

TU Wien – Institut für Computertechnik

Projekttitel

Software Design Description (SDD)

Gruppe	Autor(en)	Matrikelnr.
TRAD Gruppe X	NACHNAME Vorname	XXXXXXX
	NACHNAME Vorname	XXXXXXX
Ausgabe-Datum: 21.1.2021		Version: 2.3

0 INHALTSVERZEICHNIS

0	Inhaltsverzeichnis	1
0.1	Versions-Historie	3
1	Einleitung	4
2	Beschreibung des Gesamtsystems	5
2.1	Schnittstellenbeschreibung des Gesamtsystems nach außen	5
2.2	Beschreibung der Architektur.....	6
3	Beschreibung des LAYER UI.....	9
3.1	Schnittstellenbeschreibung	9
3.2	Beschreibung der Struktur	10
3.3	Beschreibung des Verhaltens	11
3.3.1	Inbetriebnahme	12
3.3.2	Authentisierung des Benutzers.....	13
3.3.3	Auslösen eines Alarms	14
3.3.4	Entriegeln.....	15
3.3.5	Benutzer erstellen	16
4	Beschreibung des LAYER BO	17
4.1	Schnittstellenbeschreibung	17
4.2	Beschreibung der Struktur	18
4.3	Beschreibung des Verhaltens	20
4.3.1	Inbetriebnahme	20
4.3.2	Authentisierung des Benutzers.....	21
4.3.3	Auslösen eines Alarms	22
4.3.4	Entriegeln.....	23
4.3.5	Benutzer erstellen	24
5	Beschreibung des LAYER DATA	26
5.1	Schnittstellenbeschreibung	26
5.2	Beschreibung der Struktur	27
5.3	Beschreibung des Verhaltens	28
5.3.1	Inbetriebnahme	28
5.3.2	Authentisierung des Benutzers.....	29
5.3.3	Auslösen eines Alarms	29
5.3.4	Entriegeln.....	30

5.3.5 Benutzer erstellen	30
6 Verweise auf andere Dokumente.....	31
7 Annex.....	31
8 Definition von Begrifflichkeiten.....	31

0.1 Versions-Historie

Version	Datum	Grund der Erstellung/Änderung
1.0	2020-12-04	Erstellen des Dokuments
1.1	2020-12-06	Hinzufügen der Beschreibung des Gesamtsystems und der Layer UI
1.2	2020-12-09	Hinzufügen der Beschreibung der Layer BO
1.3	2020-12-12	Hinzufügen der Beschreibung der Layer DATA
1.4	2020-12-13	Letzte Änderungen für die Zwischenabgabe
2.0	2020-12-22	Hinzufügen der Sequenzdiagramme der Layer BO und Layer DATA
2.1	2021-01-06	Überarbeiten der Sequenzdiagramme
2.2	2021-01-07	Review des gesamten Dokuments für die Endabgabe
2.3	2021-01-11	Letzte Änderungen für die Endabgabe

Tabelle 1: Versions-Historie

1 EINLEITUNG

Der Auftraggeber, im folgenden auch „Kunde“ genannt, ist ein Hersteller von Safes. In der Vergangenheit wurden wiederholt Beeinträchtigungen der Sicherheit des Systems gemeldet, die darauf zurückzuführen waren, dass der Schlüssel in unmittelbarer Nähe zum Safe gelagert wurde.

Durch die Umstellung des Verriegelungsmechanismus von traditionellen Schlüsseln auf RFID-Karten soll die Sicherheit des Systems gesteigert werden. Das Ziel ist, durch die Entriegelung per RFID-Karte, die Usability für den Endkunden zu verbessern. Dadurch soll die Anzahl der Fälle, in denen der Schlüssel in der Nähe des Safes gelagert wird, verringert werden.

Zusätzlich wird die Kontrolle für den Endkunden, durch diverse zusätzliche Funktionalitäten, wie einem eigenen Benutzer-Management System sowie einem Logging-Service, gesteigert.

Bei der Umsetzung ist besonders auf die Integrität des Systems zu achten.

Dieses Dokument stellt eine Repräsentation der, im Rahmen des Auftrags, zu entwickelnden Lösung dar. Im Folgenden wird die Architektur des Gesamtsystems und dessen Schnittstellen bestimmt. Im weiteren Verlauf werden die einzelnen Komponenten des Systems genauer beschrieben und deren Verhalten definiert.

2 BESCHREIBUNG DES GESAMTSYSTEMS

Das Gesamtsystem umfasst den Safe, inklusive der darin verbauten Hardware, sowie die zu entwickelnde Lösung „Safe RFID“, zur Steuerung des Safes.

Im Folgenden werden die Schnittstellen, zur Kommunikation mit dem Gesamtsystem, sowie die Architektur dessen beschrieben.

2.1 Schnittstellenbeschreibung des Gesamtsystems nach außen

Die Kommunikation mit dem System, vor Ort, erfolgt über ein Eingabepanel, welches an der Vorderseite des Safes angebracht ist. Hierbei handelt es sich um einen Touchscreen, auf dem ein User Interface verfügbar ist, dass die Steuerung des Safes erlaubt.

Außerdem bietet die zu entwickelnde Lösung eine REST-Schnittstelle, um dem Hersteller, Systemadministratoren und weiteren Applikationen die Kommunikation mit dem System zu ermöglichen. Der Datenaustausch, über die REST-Schnittstelle, ist nur über das Kommunikationsprotokoll HTTPS möglich, um den unverschlüsselten Austausch von Daten zu vermeiden.

Unter anderem erfolgt die Kommunikation der Layer UI mit der Layer BO ebenfalls über die REST-Schnittstelle.

Die Kommunikation der zu entwickelnden Lösung, mit den Hardware-Elementen, erfolgt über eine Hardware Abstraction Layer (HAL). Der Informationsaustausch ist bidirektional und basiert auf Events, die über das Prinzip des Polymorphismus für den jeweiligen Use-Case optimiert sind. Sowohl die Hardware Abstraction Layer als auch die zu entwickelnde Lösung, bieten Eventhandler, um auf spezifische Ereignisse reagieren zu können.

Um alle relevanten Systeme im Falle eines Ereignisses zuverlässig informieren zu können, wird zur Kommunikation durch Events, das Observer Pattern genutzt.

2.2 Beschreibung der Architektur

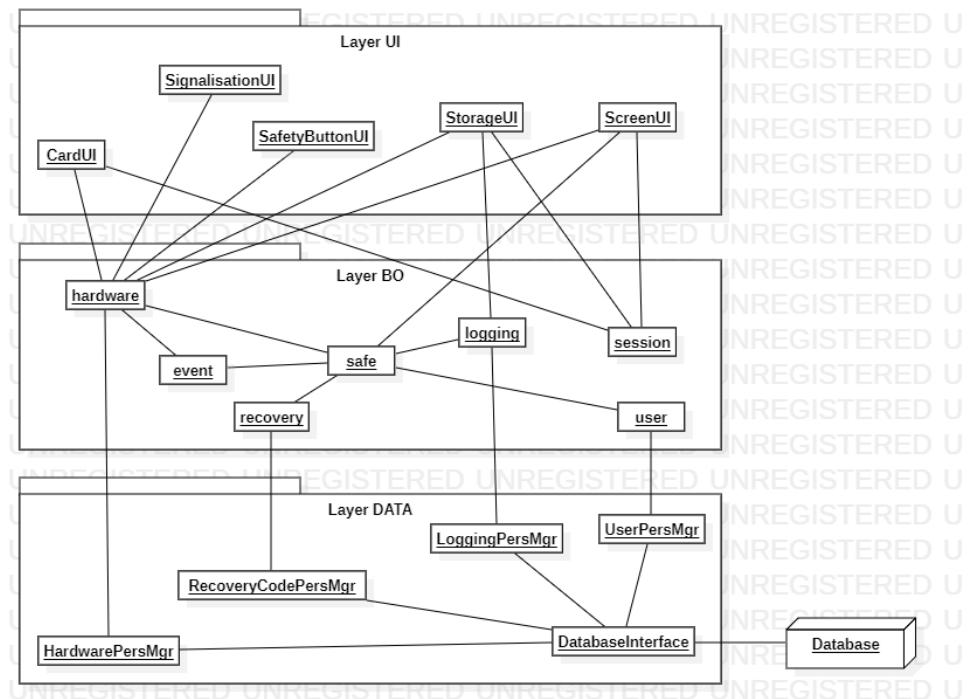


Abbildung 2-1: Architektur des Gesamtsystems

Die Architektur des Gesamtsystems orientiert sich an der Drei-Schichten-Architektur. Diese unterteilt die Software in die Schichten User-Interface, Business-Objects und die persistente Datenschicht.

Die User-Interface-Schicht ist die oberste Schicht und ist für die Kommunikation zwischen dem Benutzer und dem System verantwortlich. Sie beinhaltet unterschiedliche Klassen, die jeweils zur Darstellung bestimmter Informationen und Eingaben konzipiert sind.

Die User-Interface-Schicht besteht aus fünf (5) unterschiedlichen Modulen.

- CardUI: Das CardUI-Modul dient dem Einlesen der an das RFID-Lesegerät angelegten RFID-Karte. Über entsprechende Kommunikations-Methoden wird in weiterer Folge die jeweilige Benutzer-Kennung an das Gesamtsystem weitergereicht.
- SignalisationUI: Die Ansteuerung der im Safe verbauten LEDs und des Lautsprechers, finden über das SignalisationUI-Modul statt.
- SafetyButtonUI: Das SafetyButtonUI-Modul dient der Reaktion, im Falle der Betätigung des Notfalltasters, im Inneren des Safes. Bei Betätigung wird ein entsprechendes Event an das Gesamtsystem gesendet.
- ScreenUI: Die Visualisierung des Displays sowie das Auslesen des Eingabepanels wird über dieses Modul bewerkstelligt.
- StorageUI: Die Visualisierung am Display basiert auf dem MVC Pattern (Model View Controller Pattern), wobei sich das ScreenUI-Modul um die Visualisierung und das StorageUI-Modul um die Modell-Schicht kümmert.

Für eine genaue Beschreibung der User-Interface-Schicht siehe Punkt 3.

Die Business-Objects-Schicht beinhaltet die Logik der Software und somit die eigentliche Funktionalität des Gesamtsystems. Die darüber liegende User-Interface-Schicht kommuniziert mit dieser Schicht, um die verarbeiteten Daten aus der persistenten Datenschicht zu erhalten und diese zu modifizieren. Die Business-Objects-Schicht kann auch ohne die User-Interface-Schicht agieren.

