division von Integern in Java stets abgerundete Ergebnisse produzieren.

=	gleichheit	gibt 1 zurück, wenn die linke Seite gleich der rechten ist, andernfalls gibt es 0 zurück VORSICHT: Es ist kein Zuweisungsoperator
&	AND	gibt den Integer zurück, den die bitweise Und–Operation auf linker und rechter Seite produziert. Die Verant–wortung, gültige Ausdrücke zu finden wird dem Benutzer auferlegt.
|	OR	gibt den Integer zurück, welchen die bitwesie Oder–Operation auf linker und rechter Seite produziert. Die Verantwortung, Regeln gültig aufzuschreiben wird dem Benutzer auferlegt.
#	XOR	gibt den Integer zurück, welcher bei der bitweisen XOR–Verknüpfung zwiischen linker und rechter Seite entsteht. Benutzung auf eigene Gefahr.
<	kleiner	gibt 1 zurück, wenn der linke Wert strikt kleiner ist als der rechte, anson–sten 0
>	größer	gibt 1 zurück wenn der linke Wert strikt größer ist als der rechte, ansonsten 0
()	Klammern	Klammern legen wie üblich fest, welche Operationen vor anderen Operationen ausgeführt werden sollen. Sie entfallen in polnischer Schreibweise

a, b, c, d, e, f, g, h	Nachbarn	Variablen, welche die Werte der benachbarten Zellen enthalten. Die Anordnung entnehmen Sie bitte dem Bild "Regeleditor"
z	Zellzustand	Variable, welche den Wert der betrachteten Zelle zurückgibt, sodass dieser in der Berechnung verwendet werden kann.
m	moore–Nachbar	Variable welche die Summe aller Nachbarzellen zurückgibt. Nützlich, um Schreibaufwand zu sparen, wenn die einzelnen Zellzustände nicht interessant sind.Äquivalent zu (a+b+c+d+e+f+g+h)
n	Neumann–Nachbar	Variable welche die Summe aller Neumann–Nachbarzellen zurückgibt, um Schreibaufwand zu sparen. Äquivalent zu (b+d+f+h)

In dieser Schrebweise sieht die Transitionsregel für Conways Game of Life wie folgt aus:

$$Zn : ((m = 2)\&(z = 1))|(m = 3)$$

In umgekehrt polnischer Notation für den Rechner:

$$|, =, m, 3, \&, =, z, 1, =, m, 2$$

Dabei stehen Zahlen bzw. Variablen für die Operation "lege auf den Stack", ein Operator nimmt die beiden vorhergehenden Werte von links nach rechts vom Stack und legt das Ergebnis zurück auf den Stack. Wird dieser Ausdruck von Rechts nach links durchlaufen, so ist die Berechnung dieselbe wie in geklammerter Schreibweise, weniger Übersichtlich für einen Menschen, aber für einen Computer mittels eines nachgebauten Stacks und switch–Case–Anweisungen schneller ausführbar als der geklammerte Ausdruck. Hoffentlich ist diese Methode effizient genug, um eine zügige Simulation zu ermöglichen. Natürlich würde es in Hardware direkt schneller gehen, aber wir habe

3.3 Nichtfunktionale Anforderungen

3.3.1 Performance

- Lineare Laufzeit der Generationsberechnung pro Spielfeldgröße durch nutzung der Pol–nischen notation.
- Vervielfachen der Geschwindigkeit von Bild und Regelberechnung durch Multithreading.

4 Testszenarien

Alle Testscenarien beziehen sich auf den Fall, dass das Programm Gestartet ist. Alles andere steht im Handbuch des jeweiligen Betriebssystems.

Ablauf	Klick außerhalb eines Dialogfensters während dieses Geöffnet ist.
Erwartetes Ergebnis	Dialogfenster behält Fokus. Aktion nicht ausführen.

4.1 Hauptfenster

Vorbedingung: Automatische Simulation gestoppt(Ausgangszustand)

Ablauf	Rechtsklick auf Lupe ohne Vorherige Wertzuweisung durch Linksklick ins Feld.
Erwartetes Ergebnis	Einfügen Modus wird Gestartet und ermöglicht einfügen von 0.

Vorbedingung: Simulation Gestartet

Ablauf	Benutzung von Step, Lupe oder Klick ins Spielfeld.
Erwartetes Ergebnis	Keine Aktion.

Ablauf	Klick auf Spielfeldeditor oder Regeleditor.
Erwartetes Ergebnis	Simulation wird Angehalten. Start/Stop wird Grau.

