EH2760 Management of Projects

ESS-CAR/ESS-NW

Stockholm, Sweden

Project Plan

Jonas Ekman
Yini Gao
Jacob Kimblad
Leon Fernandez
Fredrik Hyyrynen
Yifan Ruan

October 10, 2018

Background

Advanced Driver Assistance Systems (ADAS) is one of the hottest research areas in the automotive industry. ADAS should be capable of sensing the vehicle's surroundings, as well as vehicle's own status. To achieve those abilities, multiple sensors and Embedded Control Units (ECUs) are used in the ADAS and a network inside the vehicle is established to combine the data gathered from different nodes.

The two projects "ESS-Car Embedded Services for a Self-Adaptive Car" and "ESS-Network Embedded services for Self-Adaptive Networking" were provided by professor De-Jiu Chen at KTH Royal Institute of Technology. The idea of our projects are from Viktor Karlquist's Master's thesis [1]. He presented a design and implementation of an automotive experimental platform for ADAS, and the "ESS-Car" team from last year's MF2063 course round has already built a autonomous vehicle prototype based on the thesis [2]. Our projects will be focused on the development and extension of the prototype. Instead of traditional IP networks, software defined networking, which centralizes the control plane in the network and allows for a better granular and agile network management, will be implemented by "ESS-NW" team. "ESS-Car" team will implement error injectors on the new prototype which makes the car more autonomous.

Reference Documents

- EH2760 course-pm [3]
- ESS-CAR and ESS-NW entries in the MF2063 project catalog [4]

1 Goals

The goals of the project can be divided into two parts, goals concerning the network implementation on the car and goals concerning the remaining software and hardware system architecture of the car. As both parts are heavily tied into each other some of the sub-goals are dependent on sub-goals from the opposing . The overarching goal of the network implementation is of a more exploratory nature as it is a problem that has not been studied clearly, thus the project aims to explore what different network implementations can be beneficial in this type of infrastructure. The overarching goal can be stated as "The goal of the project is to reconfigure the network using SDN-switches and a Raspberry Pi as a SDN controller". The overarching goal for the software and hardware system architecture can be stated as "The goal of the project is to increase the robustness by revising the existing architecture and implementing additional features such as vision control and a display service." Not related to the technical aspects of the car, but rather the ambitions of the project members, are also the business goals of the project.

1.1 Technical Goals

The technical subgoals are divided up into two parts. Subgoals concerning the ethernet network which takes care of all of the communication between different parts of the system. Subgoals concerning the hardware and system architecture which implements all of the functionalities of the system except for the network communication.

1.1.1 Hardware and Software System Architecture

As discussed in the background, the project builds on an existing architecture. Thus most goals consists of improving the existing architecture or examining alternative implementations. The first subgoal consists of making the system more robust by extending the hardware architecture. This should be

Goal	Test	Delivery
Using computer vision and AI the car should be able to follow a green circle held in front of it	The car should be able to follow a green circle attached to a stick and keeping a distance of 30cm (+-5cm). When the circle turns red, the car should stop within 5 seconds.	2018-11-08
Extend the hardware to using a unique microprocessor for each sensor or actuator and still maintain functionality of following a cardboard within 30cm.	Using new hardware, the car should be able to follow a cardboard held in front of it within 30cm using its sonar. The car should also be able to make a full rotation to the left and the right using its steering mechanism.	2018-11-26
The car should be made fail-safe in the sense that low battery, faulty actuators or sensors and watchdogs should make the vehicle fail in a safe way by stopping the vehicle within five seconds.	The car should be able to stop fully within 5 seconds of injecting a fault into the system.	2018-11-27
The car should perform a series of formal handshakes between components at boot-up before initializing normal functionality.	The control system should receive operational data from all parts of the system before initializing any type of functionality. This should happen within 5 seconds of boot-up.	2018-11-27

Figure 1: The goals and associated tests for the system architecture.

done such that there is one unique microprocessor for each sensor and actuator in the system while maintaining the existing functionalities of the car. The next subgoal is to monitor the operations of the ECU's such that the car can be made fail-safe by implementing fault-detection, watchdogs and battery level indication. The next subgoal is to develop a formal boot- and shutdown-sequence such the the car becomes reliable and fail-safe under normal operation. The final subgoal is to extend the functionality of the existing system by setting up a camera module and choosing an AI algorithm to detect some external stimuli such as a traffic sign such that the rest of the system can react to it (choosing the appropriate speed or steering for example).

