|  |  |
| --- | --- |
| Mitigating software vulnerabilities | November 10  2018 |
| Managing a successful computer project | |
| K.P.I. Shenesh Perera  IDM | |

Contents

[Project analysis 3](#_Toc531344955)

* [Scope 3](#_Toc531344956)
* [Objective 3](#_Toc531344957)
* [Work breakdown 3](#_Toc531344958)
* [Project principles 4](#_Toc531344959)

[Finding Report- Oct 16th 2018 5](#_Toc531344960)

* [Incorrect implementations of data structures. 5](#_Toc531344961)
* [Solutions 6](#_Toc531344962)

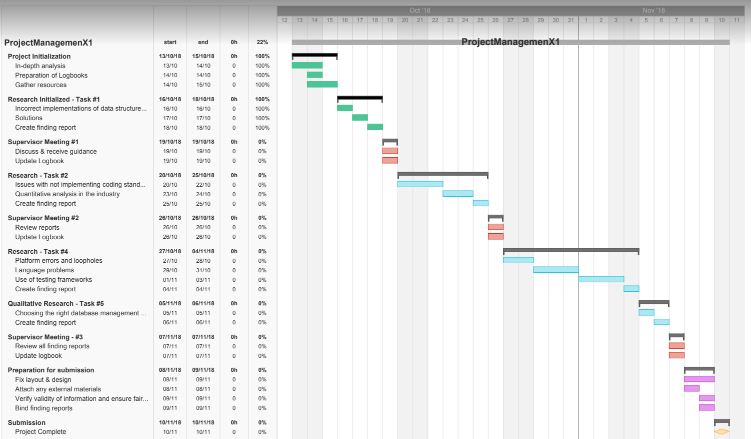
[Finding Report- Oct 25th 2018 7](#_Toc531344963)

* [Issues with not implementing coding standards 7](#_Toc531344964)
* [The Solution: a Coding Standards Document 7](#_Toc531344965)
* [How exactly do you enforce a coding standard? 8](#_Toc531344966)

[Finding Report- Nov 04th 2018 9](#_Toc531344967)

* [The issues in the language of the web 9](#_Toc531344968)

# Project analysis

****

## Scope

The primary scope of this project is to properly identify what vulnerabilities are created during development and how to effectively counter said problems. As such small scale research will be done both qualitatively and quantitatively in order to satisfy the scope of this project.

The Gantt chart above illustrates the path the project will follow at a glance.

## Objective

During this project all the information that has been obtained through research will be documented below as findings. Findings will contain not only about problems but also solutions and recommendations to avoid said problems.

Upon the completion of the project, we hope to bring awareness to the developer teams in such a way that the new banking project will follow suit with as less vulnerabilities in the system as possible.

## Work breakdown

Initially as project manager, I analyzed the project thoroughly and extracted what information I can use to equip myself with in order to narrow down the margins during research. Then I gathered or created the necessary resources I required, such as the software. An empty logbook template was designed by my hand as the one I was provided with was a hassle to maintain.

Afterwards, I did a simple research to discover what causes vulnerabities in software, and then decided on doing quantitative and qualitative research depending on the background of the specific topic.

## Project principles

All these steps have been taken in order to critically organize my workflow. Project management tips and standards have been followed; this document will bear witness to their usage.

I have employed quantitative and qualitative research methodologies that will satisfy the scope of the project.

*Qualitative research interviews*

I performed one to one interviews with my supervisor in order to gather any information I could about the topics I’ve already picked. I ensured that I kept an open mind and never came close to a conclusion. I did this as it’s a great opportunity to get a deeper understanding of a user’s point of view regarding a product, service or company.