4.2 Spielfeldeditor

Vorbedingung: Simulation ist immer gestoppt bei geöffnetem Spielfeldeditor.

Ablauf	Klick auf Bedienelemente außerhalb des Spielfeleditors.
Erwartetes Ergebnis	Spielfeldeditor wird Geschlossen.

Ablauf	Klick auf Laden oder Einfügen. Auswahl von Datei ohne Lesezugriff oder falschem Dateiformat.
Erwartetes Ergebnis	Programminterne Fehlermeldung ausgeben und Aktion nicht ausführen.

Ablauf	Klick auf Speichern. Auswahl von Ordner ohne Schreibzugriff, falschem Dateiformat, oder Partition ohne Ausreichenden Speicherplatz.
Erwartetes Ergebnis	Programminterne Fehlermeldung ausgeben und Aktion nicht ausführen.

Ablauf	Klick auf Speichern. Auswahl von mitgelieferter standard Datei.
Erwartetes Ergebnis	Fehlermeldung: Andere Datei wählen und Aktion nicht ausführen.

Ablauf	Klick auf Eingabefeld Größe. Eingabe von Nicht Positivem Integer Wert oder unzulässigen Datentyp. Klick auf anwenden.
Erwartetes Ergebnis	Fehlermeldung: nur Positive Integer zulässig. Aktion nicht ausführen.

Ablauf	Klick auf Anwenden ohne, dass die Textfelder mit 2 Positiven Integern gefüllt sind.
Erwartetes Ergebnis	Fehlermeldung: 2 Dimensionen benötigt. Aktion nicht ausführen.

4.2.1 Zufallsgenerator

Vorbedingung: Hauptfenster des Zufallsgenerators mit Tabelle ist geöffnet.

Ablauf	Klick auf Entfernen oder Bearbeiten, ohne dass Zeile ausgewählt ist.
Erwartetes Ergebnis	Fehlermeldung: Bitte Zeile auswählen. Keine Aktion.

Ablauf	Klick auf ok bei leerer Tabelle.
Erwartetes Ergebnis	Dialogfenster wird Geschlossen. Änderungen werden verworfen.

Vorbedingung: Unterfenster zum Ändern oder Hinzufügen geöffnet.

Ablauf	Klick auf ok bei leeren Textfeldern, nicht Positiven Integern in Anzahl, oder nicht Integern in mindestens einem der 3 Felder.
Erwartetes Ergebnis	Dialogfenster wird Geschlossen. Änderungen werden verworfen.

4.3 Regeleditor

Vorbedingung: Simulation ist immer gestoppt bei geöffnetem Regeleditor.

Ablauf	Klick auf Bedienelemente außerhalb des Regeleditors.
Erwartetes Ergebnis	Regeleditor wird Geschlossen.

Ablauf	Klick auf Speichern. Auswahl von mitgelieferter standard Datei.
Erwartetes Ergebnis	Fehlermeldung: Andere Datei wählen und Aktion nicht ausführen.

Ablauf	Klick auf Laden. Auswahl von Datei ohne Lesezugriff oder falschem Dateiformat.

Erwartetes Ergebnis	Programminterne Fehlermeldung ausgeben und Aktion nicht ausführen.

Ablauf	Klick auf Speichern. Auswahl von Ordner ohne Schreibzugriff, falschem Dateiformat, oder Partition ohne Ausreichenden Speicherplatz.
Erwartetes Ergebnis	Programminterne Fehlermeldung ausgeben und Aktion nicht ausführen.

4.4 Verarbeitung

4.4.1 Speichern und Laden

Ablauf	Öffnen eines Filechoosers wenn automatisch erzeugte Ordner oder Dateien gelöscht oder geandert wurden.
Erwartetes Ergebnis	Wiederherstellen der nicht vorhandenen Ordner und Dateien. Öffnen des Filechoosers.

4.4.2 Performance

Ablauf	Einstellen der Spielfeld Größe auf `3000 * 3000`. Ui Elemente Bedienen
Erwartetes Ergebnis	Angeforderte Operationen werden ohne störende Verzögerung (Max 1 Sekunde) ausgeführt.

Ablauf	Bei Standard Spielfeld Größe, Game of Life Regelsatz, 40 Prozent gefülltem Spielfeld und ohne Verzögerung Simulation starten.
Erwartetes Ergebnis	Simulation wird Flüssig (Mindestens 30 FPS) ausgeführt.