Die Business-Objects-Schicht setzt sich aus sieben (7) unterschiedlichen Modulen zusammen:

- Safe: Das Safe-Modul ist für den operativen Betrieb des Safes zuständig. Es beinhaltet Operationen, um den Programmablauf des Safes zu starten, diesen in einen gesperrten Zustand zu versetzen oder einen Alarm auszulösen.
- Session: Das Session-Modul dient der Verwaltung der aktiven Benutzer-Sessions. Benutzer können sich über das Modul authentifizieren, sich ausloggen und die Validität eines Benutzer-Tokens kann ermittelt werden.
- User: Die Verwaltung aller, am System registrierten, Benutzer, findet über das User-Modul statt. Dieses dient der Speicherung der Informationen aller, am System registrierten, Benutzer, der Registrierung neuer Benutzer und der Aktivierung, beziehungsweise Deaktivierung, bestehender Benutzer.
- Logging: Über das Logging-Modul werden zu sicherheitsrelevanten Vorgängen, die am System stattfinden, entsprechende Log-Einträge erstellt und in weiterer Folge gespeichert. Es ist ebenfalls möglich die existierenden Log-Einträge über besagtes Modul abzufragen.
- Hardware: Die eventbasierte Kommunikation zwischen dem Gesamtsystem und der im Safe verbauten Hardware, findet über das Hardware-Modul statt. Dieses umfasst Operationen, um den aktuellen Status der Safe-Türe auszulesen, diese zu verriegeln beziehungsweise zu entriegeln und die im Safe verbauten LEDs sowie den verbauten Lautsprecher zu aktivieren beziehungsweise zu deaktivieren.
- Event: Das Event-Modul umfasst diverse Klassen, um die eventbasierte Kommunikation zwischen dem Gesamtsystem und der im Safe verbauten Hardware zu ermöglichen. Dabei wird zwischen unterschiedlichen Events differenziert. Inkludiert ist ebenfalls ein System zur Verwaltung aufgetretener Events.
- Recovery: Die Verwaltung der Wiederherstellungs-Codes findet über das Recovery-Modul statt. Es beinhaltet Operationen, um ein neues Set an Wiederherstellungs-Codes zu generieren, einzelne Wiederherstellungs-Codes zu invalidieren und die Validität eines Wiederherstellungs-Codes abzufragen.

Für eine genaue Beschreibung der Business-Objects-Schicht siehe Punkt 4.

Die persistente Datenschicht dient der Archivierung systemrelevanter Daten. Das Speichern der Daten findet in einer Datenbank statt. Die einzelnen Klassen der persistenten Datenschicht ermöglichen das Speichern und Auslesen von Daten in, beziehungsweise aus, der Datenbank.

Die persistente Datenschicht umfasst fünf (5) unterschiedliche Module:

- UserPersMgr: Das User-Persistenz-Manager-Modul ist für das Speichern der Informationen, zu den registrierten Benutzern, verantwortlich. Diese umfassen die eindeutige Kennung eines registrierten Benutzers, das persönliche Passwort und den Aktivierungsstatus. Es werden Operationen geboten, um neue Benutzer zu erstellen, bestehende Benutzer zu aktualisieren und Informationen zu einem oder mehreren Benutzern abzufragen.
- LoggingPersMgr: Das Logging-Persistenz-Manager-Modul inkludiert diverse Operationen, um Log-Einträge anzulegen und bestehende Einträge aus der Datenbank auszulesen. Log-Einträge beinhalten Informationen bezüglich dem Auftrittsdatum, der Nachricht, der Priorität und dem Benutzer, der den Log-Eintrag ausgelöst hat.
- HardwarePersMgr: Informationen bezüglich den im Safe verbauten Hardware-Elementen, werden mit Hilfe des Hardware-Persistenz-Manager-Moduls gespeichert. Dabei werden Operationen zum Auslesen, Erstellen, Aktualisieren und Löschen einzelner Hardware-Elemente zur Verfügung gestellt.
- RecoveryCodePersMgr: Die Verwaltung der persistenten Wiederherstellungs-Codes erfolgt durch das Recovery-Code-Persistenz-Manager-Modul. Über dieses werden neue Wiederherstellungs-Codes gespeichert, bereits existierende Codes abgefragt und die Validität bestehender Wiederherstellungs-Codes aktualisiert.
- DatabaseInterface: Die Kommunikation der jeweiligen Persistenz-Manager-Module, mit der Datenbank, findet über das Database-Interface-Modul statt. Dieses beinhaltet den Connection-String, um die Verbindung zu der Datenbank aufzubauen. Über die Methode „performRequest“ können in weiterer Folge Anfragen an die Datenbank gesendet werden.

Für eine genaue Beschreibung der persistenten Datenschicht siehe Punkt 5.

3 BESCHREIBUNG DES LAYER UI

Die User-Interface-Schicht stellt die Schnittstelle zwischen einem Benutzer mit dem Gesamtsystem dar. Sie stellt eine visuelle Schnittstelle zur Verfügung, über die Benutzer mit der Business-Objects-Schicht interagieren können. Aus diesem Grund muss das User Interface übersichtlich sein und nur die notwendigen Informationen darstellen.

Zu möglichen Eingaben eines Benutzers, über das User Interface, zählen Eingabe eines Codes (zur Authentifizierung oder zum Zurücksetzen auf den Werkszustand) sowie das Auswählen einer gewünschten Menüfunktion. Eine weitere mögliche Interaktion, mit dem Gesamtsystem, stellt das Anlegen einer RFID-Karte dar.

Das User Interface dient der Darstellung der, für einen Benutzer verfügbaren Funktionen, wichtige Informationen, aber auch Fehlermeldungen und Benachrichtigungen. Die Signalisierung des Verriegelungsstatus erfolgt über die, auf der Vorderseite des Safes verbauten, LEDs. Ein aktiver Alarm wird durch fünf rot blinkende LEDs in Kombination mit einem akustischen Signal, bekanntgegeben.

3.1 Schnittstellenbeschreibung

Die graphische Anzeige am Eingabepanel stellt die visuelle Schnittstelle des menschlichen Benutzers, mit der „Safe RFID“ Software, dar. Der Datenaustausch, mit der Business-Objects-Schicht erfolgt über REST-Anfragen. Dies beinhaltet sowohl das Holen von Daten als auch das Mitteilen von Änderungen, die über das User-Interface vorgenommen werden.

Das User Interface basiert auf der JavaScript-Laufzeitumgebung Node.js. Zur optimierten Darstellung der Informationen und Interaktionsmöglichkeiten, wird das Frontend-CSS-Framework Bootstrap genutzt.

Die Kommunikation mit der Business-Objects-Schicht, über REST-Anfragen, wird durch das Framework Axios unterstützt.

3.2 Beschreibung der Struktur

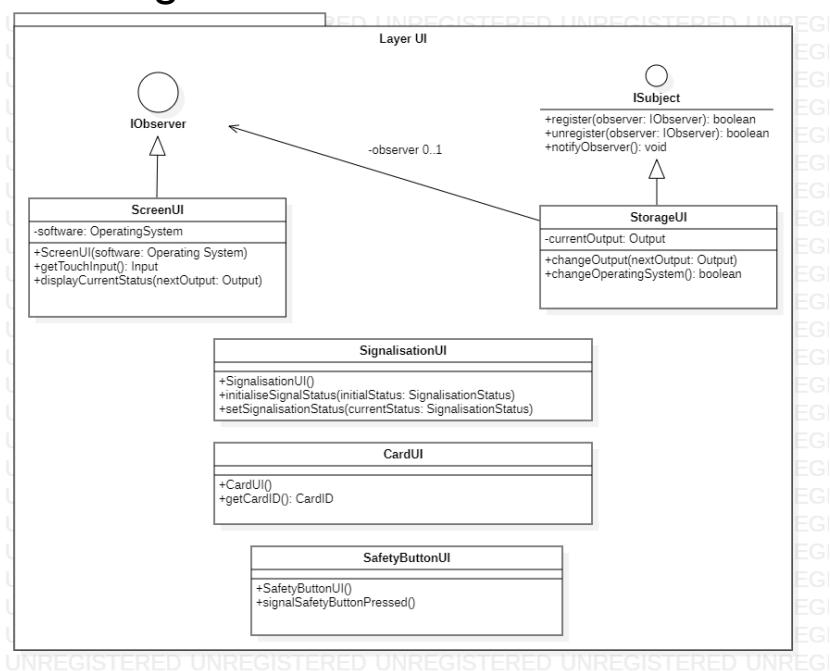


Abbildung 3-1: Struktur der Layer UI

Das User Interface besteht aus den Klassen ScreenUI, SignalisationUI, CardUI und SafetyButtonUI.

In der Klasse ScreenUI werden alle Interaktionsmöglichkeiten und Anzeigen, auf dem Eingabepanel, zusammengefasst und dargestellt. Das Attribut „software“ stellt das aktuelle Betriebssystem dar, welches über den Konstruktor initialisiert wird. Über die Methode „getTouchInput“ wird der auszuführende Befehl ausgelesen. Mit „displayCurrentStatus“ wird die von der Software Safe-RFID vorgegebene Display Anzeige ausgegeben.

In SignalisationUI sind alle verbauten LEDs und der Lautsprecher zusammengefasst. Die Klasse beinhaltet Funktionen zum Initialisieren und zum Ändern des Status, des jeweiligen Hardwareelements.

In CardUI wird das Einlesen der RFID-Karte behandelt. Die Methode „getCardID“ liefert die ID der eingelesenen Karte zurück.

SafetyButtonUI beinhaltet den Notfallknopf im Inneren des Safes. Beim Drücken dieses Knopfes wird ein Event an die darunter liegenden Business-Objects-Schicht gesendet.

Um den Programmierwurf möglichst flexibel zu gestalten und damit auch die Wartbarkeit zu erhöhen, wird ein MVC Pattern für die Anzeige am Eingabepanel verwendet. Dabei handelt es sich um ein Entwurfsmuster, das in drei Unterpunkte unterteilt werden kann. Die Steuerung, die Präsentation und das Modell. Im Modell befinden sich die Daten, die in der Präsentation dargestellt werden sollen, die Präsentation wird über Änderungen im Modell durch das Observer Pattern benachrichtigt und zeigt diese in weiterer Folge an. In der Steuerung werden die gewünschten Änderungen der Anzeige verarbeitet und dann folgerichtig im Modell abgelegt. Die Präsentation wird durch die Klasse ScreenUI, in der User-Interface-Schicht, dargestellt. Das Modell und damit die anzuzeigenden Daten, werden von der Klasse HardwareAbstractionLayer in der Business-Objects-Schicht in die Klasse StorageUI, in der User-Interface-Schicht, übertragen. Bei einer Änderung in der Klasse StorageUI wird automatisch die Klasse ScreenUI, durch das Observer Pattern, benachrichtigt. Dieser modulare Aufbau ermöglicht es Veränderung der Anzeigesoftware besser umzusetzen.

3.3 Beschreibung des Verhaltens

Da die nachfolgende Abfolge öfters verwendet wird und um die Übersichtlichkeit zu gewährleisten, wurde hierfür ein Referenzsequenzdiagramm angefertigt, auf das in den nachfolgenden Use Cases verwiesen wird.

