Ostfalia Hochschule

EKdI-Projekt

Essay

Martin Krause 70478294

supervised by
Prof. Jensen

20. Januar 2022

1 Informatik ist ...

...„die Wissenschaft von den elektronischen Datenverarbeitungsanlagen und den Grundlagen ihrer Anwendung“[1], wenn man im Duden nachschlägt, oder „die Wissenschaft von der systematischen Darstellung, Speicherung, Verarbeitung und Übertragung von Informationen, wobei besonders die automatische Verarbeitung mit Digitalrechnern betrachtet wird“[2], falls man bei Wikipedia nachguckt.

Informatik ist mittlerweile eine sehr große Wissenschaft geworden und lässt sich eigentlich gar nicht durch so eine kurze Definition beschreiben. Viele denken immer, dass die Informatik immer nur mit Computern zu tun hat - was irgendwo auch stimmt - jedoch liegen die Anfänge der Informatik weit zurück. Schon vor tausenden Jahren haben die Menschen simple Rechenregeln bzw. Ablaufregeln erschaffen und damit die ersten Algorithmen kreiert. Man bedenke, dass der „Euklidische Algorithmus“bereits um 300 vor Christus von Euklid beschrieben wurde, um den größten gemeinsamen Teiler zweier natürlicher Zahlen zu finden[3].

Seit dem wuchs das Themengebiet wa wir heute als Informatik bezeichnen würde und es wurden mehr Rechenregeln aufgestellt oder schon simple Geräte für einfache Rechensysteme verwendet wie die jedem bekannten Rechenschieber[4], damit man sich Arbeit sparen konnte. Ende des 19. Jahrhunderts hat schließlich Hermann Hollerith im Rahmen einer Volkszählung in den USA eine elektronische Zählmaschine entwickelt, die die Volkszählung mithilfe von stanzbaren Lochkarten vereinfachte, sodass diese bereits nach einem halben Jahr abgeschlossen werden konnte[5]. Er gründete außerdem eine Firma, die später in die Firma IBM überging, welche quasi revolutionär für die Entwicklung der ersten Computer war.

Seit dem vergangenen Jahrhundert, hat sich die Informatik dann sehr weiterentwickelt. Aber zuerst ging das alles nur theoretisch von statten. Das Themengebiet der theoretischen Informatik wurde immer präsenter, bis es schließlich Alan Turing (1912-1954) während des zweiten Weltkrieges gelang den quasi ersten Computer zu entwickeln, der eigentlich dazu diente, die Enigma-Codes der deutschen Wehrmacht zu knacken. Diese Maschine galt zumindest als die erste, die turing-vollständig ist, das heißt, sie beschreibt eine universelle Programmierbarkeit.

Durch die ständige Entwicklung wurden dann auch recht schnell erste Computer gebaut, der größtenteils aus elektromagnetischen Bauteilen bestanden. Dazu beigetragen hat der deutsche Bauingenieur Konrad Zuse (1910-1955) (siehe „Grundlagen der Informatik“[6].
In den 1970er Jahren wurde bereits die erste Hochsprache unter den Programmiersprachen entwickelt, also abseits von Assembler Code, namens „C“, welche auch heute noch sehr oft für schnelle, eingebettete Systeme benutzt wird. Seit dem haben sich hunderte Programmiersprachen geformt, mit allen möglichen Paradigmen, wie z.B. funktional, oder objekt-orientiert[6].

Etwa im selben Zeitraum wurden dann auch Grafikkarten entwickelt, die anders als die Central Processing Unit eines Computers viel mehr Kerne besitzt und deshalb gerade die grafiklastigen Aufgaben besser bearbeiten kann, da viele Berechnungen parallel erfolgen können. Diese Technologie war natürlich auch für die Filmindustrie wertvoll und führte dazu, dass „Star Wars“der erste Film war, in dem wirklich gute computergenerierte Grafikeffekte vorkommen konnten.

