Software Engineering for Robotics: Introduction

Software development vs. Programming

Programming is simply a process of writing a program for a specific purpose.
Software development is more than writing programs - it is a process of developing individual programs and combining those together into a coherent set that works in unison for achieving a concrete goal.
In large domains such as robotics, software development is commonly done by teams rather than by individuals; management is thus an essential element of the software development process.

Robot Software development

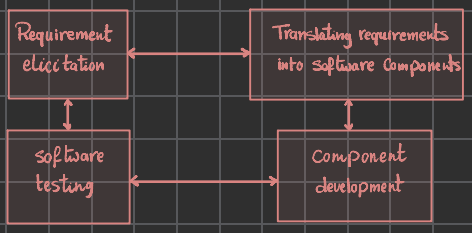
- **Hardware:** Robots are hardware platforms that act in the physical world, this distinguishes them from software-only applications.
- **Real-world unpredictability:** Except under special circumstances, the real world can sometimes be unpredictable, so robot software needs to take this into account.
- **Domain-Complexity:** Robotics combines many domains (e.g. navigation, manipulation, computer vision, etc.) and thus requires the collaboration of a variety of domain experts.
- **Network-based Communication:** A modern robot is distributed system, which sometimes complicates the use of well-established software paradigms, such as object-oriented programming.

Robot software and reusability

Robot software is typically used for a specific robot platform.
The question then becomes: Can the developed software be reused on different robots?
- In some cases, reusability is not possible due to major hardware differences. (e.g. software developed for a flying robot is likely to have limited usability for a wheeled robot)
- More often that not, robots have many physical similarities though; it should then, in principle be possible to adapt the software from one platform for another one.
  - ↳ Reusability is often achieved by creating reconfigurable components. (at design or runtime)

Essential steps in Software development

While the concrete details of every software project differs, there are a few general steps that always need to be done:



In practice, these are not performed just once in a sequential order, but instead need to inform each other and may need to be performed in an iterative process.

1. Requirement Elicitation:

- for a small scope project, the developer know what the robot should achieve.
- When a robot is developed for a concrete application, in which case the needs of the robot's prospective users are of primary importance: A robot that doesn't fulfill the needs of its users is very unlikely to be used by them.
- In long-running projects, requirements can change over time; elicitation shouldn't be seen as a one-time process.

Requirements elicitation is a process of identifying the objectives and expectations of a software system before the software development process starts.

2. Translating requirements into Software Components

- Once the requirements of a project is known, the next relevant step is to map them to software components that need to be developed.
- This translation includes: the design of a software architecture or a plan on how the new components will be integrated into an existing architecture, during this step the (APIs) of the components are defined.
- In the context of collaborative software development, this result in a concrete set of development tasks that each developer can work on.

The translation of requirements into software components is a process of conceptually designing components and their APIs so that objectives defined by the requirements can be satisfied.

3. Component Development:
- After designing the Components, the next step is to actually develop them.
- The development process should be done according to Concrete development and Collaboration Standards:
  - Typically, each Software project has its own rules and guidelines.
  - Even for own small Project, it is useful to develop own development guidelines.
- Components need to be developed based on their agreed upon design and by following the development guidelines of the overall Software Project.

4. Software Testing:
- while developing Components, it is necessary to ensure that they do what you actually intend them to do. "Purpose of testing"
- Testing Can be done at different levels:
  - at the level of individual Components (component testing)
  - to verify that multiple Components work well together (integration testing)
  - at the level of a Complete System (System testing)
- Testing Should accompany the complete development Process and is often supported by automated tools.
- Software testing is a process of ensuring that developed Components or the System as a whole Comply to their actual requirements.