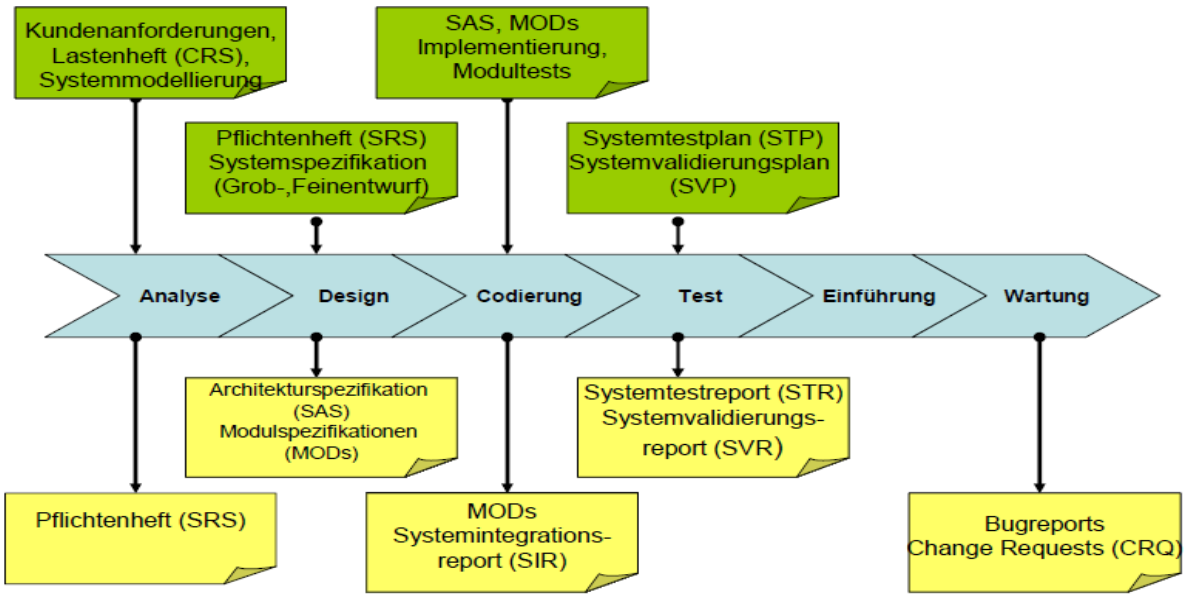
Informationen zum Softwaretechnik-Projekt 2020/21



1. Verfahrensanweisungen

1.1. Ziel

Das vorlesungsbegleitende Software-Projekt soll Ihnen erste Erfahrungen hinsichtlich des Entwurfs, der Implementierung und der Qualitätssicherung von Software in größerem Umfang und innerhalb eines Projektteams geben.
Dies umfasst die Phasen Analyse, Design, Implementierung bis zum Test im abgebildeten Top-Down-Entwicklungsprozess nach einem Phasenmodell.
Es sollen die in den Vorlesungen besprochenen Techniken zur Anwendung kommen.

1.2. Rahmenbedingungen

- Gruppen zu 5-6 Studenten.
- Jeder im Team hat eine oder mehrere bestimmte Rollen und die zugehörigen Aufgaben. Folgende Rollen müssen pro Team besetzt werden:
  - Projektleiter
  - Produktmanager
  - Systemarchitekt (Leitender Entwickler)
  - Testmanager
  - Technischer Redakteur
- Jeder im Team muss für mindestens ein Programmmodul oder Dokument die Verantwortung inkl. Implementierung übernehmen und dies deutlich kennzeichnen.
- Die Teamsitzungen mit Beschlüssen, Status und Ergebnissen müssen in einem Protokoll im Repository (fortlaufend aktualisiertes Issue im Issue-Tracker) dokumentiert werden.

1.3. Bewertung

Die Note ergibt sich aus Folgender prozentualer Verteilung der Bewertung der Gruppenarbeit:

- 2/3 kollektive Leistung für die Implementierung, Entwicklungsdokumentation und Präsentation
- 1/3 individuelle Leistung des Einzelnen in seiner Rolle und für sein verantwortliches Modul bzw. Dokument. Bitte sorgen Sie dafür, dass Ihr individueller Beitrag klar identifizierbar ist, auch in den Wiki-Beiträgen im Repository!

1.4. Aufgabenstellung

Zuerst die allgemeine Aufgabenstellung. Einzelne Hinweise zu Inhalten und Tipps folgen danach und werden dann auch bei Bedarf ergänzt.

