



Dokumentation - MindMap

Dokumentation für das Modul Projekt 1

Studiengang: [Informatik]
Autoren: [Dominik Meister & Lorenz Rasch]
Betreuer: [Prof. Dr. Olivier Biberstein]
Datum: 02.04.2019

Versionen

Version	Datum	Status	Bemerkungen
0.1	13.03.2019	Entwurf	Beginn Projekt
0.2	02.04.2019	Entwurf	Bericht nach BFH Vorlage

Inhaltsverzeichnis

1. Vision	1
2. Analysis	3
2.1. Domainmodel	3
2.2. Use Cases	4
2.3. User Stories	5
2.4. Systemkontext	6
2.5. Anforderungen	6
3. Design	7
3.1. System Sequence Diagram	7
3.2. Sequence Diagram	8
3.3. UML Diagramme	14
4. Entwicklung	15
4.1. Sprint 1	15
4.2. Sprint 2	15
4.3. Sprint 3	15
4.4. Sprint 4	15
5. Fazit	17
Abbildungsverzeichnis	19
A. Use Cases	21

1. Vision

Jeder kennt diese Situation, man sitzt in einem Meeting und möchte kurz ein Brainstorming machen. Oder man möchte für die nächste Arbeit zu einem Thema eine Übersicht zu den Informationen erstellen. Eine der einfachsten Methoden so eine Übersicht darzustellen ist das Mindmap. Das Mindmap auch Gedankenlandkarte genannt ist eine Technik welche von Tony Buzan geprägt und entwickelt wurde. Das Mindmap basiert auf dem Prinzip der Assoziation. Dies kommt nicht von ungefähr, unser Gehirn arbeitet ebenfalls mit Assoziationen, es versucht ständig neue Informationen mit gewissen Kategorien und anderen Informationen zu verknüpfen. Das Mindmap basiert auf derselben Technik, deshalb fällt es uns auch sehr einfach ein Mindmap zu erstellen. Dieses zu erstellen ist jedoch mit den meisten Programmen eher mühsam, deshalb greift man auf den Stift und Papier zurück. Um hier Abhilfe zu schaffen kommt unser Projekt ins Spiel.

Ziel unseres Projektes, ist das Entwickeln einer Software mit welcher man möglichst einfach und schnell ein Mindmap erstellen kann. Dabei werden wir besonders Wert auf die Benutzerfreundlichkeit legen. Es sollte möglich sein in kürzester Zeit ein Mindmap zu erstellen. Das Programm sollte aber auch reif sein für komplexere Mindmaps. Deshalb wird das Programm auch Funktionen wie verschiedene Verbindungstypen und Farben unterstützen, um auch komplexere Mindmaps übersichtlich zu gestalten. Wichtige Funktionen welche das Programm ebenfalls bieten muss sind: das Speichern, Laden und Drucken/Exportieren der Mindmaps.

Das Programm wird als Standalone Software in JavaFX erstellt. JavaFX ist ein Framework von Oracle welches auf die Erstellung von GUI's und Multimedialen Inhalten spezialisiert ist.

Wir hoffen wir können durch dieses Projekt vielen Menschen helfen Ihre Ideen mithilfe unseres Programmes festzuhalten.

2. Analysis

In diesem Abschnitt wird eine grobe Übersicht über die Software erstellt. Fragen wie, was sind die Anforderungen an die Software, was sind die Anforderungen der Akteure etc. werden hier beantwortet. Ebenfalls wird eine grobe Übersicht über die Programmstruktur erstellt.

2.1. Domainmodel

In diesem Abschnitt wird das Domain Model ("Fachmodell") des Mindmap Programmes beschrieben. Es stellt die wichtigsten Fachklassen und Assoziationen dar und zeigt die jeweiligen Multiplizitäten.

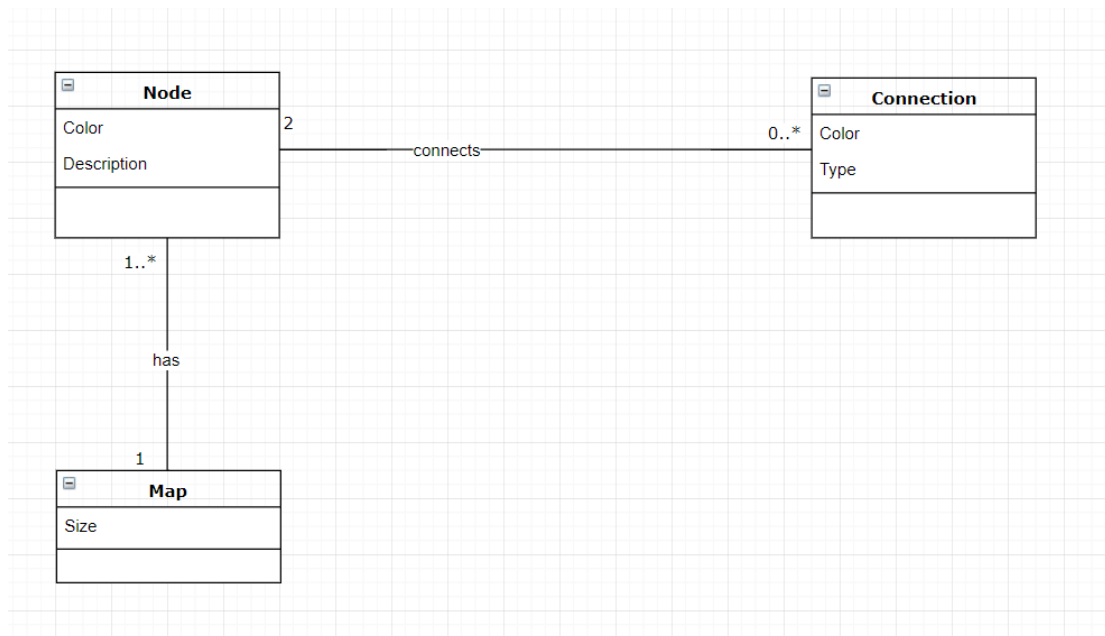


Abbildung 2.1.: Domainmodel

Wie oben in der Grafik zu sehen ist, haben wir uns dafür entschieden, dass eine Map mindestens einen Knoten haben muss um eine Map zu sein. Im Moment denken wir das die Connection und die Map keine direkte Verbindung benötigen dies könnte sich aber beim Implementieren vielleicht noch ändern. Ebenfalls haben wir definiert, dass eine Verbindung immer zwischen 2 Knoten besteht.

