School of Innovation, Design and Engineering
28 February 2018
DVA422 - Software Engineering 3

Albert Bergström
abm13007@student.mdh.se

Kevin Oswaldo Cabrera Navarro
kco17001@student.mdh.se

Laboration 2: Software Architecture Design and Documentation

Table of Contents

Laboratory 2

Part 2.1 (estimated time 3:30).

Key stakeholders:

Analyst

- Decomposition - Knowing the architecture and how the modules interacts.

Describe their interest in the software architecture:
The analyst task is to analyse the software architecture and how well it's able to satisfy the quality attributes. In this case, the analyst might analyse availability and performance since these attributes are very important for a car system.

Architect

- Decomposition - Is responsible for the structure. Wants to know how the system is decomposed.
- Uses - Wants to know how and which of the resources and the modules are allocated to describe how they depend of each other, to allow him/her to find any possibility of extending the system.
- Concurrency - The systems resources is going to be shared, which the architecture is responsible for. The concurrency view is useful for identifying opportunities for parallelization of the modules.

Describe their interest in the software architecture:
The architect establishes the software architecture at a large scale. Needs to choose which approach is best for the software architecture and control that the software architecture can fulfil the requirements. Must also communicate with the stakeholders and mediate clashing requirements and design approaches.

Designer

- Decomposition - The designer is responsible for determining each modules responsibility, what is should contain and be able to do (or not do). To accomplish this, the designer uses the decomposition view to assign what responsibilities each module should have.
- Uses - To have an actual design that meets the specific requirements defined by the architecture, the designer needs to know how the modules interact with each other.
- Concurrency - Designer oversees the resolving of the resource contention, the best way to get this data is using the concurrency view.

Describe their interest in the software architecture:
Uses the software architecture when designing parts of the system with more detail. Might be reasonable for the design of the "collision avoidance" so it fulfils its function while being in line with the requirements set in the software architecture.

Evaluator

- Decomposition -

Describe their interest in the software architecture:
The evaluator is responsible for evaluating the software architecture, to study its ability to deliver on the requirements specified.

Implementer.

- Decomposition - The modules structure delivered by the decomposition view enumerates what the units of software will have to do, which is indeed a super important information the implementer needs.
- Implementation, the best way to understand the inviolable constraints for the activities development, the implementation view is suitable, since it is really focused in the integration and the test activities.

Describe their interest in the software architecture: The implementer needs to know how the system is structured to accomplish its job. Without a good knowledge of the architecture, the final product has a bigger chance of having failing/bugs or result in a different product from which was intended by the architect.

Integrator

- Decomposition - The integrator may be interested in the decomposition for getting the information of the modules structure.
- Use - Integrate the individual components without any knowledge of how each other module depend of each other can be a problem without a use structure view.
- Concurrency - The integrator must know where the resources contention exists, because he is in charge of locating the source of the integration failures, the concurrency view will give the needed information to achieve it.
- Implementation - The implementation view contains information on how the modules are mapped to each other, which is relevant when it comes to performing the actual integration of components into the system.

Describe their interest in the software architecture: His work is based according to the architecture, so he really depends on the documentation and the architecture to accomplish the integration of the individual components the system may have.

Project manager

- Work assignment - Needs this to plan and manage the resources.
- Implementation - To get a view of what needs to be implemented, what they need to implement and how to administrate test activities.

Describe their interest in the software architecture: The project manager has one primary function, to make the project exist. He does not need to understand the details of how every part of the system works. Nevertheless, he still needs to know the software architecture to plan and schedule the amount of resources available to get good time estimations.

