HOCHSCHULE FURTWANGEN UNIVERSITY HFU

Bachelorarbeit

in

Medieninformatik

Zu Erlangung des Grades Bachelor of Science

Entwicklung eines speziellen DevOps Projekt-Konfigurator Werkzeuges für Projektinitialisierung /– aktualisierung

Referent:	Herr Prof. Dr. Eisenbiegler
Korreferent:	Herr Dr.-Ing- Christoph Rathfelder
Vorgelegt am:	31.12.2021
Vorgelegt von:	Nick Stecker
	25195
	Haselstiegstr. 5/1
	78052 Villingen
	nick.stecker@hs-furtwangen.de

Eidesstattliche Erklärung

Ich versichere, dass ich die vorstehende Arbeit selbständig verfasst und hierzu keine anderen als die angegebenen Hilfsmittel verwendet habe. Alle Stellen der Arbeit die wörtlich oder sinngemäß aus fremden Quellen entnommen wurden, sind als solche kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keinem anderen Studiengang als Prüfungsleistung vorgelegt oder an anderer Stelle veröffentlicht.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben kann.

_____

[Ort, Datum Name]

Abstract

[Englisch, 100 -120 Worte]

[Deutsch, 100 – 120 Worte]

Inhaltsverzeichnis

Abbildungsverzeichnis

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.

Tabellenverzeichnis

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.

Abkürzungsverzeichnis

DevOps Development und IT Operations

GUI Graphical User Interface

IDE Integrated Development Environment

UML Unified Modeling Language

1. Einleitung

Die moderne Softwareentwicklung erfordert eine Vielzahl externer Programmierwerkzeuge (Tools), wie beispielsweise Integrated Development Environments (IDEs), Versionsverwaltungen, spezielle Build Systeme oder Frameworks, um den Softwareentwicklungsprozess professionell gestalten und verwalten zu können.

Dabei werden diese Tools nacheinander an bestimmten Stellen in dem Entwicklungsprozess eingesetzt, um spezifische Aufgaben zu lösen.

In der Praxis werden die Tools dabei zu einer sogenannten Werkzeugkette (Toolchain) miteinander verknüpft, um einzelne Prozesse in der Softwareentwicklung zu automatisieren.

…dadurch können einzelne Prozesse in dem gesamten Entwicklungsprozess automatisiert werden.

Allerdings erfüllen die Tools nicht immer allen Ansprüchen eines Softwareentwicklers. Eines dieser Probleme ist...

In vielen Berufsfeldern wäre die Arbeit ohne spezielle Werkzeuge undenkbar. Eines dieser Berufsfelder ist die Softwareentwicklung. Für eine professionelle Softwareentwicklung werden heutzutage nämlich viele verschiedene Programmierwerkzeuge (Tools) eingesetzt, die den Software-Entwicklungsprozess optimal unterstützen und beherrschbar machen.

In dieser Bachelorarbeit geht es um die Entwicklung eines solchen Tools. Der Projekt-Konfigurator ist ein spezielles Werkzeug für die Softwareentwicklung. Es soll Softwareentwicklern dabei helfen Softwareprojekte, basierend auf einem Projekt-Template zu erstellen und zu aktualisieren. Das Template-Projekt ist dabei einfach ein Ordner auf dem Computer mit einer gewissen Ordnerstruktur. Diese Ordnerstruktur setzt sich aus verschiedenen Unterordnern und Dateien zusammen.

Je nach Anwendungsfall können Toolchains aus unterschiedlichen Tools aufgebaut sein. Die größte Verwendung finden Toolchains jedoch in der Umsetzung der Continuous Integration und Continuous Delivery (CI/CD) Methode.

1.1 Nutzen dieser Bachelorarbeit

Diese Abschlussarbeit löst ein spezifisches Problem der Abteilung Anwendungsentwicklung von der Hahn-Schickard Gesellschaft in Villingen-Schwenningen. Das Problem bezieht sich hierbei auf die Initialisierung bzw. Aktualisierung eines bestehenden Projektes durch ein existierendes Template. Das Problem soll durch den in Abschnitt 1 beschriebene Projekt-Konfigurator gelöst werden.