2.1.1. Assoziationen

- Zwei Knoten haben keine oder mehrere Verbindungen. Eine Verbindung besteht aus 2 Knoten.
- Eine Map hat einen oder Mehrere Knoten. Ein Knoten gehört zu einer Map.

2.2. Use Cases

Use-Cases beschreiben die Requirements der einzelnen Akteure, es zeigt die Abläufe des Programmes auf. Zeigt dabei aber nicht wie die einzelnen Teile funktionieren.

Eine Liste der Use Cases befindet sich im Anhang A.

2.2.1. Use Case Diagramm

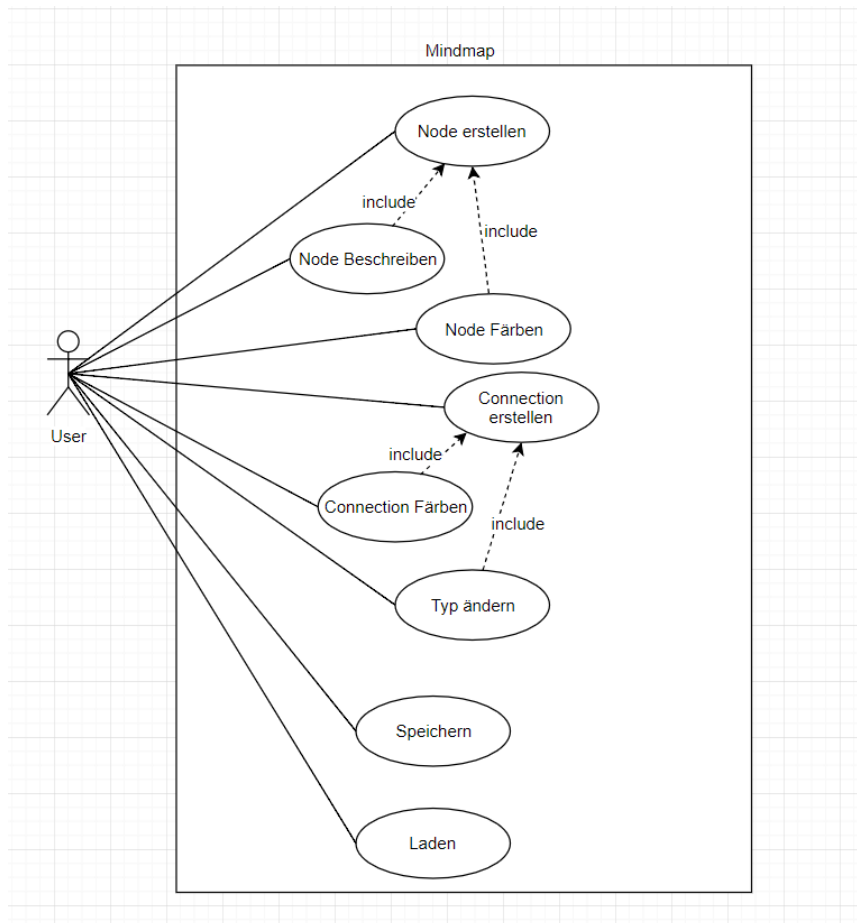


Abbildung 2.2.: Use Case Diagramm

2.2.2. Mindmap Process

Neues Mindmap Szenario:

1. Benutzer erstellt neues Mindmap
2. GUI Service zeigt leeres Mindmap an
3. Benutzer erstellt Nodes und Connections
4. Benutzer bearbeitet die Nodes und Connections z.B. Typ, Beschreibung, Farbe.
5. GUI Service stellt dies dar
6. Punkt 3-5 werden wiederholt bis der Benutzer zufrieden ist
7. Benutzer speichert seine Arbeit

8. GUI Service übergibt Fileservice die Map
9. Fileservice speichert die Map in ein Dokument

Mindmap bearbeiten Szenario:

1. Benutzer lädt Mindmap
2. Fileservice lädt das alte Mindmap
3. GUI Service zeigt das Mindmap an (Danach analog zu oberem Szenario)

2.3. User Stories

Eine User-Story beschreibt in ein oder zwei Sätzen eine gewünschte Funktion des Programmes.

1. Als Benutzer möchte ich, dass das Programm selbsterklärend und einfach zu bedienen ist.
2. Als Benutzer möchte ich ein Programm welches schnell startet und sich nicht langsam anfühlt.
3. Als Benutzer möchte ich ein optisch Ansprechendes Design. Als Benutzer finde ich ein Programm besser wenn sein Design State of the Art ist.
4. Als Benutzer möchte ich, dass das Mindmap auf dem Papier gleich aussieht wie es im Programm ausgesehen hat.
5. Als Benutzer möchte ich nicht einen Experten konsultieren müssen um ein so triviales Programm zu installieren.
6. Als Benutzer möchte ich mit dem Programm neue Mindmaps erstellen können.
7. Als Benutzer möchte ich neue Knoten hinzufügen können, welcher eine Beschreibung besitzt und wenn gewünscht auch eine spezielle Farbe haben kann.
8. Als Benutzer möchte ich die verschiedenen Informationen, also Knoten miteinander vernetzen und so gruppieren können.
9. Als Benutzer möchte ich meine Arbeit speichern können und gespeicherte Projekte auch wieder laden können
10. Als Benutzer möchte ich die Beschreibung eines Knoten auch nach seiner Erstellung bearbeiten können.
11. Als Benutzer möchte ich die Farbe eines Knotens nach seiner Erstellung bearbeiten können.
12. Als Benutzer möchte ich verschiedene Verbindungstypen definieren können, ich möchte Verbindungen zwischen verschiedenen Themen speziell hervorheben können.
13. Als Benutzer möchte ich mein Mindmap drucken können.
14. Als Benutzer möchte ich mein Mindmap exportieren können, um es zum Beispiel als Bilddatei in einem Worddokument einfügen zu können.
15. Als Benutzer möchte ich ein Mindmap mit möglichst wenigen überschneidenden Verbindungen, das Programm sollte eine Möglichkeit bieten dieses Problem zu lösen.