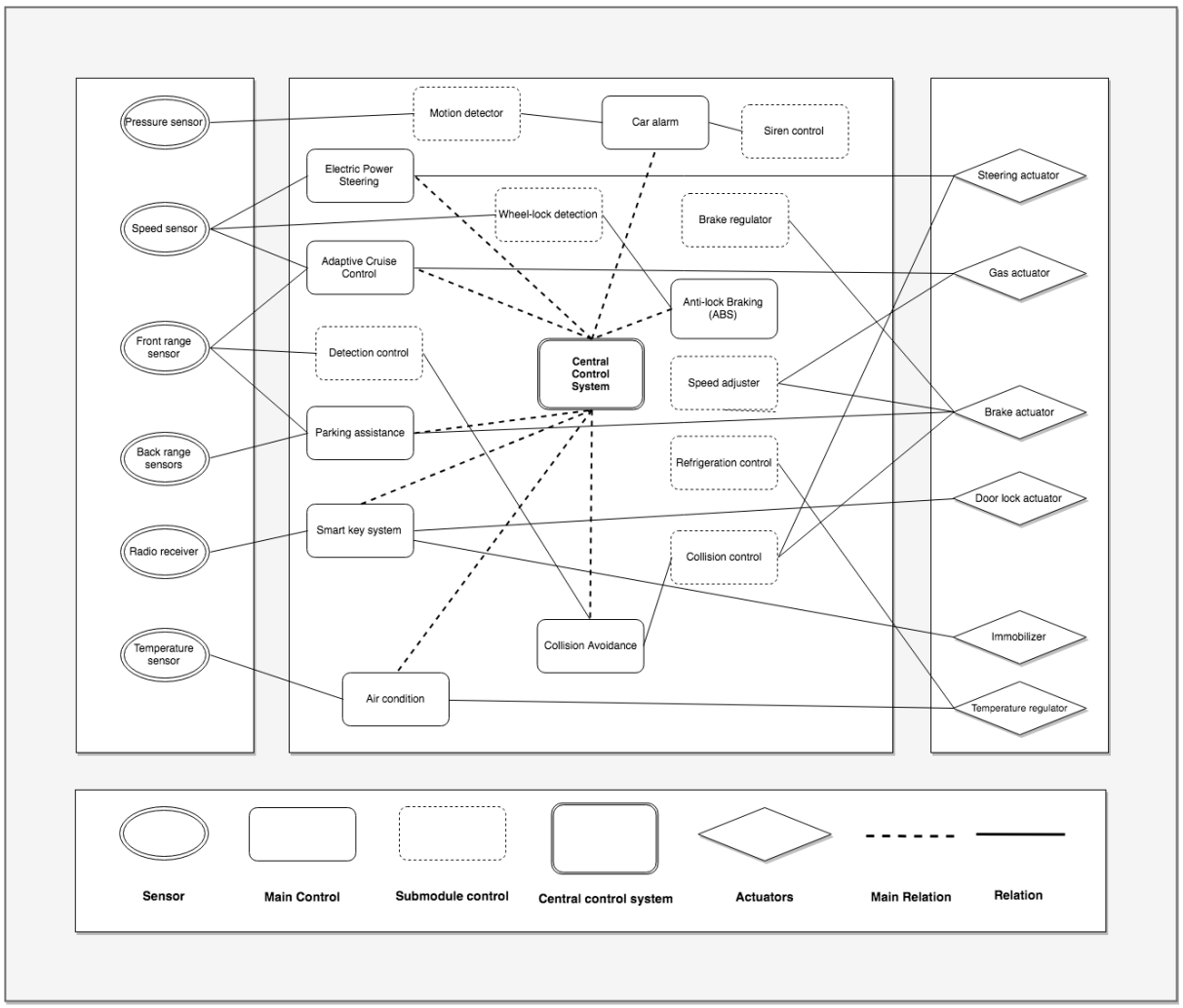
System engineer

- Decomposition - The person responsible of the design, must know the structure and configuration of the system. Decomposition view provides the needed information for achieving this task thanks to the big picture that it gives in the starting point of the design.
- Uses - The system engineer needs to know how the modules depend on each other to assure the environment provided by the system is sufficient.

Described their interest in the software architecture: You can say that the design and architecture are two things that are tied. A system engineer needs to understand the relationships and properties provided by the architecture for the design and development of the system.

Laboratorion 2.2 (estimated time 6:14)

