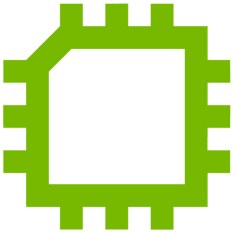
Hochschule für Technik und Wirtschaft

Sommersemester 2018

Fachbereich 1



Fortgeschrittene Algorithmen und Programmierung

Computer Engineering
Sommersemester 2018

Pflichtenheft

C++ Projekt

Gruppe: 4

Experimentatoren: Fabian Rod [s0546837]

Sören Größer [s0554662]

Ahmed Jenbaz [s0564915]

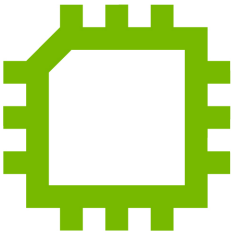
Dozent: Carsten Thomas

Berlin, 15. Juli 2018

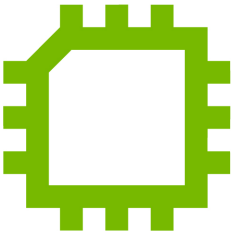
Hochschule für Technik und Wirtschaft

Sommersemester 2018

Fachbereich 1



Inhaltsverzeichnis



Abbildungsverzeichnis

Vorwort an den Dozenten

Setzt man sich mit dem Kontext des Pflichtenheftes im Allgemeinen auseinander, findet man im Internet eine Fülle an Informationen aus verschiedenen Bereichen der Produktentwicklung. Die Spezifizierung auf Softwareentwicklung grenzt die Gliederung weitestgehend ein. Im Folgenden, nutzen und beziehen wir uns explizit auf die praxisrelevanten Gliederungsvorschläge von Helmut Balzert aus seinem Buch „Lehrbuch der Software – Technik“. Des Weiteren gehen wir bei den parallel ablaufenden Fertigungsschritten davon aus, dass der Fertigungsschritt „Anbauteile bereitstellen“ bezüglich seiner zeitlichen Realisierung erst nach dem „Gehäuse entgraten“ ablaufen kann. Dementsprechend parallel zum Fertigungsschritt „Gehäuse lackieren“ verläuft und nicht bereits am Anfang der Fertigungsstrecke vorgefertigt werden kann. In einem sinnvollen Kontext könnte dies an Lagerplatz Kosten liegen.
Der Exzeption-Mechanismus wurde eingesetzt um schwerwiegende Fehler abzufangen. Fehler in der Eingabe jedoch wurden mit sinnvoller Überprüfung abgefragt und dem Benutzer die Möglichkeit gegeben seine Eingabe zu wiederholen

Kontext für den Projektleiter

Der Auftraggeber entwirft, fertigt und vertreibt elektronische Geräte. Die Geräte werden in unterschiedlichen Stückzahlen (Losgrößen) gefertigt. Die zur Fertigung eines Geräts nötigen Fertigungsschritte sind in einem Fertigungsplan beschrieben (siehe Abbildung 1). Einzelne Fertigungsschritte laufen parallel ab.

