International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

http://ijpsat.ijsht-journals.org
Vol. 24 No. 1 December 2020, pp. 510-519

# *Embedded Devices Overview: Does Security Match The Evolution Of Technology*

Costin Carabaș[1], Nicolae Țăpuș[1]

[1]Computer Science, POLITEHNICA University of Bucharest,
313 Splaiul Independenței, Bucharest, Romania



**Abstract – The Internet and IT&C industry started in the middle 20th century and had an extremely rapid evolution. It has had an impact on the whole planet and has helped humankind achieve greatness. This evolution also had some negative aspects. Technology was often used to do harm to people, institutions, corporations and so on. Nowadays, security has an important role in any computer science project development.**

**This paper aims to create an overview of the security aspects in major projects of our time, such as operating systems, applications, internet communication. The Introduction describes a short history of the IT&C evolution, and its importance. The Mobile Devices section presents a classification of the current mobile devices regarding software and hardware. The Security Development chapter is an overview of the security solutions used in computer science, its evolution and the direction the industry is heading. The following chapters are Motivation and Conclusion.**

**Keywords – Tehnology Evolution, Analysis, Software, Hardware, Security.**

## I. INTRODUCTION

The history of computers and the internet begins in the mid-20th century as several science laboratories from different parts of the world. The architects did not expected the impact that IT&C would have on the world and the fact that everyone would use it, so the evolution disregarded the importance of security.

**Corresponding Author:** Costin Carabaș

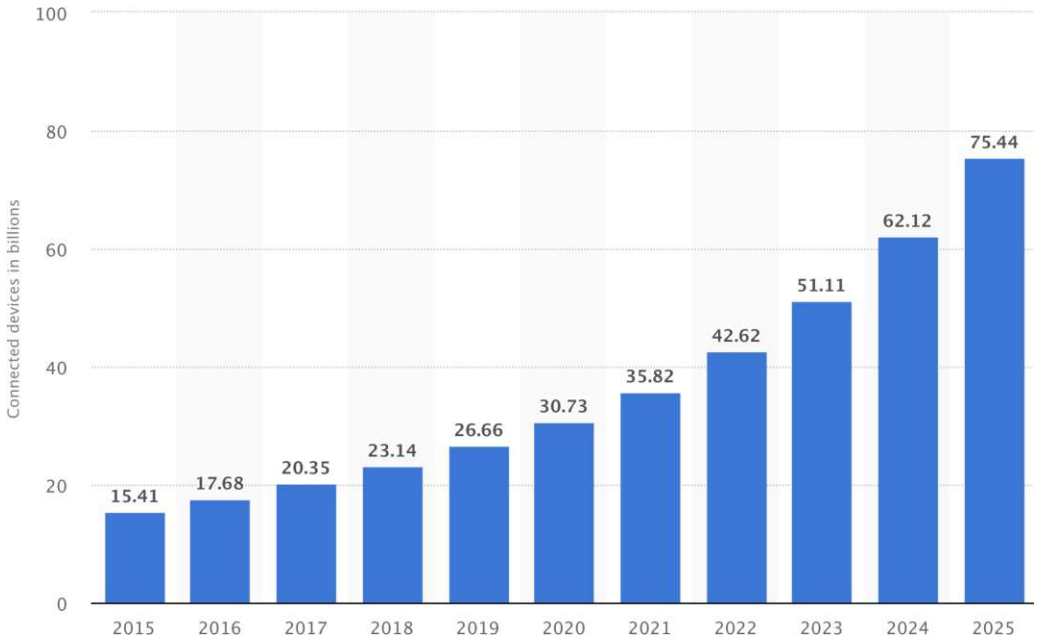


Figure 1: Evolution of electronic devices [1]

There are 7600 devices being connected to the internet every minute. That's 328 million every month. There is an estimation that 50 billion devices will be connected to the internet by 2023 (Figure 1) . In this pace, there could be 500 devices in each home by that time and all of these devices could be a possible target of attackers.

Cybersecurity is becoming more and more relevant with the growth of the internet. Along with this increase, more people are starting to replace the usual objects in their life with smart devices in order to make it easier, which makes the cybersecurity more relevant. The number of malicious people as well as software and hardware are rising and there are more attacks on people, businesses and governments for financial, political or activist purposes. In this context, cybersecurity is a critical component throughout the process of developing and maintaining a hardware or software product.