2.4. Systemkontext

Der Systemkontext ist eine Beschreibung des Systems, seiner Teile und äusserlichen Einwirkungen auf das System.

Der Benutzer kann mit unserer Software ein Mindmap erstellen. Dies geschieht über eine grafische Benutzeroberfläche. Mit Hilfe eines Algorithmus können die Themen und Verbindungen des Mindmap optimal verteilt werden.

Die Benutzeroberfläche und der Algorithmus sind Teil der Software

Die Software greift auf das Dateisystem des Computers zu um ein Mindmap zu speichern oder zu laden. Der Druckservice des Computers erlaubt das Drucken des Mindmaps. Diese beiden Teile liegen ausserhalb unserer Software.

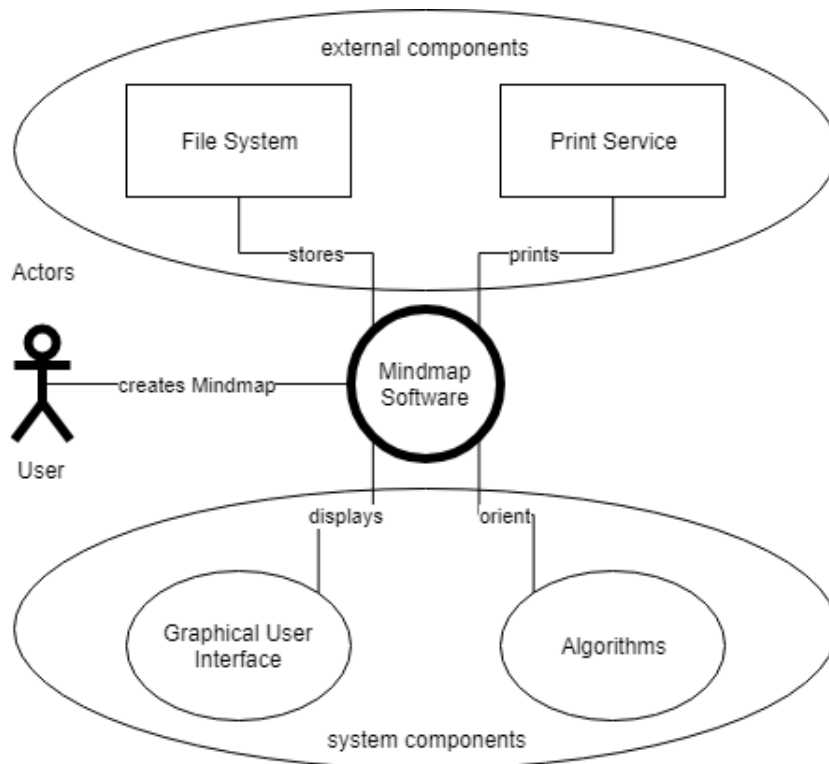


Abbildung 2.3.: Systemkontext

2.5. Anforderungen

→ Teil Modul Requirements Engineering

3. Design

3.1. System Sequence Diagram

Die System Sequence Diagramme zeigen die möglichen Interaktionen von extern mit unserem Mindmap Programm.

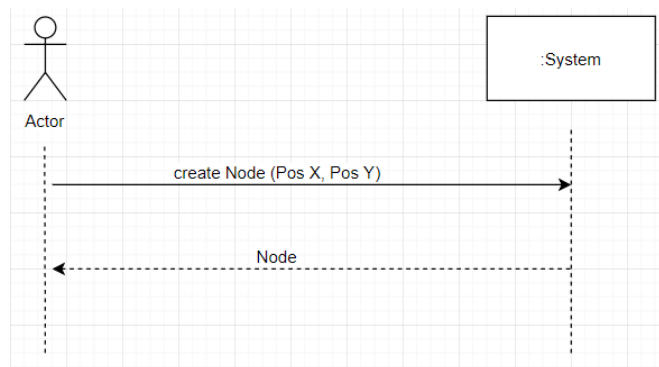


Abbildung 3.1.: Create Node SSD

Der Benutzer übergibt dem System eine Position in welcher er den Node erstellen möchte, das System erstellt dann den Node mit allen seinen Komponenten, also der Form, dem Text und den Anchors für das Verbinden.

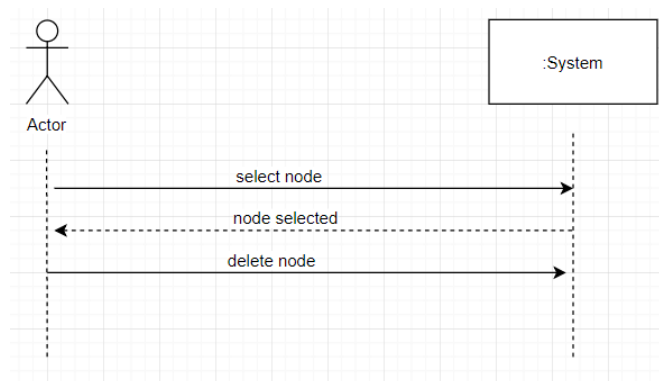


Abbildung 3.2.: Delete Node SSD

Der Benutzer wählt einen Node oder eine Verbindung (der Ablauf zum Löschen eines Nodes oder einer Verbindung ist gleich), das System löscht diesen Node dann vom GUI und von der Liste.

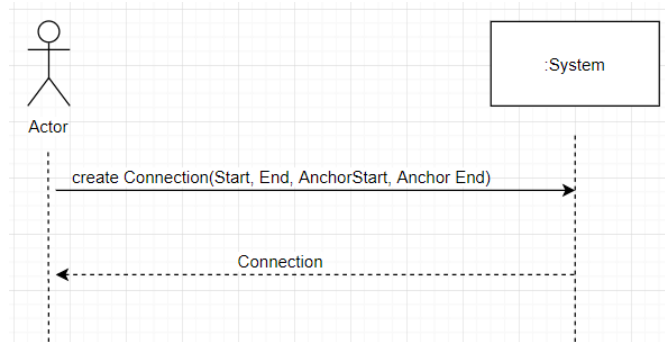


Abbildung 3.3.: Create Connection SSD

Der Benutzer wählt einen Start und einen Ziel Anchor für seine Verbindung aus, das System erstellt dann anhand dieser Informationen die Verbindung.