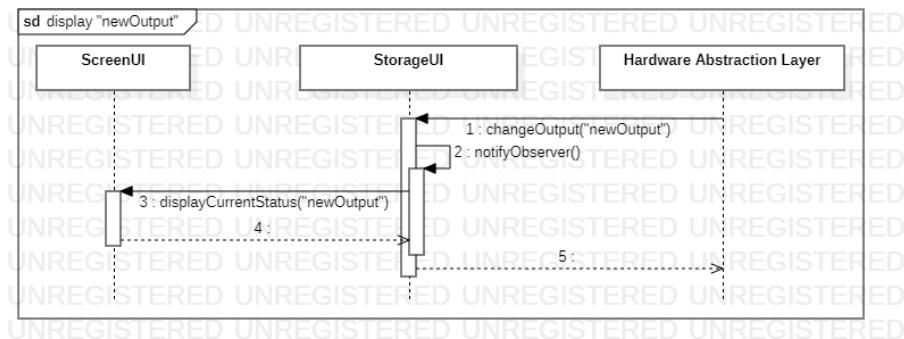


Abbildung 3-2: Sequenzdiagramm "Inbetriebnahme"

Hier ist die Interaktion der Klassen untereinander für die Anzeige eines neuen Display Bildes dargestellt. Das in Anführungszeichen gesetzte „newOutput“ stellt eine Beispiel Anzeige dar. In den nachfolgenden Sequenzdiagrammen ist diese Beispielzeichenkette durch den jeweiligen Text in Anführungszeichen zu ersetzen.

3.3.1 INBETRIEBNAHME

Diese Darstellung in Abbildung 3-3 veranschaulicht die Interaktion des Benutzers mit der Layer UI, bei dem Use-Case „Inbetriebnahme“.

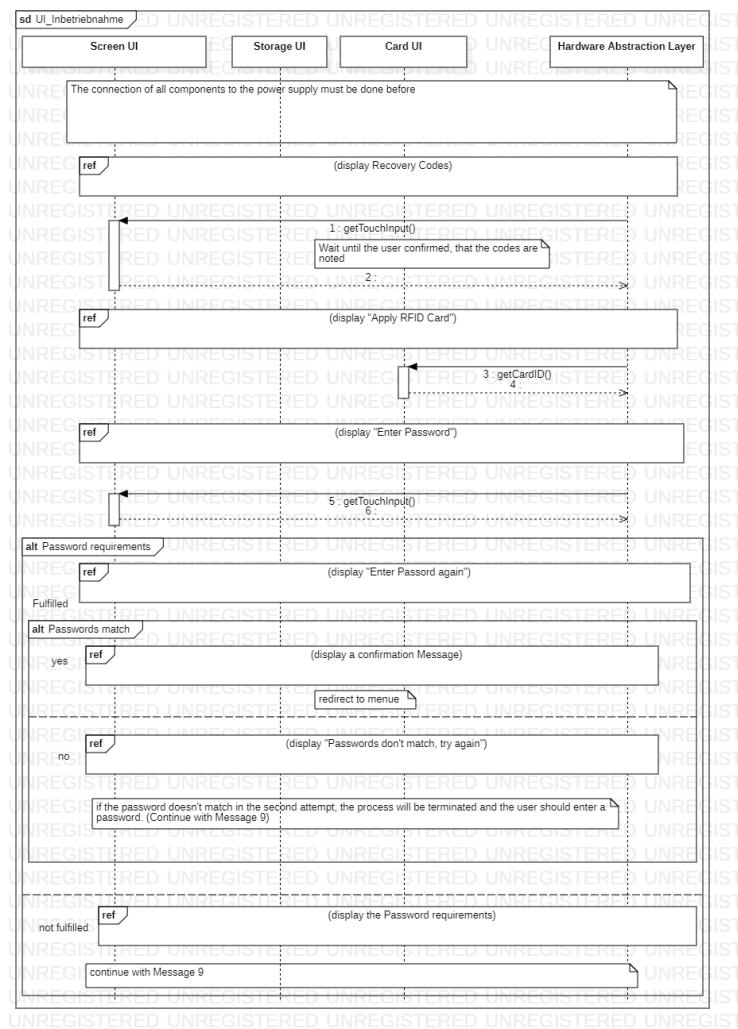


Abbildung 3-3: Sequenzdiagramm "Inbetriebnahme"

Da es sich bei der Inbetriebnahme um die Erstellung eines ersten Systemadministrators handelt, wird hier bei Falscheingabe des Passwortes anders vorgegangen, als bei „Benutzer erstellen“. Hier wird dieser Prozess solange wiederholt, bis die Passwörter einerseits den Anforderungen genügen und andererseits die Passwörter auch übereinstimmen.

3.3.2 AUTHENTISIERUNG DES BENUTZERS

Diese Darstellung veranschaulicht die Interaktion des Benutzers mit der Layer UI, bei dem Use-Case „Authentisierung des Benutzers“.

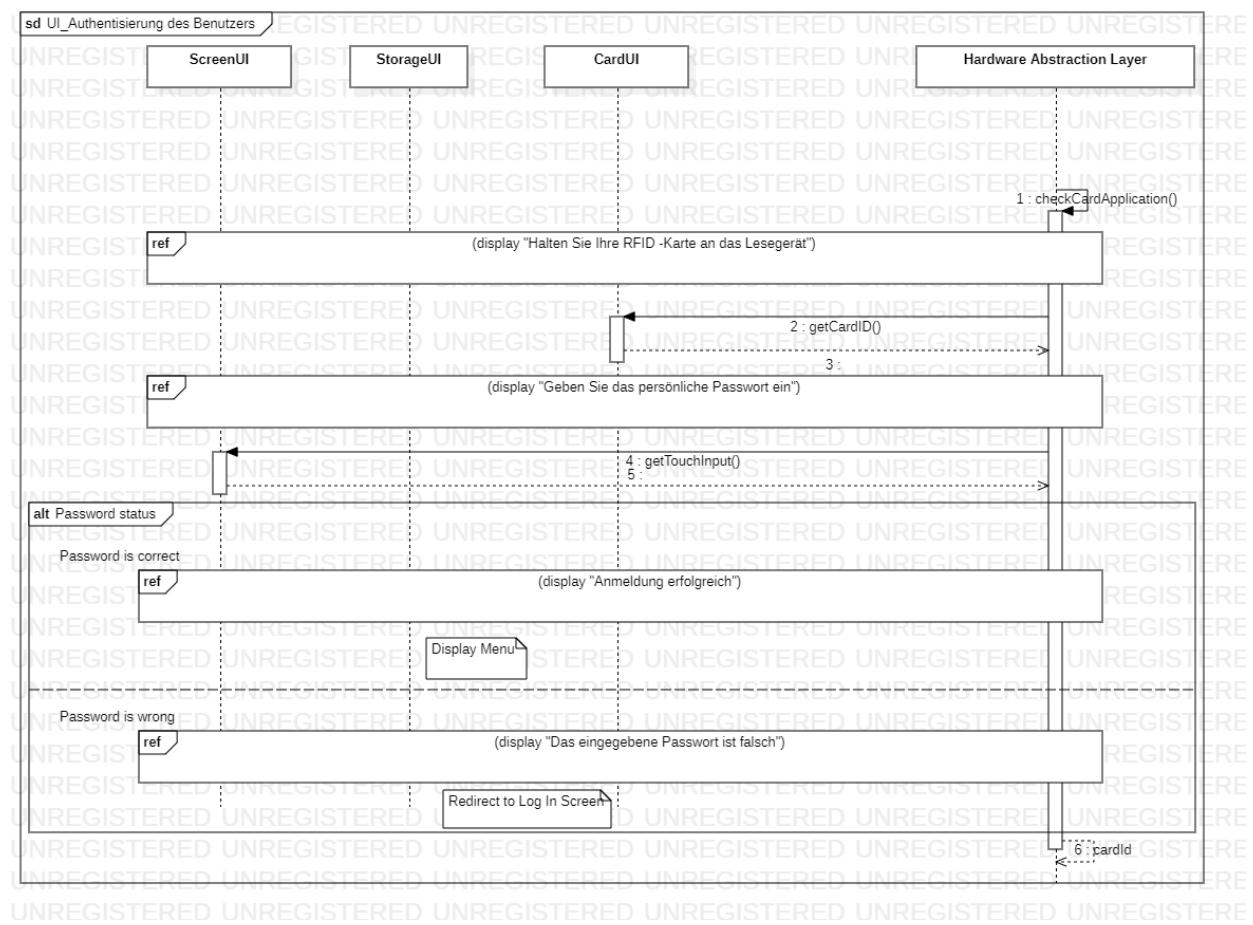


Abbildung 3-4: Sequenzdiagramm "Authentisierung des Benutzers"

Wurde die Anmeldung erfolgreich, wird eine kurze Statusmeldung angezeigt und dann das Menü. Wird jedoch das Passwort falsch eingegeben, so wird nach dem Anzeigen einer Statusmeldung sofort der Log-In Screen dargestellt und der Benutzer muss den ganzen Anmeldeprozess erneut durchlaufen.

3.3.3 AUSLÖSEN EINES ALARMS

Diese Darstellung veranschaulicht die Interaktion des Benutzers mit der Layer UI, bei dem Use-Case „Auslösen eines Alarms“.

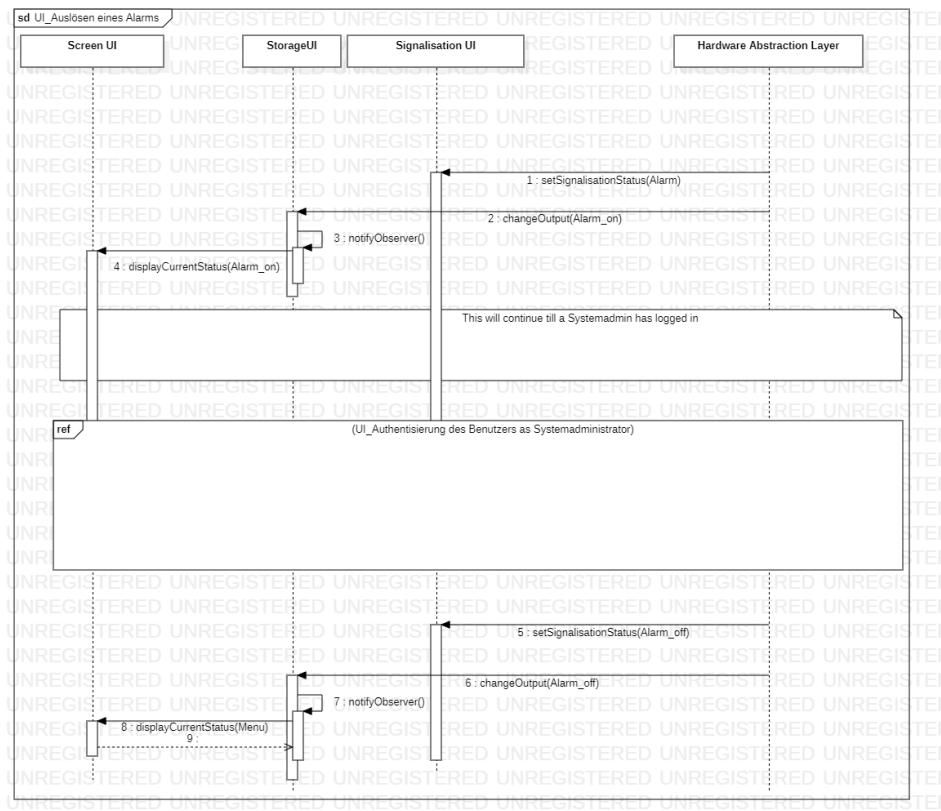


Abbildung 3-5: Sequenzdiagramm "Auslösen eines Alarms"

Hierbei ist zu erwähnen, dass der Alarmzustand solange aufrechterhalten wird, bis sich ein Systemadministrator erfolgreich authentisiert hat.

