

Software Technik Projekt: CrystalDiskInfo

Computer Engineering, Wintersemester 2019/2020

Philipp Horländer, 566045
Konrad Münch, 565929

Dozent: Prof. Dr. Baar

Berlin, 23. Dezember 2019

Inhaltsverzeichnis

1	Einführung	3
1.1	Aufgabenstellung	3
1.2	Beschreibung der Software	3
2	Aufbau des Programmes	3
2.1	Programmiersprache(n)	3
2.2	Übersicht des Quellcodes	4
2.3	Ordnerstruktur	5
3	Modellierung	7
3.1	Use-Case-Modell	7
3.2	Neue Klasse	7
3.2.1	anzeigenDatenträgerInformation	7
3.2.2	einstellenRateAktualisierung	7
3.2.3	anzeigenGraph	7
3.2.4	einstellenBenachrichtigung	8
3.2.5	benachrichtigenNutzer	8
3.2.6	auswahlTheme	8
3.2.7	auswahlSprache	8
3.3	Domänenmodell	9
4	Geforderte Änderungen	9
5	Modellierung nach den Änderungen	10
5.1	Use-Case-Modell	10
5.1.1	anzeigenDiagramm	10
5.2	Domänenmodell	10
5.3	Weitere Modelle	10
6	Beschreibung der Implementierung	10
7	Fazit	10

1 Einführung

1.1 Aufgabenstellung

Es gilt sich eine Open Source Software auszusuchen und diese zu "Reverse-Engineeren" und abzuändern. Dafür sollen verschiedenen Modelle der Software angefertigt werden, wie Use-Case-, Domänen- und ggf. weitere Modelle. Das ganze soll mit einer Belegarbeit dokumentiert werden.

1.2 Beschreibung der Software

"CrystalDiskInfo" ist ein nützliches Open-Source Programm, für die Bewertung und Überwachung von internen als auch externen HDDs und SSDs. Mit Hilfe der S.M.A.R.T. - Technologie (Self-Monitoring, Analysis, Reporting Technology) werden detaillierte Daten, wie zum Beispiel Seriennummer, Temperatur, Lese- und Schreibzyklen, Fehlerrate beim Lesen und Schreiben, Speicher und Puffergrößen, als auch die gesamten Betriebsstunden der Festplatten, aus den Festplatten-Controllern ausgelesen. Diese sind auf dem Controller-Chip gespeichert und werden in einem einstellbaren Intervall von der Software ausgelesen. Die entsprechenden Daten werden auf dem jeweiligen Host-System in Form von CSV-Dateien abgelegt.

Wie und welche dieser Daten angezeigt werden lässt sich einstellen. Beispielsweise lässt sich die Seriennummer ausblenden um auf Screenshots nicht aufzutauchen.

Auch lässt sich ein Diagramm über ausgewählte, gespeicherte Daten und Datenträger in Abhängigkeit der Zeit anzeigen.

Eine weitere nützliche Funktion für Administratoren ist es sich bei zu hohen Temperaturen oder einer geringen Lebenszeit der Festplatten benachrichtigen zu lassen. Die geschieht mit einem Warnton, als auch via E-Mail.

Wenn man möchte lassen sich auch Farbschemata der Buttons und Anzeigen ändern.

2 Aufbau des Programmes

2.1 Programmiersprache(n)

Crystal Disk Info wurde von Noriyuki Miyazaki in der Sprache C++ für Microsoft Windows Betriebssysteme programmiert. Zur Darstellung der grafischen Benutzeroberfläche wurde die Microsoft Foundation Classes kurz MFC¹ verwendet. Des Weiteren wurde für die Realisierung der Darstellung der Graphen, die JAVA -Bibliothek "FLOT" verwendet.

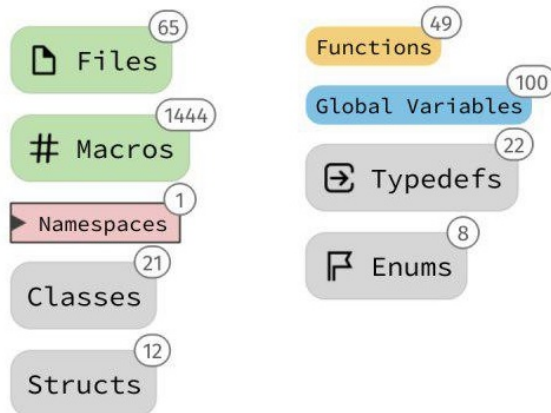
Zudem werden noch HTML Dateien genutzt welche zur Darstellung der Graphen dienen. Der C++ Code hat in diesem Programm jedoch mit 94,7% den größten Anteil, diese Auswertung ergab die Diagnosefunktion von GitHub.

