

# **Basic Antibiotics Stewardship Interface**

-

## **Systementwurf**

---

Gruppe 3

## Einleitung

In diesem Dokument soll ein Überblick über das zu entwickelnde System "Basic Antibiotic Stewardship Interface" gegeben werden. Verteilungs-, Paket- und Klassendiagramm geben dabei einen Überblick über die voraussichtliche statische Struktur des Systems. Weiter zeigen einige Sequenzdiagramme einen Einblick in das System zur Laufzeit, dabei orientieren sich die Diagramme an den Anwendungsfällen aus dem Pflichtenheft.



Zur Entwicklung werden folgende Programme eingesetzt:

Visual Paradigm 10.2 zur Erstellung der UML-Diagramme für den Systementwurf  
<http://www.visual-paradigm.com/>

Eclipse "Kepler" als Entwicklungsumgebung  
<http://www.eclipse.org/>

Subclipse-Plugin für Eclipse zur Versionsverwaltung (SVN)  
<http://subclipse.tigris.org/>

Groovy/Grails Tool Suite 3.3.0 für Eclipse zur Entwicklung mit GRAILS in Eclipse  
<http://grails.org/products/ggts>

Android Development Tools 22.0.5 für Eclipse zur Entwicklung der Android-App in Eclipse  
<http://developer.android.com/tools/sdk/eclipse-adt.html>

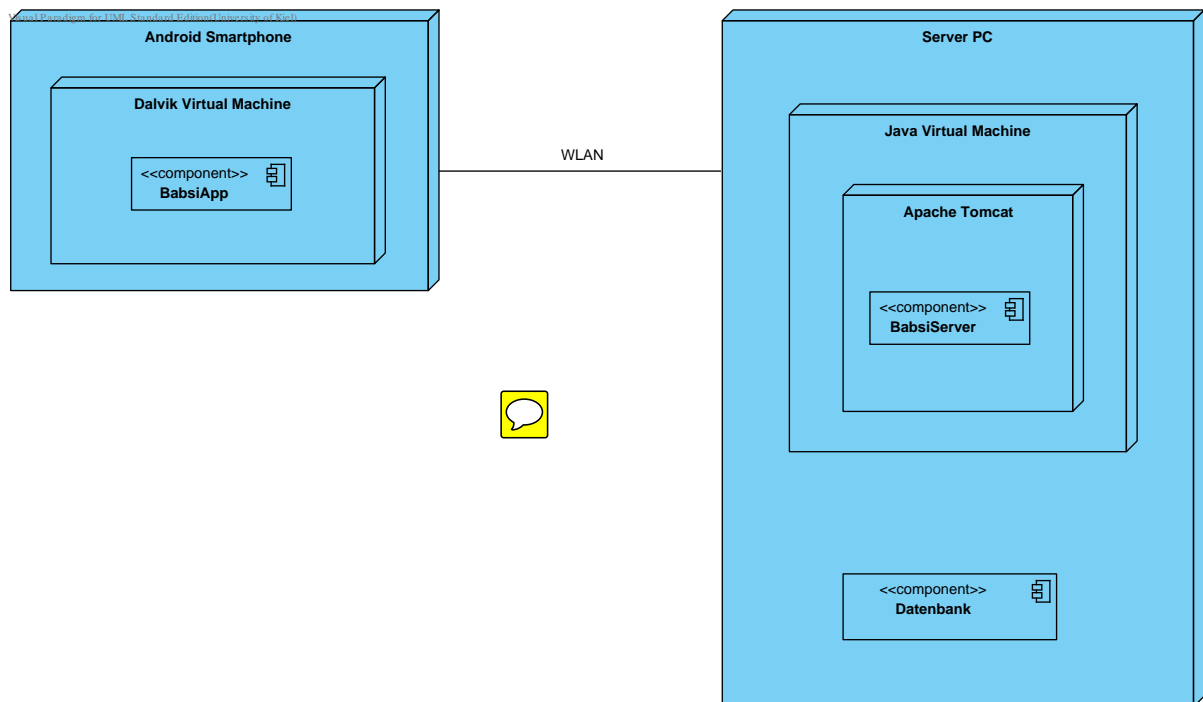
Java Development Kit 7u25 als Grundlage der Java-Entwicklung  
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>



# Inhaltsverzeichnis

Verteilungsdiagramm.....	3
Komponentendiagramm.....	4
App-Paketdiagramm.....	5
Web-Paketdiagramm.....	6
App-Klassendiagramm.....	7
Web-Klassendiagramm.....	12
A2 Visite fortsetzen.....	17
A3 Visite-pausieren.....	18
A4 Checkliste ausfüllen.....	19
A5 Visite starten.....	20
A6 Visite beenden.....	21
A7 Patient hinzufügen.....	22
A8 Logout.....	23
A9 Login.....	24
W1 Login.....	25
W3 Logout.....	26
W2 Benutzer anlegen.....	27
W4 Benutzer löschen.....	28
W6 Station anlegen.....	29
W7 Station löschen.....	30
W8 Benutzerdaten ändern.....	31
W10 Stationsdaten ändern.....	32

# Verteilungsdiagramm








## Zusammenfassung

Name	Dokumentation
Android Smartphone	<b>Beschreibung:</b> Über das Android Smartphone wird die App genutzt.
Server PC	<b>Beschreibung:</b> Der ServerPC ist entsprechend des Namens ein als Server genutzter PC.
Dalvik Virtual Machine	<b>Beschreibung:</b> Auf der Dalvik Virtual Maschine läuft die eigentliche App.
Java Virtual Machine	<b>Beschreibung:</b> Auf der Java Virtual Maschine läuft Apache Tomcat.
Apache Tomcat	<b>Beschreibung:</b> Apache Tomcat ist der Webserver, der es ermöglicht die Web-Anwendungen auszuführen.

# Komponentendiagramm



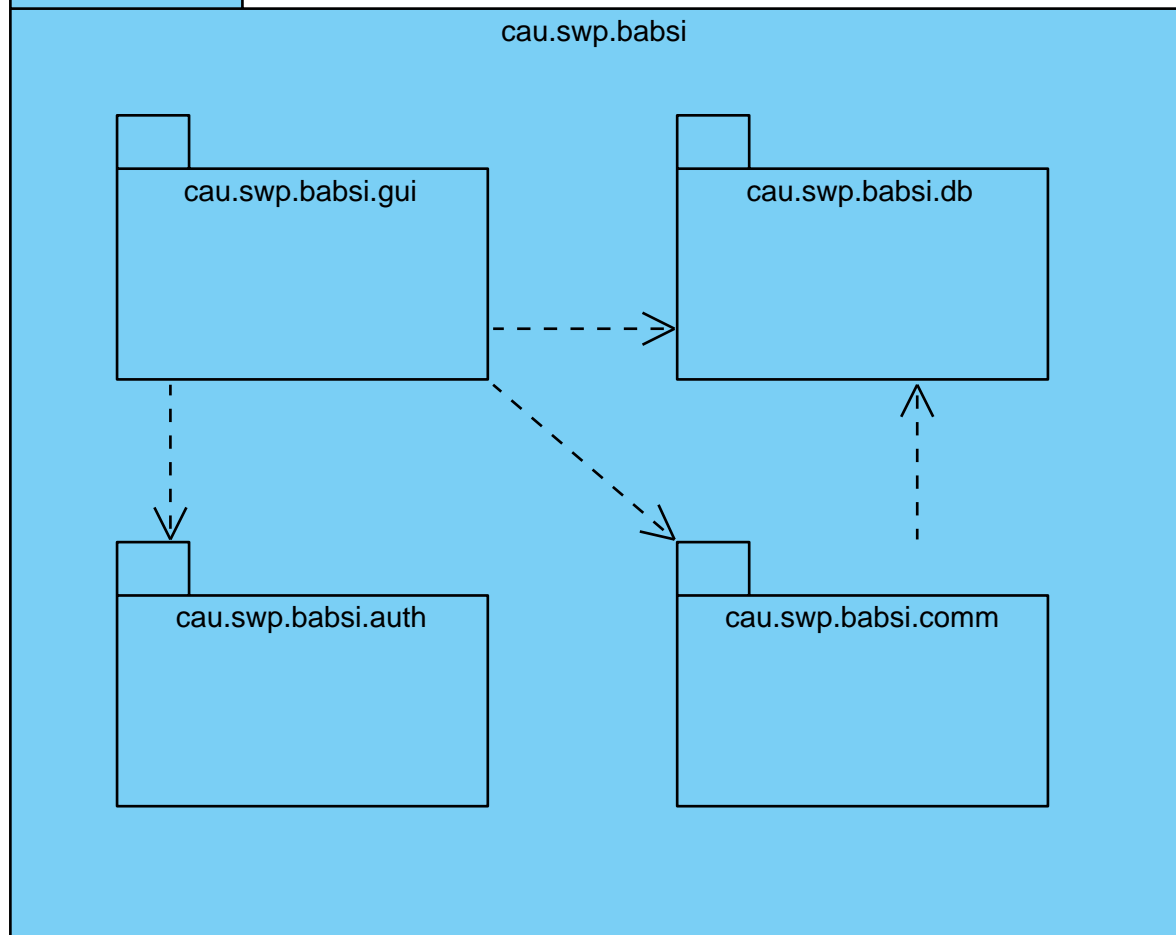
## Zusammenfassung

Name	Dokumentation
 Datenbank	<b>Beschreibung:</b>  Hier werden die (alten) Checklisten gespeichert um für spätere Nutzung, zum Beispiel zur Erstellung von Statistiken, abgerufen werden zu können.
 BabsiServer	<b>Beschreibung:</b>  Der BabsiServer ist der Systemteil, auf dem die Webanwendung läuft. Außerdem greift der Server auf die Datenbank zu.
 BabsiApp	<b>Beschreibung:</b>  BabsiApp ist der Teil des Systems, der als App auf Android-Smartphones läuft.
 Datenbank Schnittstelle	<b>Beschreibung:</b>  Über die DatenbankSchnittstelle kommunizieren Datenbank und BabsiServer.
 Server Schnittstelle	<b>Beschreibung:</b>  Bei der ServerSchnittstelle handelt es sich entsprechend des Namens um eine Schnittstelle des BabsiServer. Hierüber läuft die Kommunikation mit der App auf den Android-Smartphones.