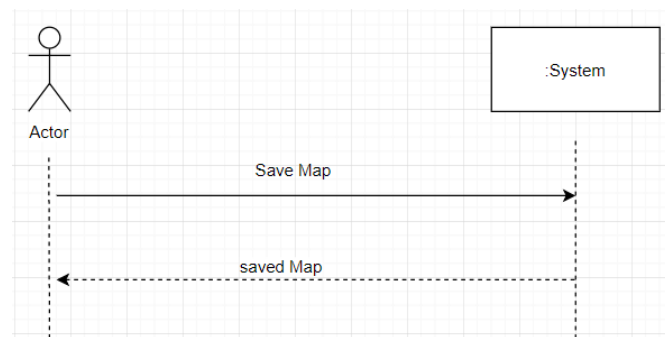


Abbildung 3.4.: Save Map SSD

Der Benutzer speichert die Map, das System speichert nun alle Informationen der Map in eine Datei und übergibt sie dem Benutzer.

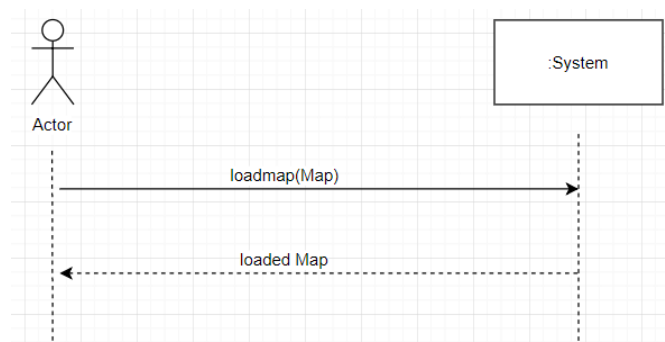


Abbildung 3.5.: Load Map SSD

Der Benutzer lädt eine alte Map und übergibt dem System eine Datei mit den Informationen, das System arbeitet nun die Informationen ab und stellt die Map dar.

3.2. Sequence Diagram

Das Sequence Diagramm zeigt die Interaktion zwischen den verschiedenen Objekten in dem Mind Map System.

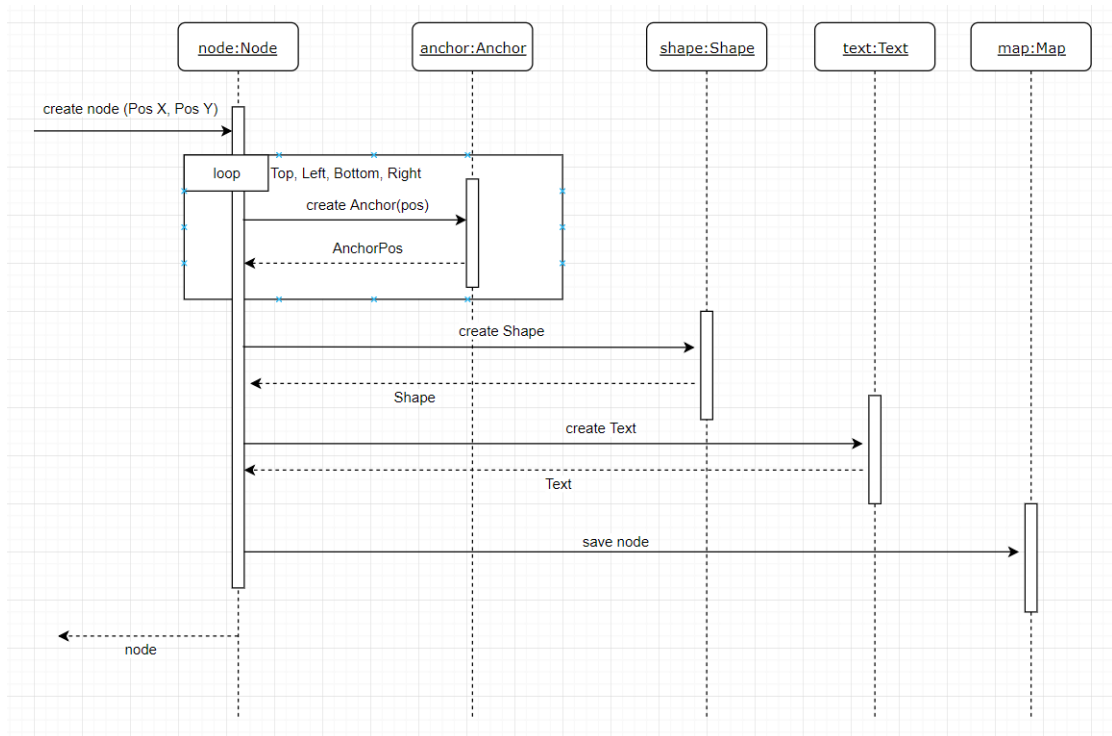


Abbildung 3.6.: Create Node SD

Das System erstellt einen neuen Node, der Node erstellt dann seine Objekte welche er benötigt. Er erstellt ein Shape für seine Form, er erstellt einen Text für die Beschreibung und er erstellt 4 Anchors(TOP,LEFT,BOTTOM,RIGHT) für das Verbinden der Knoten. Danach wird der Knoten in die Maplist gespeichert.

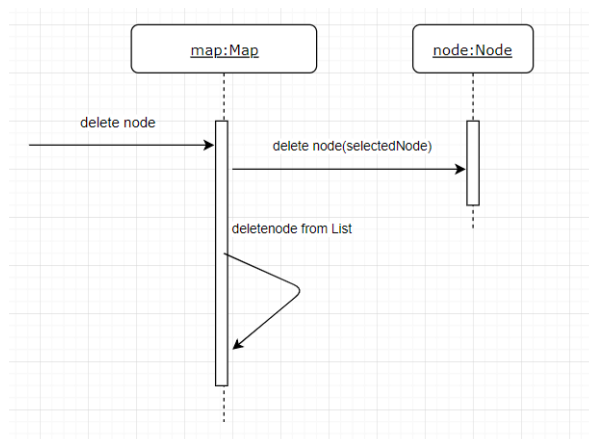


Abbildung 3.7.: Delete Node SD

Das System löscht den selektieren Knoten von der Map und löscht diesen dann von der Liste. Dies funktioniert gleich bei der Verbindung.