3.3.4 ENTRIEGELN

Diese Darstellung veranschaulicht die Interaktion des Benutzers mit der Layer UI, bei dem Use-Case „Entriegeln“.

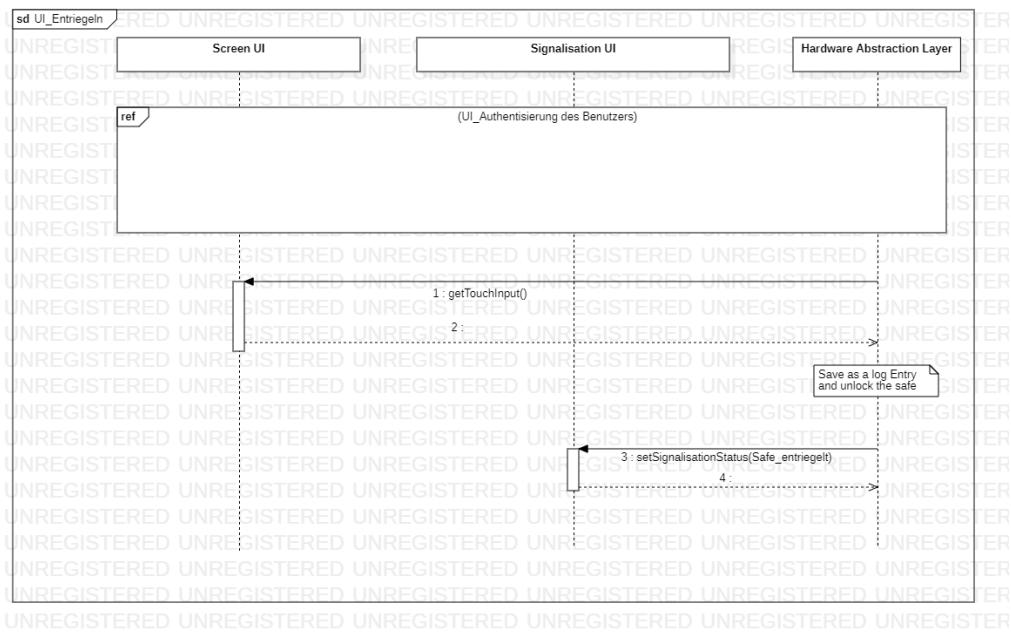


Abbildung 3-6: Sequenzdiagramm "Entriegeln"

Als bald durch einen berechtigten Benutzer die Funktion Entriegeln ausgewählt wurde, wird der Safe elektronisch entriegelt und die Signalisierungs-LEDs werden dementsprechend angesteuert.

3.3.5 BENUTZER ERSTELLEN

Diese Darstellung veranschaulicht die Interaktion des Benutzers mit der Layer UI, bei dem Use-Case „Benutzer erstellen“.

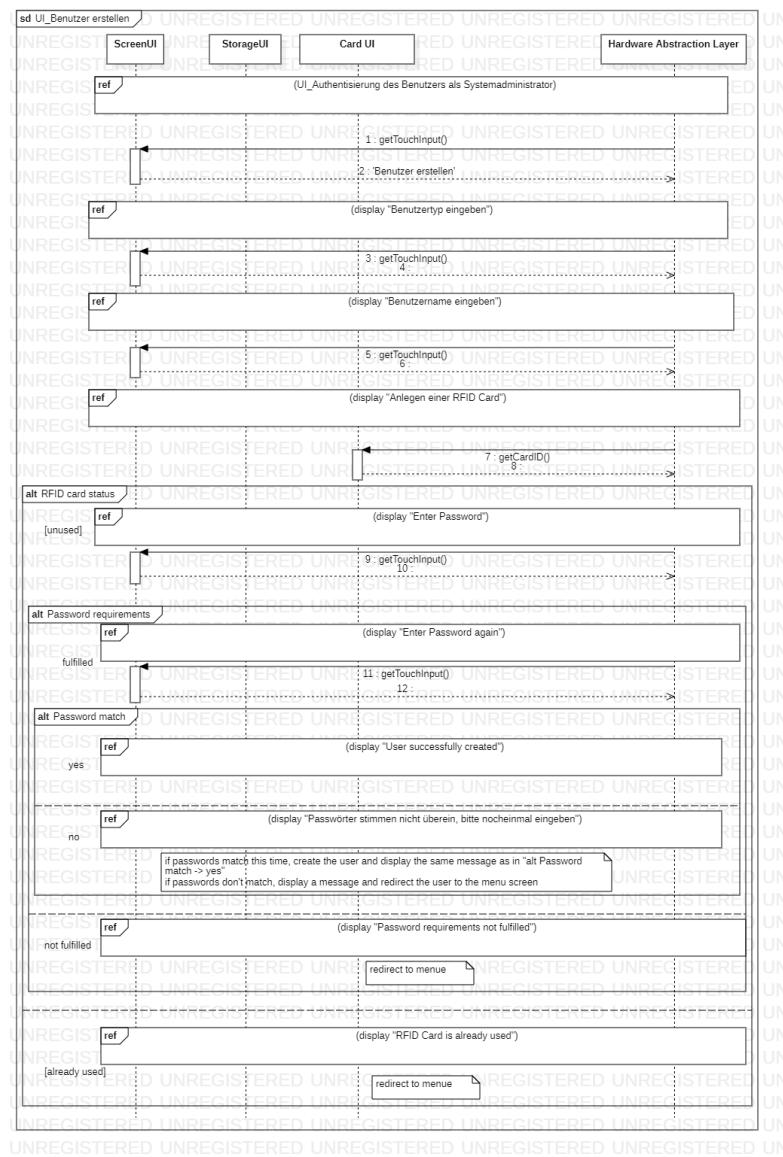


Abbildung 3-7: Sequenzdiagramm "Benutzer erstellen"

Wird beim Erstellen eines neuen Benutzers eine bereits verwendete RFID Card angelegt, so wird eine Meldung angezeigt und der Erstellvorgang abgebrochen. Werden weiters die Passwort-Anforderungen nicht erfüllt, so wird dies auch mit einer Statusmeldung angezeigt und dabei sollen auch die Passwortanforderungen enthalten sein. Der Erstellvorgang des neuen Benutzers wird hier dann auch abgebrochen. Stimmen die Passwörter bei der erneuten Eingabe nicht überein, so hat der Systemadministrator noch einmal die Möglichkeit das gewünschte Passwort einzugeben, decken sich die Passwörter erneut nicht, so wird der Erstellvorgang abgebrochen und es wird wieder zum Menü Screen zurückgekehrt. Andernfalls wird ein neuer Benutzer erstellt.

4 BESCHREIBUNG DES LAYER BO

Die Business-Objects-Schicht beinhaltet die Logik der Software und somit die eigentliche Funktionalität des Gesamtsystems. Sie bildet die Schnittstelle zwischen der User-Interface-Schicht, die für die Kommunikation mit dem Benutzer verantwortlich ist, und der persistenten Datenschicht, die für das Management der gespeicherten Daten zuständig ist.

Die Business-Objects-Schicht ist nicht auf weitere Schichten angewiesen. Mit ihr kann über die integrierte REST-Schnittstelle, ohne Zutun der User-Interface-Schicht, kommuniziert werden. Für den Fall, dass die persistente Datenschicht nicht verfügbar ist, beziehungsweise die Kommunikation zu dieser gestört ist, werden die Daten nicht persistent gespeichert, sondern sind nur im Rahmen der aktuellen Laufzeit verfügbar.

4.1 Schnittstellenbeschreibung

Sowohl über die User-Interface-Schicht als auch über die integrierte REST-Schnittstelle, erfolgt die Kommunikation des Benutzers, mit der Business-Objects-Schicht, über REST-Anfragen.

Die Business-Objects-Schicht verwendet das Java-Framework „Spring“, um eine möglichst sichere und flexible Implementierung der Logik zu ermöglichen. Unter anderem basiert die REST-Schnittstelle auf besagtem Framework.

4.2 Beschreibung der Struktur

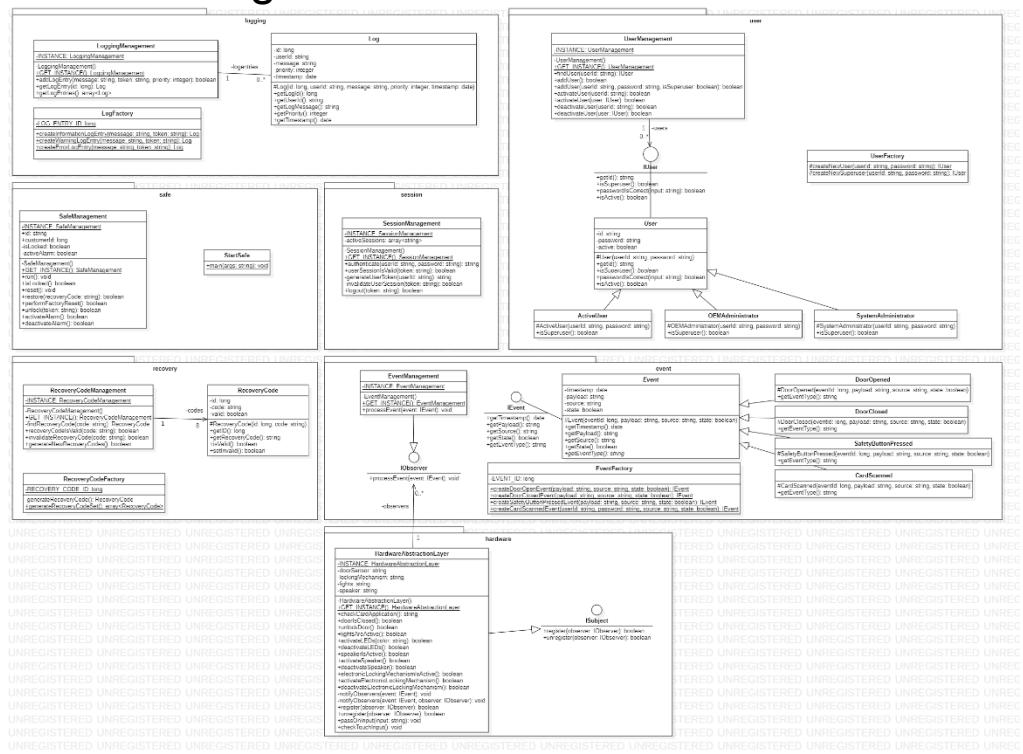


Abbildung 4-1: Struktur der Business-Objects-Schicht

Auf der Basis, des im Software Requirements Specifications Dokument definierten Domänenmodells, wurde das in Abbildung 4-1: Struktur der Business-Objects-Schicht dargestellte Klassendiagramm entworfen. Darin wurden die einzelnen Entitäten, des Domänenmodells, zu Modulen zusammengefasst. Unter anderem wurden die Benutzergruppen „OEM Administrator“, „System Administrator“ und „Common User“ in das Modul „users“ integriert. Die einzelnen Hardware-Elemente des Safes sind im Modul „hardware“ enthalten. Um die bereits eingangs beschriebenen Vorteile der zu entwickelnden Lösung, gegenüber dem bisherigen Verriegelungsmechanismus, zu verwirklichen, wurden zudem die Module „recovery“ und „logging“ entworfen, um die Wiederherstellung des operativen Betriebs und das Aufzeichnen der Zugriffe zu ermöglichen. Weiters ist das Modul „events“ enthalten, welches die eventbasierte Kommunikation, zwischen dem Gesamtsystem und der Hardware Abstraction Layer, ermöglicht. Die einzelnen Module werden durch eigene Management-Klassen verwaltet.