1.2 Die Problemstellung

Die Entwicklung großer Softwareprojekte kann eine sehr schwierige Aufgabe sein, insbesondere wenn das Projekt nicht gut strukturiert ist. Darüber hinaus sind viele moderne Projekte aus modularen Teilen aufgebaut, die später für verschiedene Anwendungsfälle wiederverwendet werden können. Um die Wiederverwendbarkeit und Verbesserung dieser Module zu verbessern, werden sie oft selbst als Softwareprojekte mit eigenem Layout und zugehörigen Werkzeugen erstellt. Es ist kein unwahrscheinliches Szenario, dass sich das Layout oder die Tools dieser Projekte im Laufe der Zeit mit zunehmendem Entwicklungswissen ändern und daher zu einer Aktualisierung auffordern. Dies manuell zu tun, kann eine mühsame Aufgabe sein, insbesondere wenn es eine große Anzahl dieser Module gibt, sowie fehleranfällig. Um dieses Problem zu lindern, könnte eine grafische Benutzeroberfläche (GUI) verwendet werden, um die Projektlayoutänderungen auf einfache und leicht verständliche Weise auf jedes Modul anzuwenden. Diese Benutzeroberfläche sollte...

Die Abteilung Anwendungsentwicklung arbeitet täglich an vielen verschiedenen Softwareprojekten. Jedes Softwareprojekt, besitzt dabei, je nach Programmiersprache, eine exakte Vorgabe über die Basis der zu verwendenden Verzeichnisstruktur. Das bedeutet, dass jedes Softwareprojekt, einer Programmiersprache, nach der Initialisierung dieselbe Ordnerstruktur aufweist. Inwiefern die Ordnerstruktur jedes Softwareprojekts in ihrer Entwicklungszeit ausgebaut wird (wie viele Dateien also hinzugefügt und bearbeitet werden), ist für jedes Softwareprojekt spezifisch. Allerdings darf sich dabei die Grundstruktur des Projekts nicht ändern.

Wenn ein Mitarbeiter der Abteilung Anwendungsentwicklung nun ein neues Softwareprojekt erstellen will, muss dieser erst einmal wissen wie die Verzeichnisstruktur für die jeweilige Programmiersprache des Projekts aufgebaut ist.

Kennt der Mitarbeiter die Struktur, muss er diese auf dem Computer manuell nachbauen. Das heißt, der Mitarbeiter muss manuell alle nötigen Ordern und Dateien in der richtigen Ordnerstruktur erstellen. Kennt der Mitarbeiter die Struktur jedoch nicht, muss der Mitarbeiter diese erst herausfinden.

Um dieses Problem zu umgehen, besitzt die Abteilung für die meisten Softwareprojekte einer Programmiersprache ein vorgefertigtes Template-Projekt. Die Template-Projekte sind in den vorgegebenen Verzeichnisstrukturen aufgebaut und besitzen alle notwenigen Order bzw. Dateien. Die meisten Dateien sind so aufgebaut, dass erstellte Projekte diese ohne Veränderungen benutzen können. Allerdings beinhalten einige Dateien Platzhalter, die bei der Projekterstellung mit den Metadaten eines Softwareprojekts ersetzt werden müssen. Ohne diese Bearbeitungen, haben die Entwickler der Abteilung nur eine exakte Kopie eines Template-Projektes – das erstellte Softwareprojekt würde bis dato also noch kein reales Projekt repräsentieren.

1.2.1 Die Projekterstellung

Die Projekterstellung mithilfe des beschriebenen Template-Projektes ist das erste große Problem dieser Bachelorarbeit. Denn um ein neues Softwareprojekt zu erstellen, müssen die Mitarbeiter das passende Template-Projekt kopieren und die Platzhalter mit den Metadaten des Projekts bearbeiten. Jedoch müssen die Mitarbeiter vor der Bearbeitung wissen, welche Template-Dateien Platzhalter beinhalten und an welchen Stellen sie sich in der Template-Datei befinden. Wenn die Mitarbeiter dies nicht wissen, müssen sie das, vor der Bearbeitung herausfinden. Wenn sie es allerdings wissen, müssen sie jede zu bearbeitende Datei manuell öffnen und alle Platzhalter eigenständig ersetzen.

Ein Beispiel: Wenn ein Entwickler den Namen eines Projekts bearbeiten/ändern möchte, muss er den Projektnamen manuell in dem Readme-Titel, in den Buildscript-Variablen und der Dokumentierung des Projekts ändern. Das sind drei Dateien die der Entwickler nur für den Projektnamen öffnen und bearbeiten muss.