# App-Paketdiagramm

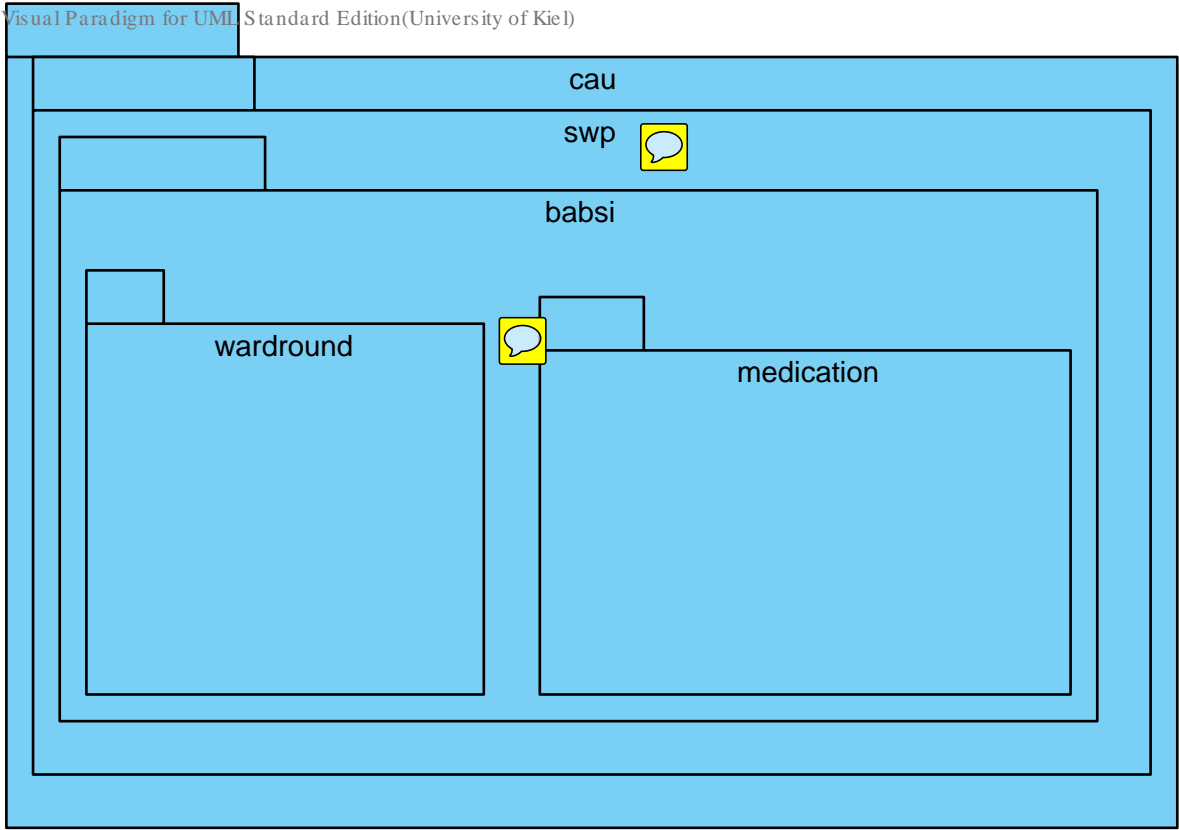
Visual Paradigm for UML Standard Edition (University of Kiel)








## Zusammenfassung

Name	Dokumentation
 cau.swp.babsi.gui	Hierzu gehören alle Klassen, die zur Oberflächengestaltung gehören.
 cau.swp.babsi.db	Zu .db gehören jene Klassen, welche die Schnittstelle zur Datenbank realisieren.
 cau.swp.babsi.auth	In diesem Paket sind alle Klassen enthalten, die ein System zur Authentifikation ermöglichen.
 cau.swp.babsi.comm	Alle Klassen, die der Synchronisation der Daten dienen werden hier gesammelt.

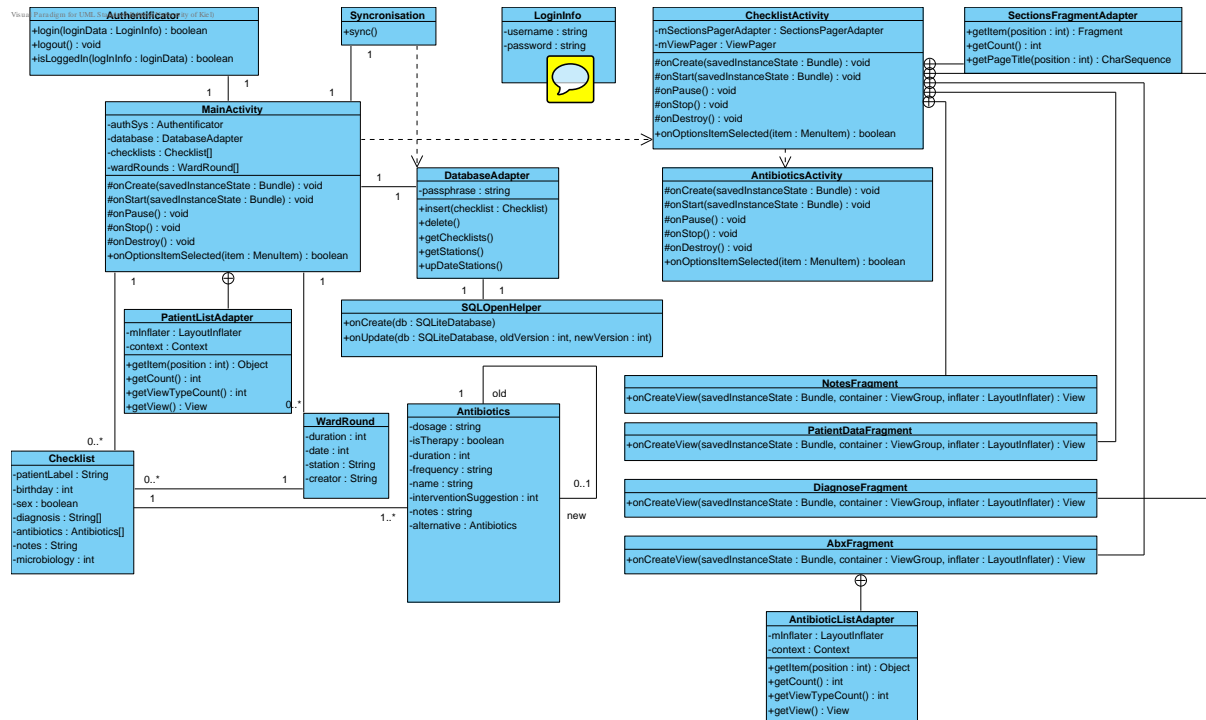
# Web-Paketdiagramm






## Zusammenfassung




Name	Dokumentation
 cau	Namenskonvention.
 swp	Namenskonvention.
 babsi	Hauptpaket des Projekts, beinhaltet alle übergreifend genutzten Klassen.
 wardround	Klassen, die speziell für die Verarbeitung von Checklisten benötigt werden
 medication	Beinhaltet alle Klassen, die für die Darstellung von Medikamenten und deren Anwendung benötigt werden.




# App-Klassendiagramm












## Zusammenfassung

Name	Dokumentation
 <b>Authenticator</b>	<p>Dient als Authentifizierungssystem. User werden hier ein- und ausgeloggt. Weiter kann abgefragt werden, ob ein User bereits im System eingeloggt ist.</p> <p><b>Methoden:</b></p> <p><u>+login(loginData : LoginInfo) : boolean</u> - loggt den User ins System ein  <i>Parameter:</i>  loginData - Informationen zum User</p> <p><u>+logout() : void</u> - loggt den User aus dem System aus</p> <p><u>+isLoggedIn(loginData : LoginInfo) : boolean</u> - prüft ob der gegebene User eingeloggt ist  <i>Parameter:</i>  loginData - Informationen zum User</p>
 <b>Synchronisation</b>	<p><b>Beschreibung:</b></p> <p>Diese Klasse stellt Funktionen bereit, welche dazu dienen eine Synchronisation mit dem Server durchzuführen.</p>
 <b>LoginInfo</b>	<p><b>Beschreibung:</b></p> <p>Hier werden Login-Informationen zu Usern gespeichert.</p> <p><b>Attribute:</b></p> <p><u>-username : String</u> - Nutzernamen  <u>-password : String</u> - Passwort</p>

 ChecklistActivity	<p><b>Beschreibung:</b> Diese Klasse dient dazu, die Oberfläche für die Patientencheckliste bereitzustellen. Für die horizontale Navigation ist der SectionsPagerAdapter zuständig. Bei dem Aufruf von onStart werden die relevanten Daten geladen und den einzelnen Fragmenten weitergeleitet.</p> <p><b>Attribute:</b> -mSectionsPagerAdapter : SectionsPagerAdapter - Adapter für die horizontale Navigation -mViewPager : ViewPager - View für die horizontale Navigation</p> <p><b>Methoden:</b> siehe MainActivity</p>
 SectionsPagerAdapter	<p><b>Beschreibung:</b> Die Klasse dient nur dazu die horizontale Navigation zu ermöglichen. Die implementierten Methoden werden von der Android API benötigt.</p> <p><b>Methoden:</b> +getItem(position : int) : Object - gibt das jeweilige Objekt an gegebener Position zurück <i>Parameter:</i> position - die Position des gesuchten Elementes +getCount() : int - gibt die Anzahl an Fragmenten zurück +getPageTitle(position : int) - gibt den Titel des jeweiligen Fragmentes zurück <i>Parameter:</i> position - die Position des gesuchten Elementes</p>
 MainActivity	<p><b>Beschreibung:</b> Diese Klasse ist die primäre Klasse der Smartphone-App. Beim Start der App sorgt dieser dafür, dass der Hauptbildschirm aufgebaut wird und andere wichtige Klassen (wie DatabaseAdapter, usw.) initialisiert werden. Somit werden die Schnittstellen zur Datenbank und Authorisierung sofort bei App-Start verfügbar gemacht. Weiter kann von hier aus zur ChecklistActivity gegangen werden.</p> <p><b>Attribute:</b> -authSys : Authenticator - Obj. der Authentifizierung -database : DatabaseAdapter - Datenbank-Objekt -checklists : Checklist[]- Liste aller Checklisten der Patienten -wardRound : WardRound[]- Liste aller Visiten</p> <p><b>Methoden:</b> #onCreate(savedInstanceState : Bundle) : void - wird beim erstellen der Activity aufgerufen; Oberfläche wird geladen und aufgebaut <i>Parameter:</i> savedInstanceState - Daten die übergeben werden müssen  #onStart(savedInstanceState : Bundle) : void - wird aufgerufen, wenn die Oberfläche erscheint; Daten laden und in die Elemente einfügen <i>Parameter:</i> savedInstanceState - Daten die übergeben werden müssen  #onPause() : void - wenn die App in den Hintergrund fällt, wird dies aufgerufen; Stoppuhr pausieren;</p>