Man sieht also dass die Informatik einen riesigen Teil unseres Lebens ausmacht, der gar nicht mehr wegzudenken ist. Schließlich ist heutzutage in jeder Waschmaschine und jedem noch so unscheinbaren Gerät ein Prozessor enthalten und ohne Internet würde in der heutigen Welt natürlich gar nichts mehr gehen. Man kann sich nur sehr schlecht vorstellen was passieren würde, wenn das Internet für nur einen Tag zusammenbrechen würde, aber die Folgen wären fatal, allein in der Infrastruktur, oder in der medizinischen Versorgung, Elektrische Werke und viele andere unscheinbare Systeme würden wahrscheinlich einen Totalausfall erleiden.

Es gibt sehr viele Möglichkeiten sich mit Themen aus Bereichen der theoretischen, technischen oder praktischen Informatik auseinanderzusetzen, auch wenn vieles davon wahrscheinlich Logik oder nur Mathematik ist, aber ich persönlich sehe die Informatik als eine große Kunstform an.

1.1 Quellenverzeichnis

[1] Duden, https://www.duden.de/rechtschreibung/Informatik, Stand: 18.01.2022.
[2] Wikipedia, https://de.wikipedia.org/wiki/Informatik, Stand: 18.01.2022.
[3] Wikipedia, https://de.wikipedia.org/wiki/Euklidischer_Algorithmus, Stand: 18.01.2022.
[4] Rechnerlexikon, http://rechnerlexikon.de/artikel/Geschichte_des_Rechenschiebers, Stand: 18.01.2022.
[5] Damals.de, https://www.wissenschaft.de/zeitpunkte/8-januar-lochkarten-fuer-die-volks
Stand: 19.01.2022.
[6] Herold, H.; Lurz, B.; Wohlrab, J.; Hopf, M.; „Grundlagen der Informatik “, 3. Auflage, Pearson-Verlag, 2017.

2 Ein erfolgreiches IT-Projekt ...

...ist gar nicht so einfach zu realisieren. Wenn ich mein theoretisches Wissen und meine eigene praktische Erfahrung eines Software-Team-Projektes von meiner Zeit an der Technischen Universität Braunschweig mit einfließen lasse, gehört viel dazu ein IT-Projekt erfolgreich abzuschließen.

Angefangen bei der Teambildung, die schon Schwierigkeiten machen kann, da eventuell einige Leute im Team weniger Erfahrung mit diesem Tool oder dieser Programmiersprache haben als andere, oder weil Streit ausbricht. Es braucht teilweise gute Teamleiter, die das gesamte Projekt steuern und unterstützen. Bei der Teambildung gilt es auch die Vier Phasen der Teambildung zu beachten und im Hinterkopf zu haben. Diese unterscheiden sich in „Forming“, „Storming“, „Norming“und „Performing“. Die erste Phase ist zur Orientierung der Teammitglieder gut und einzelne Rollen werden gebildet. Die zweite Phase führt oft zu Konflikten im Team welche oft Folgen für das Projekt haben können. Die dritte Phase dient eher zur Regelfindung im Team, um solche Konflikte bestenfalls zu verhindern und die letzte Phase ist die, in der das Team gut eingespielt effizient arbeiten kann[1].

Nach dem Erhalt des Auftrages für eine Software muss erstmal viel geplant werden bevor es überhaupt mit der Programmierung losgeht, da sonst direkt am Anfang schlechter Code entsteht, der am Ende überhaupt nicht das tut, was er soll. Solche Teams arbeiten für gewöhnlich nach bestimmten Strategien, wie z.B. dem V-Modell, dem Wasserfall-Modell, oder Scrum, die als sehr agile Strategie gilt. Bei Scrum arbeitet das Team in sich ständig wiederholenden Zyklen, welche sehr effizient ablaufen, weshalb das Team gute fachliche Kompetenzen aufweisen sollte und schon gut zusammenarbeiten können sollte[2].

