Guten Tag, heute mache ich eine Präsentation über unsere Entwurfsphase.

Letztes Mal hat Sophie in ihrer Präsentation schon gezeigt, unser Programm hat drei Eigenschaften: Konfigurationen erstellen, Simulationen starten und Ergebnisse anzeigen.

Zuerst brauchen wir natürlich die Klassen, um Konfigurationen und Elementen von Konfigurationen zu speichern. Und brauchen wir auch die Klassen, um Konfigurationen zu benutzen und auch Simulation zu laufen. Wir zeigen auch dem Benutzer die Ergebnisse an, deshalb wählen wir so ein MVC-Modell aus. Das Controller verwaltet das View und Model, in View liegen GUI Elemente, Klassen in View werden die Eingebe vom Benutzer temporär gespeichert, und überprüft ob Eingabe korrekt ist. Controller überträgt Objekte aus ViewModell in Modell. Controller erhält Action- und EventListener, erzeugt View und Modell Objekte, Programmflusssteuerung. In Modell gibt es Business Logic.

In Model schreiben wir Klassen für Algorithms, Condinations, Strategies, Action, Agent, Game, Group und so weit. Hier kommt ein ganz groß Klassendiagramm für Model.

Und hier ist das Diagramm für Controller, es gibt die ModelProvider Klassen, um Klassen aus Model zu benutzen. Hier gibt es auch FileManager Klassen, um Konfigurationen oder Ergebnissen zu speichern. Und auch AlgorithmLoader, um Algorithmen zu laden. Und SwingGuiFactory erzeugt View.

Und hier ist Diagramm für View, alle Klassen sind für Fenster. Hier ist für ViewModel,

Es ist ein Sequenzdiagramm, es zeigt, wenn Benutzer die Simulation startet, die Interaktion zwischen Klassen. Bevor SimulationStarten, muss ein ActionListener der MainView Klasse hinzugefügt werden. Und dann Benutzer den Start-Button drücken, dann sendet eine Sinal zu StartSimulationHandler Klasse, die MainView Klasse gibt die Name von Konfiguration für die Simulation, und die StartSimulationsHandler sucht die Konfiguration durch die Name aus, und dann erstellt eine Simulation und ein Observer, danach wird das Observer der Simulation hinzugefügt, dann läuft die Simulation, wenn die Simulation abgelaufen ist, ruft die notifyObserver Methode auf, die Simulation aktuliert die Zustand vom Observer, dann sendet das Observer eine Signal zurück, dann sendet die Simulation die Nachricht „done“ .

Und hier ist das Klassendiagramm zu diesem Sequenzdiagramm. Und hier benutzen wir ein Entwurfsmuster „Beobachter“, also warum wir den Entwurfsmuster benutzen? Der erste Grund ist, wir können die enge Kopplung zwischen Klassen vermeiden. Und die Änderung erhält das abhängige Objekt automatisch. In der Simulation ist der Agent ganz wichtig, ein Agent kann „kooperieren mit jemandem“ oder „ kooperieren nicht“ zwei Aktionen entscheiden. Verschiedener Agent kann verschiedene Strategie haben, um zwei Aktion in jeder Runde zu entscheiden. Z.B ein Agent kooperieren gern immer, der anderer kooperieren nie oder Tit for Tat usw. Es gibt zu viele verschiedene Strategie für verschiedenen Agent, dann entscheiden wir hier Entwurfsmuster „Strategie“. Es gibt noch weitere Entwurfsmuster, Z B Fassade bei Algorithms.

Am ende spreche ich noch einige vom uns verwendete Resourcen, wir haben JFreeChart, IntelliJ und Gson verwendet.

Vielen Dank für Ihre Aufmerksamkeit.