Lab 1 – Requirements Engineering

Regulations, Gitlab

- **Lab Teams, Lab Quartets, Gitlab:** Each lab team consists of 2 team members. Two lab teams form a teampair, i.e. lab quartet with a gitlab project that is only accessible to the team members and the supervisor @wrenzhamburg.
- **Setting up a gitlab project:** Create a group with the 5 members (limit of free tier). In the group create an empty SElab-project with a unique name. (You can rename it.)
- **1st step:** The two teams in the quartet can choose different solutions in the following. Decide in the team of 2, whether you want to provide your lab 1 report upload with LaTeX or with markdown. If you would like to stay with docx, you need to experience your own different merge solutions based on markup-conversion via pandoc or turtoisedoc etc.. In case of LaTeX, it is recommended to use a main.tex as frame and to put the report parts in separate tex files so that these can be developed independently. For beginners it is recommended to just work with your team of two in the main branch.
- **2nd step:** When starting your lab work also start reporting all your activities in the lab 1 report via git. Every team member is an author of your common report.
- **Responsibilities:** Decide who will be responsible for which task parts and activities.
- **Resources:** All resources used have to be cited/quoted in the reference list, in particular when it is taken from a class mate or from the web, proceed identically with AI text passages. *Recommendation:* Right from the start store all used URLs including the access-time in your report's working directory (e.g. by drag and drop of the URL in the browser).
- **Preparation:** Preparation (via documents in gitlab) for and presence in each of the 6 labs are mandatory. Each team is allowed to not pass the preparation once in term.
- **Plagiarism:** If you copy text or figures from somewhere, a reference to that source must be given. For each section of the report, the document has to indicate the corresponding author(s) in agreement with the git history.

Context Description

You are part of the development team of a SME specializing in developing new products and services in the field of smart home environments, in particular household service robots. As a quartet you decide if you want to work for a **robotic lawn mower company branch** or for a **robot vacuum cleaner company branch**. The parent company has decided to enter the robot market with some innovative features in hardware, e.g. sensors and actors for adaptive robot behavior depending on detected surface properties, and software including but not limited to integration with established smart home environments.

Assume that the companies hardware team has already decided about the robot base system for both company branches, a mechanical platform with drive and navigation system, which they will get from a component supplier, in agreement with the technical tests of the base navigation software delivered with it, that uses the mechanical and infrared sensors as well as the navigation laser mounted and provides higher level navigation that your software development team can build on.

Problem 1

You want to develop a **robot app** for mobile devices (IOS- or Android-based smartphones). The app shall be used for controlling robot activities such as starting the robot in "auto" mode while it is generating a map of the area, and to pause or explicitly send it back to the charging dock, where it adds an item to the history of jobs with area operated on. For starting in "room" or "area" mode that can also be used for future job planning, the map offers autogenerated area segments ("rooms"). Furthermore, the user shall be able to define regions for robot jobs in "region" mode. Presume that local WIFI is available, and that the robot offers a location service (based on its sensors). To establish connection with your robot for the first time, the app must discover the robot in the local WIFI system, and the user shall register then the robot with the app's cloud system. Later, only login to the app is required to authorize the robot use by the app as registered. Also read problem 2 for a full understanding of the app's functionality.

Problem 2

You want to develop a **robot system/firmware** for a mobile robot platform that is equipped with all sensors and effectors necessary for securely moving over plain and rough ground and not falling down any steps, and for navigation with a laser scanner. Furthermore, an on/off switch as well as "auto", "pause" and "home" buttons are available to start operation in auto mode while the robot is generating a map of the operation area, to pause it or to send it back to the charging station, respectively. For communication with a robot app on IOS- or Android-based smartphones, the local WIFI shall be used. The current status including map updates shall be send to the app in regular time intervals. Also, commands sent by the authorized app shall be executed asap. Consider also required failure behaviour and exception handling. Also read problem 1 for a full understanding of the robot's system.

Lab 1 Report Activities

1. Identify the relevant slides of the related slide sets provided for the lecture and put reference to these into the reference list. Clarify questions in the Moodle FAQ - labs!
2. Conduct a market analysis and identify typical product features for the robot system and the app. Contact your team members to study different existing apps or robots resp. and keep track of who studied what for documentation in the report. Put all that information including links in the report appendix "Market Analysis".
3. After the market analysis, list the identified product features in a features section (and classify them according to the Kano Model), still for robot and app together.
4. Make a list of stakeholders who you would include into your requirements engineering phase.
5. Individually in your team decide for one of the components, app or robot system, analyse requirements w.t.r. and prepare user stories for software development of it. (Mark which of your user stories obey to the INVEST criteria and re-formulate your user stories accordingly. This list could form the first version of a scrum backlog.)
6. Deadline: finish on Sunday! (Parts in brackets can be done on Monday.)