

---

## **COMP1833: Project Planning, Academic Essay and Quality Plan**

---

Trevor Kiggundu: 001001720

December 2020

MSc Computing and Information Systems

---

A report submitted in fulfilment of the requirements for the module, Software Quality Management,  
Computing and Information Systems Department, University of Greenwich.

---

## Table of Contents

<i>1</i>	<i>Project Planning Case Study</i> .....	<i>3</i>
1.1	Group Requirements Specification .....	3
1.2	Group Design Work.....	5
1.3	Financial Predictions .....	6
1.4	Social, Legal and Ethical Review of the Project .....	7
1.5	References.....	7
<i>2</i>	<i>Quality Assurance Plan</i> .....	<i>9</i>
2.1	Product Introduction .....	9
2.2	Process Description.....	9
2.3	Product Plans .....	10
2.4	Time Management .....	12
2.5	Risk assessment .....	13
2.6	Change control.....	15
2.7	Review .....	16
2.8	References.....	17
<i>3</i>	<i>Academic Essay</i> .....	<i>18</i>
<i>4</i>	<i>Appendix</i> .....	<i>22</i>

# 1 Project Planning Case Study

## 1.1 Group Requirements Specification

A small chain of supermarkets called ‘MAST-Local’ have hired the team to provide an information and electronic point of sale system for store managers to monitor sales, manage stock, and place orders much more efficiently. In addition to this, a special request has been made by the company to digitize some of the processes that were originally done in store/on paper. Some of these processes include but are not limited to automatically ordering new products online at any time, as opposed to doing so through the delivery driver once a week, integrated accounts and personnel systems, and the possibility of customers being able to order online as well.

### 1.1.1 Stakeholders

In order to fully understand the scope of this project, the team has been given a list of key stakeholders and their potential roles in both the old and new system. Defining stakeholders is important, as it allows for the creation of goals and objectives that the specific business wants to achieve. Businesses often have numerous internal and external stakeholders, which can be good, as a lot of people are interested in the success of the business. However, more often than not, having a variety of stakeholders results in different “interests and priorities” (BBC, 2020), and these interests can conflict. Luckily for our company (and unlucky for them), ‘MAST-Local’ had already gone through the process of trying to digitize their system once before and failed, meaning that they know not only what they want from us, but also what they do not want from us. Our company will make sure not to make mistakes that others did in the past regarding issues such as timescale and budgeting issues. Listed below are those stakeholders for the MAST-Local system:

Stakeholder	Type	Role
Michael Peterson	Internal	CEO of MAST-Local
John Hacker	Internal	Director of resources and logistics
George Manning	Internal	Store manager of store that new system will be piloted on
Managers	Internal	Learn the new system and teach it to the employees
Employees	Internal	Use the new system to service

		customer needs effectively
Customers	External	Provide unbiased feedback on the success/failure of the new system
Bank/Lender	External	Fund the project
Software Company (Us)	Internal	Develop new IT system for MAST-Local
Head Office	Internal	Provides logistics for store managers to look at
Community	External	Provide feedback on business practices/local support (Store/Warehouse expansion, number/size of stores, environmental implications of these actions)