	<p><u>#onStop() : void</u> - beim beenden der App aufgerufen; ggf Visite stoppen; Daten sichern;</p> <p><u>#onDestroy() : void</u> -</p> <p><u>+onOptionsItemSelected(item : MenuItem) : void</u> - Listener für alle Tasten in der ActionBar; führt nötige Anweisungen aus</p> <p><i>Parameter:</i> item - Das MenuItem, welches angeklickt wurde und diese Funktion aufgerufen hat</p>
 DatabaseAdapter	<p><b>Beschreibung:</b> Dient als Schnittstelle zwischen Datenbank und dem Rest der App. Hierfür bietet die Klasse Funktionen zum auslesen, speichern und löschen einzelner Elemente.</p> <p><b>Attribute:</b> <u>-passphrase: string</u> - Dies ist der sprachliche Ausdruck mit dem die die Verschlüsselung für die Datenbank erstellt wird.</p> <p>Methoden: <u>+insert(checklist: Checklist)</u> - Hiermit werden am Ende einer Visite die einzelnen Checklisten in die Datenbank übertragen.</p> <p><u>+delete()</u> - Hiermit werden Checklisten aus der Datenbank gelöscht nachdem sie auf den Server übertragen wurden.</p> <p><u>+getChecklists()</u> - Hiermit werden die Checklisten wieder aus der Datenbank herausgeholt um sie zum Beispiel an den Server zu übermitteln.</p> <p><u>+getStations()</u> - Gibt die Liste der Stationen aus.</p> <p><u>+updateStations()</u> - Aktualisiert die Liste der Stationen.</p>
 AntibioticsActivity	<p><b>Beschreibung:</b> Diese Klasse stellt das Fenster für die einzelnen Antibiotika bereit.</p> <p><b>Methoden:</b> siehe MainActivity</p>
 SQLOpenHelper	<p><b>Beschreibung:</b> Der SQLOpenHelper erbt von SQLiteOpenHelper und stellt eine Hilfsklasse dar, die dabei unterstützt die verwendete Datenbank zu erstellen und zu bearbeiten.</p> <p><b>Methoden:</b> <u>+onCreate(db : SQLiteDatabase)</u> - Wird aufgerufen wenn die Datenbank erstellt wird. In dieser Methode werden auch die Tabellen angelegt, die in der Datenbank enthalten sind.</p> <p><u>+onUpdate (db : SQLiteDatabase, oldVersion : int, newVersion : int)</u> - Wird aufgerufen um die Datenbank zu aktualisieren.</p>

 PatientListAdapter	<p><b>Beschreibung:</b> Stellt eine Patientenliste bereit, welche auch den Status einzelner Patienten darstellt.</p> <p><b>Attribute:</b>  <u>-mInflater : LayoutInflater</u> - Objekt welches aus Layouts fertige Views erstellt  <u>-context : Context</u> - verwendete ViewGroup der Activity</p> <p><b>Methoden:</b>  <u>+getItem(position : int) : Object</u> - gibt das jeweilige Objekt an gegebener Position zurück  <i>Parameter:</i>  position - die Position des gesuchten Elementes</p> <p><u>+getCount() : int</u> - gibt die Anzahl der Elemente zurück</p> <p><u>+getViewTypeCount() : int</u> - gibt die Anzahl der verschiedenen View-Typen zurück</p> <p><u>+getView() : View</u> - gibt das View der gegebenen Stelle zurück</p>
 NotesFragment	<p><b>Beschreibung:</b> Stellt das Fenster für die Bemerkungen bereit.</p> <p><b>Methoden:</b>  <u>+onCreateView(savedInstanceState : Bundle, container : ViewGroup, inflater : LayoutInflater)</u> - wird aufgerufen sobald ChecklistActivity geladen wird; Daten werden entsprechend geladen und angezeigt  <i>Parameter:</i>  savedInstanceState - übergebene Daten als Paket  container - Die ViewGroup zu der das Fragment hinzugefügt wird  inflater - LayoutInflater welcher genutzt wird um Layouts im Fragment zu erstellen</p>
 Antibiotics	<p><b>Beschreibung:</b> Hier werden Informationen für eine jeweilige Antibiotikatherapie abgelegt.</p> <p><b>Attribute:</b>  <u>-dosage : String</u> - Dosierung der aktuellen Therapie  <u>-isTherapy : boolean</u> - Wenn #t dann therapeutisch, sonst prophylaktisch  <u>-duration : int</u> - Anwendungsdauer  <u>-frequency : String</u> - Verabreichungsintervall  <u>-name : String</u> - Bezeichnung des Antibiotikums  <u>-interventionSuggestion : int</u> - Interventionsmöglichkeit  <u>-notes : String</u> - Notizen zu dieser Therapie  alternative - Alternativvorschlag für neue Therapie  <u>-alternative : Antibiotics</u> - Alternative Behandlung</p>
 WardRound	<p><b>Beschreibung:</b> Speichert die Informationen einer Visite.</p> <p><b>Attribute:</b>  <u>-duration : int</u> - Dauer der Visite  <u>-date : int</u> - Datum und Uhrzeit des Visitenstarts (Unixzeit)  <u>-station : String</u> - Name der Station in dem die Visite durchgeführt wurde</p>

	-creator : String - Name des Mitglieds, welches die Visite durchgeführt hat
 PatientDataFragment	<b>Beschreibung:</b> Stellt das Fenster für die Patientendaten bereit. <b>Methoden:</b> siehe NotesFragment
 Checklist	<b>Beschreibung:</b> Checklist speichert eine komplette Checkliste für einen Patienten. <b>Attribute:</b> -patientLabel : String - Patientenbezeichnung -birthday : int - Geburtstag des Patienten -sex : boolean - Geschlecht des Patienten -diagnosis : String[] - Liste aller festgestellten Diagnosen -antibiotics : Antibiotics[] - Liste aller Antibiotiker des Patienten -notes : String - Bemerkungen -microbiology : int - Ergebnis der Mikrobiologie
 DiagnoseFragment	<b>Beschreibung:</b> Stellt das Fenster für die Diagnoseauswahl bereit. <b>Methoden:</b> siehe NotesFragment
 AbxFragment	<b>Beschreibung:</b> Stellt das Fenster für die Antibiotika bereit. <b>Methoden:</b> siehe NotesFragment
 AntibioticListAdapter	<b>Beschreibung:</b> Stellt die Liste für die verabreichten Antibiotika bereit. <b>Attribute:</b> siehe PatientListAdapter <b>Methoden:</b> siehe PatientListAdapter

The diagram is a UML class diagram for a medical system, divided into two main packages: **Domain** and **Controller**.

**Domain Package:**

- medication** (package):
  - InterventionType**: +string : name
  - Antibiotic**: +name : string
  - Intervention**: +comment : string
  - InterventionStatus**: +description : string
  - DrugData**: +comment : string
  - DrugApplicationType**: +name : string
  - ApplicationRate**: +interval : string
  - Strength**: +strength : string
- wardround** (package):
  - Checklist**: +name : string
  - Infection**: +name : string
  - MibStatus**: +description : string
  - WardRound**: +date : Date, +duration : int
  - Cycle**: +start : Date, +end : Date
  - User**: +username : string, +passwordHash : string
  - Station**: +name : string
  - Patient**: +description : string, +dateOfBirth : date, +gender : string

**Controller Package:**

- ChecklistController**: +index()
- StationController**: +index()
- PatientController**: +index()
- CycleController**: +index()
- UserController**: +index()
- WardRoundController**: +index()
- DataController**: +index(), +export(), +login(), +synchronize(), +signOut()
- StatisticController**: +index()
- AuthController**: +shiroSecurityManager, +index(), +login(), +signIn(), +signOut(), +unauthorized()

**Interfaces:**

- Listable** (interface): +show(id : long), +list(max : Integer)
- Editable** (interface): +edit(id : long), +delete(id : long), +create(), +update(id : long, version : long), +save()










**Associations:**









- InterventionType** (1) to **Intervention** (\*)
- Intervention** (\*) to **InterventionStatus** (1)
- Intervention** (\*) to **DrugData** (\*)
- DrugData** (\*) to **DrugApplicationType** (1..\*)
- DrugApplicationType** (1..\*) to **ApplicationRate** (1)
- DrugApplicationType** (1..\*) to **Strength** (1)
- Station** (1) to **Patient** (\*)
- Station** (1) to **Checklist** (1..\*)
- Patient** (1) to **Checklist** (1..\*)
- Checklist** (1..\*) to **Infection** (1..\*)
- Checklist** (1..\*) to **MibStatus** (1)
- Checklist** (1..\*) to **WardRound** (1..\*)
- WardRound** (1..\*) to **Cycle** (1)
- WardRound** (1..\*) to **User** (\*)
- User** (\*) to **Station** (1)








**Generalization:**








- ChecklistController**, **StationController**, **PatientController**, **CycleController**, **UserController**, and **WardRoundController** all generalize the **Listable** interface.
- ChecklistController**, **StationController**, **PatientController**, **CycleController**, **UserController**, and **WardRoundController** all generalize the **Editable** interface.