1.1.2 Network Goals

The first subgoal is to assemble the car and set up the physical layer which includes the SDN-switches and the hardware running the SDN-controller (connecting everything using ethernet cables and powering it up). Another goal is that the SDN-controller needs to be picked from given specifications such that it can run on a Raspberry Pi which is part of the system, and that it can contact the specific SDN-switches that has already been chosen. When choosing the SDN-controller it should also be put into consideration that it can have an impact whether or not the remaining two goals can be met. The remaining two goals consists of customizing the SDN-network in different ways. The first one is to explore how the SDN-controller can be used to manually assign different pre-configured routes in the network such that packets that are sent from any given node to any other given node are guaranteed to take a specifically assigned route through the network. The last goal is to extend this functionality by automatically reassigning routes in the network based on given a set of given rules and circumstances in the network. This means, for example, that if we detect congestion in some part of the network we

Goal	Test	Delivery
Set up the physical layer by connecting all of the SDN-switches and the SDN-controller in a desired topology such that packets can be routed to and from all nodes of the system.	After setting up the physical layer the cars functionality should remain the same, thus this can be tested in the same way as the first goal for the hardware and software system architecture, by following a cardboard within 30cm using its sonar.	2018-11-13
Choose a SDN-controller that meets the requirements of being able to run on a Raspberry Pi and connect with the Zodiac FX OpenFlow Switches.	The Raspberry Pi should be able to run the chosen software and connect and talk to the SDN-switches in the network.	2018-11-13
Manually assign network flow to preconfigured routes by programming the SDN-switches and the SDN-controller such that packets are guaranteed to take any specifically given route through the network when traveling from node A to node B.	The network should be checked using a debugger or wireshark such that the packages travel the assigned route through the network.	2018-11-13
Monitor the network using the SDN-controller such that network resources can be reassigned at runtime if congestion emerges or behavior changes.	The network should be purposely congested by a node in the network to see that the network drops the extra packages or reassigns resources based on predefined rules.	2018-11-27

Figure 2: The goals and associated tests for the network.

Name	Ambition-level
Jonas Ekman	A
Leon Fernandez	B
Yini Gao	A
Fredrik Hyyrynen	B
Jacob Kimblad	B
Yifan Ruan	A

Figure 3: Non-technical goals for the projects.

can take action to either reroute some packages or simply drop them depending on their criticality.

1.2 Business Goals

The first subgoal is to assemble the car and set up the physical layer, which includes the SDN-switches and the SDN-controller. This is a dependency the other goals.

2 Organization

The requirements of the project were divided into a number of topics, with each team member being focused on two topics. Additionally, Leon and Fredrik have been appointed roles of Project Manager and Secretary, respectively.

2.1 People

2.1.1 Project Members

- Fredrik Hyyrynen, fhyy@kth.se, Software and Computer Vision, Secretary
- Jacob Kimblad, jacobki@kth.se, Software and Control
- Jonas Ekman, jonekm@kth.se, Network and Electronics
- Leon Fernandez, leonfe@kth.se, Network and Software, Project Manager
- Yifan Ruan, yifanr@kth.se, Control and Electronics
- Yini Gao, yini@kth.se, Computer Vision and Software

2.1.2 Stakeholders

- Dejiu Chen, chen@md.kth.se, Stakeholder, Supervisor
- Matthias Becker, mabecker@kth.se, Stakeholder, Supervisor

2.2 Topics and Roles Elaboration

Software - Consists of scheduling the tasks running at the different nodes in the system, managing the bootstrapping of the system upon startup and lastly, writing code for the arduino nodes

Computer Vision - Consists of setting up the Raspberry Pi + Camera subsystem and implementing the commputer vision algorithm

Control - Consists of devising and implementing the main control algorithm that will run on one of the nodes

Network - Consists of configuring the SDN switches and SDN controller as well as writing the application-layer programs that the nodes use to communicate

Electronics - Consists of doing the PCB design for the "motherboard" in the car as well as mounting the sensors and wiring in the car

Project Manager - Consists of communicating with stakeholders, schedule and lead meetings and handling parts orders

Secretary - Consists of taking notes during meetings and being a temporary managar in the absence of the project manager

3 Project Model

Figure 4 and 5 shows the model for the project. The most important milestones are also present in 1 and 2.

4 Commentary: Time Plan

The timeplan has been derived from the WBS and deadlines in EH2760 and MF2063. The timeplan is subject to change depending on if milestones are completed on their deadlines or not. Tollgates have been intentionally placed on Thursday's since the stakeholder meetings will take place on those days and thus, the progress can be evaluated.