Aufgabe im 3. Semester:

Analysephase VL-Wochen 1-6

1. Bilden Sie Gruppen zu 5-6 Studenten und verteilen Sie Rollen wie beschrieben. Diese Gruppen bleiben für das Semester 3 und 4 bestehen.
2. Suchen Sie sich eines der ausgegebenen Projekte aus.
3. Systematisieren Sie die Anforderungen und ergänzen Sie ggf. zusätzliche Kundenanforderungen, die sich im Gespräch mit dem Auftraggeber (Dozenten) ergeben, in einem CRS.
4. Erstellen Sie einen einfachen (!) Business Case (BC), indem Sie von einer fiktiven kommerziellen Auftragsentwicklung ausgehen.
5. Erstellen Sie eine erste Projektplanung für Ihr Team. Verplanen sie maximal 180 Stunden pro Teammitglied (inklusive Administration und Dokumente schreiben).
6. Erstellen Sie CRS und BC jeweils als Word-Dokument.

Achten Sie auf die in der Vorlesung behandelten Punkte zum Thema Requirements Engineering und Systemanalyse. Versuchen Sie, in Ihrer Gruppe durch Reviews zu gut spezifizierten Anforderungen unter Anwendung der geeigneten Modellierungstechniken zu kommen.

Designphase VL-Wochen 7-11

1. Legen Sie pro Team ein Repository mit dem Root "**TINF19C/TeamNr/ProjektName**" in GitHub an, in dem Sie ab dann Ihre Projektartefakte verwalten. Legen Sie dazu jeweils eine Verzeichnisstruktur an wie in Kapitel 1.6 „Dokumentation" beschrieben.
2. Richten Sie im Repository den Issue-Tracker für die diversen Dokumentreviews und das Team-Protokoll ein, mit jeweils einem eigenen Issue pro Dokument (d.h. CRS, BC, SRS, SAS, STP und für jedes MOD). In diesen Issues werden ggf. auch die jeweiligen Dokumentenstände angehängt.
3. Erstellen Sie im Wiki des Repositories das Pflichtenheft (SRS), in dem das System aus Black-Box-Sicht definiert und modelliert wird.
4. Erstellen Sie im Wiki des Repositories die Architekturspezifikation (SAS), modellieren Sie dort das System aus White-Box-Sicht. Leiten Sie die zu implementierenden Module daraus ab und weisen Sie diese im Projektplan als Arbeitspakete den Teammitgliedern zu. Legen Sie eine entsprechende Verzeichnisstruktur für die zugehörigen Sourcecodedateien in GitHub an.
5. Updaten Sie die Projektplanung im Repository („Projects" in GitHub). Versuchen Sie dabei, die Aufgaben zu parallelisieren und beachten Sie Abhängigkeiten zwischen Arbeitspaketen.
6. Die Dozenten raten dazu, bereits im 3. Semester mit der Implementierung eines Prototyps zu beginnen, um das Projektrisiko im 4.Semester entsprechend zu verringern.
7. Jedes Team präsentiert die Ergebnisse in einer 20minütigen Präsentation mit anschließender 10minütiger Verteidigung (üblicherweise letzte Vorlesung).

Abzugeben sind am Ende des 3. Semesters zwei Tage vor dem Präsentationstermin folgende Dokumente in elektronischer Form (PDF und Basisformat) als ZIP-Datei zum Download in der DHBW-NextCloud:

- CRS, BC, SRS, SAS, Projektplan
- Sitzungsprotokolle
- Präsentation

<u>Aufgabe bis Ende des 4. Semesters:</u>

1. Implementieren Sie das System, testen und integrieren Sie Ihre Module und erstellen Sie dazu die Moduldokumentation (MOD) im Repository.
2. Richten Sie im Repository den Issue-Tracker für den Systemtest ein.
3. Erstellen Sie den Systemtestplan (STP) im Repository.
4. Führen Sie den Systemtest durch und dokumentieren Sie die Ergebnisse in einem Systemtestreport (STR).
5. Finalisieren Sie alle abzugebenden Dokumente.
6. Jedes Team präsentiert seine Ergebnisse in einer 20minütigen Präsentation mit anschließender 10minütiger Verteidigung.

Abzugeben sind eine Woche vor dem Präsentationstermin sämtliche Projektdokumente in elektronischer Form (PDF und Basisformat) als ZIP-Datei zum Download in der DHBW-NextCloud:

- CRS, BC, SRS, SAS, Projektplan, MODs, STP, STR, Benutzer Manual, Protokolle, Sourcen, Executable.