Bei der Umsetzung werden Elemente wiederverwendbarer objektorientierter Software, sogenannte Entwurfsmuster, angewandt:

- Um sicherzustellen, dass zu jedem Zeitpunkt nur eine Instanz der Klassen „SafeManagement“, „UserManagement“, „RecoveryCodeManagement“, „LoggingManagement“, „SessionManagement“, „EventManagement“ und „HardwareAbstractionLayer“ existiert, wird in diesen Fällen das Singleton Pattern angewandt.
- Das Erstellen von Log-Einträgen, Wiederherstellungs-Codes und Events, geschieht über entsprechende Softwareelemente, die das Factory Pattern implementieren.

- Die Umsetzung der eventbasierten Kommunikation des Gesamtsystems mit der Hardware Abstraction Layer, basiert auf dem Observer Pattern. Da für die Anwendung, zur Kommunikation über Events, das Observer Pattern nicht optimal ist, erfolgt die Umsetzung als Mischform zwischen dem Observer Pattern und dem Listener Pattern.

4.3 Beschreibung des Verhaltens

Im folgenden Teil wird das dynamische Verhalten der Business-Objects-Schicht, mithilfe von Sequenzdiagrammen, beschrieben.

Alle im Weiteren beschriebenen Fehlerfälle werden durch einen entsprechenden Log-Eintrag festgehalten und von dem System gespeichert.

4.3.1 INBETRIEBNAHME

Die Inbetriebnahme beschreibt den erstmaligen Start der Software, bei dem Endkunden. Bei der Inbetriebnahme werden die Sicherheitsrichtlinien für die Operation des Safes definiert.

Normaler Ablauf:

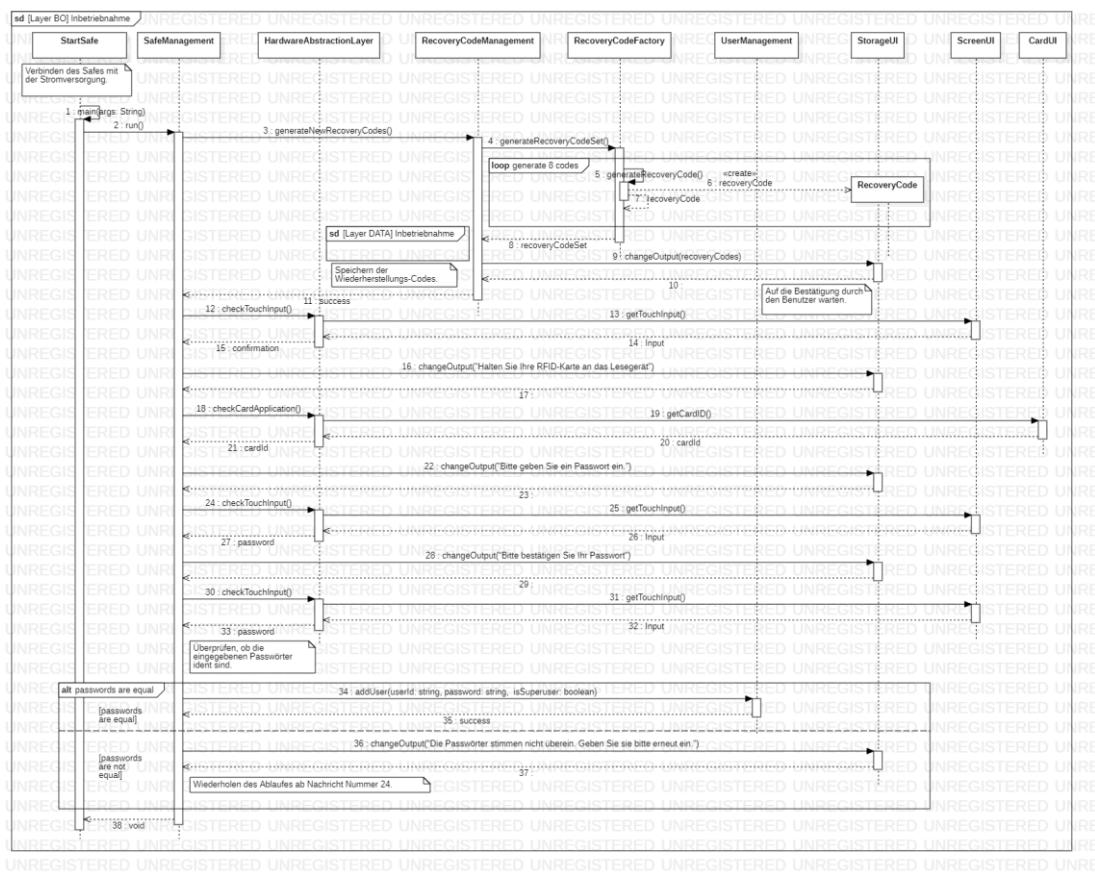


Abbildung 4-2: Business-Objects-Schicht Sequenzdiagramm "Inbetriebnahme"

Alternative Abläufe: Im Rahmen der Erstellung des Systemadministrator-Accounts, wird der Benutzer wird zur Eingabe eines persönlichen Passwortes aufgefordert. Entspricht dieses nicht den Vorgaben, muss ein anderes Passwort gewählt werden. (siehe Abbildung 4-2, Nachricht Nummer 8)

Funktionale Anforderungen: Der Safe muss im Rahmen der Inbetriebnahme acht (8) unterschiedliche, zufällig generierte Codes anzeigen. Weiters muss der Benutzer aktiv aufgefordert werden einen Systemadministrator-Account zu erstellen. Der Safe muss nach der Inbetriebnahme in den operativen Betrieb übergehen.

Fehlerfälle: Kommt es während der Inbetriebnahme zu einer Unterbrechung der Stromversorgung, wird der Safe auf den Werkszustand zurückgesetzt.

Tritt während dem Speichern der Wiederherstellungs-Codes oder des anzulegenden Systemadministrator-Benutzers ein Fehler auf, werden diese dem Benutzer durch ein entsprechendes Dialogfeld mitgeteilt.

4.3.2 AUTHENTISIERUNG DES BENUTZERS

Die Authentisierung beschreibt das Feststellen der Identität des aktuellen Benutzers. Nach der erfolgreichen Authentisierung erhält der Benutzer einen Benutzer-Token und wird ermächtigt den Safe, gemäß den zugewiesenen Berechtigungen, zu bedienen.

Normaler Ablauf:

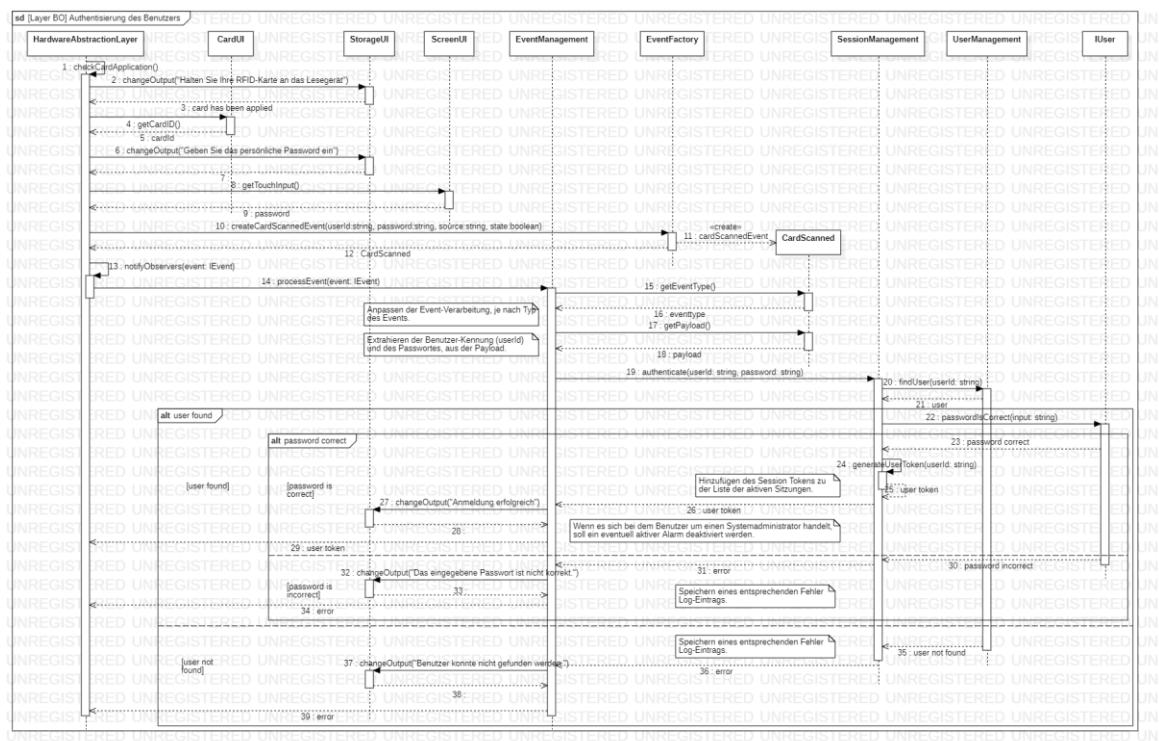


Abbildung 4-3: Business-Objects-Schicht Sequenzdiagramm "Authentisierung des Benutzers"

Alternative Abläufe: Sollte die Benutzer-Kennung, der gescannten RFID-Karte, nicht in der Liste der aktiven Benutzer enthalten sein, wird der Anmeldeprozess abgebrochen und eine entsprechende Warnung in die Liste der gespeicherten Log-Einträge aufgenommen. (siehe Abbildung 4-3, Nachricht Nummer 4)

Sollte der Benutzer den persönlichen Zugangscode wiederholt (dreimal) falsch eingeben, wird ein Alarm ausgelöst. (siehe Abbildung 4-3, Nachricht Nummer 12; siehe Use-Case „Auslösen eines Alarms“)

Funktionale Anforderungen: Nach dem erfolgreichen Auslesen einer vor dem RFID-Lesegerät platzierten RFID-Karte, muss die ausgelesene Benutzerkennung mit denen der aktiven Benutzer verglichen werden. Existiert für die eingescannte Benutzer-Kennung ein gespeicherter Eintrag, wird der Benutzer zur Eingabe des persönlichen Passworts aufgefordert. Stimmt dieses, mit dem für den Benutzer gespeicherten Passwort überein, wird der Benutzer angemeldet. Ein

entsprechender Log-Eintrag wird gespeichert. Für den Fall, dass es sich bei der Kennung um keine bekannte Benutzer-Kennung handelt, muss der Anmeldeprozess abgebrochen werden. Im Falle der mehrmaligen (dreimal) falschen Eingabe des persönlichen Passwortes, muss ein Alarm ausgelöst werden. In beiden Fehlerfällen soll eine entsprechende Warnung in die Liste der gespeicherten Log-Einträge aufgenommen werden.