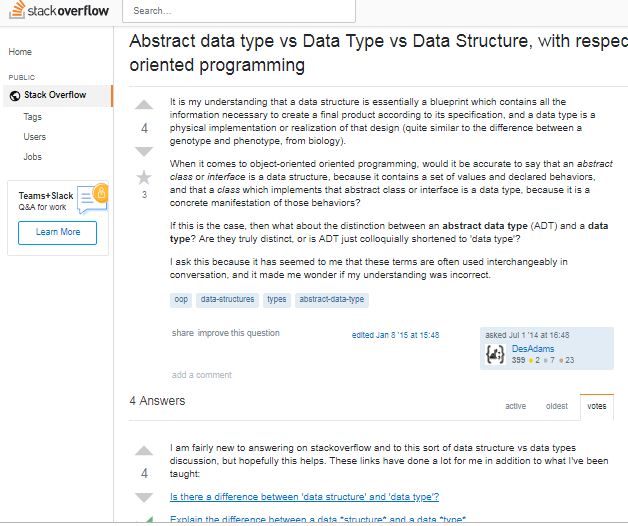
An interview allows you to ask follow-up questions to delve deeper, but this brings with it its own hazards too. A careless interviewer can potentially bias interviewee answers by asking leading questions that don’t subsequently provide a true reflection of a user’s opinion.

*Qualitative case studies*

Case studies provide projects with an even more comprehensive understanding of how an individual interacts with a product or service. It gives you a more complete picture of their satisfaction, usage and attitudes towards a specific product, service or industry in appropriate context.

**Figure 1.0**, stackoverflow.com

*Note: This is not my question in the forum*

*Quantitative research performed with existing forums*

This personally allowed me to identify the topics clearly, and why one should conduct research in order to produce a report with related to said topics.

Referring to forums like stackoverflow.com can help quantitative research on topics very simple.

I highly recommend developers to refer stackoverflow during times of trouble as this forum contains answers to plenty of development related questions.

# Finding Report- Oct 16th 2018

## Incorrect implementations of data structures.

A

*Data structure* is a mechanism for representing data in your programs. Many of you have heard terms like linked list, tree, and array. Each of these is logical representations of data that correspond to some architectural structure of what you're trying to represent.

One of the most common mistakes I’ve found upon researching about programmers, both experienced coders and newbies alike is paying too little attention to data structure choice.

Since almost all your code builds on your choice of data representation method, choosing the wrong data structure can have costly implications down the line.

Here's one example that illustrates this sort of design error: choosing a simple stack or queue, instead of a circular queue. Think of a stack as a stack of dishes. You put the bottom dish down, then another dish on top, then another, and so forth.

If you want to remove a dish, you take it from the top of the stack. This is called *last-in, first-out.* The problem is, if you need to remove something earlier in the stack, it's a hassle. Let's say you have ten dishes in the stack. To get to the first one, you have to remove all the others first.

Let’s take a queue for the sake of explaining; when you stand in line, you're in a queue. The first person in is also the first person out. As soon as the first person has been taken care of, the next person is up, and that person is served. The other thing that happens is that each person takes a step forward, moving up in the queue.

What happens when too many people show up? They're either turned away or the line goes out the door. And when the first person is called, all these people have to move.

When you have a lot of data, a queue of this sort can be enormously inefficient. Each time data is pulled from the beginning of the queue, all the data needs to move.

In this context, it might be better to implement a circular queue. In this case, the data never moves. Instead, a pointer is set up to point to the beginning and end of the queue and, internally, the queue wraps around itself, so that the data is organized in a ring instead of a line.

When a data element is used and eliminated from the ring, there's no need to move all the data in the ring. All that happens is the first element pointer points to a new element in the ring.

This is but one of many examples of how the choice of correct data structure can have enormous implications on the efficiency and effectiveness of your code.

Regardless, the poor choice of a data structure will lead to bad results at any time it will create a mismatch between

* The operations the business most often needs to perform on its data, and
* The operations the selected data structure allows to be performed most efficiently.

What will happen in either case is that the performance of the system will rapidly degrade as the size of the data involved grows (which happens naturally, in most cases, as the business itself grows), with the risk that by the time a decision is made to correct it the system will be too poorly understood or the quantity of data involved too unmanageable for there to be a straightforward, cost-effective way out of the situation.