4.4.3 Stabilität

Ablauf	Bei Standard Spielfeld Größe, Game of Life Regelsatz, eine Glidergun auf dem Spielfeld und ohne Verzögerung Simulation starten und eine Stunde Laufen lassen.
Erwartetes Ergebnis	Nach einer Stunde läuft das Programm unverändert weiter und produziert immernoch Glider.

Ablauf	Erfahrene Testperson benutzt 1 Stunde lang alle Funktionen des Programms.
Erwartetes Ergebnis	Nach einer Stunde läuft das Programm noch stabil und ist nicht zwischendurch abgestürzt.

5 Entwicklungsumgebung

5.1 Verwendete Software

Betriebssysteme:	MacOS X, Windoof X, Linux X
Bildbearbeitung & Diaagramme	GIMP, Photoshop, Modelio
Programmierung & Versionierung	Eclipse, Eclipse Window builder, GIT

5.2 Verwendete Hardware

Intelligente Frühstücksbrettchen mit abwaschbarer Benutzeroberfläche verschiedener Hubraumklassen.

5.3 verwendete Organisation

Haben Sie wirklich den Eindruck, dass hier irgendwas organisiert abläuft? Aber gut, ein Versuch: Wenn etwas schief geht, hat Alex schuld. Wenn jemand Ahnung hat, dann Nico. Wenn jemand Protokoll schreibt, dann Felix. Wenn jemand gute Laune hat, dann Jörg. Wenn jemand Photoshop macht, dann Diaa.

Glossar

Begriff	Erklärung
Performance	Geschwindigkeit der Software.
JRE	Java Runtime Environment. Ein Stück frei erhältliche Software, die es Ermöglicht Java Programme auszuführen.
.csv	”Comma separated values“ simples Tabellen–Dateiformat. Trennung von Spalten durch Kommata und Zeilen durch Umbrüche.
.txt	Dateiendung für Textdateien.
Zellulärer Automat	Ein Konzept zur Modellierung dynamischer Systeme. Zellen die eine bestimmte Menge von zuständen einnehmen können befinden sich in einem Raum. Die Räumlich nächsten Zellen bilden die Nachbarschaft. Aus dem eigenen Zustand und dem der Nachbarn ergibt sich über eine Transitionsregel der Folgezus–tand.
Transitionsregel	Vorschrift die unter Verwendung vorhandener Daten den Zustand einer Zelle in den Nächsten überfuhrt.
Simulation	Dynamische Abbildung, meist realer Sachver–halte, anhand eines Modells, durch Anwen–dung des Modells über Zeit.
Spielfeld	Zweidimensionales Feld aus Zellen
Zelle	Ein zellular Automat (Zelle), der 2 hoch 32 Zustände annehmen kann.
Invers Polnische Notation	Beste Erfindung der Welt. Klammerfreie Nota–tion von Algorithmen, die auf einem Stack aus–geführt werden können.
Syntax	Regeln zur Anordnung und Reihenfolge von Zeichensystemen
Spielfeld	Ein abgegrenztes 2D Feld aus Zellen.

Multithreading	Aufteilen einer von Aufgaben auf mehrere Threads. In diesem Kontext lässt sich das Spielfeld in mehrere Bereiche unterteilen die dann Parallel bearbeitet werden können.
Thread	Logischer Prozessor
Zustand	Der Zustand einer Zelle (lebendig oder tot) in der Folgegeneration hängt nur vom aktuellen Zustand der Zelle selbst und den aktuellen Zuständen ihrer Nachbarzellen ab.
Nachbarn	Eine Auswahl an Zellen die über Ecken und Kanten mit der derzeit Betrachteten Zelle ver–bunden sind. Wie sonst unüblich kann man sich hier die Nachbarschaft aussuchen, indem man die Gewollten in der Regel verbaut, oder eben auch nicht. Wer seine Ruhe haben will, wird schnell merken, dass ein ruhiges Leben auch langweilig sein kann.
Nachbarschaft	Leute die auf MEINEM Parkplatz parken und den Hund mitten in der Nacht bellen lassen.
Moor Nachbarschaft	Benachbarte Zellen sind die, die über Ecken und Kanten verbunden sind.
Neumann Nachbarschaft	Benachbarte Zellen sind die, die über Kanten verbunden sind. 4 an der Zahl.
Hauptbenutzeroberfläche	Das was man sieht wenn man das Programm Startet. Enthält alle Editoren, das Spielfeld und die Knöpfe zur Steuerung der Simulation.
Spielfeldeditor	Enthält alle Einstellungen zum festlegen der Zellen Zustände und Größe.
Regeleditor	Enthält alle Einstellungen der Transisition–sregeln.