Software Engineering for HPC - Notes

260236

March 2024

Preface

Every theory section in these notes has been taken from two sources:

- None

About:

GitHub repository

Contents

1 Introduction

1.1 The importance of software engineering

Software engineering is so important because it is everywhere. Our society is now totally dependent on software-intensive systems. Think about it. The society could not function without software, for example:

- Transportation systems;
- Energy systems;
- Manufacturing systems.

For these reasons, **software failures cannot be tolerated**.

In the following list, we can see some famous software issues:

- **911 Outage on April 2014**. On 10th April 2014, Washington State had no 911 service for six hours. A software issue causes this event. The software dispatching the calls had a counter used to assign a unique identifier to each call. The counter went over the threshold defined by developers. All calls from that moment on were rejected.

  More info is here.

- **Ariane 5, 1996**. On 4th June 1996, forty seconds after take off, Ariane 5 broke up and exploded. The total cost for developing the launcher has been 8000 million dollars. The launcher contained a cluster of satellites for 500 million dollars. Again, the explosion was caused by software failure.

  More info is here: accident tech report and video.

1.2 Software engineering: definition

There are some fields of computer science dealing with software systems:

- Large and complex;
- Built by teams;
- It exists in many versions;
- Last many years;
- Undergo changes.

In each field, a software engineer needs to have some skills. In contrast to a *programmer* that has the following abilities:

- They develop a complete program;
- They work on known specifications;
- They work individually.

A **software engineer** has the following **skills**:

- **Identifies** *requirements* and develops *specifications*;
- **Designs** a component to be combined with other components, developed, maintained, and used by others; component can become part of several systems;
- **Works in a team**.

We can <u>summarize</u> the skills of a software engineer as follows:

- Technical
- Project management
- Cognitive
- Enterprise organization
- Interaction with different cultures
- Domain knowledge

The **main goal** of a software engineer is to **develop software products**. Not only is the product significant, but the **process is also fundamental**. The quality of the process affects the quality of the product.

1.3 The software product and the process

The product developed by a software engineer differs from traditional product types. It isn't easy to describe and evaluate because it is intangible. Some **aspects affecting the product quality**:

- **Development technology**;
- **Process quality**;
- **People quality**;
- **Cost, time and schedule**.

1.3.1 ISO/IEC 25010

An ISO (International Organization for Standardization) called **ISO/IEC 25010** comprises the **nine quality characteristics**:

SOFTWARE PRODUCT QUALITY								
FUNCTIONAL SUITABILITY	PERFORMANCE EFFICIENCY	COMPATIBILITY	INTERACTION CAPABILITY	RELIABILITY	SECURITY	MAINTAINABILITY	FLEXIBILITY	SAFETY
FUNCTIONAL COMPLETENESS FUNCTIONAL CORRECTNESS FUNCTIONAL APPROPRIATENESS iso25000.com	TIME BEHAVIOUR RESOURCE UTILIZATION CAPACITY	CO-EXISTENCE INTEROPERABILITY	APPROPRIATENESS RECOGNIZABILITY LEARNABILITY OPERABILITY USER ERROR PROTECTION USER ENGAGEMENT INCLUSIVITY USER ASSISTANCE SELF-DESCRIPTIVENESS	FAULTLESSNESS AVAILABILITY FAULT TOLERANCE RECOVERABILITY	CONFIDENTIALITY INTEGRITY NON-REPUDIATION ACCOUNTABILITY AUTHENTICITY RESISTANCE	MODULARITY REUSABILITY ANALYSABILITY MODIFIABILITY TESTABILITY	ADAPTABILITY SCALABILITY INSTALLABILITY REPLACEABILITY	OPERATIONAL CONSTRAINT RISK IDENTIFICATION FAIL SAFE HAZARD WARNING SAFE INTEGRATION

Figure 1: ISO/IEC 25010

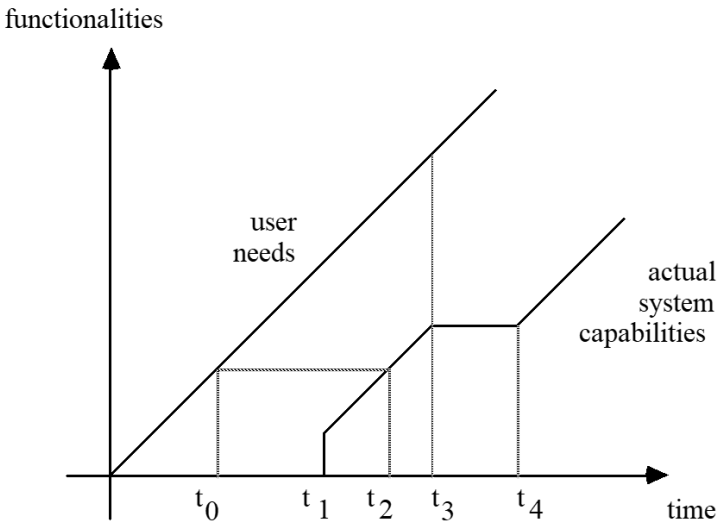
1.3.2 Productivity

A process quality to consider is **productivity** (the **process of producing a product**). The definition can be: "ability to produce a good amount of product". To **measure it**, we can use **delivered items by unit of effort**, where:

- **Delivered items**: lines of code (and variations) function points;
- **Unit of effort**: person month (note: persons and months cannot be interchanged).

1.3.3 Timeliness

Another process quality to consider is timeliness. The definition is: "**the ability to respond to change requests in a timely fashion**".



As you can see by the graph, the "*user needs*" is a linear function (and sometimes can be exponential!). A software engineer should be able to respond to the client's requests as soon as possible. As the graph shows, a request made on time t_0 is completed on time t_2; but another request can be made at that time, and so on. The actual system capabilities can't grow up always because sometimes there are "brainstorming times" to increase product quality (ISO/IEC 25010).

References