The security problem was brought to attention after the attacks over the computers belonging to governments, corporations or even persons. This is a problem which cannot be solved with a software or hardware fix due to the complexity and the multitude of products. Even more, these products are developed by different entities which would make a secure solution more difficult to achieve.

## II. Mobile Devices

Mobile devices have evolved exponentially in the past 12 years, from the time when the first smartphone was launched. Although it could be seen as a short period, it was enough to surpass Personal Computers (PC) market on certain segments [2].

Technology has been integrated in most of our activities, from simple to more critical tasks such as receiving a work related e-mail or making a remote bank payment. The software that is developed and runs on mobile devices is an operating system adapted for low power devices. One such example is Android, and it has been very feature oriented while the importance of security requirements decreased.

Following, it will be presented a classification of mobile devices and an overview about each subsystem in the market. This is important because each component evolved, most of the times, separately. Hardware developers usually don't build software for production and vice versa. Kernel (low level) programmers usually don't develop User Interface (UI) applications. Therefore, a systematic approach of each component is necessary.

### 2.1. Hardware

This chapter presents an overview of hardware evolution. It describes how hardware adapted for mobile devices.

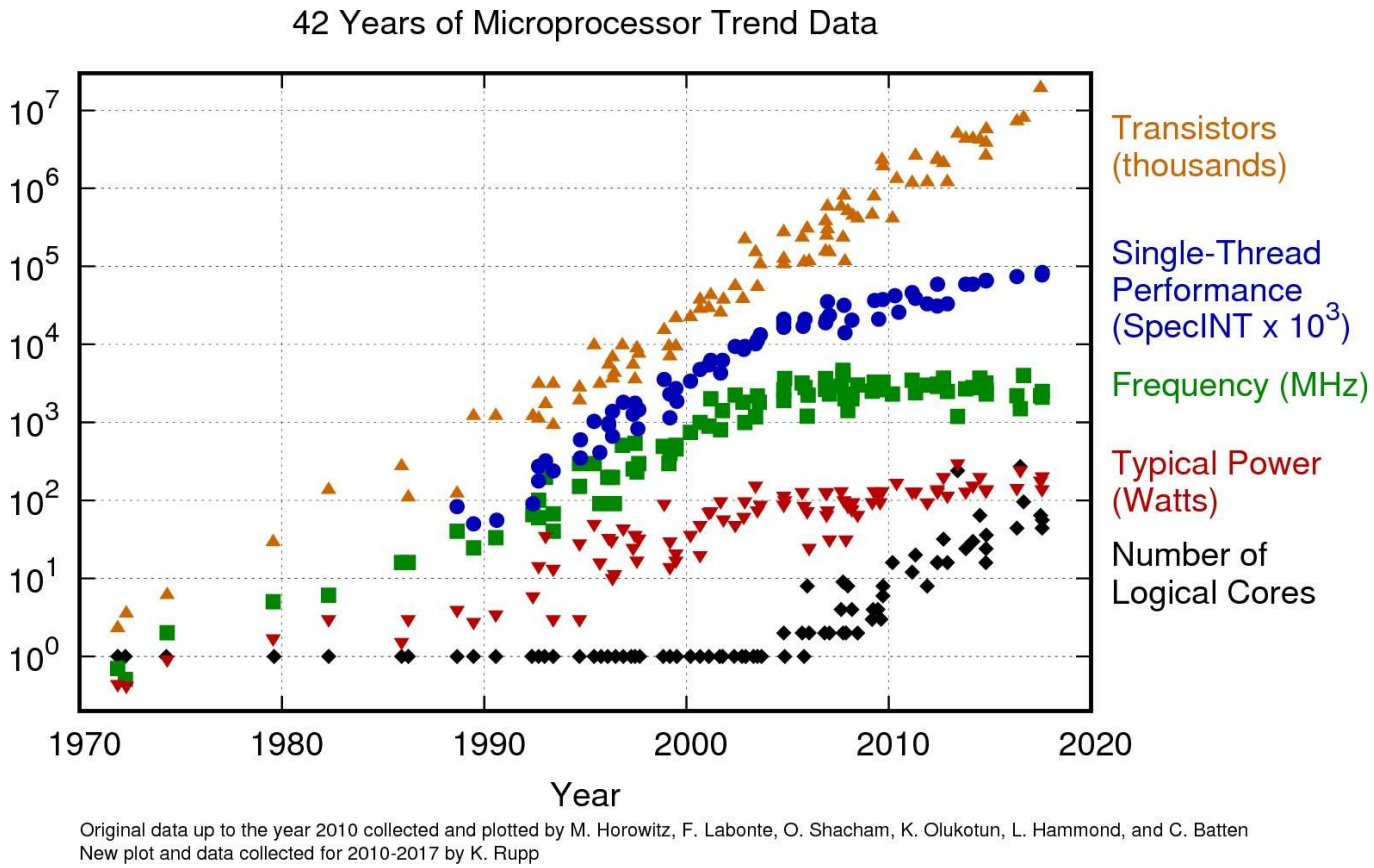


Figure 2: Evolution of Microprocessors [9]

Gordon Moore described in his paper [3] the future of integrated electronics. He predicted the last 50 years of hardware evolution. He's "law" states that the chip performance will double every 18 months (see Figure 2). This theory has been lately disputed and it's supposed to end in 2025 [3].

Hardware components had to be adapted to mobile devices specifications like dimension, power consumption, release of heat, cost of production and assembly, etc. The first smartphone was released in 2007 by Steve Jobs. Apple disrupted the mobile devices industry by creating a product that satisfied all the requirements mentioned above.