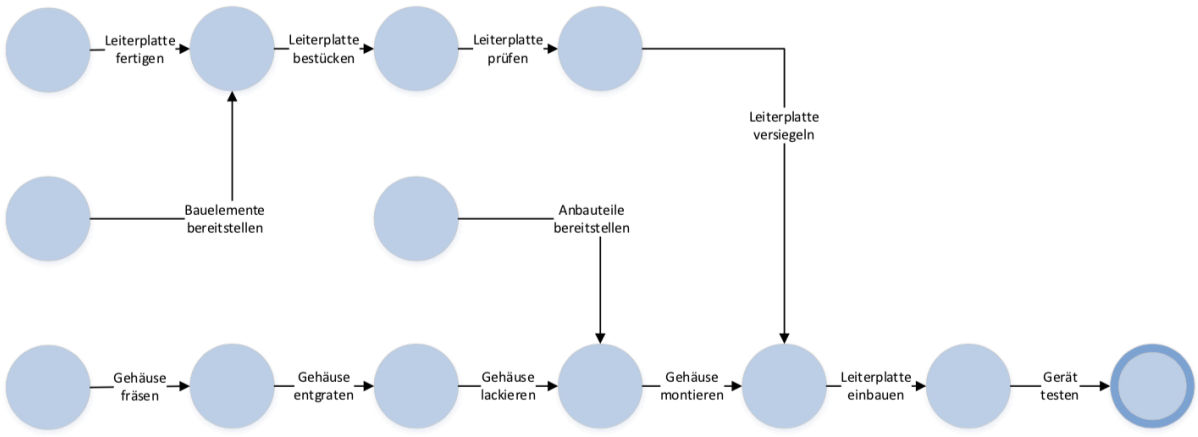
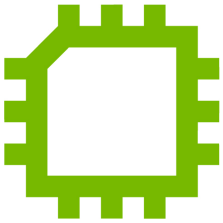


Abbildung 1 Fertigungsplan für Gerät A

Für die Fertigung kauft die Firma Dienstleistungen von Unterauftragnehmern ein. Die Preise der Dienstleistungen variieren nach Losgrößen (d.h. nach der Anzahl der beauftragten Werkstücke) und Durchlaufzeit für den einzelnen Fertigungsschritt. Die Liste der gültigen Dienstleistungsangebote ist in einer Textdatei gespeichert, die die Werte (Firmenname, Dienstleistung, Preis für Losgröße 1, Preis für Losgröße 10, Preis für Losgröße 100) als semikolon-separierte Liste enthält



Zielbestimmung

1. Musskriterien

Die Software soll eine Konsolenanwendung sein. Auf Grundlage des Fertigungsplans (siehe Abbildung 1) und der Dienstleistungsangebote muss es einen optimalen Fertigungsablauf erstellen. Sie soll Textdokumente mit den Informationen einlesen und sinnvoll verarbeiten. Der Benutzer soll sich die Dienstleistungen nach Name oder Service sortieren lassen, ausgeben und in einer eigens gewählten Datei abspeichern können.

2. Wunschkriterien

Alle möglichen Fehlersituationen sollen abgefangen werden und entsprechend darauf reagiert werden. Beachtung aller Programmierrichtlinien.

3. Abgrenzungskriterien

Die Software erkennt nicht den als Grafik abgebildete Graphen und implementiert ihn automatisch. Auch Fehler in der Struktur der Daten im Textdokument werden nicht behoben. Es gibt keine grafische Ausgabe oder GUI. Es gibt keine besonderen Anforderungen an den abgespeicherten Fertigungsplan. Es können nicht mehrere Textdateien gleichzeitig verarbeitet werden.

Produkteinsatz

1. Anwendungsbereiche

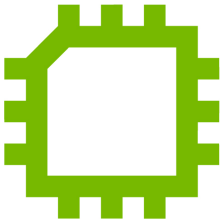
Jede Firma mit Fremddienstleitungen kann mit der Software Ihre Fertigungsstrecke planen, optimieren und abspeichern. Das Programm richtet sich vor allem an Unternehmen, welche auf der Suche nach einer Lösung sind, möglichst hohe Kosten zu vermeiden oder aber in einer möglichst kurzen Zeit viele Produkte zu entwickeln. Dabei dürfen sie eigene Firmen auswählen, die in der Software verglichen werden sollen.

2. Zielgruppen

Sie dient dem Produktionsleiter beziehungsweise Abteilungsleiter zur Planung des Einkaufs, der zeitlichen Abfolge in Hinblick auf verschiedene Optimierungsgrundlagen

3. Betriebsbedingungen

Die Software ist ein reines Konsolenprogramm und wird nicht durch jegliche zeitlichen oder physikalischen Umgebungsbedingungen bedingt.



