

## Über diese Vorlage

Diese L<sup>A</sup>T<sub>E</sub>X-Vorlage wurde von Stefan Macke<sup>1</sup> als Grundlage für die Projektdokumentationen der Auszubildenden zum Fachinformatiker mit Fachrichtung Anwendungsentwicklung bei der ALTE OLDENBURGER Krankenversicherung entwickelt. Nichtsdestotrotz dürfte sie ebenso für die anderen IT-Berufe<sup>2</sup> geeignet sein, da diese anhand der gleichen Verordnung bewertet werden.

Diese Vorlage enthält bereits eine Vorstrukturierung der möglichen Inhalte einer tatsächlichen Projektdokumentation, die auf Basis der Erfahrungen im Rahmen der Prüfertätigkeit des Autors erstellt und unter Zuhilfenahme von ROHRER UND SEDLACEK [2011] abgerundet wurden.

Sämtliche verwendeten Abbildungen, Tabellen und Listings stammen von GRASHORN [2010].

Download-Link für diese Vorlage: <http://fiae.link/LaTeXVorlageFIAE>

Auch verfügbar auf GitHub: <https://github.com/StefanMacke/latex-vorlage-fiae>

## Lizenz



Dieses Werk steht unter einer Creative Commons Namensnennung - Weitergabe unter gleichen Bedingungen 4.0 International Lizenz.<sup>3</sup>



**Namensnennung** Sie müssen den Namen des Autors/Rechteinhabers in der von ihm festgelegten Weise nennen.<sup>4</sup>

**Weitergabe unter gleichen Bedingungen** Wenn Sie das lizenzierte Werk bzw. den lizenzierten Inhalt bearbeiten oder in anderer Weise erkennbar als Grundlage für eigenes Schaffen verwenden, dürfen Sie die daraufhin neu entstandenen Werke bzw. Inhalte nur unter Verwendung von Lizenzbedingungen weitergeben, die mit denen dieses Lizenzvertrages identisch oder vergleichbar sind.

---

<sup>1</sup>Blog des Autors: <http://fachinformatiker-anwendungsentwicklung.net>, Twitter: @StefanMacke

<sup>2</sup>z. B. IT-Kaufleute, Fachinformatiker mit Fachrichtung Systemintegration usw.

<sup>3</sup><http://creativecommons.org/licenses/by-sa/4.0/>

<sup>4</sup>Die Namensnennung im L<sup>A</sup>T<sub>E</sub>X-Quelltext mit Link auf <http://fiae.link/LaTeXVorlageFIAE> reicht hierfür aus.

## Inhalt der Projektdokumentation

Grundsätzlich definiert die [REGIERUNG DER BUNDESREPUBLIK DEUTSCHLAND](#) [1997, S. 1746]<sup>5</sup> das Ziel der Projektdokumentation wie folgt:

„Durch die Projektarbeit und deren Dokumentation soll der Prüfling belegen, daß er Arbeitsabläufe und Teilaufgaben zielorientiert unter Beachtung wirtschaftlicher, technischer, organisatorischer und zeitlicher Vorgaben selbständig planen und kundengerecht umsetzen sowie Dokumentationen kundengerecht anfertigen, zusammenstellen und modifizieren kann.“

Und das [BUNDESMINISTERIUM FÜR BILDUNG UND FORSCHUNG](#) [2000, S. 36] ergänzt:

„Die Ausführung der Projektarbeit wird mit praxisbezogenen Unterlagen dokumentiert. Der Prüfungsausschuss bewertet die Projektarbeit anhand der Dokumentation. Dabei wird nicht das Ergebnis – z. B. ein lauffähiges Programm – herangezogen, sondern der Arbeitsprozess. Die Dokumentation ist keine wissenschaftliche Abhandlung, sondern eine handlungsorientierte Darstellung des Projektablaufs mit praxisbezogenen, d.h. betriebüblichen Unterlagen. Sie soll einen Umfang von maximal 10 bis 15 DIN A 4-Seiten nicht überschreiten. Soweit erforderlich können in einem Anhang z. B. den Zusammenhang erläuternde Darstellungen beigelegt werden.“

Außerdem werden dort die grundlegenden Inhalte der Projektdokumentation aufgelistet:

- Name und Ausbildungsberuf des Prüfungsteilnehmers
- Angabe des Ausbildungsbetriebes
- Thema der Projektarbeit
- Falls erforderlich, Beschreibung/Konkretisierung des Auftrages
- Umfassende Beschreibung der Prozessschritte und der erzielten Ergebnisse
- Gegebenenfalls Veränderungen zum Projektantrag mit Begründung
- Wenn für das Projekt erforderlich, ein Anhang mit praxisbezogenen Unterlagen und Dokumenten. Dieser Anhang sollte nicht aufgebläht werden. Die angehängten Dokumente und Unterlagen sind auf das absolute Minimum zu beschränken.

In den folgenden Kapiteln werden diese geforderten Inhalte und sinnvolle Ergänzungen nun meist stichwortartig und ggfs. mit Beispielen beschrieben. Nicht alle Kapitel müssen in jeder Dokumentation vorhanden sein. Handelt es sich bspw. um ein in sich geschlossenes Projekt, kann das Kapitel 1.5: Projektbegrenzung entfallen; arbeitet die Anwendung nur mit XML-Dateien, kann und muss keine Datenbank beschrieben werden usw.

---

<sup>5</sup>Dieses Dokument sowie alle weiteren hier genannten können unter <http://fiae.link/LaTeXVorlageFIAEQuellen> heruntergeladen werden.

## Formale Vorgaben

Die formalen Vorgaben zum Umfang und zur Gestaltung der Projektdokumentation können je nach IHK recht unterschiedlich sein. Normalerweise sollte die zuständige IHK einen Leitfaden bereitstellen, in dem alle Formalien nachgelesen werden können, wie z. B. bei der [IHK OLDENBURG \[2006\]](#).

Als Richtwert verwende ich 15 Seiten für den reinen Inhalt. Also in dieser Vorlage alle Seiten, die arabisch nummeriert sind (ohne das Literaturverzeichnis und die eidesstattliche Erklärung). Große Abbildungen, Quelltexte, Tabellen usw. gehören in den Anhang, der 25 Seiten nicht überschreiten sollte.

Typographische Konventionen, Seitenränder usw. können in der Datei `Seitenstil.tex` beliebig angepasst werden.

## Bewertungskriterien

Die Bewertungskriterien für die Benotung der Projektdokumentation sind recht einheitlich und können leicht in Erfahrung gebracht werden, z. B. bei der [IHK DARMSTADT \[2011\]](#). Grundsätzlich sollte die Projektdokumentation nach der Fertigstellung noch einmal im Hinblick auf diese Kriterien durchgeschaut werden.

# Erklärung

## Bestätigung über die durchgeführte betriebliche Aufgabe<sup>1</sup>

(Diese Bestätigung ist als Deckblatt online einzureichen, gemeinsam mit dem Report/der Dokumentation.)

Prüfling (vollständige Anschrift und Telefonnummer)	Ausbildungsbetrieb (vollständige Anschrift)
Vorname, Name	Firma
Straße, Hausnr.	Straße, Hausnr.
PLZ, Ort	PLZ, Ort
Tel.Nr.:	Tel.Nr.:

Hinweis vorab: Aus Gründen der besseren Lesbarkeit wird auf die gleichzeitige Verwendung männlicher und weiblicher Sprachformen verzichtet. Sämtliche Personenbezeichnungen gelten gleichermaßen für alle Geschlechter.

Ausbildungsberuf

Bezeichnung der betrieblichen Aufgabe

## Erklärung des Prüflings

Hiermit versichere ich, dass ich die betriebliche Aufgabe unter der Betreuung von

Verantwortlicher im Unternehmen

selbstständig durchgeführt und die Unterlagen selbstständig zusammengestellt habe.

Dokumente und Textpassagen, die ich nicht selbstständig erstellt habe, sind von mir gekennzeichnet.

Ort, Datum

Unterschrift des Prüflings

## Bestätigung des Ausbildungsbetriebes

Wir bestätigen, dass die Angaben des Prüflings richtig sind.

Ort, Datum

Unterschrift des Verantwortlichen, der die Aufgabe betreut hat.

Ort, Datum

Unterschrift des Ausbilders

<sup>1</sup>Zur Vereinfachung wird einheitlich der Begriff „betriebliche Aufgabe“ verwendet. Gemeint sind die Fachaufgabe/die Projektarbeit/der betrieblicher Auftrag. Die unterschiedlichen Bezeichnungen entstehen durch die verschiedenen Berufe, die eine Aufgabe online einstellen.