Fehlerfälle: Kommt es während dem Auslesen der, vor dem RFID-Lesegerät platzierten, RFID-Karte zu einem Fehler, wird dieser dem Benutzer durch ein entsprechendes Dialogfeld mitgeteilt. Treten bei dem Auslesen der gespeicherten Benutzer, beziehungsweise bei der Kommunikation mit der Datenbank, Fehler auf, werden diese dem Benutzer auf dem Eingabepanel beziehungsweise in der Antwort der REST-Abfrage angezeigt. In beiden Fehlerfällen wird der Anmeldevorgang abgebrochen.

4.3.3 AUSLÖSEN EINES ALARMS

Das Auslösen eines Alarms bedeutet die Abgabe visueller und auditiver Signale, des Safes, um auf einen möglicherweise sicherheitsbeeinträchtigenden Zustand hinzuweisen.

Normaler Ablauf:

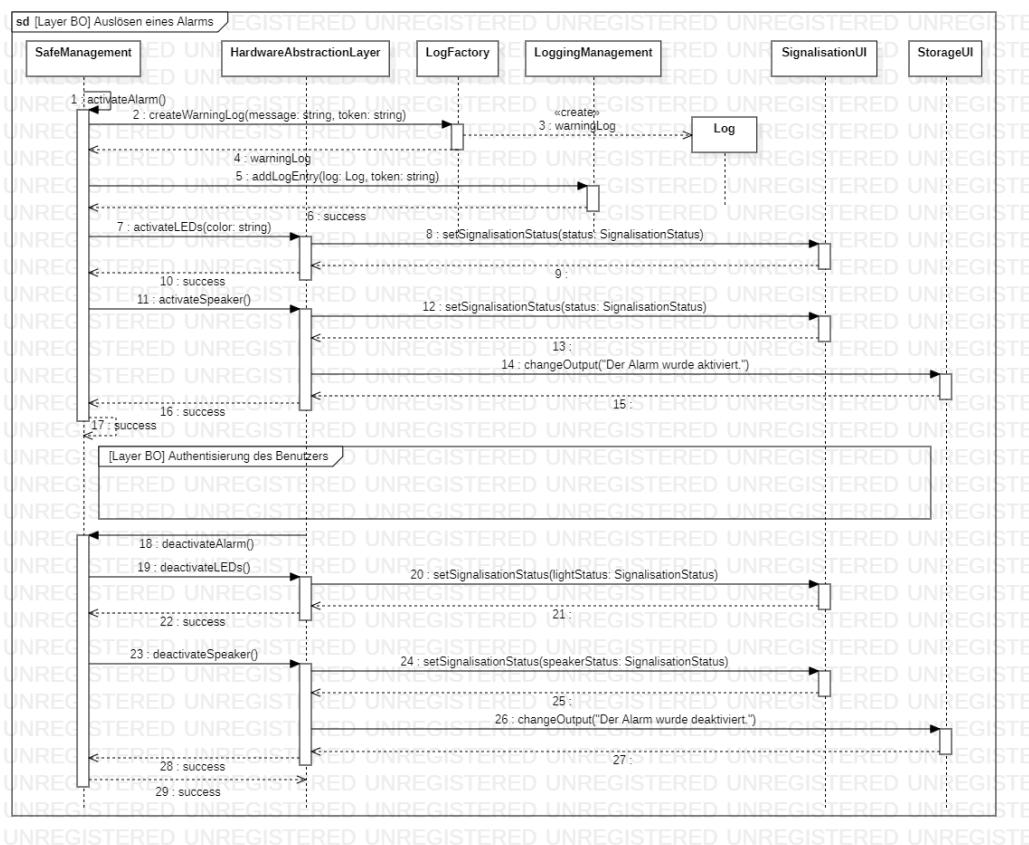


Abbildung 4-4: Business-Objects-Schicht Sequenzdiagramm "Auslösen eines Alarms"

Funktionale Anforderungen: Sicherheitsbeeinträchtigende Zustände sollen einen Alarm auslösen. Im Falle eines Alarms blinken die fünf (5) LEDs, am oberen Rand des Safes, rot und ein Signalton ertönt. Die Deaktivierung des Alarms soll durch die Authentisierung eines Systemadministrators erfolgen.

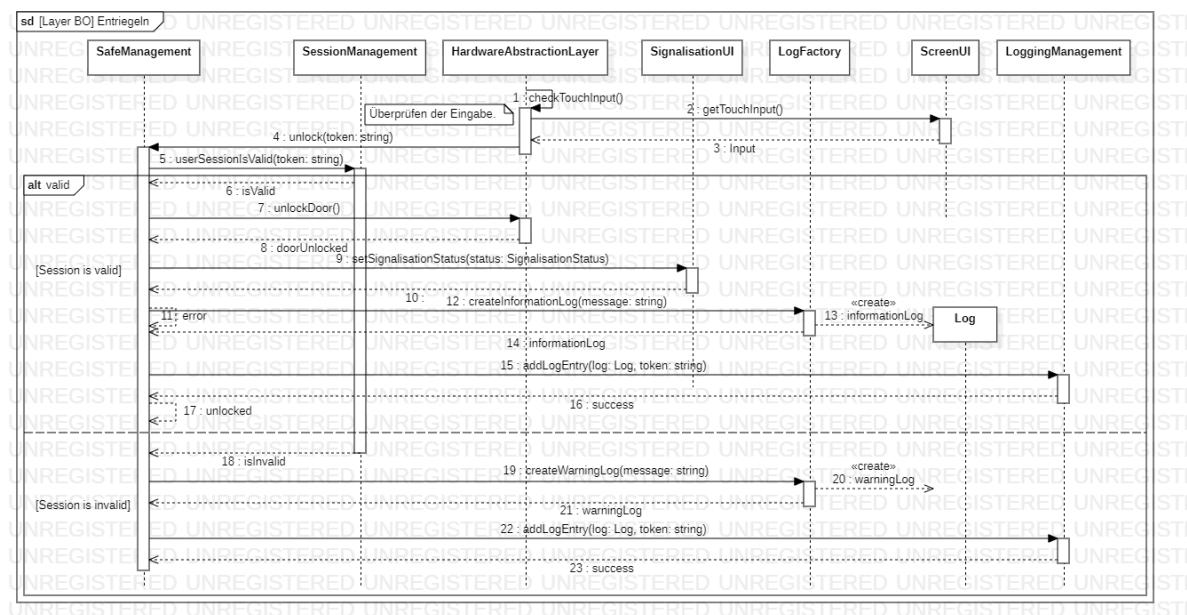
Fehlerfälle: Tritt bei dem Speichern des Log-Eintrags ein Fehler auf, soll dieser über ein entsprechendes Dialogfeld dem Benutzer mitgeteilt werden.

Tritt bei der Kommunikation mit der Hardware Abstraction Layer ein Fehler auf, beziehungsweise ist diese nicht in der Lage die entsprechenden Hardware-Komponenten anzusteuern, soll dies ebenfalls über eine entsprechende Mitteilung dem Benutzer angezeigt werden und als Log-Eintrag gespeichert werden.

4.3.4 ENTRIEGELN

Aktive Benutzer sollen nach der erfolgreichen Authentisierung die Möglichkeit haben den Safe zu entriegeln.

Normaler Ablauf:



Alternative Abläufe: Wird die Tür des Safes innerhalb von 30 Sekunden nach der Entriegelung, nicht geöffnet, wird die Tür automatisch verriegelt und der Benutzer abgemeldet.

Funktionale Anforderungen: Aktive, autorisierte, Benutzer sollen die Möglichkeit haben die Entriegelung des Safes zu beantragen. Findet diese statt, hat der Benutzer 30 Sekunden lange die Möglichkeit die Safe-Türe zu öffnen. Passiert dies nicht, wird der Safe verriegelt und der Benutzer automatisch abgemeldet. Zutritte sollen in einer integren Liste gespeichert werden. Der aktuelle Verriegelungsstatus muss dem Benutzer jederzeit entsprechend signalisiert werden. Wird der Notfall-Taster im Inneren eines begehbarren Safes betätigt, soll der Safe sofort entriegelt werden.

Fehlerfälle: Tritt bei der Kommunikation des Gesamtsystems mit der Hardware Abstraction Layer ein Fehler auf, beziehungsweise ist diese nicht in der Lage die entsprechenden Hardware-Komponenten anzusteuern, soll dies dem Benutzer in Form einer entsprechenden Mitteilung angezeigt werden und als Log-Eintrag gespeichert werden.

Tritt bei dem Speichern des Log-Eintrags ein Fehler auf, soll dieser über ein entsprechendes Dialogfeld dem Benutzer mitgeteilt werden.

4.3.5 BENUTZER ERSTELLEN

Systemadministratoren soll es möglich sein, neue Benutzer anzulegen, die wiederum mit der ihnen zugewiesenen RFID-Karte und einem persönlichen Zugangscode Zutritt zu dem Safe erhalten.