At first, industry focused the development of hardware based on what consumers needed: more power, more time between charges, smaller devices. Nowadays, with more attacks than ever, hardware producers are trying to add security features. From industry direction, this trend could be classified into two categories: hardware components designed for consumers and for developers. The components developed for users are mainly used for authentication: fingerprint scanner, hardware for face recognition. CPUs created especially for security are used by the developers to create secure applications.

Hardware Security Module (HSM) provides cryptographic keys for critical functions like encryption/decryption, authentication, etc. HSM has been widely researched [4] and used in automotive.

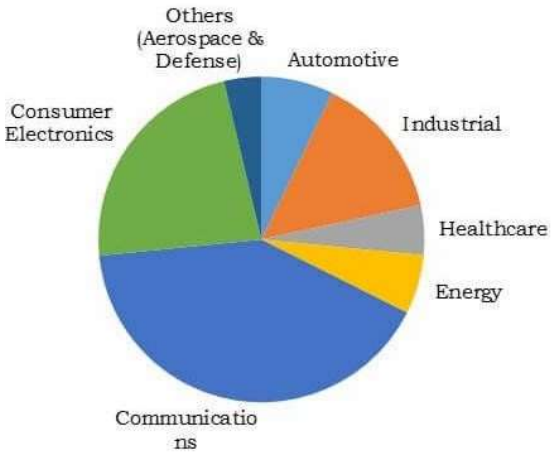


Figure 3: Global embedded computing market by end users*

Most devices contain components like: Central Processing Unit (CPU), memory - Random Access Memory (RAM), storage, batteries, screen, sensors, etc. Lately new hardware was created for functionalities (WiFi, Bluetooth) and security (fingerprint scanner, face detection hardware).

In this century embedded computing market was adopted by automotive, industrial, healthcare, energy, communication, consumer electronics and others (banking, transport, government, robotics, defense), as shown in Figure 3.

### 2.2. Software

This chapter presents an overview of software evolution in mobile devices. The software components are the easiest to change and adapt. Therefore, this is the most dynamic part of a system. If the hardware development cycle can take to a couple of years, the software can have a time to market (TTM) of months (Figure 4).


Figure 4: Software time to market

* Source: Primary and secondary research, and AMR Analysis

In order to achieve such a fast time to market, several development methodologies were created. These procedures had to follow certain phases of development (Figure 5), but each of them has different principles. The next chapter describes some of these techniques.