Abschlussprüfung Winter 2024

Fachinformatiker für Anwendungsentwicklung  
Dokumentation zur betrieblichen Projektarbeit

## **RESTfull-API-Framework**

**RESTfull-API-Microservice mit Lasttestmodul**

Abgabetermin: Stuttgart, den 03.12.2024

**Prüfungsbewerber:**

Alexander Knueppel  
Brucknerstraße 13  
73054 Eislingen



**Fraunhofer**  
**IPA**

**Ausbildungsbetrieb:**

FRAUNHOFER (IPA)  
Nobelstraße 12  
70569 Stuttgart

---

## Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>III</b>
<b>Tabellenverzeichnis</b>	<b>IV</b>
<b>Listings</b>	<b>V</b>
<b>Abkürzungsverzeichnis</b>	<b>VI</b>
<b>1 Einleitung</b>	<b>1</b>
1.1 Projektumfeld . . . . .	1
1.2 Projektziel . . . . .	2
1.3 Projektbegründung . . . . .	2
1.4 Projektschnittstellen . . . . .	2
1.5 Projektabgrenzung . . . . .	3
<b>2 Projektplanung</b>	<b>3</b>
2.1 Projektphasen . . . . .	3
2.2 Abweichungen vom Projektantrag . . . . .	4
2.3 Ressourcenplanung . . . . .	4
2.4 Entwicklungsprozess . . . . .	5
<b>3 Analysephase</b>	<b>5</b>
3.1 Ist-Analyse . . . . .	5
3.2 Wirtschaftlichkeitsanalyse . . . . .	5
3.2.1 „Make or Buy“-Entscheidung . . . . .	6
3.2.2 Projektkosten . . . . .	6
3.2.3 Amortisationsdauer . . . . .	6
3.3 Nutzwertanalyse . . . . .	7
3.4 Anwendungsfälle . . . . .	7
3.5 Qualitätsanforderungen . . . . .	7
3.6 Lastenheft/Fachkonzept . . . . .	8
<b>4 Entwurfsphase</b>	<b>8</b>
4.1 Zielplattform . . . . .	8
4.2 Architekturdesign . . . . .	8
4.3 Entwurf der Benutzeroberfläche . . . . .	9
4.4 Datenmodell . . . . .	9
4.5 Geschäftslogik . . . . .	9
4.6 Maßnahmen zur Qualitätssicherung . . . . .	10
4.7 Pflichtenheft/Datenverarbeitungskonzept . . . . .	10

---

<b>5</b>	<b>Implementierungsphase</b>	<b>10</b>
5.1	Implementierung der Datenstrukturen . . . . .	10
5.2	Implementierung der Benutzeroberfläche . . . . .	11
5.3	Implementierung der Geschäftslogik . . . . .	11
<b>6</b>	<b>Abnahmephase</b>	<b>11</b>
<b>7</b>	<b>Einführungsphase</b>	<b>12</b>
<b>8</b>	<b>Dokumentation</b>	<b>12</b>
<b>9</b>	<b>Fazit</b>	<b>12</b>
9.1	Soll-/Ist-Vergleich . . . . .	12
9.2	Lessons Learned . . . . .	13
9.3	Ausblick . . . . .	13
	<b>Literaturverzeichnis</b>	<b>14</b>
	<b>Eidesstattliche Erklärung</b>	<b>15</b>
<b>A</b>	<b>Anhang</b>	<b>i</b>
A.1	Detaillierte Zeitplanung . . . . .	i
A.2	Lastenheft (Auszug) . . . . .	ii
A.3	Use Case-Diagramm . . . . .	iii
A.4	Pflichtenheft (Auszug) . . . . .	iii
A.5	Datenbankmodell . . . . .	v
A.6	Oberflächenentwürfe . . . . .	vi
A.7	Screenshots der Anwendung . . . . .	viii
A.8	Entwicklerdokumentation . . . . .	x
A.9	Testfall und sein Aufruf auf der Konsole . . . . .	xii
A.10	Klasse: ComparedNaturalModuleInformation . . . . .	xiii
A.11	Klassendiagramm . . . . .	xvi
A.12	Benutzerdokumentation . . . . .	xvii

## Abbildungsverzeichnis

1	Vereinfachtes ER-Modell . . . . .	9
2	Prozess des Einlesens eines Moduls . . . . .	10
3	Use Case-Diagramm . . . . .	iii
4	Datenbankmodell . . . . .	v
5	Liste der Module mit Filtermöglichkeiten . . . . .	vi
6	Anzeige der Übersichtsseite einzelner Module . . . . .	vii
7	Anzeige und Filterung der Module nach Tags . . . . .	vii
8	Anzeige und Filterung der Module nach Tags . . . . .	viii
9	Liste der Module mit Filtermöglichkeiten . . . . .	ix
10	Aufruf des Testfalls auf der Konsole . . . . .	xiii
11	Klassendiagramm . . . . .	xvi



## Tabellenverzeichnis

1	Kostenaufstellung . . . . .	6
2	Entscheidungsmatrix . . . . .	8
3	Soll-/Ist-Vergleich . . . . .	13

## Listings

1	Testfall in PHP . . . . .	xii
2	Klasse: ComparedNaturalModuleInformation . . . . .	xiii

## Abkürzungsverzeichnis

<b>API</b>	Application Programming Interface
<b>CSV</b>	Comma Separated Value
<b>EPK</b>	Ereignisgesteuerte Prozesskette
<b>ERM</b>	Entity-Relationship-Modell
<b>HTML</b>	Hypertext Markup Language
<b>MVC</b>	Model View Controller
<b>NatInfo</b>	Natural Information System
<b>Natural</b>	Programmiersprache der Software AG
<b>PHP</b>	Hypertext Preprocessor
<b>SQL</b>	Structured Query Language
<b>SVN</b>	Subversion
<b>UML</b>	Unified Modeling Language
<b>XML</b>	Extensible Markup Language

## 1 Einleitung

### 1.1 Projektumfeld

#### **Zum Praktikumsbetrieb:**

Das Fraunhofer Institut für Produktionstechnik und Automatisierung(IPA) - eines der größten Institute der Fraunhofer-Gesellschaft - wurde 1959 gegründet und beschäftigt rund 1200 Mitarbeiterinnen und Mitarbeiter. Organisatorische und technologische Aufgabenstellungen aus der Produktion bilden die Forschungs- und Entwicklungsschwerpunkte. Verfahren, Komponenten und Geräte bis hin zu kompletten Maschinen und Anlagen werden vom Institut entwickelt, erprobt und beispielhaft eingesetzt.

#### **Zum Forschungsprojekt:**

Das Forschungsprojekt »H2GO – Nationaler Aktionsplan Brennstoffzellen-Produktion« des Fraunhofer Instituts hat das Ziel, die Produktion von Brennstoffzellen in Deutschland zu revolutionieren und damit einen wesentlichen Beitrag zur Reduzierung von CO<sub>2</sub>-Emissionen im Schwerlastverkehr zu leisten.

(Pressemitteilung H2Go: [www.iwu.fraunhofer.de](http://www.iwu.fraunhofer.de))

H2GO bündelt die Aktivitäten von 19 Fraunhofer-Instituten, um Technologien zu entwickeln, die eine kosteneffiziente und großflächige Produktion von Brennstoffzellen ermöglichen.

Das Projekt lässt sich in zwei Teilbereiche aufteilen:

- Produktion: auf vier Standorte verteilt, forscht an Umsetzungen und fertigen Komponenten für diese.
- Digitalisierung/Virtualisierung: Erstellt die digitalen und virtuellen Anteile der Referenzfabrik, koordiniert die Netzwerkstrukturen.

#### **Zum Teilbereich in welchem das Projekt realisiert wurde:**

Im Teilbereich der virtuellen Referenzfabrik wird die Digitalisierung und virtuellen Abbildung einer automatisierten Brennstoffzellenproduktion durch digitale Zwillinge nach [Industrie-4.0](#)-Standards umgesetzt.

Realisiert werden die digitalen Zwillinge dabei als Verwaltungsschalen (= AAS = Asset Administration Shell).

Aufgabe des Teilbereichs ist, eine möglichst exakte virtuelle Abbildung des gesamten Prozesses, des Produktes, der Produktionsdaten und -mittel, sowie Teilprodukte erzeugt werden.

#### **Auftraggeber:**

Auftragsvergabe für das REST-Frameworks erfolgte aufgrund meiner Überlegungen und Anregungen zur Ausfallsicherheit im Teilbereichs-Meeting und wurde durch Praktikums-Betreuer Dipl.-Ing Bumin Hatiboglu, wodurch dies ein internes Projekt des IPA und der betreffenden Abteilung ist.

## 1.2 Projektziel

- **Was ist das Problem?**

in der derzeitigen Planung und Umsetzung der Netzwerkarchitektur sollen die verschiedenen Daten des digitalen Zwillings von einem zentralen Registry-Server verwaltet werden, welcher die Speicherorte, Beziehungen und Verknüpfungen aller im Projekt vorhandenen Shells referenziert, und letztlich allen qualifizierten Clients ( Client = Stakeholder) bereit stellt.

Dieser eine Registry-Server weist jedoch bisher keine wirkliche Ausfallsicherheit bei einem Netzerkausfall auf, was auf Basis einer Risikoanalyse besonders kritisch für die Ausfallsicherheit des Projektnetzwerk ist und bei Ausfall die Produktion letztlich zum Erliegen bringen würde.

- **Ziel des Projektes ist deshalb:**

die Engstelle eines einzelnen Registryservers durch eine Redundantes System zu erweitern, was als Backupstrategie in der Regel auch dem tatsächlich üblichen Industriestandard zur Risikominimierung entspricht.

## 1.3 Projektbegründung

- **Möglicher Benefit:**

Abgesehen von dem offensichtlichen Vorteil einer Ausfallsicherheit bereits im reinen Forschungsbetrieb, ist es durch das Stadium der Umsetzung des H2Go-Projektes auch ein weitere, bereits sehr frühe implementierbare Verbesserung, und kann dadurch helfen Standards für spätere Umsetzungen zu etablieren. Da in der Umsetzung nur kostenlose Open-Source Tools eingesetzt wurden, sorgt das Framework zudem für eine gleichzeitige Transparenz und Kostenersparnis.

- **Motivation:**

Motivation zur Erstellung des Projekts war, von Seiten des Durchführenden, in seiner Projektarbeit einerseits eine gut umsetzbare Lösung zu präsentieren. Andererseits war eine Motiv dahinter die Thematik einer robusten Infrastruktur bereits auf dieser Ebene zu etablieren um dazu beizutragen unserer sich ständig wachsenden digitalen Welt stabile Systeme zugrunde zu legen.