Am Tag vor der Präsentation ist auch die finale Projektpräsentation in der NextCloud bereitzustellen.

1.5. Präsentationen

Allgemeines

- Zeitvorgabe (möglichst genau) 20 min
- Etwa 10 min Fragen
- Sie dürfen selbst entscheiden, wer und wie viele präsentieren.
- Da Sie sehr viele Informationen in 20 Minuten darstellen müssen, überlegen Sie sich genau, was Sie sagen wollen und welche Informationen auf die Folien kommen. Planen Sie daher zur Vorbereitung der Präsentation ausreichend Zeit ein (30 min bis 1h pro Folie!).

Präsentation 1 - BC/CRS/SRS/SAS (3. Semester)

Inhalt :

- Teamvorstellung in der der PPT mit Namen, Bild, eMail-Adresse und Projektrolle auf der ersten Folie. Die Dozenten benötigen das um Sie auseinanderhalten und ggf. kontaktieren zu können.
- Vorstellung des Projektes (Master Usecase)
- Darstellung des Funktionsumfangs, Beschreibung der funktionalen und nichtfunktionalen Anforderungen sowie Entscheidung darüber, ob Mandatory oder Optional.
- Business Case (1 Folie)
- Erläutern Sie Ihre Vorgehensweise und verwendete Tools.
- Zeigen Sie, wie Sie modularisiert haben und wie der daraus abgeleitete Projektplan aussieht.
- Beschreiben Sie die Systemarchitektur und die Modularisierung (Top Down)
- Zeigen Sie wesentliche Lösungsansätze aus der SAS (zum Beispiel Paketformat, Anbindung an Java, ERD …).
- Bei Bedarf können Sie auch schon Prototypen vorstellen.

Präsentation 2 - STP/STR/Produkt Präsentation/Selbstkritik (4. Semester)

Beispielhafter Inhalt:

- Teamvorstellung in der PPT mit Namen, Bild, eMail-Adresse und Projektrolle (erste Folie)
- Vorstellung des Projektes (Master Usecase)
- Produktübersicht (Black-Box)
- Architekturübersicht und Module (White-Box)
- Vorgehensweise beim Testen
- Live Demo
- Fazit / Ausblick

1.6. Dokumentation

Allgemeines

- Für die meisten Dokumente erhalten Sie Word-Vorlagen in der NextCloud. Diese geben das allgemeine Gliederungsschema vor. Sie können eigene Vorlagen und Designs verwenden, Sie sollten aber die vorgegebene Gliederung weitgehend beibehalten.
- Diese Vorlagen sind sehr allgemein. Es kann also sein, dass Sie nicht zu allen Punkten, die erwähnt sind, etwas zu schreiben haben.
- Ausfüllhinweise im Dokument bei Finalisierung bitte löschen.
- Als Alternative zu Word-Dokumenten besteht teilweise die Anforderung, die Dokumente im Projektwiki des Repositories (GitHub) zu erstellen und zu pflegen. Diese müssen dann zur Abgabe als PDF extrahiert und versioniert werden.
- Erlaubte Sprachen für die Projektdokumention sind Deutsch und Englisch, für produktbegleitende Dokumente (User Manual, Readme, SRS, SAS, STP) ist Englisch verpflichtend.

Ablage der Dokumente

Bitte verwenden Sie für die Abgaben und im Repository (GitHub) **IMMER** folgendes Ablageschema. Denken Sie an die Versionierung der Dateien. Die Dateinamen sollen wie folgt aufgebaut sein:

TINF19C_CRS_Team_x_0v1,
TINF19C_MOD_Team_x_Modulname_0v1

CRS, BC, SRS, SAS, MODs, STP, STR, Präsentationen und MeetingMinutes (bzw. Links auf deren Position im Wiki) hier ablegen (keine weiteren Unterverzeichnisse):

Team_x_Projektname\PROJECT

Sourcecodedateien in den entsprechenden Unterstrukturen hier ablegen:

Team_x_Projektname\SOURCE\MOD_y_Modulname

Das entwickelte ausführbare Programm zusammen mit dem MANUAL hier ablegen:

Team_x_Projektname\EXECUTABLE

Bei der Übermittlung der Projektdateien in einem ZIP-Archiv müssen die oben geforderten Verzeichnisstrukturen enthalten sein. In jeder Übermittlung müssen auch die vorherigen bereits übermittelten Dateien miteingepackt sein, d.h. es soll immer ein vollständiger Projektordner übermittelt werden. Das gibt Ihnen auch die Chance, ggf. noch Korrekturen bei den bereits abgegeben Dateien einzubringen. In dem Fall immer die Versionsnummer des Dokuments hochzählen.