Dieses Vorgehen ist im gleichen Maße zeitaufwändig wie fehleranfällig.

1.2.2 Die Projektaktualisierung

Das zweite große Problem dieser Bachelorarbeit ist, dass ein Template-Projekte jederzeit aktualisiert werden kann. Die Entwickler können jederzeit bestimmte

Order/Dateien des Template-Projektes überarbeiten, oder ihm neue Order/Dateien hinzufügen.

Die Problematik hierbei ist, dass diese Änderungen im Anschluss auf alle bestehenden Softwareprojekte, die aus dem veränderten Template-Projekt entstanden sind, übertragen werden müssen. Dazu müssen die Entwickler jedes dieser Softwareprojekte manuell auf dem Computer öffnen und in diesen die gleichen Änderungen vornehmen, die sie davor im Template-Projekt durchgeführt haben.

Dieses Vorgehen ist wie im vorherigen Abschnitt extrem zeitaufwändig und fehleranfällig.

1.3 Ziel dieser Bachelorarbeit

Das Ziel dieser Bachelorarbeit ist die Konzeption und Implementierung eines Projekt-Konfigurators, der die beschriebenen Probleme in Abschnitt 1.2.1 und Abschnitt 1.2.2 löst.

Der Projekt-Konfigurator ist ein spezielles Programmierwerkzeug und besteht aus einer Software sowie einer grafischen Benutzeroberfläche (in Englisch, Graphical User Interface; abgekürzt GUI), die von den Entwicklern der Abteilung Anwendungsentwicklung ohne weitere Vorkenntnisse benutzt werden kann. Mithilfe der GUI können die Probleme wie folgt gelöst werden:

1.4 Weiterer Aufbau

In Kapitel 2 wird der aktuelle Stand der Technik bezüglich aktuellen Möglichkeiten für die Projekterstellung /-aktualisierung beleuchtet. Bereits vorhandene Werkzeuge/Software und deren Eigenschaften werden mittels Recherche ermittelt.

Das dritte Kapitel behandelt den Lösungsansatz des beschriebenen Problems (in Abschnitt 1.2).

In Kapitel vier geht es um den Softwareentwurf des Projekts. Hier wird das geplante Aussehen der GUI anhand von Skizzen gezeigt und erklärt. Darüber hinaus werden an dieser Stelle die geplante Architektur, sowie alle Funktionalitäten mithilfe von UML Diagrammen erklärt und begründet. Ein wesentlicher Fokus liegt hier auf der Modularität des Quellcodes

Im fünften Kapitel geht es um die Implementierung und Evaluierung des Projekts. Zum einen wird hier auf die Umsetzung des Projekt-Konfigurators eingegangen. Es werden einzelne Teile des Quellcodes und verschiedene Konfigurationsdateien näher beschrieben. Zum anderen geht es in diesem Kapitel um die Evaluierung aller Funktionalitäten. Jedes Anwendungsbeispiel wird aus Anwendersicht mithilfe von GUI-Screenshots beschrieben. Implementierte Extrafunktionalitäten, die über die Aufgabenstellung hinausgehen werden, hier besonders erwähnt.

Im sechsten und letzten Kapitel geht es um eine Zusammenfassung der geleisteten Arbeit und um die Beantwortung der Frage inwiefern der Projekt-Konfigurator das beschriebene Problem (in Abschnitt 1.3) löst. Außerdem gibt dieses Kapitel noch einen Ausblick auf künftige Erweiterungspunkte.

2. Grundlagen und Stand der Technik

2.1 Die Bedeutung von DevOps

Bei der Bezeichnung DevOps handelt es sich um ein sogenanntes Kofferwort, dass sich aus den Begriffen „Development (Entwicklung)“ und „IT Operations (IT-Betrieb)“ zusammensetzt. Damit sind die jeweiligen Organisationseinheiten im IT-Bereich gemeint, die in einer traditionellen Organisation klassischerweise getrennt sind und mit unterschiedlichen Zielsetzungen arbeiten [1, 2]. Die Softwareentwicklung arbeitet meist in Projektform an der Umsetzung von Kundenanforderungen. Diese müssen schnell realisiert werden, damit das IT-Produkt und/oder –Service nicht an Akzeptanz verliert und weiterhin erfolgreich auf dem Markt bleibt. Der IT-Betrieb strebt dagegen eine entgegengesetzte Arbeitsweise an. Seine Ziele sind vor allem ein hohes Maß an Stabilität, Verfügbarkeit sowie Sicherheit [1].