## 1.4 Projektschnittstellen

- Die Anwendung soll zukünftig mit einer weiteren REST-API aus dem AAS-Registry-Server des [Eclipse-BaSyx-Projektes](#) verbunden werden, jedoch ist dies noch nicht implementiert worden, da diese Strukturen gerade im Aufbau sind. Als weiters System mit welchem die Anwendung aktuell bereits verbunden und getestet wurde, darf man wohl den als Teil

des Projekts erstellten Locust-Lasttest verstehen. Dieser hilft, Tests der Auslastung des Systems durchzuführen.

- Da dieses Projekt Teil eines bereits genehmigten Forschungsprojektes ist, wird es durch das Förderprogramm "go-digital" finanziert, das vom Bundesministerium für Wirtschaft und Klimaschutz (BMWK) ins Leben gerufen wurde.
- im weiteren Verlauf des Projektes muss das Framework von Anwendungsentwicklern der Fraunhofer weiterentwickelt und angepasst werden und von diesen, oder den Verantwortlichen der jeweiligen am H2Go-Projekt teilnehmenden Instituten installiert und gewartet werden. Jedoch besteht hier weder eine konkrete Umsetzung noch eine konkrete Planung.
- Das Projekt wurde den verantwortlichen Betreuern vorgelegt und wurde vom Softwareverantwortlichen Sven Lieckfeld bereits einer Endabnahme unterzogen. Zum aktuellen Zeitpunkt konnte die Endabnahme durch den Projektverantwortlichen Bumin Hatiboglu aus gesundheitlichen Gründen noch nicht durchgeführt werden.

## 1.5 Projektabgrenzung

- Das Projekt soll explizit nicht als fertig entwickelte und einsatzfähige synchrone Backuplösung für den Registry-Server verstanden werden, da keine derartigen Funktionalitäten implementiert oder integriert wurden. Das Projekt stellt lediglich ein Framework zur Verfügung, auf welchem eine solche Lösung implementiert werden kann. Es sollte somit eher als Werkzeug und spezialisierte, aber nicht spezifizierte Schnittstelle verstanden werden.

## 2 Projektplanung

### 2.1 Projektphasen

- Das Projekt wurde im Zeitraum vom **15.11.2024 - 29.11.2024** unter Einsatz einer täglichen Arbeitszeit von **8 Stunden** erstellt.
- Die im Projektantrag angefertigte Zeitplanung muss an mehreren Stellen angepasst werden aufgrund von Gründen welche an gegebener Stelle näher erläutert werden. **!!!!!!anpassung einfügen. gerne auch Bild einer Tabelle!!!!!!**
- Verfeinerung der Zeitplanung, die bereits im Projektantrag vorgestellt wurde.

## 2.2 Abweichungen vom Projektantrag

- **UML-Diagramme:** Bei den im Antrag angegebenen UML-Diagrammen wurde eine Satz an Diagrammen angekündigt, welcher der tatsächlichen Umsetzung nur bedingt genützt hätte. Daher wurden folgende Diagramme ausgetauscht:
  - **Klassendiagramm wurde durch Komponentendiagramm ersetzt:** Da Python zwar klassenorientiert konzipiert wurde, das Projekt jedoch keine Klassenobjekte benötigte, wurde ein ausführliches Komponentendiagramm erstellt. Dieses Diagramm wurde als geeigneter angesehen, um die recht komplexen Abläufe und Verbindungen der Komponenten untereinander zu veranschaulichen.
  - **Aktivitätsdiagramm wurde zu drei Diagrammen ausgeweitet:** Aufgrund einer modularen Abtrennung der Komponenten untereinander wurde hier entschieden, das ursprüngliche Diagramm um zwei weitere zu ergänzen.

Die Gründe hierfür werden in der UML-Dokumentation näher erläutert.

- **Architektur der Anwendung:** Ursprünglich wurden im Projekt drei Module geplant:
  - main.py
  - com-server.py
  - test.db

In der Umsetzung zeigte sich, dass die main.py, welche als der Startpunkt aller Komponenten definiert wurde, unnötig war. Da bei der Erstellung des für den Flaskserver benötigten Docker-Images ein Befehl zum ausführen der comserver.py-Datei integriert wurde und sonstige Startprozesse wie die Verknüpfung von Server mit Datenbank und das etablieren von Ports, Verbindungen und bereitstellen von API-Endpunkten vollständig zwischen Docker-Compose, Der Dockerfile und der REST-API realisiert wurde, hätte das Erstellen einer externen, lokalen App eine Redundanz und unnötigen Ressourceneinsatz bedeutet.

- **Testphasen wurden entsprechend an dem veränderten Umsetzung angepasst.**

## 2.3 Ressourcenplanung

### Hardware/Software:

- **Laptop:** Dell Inc. Latitude E6540 (OS: Fedora 41, Linux-Distribution)
  - Visual Studio Code
  - Docker
    - \* Docker-Compose

\* PostgreSQL (offizielles Image)

- **Python-Versionen:** Python 3.12

Zweiter Laptop mit Windows 10

- **Laptop:** Lenovo ThinkPad (OS: Windows 10, Enterprise Edition)
  - Python-Version: Python 3.x
  - Python-Pakete: locust==2.32.3

Hardware-Zubehör:

- Monitore
- Tastatur
- Maus

Sonstiges:

- Ausführender Entwicklung, Planung, Umsetzung, Dokumentation: 80h
- Betreuer: 5 h
- Büroplatz: Tisch, Stuhl, Steckdosen
- sonstige Kosten(schwer zu ermitteln): Strom, Wasser, Heizung, etc.

## 2.4 Entwicklungsprozess

- Welcher Entwicklungsprozess wird bei der Bearbeitung des Projekts verfolgt (z. B. Wasserfall, agiler Prozess)?

# 3 Analysephase

## 3.1 Ist-Analyse

- Wie ist die bisherige Situation (z. B. bestehende Programme, Wünsche der Mitarbeiter)?
- Was gilt es zu erstellen/verbessern?

## 3.2 Wirtschaftlichkeitsanalyse

- Lohnt sich das Projekt für das Unternehmen?



### 3.2.1 „Make or Buy“-Entscheidung

- Gibt es vielleicht schon ein fertiges Produkt, dass alle Anforderungen des Projekts abdeckt?
- Wenn ja, wieso wird das Projekt trotzdem umgesetzt?

### 3.2.2 Projektkosten

- Welche Kosten fallen bei der Umsetzung des Projekts im Detail an (z. B. Entwicklung, Einführung/Schulung, Wartung)?

**Beispielrechnung (verkürzt)** Die Kosten für die Durchführung des Projekts setzen sich sowohl aus Personal-, als auch aus Ressourcenkosten zusammen. In Absprache mit der Personalabteilung wird für einen Auszubildenden im dritten Lehrjahr ein Stundensatz von 7,56 € und für die anderen Mitarbeiter ein Stundensatz von 25 € angesetzt. Für die Nutzung von Ressourcen<sup>6</sup> wird ein Stundensatz von 15 € angenommen. Die Durchführungszeit des Projekts beträgt 70 Stunden. Eine Aufstellung der Kosten befindet sich in Tabelle 1 und sie betragen insgesamt 2739,20 €.

Vorgang	Zeit	Kosten pro Stunde	Kosten
Entwicklungskosten	70 h	7,56 € + 15 € = 22,56 €	1579,20 €
Fachgespräch	3 h	25 € + 15 € = 40 €	120 €
Abnahmetest	1 h	25 € + 15 € = 40 €	40 €
Anwenderschulung	25 h	25 € + 15 € = 40 €	1000 €
			<b>2739,20 €</b>

Tabelle 1: Kostenaufstellung

### 3.2.3 Amortisationsdauer

- Welche monetären Vorteile bietet das Projekt (z. B. Einsparung von Lizenzkosten, Arbeitszeiterparnis, bessere Usability, Korrektheit)?
- Wann hat sich das Projekt amortisiert?

---

<sup>6</sup>Räumlichkeiten, Arbeitsplatzrechner etc.

**Beispielrechnung (verkürzt)** Bei einer Zeiteinsparung von 10 Minuten am Tag für jeden der 25 Anwender und 220 Arbeitstagen im Jahr ergibt sich eine gesamte Zeiteinsparung von

$$25 \cdot 220 \text{ Tage/Jahr} \cdot 10 \text{ min/Tag} = 55000 \text{ min/Jahr} \approx 917 \text{ h/Jahr} \quad (1)$$

Dadurch ergibt sich eine jährliche Einsparung von

$$917 \text{ h} \cdot (25 + 15) \text{ €/h} = 36680 \text{ €} \quad (2)$$

Die Amortisationszeit beträgt also  $\frac{2739,20 \text{ €}}{36680 \text{ €/Jahr}} \approx 0,07 \text{ Jahre} \approx 4 \text{ Wochen}$ .

### 3.3 Nutzwertanalyse

- Darstellung des nicht-monetären Nutzens (z. B. Vorher-/Nachher-Vergleich anhand eines Wirtschaftlichkeitskoeffizienten).

**Beispiel** Ein Beispiel für eine Entscheidungsmatrix findet sich in Kapitel 4.2: Architekturdesign.

### 3.4 Anwendungsfälle

- Welche Anwendungsfälle soll das Projekt abdecken?
- Einer oder mehrere interessante (!) Anwendungsfälle könnten exemplarisch durch ein Aktivitätsdiagramm oder eine Ereignisgesteuerte Prozesskette (EPK) detailliert beschrieben werden.

**Beispiel** Ein Beispiel für ein Use Case-Diagramm findet sich im Anhang A.3: Use Case-Diagramm auf Seite iii.

### 3.5 Qualitätsanforderungen

- Welche Qualitätsanforderungen werden an die Anwendung gestellt (z. B. hinsichtlich Performance, Usability, Effizienz etc. (siehe [ISO/IEC 9126-1 \[2001\]](#)))?