Es sollen am Schluss alle Dokumente aus Semester 3 und 4 enthalten sein (die neuesten Versionen der Dokumente). Alle finalen Dokumente bitte als PDF UND im Originalformat (Word, MD oder HTML). Bitte KEINE getrennten Unterverzeichnisse für PDF und Originalformat.

Einzelne Dokumente / Items

CRS	
Zweck:	Beschreibt die Kundenwünsche an das Produkt
Umfang:	5 -10 Seiten
Vorlage:	Ja
Methoden:	Strukturierte Anforderung (Schablone), Fließtext, Skizzen der Oberfläche, Use Cases, Anwendungsfälle, Geschäftsvorfälle, …
Inhalt:	Siehe Vorlage

BC	
Zweck:	Kurze Betrachtung der wirtschaftlichen Aspekte des Projekts Max. 4 Seiten
Vorlage:	Nein
Methoden:	Voll / Teilkosten Rechnung, Verfahren zur Risikoanalyse, ...
Inhalt:	Kostenrechnung: - Fixe Kosten und Kosten für die einzelnen Arbeitspakete/Phasen. Rentabilitätsrechnung: - Wie verdiene ich Geld damit? Erste grobe Projektplanung als GANTT-Diagramm - Wie kann ich das Projekt abwickeln?

SRS	
Zweck:	Beschreibt den Produktumfang und fixiert das Feature-Set aus Black-Box-Sicht. 10 -15 Seiten
Vorlage:	Ja
Methoden:	Trackbare strukturierte Anforderungen, klar formuliert (Schablone), Skizzen der Oberfläche, Prototypen, Use Cases mit Ablaufdiagrammen, …
Inhalt:	Siehe Vorlage

SAS	
Zweck:	Beschreibt die konzeptionelle Lösung des Gesamtsystems (Grey-Box, White-Box) 5 -15 Seiten
Vorlage:	Ja
Methoden:	Architektur-, Komponenten-, Klassen-, Kommunikations-, Ablauf- und Sequenzdiagramme, ERM, Datenmodelle, …
Inhalt:	Siehe Vorlage

STP	
Zweck:	Beschreibt die Systemtestfälle und die Testplanung für den Systemtest des Gesamtsystems (Black-Box) 5 -15 Seiten, ca. 10 ausgearbeitete Testfälle
Vorlage:	Ja
Methoden:	Anforderungsbasiertes Testen, Äquivalenzklassenbildung, Grenzwertanalyse, Exploratives Testen, Trennung von Testanweisungen und Testdaten.
Inhalt:	Siehe Vorlage

MeetingMinutes	
Zweck:	Dokumentiert Ihre Projektaktivitäten
Umfang:	10 - 15 Zeilen Status pro Sitzung 1 Sitzung pro Woche
Vorlage:	Nein
Methoden:	Fortlaufend ergänztes Dokument mit Inhaltsverzeichnis, für jede Sitzung ein Unterkapitel.
Inhalt:	Status der Arbeiten, Probleme, Beschlüsse
Hinweis:	Fortlaufende Protokolle als Kapitel in einer einzelnen Datei.

PM Dokument	
Zweck:	Projektplanung
Umfang:	Max. 5 Seiten
Vorlage:	Nein
Methoden:	Siehe PM VL
Inhalt:	GANTT, PSP

1.7. FAQ

F: Woher kommen die Projektvorschläge?
A: Die Projektvorschläge wurden von den Dozenten erstellt und sind meist in deren fachlicher Anwendungsdomäne angesiedelt. Das ermöglicht zusätzlichen Wissenstransfer im Rahmen der Projektarbeit.

F: Was soll dieses Angebot mit dem Open-Source Projekt?
A: Um die Wertigkeit der Projektarbeit für Sie zu steigern, wird angeregt, diese im Rahmen von Open-Source-Projekten abzuhandeln. Das hat folgende Vorteile:

- Die Studenten können später in Ihrem Lebenslauf auf Mitarbeit an einem veröffentlichten Open-Source-Projekt verweisen.
- Die Dozenten können die Software von nachfolgenden Studenten-Jahrgängen weiterentwickeln lassen (siehe rechtliche Hinweise weiter unten).