Sobald man die Planung abgeschlossen hat und sämtliche Diagramme der „UML- Modellierung“entworfen hat, wie Aktivitäts- oder Zustandsdiagramme, oder auch Klassendiagramme[3], kann man mit der Programmierung anfangen. Dabei sollte allerdings auf bestimmte Coding-Richtlinien geachtet werden für Zeicheneinrückung oder Variablenbenennung. Es ist schon schwer in fremdem Code durchzusteigen, oder sogar teilweise in dem eigenen, deshalb muss auf eine gute Kommentierung und das Einhalten von gewissen Codingquidelines geachtet werden[4].

Ein ganz wichtiger Punkt eines solchen Projektes stellt auch das ausgiebige Testen dar. Das bedeutet man testet kleine Programmteile für sich und setzt diese Stücke immer mehr zusammen, sodass man in Integrationstests testet, wie und ob die Teile zusammenarbeiten, bist hin zum kompletten Programm. Dabei testet man am besten durch Whitebox-Tests, bei denen man die Bedingungen und den Funktionsinhalt kennt, und durch Blackbox-Tests, bei denen man keinen Einblick in Code hat und nur Randbedingungen und besondere Fälle von Variablen testet. Man testet also ohne die innere Struktur der Funktionen zu kennen das grundlegende Verhalten dieser. Dabei kann man

eigentlich auch nicht die Abwesenheit von Fehlern beweisen, aber man kann meistens welche finden und diese dann verbessern[5].

Und wenn man nach all dem dann das vermeintlich fertige Produkt dem Kunden vorstellt, kann es immer noch sein, dass dieser einige Stellen nicht gut findet und daraufhin nochmal sämtliche Codestücke angepasst werden müssen. Deswegen ist es wichtig von vornherein gut verständlichen und gut wartbaren Code zu schreiben, denn eine Funktion oder Klasse neu zu schreiben ist in den meisten Fällen einfacher, als den alten Code zu verbessern und würde viele IT-Projekte zu einem schnelleren und effizienteren Ende führen.

Es gab in der Geschichte schon viele gescheiterte Softwareprojekte, die unter anderem zu großen Fehlern in Programmen führten, welche sogar ziemlich berühmt wurden. Zum Beispiel musste die „Mariner-1-Rakete“kurz nach ihrem Start abgelenkt werden, weil ein Fehler sie auf eine falsche Flugbahn lenkte, oder eine tragische Fehlerberechnung für die Dosierung einer Bestrahlungsmaschine, die Anfang 2000 acht Menschen das Leben kostete[6].

Insgesamt sollte ein erfolgreiches IT-Projekt also ein gutes Team haben, mit guter Arbeitsmoral, kompetenten Teammitgliedern und einer guten Arbeitsstrategie, sodass durch eine gute, überlegte Planung auch funktionierender Code geschrieben werden kann, der möglichst schnell zu guten Testergebnissen führt und ein erfolgreiches Produkt liefert.
Wenn man all das beachtet, sollten solche verheerende Fehler bzw. Software-Krisen eigentlich nicht mehr vorkommen können und die Grundlage für ein erfolgreiches IT-Projekt ist geebnet.

2.1 Quellenverzeichnis

[1] Weblog, https://blog.seibert-media.net/blog/2011/06/24/team-building-prozesse-softw
Stand: 18.01.2022.
[2] AgileScrumGroup, https://agilescrumgroup.de/strategie-scrummen/, Stand: 18.01.2022.
[3] Herold, H.; Lurz, B.; Wohlrab, J.; Hopf, M.; „Grundlagen der Informatik “, 3. Auflage, Pearson-Verlag, 2017.
[4] embedded-software-engineering, https://www.embedded-software-engineering.de/coding-guidelines-sind-programmierrichtlinien-fluch-oder-segen-a-795748/, Stand: 19.01.2022.
[5] Software-Testing-Academy, https://www.software-testing.academy/black-box-test.html, Stand: 18.01.2022.
[6] entwickler.de, https://entwickler.de/programmierung/top-10-der-software-katastrophen, Stand: 18.01.2022.