### 3.6 Lastenheft/Fachkonzept

- Auszüge aus dem Lastenheft/Fachkonzept, wenn es im Rahmen des Projekts erstellt wurde.
- Mögliche Inhalte: Funktionen des Programms (Muss/Soll/Wunsch), User Stories, Benutzerrollen

**Beispiel** Ein Beispiel für ein Lastenheft findet sich im Anhang A.2: Lastenheft (Auszug) auf Seite ii.

## 4 Entwurfsphase

### 4.1 Zielplattform

- Beschreibung der Kriterien zur Auswahl der Zielplattform (u. a. Programmiersprache, Datenbank, Client/Server, Hardware).

### 4.2 Architekturdesign

- Beschreibung und Begründung der gewählten Anwendungsarchitektur (z. B. MVC).
- Ggfs. Bewertung und Auswahl von verwendeten Frameworks sowie ggfs. eine kurze Einführung in die Funktionsweise des verwendeten Frameworks.

**Beispiel** Anhand der Entscheidungsmatrix in Tabelle 2 wurde für die Implementierung der Anwendung das PHP-Framework Symfony<sup>7</sup> ausgewählt.

Eigenschaft	Gewichtung	Akelos	CakePHP	Symfony	Eigenentwicklung
Dokumentation	5	4	3	5	0
Reenginierung	3	4	2	5	3
Generierung	3	5	5	5	2
Testfälle	2	3	2	3	3
Standardaufgaben	4	3	3	3	0
<b>Gesamt:</b>	<b>17</b>	<b>65</b>	<b>52</b>	<b>73</b>	<b>21</b>
<b>Nutzwert:</b>		<b>3,82</b>	<b>3,06</b>	<b>4,29</b>	<b>1,24</b>

Tabelle 2: Entscheidungsmatrix

<sup>7</sup>Vgl. SENSIO LABS [2010].

### 4.3 Entwurf der Benutzeroberfläche

- Entscheidung für die gewählte Benutzeroberfläche (z. B. GUI, Webinterface).
- Beschreibung des visuellen Entwurfs der konkreten Oberfläche (z. B. Mockups, Menüführung).
- Ggfs. Erläuterung von angewendeten Richtlinien zur Usability und Verweis auf Corporate Design.

**Beispiel** Beispielentwürfe finden sich im Anhang A.6: Oberflächenentwürfe auf Seite vi.

### 4.4 Datenmodell

- Entwurf/Beschreibung der Datenstrukturen (z. B. ERM und/oder Tabellenmodell, XML-Schemas) mit kurzer Beschreibung der wichtigsten (!) verwendeten Entitäten.

**Beispiel** In Abbildung 1 wird ein Entity-Relationship-Modell (ERM) dargestellt, welches lediglich Entitäten, Relationen und die dazugehörigen Kardinalitäten enthält.

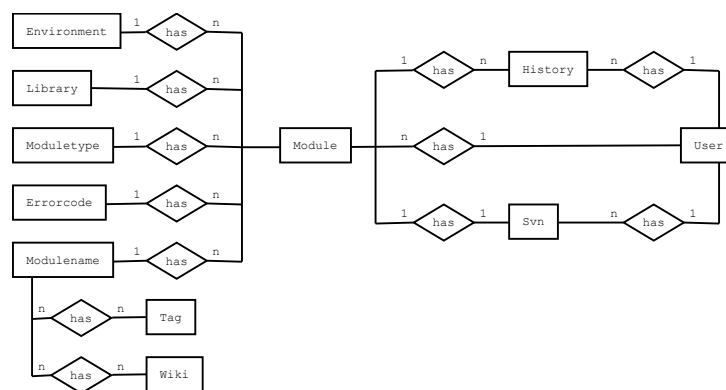


Abbildung 1: Vereinfachtes ER-Modell

### 4.5 Geschäftslogik

- Modellierung und Beschreibung der wichtigsten (!) Bereiche der Geschäftslogik (z. B. mit Komponenten-, Klassen-, Sequenz-, Datenflussdiagramm, Programmablaufplan, Struktogramm, EPK).
- Wie wird die erstellte Anwendung in den Arbeitsfluss des Unternehmens integriert?

**Beispiel** Ein Klassendiagramm, welches die Klassen der Anwendung und deren Beziehungen untereinander darstellt kann im Anhang A.11: Klassendiagramm auf Seite xvi eingesehen werden.

Abbildung 2 zeigt den grundsätzlichen Programmablauf beim Einlesen eines Moduls als EPK.

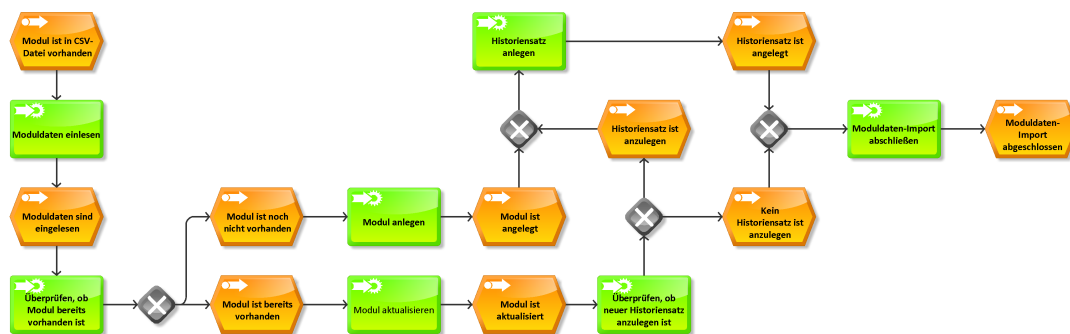


Abbildung 2: Prozess des Einlesens eines Moduls

## 4.6 Maßnahmen zur Qualitätssicherung

- Welche Maßnahmen werden ergriffen, um die Qualität des Projektergebnisses (siehe Kapitel 3.5: Qualitätsanforderungen) zu sichern (z. B. automatische Tests, Anwendertests)?
- Ggfs. Definition von Testfällen und deren Durchführung (durch Programme/Benutzer).

## 4.7 Pflichtenheft/Datenverarbeitungskonzept

- Auszüge aus dem Pflichtenheft/Datenverarbeitungskonzept, wenn es im Rahmen des Projekts erstellt wurde.

**Beispiel** Ein Beispiel für das auf dem Lastenheft (siehe Kapitel 3.6: Lastenheft/Fachkonzept) aufbauende Pflichtenheft ist im Anhang A.4: Pflichtenheft (Auszug) auf Seite iii zu finden.

# 5 Implementierungsphase

## 5.1 Implementierung der Datenstrukturen

- Beschreibung der angelegten Datenbank (z. B. Generierung von SQL aus Modellierungswerkzeug oder händisches Anlegen), XML-Schemas usw..

## 5.2 Implementierung der Benutzeroberfläche

- Beschreibung der Implementierung der Benutzeroberfläche, falls dies separat zur Implementierung der Geschäftslogik erfolgt (z. B. bei HTML-Oberflächen und Stylesheets).
- Ggfs. Beschreibung des Corporate Designs und dessen Umsetzung in der Anwendung.
- Screenshots der Anwendung

**Beispiel** Screenshots der Anwendung in der Entwicklungsphase mit Dummy-Daten befinden sich im Anhang A.7: Screenshots der Anwendung auf Seite viii.

## 5.3 Implementierung der Geschäftslogik

- Beschreibung des Vorgehens bei der Umsetzung/Programmierung der entworfenen Anwendung.
- Ggfs. interessante Funktionen/Algorithmen im Detail vorstellen, verwendete Entwurfsmuster zeigen.
- Quelltextbeispiele zeigen.
- Hinweis: Wie in Kapitel 1: Einleitung zitiert, wird nicht ein lauffähiges Programm bewertet, sondern die Projektdurchführung. Dennoch würde ich immer Quelltextausschnitte zeigen, da sonst Zweifel an der tatsächlichen Leistung des Prüflings aufkommen können.

**Beispiel** Die Klasse `ComparedNaturalModuleInformation` findet sich im Anhang A.10: Klasse: `ComparedNaturalModuleInformation` auf Seite xiii.

## 6 Abnahmephase

- Welche Tests (z. B. Unit-, Integrations-, Systemtests) wurden durchgeführt und welche Ergebnisse haben sie geliefert (z. B. Logs von Unit Tests, Testprotokolle der Anwender)?
- Wurde die Anwendung offiziell abgenommen?

**Beispiel** Ein Auszug eines Unit Tests befindet sich im Anhang A.9: Testfall und sein Aufruf auf der Konsole auf Seite xii. Dort ist auch der Aufruf des Tests auf der Konsole des Webservers zu sehen.

## 7 Einführungsphase

- Welche Schritte waren zum Deployment der Anwendung nötig und wie wurden sie durchgeführt (automatisiert/manuell)?
- Wurden ggfs. Altdaten migriert und wenn ja, wie?
- Wurden Benutzerschulungen durchgeführt und wenn ja, Wie wurden sie vorbereitet?

## 8 Dokumentation

- Wie wurde die Anwendung für die Benutzer/Administratoren/Entwickler dokumentiert (z. B. Benutzerhandbuch, API-Dokumentation)?
- Hinweis: Je nach Zielgruppe gelten bestimmte Anforderungen für die Dokumentation (z. B. keine IT-Fachbegriffe in einer Anwenderdokumentation verwenden, aber auf jeden Fall in einer Dokumentation für den IT-Bereich).

**Beispiel** Ein Ausschnitt aus der erstellten Benutzerdokumentation befindet sich im Anhang A.12: Benutzerdokumentation auf Seite xvii. Die Entwicklerdokumentation wurde mittels PHPDoc<sup>8</sup> automatisch generiert. Ein beispielhafter Auszug aus der Dokumentation einer Klasse findet sich im Anhang A.8: Entwicklerdokumentation auf Seite x.