Beispiel:
http://studium.dhbw-stuttgart.de/informatik/studierendenprojekte/multicastor/

F: Muss ich alle auch optionalen Features implementieren?
A: Es besteht kein Zwang, alle optionalen Requirements zu implementieren, wenn die Aufwände zu groß werden.
Sie sind angehalten, eine realistische Umfangsabschätzung der Projektaufgabe durchzuführen, die der eigenen Leistungsfähigkeit im zur Verfügung stehenden Zeitrahmen entspricht. Entstehen muss am Ende ein funktionierendes sog. „Minimum Viable Product“ (MVP). Features, die anfangs gemeinsam in der SRS als Mandatory vereinbart wurden, dürfen nicht ohne mit dem Auftraggeber (Dozenten) abgestimmte Begründung fehlen. Die optionalen Features dienen als Manövriermasse für Projektrisiken in der Umsetzungsphase.

Rechtliche Hinweise

- Die erstellte Software ist ihr geistiges Eigentum. Sie haben das Vollkommene und alleinige Nutzungsrecht.
- Die Dozenten werden den Source-Code nicht ohne ihr Einverständnis weiterverwenden.
- Bei Veröffentlichung bzw. Offenlegung (z.B. als Open-Source-Projekt im GitHub-Repository) sind sie selbst für die Formulierung entsprechender Lizenzrechte verantwortlich. Es wird die Verwendung der MIT-Lizenz empfohlen.

2. Projektthemen 2020/2021

Es werden Projektthemen mit ähnlichem Schwierigkeitsgrad für 5 Teams angeboten. Bei Unstimmigkeiten unter den Teams, wer welches Projekt machen möchte, treffen die Dozenten die finale Zuweisung. Folgende Themen werden bearbeitet:

Team 1: **PLC-Editor**

~~Jascha Lange (inf19003@lehre.dhbw-stuttgart.de)~~
~~Luca Braun (inf19048@lehre.dhbw-stuttgart.de)~~
Projektleiter: Mouaz Tabboush (inf19151@lehre.dhbw-stuttgart.de)
Produktmanager: Leonie de Santis (inf19107@lehre.dhbw-stuttgart.de)
Systemarchitekt: Elian Yildirim (inf19174@lehre.dhbw-stuttgart.de)
Testmanager: Alle
Technischer Redakteur: Franziska Kopp (inf19082@lehre.dhbw-stuttgart.de)
GIT-Repository: https://github.com/elian15122000/TINF19C-PLCOpen-Editor

Team 2**: Modelling Wizard for Device Descriptions**

Projektleiter: Stefan-Nemanja Banov (inf19014@lehre.dhbw-stuttgart.de)
Produktmanager: Timo Zaoral (inf19133@lehre.dhbw-stuttgart.de)
Systemarchitekt: Simon Jess (inf19182@lehre.dhbw-stuttgart.de)
Testmanager: Jakob Schmidt (inf19205@lehre.dhbw-stuttgart.de)
Technischer Redakteur: Tobias Roth (inf19202@lehre.dhbw-stuttgart.de)
Entwickler: Phillip Thanh Vu Tran (inf19105@lehre.dhbw-stuttgart.de)
GIT-Repository: https://github.com/DekaAthlos/TINF19C-ModellingWizard/tree/master

Team 3: **OPC UA Server Farm**

Projektleiter: Niclas Hörber (inf19046@lehre.dhbw-stuttgart.de)
Produktmanager: Kay Knöpfle (inf19067@lehre.dhbw-stuttgart.de)
Systemarchitekt: Nico Fischer (inf19034@lehre.dhbw-stuttgart.de)
Testmanager: Daniel Zichler (inf19055@lehre.dhbw-stuttgart.de)
Technischer Redakteur: Niklas Huber (inf19216@lehre.dhbw-stuttgart.de)
Entwickler: Phillip Förster (inf19091@lehre.dhbw-stuttgart.de)
GIT-Repository: https://github.com/ghNico/TINF19C-Team-3-OPC-UA-Server-Farm

