Karlsruhe Institute of Technology

Institut für Theoretische Informatik
Anwendungsorientierte Formale Verifikation

Vorlesung Formale Systeme 2 – Anwendung, SS 2017, 2.6.17
Prof. Dr. B. Beckert, Dr. M. Ulbrich

Formaler Entwurf mit Event-B – Die Eventbank

Ziel dieser Aufgabe ist es, dass Sie selbständig ein Event-B-Modell samt einiger Verfeinerungen entwerfen und verifizieren. Es soll dazu ein kleiner Ausschnitt aus der Beziehung zwischen Banken und ihren Kunden modelliert werden. Ein Szenarium, das auch später im Praktikum noch einmal im Teil über die Modellierung mit JML aufgegriffen werden wird.

In fünf Verfeinerungsschritten wollen wir einige Aspekte dieser Beziehung und/oder verschiedene Arten von Operationen modellieren:

M0 Generelle Beziehung zwischen Banken und Kunden

M1 Betrachtung von Konten und deren Einrichtung

M2 Hinzuziehen von Kontoständen für die Konten

M3 Limitieren der Kontostände nach unten durch Einführung eines Überziehungslimits.

M4 Modellierung von Bankkarten und -automaten

M5 Hinzunehmen der Authentifizierung per PIN.

Die Anforderungen, die in den einzelnen Verfeinerungen umgesetzt werden sollen, entnehmen Sie bitte den Abschnitten *M0* bis *M5* dieses Dokumentes. Natürlich lassen die natürlichsprachlichen Anforderungen Platz für verschiedene Art der Interpretierung, und die Umsetzung in den formalen Kontext kann verschieden lauten. Treffen Sie daher ggf. Entwurfsentscheidungen und seien bereiten Sie darauf vor, diese Entscheidungen bei der Abnahme der Aufgabe zu verteidigen.

Erstellen Sie in RODIN ein neues Projekt für diese Aufgabe. Benennen Sie Kontexte, Maschinen, Variablen, ... stets mit sprechenden Namen, um Ihre Modellierung verständlicher zu machen. Entlasten Sie alle aufgekommenen Beweisverpflichtungen.

Einschränkung: Sie müssen in dieser gesamten Aufgabe für neu eingeführte Ereignisse **nicht** zeigen, dass diese konvergent sind, i. Allg. sind sie es nämlich nicht. Ebenso wollen wir keine Beweise für Verpflichtungen der Art "Feasibility" oder "Deadlockfreeness" führen.

Allgemeine Hinweise

Die Anforderungen, die in den folgenden Abschnitten eingeführt werden, sagen nichts über die Art und Weise aus, in der das Modell initialisiert werden soll, also über jenen Zeitpunkt, zu dem alle Banken "eröffnet" worden sind. Wählen Sie dafür jeweils sinnige Annahmen und definieren Sie Ihre Initialisierungsereignisse entsprechend. Natürlich müssen Sie sicherstellen, dass diese Ereignisse auch Ihre Invarianten etablieren.

Viele Ereignisse beruhen auf Eingaben. Wählen Sie diese mit Bedacht, nicht immer ist aus dem Fließtext sofort ersichtlich, welches die Eingabegrößen sind. Es kann zum Beispiel sein, dass ein Kunde ebenfalls Parameter eines Ereignisses sein soll, auch wenn er aus der Kontoinformation heraus berechnet werden könnte.

M0: Banken und Kunden

Wir beginnen mit dem gröbsten Modell, das zunächst einmal nur zwei Typen von Objekten in Beziehung zu einander setzt:

Personen und Banken können in einem Kundschaftsverhältnis zu einander stehen.	REQ-1

Wir betrachten Personen als abstrakte Objekte ohne weiteren Eigenschaften (Name, Alter, Adresse). Banken sind ebenfalls Objekte, zunächst ohne weitere Eigenschaften. Daher sind für die Modellierung dieser Gesamtheiten entsprechende nicht weiter eingeschränkte Trägermengen in einem Kontext sinnvoll.

Wir nennen Personen, die mit einer Bank in Beziehung stehen, "Kunden" dieser Bank. Für diese Beziehung gilt:

Eine Person kann Kunde bei beliebig vielen Banken sein (einschließlich bei keiner).	REQ-2

Eine Bank kann beliebig viele Kunden haben (auch keine).	REQ-3

Diese Aussage kann beispielsweise durch eine neue Variable, die die Beziehung zwischen Banken und ihren Kunden widergespiegelt, dargestellt werden. Eine Invariante schränkt die Variable dabei in ihrem Wertebereich ein.

Als Operation wollen wir auf dieser Ebene zunächst nur zulassen, dass eine Person Kunde einer Bank werden kann. Das Beenden einer Kundenbeziehung wollen wir, um den Rahmen überschaubar zu halten, in dieser Aufgabe ausblenden.

