Institut für Informatik
Prof. Dr.-Ing. Elke Pulvermüller
Nils Baumgartner, M.Sc.

Universität Osnabrück, 15.12.2020
Testate vom 09.01.2023 bis zum 12.01.2023, 18:00 Uhr

Übungen zu Software Engineering

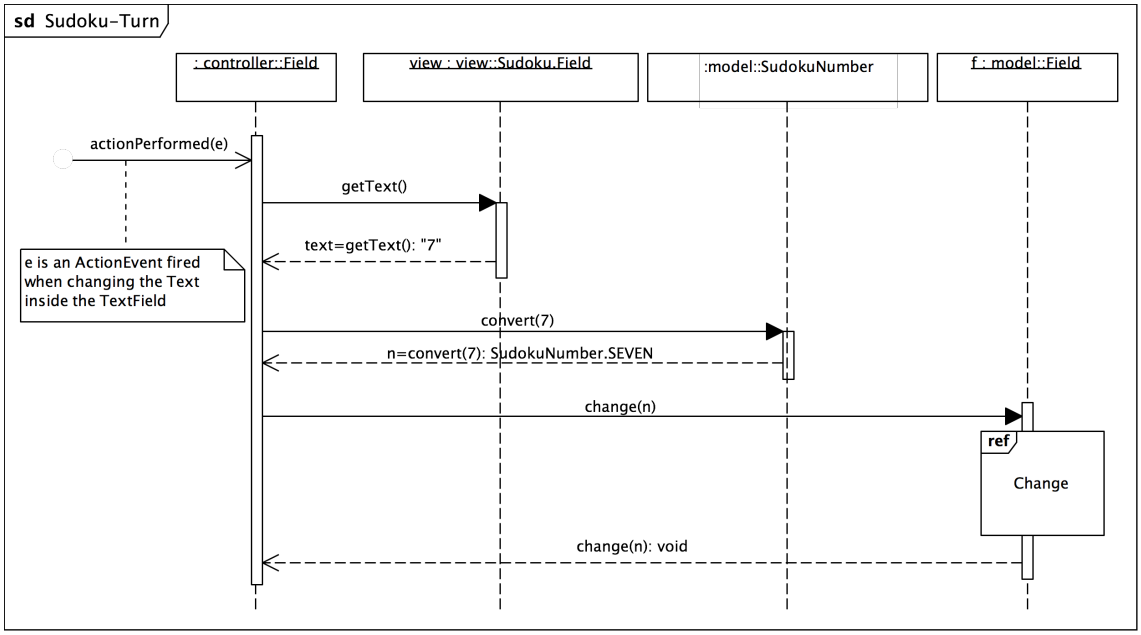