¹MFC ist eine von Microsoft seit 1992 entwickelte Bibliothek zur Erstellung von Windows Desktop Anwendungen.

2.2 Übersicht des Quellcodes

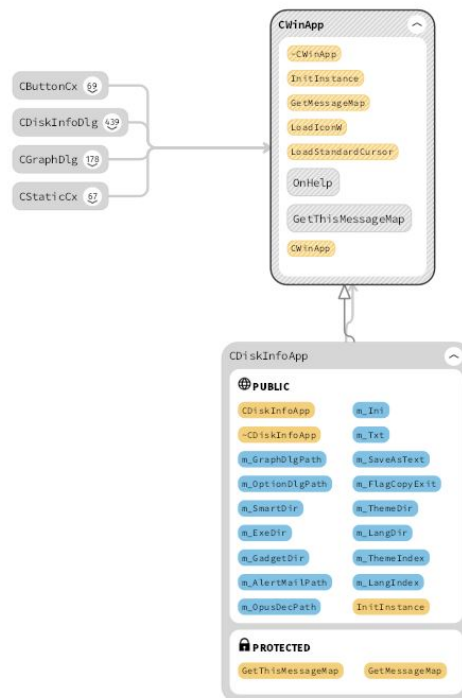
Zum Anfang des Projekts, wurden die Analyse-Tools Sourcetrail und Doxygen von uns genutzt, um eine Übersicht von dem Umfang und Beschaffenheit des Quellcodes zu bekommen. Doxygen erwies sich als zu mächtig und umfangreich für unsere Zwecke. Die erzeugte RTF-File umfasste zwar jegliche Art von Diagrammen, jedoch war die Datei mit über 32500 Wörtern nicht praktikabel einsetzbar.

Sourcetrail erwies sich hierbei als komfortabler und übersichtlicher. Wir konnten mit Hilfe dieses Tools einfacher den Aufbau und die Beschaffenheit der Klassen und Funktionen erfassen.



Insgesamt umfasst der Code 34207 Zeilen an Code in 65 Dateien. Wie unter Punkt 2.1 schon erwähnt wurde MFC für den Aufbau des Programmes als Grundlage genutzt.

Durch die Analyse mit Sourcetrail konnten wir uns einen Überblick, über die wichtigste Ursprungs-klasse von der alle GUI Elemente abhängen verschaffen. Aus der Klasse CWinApp werden alle erstellten Klassen wie Dialogfenster und Fensterfunktionen abgeleitet.

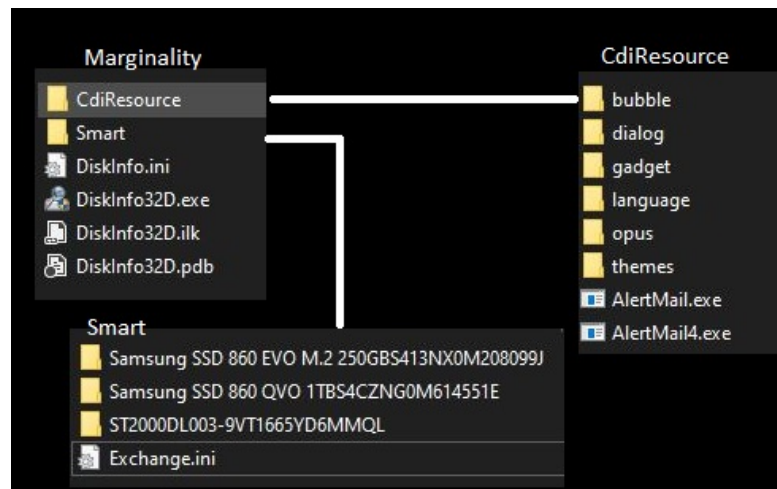


Aufgrund der Größe des Programms, beschränken wir uns anfänglich nur auf die Klasse CWinApp und der des Hauptdialogs CDiskInfoApp. Letztere ist die von dem Autor erstellte Elternklasse aus der sich alle weiteren Dialoge des GUI ableiten.

2.3 Ordnerstruktur

Das Programm erzeugt als Root-Verzeichnis den Ordner "Marginality". Dieser enthält die EXE-Datei und die Ordner CdiResource sowie Smart. CdiResource enthält wichtige externe Dateien wie Sprachdateien, Themen, die JAVA-Skripte für den Graphen sowie das Programm AlertMail welches genutzt wird um Warnungen via E-Mail zu versenden.