Eine Person kann jederzeit Kunde bei einer Bank werden, wenn sie nicht bereits Kunde derselben Bank ist.	REQ-4

M1: Konten

In dieses erste Modell wollen wir nun in einer Verfeinerung eine *Indirektion* einbauen, d.h. die ursprüngliche Relation zwischen Kunden und Banken soll nun über einen Zwischenschritt beschrieben werden: Bankkonten. Wir wollen nicht näher festlegen, welcher Natur diese Konten sind (Girokonto, Sparkonto, ...) und betrachten die Menge aller möglichen Konten als gegeben.

Die Kundenbeziehung zu einer Bank aus der vorherigen Maschine koppeln wir nun an den Besitz wenigstens eines Kontos bei ihr:

Jeder Kunde einer Bank hat bei dieser Bank ein oder mehrere Konten.	REQ-5

Und das soll in beide Richtungen gelten. Eine kontolose Mitgliedschaft bei einer Bank wird also ausgeschlossen.

Eine Person ist Kunde einer Bank, wenn sie wenigstens ein Konto dort besitzt.	REQ-6

Damit wird die Kundschaftsrelation aus der vorherigen Ebene ersetzt durch entsprechende Aussagen über die Kontenrelationen. Die Konten, die hier betrachtet werden, sind einfach: Sie können nicht mehreren Personen gehören und sind fest bei einer Bank angesiedelt.

Ein Bankkonto gehört genau einer Person.	REQ-7

Jedes Konto gehört zu genau einer Bank.	REQ-8

Anders als bei *M0*, wo der Beitritt nur zum Zeitpunkt der Beziehungsaufnahme erfolgen konnte, ist jederzeit die Erstellung eines neuen Kontos möglich.

Eine Person kann jederzeit ein neues Konto bei einer Bank eröffnen.	REQ-9

Durch Kontoeröffnung wird eine Person zum Kunden der Bank, wenn sie es nicht bereits ist. Es ist klar, dass das erste Konto zu eröffnen aus Sicht der Verfeinerung eine andere Qualität hat als ein weiteres Konto zu eröffnen.

Die folgende letzte Anforderung dieser Verfeinerung ist mehr technischer Natur und ist eigentlich selbstverständlich, muss aber modelliert werden.

Ein bereits existierendes Konto kann nicht erneut eröffnet werden.	REQ-10

M2: Kontostände

Jedes Konto hat einen Saldo.	REQ-11

Mit der Einführung von Konten ist es sehr naheliegend, auch Kontostände zu modellieren.

Der Saldo wird als ganze Zahl von Geldeinheiten gespeichert.	REQ-12

RODIN unterstützt keine gebrochenen Zahlen, daher muss der Betrag z. B. in Cents statt in Euros gespeichert werden. Beachten Sie, dass diese Modellierung auch die negativen Zahlen einschließt:

Konten können überzogen werden.	REQ-13

Da neue Konten eingeführt werden können, ist eine Forderung über den initialen Kontostand von Nöten:

Neu eingerichtete Konten sind zu Beginn leer.	REQ-14

Unter "leer" ist diesem Kontext zu verstehen, dass sie einen Saldo von 0 Geldeinheiten haben.

Wo wir nun die Kontostände zur Verfügung haben, liegt es nahe, neue Ereignisse zu definieren, die diese Stände verändern können.

Kunden können beliebige positive Beträge von einem ihrer eigenen Konten abheben.	REQ-15

Kunden können beliebige positive Beträge auf eines ihrer eigenen Konten einzahlen.	REQ-16

Diese Vorgänge sind jedoch beschränkt auf die Konten, die einer Person wirklich zugeordnet sind. Es soll unmöglich sein, Geld von einem fremden Konto abzuheben.

Über die Authentifizierung der Person wird zu diesem Zeitpunkt noch nichts gesagt, dies folgt in einer späteren Verfeinerung.

Personen können nicht Geld von Konten anderer Personen abheben oder darauf einzahlen.	REQ-17

Neben Abheben und Einzahlen gibt es noch die Überweisungs-Aktion, an der zwei Konten beteiligt sind, aber nur das Ausgangskonto der Person selbst gehören muss. Man kann also beliebigen anderen Personen mit Konto Geld überweisen.

Kunden können von ihren Konten Geld auf existierende Konten überweisen.	REQ-18
Bei einer Überweisung verringert sich der Saldo des Ausgangskonto um einen Betrag. Um den selben Betrag erhöht sich der Saldo des Zielkontos.	REQ-19
Kunden können nicht von Konten anderer Personen Geld überweisen.	REQ-20

Diese Anforderungen alleine reichen nicht aus, um den Vorgang einer Überweisung vollständig zu spezifizieren. Eine Aussage über das Verhältnis von Ausgangs- und Zielkonto ist nötig, um alle benötigten Beweise führen zu können. Füllen Sie Anforderung REQ-21 entsprechend aus.