Team 4**: Service-Registry**

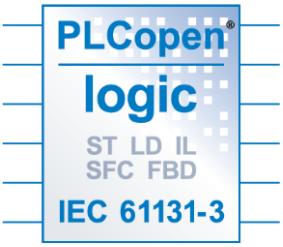
Projektleiter: Goran Erdeljan (inf19170@lehre.dhbw-stuttgart.de)
Produktmanager: Tim Diehl (inf19137@lehre.dhbw-stuttgart.de)
Systemarchitekt: Benedict Wetzel (inf19052@lehre.dhbw-stuttgart.de)
Testmanager: Serdar Ilhan (inf19223@lehre.dhbw-stuttgart.de)
Technischer Redakteur: Daniel Baumann (inf19188@lehre.dhbw-stuttgart.de)
Entwickler: ~~Kai Zinsser (inf19180@lehre.dhbw-stuttgart.de)~~
GIT-Repository: https://github.com/goranerdeljan/TINF19C-Team-4-Service-Registry

Team 5**: AML NoSQL Datenbank Datenverwaltung**

Projektleiter: Jonas Bihr (inf19053@lehre.dhbw-stuttgart.de)
Produktmanager: Namid Faro Marxen (inf19054@lehre.dhbw-stuttgart.de)
Systemarchitekt: Nils-Christopher Wiesenauer (inf19161@lehre.dhbw-stuttgart.de)
Testmanager: Johannes Emanuel Timter (inf19220@lehre.dhbw-stuttgart.de)
Technischer Redakteur: ~~Max Scheub (inf19106@lehre.dhbw-stuttgart.de)~~
GIT-Repository: https://github.com/NurNils/TINF19C_Team_5_AML_Database_Management

2.1. PLCopen-Editor

Es soll ein Programm Editor entwickelt werden, der typische Programmiersprachen für Speicherprogrammierbare Steuerungen (SPS, engl. PLC) nach IEC 61131-3 [1] unterstützt:



- Kontaktplan KOP (engl. Ladder Logic (LD)
- Funktionsbausteinsprache FBS (engl. Function Block (FBD)
- Anweisungsliste AWL (engl. Instruction List (IL)
- Strukturierter Text ST (engl. Structured Text (ST bzw. SCL)
- Ablaufsprache AS (engl. Sequential Function Chart (SFC)

Als Referenz kann der Editor des Open-Source-Projekts OpenPLC [2] verwendet werden. Die GUI soll als browserbasierte Anwendung basierend auf ANGULAR [3] implementiert werden. Der Editor soll die grafische Erstellung von SPS-Programmen ermöglichen. Die Programme sollen im herstellerneutralen XML-Format PLCopen [4] abgespeichert und geladen werden können.

Folgende Teilaufgaben müssen im Wesentlichen bearbeitet werden:

1. Entwurf und Implementierung der GUI basierend auf ANGULAR
2. Unterstützung mindestens von FBS und AS
3. Alle gängigen Logikbausteine sollen als grafische Bibliothekselemente bereitgestellt werden, die in das Bearbeitungsfenster gezogen und über Ihre Ein-/Ausgänge miteinander verknüpft werden können. Dabei ist eine Plausibilitätsprüfung vorzunehmen.
4. Implementierung eines Exporters für PLCopen und AutomationML
5. Implementierung eines Importers für PLCopen und AutomationML
6. Nachweis der Funktion eines Beispielprogramms auf einem Runtime-System und einer interaktiven SCADA-HMI auf einem geeigneten System (z.B. RPi) ohne Hardware E/A bspw. mit einem Modbus TCP Slave (Server) [5]

Die browserbasierte Applikation muss benutzerfreundlich auf dem Host-System eingerichtet werden können und eine ausreichende Online-Benutzerdokumentation bereitstellen.

[1] http://tiegelkamp.eu/Buch/IEC61131-3JohnTiegelkampDeutschV1.2.pdf
[2] https://www.openplcproject.com/plcopen-editor/
[3] https://angular.io/
[4] https://plcopen.org/
[5] http://www.bernhardt.de/openplcproject/

Diese Aufgabe ist gut geeignet für Leute mit Interesse an GUI-Entwicklung unter ANGULAR sowie industrieller Steuerungstechnik und dem XML-basierten plcOpen. Alle Dokumentation ist englischsprachig zu erstellen.

2.2. Modelling Wizard for Device Descriptions

Es soll eine Applikation weiterentwickelt werden, die über eine einfache GUI das Anlegen eines Devices und das Hinzufügen von Geräteschnittstellen (z.B. physikalische Ports) und Datei-Attachements ermöglicht [1][2]. Das Anlegen eines Devices kann manuell erfolgen, aber auch durch Einlesen bestehender Gerätebeschreibungsdateien unter Zuhilfename des DD2AML-Converters [3]. Als Ausgabe soll ein AutomationML-Package erzeugt werden, dass den Regeln für AML-DDs entspricht [4].

