SWT WS 2020/21
Gruppe 010
15. November 2020
Laura Koch, 406310
Marc Ludevid, 405401
Til Mohr, 405959

Aufgabenblatt 1

Andrés Montoya, 405409
Dobromir I. Panayotov, 407763
Fabian Grob, 409195
Lennart Holzenkamp, 407761
Simon Michau, 406133
Tim Luther, 410886

Aufgabe 1.1

a)

Tätigkeit	A	E	I	T	W
Benutzer der Software schulen				x	x
Qualitätssicherung des Pflichtenheftes prüfen				x	
Gesetzliche Rahmenbedingungen prüfen	x				
Konzept und Prototyp einer Benutzeroberfläche erstellen		x			
Entwicklerteam zusammenstellen	x	x			
Code eines Programmmoduls debuggen			x	x	x
Zwei Subsysteme verbinden und testen			x	x	
Termine und Kosten des Projektes planen	x				
Datenstrukturen festlegen		x	x		
Vorhandene Altlasten des Kunden analysieren	x				
Schnittstellen von Programmmodulen definieren		x			
Leistung der Entwickler bewerten und belohnen					
Software an neue Umgebung anpassen				x	x
Kunden eine Rechnung stellen					
Test-Eingabedaten für ein Programmmodul ermitteln				x	
Strukturmodell des gesamten Softwaresystems entwerfen		x			
Dokumentation des Projektablaufs bewerten und archivieren					
Nach bereits vorhandenen, wiederverwendbaren... ...Software-Bibliotheken suchen	x	x	x		
Performance-Prognose des Softwaresystems erstellen		x	x		
Programmcode kommentieren			x		

*A=Analyse; E=Entwurf; I=Implementierung; T=Test/Integration; W=Wartung

Begründungen:
Benutzer der Software schulen: Gehört zu Integration und Wartung, da ggf. nach jeder Implementierung neuer Funktionalitäten Schulungen erforderlich sein können.
Qualitätssicherung des Pflichtenheftes prüfen: Zum Test der Software, ob alles auch wie gewünscht funktioniert.
Gesetzliche Rahmenbedingungen prüfen: Der gesetzliche Rahmen sollte geklärt sein, bevor das Produkt entwickelt wird.
Konzept und Prototyp einer Benutzeroberfläche erstellen: Konzept und Prototyp müssen erstellt werden, nachdem in der Analyse bestimmt wurde welche Anforderungen erfüllt werden sollen, aber bevor die Implementierung startet, damit die Entwickler wissen was zu tun ist.
Entwicklerteam zusammenstellen: Das Entwicklerteam sollte an das Projekt angepasst werden, bevor dieses startet, aber nachdem geklärt wurde welche Qualifikationen erforderlich sind. Einzelne Teammitglieder können während der Entwicklung noch angepasst werden.
Code eines Programmmoduls debuggen: Kann immer passieren, wenn mit konkretem Code gearbeitet wird, vorausgesetzt ein Programm existiert bereits.
Zwei Subsysteme verbinden und testen: Verbindung in Implementation & Integration, Testen in Test. Abhängig, von wo die Systeme kommen und ob man sie auch selber entwickelt.
Termine und Kosten des Projektes planen: Sollte erledigt sein bevor das Projekt startet.
Datenstrukturen festlegen: Sollte im Entwurf festgelegt werden, damit die Entwickler sie implementieren können. Kann während der Implementierung noch angepasst werden.
Vorhandene Altlasten des Kunden analysieren: Wichtig zum Entwurf, sollte also in Analyse geschehen.

