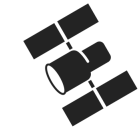
Lesson 2

Software specifications. Formal methods

In this lesson we will talk about:
Software specifications, Formal methods,
Software verification and validation

”Todays software has grown by evolution, not by intelligent design”

— *Leslie Lamport*



Space Instruments
Hubble Telescope

Military drones, Mars Curiosity Rover

Modern airplanes
Boeing Dreamliner

F35 Lightning II

Computer Software
Windows, Debian, Office

Apple macOS

Automotive
BMW, Mercedes, Tesla

Our software is **complex**, **buggy** and **difficult** to maintain.

"If our software is buggy, what does that say about its security?"

— *Prof. Robert H. Morris*


GLOBAL OUTAGE
2024 CrowdStrike
2
4
Clear Channel


:(
Your PC ran into a problem and needs to restart. We're jus
some error info, and then we'll restart for you.
100% complete
For more information about this issue and possible fixes, visit https://www.windows.com/st
Women
Gates B62 to B81
Men
Family
B63
Gate

2024 CrowdStrike-related IT outages

Article Talk

From Wikipedia, the free encyclopedia

On 19 July 2024, American cybersecurity company CrowdStrike distributed a faulty update to its Falcon Sensor security software that caused widespread problems with Microsoft Windows computers running the software. As a result, roughly 8.5 million systems crashed and were unable to properly restart[1] in what has been called the largest outage in the history of information technology[2] and "historic in scale".[3]

The outage disrupted daily life, businesses, and governments around the world. Many industries were affected—airlines, airports, banks, hotels, hospitals, manufacturing, stock markets, broadcasting, gas stations, retail stores, and more—as were governmental services, such as emergency services and websites.[4][5] The worldwide financial damage has been estimated to be at least US$10 billion.[6]

What can we do about it?

Start with a **smart design** at the specification level

Software Specifications

What is a specification?

A **specification** often refers to a set of documented requirements to be satisfied by a material, design, product, or service.[1] A specification is often a type of technical standard.

There are different types of technical or engineering specifications (specs), and the term is used differently in different technical contexts. They often refer to particular documents, and/or particular information within them. The word *specification* is broadly defined as "to state explicitly or in detail" or "to be specific".

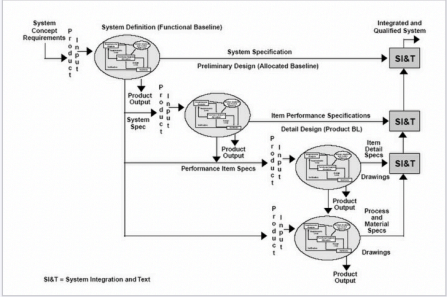
A **requirement specification** is a documented requirement, or set of documented requirements, to be satisfied by a given material, design, product, service, etc.[1] It is a common early part of engineering design and product development processes in many fields.

A **functional specification** is a kind of requirement specification, and may show functional block diagrams. [*citation needed*]

What is a functional specification?

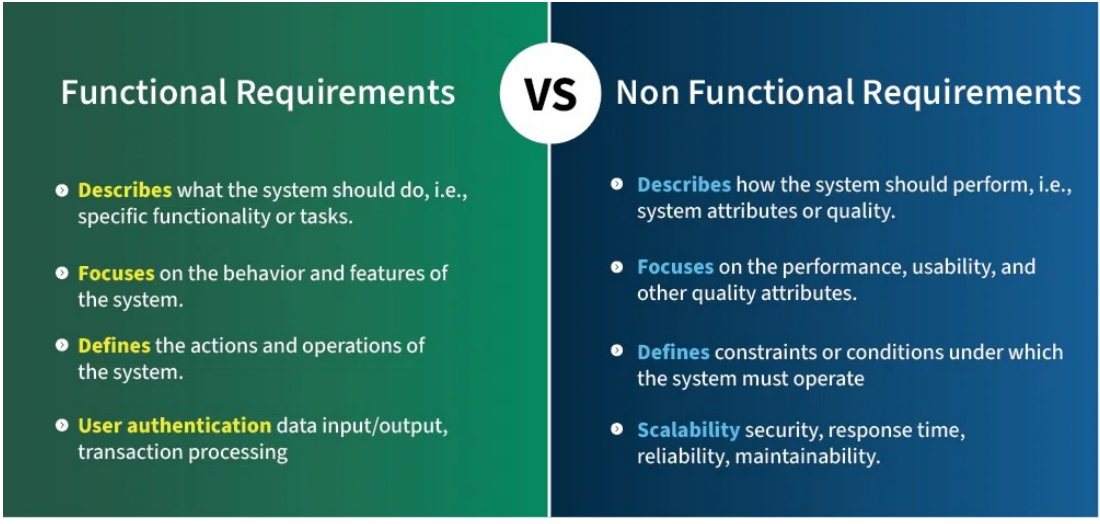
A **functional specification** (also, *functional spec*, *specs*, *functional specifications document (FSD)*, *functional requirements specification*) in systems engineering and software development is a document that specifies the functions that a system or component must perform (often part of a requirements specification) (ISO/IEC/IEEE 24765-2010).[1]

The documentation typically describes what is needed by the system user as well as requested properties of inputs and outputs (e.g. of the software system). A functional specification is the more technical response to a matching requirements document, e.g. the Product Requirements Document "PRD"[*citation needed*]. Thus it picks up the results of the requirements analysis stage. On more complex systems multiple levels of functional specifications will typically nest to each other, e.g. on the system level, on the module level and on the level of technical details.