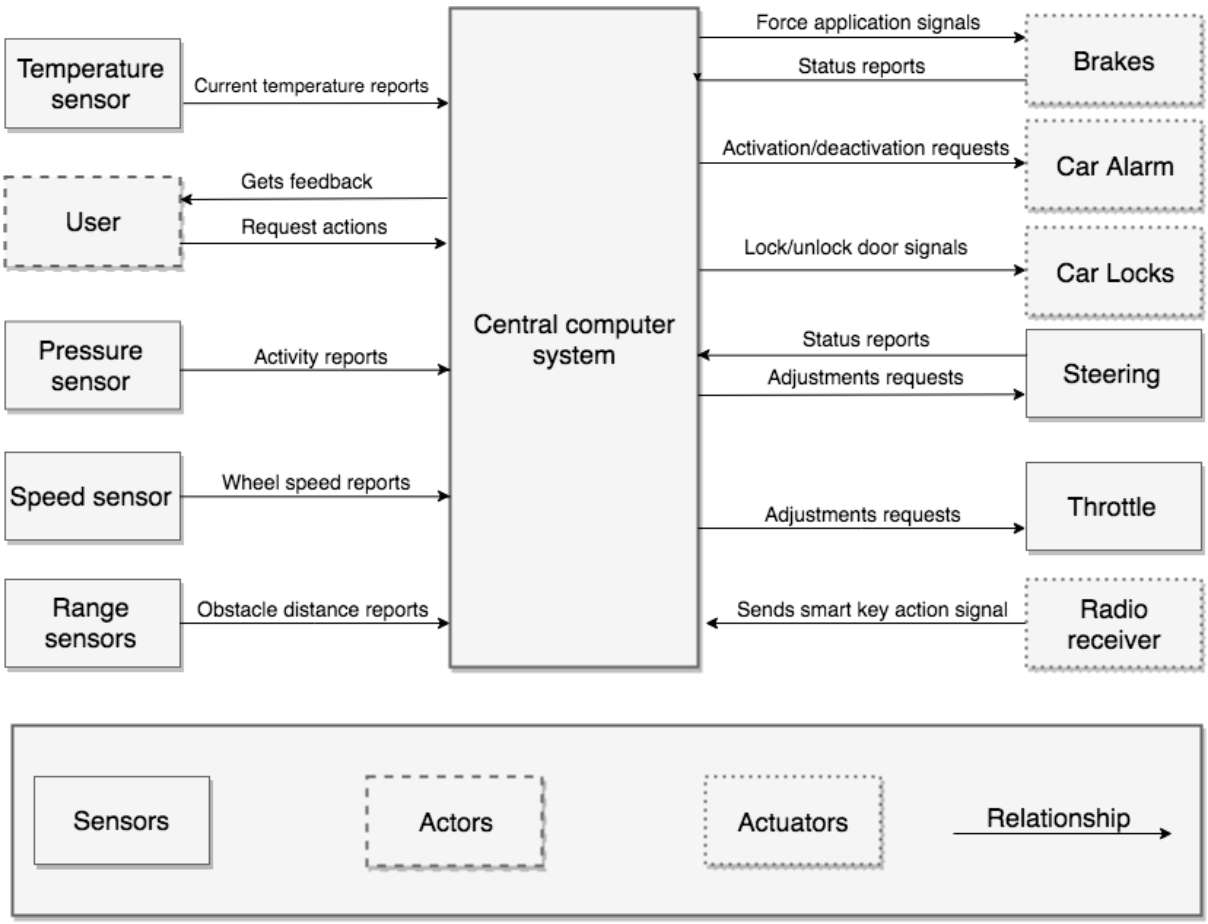
Section 1. The Primary Presentation



Section 2. Elements Catalog

- **Car alarm.** A siren sounds if someone tries to do a forced entry to the vehicle when it's locked.
    - Motion detector: Analyses the signals given by all the sensors and reacts when they are dramatically changed by sending signals to the main module.
    - Siren control: When activated, starts sending signals to the light actuator and the siren actuator which will produce blinking lights and high pitch sounds.
- **Adaptive Cruise Control.** The adaptive cruise control adapts the speed so that the car keeps a fixed distance to the vehicle in front.
- **Collision Avoidance.** This function actively takes over steering and breaking from the driver to avoid a collision.
    - Collision control: Takes over control of steering and braking. Regulates based on the signals received by the main module.
    - Detection control: Uses the signals from front range sensors to determine if the vehicle is in risk of colliding. In case a collision is imminent, signals the main module with information about the threat.
- **Parking assistance.** Parking assistance guides the driver when parking the vehicle. It starts to beep if the car gets too close to obstacles and prevents collision by breaking if necessary.
- **Anti-lock Braking System (ABS)**. This function prevents the brakes from locking. When a wheel locks due to heavy breaking from the driver the system will take over and release the brake to avoid locking.
    - Brake regulator: Sends signals which limits pressure in the brakes from going completely through.
    - Wheel-lock detection: Listens to signals about the rotation of the wheels. If it detects an abnormally high deacceleration, it signals that the wheels are at risk of locking.
- **Electric Power Steering.** This function assists the driver in steering the car with the help of an electric motor. The power of the electric motor is automatically adjusted such that more assistance is given when the car moves at a low speed.
- **Air condition:** This function controls the temperature inside the car. It is set to make the temperature inside match a set values, by adjusting based on the current temperature.
    - Refrigeration control
- **Smart key system.** This function allows the start of the vehicle motor, locking and unlocking without the need of inserting any traditional physical key.