Produktumgebung

1. Software

Zur Ausführung bedarf als Betriebssystem Windows oder nach entsprechender Kompilierung ein Linux System.

2. Hardware

Das Programm läuft auf jedem Windows Clientrechner und hat keine spezifischen Hardwareanforderungen. Ein reiner Office Rechner genügt.

3. Produktschnittstellen

Weitere Software ist nicht geplant.

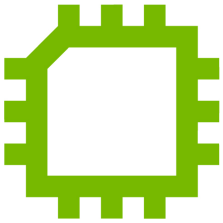
Produktfunktionen

Benutzerspezifisch

1. Namen der zu öffnenden Datei frei eingeben
2. Ansicht der Angebote als Tabelle mit [Name der Firma, Fertigungsschritt, Kosten und Zeitbedarf]
3. Sortierung der Angebote nach Firmennamen auswählen
4. Sortierung der Angebote nach Fertigungsprozess auswählen
5. Auswahl der Optimierung nach günstigstem Preis
6. Auswahl der Optimierung nach kürzester Dauer
7. Möglichkeit zur Anzeige des Fertigungsablaufs als Tabelle [Fertigungsablauf nach Fertigungsschritte, Auftragsumfang, Gesamtkosten, Gesamtzeitbedarf]
8. Möglichkeit zum Speichern des Fertigungsablaufs mit freier Namenswahl als Tabelle [siehe Ausgabe Fertigungsablauf]
9. Auswahl zum Beenden des Programms mit Abfrage zur Speicherung

Programmspezifisch (: nicht Pflichtenheft spezifisch)

10. Bereitstellung eines dynamischen Menüs.
11. Vom Benutzer gewählte Daten aus einer Datei öffnen und laden
12. Daten zur Verarbeitung aufbereiten
13. Daten zur tabellarischen Ausgabe konvertieren
14. Daten alphabetisch nach Firmenname sortieren und ausgeben
15. Daten alphabetisch nach Fertigungsprozess sortieren und ausgeben
16. Optimierten Fertigungsplan nach günstigstem Preis erstellen (Djikstra)
17. Optimierten Fertigungsplan nach geringster Zeit erstellen (Djikstra)
18. Fertigungsplan zur Ausgabe und Speicherung konvertieren
19. Ausgabe des erstellten Fertigungsablaufs
20. Speichern des Fertigungsablaufs in einem Textdokument
21. Überprüfung der Benutzereingaben
22. Reaktionen auf Fehlersituationen



Produktdaten

Keine spezifischen Anforderungen zur Datenerhaltung. Reine Speicherung einer Textdatei auf einem Speichermedium.

Produktleistungen

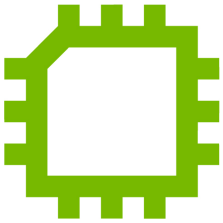
Keine spezifischen Informationen über maximale Antwortzeiten, maximaler Datenumfang gegeben oder Weiteres gegeben.

Benutzeroberfläche

1. Bildschirmlayout
   - Bildschirmausgabe der Daten in Tabellenform
2. Maskenlayout
   - Keine vorher definierten Anforderungen.
3. Drucklayout
   - Keine vorher definierten Anforderungen.
4. Speicherlayout
   - Tabelle mit Fertigungsablauf nach Fertigungsschritten
   - Gesamtzeitbedarf & Gesamtkosten
   - Auftragsumfang
5. Tastaturbelegung
   - Keine vorher definierten Anforderungen.
6. Dialogstruktur
   - Benutzer wird ein Menü zur Funktionsauswahl bereitgestellt
   - Beim Beenden des Programms Abfrage nach Speicherung des Fertigunsablaufs – sofern erstellt

Qualitäts-Zielbestimmungen

- Beachtung der Vorgaben der Programmierrichtlinien
- Nutzung des Exzeption-Mechanismus bei Fehlerreaktion
- Nutzung des Dijkstra-Algorithmus zur Ermittlung des optimalen Fertigungsablaufs
- Benutzeroberfläche in übersichtlicher Form, mit wiederkehrenden Elemente zur Orientierung
- Bereitstellung und dynamische Anpassung des Auswahlmenüs
- Bildschirmausgabe in tabellarischer Form und Tabellenkopf