## 9 Fazit

### 9.1 Soll-/Ist-Vergleich

- Wurde das Projektziel erreicht und wenn nein, warum nicht?
- Ist der Auftraggeber mit dem Projektergebnis zufrieden und wenn nein, warum nicht?
- Wurde die Projektplanung (Zeit, Kosten, Personal, Sachmittel) eingehalten oder haben sich Abweichungen ergeben und wenn ja, warum?
- Hinweis: Die Projektplanung muss nicht strikt eingehalten werden. Vielmehr sind Abweichungen sogar als normal anzusehen. Sie müssen nur vernünftig begründet werden (z. B. durch Änderungen an den Anforderungen, unter-/überschätzter Aufwand).

**Beispiel (verkürzt)** Wie in Tabelle 3 zu erkennen ist, konnte die Zeitplanung bis auf wenige Ausnahmen eingehalten werden.

---

<sup>8</sup>Vgl. [PHPDOC.ORG](http://PHPDOC.ORG) [2010]

Phase	Geplant	Tatsächlich	Differenz
Entwurfsphase	19 h	19 h	
Analysephase	9 h	10 h	+1 h
Implementierungsphase	29 h	28 h	-1 h
Abnahmetest der Fachabteilung	1 h	1 h	
Einführungsphase	1 h	1 h	
Erstellen der Dokumentation	9 h	11 h	+2 h
Pufferzeit	2 h	0 h	-2 h
<b>Gesamt</b>	<b>70 h</b>	<b>70 h</b>	

Tabelle 3: Soll-/Ist-Vergleich

## 9.2 Lessons Learned

- Was hat der Prüfling bei der Durchführung des Projekts gelernt (z. B. Zeitplanung, Vorteile der eingesetzten Frameworks, Änderungen der Anforderungen)?

## 9.3 Ausblick

- Wie wird sich das Projekt in Zukunft weiterentwickeln (z. B. geplante Erweiterungen)?



## Literaturverzeichnis

### Bundesministerium für Bildung und Forschung 2000

BUNDESMINISTERIUM FÜR BILDUNG UND FORSCHUNG: Umsetzungshilfen für die neue Prüfungsstruktur der IT-Berufe / Bundesministerium für Bildung und Forschung. Version: Juli 2000. <http://fiae.link/UmsetzungshilfenITBerufe>. Bonn, Juli 2000. – Abschlussbericht. – 476 S.

### Grashorn 2010

GRASHORN, Dirk: Entwicklung von NatInfo – Webbasiertes Tool zur Unterstützung der Entwickler / Alte Oldenburger Krankenversicherung AG. Vechta, April 2010. – Dokumentation zur Projektarbeit

### IHK Darmstadt 2011

IHK DARMSTADT: *Bewertungsmatrix für Fachinformatiker/innen Anwendungsentwicklung*. <http://fiae.link/BewertungsmatrixDokuDarmstadt>. Version: März 2011

### IHK Oldenburg 2006

IHK OLDENBURG: *Merkblatt zur Abschlussprüfung der IT-Berufe*. <http://fiae.link/MerkblattDokuOldenburg>. Version: Mai 2006

### ISO/IEC 9126-1 2001

ISO/IEC 9126-1: *Software-Engineering – Qualität von Software-Produkten – Teil 1: Qualitätsmodell*. Juni 2001

### phpdoc.org 2010

PHPDOC.ORG: *phpDocumentor-Website*. Version: 2010. <http://www.phpdoc.org/>, Abruf: 20.04.2010

### Regierung der Bundesrepublik Deutschland 1997

REGIERUNG DER BUNDESREPUBLIK DEUTSCHLAND: *Verordnung über die Berufsausbildung im Bereich der Informations- und Telekommunikationstechnik*. <http://fiae.link/VerordnungITBerufe>. Version: Juli 1997

### Rohrer und Sedlacek 2011

ROHRER, Anselm ; SEDLACEK, Ramona: *Cleverer Tipps für die Projektarbeit - IT-Berufe: Abschlussprüfung Teil A*. 5. Solingen : U-Form-Verlag, 2011 <http://fiae.link/ClevererTippsFuerDieProjektarbeit>. – ISBN 3882347538

### Sensio Labs 2010

SENSIO LABS: *Symfony - Open-Source PHP Web Framework*. Version: 2010. <http://www.symfony-project.org/>, Abruf: 20.04.2010

## Eidesstattliche Erklärung

Ich, Alexander Knueppel, versichere hiermit, dass ich meine **Dokumentation zur betrieblichen Projektarbeit** mit dem Thema

*RESTfull-API-Framework-RESTfull-API-Microservice mit Lasttestmodul*

selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, wobei ich alle wörtlichen und sinngemäßen Zitate als solche gekennzeichnet habe. Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Stuttgart, den 03.12.2024

---

ALEXANDER KNUEPPEL

## A Anhang

### A.1 Detaillierte Zeitplanung

<b>Analysephase</b>	<b>9 h</b>
1. Analyse des Ist-Zustands	3 h
1.1. Fachgespräch mit der EDV-Abteilung	1 h
1.2. Prozessanalyse	2 h
2. „Make or buy“-Entscheidung und Wirtschaftlichkeitsanalyse	1 h
3. Erstellen eines „Use-Case“-Diagramms	2 h
4. Erstellen des Lastenhefts mit der EDV-Abteilung	3 h
<b>Entwurfsphase</b>	<b>19 h</b>
1. Prozessentwurf	2 h
2. Datenbankentwurf	3 h
2.1. ER-Modell erstellen	2 h
2.2. Konkretes Tabellenmodell erstellen	1 h
3. Erstellen von Datenverarbeitungskonzepten	4 h
3.1. Verarbeitung der CSV-Daten	1 h
3.2. Verarbeitung der SVN-Daten	1 h
3.3. Verarbeitung der Sourcen der Programme	2 h
4. Benutzeroberflächen entwerfen und abstimmen	2 h
5. Erstellen eines UML-Komponentendiagramms der Anwendung	4 h
6. Erstellen des Pflichtenhefts	4 h
<b>Implementierungsphase</b>	<b>29 h</b>
1. Anlegen der Datenbank	1 h
2. Umsetzung der HTML-Oberflächen und Stylesheets	4 h
3. Programmierung der PHP-Module für die Funktionen	23 h
3.1. Import der Modulinformationen aus CSV-Dateien	2 h
3.2. Parsen der Modulquelltexte	3 h
3.3. Import der SVN-Daten	2 h
3.4. Vergleichen zweier Umgebungen	4 h
3.5. Abrufen der von einem zu wählenden Benutzer geänderten Module	3 h
3.6. Erstellen einer Liste der Module unter unterschiedlichen Aspekten	5 h
3.7. Anzeigen einer Liste mit den Modulen und geparsten Metadaten	3 h
3.8. Erstellen einer Übersichtsseite für ein einzelnes Modul	1 h
4. Nächtlichen Batchjob einrichten	1 h
<b>Abnahmetest der Fachabteilung</b>	<b>1 h</b>
1. Abnahmetest der Fachabteilung	1 h
<b>Einführungsphase</b>	<b>1 h</b>
1. Einführung/Benutzerschulung	1 h
<b>Erstellen der Dokumentation</b>	<b>9 h</b>
1. Erstellen der Benutzerdokumentation	2 h
2. Erstellen der Projektdokumentation	6 h
3. Programmdokumentation	1 h
3.1. Generierung durch PHPdoc	1 h
<b>Pufferzeit</b>	<b>2 h</b>
1. Puffer	2 h
<b>Gesamt</b>	<b>70 h</b>

## A.2 Lastenheft (Auszug)

Es folgt ein Auszug aus dem Lastenheft mit Fokus auf die Anforderungen:

Die Anwendung muss folgende Anforderungen erfüllen:

1. Verarbeitung der Moduldaten
  - 1.1. Die Anwendung muss die von Subversion und einem externen Programm bereitgestellten Informationen (z.B. Source-Benutzer, -Datum, Hash) verarbeiten.
  - 1.2. Auslesen der Beschreibung und der Stichwörter aus dem Sourcecode.
2. Darstellung der Daten
  - 2.1. Die Anwendung muss eine Liste aller Module erzeugen inkl. Source-Benutzer und -Datum, letztem Commit-Benutzer und -Datum für alle drei Umgebungen.
  - 2.2. Verknüpfen der Module mit externen Tools wie z.B. Wiki-Einträgen zu den Modulen oder dem Sourcecode in Subversion.
  - 2.3. Die Sourcen der Umgebungen müssen verglichen und eine schnelle Übersicht zur Einhaltung des allgemeinen Entwicklungsprozesses gegeben werden.
  - 2.4. Dieser Vergleich muss auf die von einem bestimmten Benutzer bearbeiteten Module eingeschränkt werden können.
  - 2.5. Die Anwendung muss in dieser Liste auch Module anzeigen, die nach einer Bearbeitung durch den gesuchten Benutzer durch jemand anderen bearbeitet wurden.
  - 2.6. Abweichungen sollen kenntlich gemacht werden.
  - 2.7. Anzeigen einer Übersichtsseite für ein Modul mit allen relevanten Informationen zu diesem.
3. Sonstige Anforderungen
  - 3.1. Die Anwendung muss ohne das Installieren einer zusätzlichen Software über einen Webbrowser im Intranet erreichbar sein.
  - 3.2. Die Daten der Anwendung müssen jede Nacht bzw. nach jedem SVN-Commit automatisch aktualisiert werden.
  - 3.3. Es muss ermittelt werden, ob Änderungen auf der Produktionsumgebung vorgenommen wurden, die nicht von einer anderen Umgebung kopiert wurden. Diese Modulliste soll als Mahnung per E-Mail an alle Entwickler geschickt werden (Peer Pressure).
  - 3.4. Die Anwendung soll jederzeit erreichbar sein.
  - 3.5. Da sich die Entwickler auf die Anwendung verlassen, muss diese korrekte Daten liefern und darf keinen Interpretationsspielraum lassen.
  - 3.6. Die Anwendung muss so flexibel sein, dass sie bei Änderungen im Entwicklungsprozess einfach angepasst werden kann.