Ein weiteres Problem der klassischen Organisationstruktur betrifft die mangelnde Zusammenarbeit zwischen den beiden Bereichen. Dabei ist das Ziel der Softwareentwicklung die Bereitstellung der fertigen Software („Release“). Die anschließende Installation („Deployment“) ist hingegen der Startpunkt des IT-Betriebs. Wurden in der Entwicklung die Anforderungen des IT-Betriebs für die Installation nicht berücksichtigt, kann das zu Verzögerungen oder fehlschlagen des Deployments führen [3].

Angesichts derartiger Probleme ist DevOps entstanden. Es umfasst eine Reihe von Technologien, Prozessen und Werkzeugen, die darauf ausgelegt sind, typische Probleme in der Zusammenarbeit zwischen Entwicklungs- und Betriebseinheiten zu lösen und dadurch das Kundenerlebnis und die Kundenzufriedenheit zu verbessern. Entscheidend dafür sind Veränderungen in der Zusammenarbeit dieser Bereiche sowie eine möglichst große Automatisierung der täglichen Aufgaben [3].

2.2 Continuous Integration, Continuous Delivery (CI/CD)

Damit der DevOps-Ansatz erfolgreich umgesetzt werden kann, werden hierzu viele einzelne Werkzeuge (Tools) für bestimmte Aufgaben eingesetzt. Diese Tools müssen allerdings intelligent miteinander kombiniert werden, damit der DevOps-Prozess fehlerfrei funktioniert. Der Projekt-Konfigurator, ist eines dieser speziellen Werkzeuge.

2.3 Build-Tools

Ein Build-Prozess ist ein Vorgang bei dem eine ausführbare Software erzeugt wird [4]. Abgesehen von einfachen Programmen, besteht jede Software aus vielen Quelldateien und externen Elementen. Diese müssen vor der Ausführung der Software in ein lauffähiges Konstrukt umgewandelt werden. Typischerweise wird dabei der Quellcode zuerst in eine, für den Computer verständliche, Sprache übersetzt („Code-Kompilierung“) und anschließend den kompilierten Code an Bibliotheken „gelinkt“. Durch die Verlinkung werden die einzelnen Programmmodule zu einer eigenständigen, ausführbaren Software verbunden [5].

Früher haben Softwareentwickler diese Arbeitsschritte manuell gelöst. Doch mit der Zeit wurden Softwareprojekte immer umfangreicher, wodurch auch der Aufwand für die Kompilierung und Verknüpfung vieler verschiedener Dateien immer weiter angestiegen ist [6]. Aus diesem Grund werden Build-Prozesse bei komplexen Softwareprojekten mit Build-Tools automatisiert. Diese Tools lesen dazu eine Konfigurationsdatei, in der allgemeine Rahmenbedingungen sowie spezielle Anweisungen für die einzelnen Schritte eines Build-Prozesses steht. Jeder Schritt kann dabei von verschiedenen Bedingungen abhängig sein, wie zum Beispiel die Aktualität oder Existenz bestimmter Dateien. Dadurch wird einerseits sichergestellt, dass die einzelnen Schritte in der richtigen Reihenfolge durchgeführt werden, andererseits kann dadurch unnötige Arbeit vermieden werden, indem noch aktuelle Teilergebnisse in weiteren Durchläufen wiederverwendet werden und nicht in jedem Arbeitsschritt neu generiert werden [5].

2.4 Integrierte Entwicklungsumgebung (IDE)

Um einfache Programme entwickeln zu können, benötigen Softwareentwickler grundsätzlich nur zwei Werkzeuge. Einen Code-Editor, um den Quellcode zu schreiben und einen Compiler bzw. Interpreter, um die Programme in den für den Computer verständlichen, ausführbaren Maschinencode zu übersetzen. Für eine effiziente und professionelle Entwicklung von komplexeren Anwendungen sind allerdings weitere Werkzeuge notwendig, um den Entwicklern einzelne Arbeitsschritte abzunehmen oder zu erleichtern [7]. Damit bei der Entwicklung einer Anwendung nicht jedes Tool einzeln verwendet werden muss, kamen in der ersten Hälfte der 1980er Jahre sogenannte integrierte Entwicklungsumgebungen (engl.: „integrated development environment“, IDE) auf den Markt [8]. Das Besondere der IDEs ist, dass

diese die einzelnen Tools, in einer gemeinsamen Benutzeroberfläche vereinen. Dadurch können verschiedene Aufgaben in der Softwareentwicklung möglichst ohne Medienbrüche bearbeitet werden. Die wichtigsten Funktionalitäten, die die meisten modernen IDEs gemeinsam haben sind z.B.:

- Direkter Aufruf des Compilers/Interpreters
- Funktionen zur effizienten Bearbeitung des Programmcodes (bspw. Syntax Highlighting oder Autovervollständigung)
- Funktionalität zur Fehlersuche und –behebung
- Versionsverwaltung der Dateien basierend auf Git
- Funktionen zur Erstellung und Verwaltung ganzer Projekte mit mehreren Dateien [7]

Letztere ist zugleich auch die wichtigste Funktionalität in Bezug auf diese Bachelorarbeit. Im Folgenden wird diese Funktionalität anhand der drei meistgenutzten IDEs für C++ Projekte näher erläutert und deutlich gemacht, warum IDEs nicht ausreichen, um das Problem dieser Bachelorarbeit zu lösen.

2.4.1 Eclipse IDE

2.4.2 Visual Studio Code IDE

Visual Studio Code (abgekürzt VSCode) ist eine kostenloser und open source (github.com/microsoft/vscode) basierter Code-Editor von Microsoft. Es ist plattformübergreifend auf den Betriebssystemen Windows, Linux und macOS verfügbar [9].

VSCode basiert auf dem Framework Electron, mit dem sich plattformübergreifende Anwendungen entwickeln lassen, und vereint viele verschiedene Tools, die ein Entwickler für die moderne Softwareentwicklung benötigt. Unter diesen Tools befinden sich sowohl die Tools, die in Abschnitt 2.4 aufgelistet wurden, als auch viele weitere, die im Rahmen dieser Bachelorarbeit allerdings nicht von Relevanz sind [9].

Wie in Abschnitt 2.4 erwähnt, ist die wichtigste Funktionalität die, der Erstellung und Verwaltung ganzer Projekte mit mehreren Dateien. Um ein neues Projekt in VSCode anzulegen, muss unter **Datei** > **Ordner öffnen…** ein Verzeichnis auswählt werden, in welches ein neues Projekt angelegt werden soll.

[Screenshot]

Nachdem ein Verzeichnis ausgewählt wurde, kann darin ein neuer Ordner angelegt werden, welcher das neue Projekt repräsentieren soll.

[Screenshot]

Diesem Ordner können im Nachhinein auf die gleiche Weise weitere Unterorder und Dateien, die für das Projekt notwendig sind, hinzugefügt werden [10].

Eine weitere Möglichkeit ein neues Projekt zu erstellen ist bietet das integrierte Terminal in VSCode. Dazu kann unter **Terminal** > **Neues Terminal** ein neues Terminal geöffnet werden.

[Screenshot]

In diesem Terminal können dann alle notwendigen Befehle eingegeben werden, um ein neuen Ordner unter dem gewünschten Verzeichnis zu erstellen und diesen danach mit notwendigen Quelldateien zu füllen [10].

Diese Vorgehensweisen sind allerdings kontraproduktiv, wenn es darum geht ein Softwareprojekt (wie in der Problemstellung (Abschnitt 1.1) erklärt) nach einer bestimmten Vorlage zu erstellen. Allerdings kann dieses Problem mit der Extension „Project Templates“ von cantonios (marketplace.visualstudio.com/project-templates) gelöst werden. Nach der Installation dieser Erweiterung ist es zum einen möglich ein bestimmtes Projekt als ein Template-Projekt zu speichern. Dazu muss als erstes ein zuvor initialisiertes Projekt in VSCode geöffnet werden. Nachdem das Projekt geöffnet wurde, muss im Explorer ein Kontextmenü geöffnet werden, indem man mit der rechten Maustaste auf das Projekt klickt. Wenn die Erweiterung richtig installiert wurde, sollte nun „Save Project as Template“ als Auswahl in dem Kontextmenü erscheinen.