Systems engineering model of Specification and Levels of Development. During system development a series of specifications are generated to describe the system at different levels of detail. These program unique specifications form the core of the configuration baselines. As shown here, in addition to referring to different levels within the system hierarchy, these baselines are defined at different phases of the design process.Note: There is one minor (and ironic) typo in the image above. SI&T is "System Integration and Test" not "System Integration and Text".

A **functional software specification** is a written description of what our program is supposed to do.


Functional Requirements
VS
Non Functional Requirements
Describes what the system should do, i.e., specific functionality or tasks.
Focuses on the behavior and features of the system.
Defines the actions and operations of the system.
User authentication data input/output, transaction processing
Describes how the system should perform, i.e., system attributes or quality.
Focuses on the performance, usability, and other quality attributes.
Defines constraints or conditions under which the system must operate
Scalability security, response time, reliability, maintainability.

How can we write the specification?

- English. Finnish. Chinese
- Graphical diagrams. Drawing
- Technical sketches. Mockups

Functional specification

Helps us to understand what our program will do. Its highly important to write first a specification of our program before implementing and developing anything.

The specification – what our program does.

But how should we describe this specification? How can we be precise? What does it mean to be precise?

Imprecision can lead to **ERRORS!**

Precise specifications

Its difficult to be precise using English or Chinese. Thats why in science and engineering precise specifications have adopted basic maths to describe the specifications.

Basic math

Elementary mathematics. Propositional logic.

Propositional Logic

Elementary algebra is the mathematics of real numbers and the operators $+$, $-$, $*$ (multiplication), and $/$ (division). Propositional logic is the mathematics of the two Boolean values TRUE and FALSE and the five operators whose names (and common pronunciations) are

$\wedge$	conjunction (and)	$\Rightarrow$	implication (implies)
$\vee$	disjunction (or)	$\equiv$	equivalence (is equivalent to)
$\neg$	negation (not)		

Propositional Logic, cont.

To learn how to compute with numbers, you had to memorize addition and multiplication tables and algorithms for calculating with multidigit numbers. Propositional logic is much simpler, since there are only two values, TRUE and FALSE. To learn how to compute with these values, all you need to know are the following definitions of the five Boolean operators:

- $\wedge$ $F \wedge G$ equals TRUE iff both F and G equal TRUE.
- $\vee$ $F \vee G$ equals TRUE iff F or G equals TRUE (or both do).
- $\neg$ $\neg F$ equals TRUE iff F equals FALSE.
- $\Rightarrow$ $F \Rightarrow G$ equals TRUE iff F equals FALSE or G equals TRUE (or both).
- $\equiv$ $F \equiv G$ equals TRUE iff F and G both equal TRUE or both equal FALSE.

Formal methods

In computer science, **formal methods** are mathematically rigorous techniques for the specification, development, analysis, and verification of software and hardware systems.[1] The use of formal methods for software and hardware design is motivated by the expectation that, as in other engineering disciplines, performing appropriate mathematical analysis can contribute to the reliability and robustness of a design.[2]

Formal methods employ a variety of theoretical computer science fundamentals, including logic calculi, formal languages, automata theory, control theory, program semantics, type systems, and type theory.[3]

Specifications using formal methods

Formal methods may be used to give a formal description of the system to be developed, at whatever level of detail desired. Further formal methods may depend on this specification to synthesize a program or to verify the correctness of a system.

Alternatively, specification may be the only stage in which formal methods is used. By writing a specification, ambiguities in the informal requirements can be discovered and resolved. Additionally, engineers can use a formal specification as a reference to guide their development processes.[4]

Formal verification is the use of software tools to prove properties of a formal specification, or to prove that a formal model of a system implementation satisfies its specification.

Once a formal specification has been developed, the specification may be used as the basis for proving properties of the specification.

Why maths?

- Quantify output behavior
- Predict behavior of systems as design time
- Understand safety margins
- Understand system interactions

Verification: Are we building the product right?

Validation: Are we building the right product?

"Building the product right" checks that the specifications are correctly implemented by the system while "building the right product" refers back to the user's needs. Ideally, formal methods provide a mathematical guarantee that software meets its specifications.

Industry


03 | 20 | 2015
DRAFT 29837611-0004
SCALE: 01 : 01
amazon
web services™
ENGINEERING TEAM
AWS-0910
PROJECT
AMAZON WEB SERVICES
A3WWSI6

Industry: Amazon, Microsoft, NASA, ESA, Intel

DOI:10.1145/2699417

Engineers use TLA+ to prevent serious but subtle bugs from reaching production.

BY CHRIS NEWCOMBE, TIM RATH, FAN ZHANG, BOGDAN MUNTEANU, MARC BROOKER, AND MICHAEL DEARDEUFF

How Amazon Web Services Uses Formal Methods

TLA+ Specifications of the Consistency Guarantees Provided by Cosmos DB