Name	Dokumentation
 Domain	Die Domain-Klassen entsprechen der Datenmodellierung, dem "Model" im Model-View-Controller Entwurfsmuster. Es handelt sich nicht um ein Package, die Klassen sind nur aus Gründen der Übersichtlichkeit in diesem Diagramm in die logischen Gruppen "Domain" und "Controller" aufgeteilt. 
 Controller	Die Controller-Klassen werden via Browser über URLs angesprochen und steuern die Datenverarbeitung und -Manipulation wie im Model-View-Controller Entwurfsmuster. Es handelt sich nicht um ein Package, die Klassen sind nur aus Gründen der Übersichtlichkeit in diesem Diagramm in die logischen Gruppen "Domain" und "Controller" aufgeteilt.
 medication	siehe Package-Diagramm
 ChecklistController	Ermöglicht die Anzeige von einzelnen Checkliste, indem es die Daten aus dem Model bezieht und die Daten an die entsprechende View weiterleitet und die Formulare und Listen ausfüllt. 
 InterventionType	<p><b>Beschreibung:</b> Ein Interventionstyp</p> <p><b>Attribute:</b> name (name) Der Name der Interventionsart, entnommen aus der gegebenen Tabelle "Interventionen", z.B. "Dosisoptimierung"</p>
 StationController	<b>Regelt</b> die Stationen, ermöglicht das Auflisten, Bearbeiten und Anlegen.
 Listable	Für Controller, die eine Datenanzeige liefern, genauer eine Liste von Daten des gleichen Typs (list) und einer Einzelansicht (show).

 PatientController	<b>Regelt</b> die Patienten, ermöglicht das Auflisten aller und Anzeigen einzelner Patienten.
 Antibiotic	<b>Beschreibung:</b> Ein Antibiotikum  <b>Attribute:</b> name (string) Der Name des Antibiotikums, entnommen aus der gegebenen Tabelle "Antibiotika"
 Intervention	<b>Beschreibung:</b> Eine Intervention, bestehend aus Interventionsart und einem Kommentar.  <b>Attribute:</b> comment (string) Der Kommentar des ABS-Stewards zu dieser Intervention  type (InterventionType) Der Typ der Intervention
 InterventionStatus	<b>Beschreibung:</b> Eine Rückmeldung des behandelnden Arztes bezüglich der vom ABS-Team vorgeschlagenen Intervention  <b>Attribute:</b> description (string) Die Beschreibung der Rückmeldung, z.B. "umgesetzt"
 Strength	<b>Beschreibung:</b> Eine Stärke der Medikation  <b>Attribute:</b> strength (name) Die Stärke der Medikation, entnommen aus der gegebenen Tabelle, z.B. "40mg" oder "körpergewichtsadaptiert"
 CycleController	Generiert die Daten für den eine Auflistung von Visitenturnussen, bei denen angezeigt wird, auf welcher Station im aktuellen Turnus bereits eine Visite durchgeführt wurde.
 Editable	Repräsentiert Controller die das Bearbeiten von Datensätzen erlauben, also Anlegen, Bearbeiten und Löschen von Daten.
 DrugData	<b>Beschreibung</b> Bündelung der Informationen zu einem verabreichten Antibiotikum: Antibiotikum, Stärke, Häufigkeit, Verabreichungsart, Intervention, Status der Interventionsumsetzung durch den Arzt  Die Attribute entsprechen den gegebenen Auswahlmöglichkeiten aus den Tabellen.  <b>Attribute:</b> antibiotic (Antibiotikum)

	<p>applicationType (DrugApplicationType)</p> <p>strength (Strength)</p> <p>applicatioRate (ApplicationRate)</p> <p>intervention (Intervention)</p> <p>interventionStatus (InterventionStatus)</p>
 DrugApplicationType	<p><b>Beschreibung:</b> Eine Verabreichungsart</p> <p><b>Attribute:</b> name (string) Name der Verabreichungsart, "iv" oder "oral"</p>
 ApplicationRate	<p><b>Beschreibung:</b> Eine Häufigkeit der Medikation</p> <p><b>Attribute:</b> interval (name) Die Häufigkeit der Medikation, entnommen aus der gegebenen Tabelle "Häufigkeiten", z.B. "4x täglich"</p>
 UserController	Ermöglicht das Verwalten von Benutzerkonten für das System. Wird in Kooperation mit Apache Shiro genutzt.
 WardRoundController	Bereitet Informationen für eine Visite auf, also welche Checklisten bei einer Visite gespeichert wurden.
 wardround	siehe Package-Diagramm
 Infection	<p><b>Beschreibung:</b> Eine Infektion</p> <p><b>Attribute:</b> name (string) Der Name der Infektion, entnommen aus der gegebenen Tabelle "Infektionsdiagnosen"</p>
 Checklist	<p><b>Beschreibung:</b> Ein vom ABS-Team bezüglich eines Patienten ausgefülltes Formular</p> <p><b>Attribute:</b> patient (Patient) Der zugehörige Patient</p> <p>wardRound (WardRound) Die zugehörige Visite</p> <p>infection (Infection) Die Infektion des Patienten</p> <p>mibiStatus (MiBiStatus)</p>

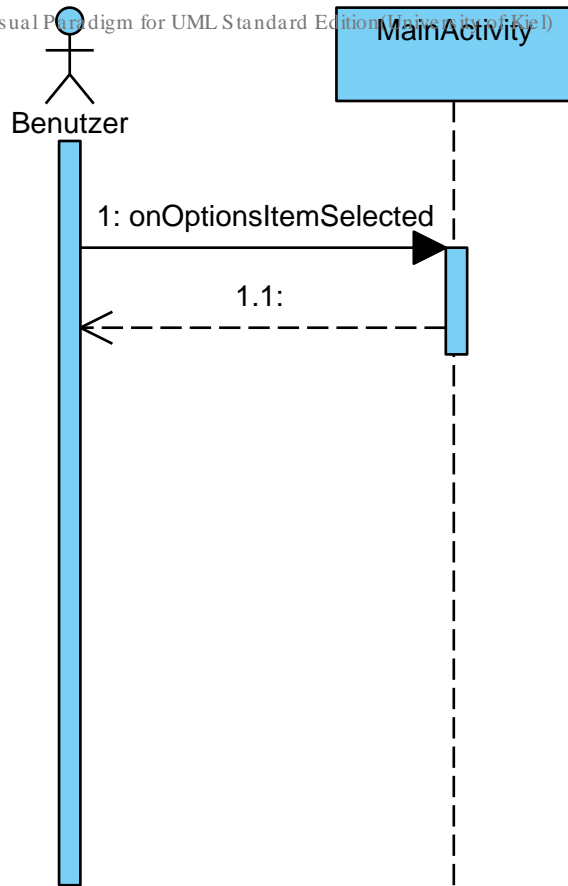
	<p>Der Status der Rückmeldung aus der Mikrobiologie</p> <p>drugs (list&lt;DrugData&gt;) Die verschriebenen Antibiotika und die damit verbundenen Informationen</p>
 Patient	<p><b>Beschreibung:</b> Ein Patient</p> <p><b>Attribute:</b> description (string) Die Bezeichnung des Patienten</p> <p>dateOfBirth (date) Das Geburtsdatum des Patienten</p> <p>gender (string) Das Geschlecht des Patienten, also "female" oder "male"</p> <p>station (Station) Die Station auf welcher der Patient liegt</p> <p>checklists (list&lt;Checklist&gt;) Die von ABS-Teams ausgefüllten Checklisten bezüglich dieses Patienten</p>
 Station	<p><b>Beschreibung:</b> Eine Station</p> <p><b>Attribute:</b> name (string) Der Name der Station</p> <p>patients (list&lt;Patient&gt;) Die auf der Station liegenden Patienten</p>
 AuthController	<p>Authorisiert Benutzer und wird weiter angesprochen, wenn Authorisierungen für andere Controller oder Methoden fehlen. In Zusammenarbeit mit Apache Shiro</p>
 DataController	<p>Regelt den Im- und Export von Rohdaten, also dem Exportieren von Checklisten als CSV Datei und der Kommunikation mit der App via REST und Co.</p>
 MiBiStatus	<p><b>Beschreibung:</b> Ein Status für die Rückmeldung der Mikrobiologie</p> <p><b>Attribute:</b> description (string) Die Rückmeldung der Mikrobiologie</p>
 StatisticController	<p>Sammelt die Daten, die für die drei geforderten Statistiken benötigt werden und bereitet sie so auf, dass sie mit Flot genutzt werden können.</p>
 User	<p><b>Beschreibung:</b> Ein Benutzer</p>

	<p><b>Attribute:</b>  username (string)  Der im System verwendete Name des Benutzers</p> <p>passwordHash (string)  Der hinterlegte Hashwert des Nutzerpassworts, das zur Authentifizierung des Benutzers verwendet wird</p>
 Cycle	<p><b>Beschreibung:</b>  Ein "Cycle" repräsentiert einen vollständigen Visitenturnus und beginnt entsprechend mit der ersten Visite ("WardRound") nach einem Turnus und endet, wenn auf allen Stationen eine Visite durchgeführt worden ist. Dies spiegelt sich auch in den Date-Attributen "start" und "end" wieder.</p> <p><b>Attribute:</b>  start (Date)  Der Startzeitpunkt eines Visitenturnus, entspricht dem Datum der ersten Visite des Turnus.</p> <p>end (Date)  Der Endzeitpunkt eines Visitenturnus, entspricht dem Datum der letzten Visite eines Turnus.</p> <p>wardRounds (List&lt;WardRound&gt;)  Die zu diesem Turnus gehörenden Visiten (WardRound).</p>
 WardRound	<p><b>Beschreibung:</b>  Eine Visite</p> <p><b>Attribute:</b>  station (Station)  Die Station, auf der die Visite durchgeführt wurde</p> <p>user (User)  Der Benutzer, der die Visite auf der App durchgeführt hat.</p> <p>checklists (list&lt;Checklist&gt;)  Die im Rahmen dieser Visite ausgefüllten Formulare bezüglich der einzelnen Patienten</p>