### 1.1.2 Non-functional Requirements

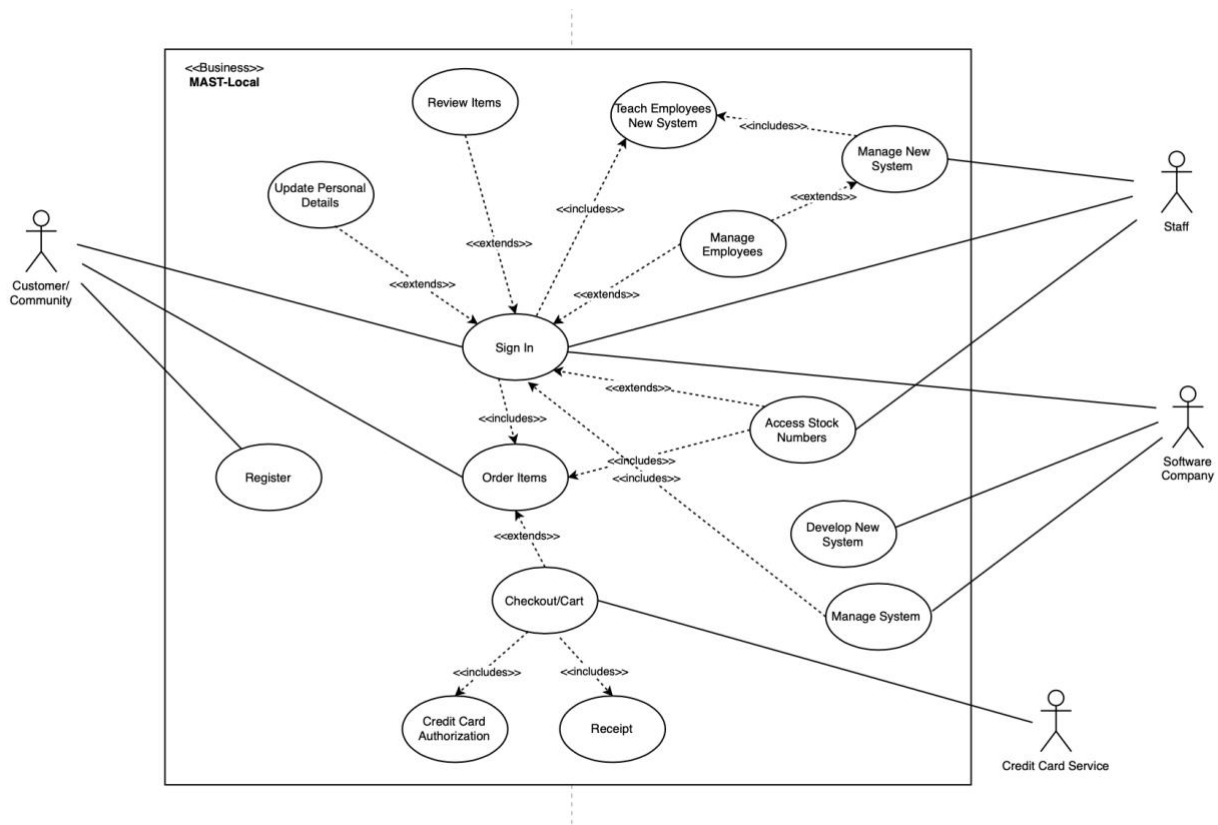
Non-Functional Requirement		MoSCoW Category
1.	Usability and Ease of Access: The new system must have a modern UX/UI design and be easy to understand and use.	MUST
2.	Interactivity and Innovativity: The new system should boast interactivity features similar to other real-world applications. However, it must also be innovative enough to replace the old paper-based system, as that is the main requirement.	SHOULD/MUST
3.	Privacy and Security: The new system must protect the personal information of all the stakeholders involved.	MUST
4.	Accessibility: The new system must be accessible to all users and take account of a wide range of circumstances, from operating system to the user's potential disabilities.	MUST
5.	Reliability and Performance benefits: The system must be much more reliable than the last expansion attempt and should be reliable enough to avoid flaws.	MUST/SHOULD

### 1.1.3 Functional Requirements

	Functional Requirement	MoSCoW Category
1.	System and integrated EPOS must work to begin with; previous attempts did not work.	MUST
2.	Well implemented UX/UI system design.	SHOULD
3.	Monitoring stock levels and allowing for stock orders to be made online.	MUST
4.	Loyalty schemes and customers ordering products online.	COULD
5.	Standardize electronic tills to store more than the basic list of items. Also allow for storage of details online rather than on USB drives/SD cards.	MUST