SWT WS 2020/21
Gruppe 010
15. November 2020
Laura Koch, 406310
Marc Ludevid, 405401
Til Mohr, 405959

Aufgabenblatt 1

Andrés Montoya, 405409
Dobromir I. Panayotov, 407763
Fabian Grob, 409195
Lennart Holzenkamp, 407761
Simon Michau, 406133
Tim Luther, 410886

Schnittstellen von Programmmodulen definieren: Wenn die Implementation in Modulen erfolgt, sollte vorher klar geregelt sein, wie die Module später miteinander arbeiten. Daher sollte das im Entwurf geschehen.
Leistung der Entwickler bewerten und belohnen: Gehört nicht wirklich zur Softwareentwicklung.
Software an neue Umgebung anpassen: Wartung der Software und Integration in neue Umgebung.
Kunden eine Rechnung stellen: Gehört nicht wirklich zur Softwareentwicklung, könnte aber durchaus in jedem Schritt geschehen.
Test-Eingabedaten für ein Programmmodul ermitteln: Können theoretisch immer ermittelt werden, aber am meisten während den Tests selber.
Strukturmodell des gesamten Softwaresystems entwerfen: Gehört klar zu dem Entwurf, sollte auf jeden Fall vor der Implementierung geschehen, und kann erst nach der vollständigen Analyse entworfen werden.
Dokumentation des Projektablaufes bewerten und archivieren: Da es sich um den Ablauf handelt, kann dies nur für die Entwickler selbst interessant sein, ist daher außerhalb dieses Entwicklungsmodells welches nur für ein einzelnenes Projekt ist.
Nach bereits vorhandenen, wiederverwendbaren Software-Bibliotheken suchen: Kann so früh beginnen wie in der späten Analyse bis hin zu der Implementation, um die Software zu entwerfen und auch implementieren.
Performance-Prognose des Softwaresystems erstellen: Sollte früh geschehen, damit man die Implementation auch darauf abrichten kann.
Programmcode kommentieren: In jeder Phase, in der man Code implementiert.

b)

Tätigkeit	A	GE	FE	I	MT	IT	ST	AT
Benutzer der Software schulen								
Qualitätssicherung des Pflichtenheftes prüfen								x
Gesetzliche Rahmenbedingungen prüfen	x							
Konzept und Prototyp einer Benutzeroberfläche erstellen		x	x					
Entwicklerteam zusammenstellen	x	x						
Code eines Programmmoduls debuggen				x	x	x	x	x
Zwei Subsysteme verbinden und testen						x	x	
Termine und Kosten des Projektes planen	x							
Datenstrukturen festlegen		x	x	x				
Vorhandene Altlasten des Kunden analysieren	x							
Schnittstellen von Programmmodulen definieren		x	x					
Leistung der Entwickler bewerten und belohnen								
Software an neue Umgebung anpassen				x	x	x	x	x
Kunden eine Rechnung stellen								
Test-Eingabedaten für ein Programmmodul ermitteln					x			
Strukturmodell des gesamten Softwaresystems entwerfen		x	x					
Dokumentation des Projektablaufes bewerten und archivieren								
Nach bereits vorhandenen, wiederverwendbaren... ...Software-Bibliotheken suchen	x	x	x	x				
Performance-Prognose des Softwaresystems erstellen			x	x				
Programmcode kommentieren				x				

*A=Analyse; GE=Grobentwurf; FE=Feinentwurf; I=Implementation; MT=Modultest; IT=Integrationstest; ST=Systemtest; AT=Abnahmetest

Begründungen:
Benutzer der Software schulen: Keinen Platz im V-Modell
Qualitätssicherung des Pflichtenheftes prüfen: Das Pflichtenheft legt fest welche Pflichten zum Ende eines Projekts erfüllt sein müssen. Genau diese werden im Abnahmetest geprüft.
Gesetzliche Rahmenbedingungen prüfen: Der gesetzliche Rahmen sollte geklärt sein, bevor das Projekt begonnen wird.
Konzept und Prototyp einer Benutzeroberfläche erstellen: Konzept und Prototyp müssen erstellt werden, nachdem in der Analyse bestimmt wurde welche Anforderungen erfüllt werden sollen, aber bevor die Implementierung startet, damit die Entwickler wissen was zu tun ist.
Entwicklerteam zusammenstellen: Das Entwicklerteam sollte an das Projekt angepasst werden, bevor dieses startet, aber nachdem geklärt wurde welche Qualifikationen erforderlich sind. Einzelne Teammitglieder können während der Entwicklung noch angepasst werden.
Code eines Programmmoduls debuggen: Sollte nach dem Modultest nicht mehr passieren, kann aber trotzdem immer mal wieder vorkommen falls ein konkreter Testfall erst in einer späteren Testphase auftaucht und vorher nicht bekannt war.
Zwei Subsysteme verbinden und testen: Sollte dann geschehen, wenn Module integriert werden und das System getestet wird.
Termine und Kosten des Projektes planen: Sollte erledigt sein bevor das Projekt startet.
Datenstrukturen festlegen: Sollte im Entwurf festgelegt werden, damit die Entwickler sie implementieren können. Kann während der Implementierung noch angepasst werden.
Vorhandene Altlasten des Kunden analysieren: Wichtig zum Entwurf, sollte also in Analyse geschehen.
Schnittstellen von Programmmodulen definieren: Wenn die Implementation in Modulen erfolgt, sollte