### A.3 Use Case-Diagramm

Use Case-Diagramme und weitere UML-Diagramme kann man auch direkt mit  $\text{\LaTeX}$  zeichnen, siehe z. B. <http://metauml.sourceforge.net/old/usecase-diagram.html>.

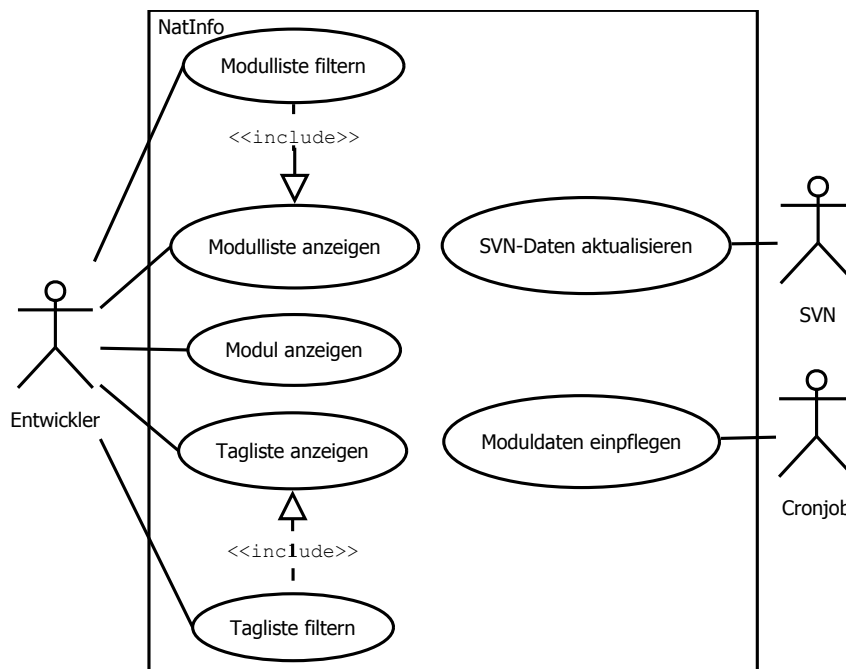


Abbildung 3: Use Case-Diagramm

### A.4 Pflichtenheft (Auszug)

#### Zielbestimmung

##### 1. Musskriterien

1.1. Modul-Liste: Zeigt eine filterbare Liste der Module mit den dazugehörigen Kerninformationen sowie Symbolen zur Einhaltung des Entwicklungsprozesses an

- In der Liste wird der Name, die Bibliothek und Daten zum Source und Kompilat eines Moduls angezeigt.
- Ebenfalls wird der Status des Moduls hinsichtlich Source und Kompilat angezeigt. Dazu gibt es unterschiedliche Status-Zeichen, welche symbolisieren in wie weit der Entwicklungsprozess eingehalten wurde bzw. welche Schritte als nächstes getan werden müssen. So gibt es z. B. Zeichen für das Einhalten oder Verletzen des Prozesses oder den Hinweis auf den nächsten zu tätigen Schritt.
- Weiterhin werden die Benutzer und Zeitpunkte der aktuellen Version der Sourcen und Kompilate angezeigt. Dazu kann vorher ausgewählt werden, von welcher Umgebung diese Daten gelesen werden sollen.

- Es kann eine Filterung nach allen angezeigten Daten vorgenommen werden. Die Daten zu den Sourcen sind historisiert. Durch die Filterung ist es möglich, auch Module zu finden, die in der Zwischenzeit schon von einem anderen Benutzer editiert wurden.

#### 1.2. Tag-Liste: Bietet die Möglichkeit die Module anhand von Tags zu filtern.

- Es sollen die Tags angezeigt werden, nach denen bereits gefiltert wird und die, die noch der Filterung hinzugefügt werden könnten, ohne dass die Ergebnisliste leer wird.
- Zusätzlich sollen die Module angezeigt werden, die den Filterkriterien entsprechen. Sollten die Filterkriterien leer sein, werden nur die Module angezeigt, welche mit einem Tag versehen sind.

#### 1.3. Import der Moduldaten aus einer bereitgestellten CSV-Datei

- Es wird täglich eine Datei mit den Daten der aktuellen Module erstellt. Diese Datei wird (durch einen Cronjob) automatisch nachts importiert.
- Dabei wird für jedes importierte Modul ein Zeitstempel aktualisiert, damit festgestellt werden kann, wenn ein Modul gelöscht wurde.
- Die Datei enthält die Namen der Umgebung, der Bibliothek und des Moduls, den Programmtyp, den Benutzer und Zeitpunkt des Sourcecodes sowie des Kompilats und den Hash des Sourcecodes.
- Sollte sich ein Modul verändert haben, werden die entsprechenden Daten in der Datenbank aktualisiert. Die Veränderungen am Source werden dabei aber nicht ersetzt, sondern historisiert.

#### 1.4. Import der Informationen aus Subversion (SVN). Durch einen „post-commit-hook“ wird nach jedem Einchecken eines Moduls ein PHP-Script auf der Konsole aufgerufen, welches die Informationen, die vom SVN-Kommandozeilentool geliefert werden, an NATINFO übergibt.

#### 1.5. Parsen der Sourcen

- Die Sourcen der Entwicklungsumgebung werden nach Tags, Links zu Artikeln im Wiki und Programmbeschreibungen durchsucht.
- Diese Daten werden dann entsprechend angelegt, aktualisiert oder nicht mehr gesetzte Tags/Wikiartikel entfernt.

#### 1.6. Sonstiges

- Das Programm läuft als Webanwendung im Intranet.
- Die Anwendung soll möglichst leicht erweiterbar sein und auch von anderen Entwicklungsprozessen ausgehen können.
- Eine Konfiguration soll möglichst in zentralen Konfigurationsdateien erfolgen.