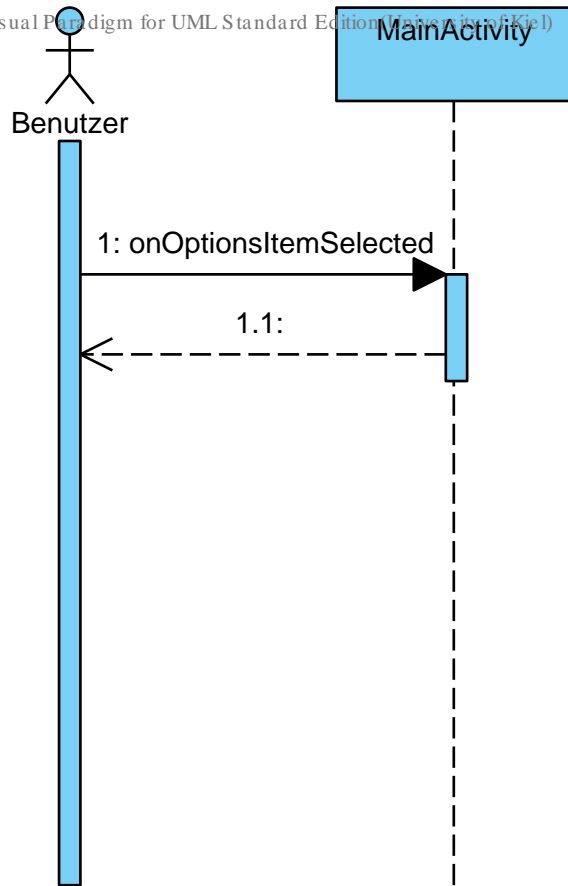
# A2 Visite fortsetzen

Visual Paradigm for UML Standard Edition (Version 8.12.0)

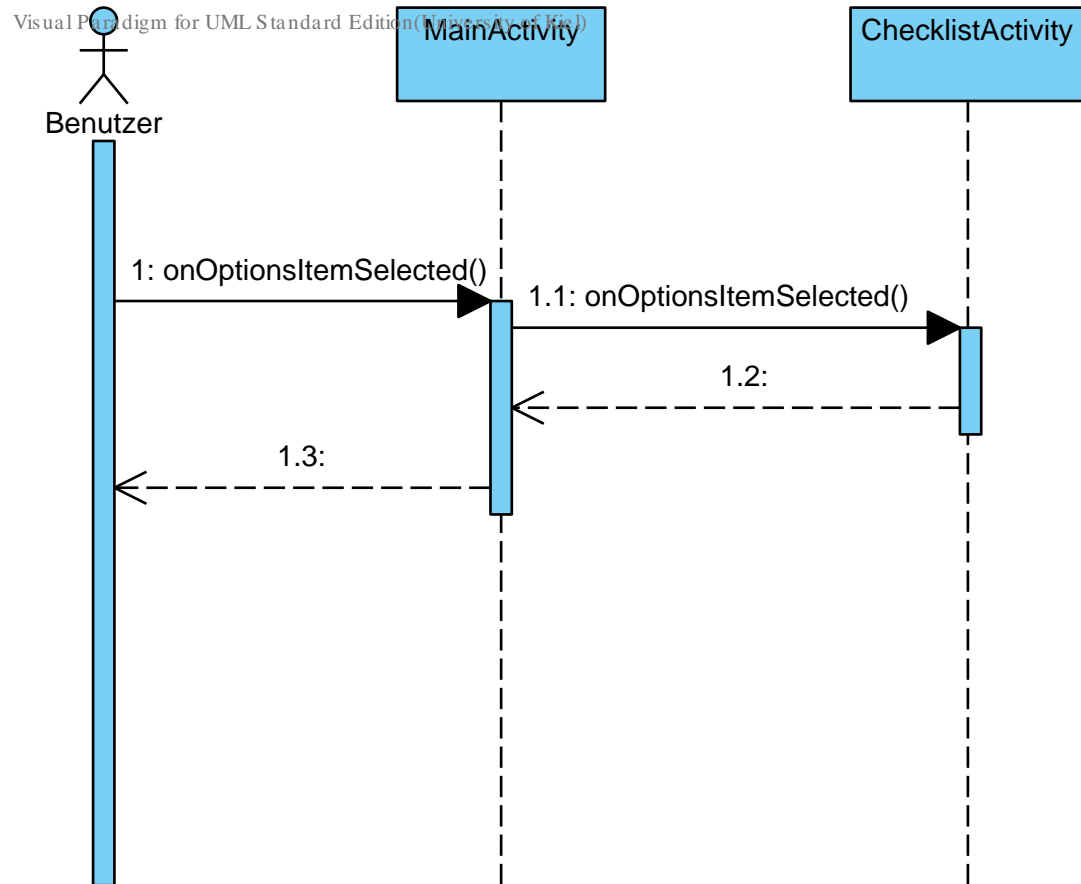


# A3 Visite-pausieren

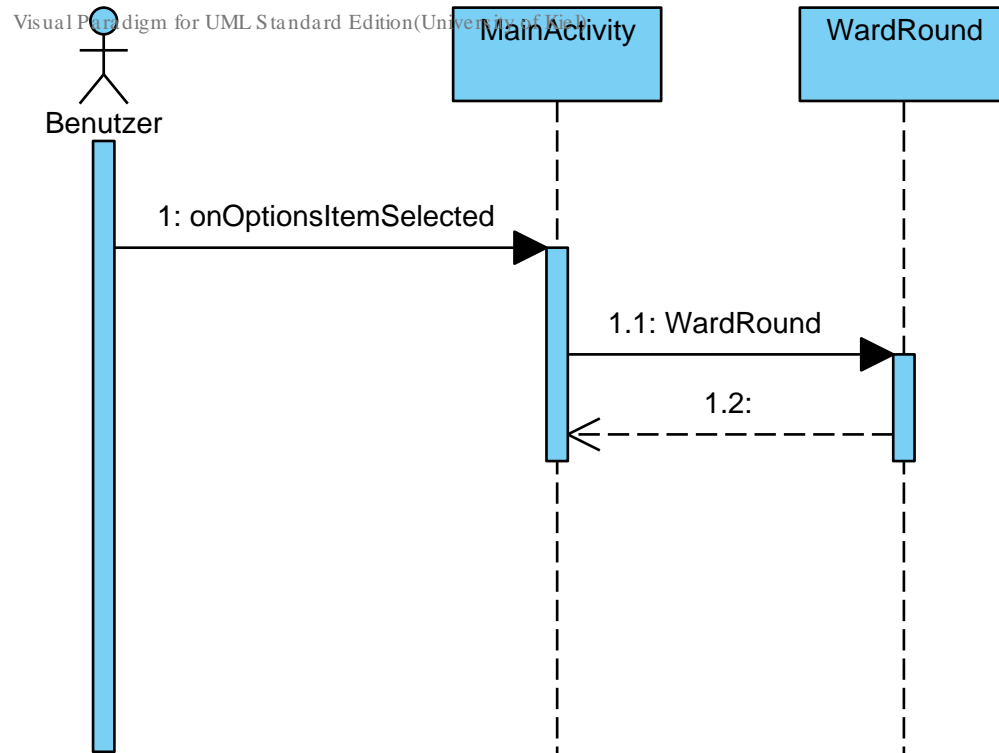
Visual Paradigm for UML Standard Edition (Version 8.12.0)