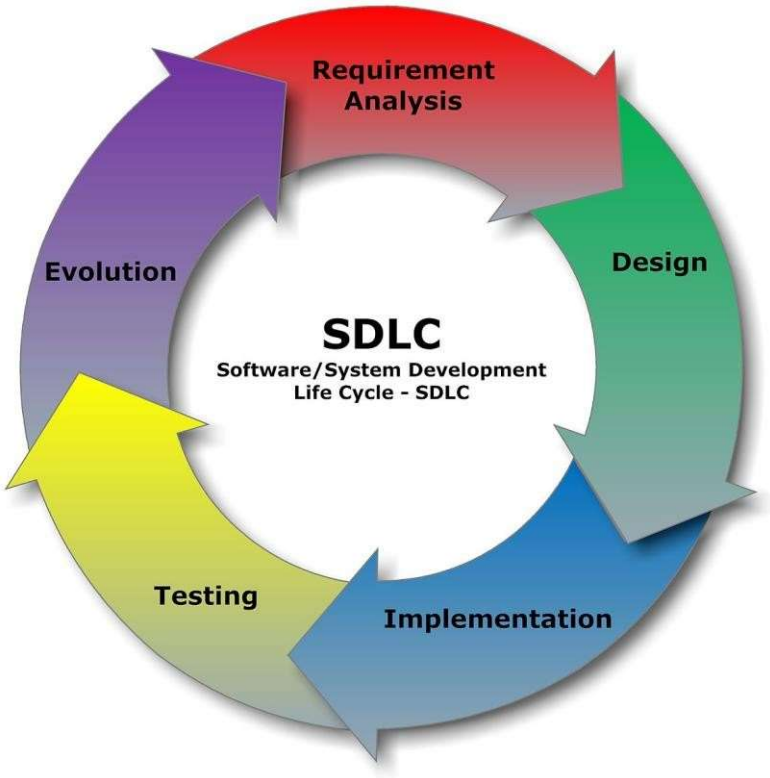


Figure 5: Software Development Life-cycle*

**Waterfall** model is a breakdown of the project modules (mentioned above and shown in Figure 5) into linear sequential phases. In [2] the author emphasis that phases with insufficient resources may be delayed; while, others with excess resources may be idled, leading to a bottleneck between the arrival and delivery of projects and to a failure in delivering an operational product on time and within budget.

**Extreme Programming and Agile Software development** are designed for requirements and solutions to evolve through the collaborative effort of self-organising and cross-functional teams and their customer(s)/end user(s). It advocates adaptive planning, evolutionary development, early delivery, and continual improvement, and it encourages rapid and flexible response to change.

In [6] the authors claim the methodology to be a solution to many of the problems that continue to plague software development projects.

In [5] Munassar et al. also describe **iteration**, **spiral** and **v-shaped** methods but they are hardly used nowadays. Moreover, they offer an overview of these methodologies and comparing the features as well as the flaws of each method.

The main software of embedded mobile devices is the Operating System (OS). This is the main key component that enables communication between the user/application and the hardware. It provides a scheduler so that all parties can access critical hardware resources: CPU, GPU, memory.

* https://commons.wikimedia.org/wiki/File:SDLC_-_Software_Development_Life_Cycle.jpg

The most dominating operating systems in the mobile industry include Google Android, Windows Phone, and Apple iOS.

**Android** is a Google's open source platform designed for mobile devices and was released in 2007. It is widely used mobile operating system in the handsets industry. The Android operating system runs on a Linux-based kernel which supports core functions, such as power management, network infrastructure, and device drivers.

**iOS** is the UNIX-based operating system first released in 2007. It is a universal OS for all Apple's mobile phones, such as iPhone, iPod touch, and iPad.

**Windows Phone** developed by a Microsoft, is a proprietary mobile operating system for Pocket PCs and smartphones. It was first introduced in 2010.