*As such are there cases when the choice of data structure doesn’t matter?*

Yes, in one case: when the quantity of data involved is sufficiently small. In this scenario almost any data structure or algorithm will perform about as well as any other, with ironically the least optimal choices sometimes performs the best due to their lower fixed overhead: It can be faster to check every element in a very short list than to invoke a hash function even once, for instance.

Be aware this actually creates a problem for inexperienced developers using unit tests to guide their work as they often rely on very small, often hand-made sets of test data.

At this scale pretty much any implementation will appear to perform well, so the tendency (especially if an impatient manager is hovering nearby) will be for the programmer to pick the simplest, most easily implemented solution, even if it is destined to fail on the volumes of data encountered in the real world.

If this error isn't caught by a code review or integration test and the solution makes it into production, it can become very costly to correct.

## Solutions

1. Employ Testing frameworks with respected to the development platform you are using.
2. Research thoroughly about a multiple data structures, do sample tests and use the data to accurately pinpoint possible candidates.
3. Do not ignore compiler warnings/errors. Look into them especially if it’s with regard to the data structure.
4. Expose your implementations for regular code review.
5. Write pseudo code before implementing a data structure to help plan the implementation properly.
6. Assess the efficiency of the algorithm with respect to the data structure using Big O.
7. Write unit and integration tests before deployment or finalizing the code base for production.
8. Attending coding meetings to learn more about different ways a data structure could be implemented.

# 

# Finding Report- Oct 25th 2018

## Issues with not implementing coding standards

M

aintainable code has been a concern since the beginning of programming despite what language is concerned. It is highly straining and extremely a hassle while having to maintain large code bases that doesn’t make its purpose or intentions adequately clear.

Looking through unfamiliar code is much easier if it is laid out well and everything is neatly commented with details that explain any complicated constructs and the reasoning behind them.

During the development cycle, having to deal with anomalies and pointless issues is a waste of time as such the solution is using a coding standard.

**The Problem**

When we learn a new language, we usually begin to code in a specific style. In most cases, we’ll write in a style that we want, not one that has been suggested to us.

How is lacking a coding standard a vulnerability, you may ask. The lack of coding improperly and not following a standard can lead to a lot of vulnerabilities that will be pushed through reviews and into production which will later on severely affect the reputation of Asia Development. During the designing of the coding standard you can implement rules that will disallow teams from producing code that will contain vulnerabilities.

But once we start to code using a particular style, it will become second nature — we’ll use that style in everything we create. Such a style might include the conventions we use to name variables and functions ($userName, $username or $user\_name for example), and how we comment our work. Any style should ensure that we can read our code easily.

However, what happens when we start to code bigger projects and introduce additional people to help create a large application? Conflicts in the way you write your code will most definitely appear.

Then the code base will become polluted and when a new developer joins, he/she will have trouble deciding which standard to follow which cuts into the time available for the project.

More time will be wasted on understanding how to code than coding itself.

## The Solution: a Coding Standards Document

A coding standards document tells developers how they must write their code. Instead of each developer coding in their own preferred style, they will write all code to the standards outlined in the document.

This makes sure that a large project is coded in a consistent style — parts are not written differently by different programmers. Not only does this solution make the code easier to understand, it also ensures that any developer who looks at the code will know what to expect throughout the entire application.

Coding standards are great — but how do you decide which standards you want to apply, and how they will be defined?

*When you formulate your ideal coding style, you should think about these points:*

1. Can you actually read the code? Is it spaced out clearly?
2. Do you use tabs or spaces?
3. Will you follow common naming conventions like “camelCase”, if you don’t what alternative will you follow?
4. Are you using indentation to show where control structures (if, else, while and other loops) begin and end, and where the code within them is?
5. Are your variables naming conventions consistent throughout the code and do they briefly describe that data that they’ll contain?
6. Are functions named in accordance with what they do?
7. If you come back to the code in a few weeks or months, will you be able to work out what’s happening without needing to look at every line?
8. How are you commenting the work? Do you use any plugins?
9. Will you use IntelliSense features in your development environment? If so which plugins?
10. Have you used complex language functions/constructs that are quicker to write but affect readability? If so are you making sure that they’re properly commented?
11. Finally and most importantly, are your standards avoiding all common pitfalls and potential to create vulnerability in the code?