	REQ-21

Die Modellierung in einer formalen Umgebung dient oft nicht nur dazu, gegebene Anforderungen formal umzusetzen, sondern auch dazu – wie hier geschehen –, die Anforderungen zu finden bzw. weiter auszuarbeiten.

M3: Überziehungslimit

Im Bankenwesen ist es bekanntermaßen üblich, dass Konten nur in einem gewissen Rahmen überzogen werden dürfen. Diesen Sachverhalt wollen wir in diesem Verfeinerungsschritt mit in unser Modell einfließen lassen.

Vereinfachend nehmen wir an, dass es einen festen, für alle Banken geltenden Überziehungsrahmen gibt, den kein Konto zu keinem Zeitpunkt sprengen darf.

Es gibt einen Minimalbetrag, der für alle Konten aller Kunden bei allen Banken derselbe ist.	REQ-22
Kein Konto darf zu irgendeinem Zeitpunkt weniger Saldo aufweisen als der Minimalbetrag.	REQ-23
	REQ-24

Für den Überziehungsrahmen gibt es eine Eigenschaft, die gelten muss, damit die hier aufgebaute Modellierung funktionieren kann. Finden Sie diese Forderung an den Minimalbetrag und tragen Sie sie als REQ-24 ein.

M4: Bankautomaten und Bankkarten

Wie bereits angekündigt wollen wir uns nun dem Thema der Authentifizierung etwas nähern. Bekanntermaßen wird dies mit Hilfe von Scheck-/Kredit-/Bankkarten erledigt, die einem bestimmten Konto zugeordnet werden. Nur wer im Besitz einer Karte für ein Konto ist, darf von diesem Konto an einem Geldautomaten Geld abheben.

Es existieren Bankkarten für Konten.	REQ-25
Jede Bankkarte ist genau einem Konto zugeordnet, wobei es mehrere Karten für ein Konto geben kann (oder auch gar keine).	REQ-26

Es können jederzeit neue Karten für existierende Konten ausgegeben werden.	REQ-27

Wer im Besitz einer Karte ist, kann an einem Geldautomaten Geld vom zugehörigen Konto abheben.	REQ-28

Dieses ist offensichtlich eine Verfeinerung der Operation "Geld abheben" der abstrakteren Ebene. Eine der voranstehenden Anforderungen steht aber im Widerspruch zu REQ-17. **Welche Anforderung erzeugt den Widerspruch?** Wir schwächen deshalb REQ-17 ab zu

Personen können *am Schalter* nicht Geld von Konten anderer Personen abheben oder darauf einzahlen.	REQ-17a

M5: Authentifizierung per PIN

Im abschließenden Schritt soll modelliert werden, dass es für jede Karte genau eine vierstellige Zahl gibt, die für jede Karte zufällig ausgewählt wird. Wir wollen diese zufällige Wahl nicht modellieren, sondern nehmen an, dass es eine fixe Zuordnung von Karten zu PINs gibt. Daher wird dieses gegebene Verhalten besser in einem Kontext als in einer Maschine modelliert.

Am Bankautomaten muss der Kunde sich mit seiner (bzw. der Karte) PIN authentifizieren. Wird eine PIN zu häufig falsch eingegeben, wird die Karte gesperrt, es kann kein Geld mehr abgehoben werden.

Jede Karte hat eine unveränderliche, vierstellige PIN.	REQ-29

Natürlich können verschiedene Karten dieselbe haben, jedoch nicht eine Karte verschiedene PINs.

Beim Abheben am Automaten muss sich der Karteninhaber durch Eingabe der PIN authentifizieren, ansonsten schlägt der Abhebeversuch fehl.	REQ-30

Die Eingabe als Vorgang wird nicht weiter in einzelne Schritte zerlegt. Die PIN wird als vierstellige Zahl direkt als Eingangsgröße für das Ereignis "Am Bankautomaten abheben" verwendet.

Wird für eine Karte dreimal in Folge (also bei drei aufeinander folgenden Abhebeversuchen) eine falsche PIN eingegeben, so ist die Karte danach gesperrt.	REQ-31

In der Realität wird eine Karte in solch einem Fall eingezogen, das wollen wir nicht nachbauen. Vielmehr soll die Karte so markiert werden, dass ein Abheben mit ihr nicht mehr möglich ist.

Mit einer gesperrten Karte kann kein Geld abgehoben werden.	REQ-32

Natürlich soll eine Karte wieder aktiviert werden können. Dazu müssen jedoch die antragstellende Person und der Kontoinhaber identisch sein.

Ein Kunde kann eine gesperrte Karte, die zu einem seiner Konten gehört, wieder freischalten lassen.	REQ-33