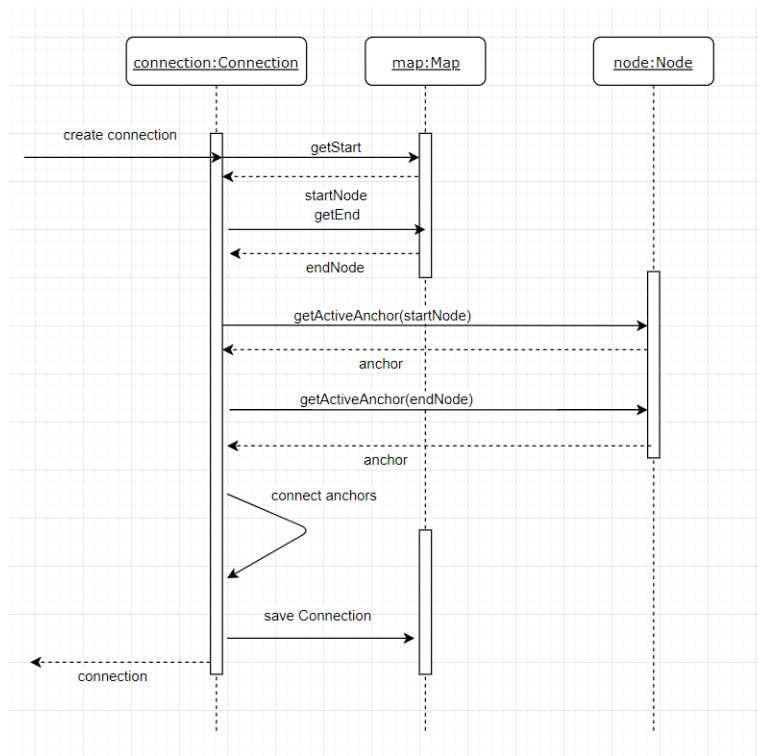


Abbildung 3.8.: Create Connection SD

Das System erstellt eine neue Verbindung, es holt sich zuerst die beiden aktiven Knoten, den Start und den End Knoten. Von diesen beiden Knoten holt es sich dann die jeweiligen aktiven Anchors. Danach verbindet das System die beiden aktiven Anchors miteinander.

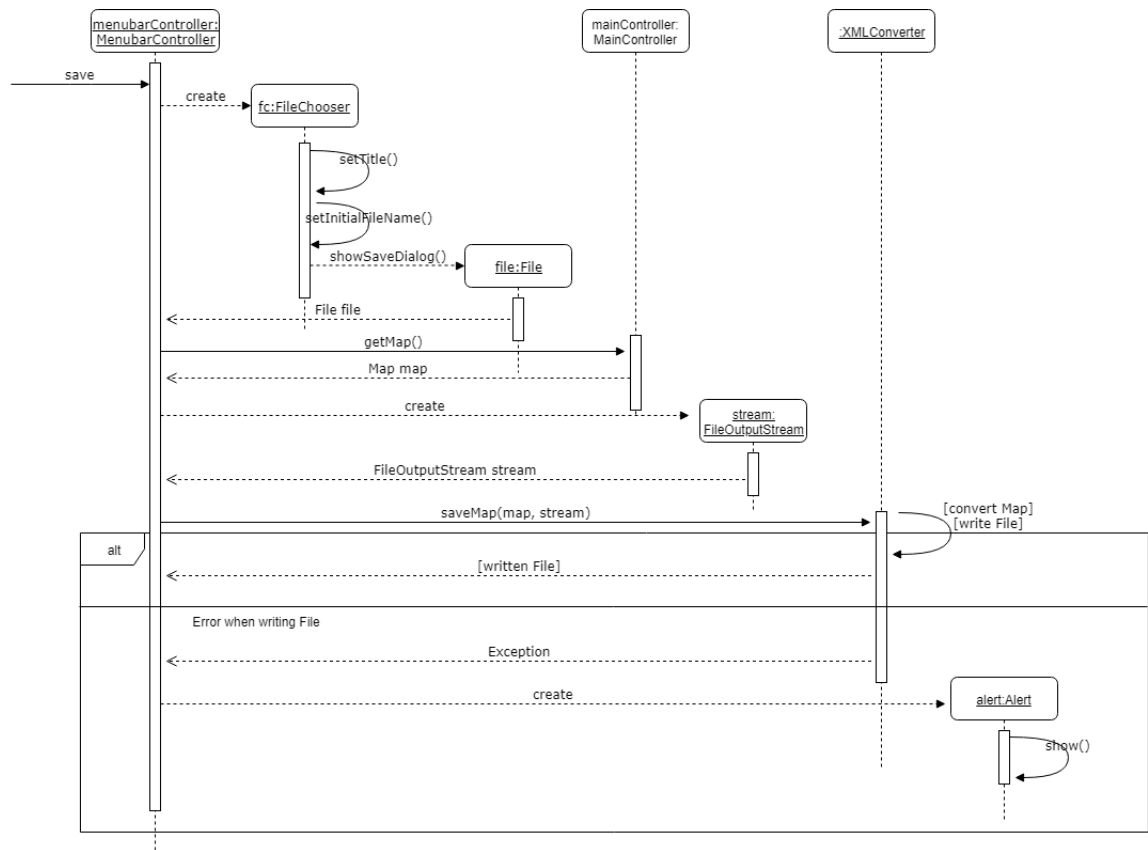


Abbildung 3.9.: Save Map SD

Das System lässt den Benutzer eine Datei wählen und erzeugt einen OutputStream. Die Map des MainControllers und der Stream werden dem XMLConverter übergeben. Dieser konvertiert die Map in ein speicherbares Objekt und schreibt die XML Datei. Die Funktionalität des XMLConverters ist im nächsten Sequence Diagram ersichtlich.