Once you’ve considered those points, you can begin to draft your coding standards. Consult with your team members (if any) and compare how they code to your own style — you shouldn’t force total change upon everyone.

Compromise and incorporate elements of everyone’s style. If someone has been coding in a specific way for a long time, it will take a while for that developer to change to the new method. Developers will likely adopt the style gradually, just as an accent develops over time.

*What if your team denies the enforcement of a coding document?*

If this approach fails, then the best and the only other alternative are using a coding standard that has been established by the community of the language(s) the team uses. This could also be the ideal alternative if you do not have time or experience to make a coding standard.

It is highly likely that most of your team members follow a coding standard like this.

## How exactly do you enforce a coding standard?

Linters. A Linter is an external plugin or a native feature of the development environment you use. Linters allow the maintaining of a document that would store all the rules per your coding standard, when a developer disobeys a rule set in the document, the linter program will show the developer that they did so, and during scans the program will provide auto-fixing even for the simplest cases like indentation.

# 

# Finding Report- Nov 04th 2018

Since it is highly likely for Asia Development to be employed for web development projects as banking is heavily and rapidly moving towards the net to handle transactions, we have taken the liberty to do a research on the problems of the widely used client-side language- JavaScript.

## The issues in the language of the web;

JavaScript is supposedly a functional programming language. Actually, it’s multi-paradigm, mainly imperative with some functional capabilities and a smattering of object orientation, via object prototypes.

Unfortunately, JavaScript is also a dysfunctional programming language, imbued with a myriad of “warts” and “gotchas.” It lacks the discipline to be a serious software engineering language, thanks to loose typing and freewheeling coercions, and their wildly inconsistent semantics.

JavaScript doesn’t even have a proper integer type, all numbers are treated the same be it float or unreal. JavaScript also doesn’t have a proper array type like you find in most normal programming languages. This has been significantly averted during the implementation of Sets and Weak Sets.

Just to be clear, despite ECMA working hard on ECMAScript to fix JavaScript, it has yet to address the abovementioned problems. For instance the simplest of all primitive data types, the Boolean cannot be properly captured to identify whether or not if it is different from “undefined”.

To solve all type-related issues, a language called Typescript evolved. This will allow you to mitigate type-related issues in JavaScript, and therefore any and all errors and vulnerabilities in JS.

JavaScript apologists are constantly telling us that all programming languages have warts. What they don’t tell us is that most languages have far fewer warts than JavaScript has. How big is the problem for this language?

Even through the introduction of ES6 (ECMAScript2015), more problems to worry about were introduced to the language in an attempt to solve previously existing issues. ES6 is the modern day JS standard, while industry is slowly adapting to ES7 as well.

*Are there similarly long lists for Java or Python?*

These same apologists also tell us that by applying good practices, including linters, you can avoid the bad parts, and you’re left with programming joy.

Isn’t that what Douglas Crockford’s book is all about? This is a fairy tale, of course. Good practices are not a perfect solution. Sooner or later, JavaScript’s loose typing and inconsistent semantics will bite you in the ass. Sooner or later, the lack of integers or proper array type will prove annoying.

These things are like land mines, lying in wait with infinite patience to blow off your legs. Good practices mitigate but do not eliminate the problems.

The need to be constantly aware of the bad parts most definitely places an added and unnecessary cognitive burden on the programmer.

*So why do JavaScript proponents continue to promote its use?*

The answer is; JavaScript’s large user community, ecosystem of tools, libraries and the large number of jobs available for JavaScript developers. This answer also explains the longevity of all the mainstream languages, irrespective of their faults, including: PHP, C++, Perl, and Visual Basic (languages everybody loves to hate)

The community and ecosystem argument is certainly quite compelling. There is no doubt that these things confer substantial benefits for both businesses and programmers.