Section 3. Context Diagram



Section 4. Variety Guide

The centralized car system architecture is designed in a way that a certain module can be skipped in future vehicle designs without the need of modifying other modules. In the other hand, the **Central computer system** structure helps the designers to add new functionalities and not having to worry about how the other modules work. These are some examples:

Car alarm could make use of motion sensors in addition to pressure sensors. With motion sensors, the **Car alarm** would be able to detect subtler breaking attempts such as when the car windows are not destroyed. Another variant of this point is replacing the pressure sensor with a simple sound sensor. This variation might be cheaper than pressure sensors, but also has the risk of falsely triggering the **Car alarm**.

The **Adaptive cruise control** can be change to a non-adaptive variation, which doesn't utilize range sensors. The non-adaptive cruise control would only keep a certain speed rather than dynamically changing the speed to keep a set distance from the car in front.

A variation of the **Collision avoidance** system, which might have improved performance, could make use of additional range sensors on each side of the car. If the **Collision avoidance** could process information about obstacles on the car’s sides, it would have

better coverage of possible threats. For example: If the driver is rapidly approaching a car or a wall on either side, the **Collision avoidance** would be able to adjust the steering to avoid the danger.

Electric Power Steering can be exchange for a hydraulic steering, maybe it will not give the same assistance when the vehicle is moving at low speed, but it will provide a more economical price if desired.

The **Parking assistance** design can be improved with the help of a camera in the back of the vehicle. This would allow **Parking assistance** to further assist the user by showing low obstacles which can't be seen from the driver's seat. The camera would also improve comfortability, since the user wouldn't need to turn around, allowing for a more intuitive way to park.

Air condition and **Smart Key** modules can be totally omitted to make an economical version of the system. Almost all the previous modules will not modify the performance of the main system since they are just an extra feature added in the vehicle. The main difference in these two modules is that they are not sharing any resources with the rest of the modules besides their own functionalities. **Air condition** can still replace the temperature sensor for an electrical fan that provides a constant temperature indicated by the user.

Section 5. Rationale

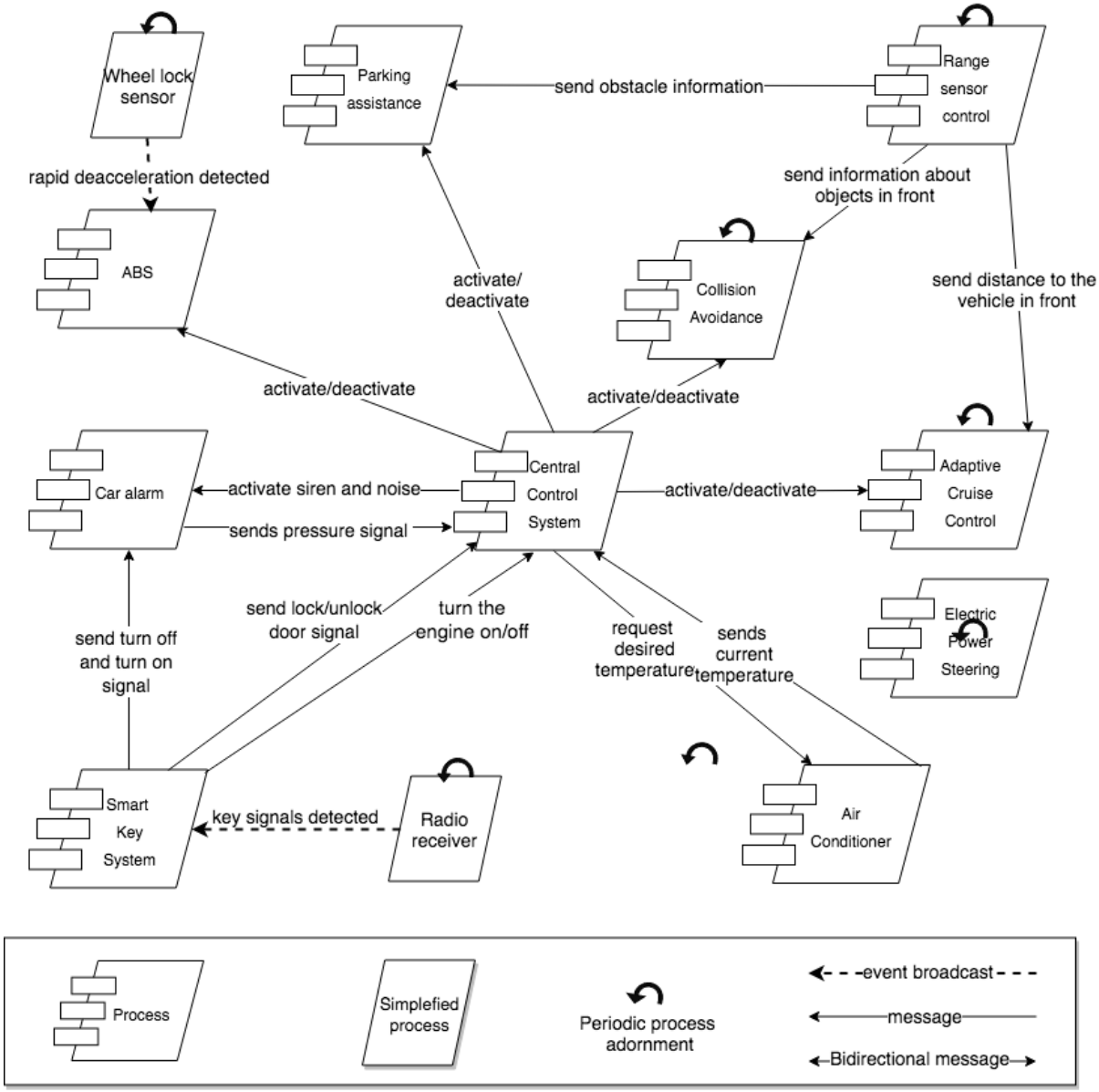
Modern cars now have a lot of Electronic Control Units (ECU) inside their systems, this make the functionality control a very hard task to accomplish. Therefore, we decided to introduce a single central computing system, where all the tasks are going to be controlled by a centralized control software. This new architecture will allow the easy addition and removal of new functionalities in case there is a variation in future vehicles designs. Some of the tactics used in the composition were:

Multi-tier pattern
The central computer system was designed to associate and control the components and submodules through a main communication medium. This is the reason why we used a multi-tier pattern, it will be helpful in the software-hardware allocation [1], which is a very common task in the vehicles systems.

Modularization
Since this system was designed to only control of a limited number of systems and would likely be expanded upon on a later date, modifiability was a very important quality of the system. As such, we made us of certain modifiability tactics, to ensure that future improvements and changes can be made with relative low implementation cost. Some of the tactics are[BCK12_7.2]:
Keeping the size of modules small. We aimed to divide complex modules with a lot of responsibilities and information to process into smaller submodules. The reasoning for this is that having smaller modules will likely reduce the cost of implementing changes, since the change would be narrowed down to one or two submodules where it would be isolated and not affect other parts of the system.

Laboratorion 2.3 (estimated time 5:59)

Section 1. The Primary Presentation



Section 2. Elements Catalog

- **ABS**: When this element is activated by the **Central control system**, it starts listening for messages sent from the **Wheel lock** element. If it receives such a message, it means that the wheels are at risk of locking. In that case the **ABS** wakes up and starts to limit the amount of force available for the braking system. This limitation will have higher priority than any other element using the braking system, such as **Collision avoidance**. This will continue until the element stop receiving messages from the **Wheel lock**. The **ABS** can also be deactivated by the **Central control system.**
- **Wheel lock**: constantly analyses the wheel speed. How big is the acceleration/deceleration compared to the elapsed time? If it detects a strong deceleration over a very short time, then it starts to send signals to the **ABS** element about the risk. It will keep sending signals as long as the wheels are at risk of locking.
- **Parking assistance**: When this element is activated by the **Central control system** it starts listening for messages from the **Range sensor control** about the area behind and in front of the car. If the received messages contain information about something nearby, meaning there exist an obstacle near the car, the **Parking assistance** will start beeping. The frequency of the beep is proportional to the distance, becoming more frequent when approaching the obstacle. The approach speed is also analysed. If the obstacle is being approached too fast, the **Parking assistance** will start applying the brakes. This element is active until the **Central control system** deactivates it.
- **Range sensor control:** This element continuously accesses the raw data from every range sensor in the car. When it notices changes in the data, the **Range sensor control** interprets it and starts sending messages to **Collision avoidance** and **Adaptive cruise control** about the front sensors and to **Parking assistance** about both front and back.
- **Collision avoidance**: The **Collision avoidance** element is activatable by the **Central control system** when the car is moving. When activated, the element will start continuously listening to messages sent from the **Range sensor control**. If an object is in front or close to the front of the car, a risk analyse will be made. If the **Collision avoidance** element determines a collision risk, meaning the car is heading towards an obstacle dangerously quickly, it will calculate what measures needs to be taken and take over the brakes and steering processes management to perform them. The processes in this system has high priority and can override processes from other submodules using the same resources.
- **Car Alarm:** Car alarm is activated by default. Whenever the pressure sensors inside the vehicle sense an unusual data, the alarm system will be activated until the user deactivates it with the help of the **Smart Key System.** The smart key also has a panic button, which will create a process to activate the alarm manually.
- **Central control system:** This element is responsible for initiating and terminating processes through the system. It is the communication media between the rest of the main processes and modules. The **Central control system** listens to input given by the user and will start or stop processes accordingly. For example: if the user presses the cruise control button on their dashboard, the **Central control system** will initiate the **Adaptive cruise control**.