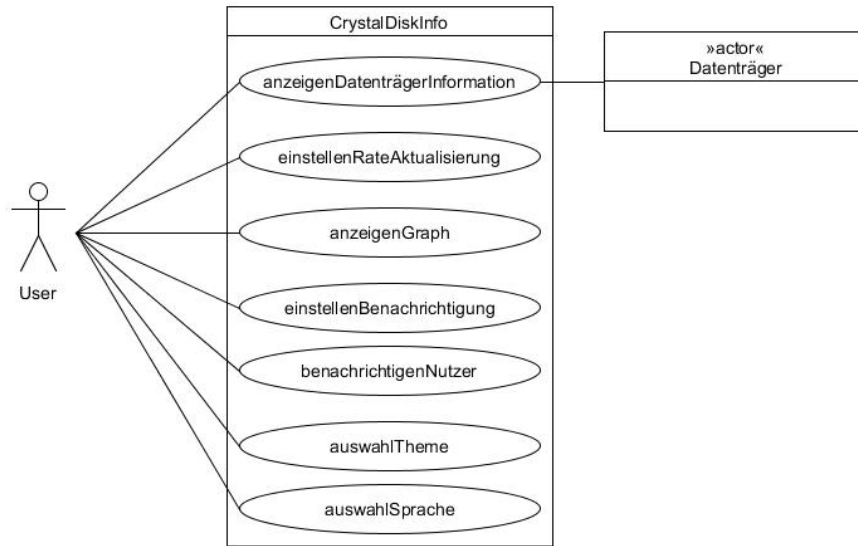
Der Ordner Smart enthält die durch Crystal Disk Info ermittelten Werte der verbauten Festplatten in dem Host-System. Diese Daten werden von dem GUI und der JAVA-Bibliothek FLOT genutzt, um die Werte im Frontend des Programmes darzustellen.



Ohne diesen Aufbau ist das Programm nicht vollkommen lauffähig. Die GIT-Repository von Crystal Disk Info, enthält nicht den vollständigen Ordneraufbau. Es fehlt der Ordner CdiResource, dies führt nach einem Compiler-Durchlauf zu Fehlern die durch Exceptions erzeugt werden. Jedoch werden diese Verzeichnisse bei der Installation mittels Setup erstellt.

3 Modellierung

3.1 Use-Case-Modell



3.1.1 anzeigenDatenträgerInformation

1. Das Programm ermittelt die verbauten Festplatten.
2. Das Programm liest SMART-Daten jedes Festplatten-Controllers aus.
3. Das Programm zeigt die ermittelten Werte im GUI an.

3.1.2 einstellenRateAktualisierung

1. Der Nutzer wählt ein Zeitintervall
2. Das Programm liest und speichert die Werte im gewählten Intervall.

3.1.3 anzeigenGraph

1. Der Nutzer wählt die Funktion Graph anzeigen.
 2. Das Programm öffnet ein neues Fenster.
 3. Das Programm zeigt einen Graph mit aus Datenreihen an.
-
- 2a. Der Nutzer wählt eine andere Datenreihe aus.
 - 2a.1. Das Programm zeigt die andere Datenreihe an.

- 2b. Der Nutzer wählt Festplatten an/ab.
- 2b.1. Das Programm zeigt die gewählten Festplatten-Datenreihen an.

3.1.4 einstellenBenachrichtigung

- 1. Der Nutzer wählt einstellen einer Benachrichtigung aus.
- 2. Das Programm wartet Eingabe der benötigten Daten des Benutzers.
- 3. Der Nutzer gibt nötigen Daten ein und bestätigt.

3.1.5 benachrichtigenNutzer

- 1. Das Programm stellt einen kritischen Wert von Temperatur oder Lebensdauer einer Festplatte fest.
- 2. Das Programm erstellt eine E-Mail und versendet diese an die eingestellte Adresse.

- 2a Der Nutzer schließt das Fenster.
- 2a.1 Das Use-Case endet erfolglos.

3.1.6 auswahlTheme

- 1. Der Nutzer wählt Theme ändern aus.
- 2. Das Programm zeigt dem Nutzer eine Auswahl an.
- 3. Der Nutzer wählt zwischen Verschiedenen Farbdarstellungen.
- 4. Das Programm lädt die ausgewählte Darstellung.

- (2-3)a Der Nutzer bricht die Auswahl ab.
- (2-3)a.1 Das Use-Case endet erfolglos.

- 4||a. Das Programm stellt fest das benötigte Dateien fehlen:
- 4||a.1. Das Programm zeigt eine Fehlermeldung an.
- 4||a.2. Das Use-Case endet erfolglos an.