Normaler Ablauf:

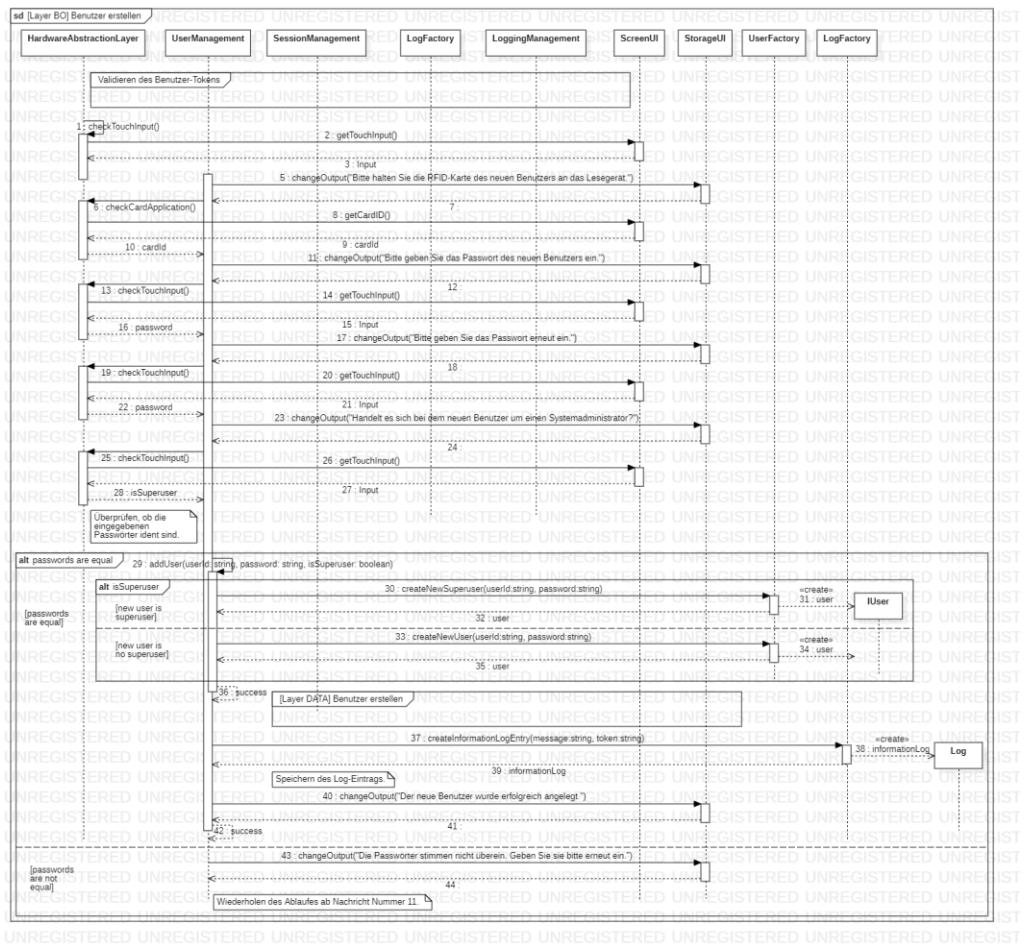


Abbildung 4-5: Business-Objects-Schicht Sequenzdiagramm "Benutzer erstellen"

Alternative Abläufe: Wird eine bereits registrierte RFID-Karte an das RFID-Lesegerät gehalten, muss eine Fehlermeldung erscheinen. (siehe Abbildung 4-5, Nachricht Nummer 9) Erfüllt das vom Systemadministrator eingegebene Passwort die formalen Bedingungen nicht, muss ein anderes Passwort gewählt werden. (siehe Abbildung 4-5, Nachricht Nummer 16)

Funktionale Anforderungen: Systemadministratoren müssen die Möglichkeit haben neue Benutzer anzulegen. Bereits zugewiesene RFID-Karten dürfen nicht erneut zugewiesen werden. Der neue Benutzer muss im System abgespeichert werden und je nach Berechtigungsstatus muss es ihm der Zugang zu dem Safe gestattet werden.

Fehlerfälle: Kommt es im Rahmen des Use-Cases zu Kommunikationsproblemen des Gesamtsystems mit der Datenbank oder ist das Speichern des erstellten Benutzers nicht möglich, soll dies dem aktuellen Benutzer in Form einer entsprechenden Fehlermeldung signalisiert werden. Zusätzlich soll ein Log-Eintrag bezüglich des Fehlers gespeichert werden.

Der erstellte Benutzer kann in diesem Fall nicht angelegt werden.
Tritt bei dem Speichern des Log-Eintrags ein Fehler auf, soll dieser über ein entsprechendes Dialogfeld dem Benutzer mitgeteilt werden.

5 BESCHREIBUNG DES LAYER DATA

Die persistente Datenschicht dient der Archivierung systemrelevanter Daten. Sie bildet die Schnittstelle zwischen der, in der Business-Objects-Schicht enthaltenen, Logik des Gesamtsystems und der Datenbank. Daten, welche in der Business-Objects-Schicht verarbeitet werden, werden über entsprechende Persistenzmanager-Klassen an die Datenbank weitergeleitet, in welche sie in weiterer Folge eingepflegt werden.

5.1 Schnittstellenbeschreibung

Die Kommunikation mit der persistenten Datenschicht erfolgt über Persistenzmanager-Klassen. Für jede, in der Business-Objects-Schicht befindliche Klasse, welche Daten enthält, existiert eine Persistenzmanager-Klasse in der persistenten Datenschicht, über welche die entsprechenden Daten an die Datenbank weitergeleitet werden.

Die Kommunikation zwischen den Persistenzmanager-Klassen und der Datenbank erfolgt über eine Schnittstellen-Klasse.

Die Verbindung zu dem Datenbankmanagementsystem PostgreSQL erfolgt über einen entsprechend optimierten JDBC Treiber.

5.2 Beschreibung der Struktur

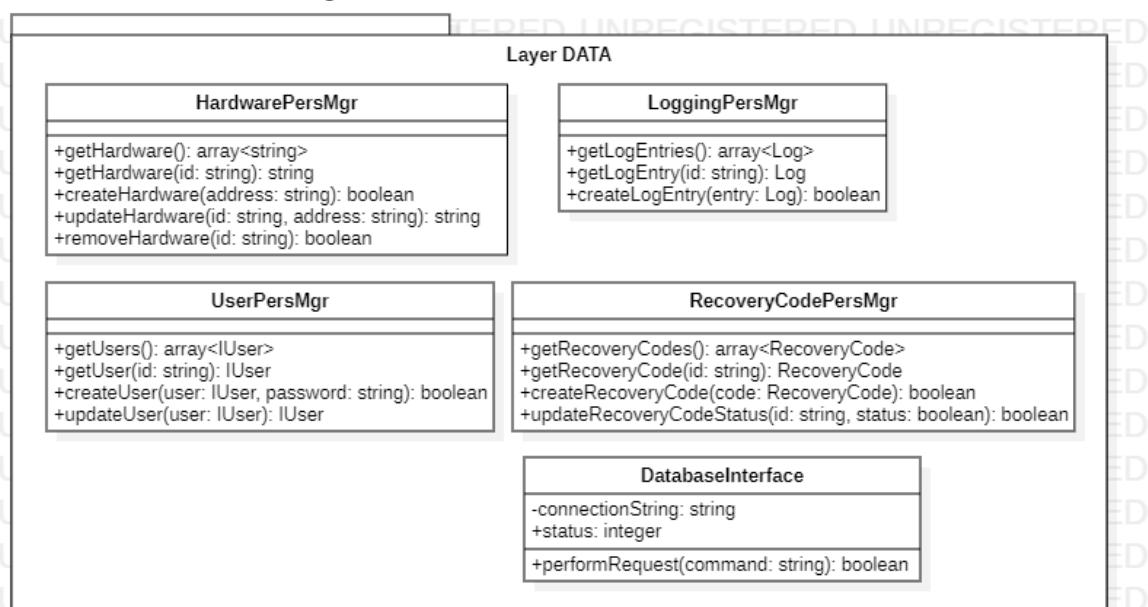


Abbildung 5-1: Struktur der persistenten Datenschicht

Aus den in Abbildung 4-1: Struktur der Business-Objects-Schicht dargestellten Modulen, ergeben sich die in Abbildung 5-1 dargestellten Persistenzmanager-Klassen. Die einzelnen Persistenzmanager-Klassen, sind für die jeweils entsprechende Klasse in der Business-Objects-Schicht optimiert. So ist es zum Beispiel nicht möglich über die „LoggingPersMgr“ Klasse Log-Einträge im Nachhinein zu verändern oder zu löschen. Ebenfalls ist es nicht möglich über die Klasse „UserPersMgr“ Benutzer zu löschen.

Die Kommunikation zwischen den einzelnen Persistenzmanager-Klassen und der Datenbank, erfolgt über die „DatabaseInterface“ Klasse.

5.3 Beschreibung des Verhaltens

Im Folgenden wird auf das Verhalten der persistenten Datenschicht eingegangen. Das dynamische Verhalten des Systems, im Rahmen des jeweiligen Use-Cases, wird anhand von UML OOD-Sequenzdiagrammen dargestellt.

Die in der Business-Objects-Schicht implementierte Logik ist für die Handhabung alternativer Abläufe verantwortlich. Die Beschreibungen der alternativen Abläufe, der einzelnen Use-Cases, können demnach den entsprechenden Use-Cases aus Punkt 4.3 entnommen werden.

Kommt es, während dem Speichern von Daten oder bei der Kommunikation mit der Datenbank (in der Methode „performRequest“) zu Fehlern, werden diese in Form von Ausnahmen (engl. „Exceptions“) an die überliegende Schicht (in diesem Fall die Business-Objects-Schicht) retourniert. Die Handhabung der Fehlerfälle erfolgt durch die in der Business-Objects-Schicht implementierte Logik und resultiert, im Falle von Fehlern in der persistenten Datenschicht, darin, dass diese dem Benutzer durch ein entsprechendes Dialogfeld angezeigt werden.

5.3.1 INBETRIEBNAHME

Im Rahmen der Inbetriebnahme werden acht (8) unterschiedliche Wiederherstellungs-Codes, nach dem Zufallsprinzip, generiert. Diese werden vom System gespeichert. Zusätzlich wird der Benutzer aufgefordert einen neuen Systemadministrator-Benutzer anzulegen. Die Informationen bezüglich dieses Accounts sollen ebenfalls vom System gespeichert werden.

Normaler Ablauf:

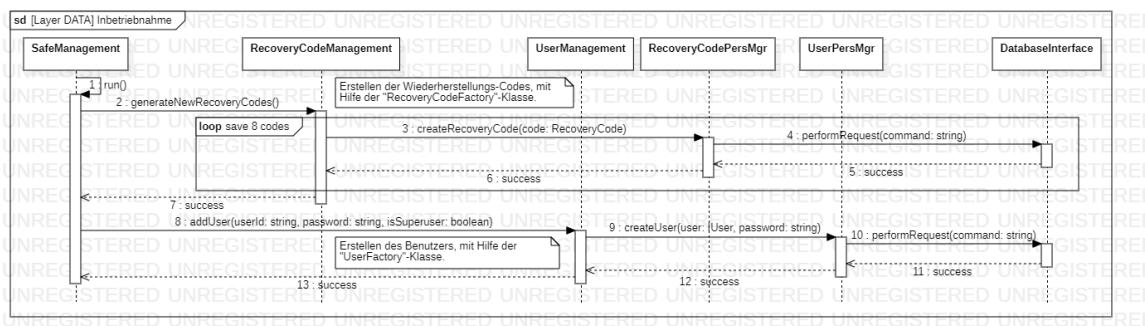


Abbildung 5-2: Persistente-Datenschicht Sequenzdiagramm "Inbetriebnahme"

5.3.2 AUTHENTISIERUNG DES BENUTZERS

Im Rahmen der Authentisierung wird die Identität des aktuellen Benutzers festgestellt. Dies geschieht anhand einer 64-stelligen Benutzer-Kennung und einem persönlichen Passwort. Da es sich hierbei um einen sicherheitsrelevanten Vorgang handelt, wird dieser in Form eines entsprechenden Log-Eintrags festgehalten. Der erstellte Log-Eintrag soll in weiterer Folge von dem System gespeichert werden.