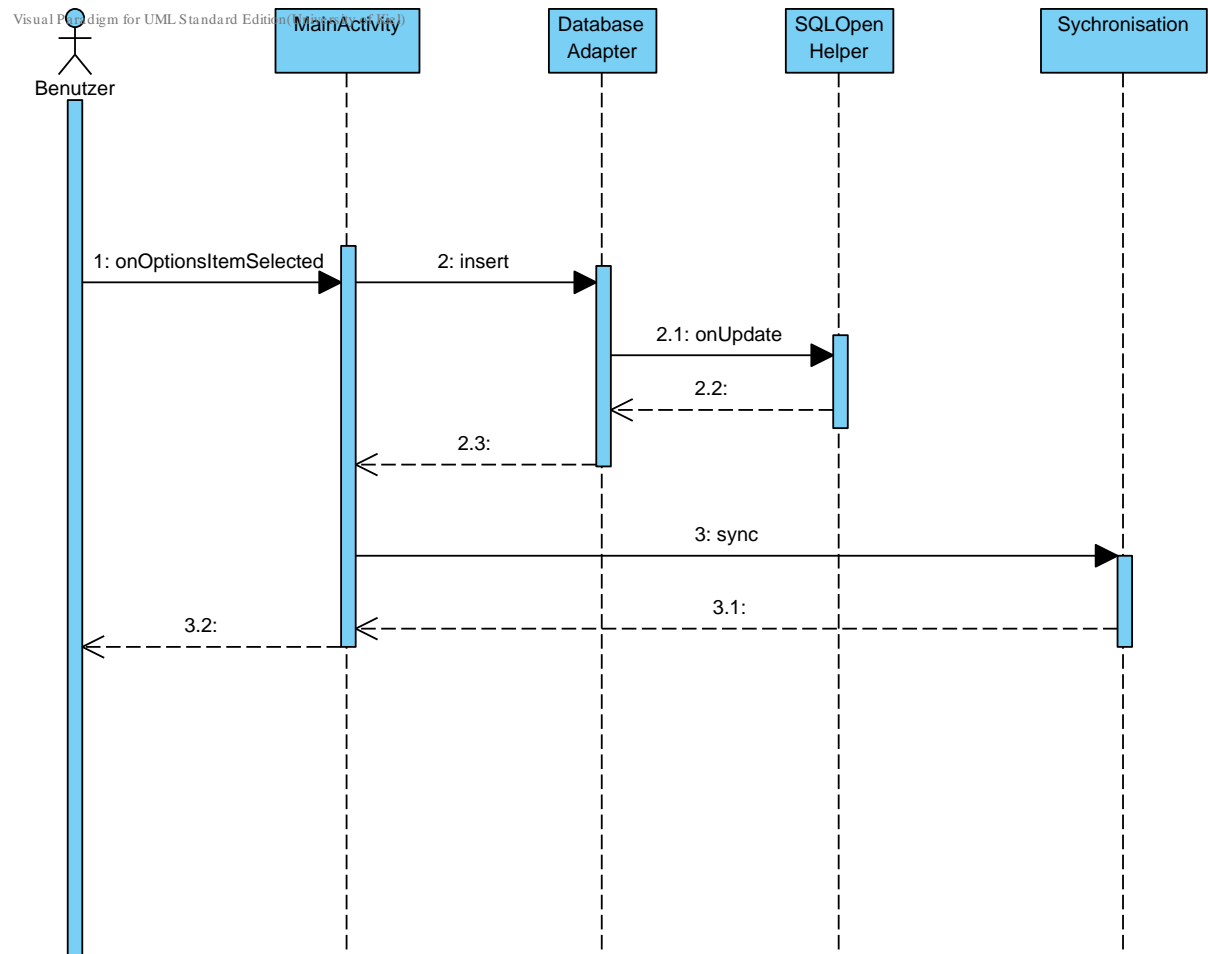
# A4 Checkliste ausfüllen



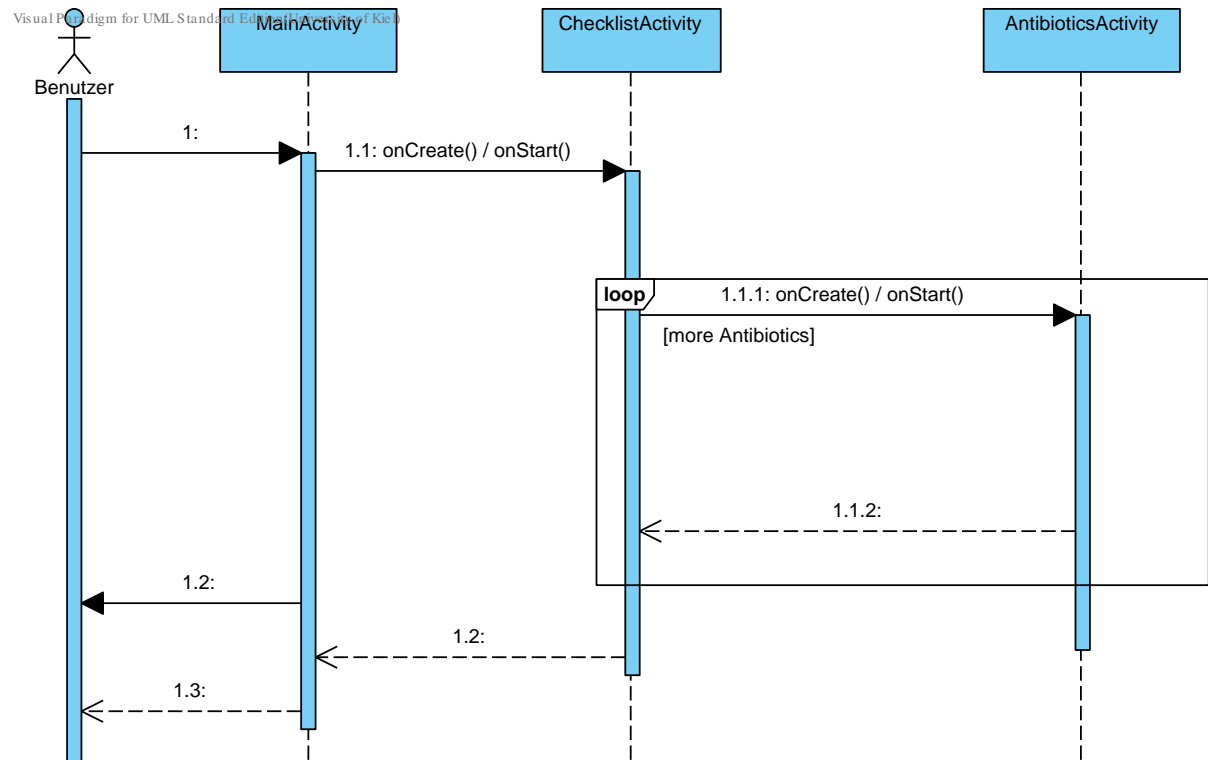
# A5 Visite starten



# A6 Visite beenden

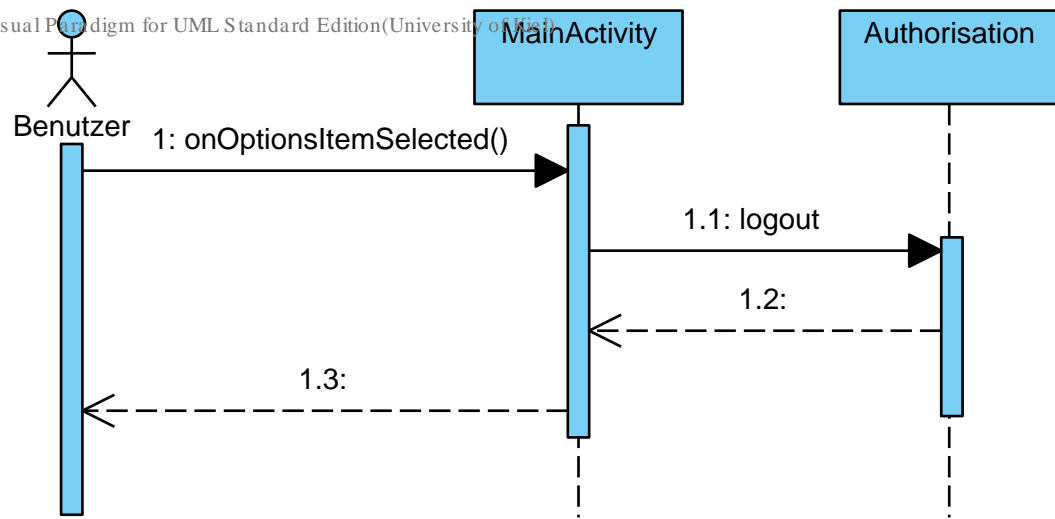


# A7 Patient hinzufügen



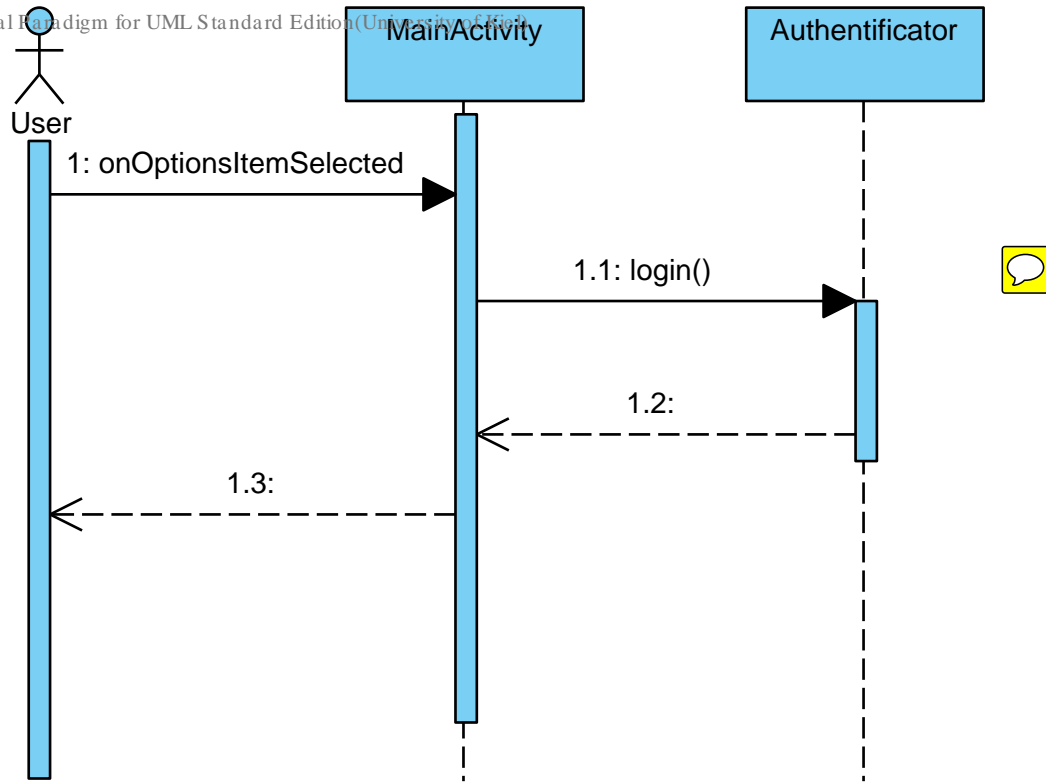
# A8 Logout

Visual Paradigm for UML Standard Edition (University of Wuppertal)