3.1.7 auswahlSprache

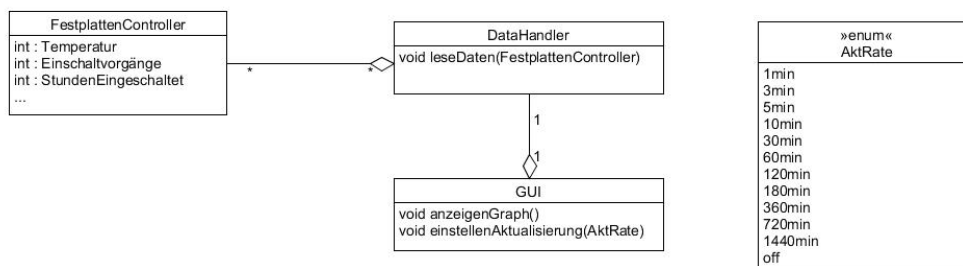
- 1. Der Nutzer wählt Sprache ändern aus.
- 2. Das Programm zeigt dem Nutzer eine Auswahl an.
- 3. Der Nutzer wählt zwischen Verschiedenen Sprachen.
- 4. Das Programm lädt die ausgewählte Sprache.

- (2-3)a Der Nutzer bricht die Auswahl ab.
- (2-3)a.1 Das Use-Case endet erfolglos.

- 4||a. Das Programm stellt fest das benötigte Dateien fehlen:
- 4||a.1. Das Programm zeigt eine Fehlermeldung an.
- 4||a.2. Das Use-Case endet erfolglos an.

3.2 Domänenmodell

Das hier gezeigte Domänenmodell stellt die Funktion vor der Änderung des Programmes da. Zur Übersicht wurde nur ein Ausschnitt des gesamten Programmes betrachtet.



4 Geforderte Änderungen

Die Änderung von CrystalDiskInfo sollte die Darstellung eines neuen Diagrammtyps ermöglichen. Das sogenannte "Bubble Diagramm" sollte in die Auswahlfunktion der Diagramme implementiert werden.

Ein "Bubble Diagramm" soll die Korrelation von zwei Datenreihen darstellen, in Form von Punkten auf einem Koordinatensystem. Diese werden durch eine dritte Abhängigkeit ergänzt welche durch die Größe der Punkte (Radius) dargestellt wird.

Ein besprochene Umsetzungsidee war die Darstellung von Temperatur auf der X-Achse, zu geschriebenen Daten auf der Y-Achse, welche nach einer Zeitperiode ihre Größe verändern. Zudem sollten noch andere Daten mit einander verglichen werden.

Nach Absprache mit dem Dozenten Herrn Prof. Dr. Baar, wurde die Art der Implementierung vereinfacht. Grund hierfür war die Komplexität des Aufbaus der Darstellungsfunktion im Programm. Eine weitere Änderung sollte darin bestehen die Aktualisierungszeit der Datenpunkte auf eine Sekunde hinzuzufügen.

5 Modellierung nach den Änderungen

5.1 Use-Case-Modell

Änderungen gab es nur an einem Use-Case.

5.1.1 anzeigenDiagramm

1. Das Programm generiert eine HTML-Datei und öffnet diese im Browser.

5.2 Neue Klasse

5.3 Domänenmodell

5.4 Weitere Modelle

6 Beschreibung der Implementierung

Um das Bubble-Chart darzustellen wurde, wie im Originalprogramm eine JavaScript Bibliothek genutzt. Es wurde sich allerdings für eine andere, modernere Bibliothek entschieden, da die in CrystalDiskInfo verwendete schon sehr alt war und der Anforderungen ein Bubble-Chart darzustellen nicht erfüllen konnte. Deshalb wurde die JavaScript Bibliothek ChartsBundle.js verwendet, welche auch OpenSource auf GitHub.com zur Verfügung steht.

Mithilfe dieser Bibliothek wurde dann eine HTML mit JavaScript erstellt. Um die Verarbeitung in C++ zu vereinfachen wurde das Skript in zwei Teile geteilt und in Text-Dateien abgelegt. Diese Dateien wurden dann in C++ mit den ausgelesenen Daten zu einer fertigen HTML/JS-Datei verknüpft und gespeichert. Die erstellte HTML Datei wird nun über einen Aufruf im Browser geöffnet.

7 Fazit

Schwierigkeiten traten im gesamten Verlauf zur genüge an. Zuerst war es recht schwer herauszufinden wie CrystalDiskInfo aus dem Source Code funktionsfähig zu bekommen ist. Nötig waren dafür Dateien von der fertigen Installation die dem Source Code nicht beilagen und auch nicht beim Compilieren erstellt wurden.

Als nächstes war es sehr schwer den Code zu verstehen, da es sich bei dem Entwickler von CrystalDiskInfo offensichtlich um einen sehr erfahrenen Programmierer handelt und das Programm entsprechend komplex war. Allerdings ist CrystalDiskInfo sehr spärlich kommentiert. Das machte das verstehen deutlich schwerer.

Die geforderte Änderung umzusetzen erwies sich als schwerer als erwartet.