Folgende Teilaufgaben müssen im Wesentlichen bearbeitet werden:

1. Entwurf einer GUI und eines zugehörigen Usability-Konzepts
2. Unterstützung von CAEX 2.15 und CAEX 3.0 als Ausgabeformat (konfigurierbar)
3. Eingabefelder für alle Parameter, die nach den Minimalregeln für AML-DDs gefordert werden.
4. Generische Funktionalität zur Anlage von Schnittstellen (elektrisch, mechanisch, hydraulisch etc.)
5. Bei elektrischen Schnittstellen soll die Verwendung der AML-Schnittstellenbibliothek ausgewählt werden können.
6. Eine ausführliche Benutzerdokumentation ist zu erstellen.

[1] https://github.com/Rajkumarpulaparthi/ModellingWizard
[2] https://github.com/tinf17c/ModellingWizard
[3] https://github.com/WAntonia/TINF19C_Team_3_DD2AML-Converter
[4] https://www.automationml.org/o.red.c/dateien.html?cat=1

Diese Aufgabe ist gut geeignet für Leute mit Interesse an Benutzeroberflächen, Beschreibungssprachen und AutomationML. Der Entwurf einer benutzerfreundlichen GUI, eines abstrakten Datenmodells und XML-Interpretation steht im Vordergrund dieses Projekts. Alle Dokumentation ist englischsprachig zu erstellen.

2.3. OPC UA Server Farm

Für den Test von OPC UA Clients ist eine Serverfarm notwendig, die multiple virtuelle OPC UA Server über das Netzwerk bereitstellt. Diese virtuellen OPC UA Server Profile sollen über eine AutomationML-Konfigurationsdatei [1] parametrierbar sein. Die notwendigen Parameter für eine hohe Server-Varianz sind im Rahmen der Arbeit zu ermitteln. Für das OPC-UA-Backend soll der open62541-Stack [2] verwendet werden.

Folgende Teilaufgaben müssen im Wesentlichen bearbeitet werden:

1. Entwicklung des Backends mit OPC UA Server in C/C++ unter MS-Windows und Linux (Raspberry Pi).
2. Definition des Konfigurationsdateiformats in CAEX 3.0 [1]
3. Definition von 10 aussagekräftigen und eine hohe Bandbreite abdeckenden OPC UA Server-Profilen.
4. Test mit dem OPC UA Client UA-Expert [3]

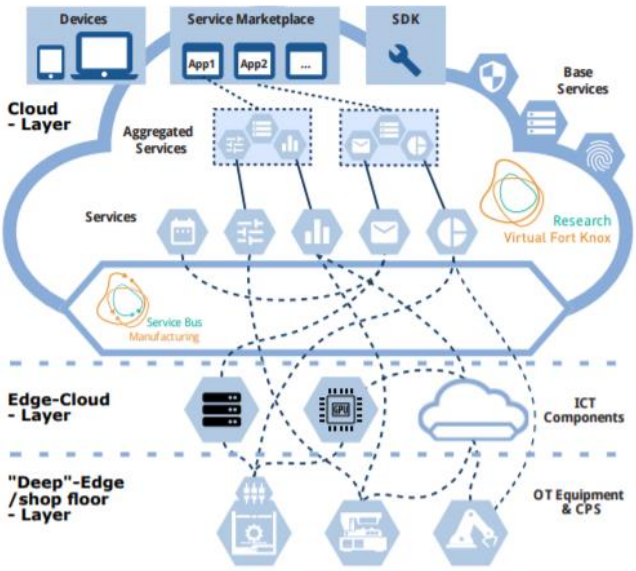
Die Applikation muss benutzerfreundlich auf dem Host-System eingerichtet werden können und ausreichende (Online-)Benutzerdokumentation bereitstellen.

[1] https://www.automationml.org/o.red.c/dateien.html?cat=1
[2] https://open62541.org/
[3] https://www.unified-automation.com/products/development-tools/uaexpert.html

Diese Aufgabe ist gut geeignet für Leute mit Interesse an OPC UA und AutomationML. Alle Dokumentation ist englischsprachig zu erstellen.

2.4. Service-Registry

Ein Asset im Sinne der Industrie 4.0 kann sowohl ein einzelnes Gerät als auch eine über Docker [1] gehostete Applikation sein, bspw. in einem Edge-Gateway. In allen diesen Fällen müssen deren jeweils angebotenen Fähigkeiten über einen Service-Discovery-Mechanismus im Netzwerk bekanntgemacht werden. Ein geeigneter Mechanismus hierfür ist DNS-SD [2][3].