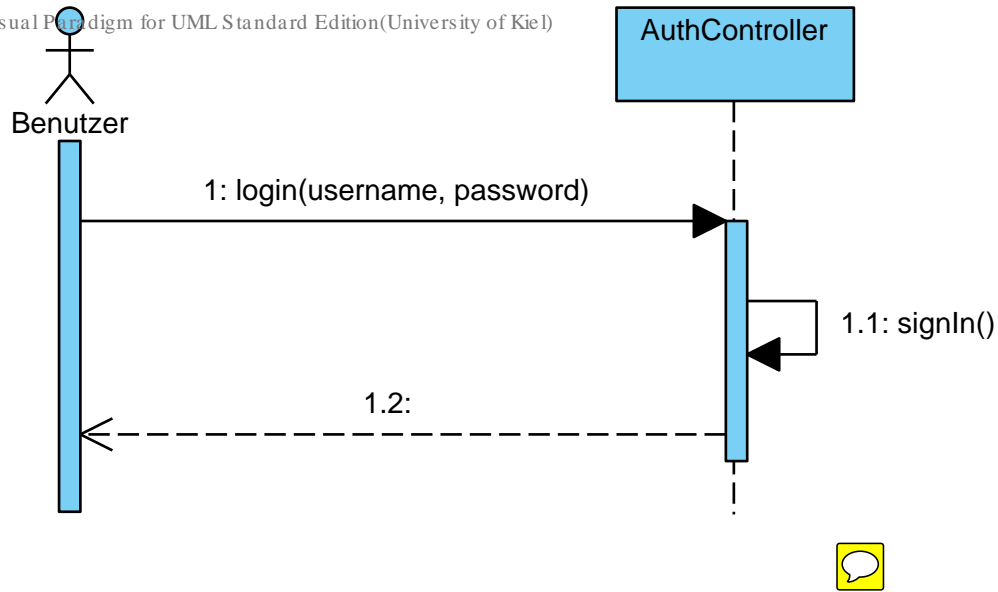
# A9 Login

Visual Paradigm for UML Standard Edition (University Edition)

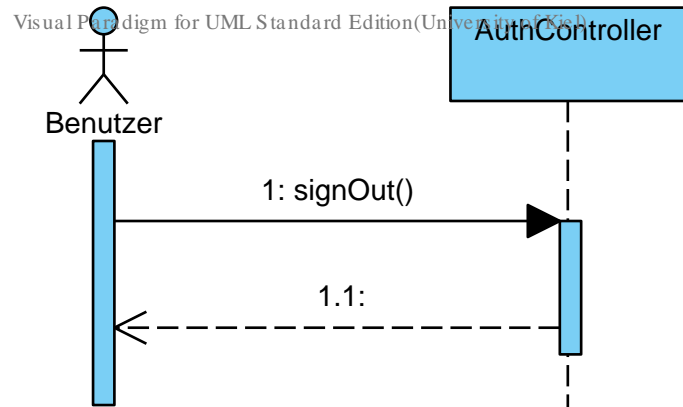


# W1 Login

Visual Paradigm for UML Standard Edition (University of Kiel)

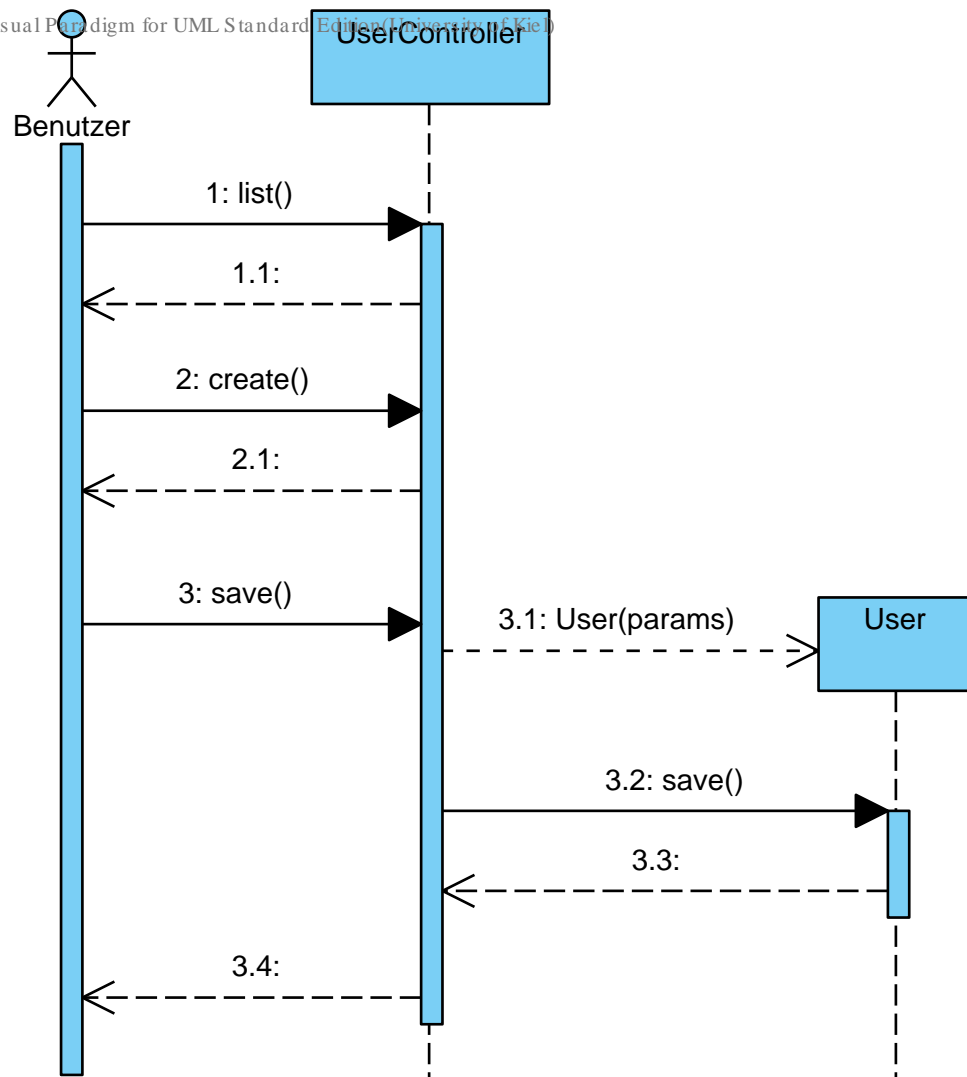


# W3 Logout

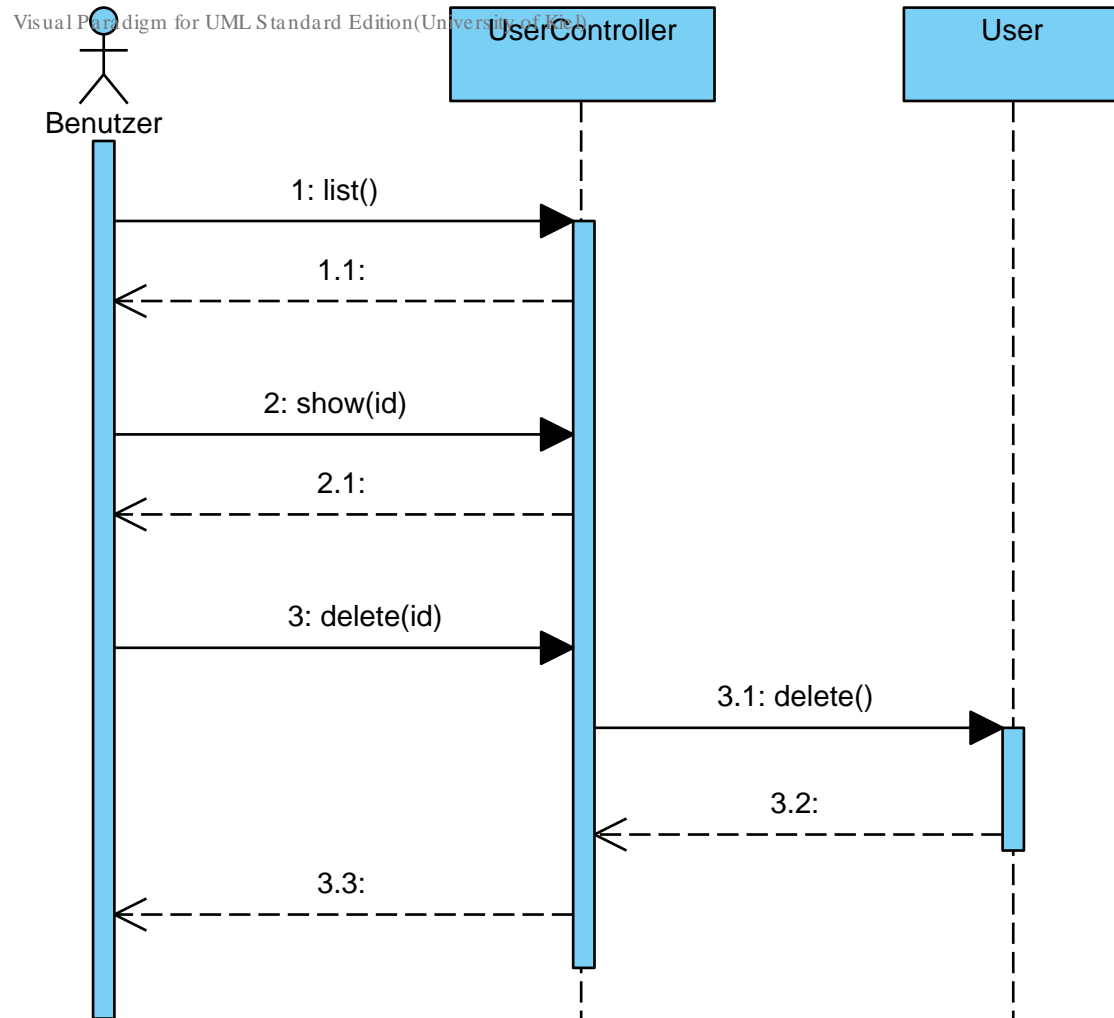


# W2 Benutzer anlegen

Visual Paradigm for UML Standard Edition (Version 4.1.0.10)