5 Commentary: Resource Plan

The resource plan was based on the number of man-hours available. This was in turn based on the number of credits (hp) that the course consists of in period 2, which is 6 hp per person. The total number of man-hours available is 960.

Project phase	Milestone	Tollgate	Ready date	Responsible
Planning done in period 1				
Protoyping	Working sensor and actuators prototype		2018-11-08	Yifan
	Vision object detection done		2018-11-08	Yini
	Status report 1 done		2018-11-09	Fredrik
		Status report 1 submission	2018-11-09	Fredrik
	Working SDN prototype		2018-11-13	Leon
Optimization	Status report 2 done		2018-11-16	Jonas
		Status report 2 submission	2018-11-16	Jonas
	Control software for car done		2018-11-21	Yifan
	Power supply for car done		2018-11-22	Jonas
	Assemble car done		2018-11-22	Jonas
	Status report 3 done		2018-11-23	Jacob
Finalization		Status report 3 submission	2018-11-23	Jacob
	Autonomous system done		2018-11-26	Yifan
	SDN Network done		2018-11-27	Leon

Figure 4: The project model.

Project phase	Milestone	Tollgate	Ready date	Responsible
Finalization (continued)	Collision detection done		2018-11-27	Yini
	Boot up sequence done		2018-11-27	Jacob
	Fault handling done		2018-11-27	Fredrik
	Working autonomous prototype		2018-11-28	Fredrik
Closing		Assemble car approved	2018-11-29	Stakeholder
		SDN Network approved	2018-11-29	Stakeholder
		Control software of car approved	2018-11-29	Stakeholder
		Vision object detection approved	2018-11-29	Stakeholder
		Collision detection approved	2018-11-29	Stakeholder
		Boot up sequence approved	2018-11-29	Stakeholder
		Fault handling approved	2018-11-29	Stakeholder
		Power supply approved	2018-11-29	Stakeholder
		Autonomous system approved	2018-11-29	Stakeholder
	Status report 4 done		2018-11-30	Yini
		Status report 4 submission	2018-11-30	Yini
	Final report done		2018-12-10	Leon
		Final report submission (Embedded Design Project)	2018-12-11	Leon
		Final report submission (Management of Projects)	2018-12-14	Yifan

Figure 5: The project model, part 2.

Risk nr.	Risk	Probability (P: 1-4)	Consequence (C: 1-4)	Risk value (P*C: 1-16)	Consequence Described	Proactive Measure	Proactive Milestone	Reactive Measure(s)	Responsible
1	Wipe working node	4	4	16	Correctly implemented nodes is mistakenly used for other purposes. Many nodes uses the same kind of hardware.	Put finished devices back in the box and mark box with sticker.	-	Fetch code from backup server and attempt to re-build the program on the node.	Everyone
2	Bugs	4	3	12	Some parts of the system behaves differently from the specifications. Other parts of the system that are relying on these faulty parts will fail or result in bad behaviour in some situations.	Always have one stable branch with tested and deployment-ready code. New features are merged into the master branch when finished and the master branch is merged into the stable branch when it has been tested for integration bugs.	Git repo has been started and rules established at the time of writing	Roll back the master branch to the latest stable version and step forward to investigate which commit was causing the issue then debug it.	Fredrik
3	Lack of knowledge	4	3	12	Lack of knowledge in some areas results in the inability to develop parts of the system correctly.	Sort tasks into independent fields (Electronics, Control etc.) so that no one member has to understand all the details of the system.	WBS has been performed at the time of writing	Responsible person reads up on the the missing knowledge. Other group members who get stalled move to other tasks in the meanwhile.	Fredrik
4	Unorganized code	4	3	12	Unreadable code or code that is hard to find makes debugging and validation hard to perform.	Set a couple of dedicated dates where the group sits down together and cleans and rebases the codebase.	<To be decided>	Hold an unplanned "emergency" session where the group cleans and rebases the codebase.	Leon
5	Unorganized documentation	4	3	12	Makes system validation and merging of system parts hard.	Use an automated documentation generator.	Doxygen has been chosen as a generator at the time of writing	Member who finds poorly documented code posts a note on the scrum board.	Leon

Figure 6: A risk matrix for the top five risks.

6 Risk Analysis

7 Communication

The main communications channel used is the kth e-mail. Meetings and scheduling is handled with Microsoft Outlook. For shorter and less important messages, Slack is used. Slack also interfaces with Trello, Google Drive and Github, which facilitates keeping the documentation up to date. Physical meetings takes place on a weekly basis during which progress and possible issuesand possible issues are discussed.