[Screenshot]

Nachdem dieser Menüpunkt ausgewählt wurde, öffnet sich die Befehlspalette von VSCode, in der man einen beliebigen Namen für dieses Template-Projekt vergeben kann.

[Screenshot]

Zum Schluss muss der eingegebene Name mit der Eingabetaste auf der Tastatur bestätigt werden, um das Projekt automatisch als Template-Projekt zu speichern. Mit

den Standardeinstellungen von „Project Templates“ werden alle Template-Projekte unter folgenden Pfaden gespeichert:

```
$HOME/.config/Code/User/ProjectTemplates                        # Linux
$HOME/Library/Application Support/Code/User/ProjectTemplates    # macOS
%APPDATA%\Code\User\ProjectTemplates                            # Windows
```

Diese können aber in den Benutzereinstellungen unter **Verwalten** > **Einstellungen** > **VSCode Project Template Configuration** > **In „settings.json“ bearbeiten** geändert werden, indem man folgende Zeile addiert:

```
"projectTemplates.templatesDirectory": "path/to/my/templates"
```

Zum anderen kann die Extension dazu benutzt werden, um ein Projekt aus einem gespeicherten Template-Projekt zu erstellen. Dazu muss in VSCode ein Verzeichnis geöffnet werden, in dem ein leerer Ordner liegt. In diesem leeren Ordner wird das neue Projekt erstellt. Wie in dem Abschnitt zuvor muss man dafür mit der rechten Maustaste auf den leeren Ordner klicken, damit sich ein Kontextmenü öffnet. In dem Menü sollte sich der Menüpunkt „Create Project from Template“ befinden.

[Screenshot]

Mit der Auswahl des Menüpunktes öffnet sich die Befehlspalette in VSCode mit einer Liste aller gespeicherten Template-Projekte.

[Screenshot]

Als letztes muss nur noch das gewünschte Template-Projekt ausgewählt werden. Dadurch wird der Inhalt (alle Unterordner und Dateien) des ausgewählten Template-Projektes in den leeren Ordner kopiert. Damit besitzt das neue Projekt am Ende des Vorgangs die gleiche Verzeichnisstruktur, wie das Template-Projekt [11].

[Screenshot]

Das Problem dieser Extension ist, dass der Funktionsumfang bei diesen Funktionalitäten endet. Wie in der Problemstellung (Abschnitt 1.1) beschrieben, ist es nicht unwahrscheinlich, dass sich der Aufbau eines Templates aus unterschiedlichen Gründen ändern kann. Durch die Änderung eines Templates, müssen alle Softwareprojekte, die aus diesem Template entstanden sind, aktualisiert werden. Nicht

nur die Extension „Project Template“, sondern keine andere Erweiterung für VSCode bietet derzeit eine Funktionalität an, die dieses Problem lösen würde.

2.4.3 CLion IDE

CLion ist eine plattformübergreifende IDE von JetBrains für C/C++. Anders als Eclipse (Abschnitt 2.4.1) und VSCode (Abschnitt 2.4.2) ist CLion eine kostenpflichtige IDE mit einer kostenlosen 30-tägigen Testphase [12].

3. Lösungsansatz

Bei der Projekterstellung führt der Projekt-Konfigurator vorgegebene Konfigurationen an dem Projekt-Template aus. Die Konfigurationsmaßnahmen werden von dem Softwareentwickler selbst bestimmt. Durch die Konfigurationen entsteht am Ende ein neues einzigartiges Projekt.

Bei der Projektaktualisierung prüft der Projekt-Konfigurator die Version eines vorgegebenen Softwareprojekts verglichen mit der Version des Projekt-Templates. Wenn die Version des Projekt-Templates neuer ist, teilt der Projekt-Konfigurator dem bedienenden Softwareentwickler mit welchem Ordner und Dateien des Projekts aktualisierbar sind.

Das erste Problem löst der Projekt-Konfigurator, indem er den Entwicklern in der GUI ein Formular zu Verfügung stellt. In dieses Formular müssen die Mitarbeiter alle Metadaten des zu erstellenden Softwareprojekts eintragen. Auf Basis der eingegebenen Daten, erstellt der Projekt-Konfigurator ein neues konfiguriertes Projekt.