Eine bestehende Registry-Applikation der Open Industry 4.0 Alliance (oi4-registry) soll erweitert werden um eine DNS-SD Anbindung zur Veröffentlichung der Registry-Inhalte über das Netzwerk.

Inhalt der Arbeit ist die Analyse der bestehenden Lösung, die Erweiterung des Datenmodells und der API sowie die Definition typischer Servicetypen als Standardisierungsvorschläge für den Kontext des “Edge Computing“ [4]. Die Service-Registry und Ihre API sollen erweitert und in einem Beispielprojekt unter Linux einige beispielhafte Docker-Applikationen die Funktionalität demonstrieren. Die Docker-Applikationen sollen über eine einfache GUI die Services auflisten können, die Ihnen von den anderen Applikationen bekanntgemacht wurden. Im Rahmen eines zu erstellenden Open-Source-Projekts soll diese Registry-Erweiterung veröffentlicht werden. Eine hochwertige Benutzerdokumentation ist zu erstellen.

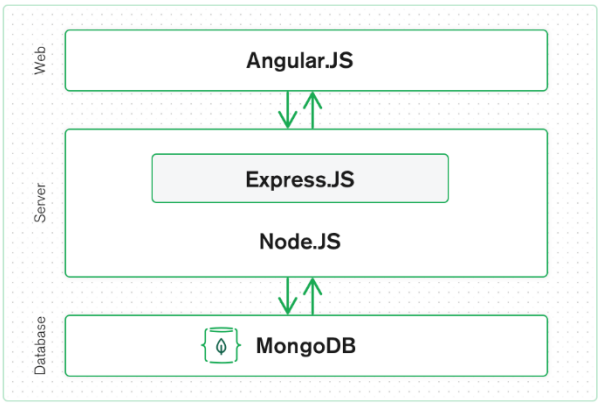
[1] https://www.docker.com/
[2] http://www.ietf.org/rfc/rfc6763.txt
[3] https://blog.stackpath.com/dns-discovery-edge-compute/
[4] https://en.wikipedia.org/wiki/Edge_computing
https://www.edgelox.com/edgelox-device-service-discovery/

Diese Aufgabe ist gut geeignet für Leute mit Interesse an IoT, Industrie 4.0 sowie Netzwerkprotokollen und Linux. Alle Dokumentation ist englischsprachig zu erstellen.

2.5. AML NoSQL Datenbank Datenverwaltung

Für die Verwaltung von AutomationML Dateien in einer NoSQL Datenbank soll eine CRUD (Create,Read,Update,Delete)-Webanwendung entwickelt werden, die mithilfe des MEAN (MongoDb, Express.js, Angular.js, Node.js)-Stack[1] implementiert werden soll.

Da AML auf XML basiert, muss eine geeignete Möglichkeit der Konvertierung in das JSON-Format modelliert und implementiert werden.



Die GUI soll als Angular-Frontend [2] implementiert werden, das Backend basierend auf Express.js/Node.js [3]. Die MongoDb [4] soll lokal als Dockcontainer per Skriptbefehle vorkonfiguriert und gestartet werden.

Die Applikation dient zur (grafischen) Verwaltung dieser lokalen MongoDb Instanz. Um die CRUD Funktionalität zu gewährleisten sollen die Funktionen eines Dateien Up- und Downloads, und zunächst ein ID-basiertes Suchfeld bereitgestellt werden.

Folgende Teilaufgaben müssen im Wesentlichen bearbeitet werden:

1. Entwurf und Implementierung der Frontend-GUI basierend auf Angular
2. Entwurf und Implementierung des Backends bestehend aus Endpunkten für die UI und der Datenbank Logik:
    a. Modellierung der Dokumentstruktur der Datenbankeinträge
    b. Konverter zwischen AML und JSON [6]
    c. Erstellen, Lesen, Löschen und Update von Einträgen in der Datenbank
3. Konfiguration und Verbinden der MongoDB Docker Instanz.

[1] https://www.mongodb.com/mean-stack

[2] https://angularjs.org/

[3] https://expressjs.com/

[4] https://www.mongodb.com/

[5] https://www.docker.com/

[6] https://www.mongodb.com/json-and-bson

Diese Aufgabe ist gut geeignet für Leute mit Interesse an WEBUI- und Datenbank-Entwicklung sowie dem XML-basierten AutomationML. Alle Dokumentation ist englischsprachig zu erstellen.