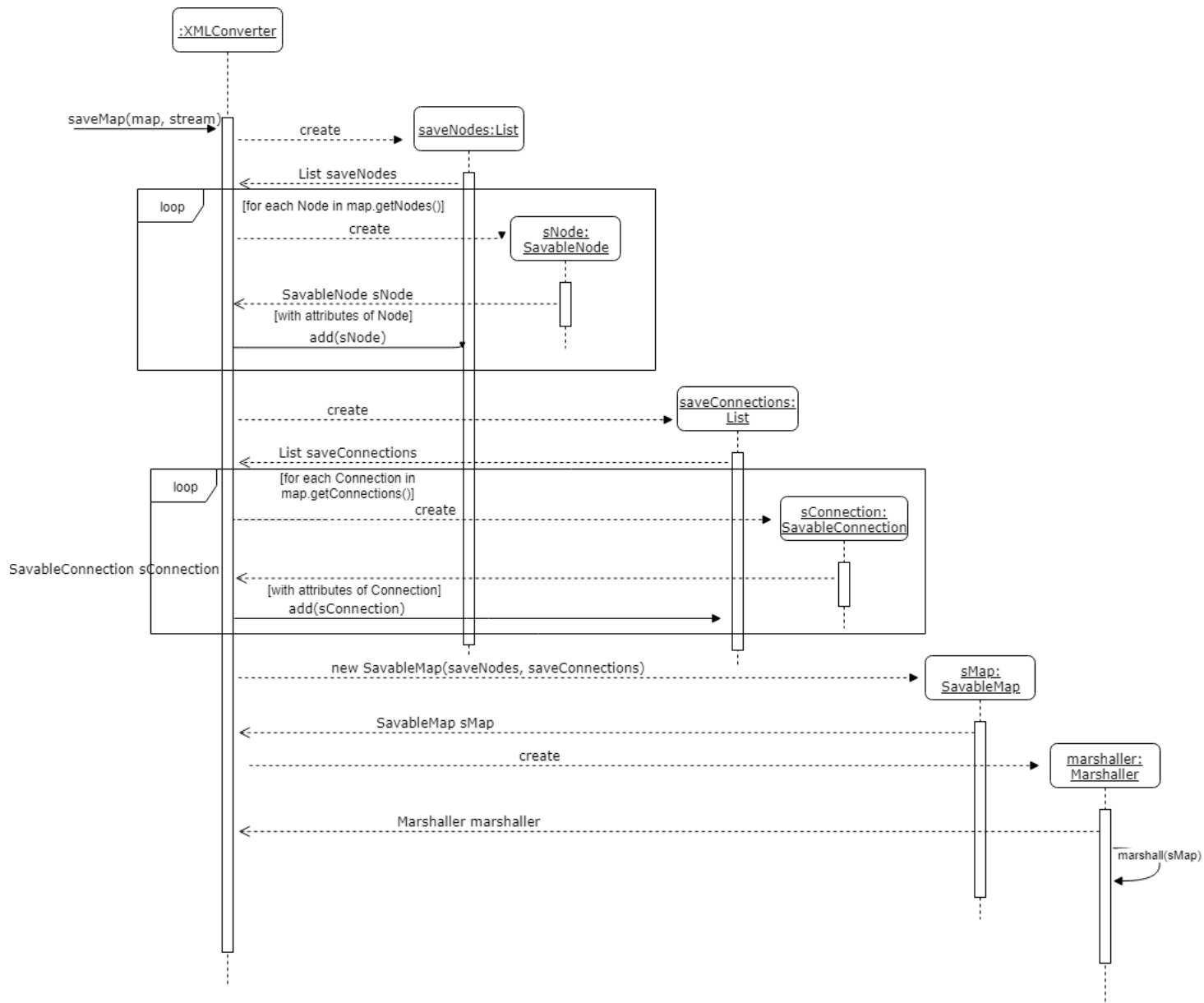


Abbildung 3.10.: Convert & Save Map SD

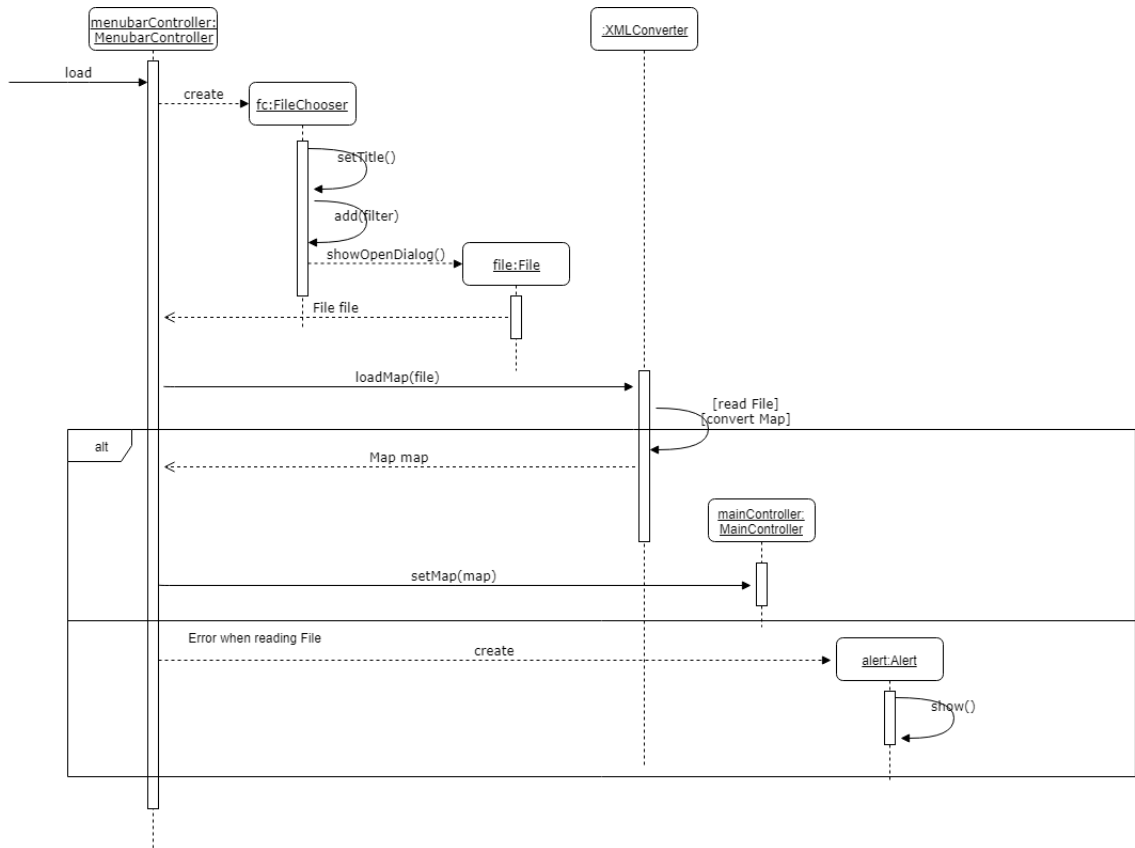


Abbildung 3.11.: Load Map SD

Das System lässt den Benutzer eine Datei wählen. Der XMLConverter liest die Datei und erzeugt daraus eine Map. Diese wird danach angezeigt.

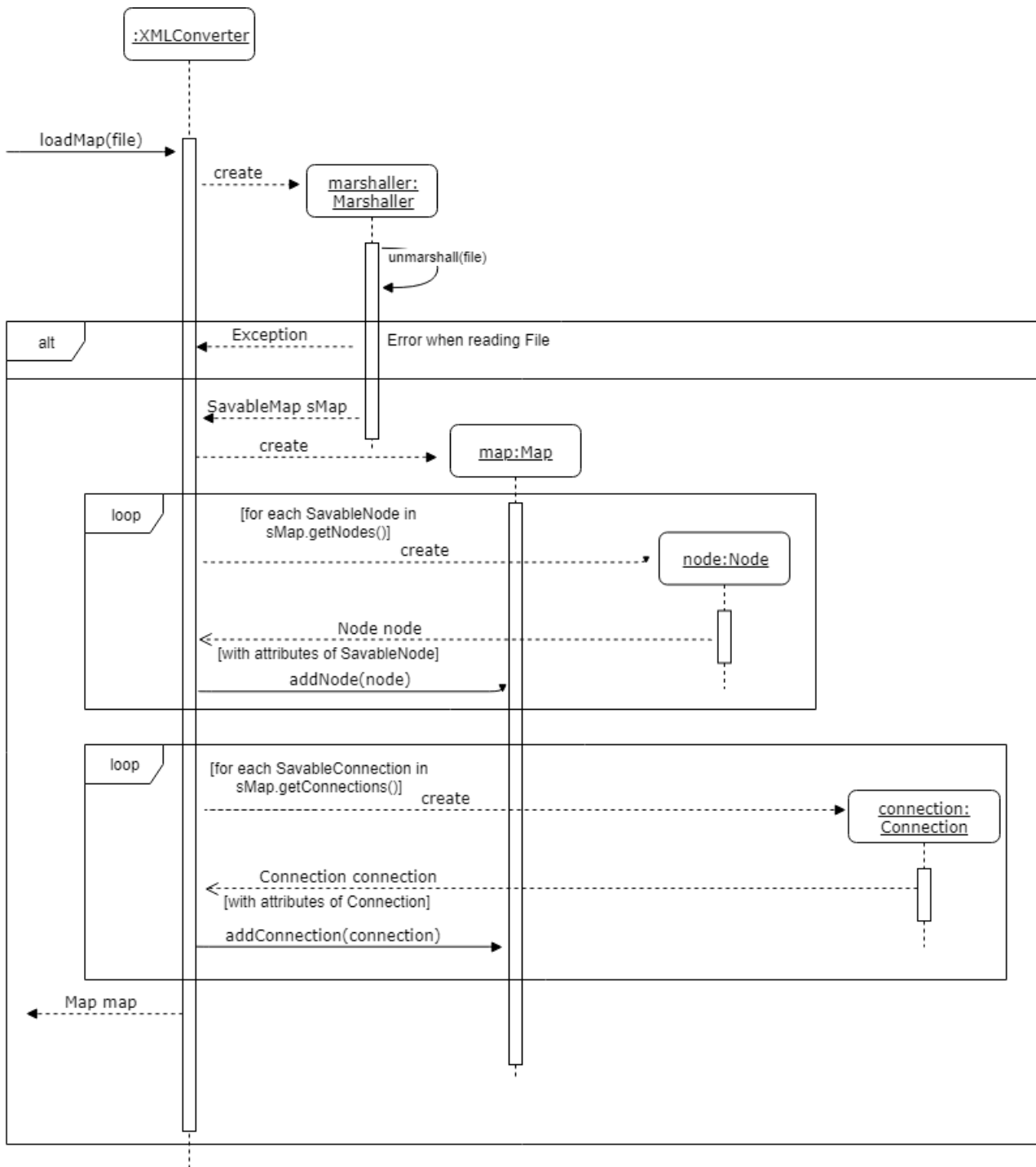


Abbildung 3.12.: Load & Convert Map SD

3.3. UML Diagramme

4. Entwicklung

4.1. Sprint 1

Sprint 1 dauert vom 13.03 - 03.04

4.2. Sprint 2

Sprint 2 dauert vom 03.4 - 24.04

4.3. Sprint 3

Sprint 3 dauert vom 24.04 - 15.05

4.4. Sprint 4

Sprint 4 dauert vom 15.05 - 29.05 (Abgabe Code Teil)

5. Fazit

Abbildungsverzeichnis

2.1. Domainmodel	3
2.2. Use Case Diagramm	4
2.3. Systemkontext	6
3.1. Create Node SSD	7
3.2. Delete Node SSD	7
3.3. Create Connection SSD	8
3.4. Save Map SSD	8
3.5. Load Map SSD	8
3.6. Create Node SD	9
3.7. Delete Node SD	9
3.8. Create Connection SD	10
3.9. Save Map SD	11
3.10. Convert & Save Map SD	12
3.11. Load Map SD	13
3.12. Load & Convert Map SD	14

A. Use Cases

Use Case 1: Knoten erstellen

- **Titel:** Als Benutzer möchte ich einen Knoten erstellen können.
- **Voraussetzung:** Programm ist gestartet
- **Erfolgsszenario:**
 1. Der Benutzer gibt den Befehl einen neuen Knoten zu erstellen.
 2. Das Programm fügt den neuen Knoten dem Mindmap hinzu.
- **Nachbedingung:** Ein neuer Knoten wurde dem Mindmap hinzugefügt.
- **Alternative Szenarien:**
 - 1.a 1 Der Benutzer schreibt eine Beschreibung für den Knoten.
 - 1.b 1 Der Benutzer ändert die Farbe des Knoten.