8 Documentation

Documentation is a critical part during software development project. It is a good way to communicate the implementation of everyone's work. So, we need to keep documentation concise and interpretable, for example, overviews and roadmaps can be shown in our documentation to let other teammates know about the work in a short time. Only important information like what is the underlying communication architecture and what type of data it outputs should be elaborate in documentation. Another significant point is the accuracy of the documentation due to some future work will be done base on it. Indexing and linking should also be added to our documentations as references project. We use Google Drive and GitHub to manage our documentations since Google Drive allows us to work on the documentation simultaneously and GitHub provides control of builds in case of mis-deleting. Another reason we choose to upload our document to the cloud is that typically the well-known cloud drive will have good maintenance work. In this case, we normally won't lose any important document if we store them on the server. In our project, documentation will be created throughout the entire project development lifecycle. And we should write our documents in a iterate way, which means that we should update our documents after getting some feedback during the whole development process. For instance, we are keeping changing the description of a programming code when we add new parts of it or if someone in the group gives feedback in case of not readable documents. Finally, we should treat our documentation like requirements. We need to estimate the priority of our documents and deal with the most important one first. This priority may change during the development process and some of documents can even be removed, which is fully depends on the project's demand.

References

[1] Karlquist, "Design and implementation of an automotive experimental platform for ADAS," KTH, Skolan för informations- och kommunikationsteknik (ICT), 2017.

[2] Bark *et al.*, "Embedded service for self-adaptive cars," KTH, Instutitionen för Maskinkonstruktion, 2017. [Online]. Available: https://kth.box.com/s/iaabvkok3fsqgpbgaq10ymgqnnf3a135

[3] Lilliesköld, Gingnell, Petterson, and Varawala, "Management of projects eh2760," KTH, Management of Technology Research Group, 2018.

[4] Chen and Derbyshire, "The embedded systems HK 2018 project descriptions," KTH, Instutitionen för Maskinkonstruktion, 2018.

