Project Task
Vehicle management and sensor infrastructure

Table of Contents

1 Introduction

A vehicle requires sensors to detect and react to its environment. A critical function, e.g. an emergency braking assistant, has to react in real-time. The vehicle administrator at the same time wants to know the state of the vehicle, i.e. the vehicle fleet, at any time. Transmitted data encompass current vehicle user as well as the state of the vehicle, e.g. speed/load level, charge, and maintenance requirements. We will implement some partial functions which operate across several levels of abstraction.

2 Introductory Project

2.1 Initial Design (2 Persons)

Task 1: Prepare an initial design for main project 3.1 with a notation of your choice (Text, Drawings, UML, SysML?). The initial design must include some degree of analysis with requirements and use cases as well as reasoning about the communication interface to the management system group. The initial design must describe a basic set of system components, and each component must be assigned exactly one responsible person.

Task 2:

1. Get a Raspberry Pi from the automation laboratory (Eric Kamguia, A233)
2. Prepare your development environment.
3. Implement the following examples with installation and start-up documentation:
   a) A regular (every 100ms) high priority data acquisition task which accumulates data. The data is handed off to a low priority task which displays the data on a graphical user interface. The raw data must include a high precision time stamp (nanoseconds) and jitter must be supervised via the GUI. Data monitored could be cpu temperature, data transferred via network or to the file system, or even from a sensor you connected yourself. Implement an independent low priority task which performs sequential data logging to disk. Use a memory restricted buffer for data input from the other tasks and handle buffer overflows gracefully, i.e., buffer overflows can occur, but data may not simply vanish. The amount of vanished data has to be logged to file.
   b) Implement an exemplary interface for communication with the management system.

Remark: You will receive one Raspberry Pi per group.

2.2 Initial Design and Web-Service with Database (2 Persons)

Task 1: Prepare an initial design for main project 3.2 with a notation of your choice (Text, Drawings, UML, SysML?). You should also consider interfaces to the RealTime groups, and should already think about security issues. The initial design must describe a basic set of system components, and each component must be assigned exactly one responsible person.

Task 2:

1. Decide whether you want to realize your web service in PHP or as a Java Servlet / Java Server Page.
2. Get yourself a running development system for a web service that uses a database.
   a) You can either install a server on your personal computer (e.g. XAMPP, which includes MariaDB for the DBMS, Apache & PHP [if you want to develop in PHP] and also Tomcat [if you want to develop using Java Servlets]), or
   b) you can use a server that is running on ea-pc165.ei.htwg-konstanz.de. (It is running MySQL and Tomcat currently, but Apache & PHP can also be installed if necessary.)
3. Create as a first web service the log-in component of our main project:
   a) Set up the database to store the user information (full name, login name, email, password, time of last login, code for password reset).
   b) Create a first web page where a user can log in with either his email address or his login name and his password. After logging in, he will see a menu (currently only the option to log out). He can also click on a link “Forgot Password?” and give his email address. He will then receive an email with a link and a random number (the code for password reset), and with a click on the link he can then reset his password.
   c) Find a solution of how to avoid having passwords in plain text in the database (Hint: using a hash function to “encrypt” the password).
   d) Investigate on how to secure the web service by using https instead of http.

2.3 Initial research and simple example of COPA-DATA / Zenon data connection

Task 1: Check the Zenon data communication drivers and evaluate options to connect to an embedded system via at least MQTT, OPC, Modbus and two other options. Describe possible communication architectures including necessary infrastructure for these options and their pros and cons.

Task 2: Set-up a simple example.

2.4 Assignment Submission (ungraded)

Deadline: week 27/2020

Group dependent:

1. Initial Design + Example projects on Raspberry Pi
2. Initial Design + Example project on database and log-in web-service
3. Research report + example project with .wsb and other necessary (source) files.

3 Main Project

We request, that each group use the faculty gitlab server https://gitlab.ei.htwg-konstanz.de. You must keep a time sheet where each responsible person records his time spent on implementing a certain component. Exactly one person for each group is responsible for discussion of the interface to the other groups.

3.1 Sensor- and Communication building block in a partly autonomous vehicle (2 Persons)

The vehicle should be capable of being logged into either via the central web-service as implemented by a web-management group, or via an RFID-Transceiver. The vehicle should continuously gather accelleration and orientation data and provide this to the central administration server.

The following components will be used for the implementation of this functionality:

- Raspberry Pi 3
- Raspbian Linux
- Java Virtual Machine
- 6-axis accelleration, gyro and compass sensor
- Interface to management system (WLAN/Ethernet)
- optional: RFID-Transceiver + RFID Tokens

You have to implement the hardware and software interfaces for attachment of the Sensors/Transceivers to the Raspberry Pi board.