Das bedeutet, dass die Entwickler die Daten des neuen Softwareprojektes nicht mehr manuell ändern müssen, sondern alles automatisiert erstellt und angepasst wird. Alle notwendigen Änderungen/Anpassungen werden durch das Werkzeug gemacht und müssen nicht umständlich per Hand erfolgen. Dadurch werden die Projekte effizient und fehlerfrei erstellt.

Das zweite Problem löst der Projekt-Konfigurator, indem er die Template-version eines ausgewählten Softwareprojekts überprüft und daraufhin selbst entscheidet, ob das Projekt auf den neusten Template-Stand beinhaltet oder nicht. Wenn das Softwareprojekt nicht auf dem neusten Template-Stand ist, gibt der Projekt-Konfigurator den Entwicklern Möglichkeiten vor, um das Projekt zu aktualisieren. Dieser Mechanismus ist der Hauptteil dieser Abschlussarbeit.

4. Softwareentwurf

4.1 GUI Sketches

4.2 Architektur

Projekt Erstellung

Projekt Aktualisierung

Erweiterung der Projekt Module

Projekt Template Aktualisierung

4.3 Extrafunktionalitäten

Suchungsmechanismus von bestehenden C++ Projekten

Run utilities

5. Implementierung und Evaluierung

Internes Projekt Template

GUI Einstellungen

Projekt Erstellung

Suchungsmechanismus von bestehenden C++ Projekten

Erweiterung der Projekt Module

Projekt Template Aktualiserung

Run utilities

6. Zusammenfassung

6.1 Zusammenfassung

6.2 Das Problem

6.3 Mögliche Erweiterungspunkte

Literatur

[1] E. Wolff, *Continuous delivery: Der pragmatische Einstieg,* 2. Aufl. Heidelberg, Germany: dpunkt.verlag, 2016. [Online]. Verfügbar unter: https://ebookcentral.proquest.com/lib/gbv/detail.action?docID=4471176

[2] M. Hüttermann, *DevOps for developers*. New York, NY: Apress, 2012. [Online]. Verfügbar unter: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=1156065

[3] R. Alt, *Innovationsorientiertes IT-Management mit DevOps: IT im Zeitalter von Digitalisierung und Software-defined Business*. Wiesbaden: Springer Gabler, 2017.

[4] J. Rossberg, *Agile Project Management with Azure DevOps: Concepts, Templates, and Metrics*. Berkeley, CA: Apress L. P, 2019. [Online]. Verfügbar unter: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=5771167

[5] S. Augsten, „Was ist ein Build?“, *Dev-Insider*, 27. Apr. 2018, 2018. [Online]. Verfügbar unter: https://www.dev-insider.de/was-ist-ein-build-a-702737/. Zugriff am: 24. November 2021.114Z.

[6] P. D. Crutcher, N. K. Singh und P. Tiegs, *Essential Computer Science: A Programmer’s Guide to Foundational Concepts,* 1. Aufl. Berkeley, CA: Apress; Imprint Apress, 2021.

[7] J. L. Zuckarelli, *Programmieren lernen mit Python und JavaScript: Eine praxisorientierte Einführung für Einsteiger,* 1. Aufl. Wiesbaden: Springer Fachmedien Wiesbaden; Imprint Springer Vieweg, 2021.

[8] *Axel Bruns: Die Geschichte des Computers - ebook - neobooks.* [Online]. Verfügbar unter: https://link.springer.com/content/pdf/10.1007%2F978-1-4842-6901-5.pdf (Zugriff am: 25. November 2021.268Z).

[9] A. Del Sole, *Visual Studio Code Distilled: Evolved Code Editing for Windows, macOS, and Linux,* 2. Aufl. Berkeley, CA: Apress; Imprint Apress, 2021.

[10] *Visual Studio Code Tips and Tricks.* [Online]. Verfügbar unter: https://code.visualstudio.com/docs/getstarted/tips-and-tricks#_files-and-folders (Zugriff am: 29. November 2021.615Z).

[11] *Project Templates - Visual Studio Marketplace.* [Online]. Verfügbar unter: https://marketplace.visualstudio.com/items?itemName=cantonios.project-templates (Zugriff am: 29. November 2021.523Z).

[12] JetBrains, *Intelligente Programmierunterstützung und Codeanalyse – Features | CLion.* [Online]. Verfügbar unter: https://www.jetbrains.com/de-de/clion/features/ (Zugriff am: 29. November 2021.450Z).

Anhang