vorher klar geregelt sein, wie die Module später miteinander arbeiten. Daher sollte das im Entwurf geschehen.
Leistung der Entwickler bewerten und belohnen: Gehört nicht wirklich zur Softwareentwicklung.
Software an neue Umgebung anpassen: Implementation nötiger Änderungen und testen in der neuen Umgebung.
Kunden eine Rechnung stellen: Gehört zwar nicht wirklich zur Softwareentwicklung, könnte aber durchaus in jedem Schritt geschehen.
Test-Eingabedaten für ein Programmmodul ermitteln: trivial.
Strukturmodell des gesamten Softwaresystems entwerfen: Gehört klar zu dem Entwurf, sollte auf jeden Fall vor der Implementierung geschehen, und kann erst nach der vollständigen Analyse entworfen werden.
Dokumentation des Projektablaufes bewerten und archivieren: Nirgendwo wirklich wichtig, es sei denn angefragt. Dann zu T+W.
Nach bereits vorhandenen, wiederverwendbaren Software-Bibliotheken suchen: Kann so früh beginnen wie in der späten Analyse bis hin zu der Implementation, um die Software zu entwerfen und auch implementieren.
Performance-Prognose des Softwaresystems erstellen: Sollte früh geschehen, damit man die Implementation auch darauf abrichten kann.
Programmcode kommentieren: Während man Code programmiert sollte man dies tun. Falls man im Modultest Anmerkungen ergänzen will, dann kann man dies auch dann noch tun.

c)

Eine starre Zuordnung der Tätigkeiten zu den Grundaktivitäten ist problematisch, weil es sinnvoller ist, einige Tätigkeiten zu unterschiedlichen Zeiten zu machen. Die Grundaktivitäten sollen auch nur einen groben Plan zur Softwareentwicklung geben, dieser ist also nicht in allen Fälle der effizienteste und effektivste Plan. Außerdem können in der Praxis dynamisch Probleme auftreten, die eine zeitnahe Abänderung des geplanten Vorgehens zwingend erforderlich machen. Zum Beispiel kann sich die gesetzliche Lage während der Implementierung ändern, so dass eventuell der Entwurf überarbeitet werden muss.

Aufgabe 1.2

Rolle\Aktivität	Sprint Planning	Sprint Review	Sprint-Retrospektive	Daily Scrum
Product Owner	x	x	x	x
Scrum Master	x	x	x	x
Entwicklungsteam	x	x	x	x

Rolle\Artefakt	Product Backlog	Sprint Backlog	Produktinkrement
Product Owner	x		x
Scrum Master		x	x
Entwicklungsteam	x	x	

Aufgabe 1.3

funktionale Anforderungen:

- Bedienbarkeit der Ladestationen
- Prüfung ob Fahrzeug angeschlossen ist
- Funktionalitäten beider Arten von Ladestation gleich
- Berechnung der benötigten Lademenge
- Äußerung des Ladewunsches
- Bezahlfunktion
- optional Schnellladen, wenn Station eine Schnellladestation ist

nicht-funktionale Anforderungen

- Bedienung über komfortables Interface
- Überprüfung des Fahrzeugs durch Übertragung von Fahrzeugdaten
- Bezahlung nur über Carmpere App oder Smartphone
- Ausfallquote $< 1\%$
- Auto und Ladestationen interagieren miteinander
- Bezahlung verschlüsselt