Wintersemester 2022/23

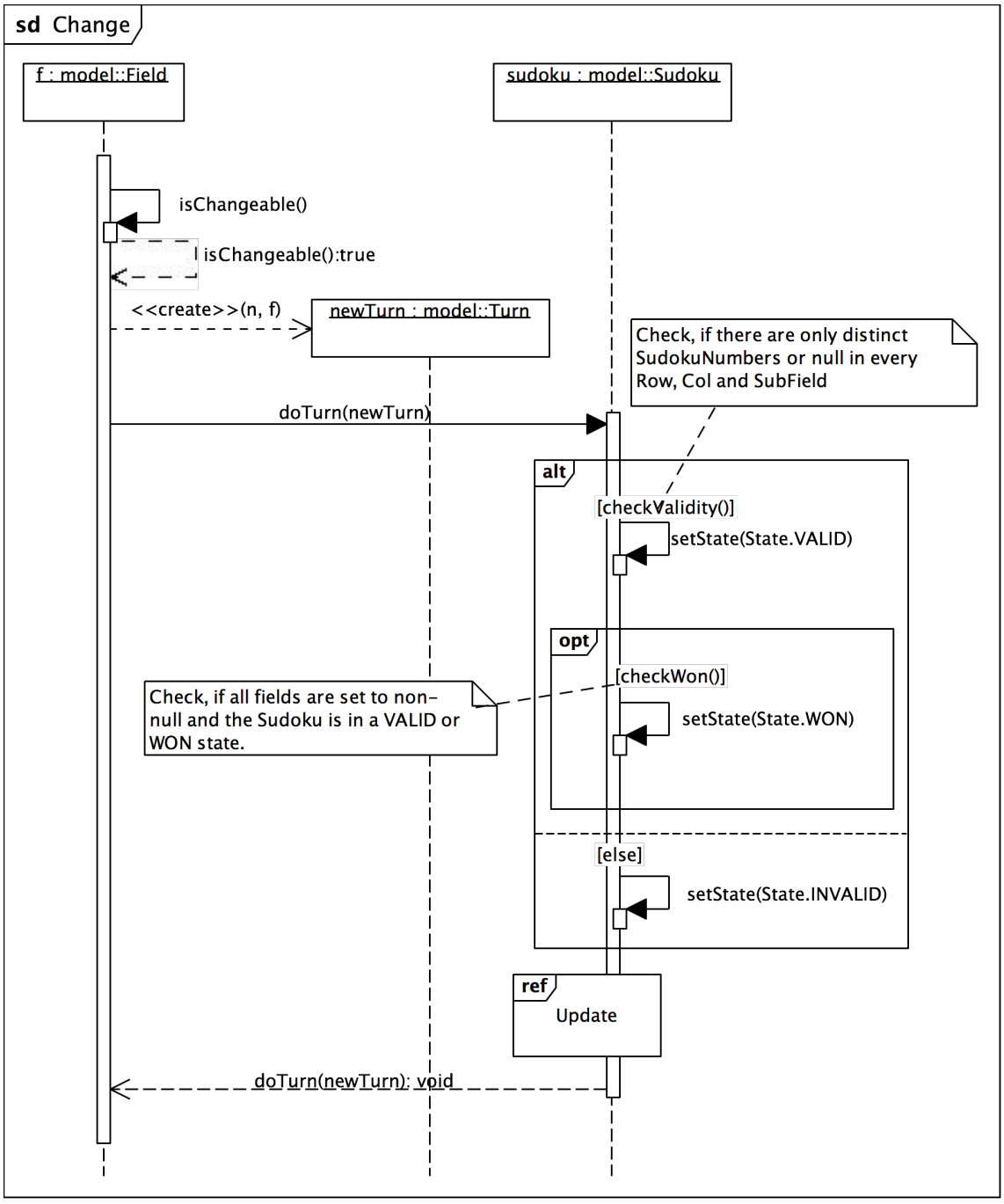
Blatt 9

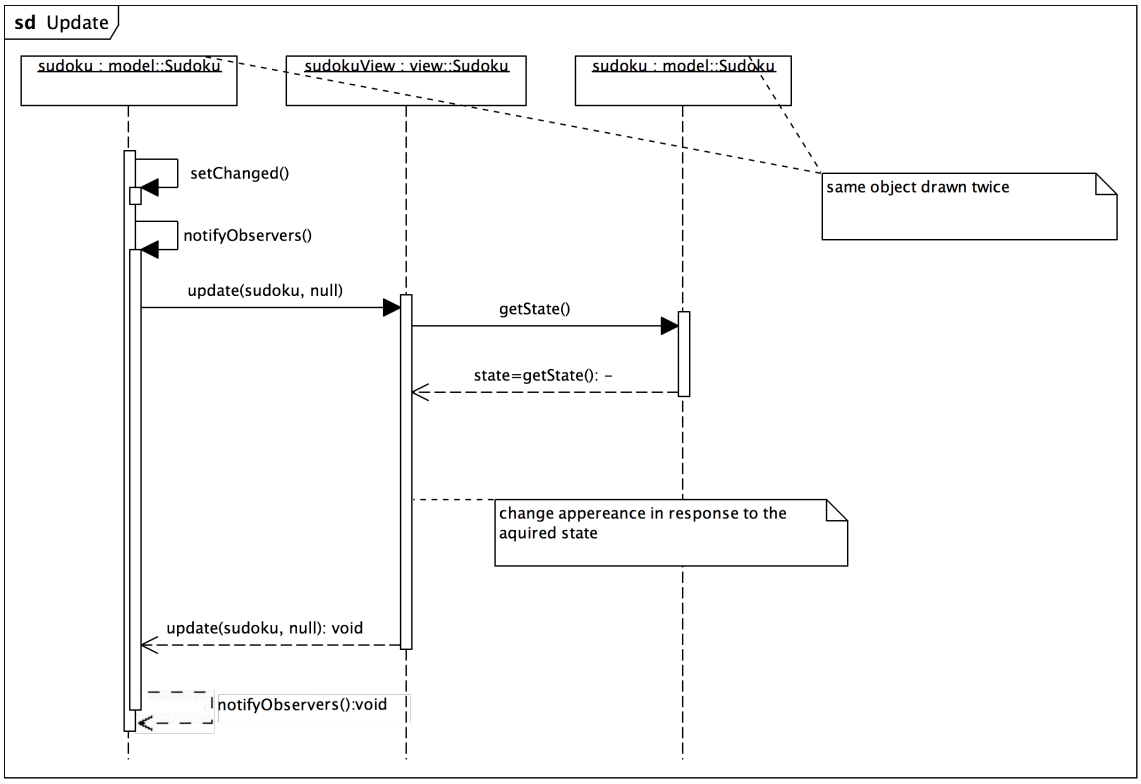
ACHTUNG! Die Testate zu diesem Aufgabenblatt finden Beginnen nach den Weihnachtsferien. Folglich wird das Blatt 10 erst im neuen Jahr veröffentlicht.