Use Case 2: Knoten verbinden

- **Titel:** Als Benutzer möchte ich zwei Knoten miteinander verbinden können.
- **Voraussetzung:** Ein Mindmap mit mindestens zwei Knoten wurde erstellt oder geladen.
- **Erfolgsszenario:**
 1. Der Benutzer wählt zwei Knoten aus.
 2. Das Programm verbindet die ausgewählten Knoten.
- **Nachbedingung:** Zwischen den ausgewählten Knoten besteht eine Verbindung.
- **Alternative Szenarien:**
 - 1.a 1 Der Benutzer ändert die Art der Verbindung.

Use Case 3: Mindmap speichern

- **Titel:** Als Benutzer möchte ich ein Mindmap speichern können.
- **Voraussetzung:** Ein Mindmap wurde erstellt.
- **Erfolgsszenario:**
 1. Der Benutzer wählt einen Speicherort.
 2. Das Programm übergibt dem Filesystem eine Datei.
- **Nachbedingung:** Eine Datei mit den Informationen des Mindmaps wurde am gewählten Ort im Filesystem gespeichert.
- **Alternative Szenarien:**

- 2.a 1 Das Filesystem verweigert das Speichern wegen zu wenig Speicherplatz.
- 2.a 2 Das Programm informiert den Benutzer, dass nicht genug Speicherplatz vorhanden ist.

- 2.b 1 Das Filesystem verweigert den Zugriff auf den Speicherort.
- 2.b 2 Das Programm informiert den Benutzer, dass der Zugriff verweigert wurde.

Use Case 4: Mindmap laden

- **Titel:** Als Benutzer möchte ich ein gespeichertes Mindmap laden können.
- **Voraussetzung:** Eine Datei mit den Informationen eines Mindmap existiert im Filesystem.
- **Erfolgsszenario:**
 1. Der Benutzer wählt eine Datei aus.
 2. Das Programm lädt die Datei.
 3. Das Programm zeigt das geladene Mindmap an.
- **Nachbedingung:** Das Programm zeigt ein Mindmap mit den Informationen aus der geladenen Datei.
- **Alternative Szenarien:**
 - 2.a 1 Das Filesystem verweigert den Zugriff auf die Datei.
 - 2.a 2 Das Programm informiert den Benutzer, dass der Zugriff verweigert wurde.

 - 2.b 1 Das Programm lädt eine beschädigte Datei.
 - 2.b 2 Das Programm informiert den Benutzer, dass die Datei beschädigt ist und nicht geladen werden kann.

Use Case 5: Knoten verändern

- **Titel:** Als Benutzer möchte ich einen erstellten Knoten verändern können.
- **Voraussetzung:** Ein Mindmap mit mindestens einem Knoten wurde erstellt oder geladen.
- **Erfolgsszenario:**
 1. Der Benutzer wählt einen existierenden Knoten.
 2. Der Benutzer schreibt eine (neue) Beschreibung für den Knoten.
- **Nachbedingung:** Der gewählte Knoten wurde verändert.
- **Alternative Szenarien:**
 - 2.a 1 Der Benutzer wählt eine (neue) Farbe für den Knoten.

Use Case 6: Verbindung verändern

- **Titel:** Als Benutzer möchte ich verschiedene Verbindungstypen definieren können.
- **Voraussetzung:** Ein Mindmap mit mindestens zwei Knoten wurde erstellt oder geladen.
- **Erfolgsszenario:**
 1. Der Benutzer wählt eine Verbindung aus.
 2. Der Benutzer ändert die Art/die Darstellung der Verbindung.
- **Nachbedingung:** Die gewählte Verbindung wurde verändert.

Use Case 7: Mindmap drucken

- **Titel:** Als Benutzer möchte ich ein Mindmap drucken können.
- **Voraussetzung:** Ein Mindmap wurde erstellt oder geladen.
- **Erfolgsszenario:**
 1. Der Benutzer gibt den Befehl das Mindmap zu drucken.
 2. Das Programm sendet eine Datei an den Drucker.
- **Nachbedingung:** Das Mindmap wurde ausgedruckt.
- **Alternative Szenarien:**
 - 2.a 1 Das Programm informiert den Benutzer, dass kein Drucker angeschlossen ist.
 - 2.b 1 Das Programm informiert den Benutzer, dass die gesendete Datei vom Drucker zurückgewiesen wurde.

Use Case 8: Mindmap exportieren

- **Titel:** Als Benutzer möchte ich ein Mindmap exportieren können.
- **Voraussetzung:** Ein Mindmap wurde erstellt oder geladen.
- **Erfolgsszenario:**
 1. Der Benutzer wählt ein Dateiformat.
 2. Der Benutzer wählt einen Speicherort.
 3. Das Programm übergibt dem Filesystem eine Datei mit dem gewünschten Format.
- **Nachbedingung:**
- **Alternative Szenarien:**
 - 2.a 1 Das Filesystem verweigert das Speichern wegen zu wenig Speicherplatz.
 - 2.a 2 Das Programm informiert den Benutzer, dass nicht genug Speicherplatz vorhanden ist.
 - 2.b 1 Das Filesystem verweigert den Zugriff auf den Speicherort.
 - 2.b 2 Das Programm informiert den Benutzer, dass der Zugriff verweigert wurde.

Use Case 9: Darstellung optimieren

- **Titel:** Als Benutzer möchte ich die Darstellung des Mindmaps optimieren können.
- **Voraussetzung:** Ein Mindmap wurde erstellt oder geladen.
- **Erfolgsszenario:**
 1. Der Benutzer gibt den Befehl das Mindmap zu optimieren.
 2. Das Programm optimiert das Mindmap.
- **Nachbedingung:** Ein Mindmap mit möglichst wenig sich überschneidenden Verbindungen.