Globale Testszenarien und Testfälle

1. Testfall: Dateiname eingeben

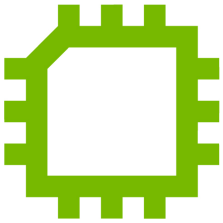
Art des Testes	• positiv
Vorbedingungen	• Programm gestartet • Menüauswahl -> Angebote einlesen
Testobjekt & Spezifikationen	• Dateimethode zum Einlesen nach Eingabe eines Namens
Eingabedaten	• Beispieldaten.txt
Handlungen	• Enter
Ergebnis/ Reaktion	• Dateiname wird eingelesen • Dateien wird geöffnet • Daten werden geladen • Datei wird geschlossen
Nachbedingungen	• Menüauswahl zum Einlesen von Angebotsdaten verschwindet • Weitere Menüauswahl erscheint
Prüfanweisungen	• Überprüfung des Testfalles im Debugger in Auswertung der entsprechenden Bezeichner (siehe Klassendiagram)

2. Testfall: Dateiname eingeben

Art des Testes	• negativ
Vorbedingungen	• Programm gestartet • Menüauswahl -> Angebote einlesen
Testobjekt & Spezifikationen	• Dateimethode zum Einlesen nach Eingabe eines Namens
Eingabedaten	• Beispiel.txt
Handlungen	• Enter
Ergebnis/ Reaktion	• Fehlerausgabe: Datei nicht vorhanden
Nachbedingungen	• Anforderung einen Dateinamen einzugeben
Prüfanweisungen	• Überprüfung im ausgeführten Programm

3. Testfall: Angebote ausgeben

Art des Testes	• positiv
Vorbedingungen	• Programm gestartet • Angebote wurden eingelesen
Testobjekt & Spezifikationen	• Dateimethode zur Ausgabe der konvertierten Angebotsdaten
Eingabedaten	• Menüauswahl -> Angebote ausgeben
Handlungen	• Enter
Ergebnis/ Reaktion	• Tabelle mit Tabellenkopf wird ausgegeben
Nachbedingungen	• Siehe -> Vorbedingung
Prüfanweisungen	• Überprüfung im ausgeführten Programm



4. Testfall: Angebote sortieren

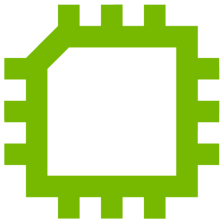
Art des Testes	• positiv
Vorbedingungen	• Programm gestartet • Angebote wurden eingelesen
Testobjekt & Spezifikationen	• Dateimethode zur Sortierung der Angebotsdaten
Eingabedaten	• Menüauswahl -> Angebote sortieren • Entsprechende Sortierung wählen
Handlungen	• Enter
Ergebnis/ Reaktion	• Angebote werden sortiert
Nachbedingungen	• Siehe -> Vorbedingung
Prüfanweisungen	• Überprüfung im ausgeführten Programm • Nach Sortierung -> Testfall Angebote ausführen

5. Testfall: Fertigungsablauf nach Optimierung erstellen

Art des Testes	• positiv
Vorbedingungen	• Programm gestartet • Angebote wurden eingelesen
Testobjekt & Spezifikationen	• Dateimethode zur Erstellung eines optimierten Fertigungsablaufs
Eingabedaten	• Menüauswahl -> Fertigungsablauf erstellen
Handlungen	• Stückzahl eingeben • Optimierung wählen
Ergebnis/ Reaktion	• Fertigungsablauf wird erstellt • Loßgröße entspricht der Stückzahl • Fertigungsablauf ist nach Bedingung optimiert
Nachbedingungen	• Menüauswahl wird erweitert
Prüfanweisungen	• Überprüfung im ausgeführten Programm • Nach Testfall Fertigungsablauf ausgeben lassen und Daten auf Richtigkeit überprüfen