Aufgabe 9.1: Kommunikationsdiagramm (25 Punkte)

Betrachten Sie untenstehende Sequenzdiagramme, welche einen Spielzug im Spiel Sudoku modellieren. Stellen Sie diesen Ablauf als UML-Kommunikationsdiagramm dar.




sd Change
f : model::Field
sudoku : model::Sudoku
isChangeable()
isChangeable():true
<<create>>(n, f)
newTurn : model::Turn
Check, if there are only distinct SudokuNumbers or null in every Row, Col and SubField
doTurn(newTurn)
alt
[checkValidity()]
setState(State.VALID)
opt
[checkWon()]
setState(State.WON)
Check, if all fields are set to non-null and the Sudoku is in a VALID or WON state.
[else]
setState(State.INVALID)
ref
Update
doTurn(newTurn): void



Aufgabe 9.2: Aktivitätsdiagramm (45 Punkte)

Walter Faber plant seinen diesjährigen Urlaub. Auf seiner Hermes-Baby notiert er sich folgendes:

- Start am 15.06. um 8 Uhr
- Ich brauche: Schlüssel, Reiseunterlagen, Koffer
- Schlüssel beim Nachbarn abgeben
- Koffer in den Kofferraum vom Auto laden
- Navigationsgerät mit Hoteladresse programmieren (steht in den Reiseunterlagen)
- wenn alles erledigt ist und die Nachricht kommt, das der Urlaub genehmigt wurde - losfahren
- am Hotel parken und mit den Reiseunterlagen einchecken
- Koffer ins Hotelzimmer bringen

Ihm fällt auf, dass die Schlüsselabgabe sich noch wie folgt differenzieren lässt:

- zum Nachbarn gehen und klingeln
- maximal 3 Minuten warten, dann Schlüssel in den Briefkasten werfen
- wenn die Tür geöffnet wird, Schlüssel abgeben und verabschieden

Zeichnen Sie ein möglichst genaues Aktivitätsdiagramm basierend auf den Notizen von Herrn Faber. Beachten Sie dabei folgende Dinge:

- Die Gegenstände Schlüssel, Reiseunterlagen und Koffer sollen initial als Aktivitätsparameterknoten (siehe Skript) modelliert werden. Achten Sie im weiteren Verlauf Ihres Aktivitätsdiagramms auf eine angemessene Trennung zwischen Kontroll- und Datenfluss.

- Für den Kofferraum können Sie einen «centralBuffer» verwenden.
- Ihr Aktivitätsdiagramm muss einen Anfangs- und einen Endknoten haben.

Aufgabe 9.3: Zeitdiagramm (30 Punkte)

Am Ende eines jeden Asterix-Heftes wird für Obelix ein eigener Tisch zum Wildschweinessen aufgebaut. An der einen Seite des Tisches werden die Schweine gebraten und dann auf den Tisch gestellt, an der anderen Seite sitzt Obelix und isst so schnell wie möglich die aufgetischten Schweine. Im Paket `Wildschwein` wird dieser Sachverhalt als *Producer* (`Schweinebrater`) -*Consumer* (`Obelix`) - Beziehung dargestellt, wobei der *Store* (`Tisch`) eine Kapazität von 10 (Wildschweinen) hat.

Betrachten Sie die ausführbare Klasse `Fest` und stellen Sie in einem UML-Zeitdiagramm dar, wie die einzelnen Objekte miteinander kommunizieren und welche Zustandswechsel Sie dabei durchlaufen. Es reicht aus, wenn Sie sich wiederholende Situationen nur einmal zeichnen und dann zusammen mit der Anzahl der Wiederholungen kennzeichnen. Hierfür können Sie ein Rechteck um den sich wiederholenden Bereich zeichnen und angeben wie häufig dieser Bereich ausgeführt werden soll.

Die konkrete Ausführungsreihenfolge des Programms hängt aufgrund der Nebenläufigkeit vom Java-Scheduler ab. Da Sie in diesen keinen Einblick haben, können Sie selbst entscheiden welche Nachricht Sie zuerst darstellen, wenn es zu potenziell zeitgleichen Ereignissen kommt. Ihr Zeitdiagramm soll sich vom Start des Programms bis zum Erreichen eines stabilen Zustands erstrecken, d.h. bis Sie mithilfe des oben erwähnten Wiederholungskastens eine korrekte Endlosschleife angeben können.

Hinweis: Die Zustände eines Java-Threads sind in der Dokumentation der Enumeration `Thread.State` beschrieben (`https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/Thread.State.html`).