- **Adaptive cruise control**: To maintain a fixed gap between another car, there is a front rear sensor sending data relating to the front vehicle distance. This information is processed by the system and calculates acceleration needed for the car for keeping the desired separation requested by the user.
- **Smart key system** There are different tasks performed by the smart key system. All these processes start in the physical vehicle key, which will send a signal to the **Radio receiver.** Only one thread at the time can be processed. Some of the processes are: open and lock the car's door, turn on and off the alarm and give signals to the immobilizer for turning on the vehicle's engine.
- **Air Conditioner (A/C):** Air conditioner is activated and deactivated by the user using an interface. The A/C system will take the temperature with its sensors and refrigerate the air until it gets the same climate requested by the user though the interface.

Most of the normal and simplified processes can work at the same running time. Nevertheless, there are some modules that can't perform on parallel, these are the cases:

- To activate the **car alarm** processes, the vehicle needs to be turned off. Therefore, the processes done by the **air conditioner** are impossible since they can't get the power supply provide by the engine to be activated.
- **Collision avoidance** is in on top priority because it is concerned with the user security. Since this module takes control of the braking system, **adaptive cruise control** and **parking assistance** processes become incompatible to perform at the same running time. Each of them uses the braking system to perform a different task. In the end, the central control system will decide which of the modules will be used depending on the situation.

Section 3. Variety Guide

How is this structure going to support different hardware and functionalities in the future?

The **Smart key system** can be replaced by an ordinary key system, which would replace the **Smart key system** in the structure with a much simpler key process. This key process would fill the same role as the **Smart key system** but achieve it in a slightly different way**.** Like the **Smart key system,** when the buttons on the key is pressed a unique radio signal is sent which will be picked up by the **Radio receiver,** which will trigger the key process to send door lock/unlock signals to the **Central control system**, as well as activating/deactivating the **Car alarm**. Instead of wirelessly turning the engine off/on, the key process would simply listen for activities in the keyhole mechanics and turn the engine off or on based on the signals.

The car could be outfitted with more range sensors at the sides of the car, which would affect the **Range sensor control**. This process would be required to process more data, since there would be more range sensors. Information about objects on the sides of the car would be forwarded as messages to the **Parking assistance** and the **Collision avoidance,** in addition to the previous messages normally sent. The **Parking assistance** would make use of the extra data when listening for obstacles around the car, while the **Collision avoidance** would be able to detect threats it previously couldn't see.

Section 4. Rationale

Just like the previous section diagrams, **Modularization** and **Multi-tier patterns** were the principal architecture tactics.

- Reduce a size of a module: The processes in the **Smart Key System** are separated from the receiver antenna to reduce the average costs in case of any of the modules malfunction or future changes. [1]
- We *increased the semantic coherence* in the **Range sensor system**. Instead of each module calculating the distance in their own, we separated this function in an independent system, so **Parking assistance**, **Collision Avoidance** and **Adaptive Cruise Control** can request the data with only a call. In the end we implemented the *using an intermediary* tactic*,* **Central Control System** oversees all processes communication, breaking all the dependencies between two modules to work, since a module now can work without another. [1]

References

[1] Len Bass, Paul Clements, Rick Kazman, Software Architecture in Practice, Third Edition, Addison-Wesley, 2012.

[2] David Parnas, "On the Criteria to Be Used in Decomposing Systems into Modules," In Communications of the ACM, Volume 15, Issue 12, December 1972.