Normaler Ablauf:

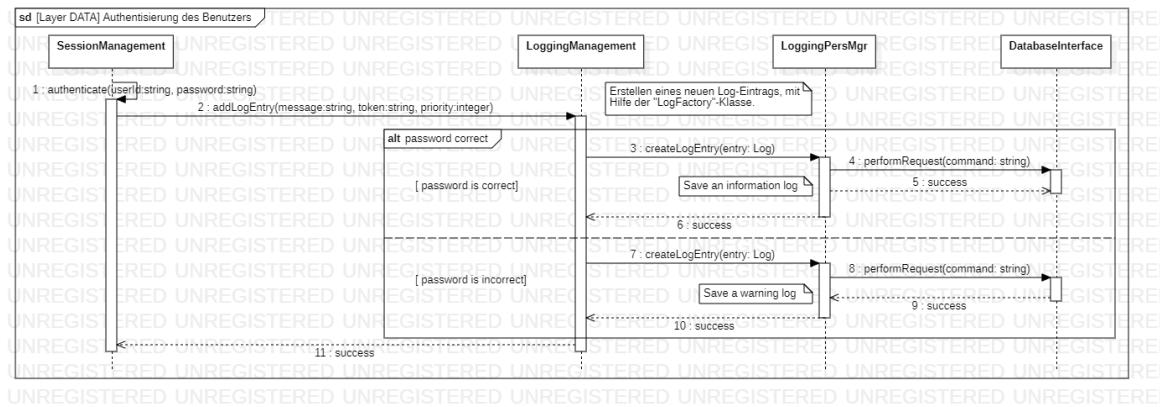


Abbildung 5-3: Persistente-Datenschicht Sequenzdiagramm "Authentisierung des Benutzers"

5.3.3 AUSLÖSEN EINES ALARMS

Das Auslösen eines Alarms beschreibt die Aktivierung der im Safe verbauten LEDs und Lautsprecher, um visuell und auditiv auf einen sicherheitsbeeinträchtigenden Zustand des Safes hinzuweisen. Die Aktivierung des Alarms wird durch entsprechende Log-Einträge festgehalten, welche im Rahmen des Use-Cases vom System gespeichert werden sollen.

Normaler Ablauf:

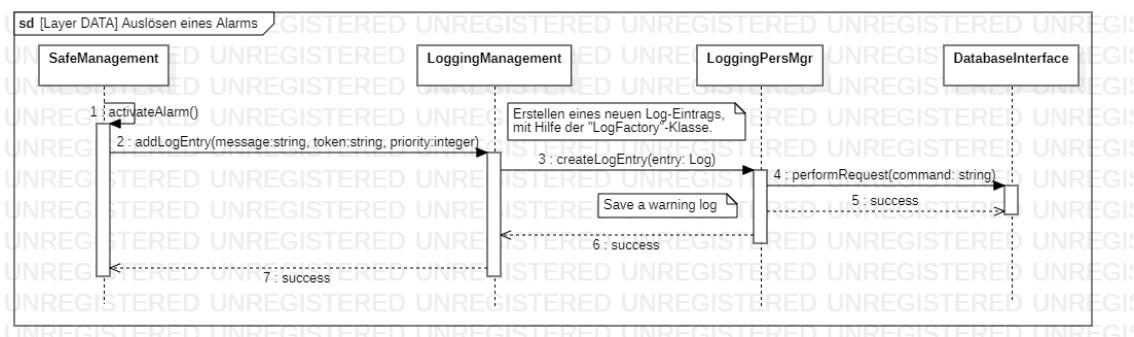


Abbildung 5-4: Persistente-Datenschicht Sequenzdiagramm "Auslösen eines Alarms"

5.3.4 ENTRIEGELN

Bei dem Entriegeln wird die Türe des Safes, durch den elektronischen Verriegelungsmechanismus, entsperrt. Da es sich hierbei um einen sicherheitsrelevanten Vorgang handelt, soll dieser in Form eines entsprechenden Log-Eintrags festgehalten werden. Der resultierende Log-Eintrag soll in weiterer Folge von dem System gespeichert werden.

Normaler Ablauf:

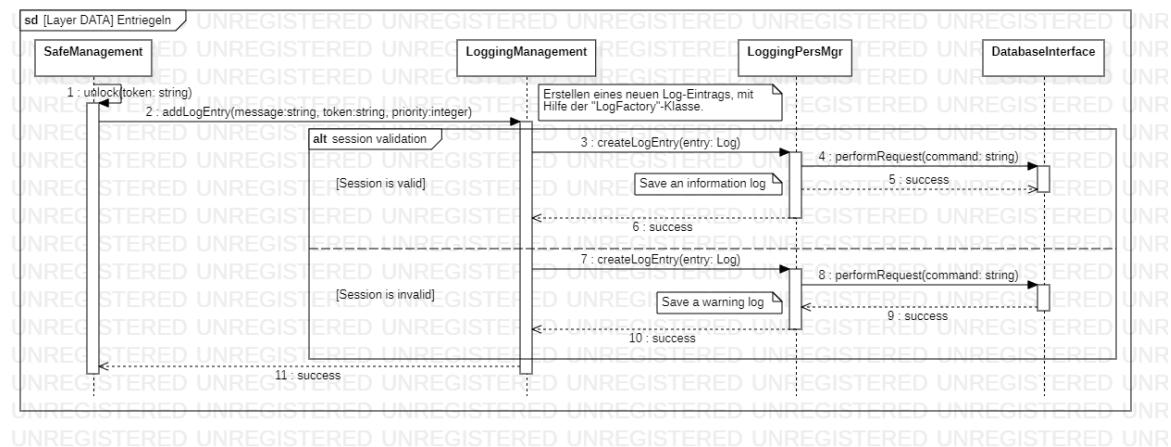


Abbildung 5-5: Persistente-Datenschicht Sequenzdiagramm "Entriegeln"

5.3.5 BENUTZER ERSTELLEN

Das Erstellen eines Benutzers beschreibt die benutzerbezogene Registrierung einer neuen RFID-Karte und das Vergeben eines persönlichen Zugangscodes, für diesen Benutzer. Im Rahmen des Use-Cases sollen die eindeutige Benutzer-Kennung, das persönliche Passwort und eine Information, ob der Benutzer aktiv ist, oder nicht, gespeichert werden.

Normaler Ablauf:

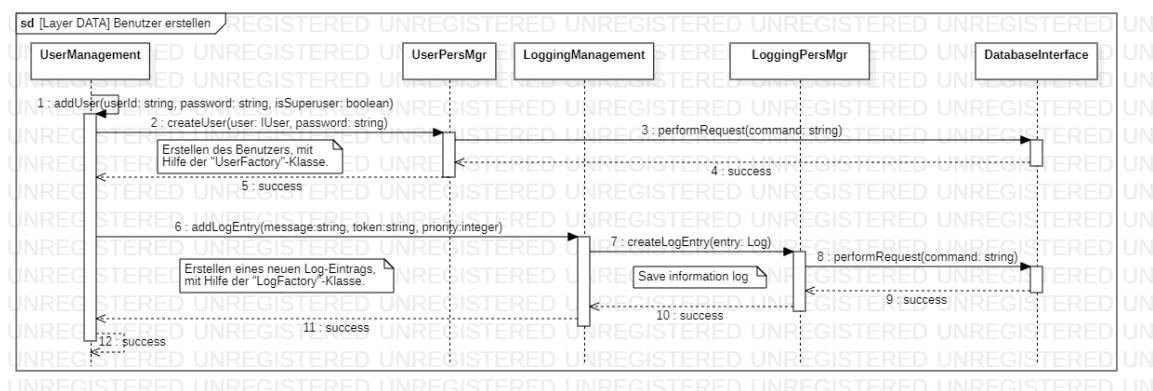


Abbildung 5-6: Persistente-Datenschicht Sequenzdiagramm "Benutzer erstellen"

6 VERWEISE AUF ANDERE DOKUMENTE

Die Definitionen, der Anforderungen des Auftraggebers, können in dem Dokument Angabe_TRAD.pdf gefunden werden.

Informationen, bezüglich des Auftrags, betreffend der allgemeinen Beschreibung, den definierten Anforderungen und den Rahmenbedingungen, können dem Dokument TRAD-Gruppe-2_Safe-RFID_SRS.pdf entnommen werden.

7 ANNEX

Die originalen UML-Diagramme sind in der Datei UML_Diagramm.mdlj enthalten.

Weiters sind folgende Exporte der UML-Diagramme angehängt:

- Architektur_Gesamtsystem.png
- Struktur_LayerUI.png
- Inbetriebnahme_LayerUI_Sequenzdiagramm.png
- Authentisierung_des_Benutzers_LayerUI_Sequenzdiagramm.png
- Ausloesen_eines_Alarms_LayerUI_Sequenzdiagramm.png
- Entriegeln_LayerUI_Sequenzdiagramm.png
- Benutzer_erstellen_LayerUI_Sequenzdiagramm.png
- DisplayNewOutput_LayerUI_Sequenzdiagramm.png
- Struktur_LayerBO.png
- Inbetriebnahme_LayerBO_Sequenzdiagramm.png
- Authentisierung_des_Benutzers_LayerBO_Sequenzdiagramm.png
- Ausloesen_eines_Alarms_LayerBO_Sequenzdiagramm.png
- Entriegeln_LayerBO_Sequenzdiagramm.png
- Benutzer_erstellen_LayerBO_Sequenzdiagramm.png
- Struktur_LayerDATA.png
- Inbetriebnahme_LayerDATA_Sequenzdiagramm.png
- Authentisierung_des_Benutzers_LayerDATA_Sequenzdiagramm.png
- Ausloesen_eines_Alarms_LayerDATA_Sequenzdiagramm.png
- Entriegeln_LayerDATA_Sequenzdiagramm.png
- Benutzer_erstellen_LayerDATA_Sequenzdiagramm.png

8 DEFINITION VON BEGRIFFLICHKEITEN

Kunde	Der Begriff „Kunde“ beschreibt den Auftraggeber.
System	„System“ beschreibt die Kombination aus der verwendeten Hardware (= der Safe) und der zu entwickelnden Lösung (= Safe RFID).
Lösung	Der Begriff „Lösung“ beschreibt die, im Rahmen des Auftrags, zu entwickelnde Software, um einen Safe steuern zu können.