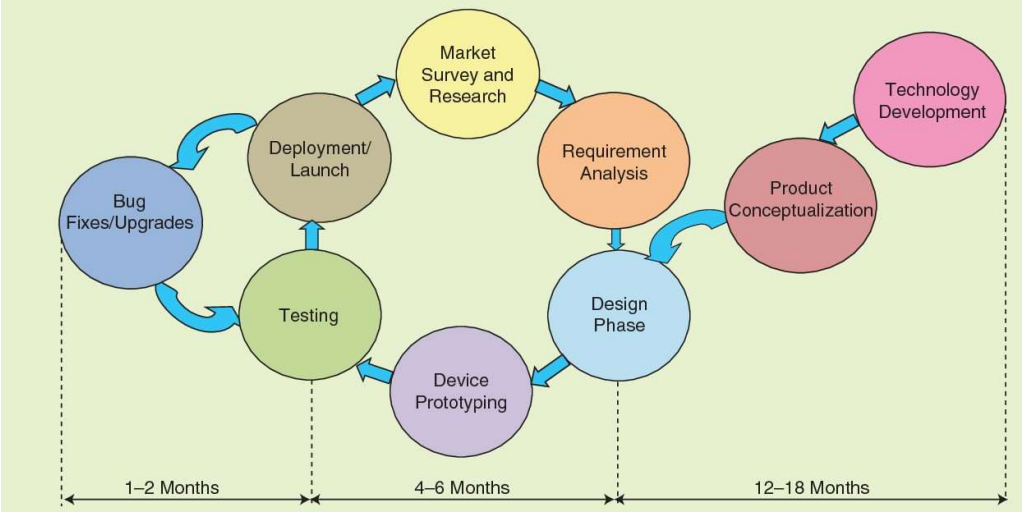


Figure 6: Software Development Life-cycle

Sometimes the Software Development Life-cycle (SDL) can look like Figure 6 where fixing bugs is a part of the developing process. Actually, a lot of vulnerabilities are based on this fact and it is described in the next chapter (Security Overview). check the following list.

## III. SECURITY OVERVIEW

### 3.1. Introduction

Although, in the beginning developers were more concentrated on the features, nowadays the security requirements are an essential part of the development process.

As security wasn't taken into consideration in the first steps of development, many vulnerabilities exist in the initial source code of the software that people are currently using. Therefore, there are an increasing number of attacks in cyberspace. In this context, companies and IT communities realized that security is a critical component throughout the process of developing and maintaining hardware or software products. Research [1] shows that the number of detected information security incidents have risen 66% year over year since 2009. A 2014 survey [1] underlines the total number of security incidents detected by respondents grew to 42.8 million around the world, up 48% from 2013 – an average of 117,339 per day. In *A comparative study of the evolution of vulnerabilities in IT systems and its relation to the new concept of cloud computing* [7], the author provides a classification of the vulnerability types emerged with time as information technology (IT) systems evolved.

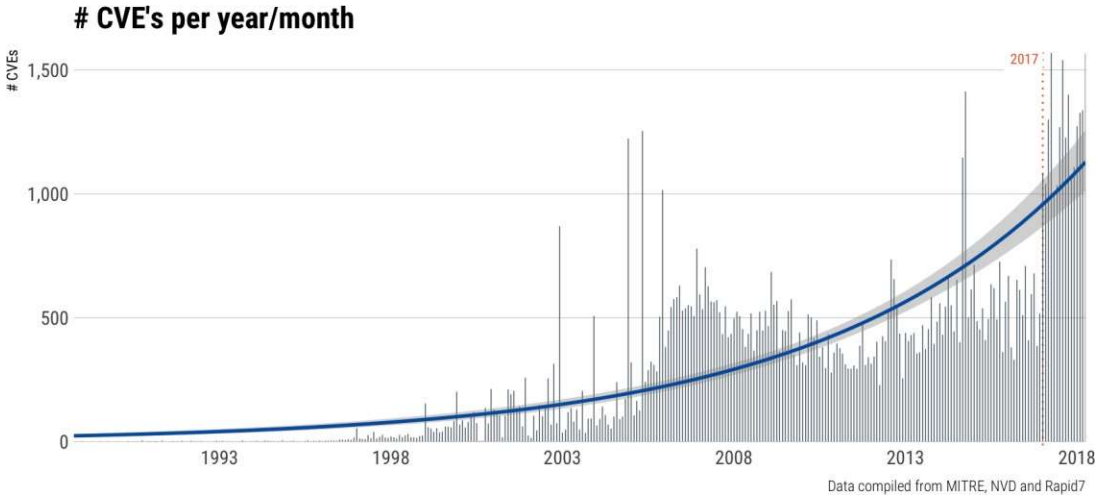


Figure 7: Number of CVEs per year

The Common Vulnerabilities and Exposures (CVE) system provides information for information security vulnerabilities that are publicly known. An evolution throughout the years is presented in Figure 7.

### 3.2. Security Motivation

This chapter highlights the importance of security and how to achieve it. It provides a set of guidelines and best practices. The Security objectives of a system should satisfy the following conditions: privacy, safety, anonymity, integrity, confidentiality and availability. A secure system should fulfil all the capabilities mentioned above. A product, as a whole (hardware and software) is rarely build by the same people and therefore, there should be a perfect communication between hardware capabilities and software implementation. The phases of development that a product usually goes through are shown in Figure 8.

Following, this chapter presents a list of security principles that should be followed when designing a system. These guidelines should be applied to the whole system, not only per component.

In *Guidelines for usable cyber-security: Past and* present *[8]*, Nurse et al. provide an up-to-date review of pertinent cyber-security usability issues and evaluation techniques applied to date.