Abbildung 4: Datenbankmodell

## A.6 Oberflächenentwürfe

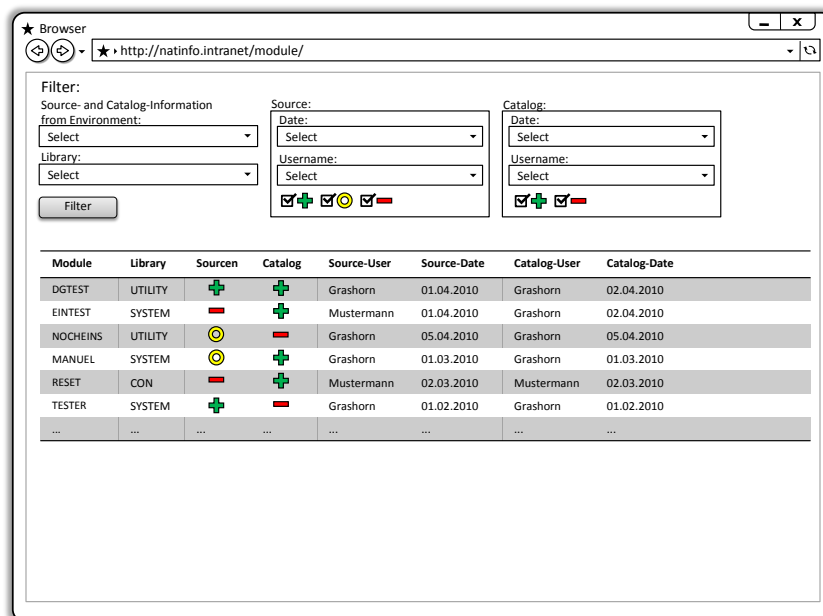


Abbildung 5: Liste der Module mit Filtermöglichkeiten



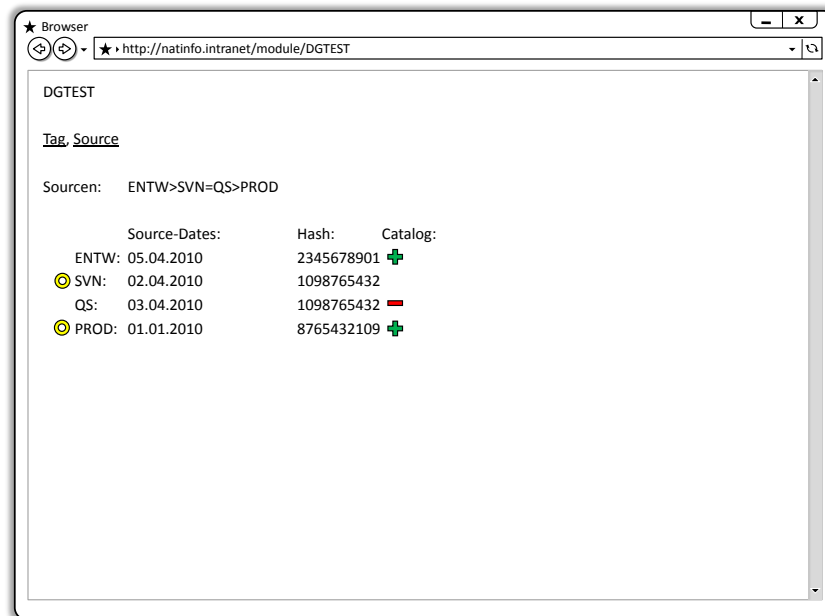


Abbildung 6: Anzeige der Übersichtsseite einzelner Module

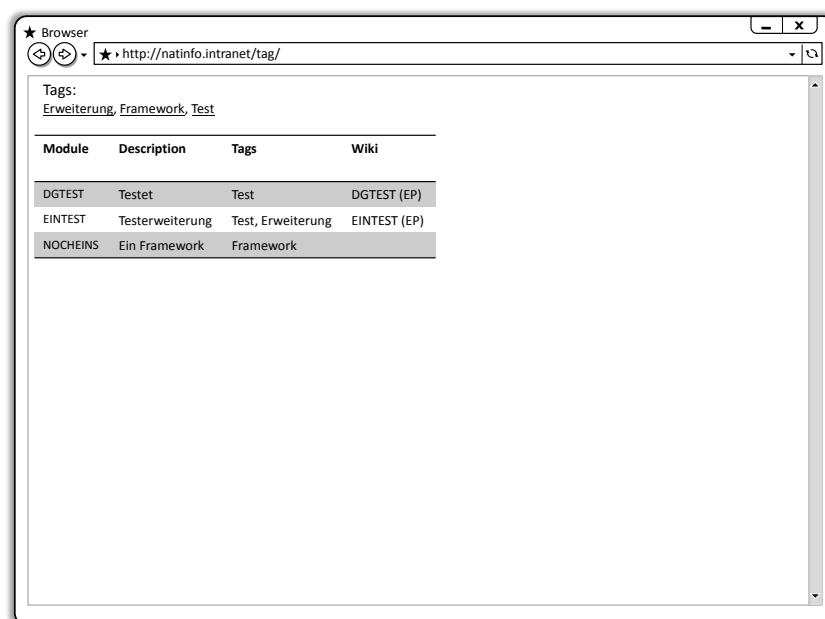


Abbildung 7: Anzeige und Filterung der Module nach Tags

## A.7 Screenshots der Anwendung



### Tags

Project, Test

Modulename	Description	Tags	Wiki
DGTEST	Macht einen ganz tollen Tab.	HGP	SMTAB_(EP), b
MALWAS		HGP, Test	
HDRGE		HGP, Project	
WURAM		HGP, Test	
PAMIU		HGP	

Abbildung 8: Anzeige und Filterung der Module nach Tags



## Modules

Environment	ENTW
Library	Select
Catalog user	Select
Catalog date	Select
Source user	Select
Source date	Select
<a href="#">Reset</a> <a href="#">Filter</a>	











Name	Library	Source	Catalog	Source-User	Source-Date	Catalog-User	Catalog-Date
SMTAB	UTILITY			MACKE	01.04.2010 13:00	MACKE	01.04.2010 13:00
DGTAB	CON			GRASHORN	01.04.2010 13:00	GRASHORN	01.04.2010 13:00
DGTEST	SUP			GRASHORN	05.04.2010 13:00	GRASHORN	05.04.2010 13:00
OHNETAG	CON			GRASHORN	05.04.2010 13:00	GRASHORN	01.04.2010 15:12
OHNEWIKI	CON			GRASHORN	05.04.2010 13:00	MACKE	01.04.2010 15:12

Abbildung 9: Liste der Module mit Filtermöglichkeiten

## A.8 Entwicklerdokumentation

lib-model

[ class tree: lib-model ] [ index: lib-model ] [ all elements ]

**Packages:**  
lib-model

**Files:**  
Naturalmodulename.php

**Classes:**  
Naturalmodulename

### Class: Naturalmodulename

Source Location: /Naturalmodulename.php

**Class Overview**

BaseNaturalmodulename  
|  
--Naturalmodulename

Subclass for representing a row from the 'NaturalModulename' table.

**Methods**

- [\\_\\_construct](#)
- [getNaturalTags](#)
- [getNaturalWikis](#)
- [loadNaturalModuleInformation](#)
- [\\_\\_toString](#)

#### Class Details

[line 10]  
Subclass for representing a row from the 'NaturalModulename' table.

Adds some business logic to the base.

[ [Top](#) ]

#### Class Methods

**constructor [\\_\\_construct](#) [line 56]**

```
Naturalmodulename __construct( )
```

Initializes internal state of Naturalmodulename object.

**Tags:**  
**see:** parent::\_\_construct()  
**access:** public

[ [Top](#) ]

**method [getNaturalTags](#) [line 68]**

```
array getNaturalTags( )
```

Returns an Array of NaturalTags connected with this Modulename.

**Tags:**

**return:** Array of NaturalTags  
**access:** public

[\[ Top \]](#)

---

**method getNaturalWikis** [line 83]

```
array getNaturalWikis( )
```

Returns an Array of NaturalWikis connected with this Modulename.

**Tags:**

**return:** Array of NaturalWikis  
**access:** public

[\[ Top \]](#)

---

**method loadNaturalModuleInformation** [line 17]

```
ComparedNaturalModuleInformation  
loadNaturalModuleInformation( )
```

Gets the ComparedNaturalModuleInformation for this NaturalModulename.

**Tags:**

**access:** public

[\[ Top \]](#)

---

**method \_\_toString** [line 47]

```
string __toString( )
```

Returns the name of this NaturalModulename.

**Tags:**

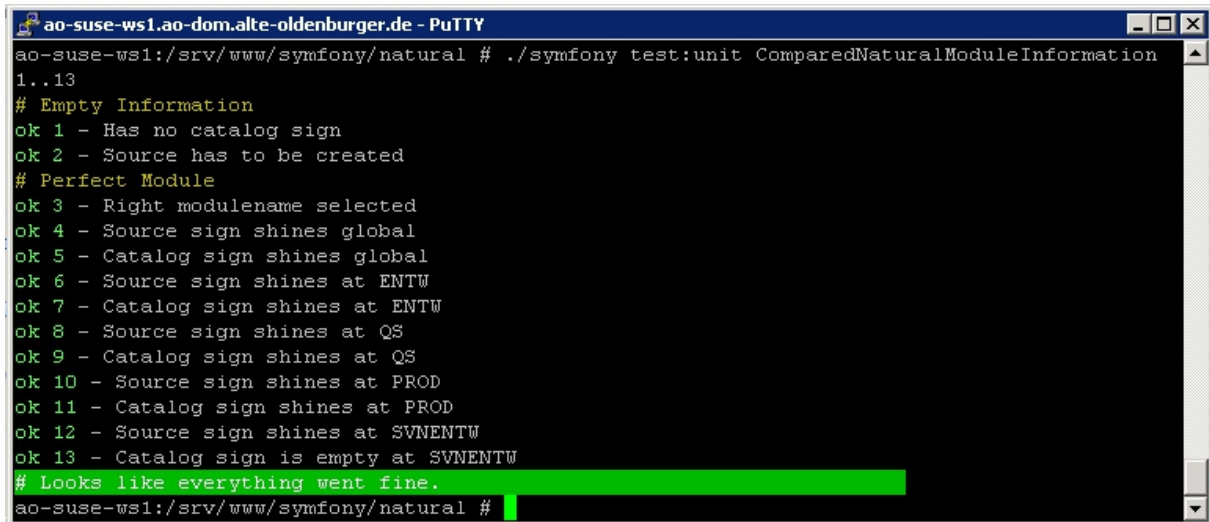
**access:** public

[\[ Top \]](#)

## A.9 Testfall und sein Aufruf auf der Konsole

```
1 <?php
2 include(dirname(__FILE__).'/../bootstrap/Propel.php');
3
4 $t = new lime_test(13);
5
6 $t->comment('Empty Information');
7 $emptyComparedInformation = new ComparedNaturalModuleInformation(array());
8 $t->is($emptyComparedInformation->getCatalogSign(), ComparedNaturalModuleInformation::
    EMPTY_SIGN, 'Has no catalog sign');
9 $t->is($emptyComparedInformation->getSourceSign(), ComparedNaturalModuleInformation::
    SIGN_CREATE, 'Source has to be created');
10
11 $t->comment('Perfect Module');
12 $criteria = new Criteria();
13 $criteria->add(NaturalmodulePeer::NAME, 'SMTAB');
14 $moduleName = NaturalmodulePeer::doSelectOne($criteria);
15 $t->is($moduleName->getName(), 'SMTAB', 'Right module name selected');
16 $comparedInformation = $moduleName->loadNaturalModuleInformation();
17 $t->is($comparedInformation->getSourceSign(), ComparedNaturalModuleInformation::SIGN_OK, 'Source
    sign shines global');
18 $t->is($comparedInformation->getCatalogSign(), ComparedNaturalModuleInformation::SIGN_OK, 'Catalog
    sign shines global');
19 $infos = $comparedInformation->getNaturalModuleInformations();
20 foreach($infos as $info)
21 {
22     $env = $info->getEnvironmentName();
23     $t->is($info->getSourceSign(), ComparedNaturalModuleInformation::SIGN_OK, 'Source sign shines at ' .
        $env);
24     if($env != 'SVNENTW')
25     {
26         $t->is($info->getCatalogSign(), ComparedNaturalModuleInformation::SIGN_OK, 'Catalog sign shines at
            ' . $info->getEnvironmentName());
27     }
28     else
29     {
30         $t->is($info->getCatalogSign(), ComparedNaturalModuleInformation::EMPTY_SIGN, 'Catalog sign is
            empty at ' . $info->getEnvironmentName());
31     }
32 }
33 ?>
```

Listing 1: Testfall in PHP



```
ao-suse-ws1.ao-dom.alte-oldenburger.de - PuTTY
ao-suse-ws1:/srv/www/symfony/natural # ./symfony test:unit ComparedNaturalModuleInformation
1..13
# Empty Information
ok 1 - Has no catalog sign
ok 2 - Source has to be created
# Perfect Module
ok 3 - Right modulename selected
ok 4 - Source sign shines global
ok 5 - Catalog sign shines global
ok 6 - Source sign shines at ENTW
ok 7 - Catalog sign shines at ENTW
ok 8 - Source sign shines at QS
ok 9 - Catalog sign shines at QS
ok 10 - Source sign shines at PROD
ok 11 - Catalog sign shines at PROD
ok 12 - Source sign shines at SVNENTW
ok 13 - Catalog sign is empty at SVNENTW
# Looks like everything went fine.
ao-suse-ws1:/srv/www/symfony/natural #
```