6. Testfall: Fertigungsablauf ausgeben

Art des Testes	• positiv
Vorbedingungen	• Programm gestartet • Angebote wurden eingelesen • Fertigungsablauf wurde erstellt
Testobjekt & Spezifikationen	• Dateimethode zur Ausgabe des konvertierten Fertigungsablaufs
Eingabedaten	• Menüauswahl -> Fertigungsablauf ausgeben
Handlungen	• Enter
Ergebnis/ Reaktion	• Tabelle mit Tabellenkopf wird ausgegeben • Kosten, Umfang werden ausgegeben
Nachbedingungen	• Siehe -> Vorbedingung
Prüfanweisungen	• Überprüfung im ausgeführten Programm



7. Testfall: Fertigungsablauf speichern

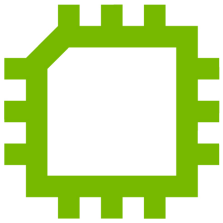
Art des Testes	• positiv
Vorbedingungen	• Programm gestartet • Angebote wurden eingelesen • Fertigungsablauf wurde erstellt
Testobjekt & Spezifikationen	• Dateimethode zur Speicherung des konvertierten Fertigungsablaufs
Eingabedaten	• Menüauswahl -> Speichern
Handlungen	• Dateiname eingeben
Ergebnis/ Reaktion	• Tabelle mit Tabellenkopf werden gespeichert • Kosten, Umfang werden gespeichert
Nachbedingungen	• Siehe -> Vorbedingung
Prüfanweisungen	• Überprüfung im ausgeführten Programm • Abgespeicherte Datei öffnen und Daten überprüfen

8. Testfall: Fertigungsablauf speichern

Art des Testes	• negativ
Vorbedingungen	• Programm gestartet • Angebote wurden eingelesen • Fertigungsablauf wurde erstellt • Kein Zugriffsrecht auf Ordner herstellen
Testobjekt & Spezifikationen	• Dateimethode zur Speicherung des konvertierten Fertigungsablaufs
Eingabedaten	• Menüauswahl -> Speichern
Handlungen	• Dateiname eingeben
Ergebnis/ Reaktion	• Fehlerausgabe -> Datei konnte nicht erstellt werden
Nachbedingungen	• Anforderung einen Dateinamen einzugeben
Prüfanweisungen	• Überprüfung im ausgeführten Programm • Abgespeicherte Datei öffnen und Daten überprüfen

9. Testfall: Fertigungsablauf speichern 2

Art des Testes	• negativ
Vorbedingungen	• Programm gestartet • Angebote wurden eingelesen • Fertigungsablauf wurde erstellt • Dateien erstellen mit bekannten Namen
Testobjekt & Spezifikationen	• Dateimethode zur Speicherung des konvertierten Fertigungsablaufs
Eingabedaten	• Menüauswahl -> Speichern
Handlungen	• Bekannten Dateiname eingeben
Ergebnis/ Reaktion	• Fehlerausgabe -> Datei bereits vorhanden
Nachbedingungen	• Anforderung einen Dateinamen einzugeben oder Überschreiben
Prüfanweisungen	• Überprüfung im ausgeführten Programm • Abgespeicherte Datei öffnen und Daten überprüfen



10. Testfall: Programm beenden

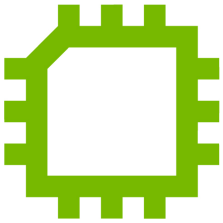
Art des Testes	• positiv
Vorbedingungen	• Programm gestartet • [Optional] Angebote wurden eingelesen
Testobjekt & Spezifikationen	• Dateimethode zum Beenden des Programms
Eingabedaten	• Menüauswahl -> Programm schließen
Handlungen	• Enter
Ergebnis/ Reaktion	• Speicher wird freigegeben • Programm wird geschlossen
Nachbedingungen	• Konsolenprogramm geschlossen
Prüfanweisungen	• Überprüfung im ausgeführten Programm