A Time Plan

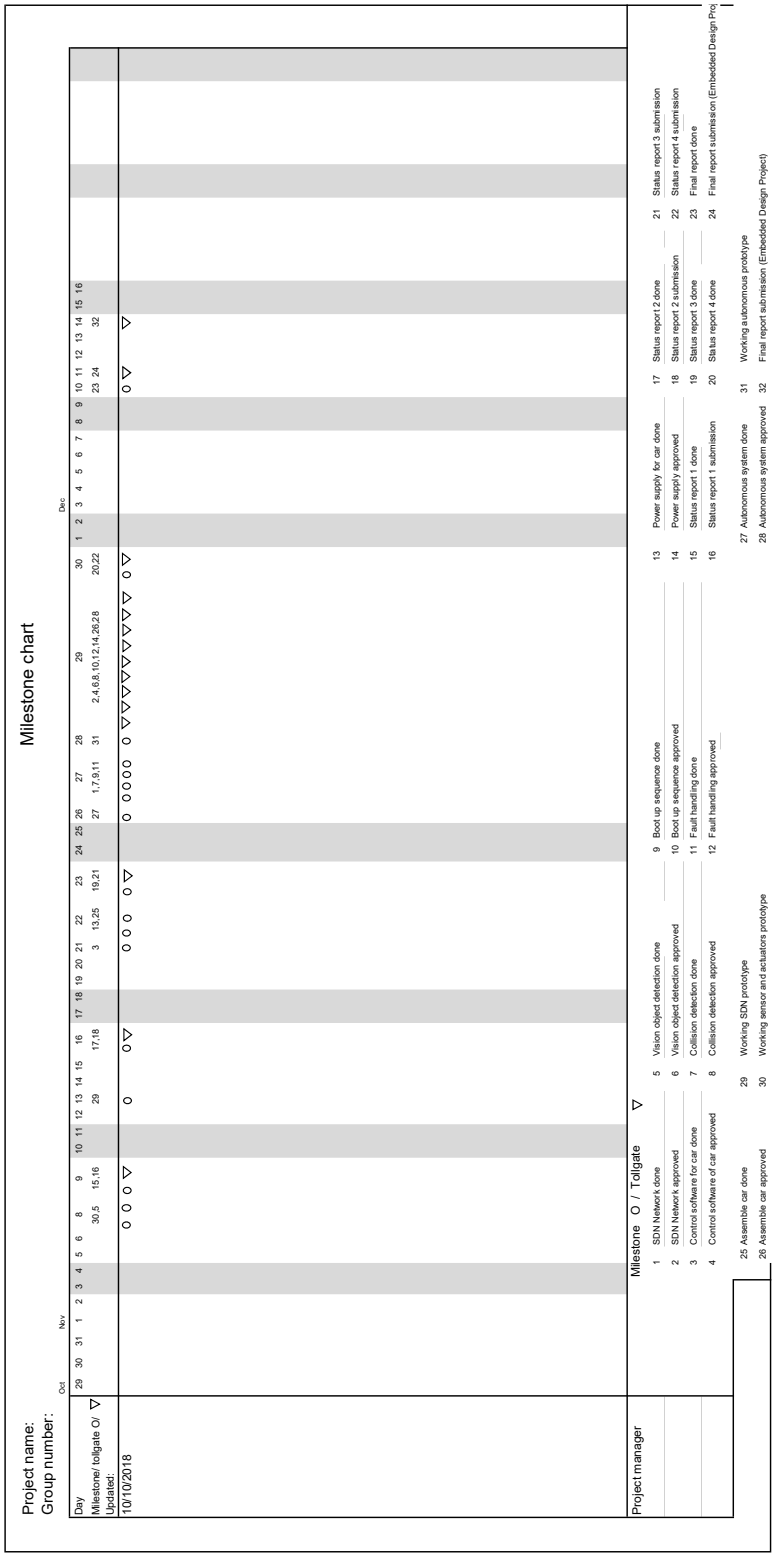

Figure 7: The timeplan for the project.

B Resource Plan

Figure 8: The EVM resource plan for the project.

C Work Breakdown Structure

The WBS was done digitally on Google Drive. It was later summarized into Figure 9.

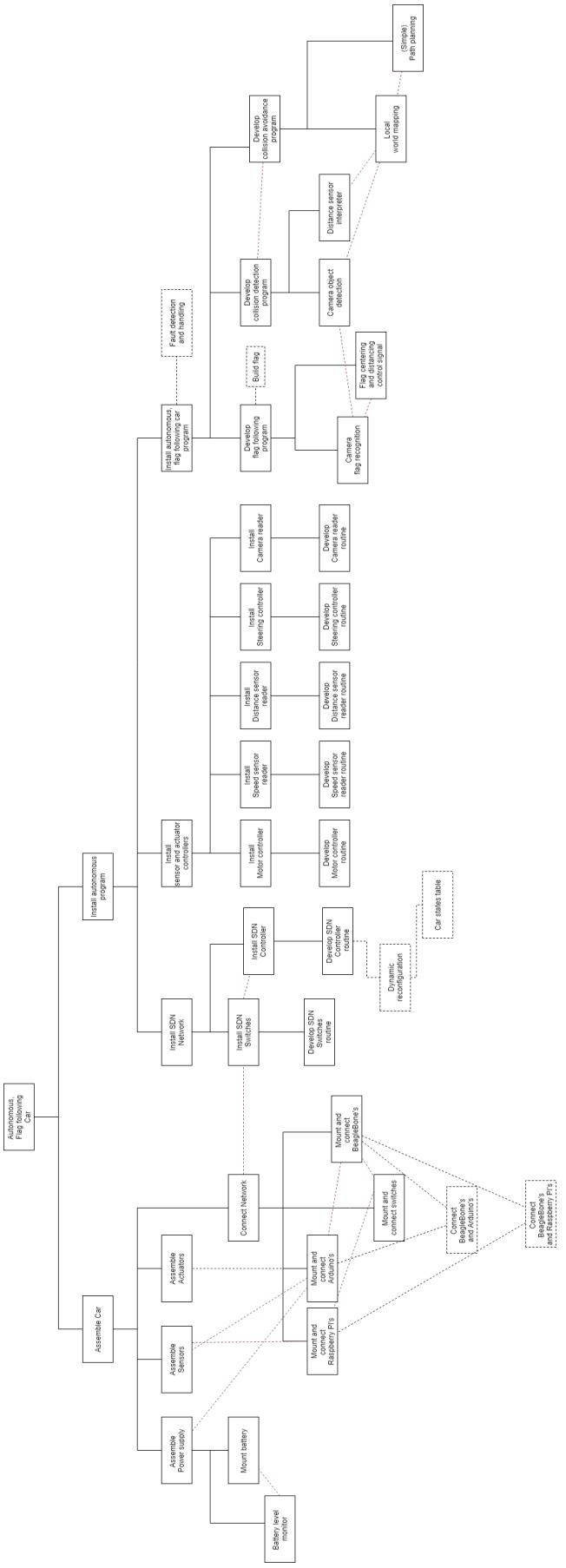

Figure 9: The Work Breakdown Structure for the autonomous car.