Abbildung 10: Aufruf des Testfalls auf der Konsole

## A.10 Klasse: ComparedNaturalModuleInformation

Kommentare und simple Getter/Setter werden nicht angezeigt.

```
1 <?php
2 class ComparedNaturalModuleInformation
3 {
4     const EMPTY_SIGN = 0;
5     const SIGN_OK = 1;
6     const SIGN_NEXT_STEP = 2;
7     const SIGN_CREATE = 3;
8     const SIGN_CREATE_AND_NEXT_STEP = 4;
9     const SIGN_ERROR = 5;
10
11     private $naturalModuleInformations = array();
12
13     public static function environments()
14     {
15         return array("ENTW", "SVNENTW", "QS", "PROD");
16     }
17
18     public static function signOrder()
19     {
20         return array(self::SIGN_ERROR, self::SIGN_NEXT_STEP, self::
21             SIGN_CREATE_AND_NEXT_STEP, self::SIGN_CREATE, self::SIGN_OK);
22     }
23
24     public function __construct(array $naturalInformations)
25     {
26         $this->allocateModulesToEnvironments($naturalInformations);
27         $this->allocateEmptyModulesToMissingEnvironments();
28         $this->determineSourceSignsForAllEnvironments();
29     }
30 }
```

```
28 }
29
30 private function allocateModulesToEnvironments(array $naturalInformations)
31 {
32     foreach ($naturalInformations as $naturalInformation)
33     {
34         $env = $naturalInformation->getEnvironmentName();
35         if(in_array($env, self::environments()))
36         {
37             $this->naturalModuleInformations[array_search($env, self::environments())] = $naturalInformation;
38         }
39     }
40 }
41
42 private function allocateEmptyModulesToMissingEnvironments()
43 {
44     if(array_key_exists(0, $this->naturalModuleInformations))
45     {
46         $this->naturalModuleInformations[0]->setSourceSign(self::SIGN_OK);
47     }
48
49     for($i = 0; $i < count(self::environments()); $i++)
50     {
51         if(!array_key_exists($i, $this->naturalModuleInformations))
52         {
53             $environments = self::environments();
54             $this->naturalModuleInformations[$i] = new EmptyNaturalModuleInformation($environments[$i]);
55             $this->naturalModuleInformations[$i]->setSourceSign(self::SIGN_CREATE);
56         }
57     }
58 }
59
60 public function determineSourceSignsForAllEnvironments()
61 {
62     for($i = 1; $i < count(self::environments()); $i++)
63     {
64         $currentInformation = $this->naturalModuleInformations[$i];
65         $previousInformation = $this->naturalModuleInformations[$i - 1];
66         if($currentInformation->getSourceSign() <> self::SIGN_CREATE)
67         {
68             if($previousInformation->getSourceSign() <> self::SIGN_CREATE)
69             {
70                 if($currentInformation->getHash() <> $previousInformation->getHash())
71                 {
72                     if($currentInformation->getSourceDate('YmdHis') > $previousInformation->getSourceDate('YmdHis'))
73                     {
74                         $currentInformation->setSourceSign(self::SIGN_ERROR);
75                     }
76                     else
77                     {
```



```
78         $currentInformation->setSourceSign(self::SIGN_NEXT_STEP);
79     }
80 }
81 else
82 {
83     $currentInformation->setSourceSign(self::SIGN_OK);
84 }
85 }
86 else
87 {
88     $currentInformation->setSourceSign(self::SIGN_ERROR);
89 }
90 }
91 elseif ($previousInformation->getSourceSign() <> self::SIGN_CREATE && $previousInformation->
    getSourceSign() <> self::SIGN_CREATE_AND_NEXT_STEP)
92 {
93     $currentInformation->setSourceSign(self::SIGN_CREATE_AND_NEXT_STEP);
94 }
95 }
96 }
97
98 private function containsSourceSign($sign)
99 {
100     foreach($this->naturalModuleInformations as $information)
101     {
102         if ($information->getSourceSign() == $sign)
103         {
104             return true;
105         }
106     }
107     return false ;
108 }
109
110 private function containsCatalogSign($sign)
111 {
112     foreach($this->naturalModuleInformations as $information)
113     {
114         if ($information->getCatalogSign() == $sign)
115         {
116             return true;
117         }
118     }
119     return false ;
120 }
121 }
122 ?>
```

Listing 2: Klasse: ComparedNaturalModuleInformation

## A.11 Klassendiagramm

Klassendiagramme und weitere UML-Diagramme kann man auch direkt mit  $\text{\LaTeX}$  zeichnen, siehe z. B. <http://metauml.sourceforge.net/old/class-diagram.html>.

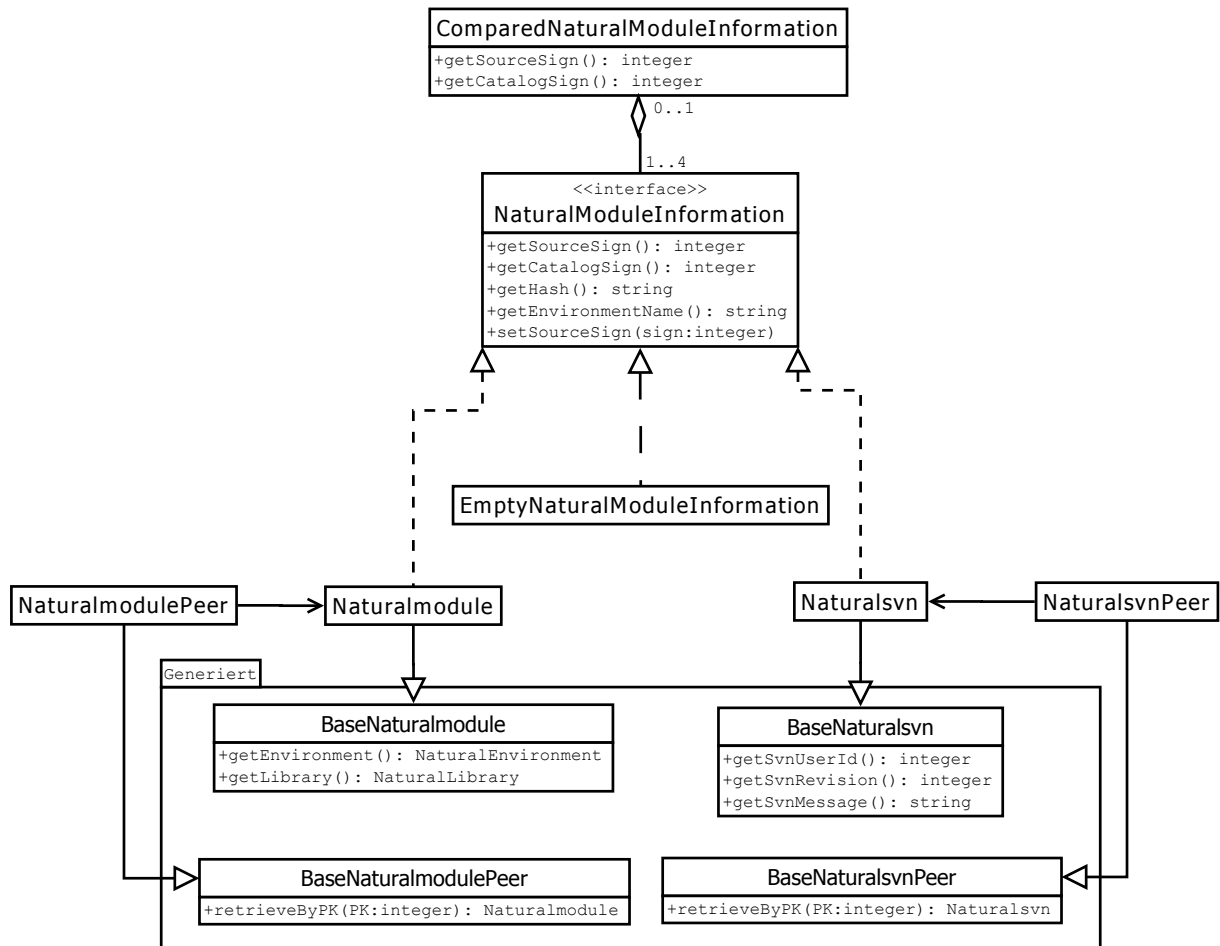







Abbildung 11: Klassendiagramm

## A.12 Benutzerdokumentation

Ausschnitt aus der Benutzerdokumentation:

Symbol	Bedeutung global	Bedeutung einzeln
	Alle Module weisen den gleichen Stand auf.	Das Modul ist auf dem gleichen Stand wie das Modul auf der vorherigen Umgebung.
	Es existieren keine Module (fachlich nicht möglich).	Weder auf der aktuellen noch auf der vorherigen Umgebung sind Module angelegt. Es kann also auch nichts übertragen werden.
	Ein Modul muss durch das Übertragen von der vorherigen Umgebung erstellt werden.	Das Modul der vorherigen Umgebung kann übertragen werden, auf dieser Umgebung ist noch kein Modul vorhanden.
	Auf einer vorherigen Umgebung gibt es ein Modul, welches übertragen werden kann, um das nächste zu aktualisieren.	Das Modul der vorherigen Umgebung kann übertragen werden um dieses zu aktualisieren.
	Ein Modul auf einer Umgebung wurde entgegen des Entwicklungsprozesses gespeichert.	Das aktuelle Modul ist neuer als das Modul auf der vorherigen Umgebung oder die vorherige Umgebung wurde übersprungen.