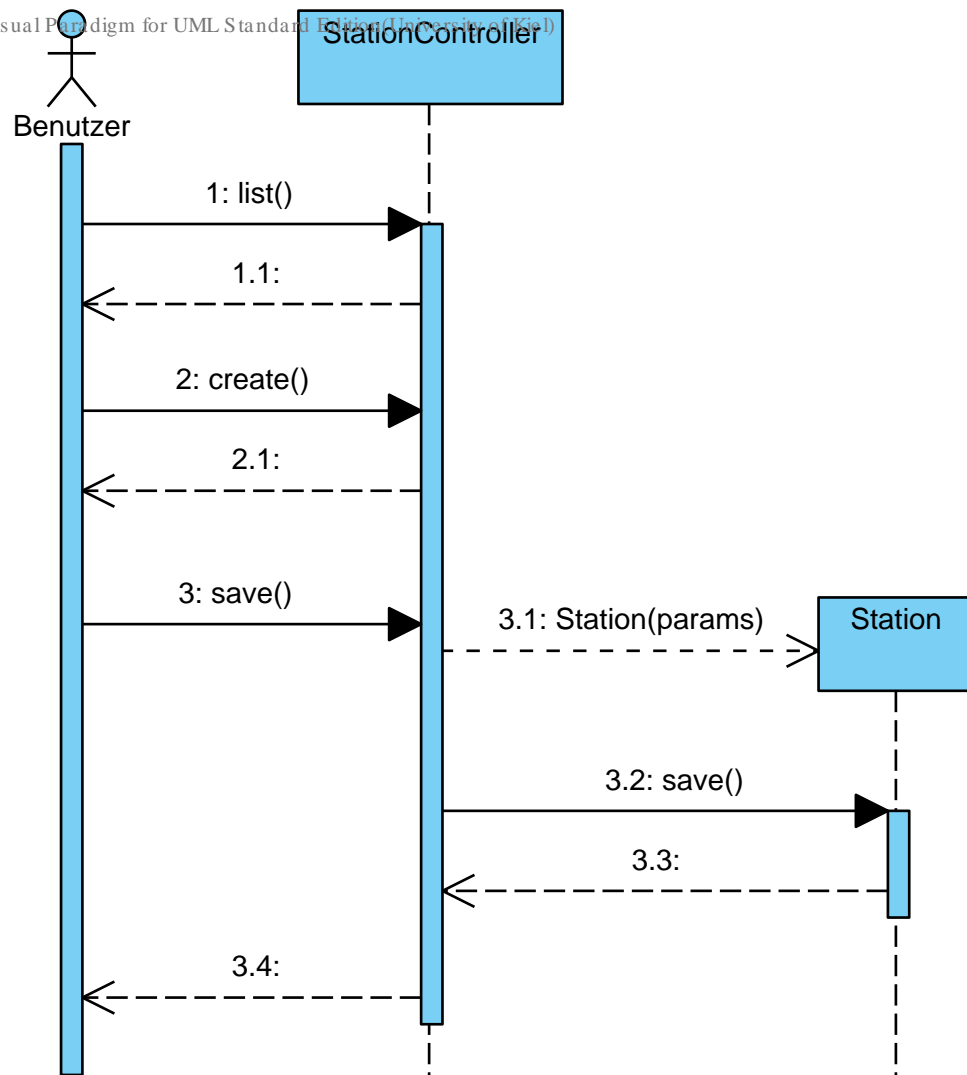


# W4 Benutzer löschen

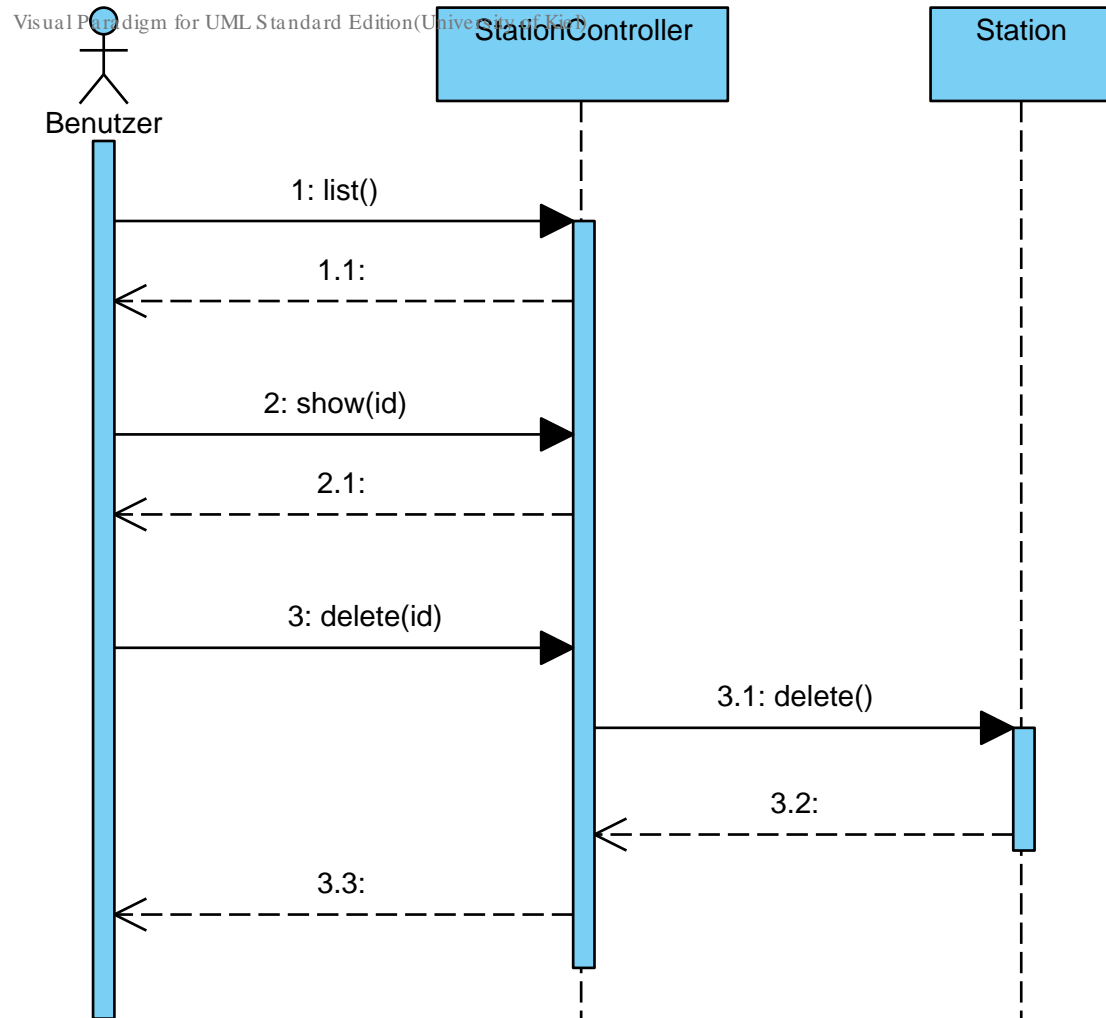


# W6 Station anlegen

Visual Paradigm for UML Standard Edition (License: 6061)

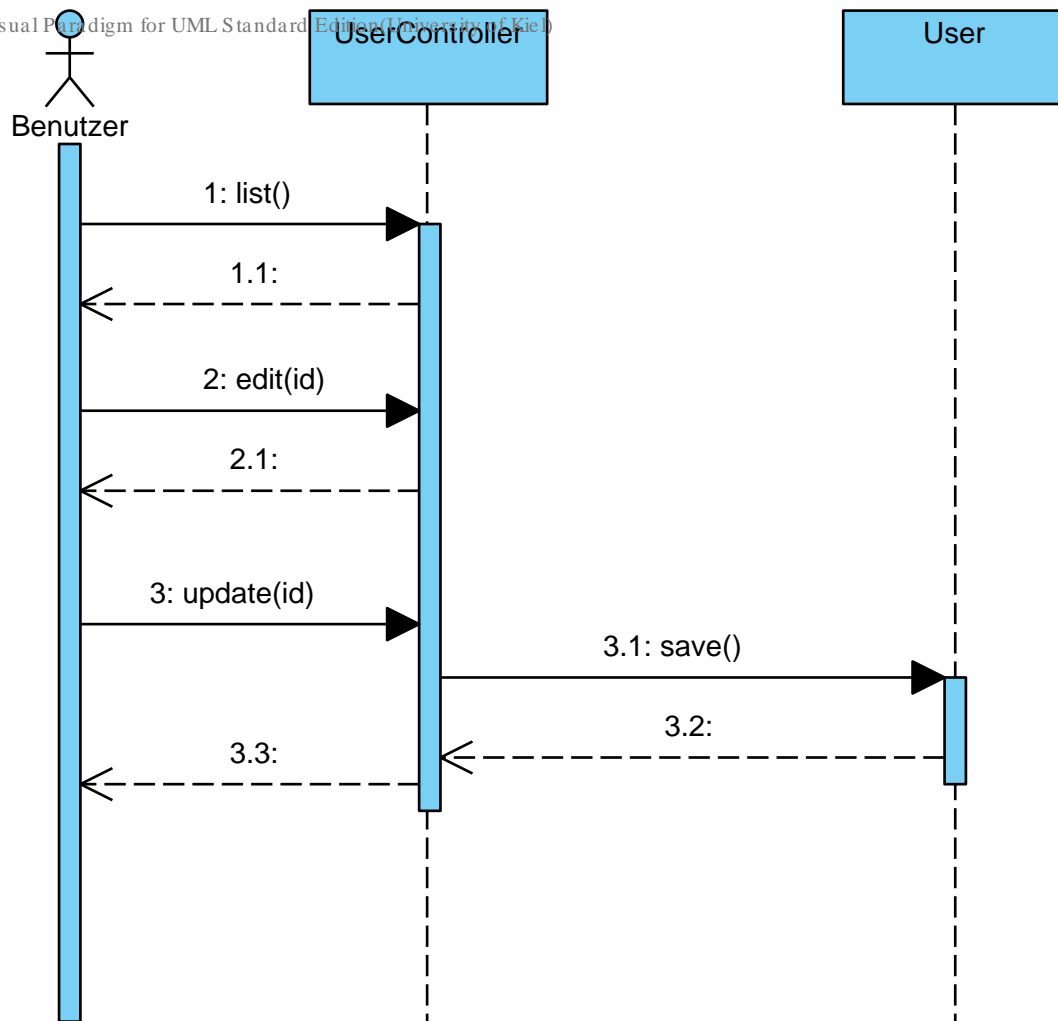


# W7 Station löschen



# W8 Benutzerdaten ändern

Visual Paradigm for UML Standard Edition (Version 1.6.0.0)



# W10 Stationsdaten ändern