In order to maintain security of a product, the creator must consider that security is a process, not a product. Continuous assessment is necessary as far as security goes. All declared vulnerabilities should be immediately fixed and all 3rd party software should be updated.

Moreover, best practice protocol states that more layers of security should be enforced. This procedure is called **Defence in Depth**. The principle states that if a security layer is breached, another layer should be in place as a fallback. This guideline assumes that there is no safe software and a vulnerability can occur any time.

Furthermore, **privilege separation** states that a user should have only the permission he needs. This way, if an user is or becomes malicious at any time, the damage is limited. **Least privilege** is a similar principle which recommends that user should have the minimum amount of control in a system.


Figure 8: An overview of Information Security Management

## IV. Security Processes

### 4.1. Prevention

In this chapter, I will describe the processes that are recommended to prevent a security breach and should be enforced as a basic configuration of any system.

**Limit Privilege** - a user should not have any more privileges than he needs. **Provide Defense in Depth** - multiple solutions should be applied for securing every system. These should be complementary solutions, but also overlapping is good. **Secure the Weakest Link** - the most attention, effort and money should be spent on the weakest point in a system. An often mistake is that engineers and system administrators are putting a lot of effort in components that are not easily accessible to attackers. **Embrace Simplicity** - secure system aren't always complicated system. **Choke Points** - are entry points in a system that can be easily monitored and secured.

### 4.2. Detection And Response

The **detection of attacks** is the most important step. The sooner the attack is detected, the better the chances to limit the damages. In this phase, the administrator should manually or automatically look for signs of attack. This includes, logs, alerts from IDSs and firewalls and any component that may fail during an attack.

The next phase is to **respond to attackers**. After the detection, the administrators should look for any vulnerabilities in the system and close (fix) them as soon as possible. The incident should be investigated by inspecting the logs and alerts from the firewalls (IDSs). Security is a process and requires continuous monitoring.

## V. Security Analysis

This chapter presents method and procedure related to security analysis. There is no general method for all the targets. Depending on what module is analysed, there can be several ways of securing the target. **Secure by construction** prevents the existence of bugs and vulnerabilities, **secure environment** prevents exploitation of vulnerabilities while an **isolated environment** provides damage control. Figure 9 reveals what is the cost of discovering errors based on the time of discovery.