The building block should measure further data of the vehicle, e.g., speed, temperature, humidity, battery charge, current driver/user. All data have to be processed sufficiently and they can be requested regularly or be submitted regularly to the management system. Choose a suitable protocol to ensure bi-directional communication. Design and implement a calibration procedure for the compass, which must be controllable from the management system.

3.1.1 Equipment and Lab Access

Please contact Mr. Kamguia to access the necessary components.

We will provide a Raspberry + sensors for your groups use. You have to assemble your system. You may assemble the system in the lab and subsequently access it via remote access for software configuration and implementation. You may also use this equipment outside of the lab. We expect careful treatment and that you return the equipment on submission of your project, or at the end of the course, respectively. If you are implementing outside the AUT Lab, you have to implement the RFID sensor for vehicle access.

3.2 Management and Control System (2 Persons)

For the vehicle administrator, and for later statistical analysis, the vehicle data must be logged to a database. As much information as possible should be stored (e.g. on/off, driver [RFID], battery level, mileage in kilometers, current speed, time stamp, ...).

A display of the current vehicle status and a simple analysis of the time series must be available using a web interface. It must be possible to filter on different criteria (vehicle, driver, time span / most current data).

The web service must be protected against unauthorized access by a login mechanism. A simple user management should be provided to enable an administrator to add and remove users and to reset passwords, etc.

The web service can be realized in PHP or as Java Servlets / Java Server Pages.

3.3 System Structure

1. Vehicle
    1. Motor Controller
    2. Collision Avoidance Controller
    3. Management and Communication Controller
    4. Vehicle User Interface
    5. Sensors
        1. Speed
        2. (Steering Angle)
        3. LIDAR

        4. Temperature
        5. Humidity
        6. RFID
    2. Management System
        1. Communication Controller
        2. Database Controller
        3. Database
        4. Web-server
        5. User Interface

3.4 Assignment submissions (graded)

Deadline: week 29 / 2020

Documentation, time sheet, and all necessary files to operate your project. If submission size is larger then approx. 20 MB, we will provide DVDs for submission.

4 Group building and process

Groups:

1) Sensor- and Communication building block (5 Groups)

2) Web- and DB-Service (5 Group)

3) Additional topics we may need a group for? Suggestions?

Sequential process:

1) Brainstorming and Definition of programming infrastructure (gitlab?)

2) Analysis, initial design

3) Group assignment

5 Analysis and initial Design with SysML

If you want to use SysML instead of UML for system diagrams, we recommend you to read the following paragraphs.

5.1 SYSMOD

The following is an excerpt from [Weilkiens2014], with a guideline for a first level of system description using, e.g., SysML:

Es gibt derzeit drei Stufen plus drei Aufbaustufen. Die Punkte unterhalb der Stufen sind Kriterien, die mindestens erfüllt sein müssen, um auf dieser Stufe zu stehen. Die Aussage ist nicht, dass weitere Artefakte nicht zulässig oder erwünscht sind.

Stufe SYSMOD1: Kommunikation

Der Fokus der Modellierung liegt auf der Kommunikation zwischen den Projektbeteiligten, insbesondere zwischen den einzelnen Disziplinen. Die Verständlichkeit hat eine höhere Priorität als die Genauigkeit und Details. Die Kriterien dieser Stufe lauten:

SYSMODla: Einfacher Systemkontext

Alle Akteure (Benutzer, Fremdsystem, Mechanisches System,…) des Systems sind modelliert.

SYSMODlb: Anforderungen und Anwendungsfälle Die Anforderungen der Stakeholder sind modelliert (Lastenheft, user specification) und nach fachlichen Kriterien in Paketen strukturiert. Zentrale funktionale Anforderungen, insbesondere an denen Benutzer beteiligt sind, sind mit Anwendungsfällen beschrieben.

SYSMODlc: Systemaufbau

Die Systemstruktur ist per Komposition baumartig modelliert. Die Wurzel des Baums ist das System selbst, die Blätter sind Systembausteine, die einem Verantwortlichen vollständig zugeordnet werden können. Eigenschaften der Systembausteine müssen nicht detailliert betrachtet werden.

SYSMODld: Systemstruktur

Die Verbindungen der Systembausteine sind in einem internen Blockdiagramm beschrieben. Schnittstellen in Form von Ports sind nicht vorhanden.

5.2 Tools

In case you want to use SysML diagrams for modeling the system, we recommend to check [SysML-Tools] for freely available tools.

Bibliography

Weilkiens2014: Tim Weilkiens, Systems Engineering mit SysML/UML : Anforderungen, Analyse, Architektur , 2014
SysML-Tools: Tim Weilkiens, SysML Tools List, 2018, https://model-based-systems-engineering.com/sysml-tools/