11. Testfall: Programm beenden 2

Art des Testes	• positiv
Vorbedingungen	• Programm gestartet • Angebote wurden eingelesen • Fertigungsablauf wurde erstellt
Testobjekt & Spezifikationen	• Dateimethode zum Beenden des Programms
Eingabedaten	• Menüauswahl -> Programm schließen
Handlungen	• Enter
Ergebnis/ Reaktion	• Abfrage ob gespeichert werden soll
Nachbedingungen	• Warten auf Benutzereingabe: „j“ „J“ „n“ „N“
Prüfanweisungen	• Überprüfung im ausgeführten Programm

12. Testfall: Angebote verändern

Art des Testes	• negativ
Vorbedingungen	• Struktur der Angebote verändern • Programm gestartet
Testobjekt & Spezifikationen	• Dateimethode zum Angebote erstellen
Eingabedaten	• Menüauswahl -> Angebote einlesen
Handlungen	• Beispieldaten.txt
Ergebnis/ Reaktion	• Fehlerausgabe -> Exzeption • Instanz von DataWorker wird nicht erstellt
Nachbedingungen	• Siehe Vorbedingung
Prüfanweisungen	• Überprüfung im ausgeführten Programm



Entwicklungsumgebung

1. Software

MacOS High Sierra, Windows 10, Microsoft Visual Studio, C++ 11, Parallels

2. Hardware

16 GB Arbeitsspeicher, Macbook Pro, 2,7 GHz Intel Core i7, Intel HD Graphics 3000 512MB, SSD 250 GB

3. Entwicklungsschnittstellen

Keine -> siehe Produktumgebung

Ergänzungen

Zur Nutzung und Integration ist stets zu beachten, dass diese Programm statisch an der Form des Graphen zum Fertigungsablaufs gebunden ist. Veränderungen in dem Produktionsablauf bedarf Nachbearbeitung des Konsolenprogramms oder ein Auftrag zum Update um ein Modul zum Verändern der Graphen Struktur. Genauso verhält es sich auch mit der Auswertung der Daten im Textdokument. Das Programm arbeitet nur den Anforderungen entsprechend bei 100% Realisierung der Datenstruktur in dem Textdokument. Später Updates könnten die Integrierung einer Datenbank sein oder die Reparatur bei fehlerhaften Datensätzen unter Zuhilfenahme eines Benutzers.

Projekt Fazit / Lessons Learned

“C macht es einfach, sich selbst ins Bein zu Schießen; C++ erschwert es, aber wenn es dir gelingt, bläst es dir das ganze Bein weg” Bjarne Stroustrup

Die Planung und Durchführung des Projektes wurde von uns in Etappen realisiert. Die Anforderungen wurden in Methoden runtergebrochen und je nach Möglichkeit in Klassen mit ähnlichem „Interesse“, gleicher Datennutzung integriert. Ob einfache Ausgabe oder komplexe Erstellung des Fertigungsprozesses, 60% der Zeit beschäftigten wir uns mit Fehlersuche und Fehlerlösung. Beim Zusammenfügen der Klassen und Ihre Methoden in ein Gesamtkonstrukt mussten sinnvolle Schnittstellen geschaffen werden und es ergaben sich die Denkfehler in der Planung des Softwareprojektes.
Letzter Schritt war das Abfangen von Fehler und das Testing mit Optimierung der Ein/ und Ausgaben. Das Klassendiagramm von der Vorüberlegung wurde überarbeitet.

Jedoch muss man eingestehen, dass Programme nie 100% fertig sind und sich bei jeder neuen Betrachtung andere oder teilweise bessere Wege eröffnen.
In diesem Sinne ist das beste Ergebnis, jetzt schon zu wissen was man nächstes Mal besser machen könnte um während der Implementierung seltener sein Bein zu verlieren.