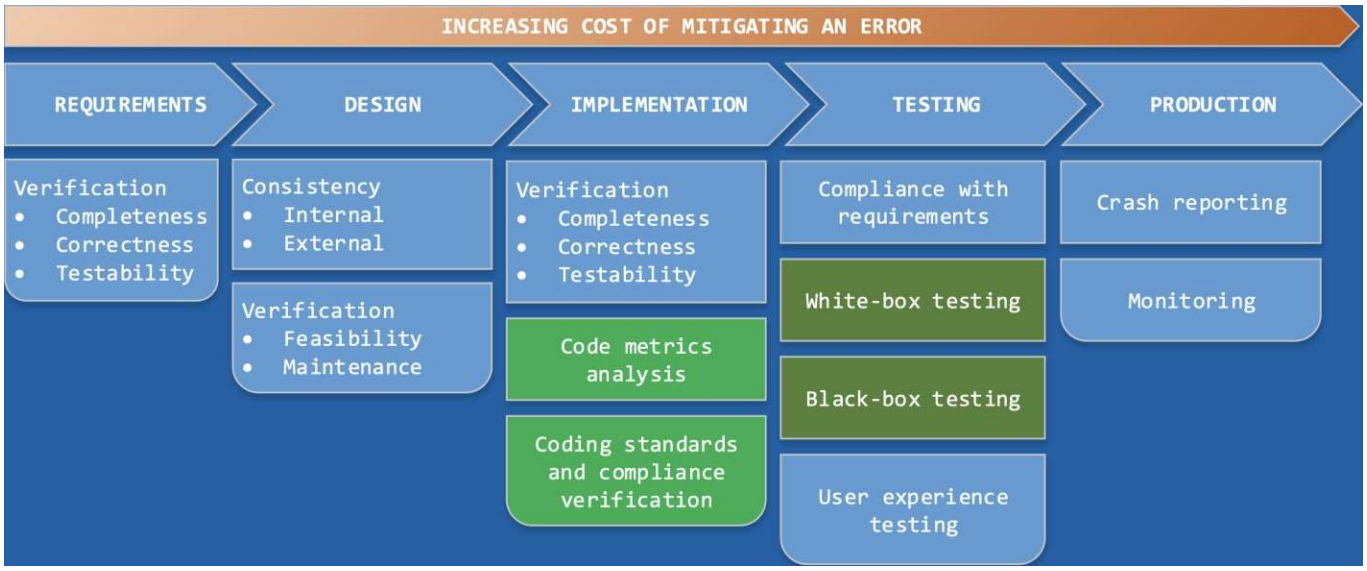


Figure 9: Cost of mitigation errors

### 5.1. Code Analysis

Code analysis expects a few steps, which will be described following. **Program comprehension**- most of the times it is required to understand the code. **Code review** - in this section the developer is fixing obvious mistakes, improves code quality and applies best programming practices. In the **code auditing** phase, usually, a third party performs a comprehensive analysis with intent of discovering bugs. **Static analysis** implies to analyse the code with executing it and it's usually performed on the source code. There are some automated tools that can be used in order to achieve better results.

### 5.2. Fuzzing And Symbolic Execution

*"Fuzzing or fuzz testing is an automated software testing technique that involves providing invalid, unexpected, or random data as inputs to a computer program."*

Fuzzing can breakdown to 5 steps. The first step is to start a program with an input. The second step is to feed inputs test to the program. The third step is to monitor program behaviour. The smart fuzzers, in step four, generate new test cases based on the results of the third step. The last step is to repeat steps two, three and four until some unexpected behaviour happens.

### 5.3. Information Flow Security

Information flow (IF) is the process where data transfer from a variable or location to another in a system or in a process. This procedure show how information is propagated. If data is classified in a system as low and high privilege, certain rules can be applied: no transfer of data from high to low privilege. Another examples are: no data from root to non-root processes, no data from unsandboxed to sandboxed processes, no data from system to 3rd party application.

### 5.4. Software Verification

The scope of software verification is to assure that the target fully complies to the specification. In order to perform a software/hardware verification, a system model is created considering the security specifications. Formal verification deduces properties from the system model, represents proprieties in a theorem prover and verifies that *system S fits a given property P*.

## VI. Conclusion

This paper presented an overview of the past and current state of mobile embedded devices. The first chapter describes how the industry evolved and introduces a first problem: security. The second chapter focuses on mobile devices. It starts with a brief classification and shows the evolution of hardware and software. The next chapter provides a security overview and presents the need of secure devices. Forwards, security principles and guideline are defined, followed by an example of security process: prevention, detection and response.

The motivation for this report is to understand how mobile devices evolved, and what's the role of security in this evolution.

Therefore I propose a research question: *How security should evolve in the next years?*

## ACKNOWLEDGMENT

The work has been funded by the Operational Programme Human Capital of the Ministry of European Funds through the Financial Agreement 51675/09.07.2019, SMIS code 125125.

## REFERENCES

[1] A. Tanweer, “A Reliable Communication Framework and Its Use in Internet of Things (IoT)”, 2018

[2] Youssef Bassil, “A simulation model for the waterfall software development life-cycle”, CoRR abs/1205.6904, 2012

[3] Suhar Kumar, “Fundamental limits to Moore’s Law”, 2015

[4] Marko Wolf and Timo Gendrullis. Design, implementation, and evaluation of a vehicular hardware security module. In Howon Kim, editor, Information Security and Cryptology - ICISC 2011 , pages 302–318, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[5] Nabil Mohammed Ali Munassar and A Govardhan. A comparison between five models of software engineering. International Journal of Computer Science Issues (IJCSI), 7(5):94, 2010

[6] Lowell Lindstrom and Ron Jeffries. Extreme programming and agile software development methodologies. Information Systems Management, 21(3):41–52, 2004

[7] Kouatli Issam. A comparative study of the evolution of vulnerabilities in it systems and its relation to the new concept of cloud computing. 20(4):409–433, 2019/12/10 2014

[8] J. R. C. Nurse, S. Creese, M. Goldsmith, and K. Lamberts. Guidelines for usable cyber-security: Past and present. In 2011 Third International Workshop on Cyberspace Safety and Security (CSS) , pages 21–26, Sep. 2011.

[9] Karl Rupp, 48 Years of Microprocessor Trend Data, https://github.com/karlrupp/microprocessor-trend-data.