## 1.2 Group Design Work

### 1.2.1 Use Case Diagram



### 1.3 Financial Predictions

Making financial predictions when running a business is important as it allows for the preparation of the “best case, expected case and worst scenarios” (Boyd, 2020) of a proposed transaction. In order to create the Pro-Forma and CoCoMo cost estimations for the MAST-Local store system, two different types of research were done; Firstly, the average income and spending costs for a small supply chain store was calculated and added to the documents. After that, the average cost of upscaling a software system was also researched and applied to the diagrams. Given that the MAST-Local system is fictional, it was hard to decide which set of research and data to use. In the end, a decision was made to use the average data sets to make Pro-Forma estimates, and to use a checklist made by ‘Soltech’, a software development company, to define the possible expenses when upscaling a software system. According to their article, the average cost of updating old software falls “somewhere between the \$40,000 and \$250,000 mark” (Soltech, 2020). The CoCoMo estimations were made based on information given to the developers in the lectures as well.

#### 1.3.1 Pro-Forma and Cost Estimation

MAST-Local			
Pro Forma Income Statement for the EPOS development time period (6 months)			
<b>Sales</b>			<b>500,000</b>
Cost of Upscaling System		100,000	
Labor (equipment, materials etc.)		<u>250,000</u>	
<b>Total Cost of Sales</b>			<b><u>150,000</u></b>
<b>Gross Profit</b>			<b>150,000</b>
<b>Expenses</b>			
Developmental Costs			
Creative Design		12,000	
Designing to Budget		80,000	
Wages		50,000	
Maintenance		<u>5,000</u>	
<b>Total Operating Expenses</b>			<b>147,000</b>
<b>Net Income</b>			<b>3,000</b>

## **1.4 Social, Legal and Ethical Review of the Project**

There are a variety of social, legal and ethical implications involved within the process of upscaling a software system. It is important for developers to pay attention and prepare for these considerations, as failure to do so can lead to legal and financial troubles somewhere down the timeline of the project. The easiest way for these developers and companies to make sure of this, is to adopt the proper professional standards, and follow the codes of conducts set by professional bodies such as the BCS, ACM and the IEEE.

### **1.4.1 Social Considerations**

When developing software, developers must make sure that the work that they are producing is non-discriminatory in nature and can be used by any and everyone. The BCS (2020) code states that all members should “make IT for everyone” and respect consumers regardless of their backgrounds. The IEEE and ACM (2020) also respectively echo anti-discriminatory messages, stating that members should “treat all persons fairly and with respect” and to “take actions not to discriminate”. Failure to follow these guidelines can result in harsh disciplinary/monetary sanctions towards the developer.

### **1.4.2 Legal and Ethical Considerations**

When developing software, developers must ensure that the work that they are producing is done legally and ethically, holding up their end of the “duty to the profession” (BCS, 2020). Developers must also make sure to avoid “unlawful conduct in professional activities” (IEEE, 2020) as well as maintaining “high standards of professional competence” while working to complete their projects. All three member bodies, once again, echo the same message regarding maintaining a good reputation, and it is up to the developer to continue upholding those standards regardless of the obstacles they face while working.

## **1.5 References**

- ACM (2020) The Code affirms an obligation of computing professionals to use their skills for the benefit of society., *Acm.org*, [online] Available at: <https://www.acm.org/code-of-ethics> (Accessed 1 January 2021).
- BBC (2020) Influence of stakeholders on business objectives - Stakeholders - GCSE Business Revision - Other - BBC Bitesize, *BBC Bitesize*, [online] Available at:

<https://www.bbc.co.uk/bitesize/guides/z4gcd2p/revision/2> (Accessed 1 January 2021).

- BCS (2020) BCS Code of Conduct, *Bcs.org*, [online] Available at: <https://www.bcs.org/membership/become-a-member/bcs-code-of-conduct/> (Accessed 1 January 2021).
- Boyd, K. (2020) Pro forma financial statements, *QuickBooks*, [online] Available at: <https://quickbooks.intuit.com/r/bookkeeping/pro-forma-financial-statements/> (Accessed 1 January 2021).
- IEEE (2020) IEEE Code of Ethics, *Ieee.org*, [online] Available at: <https://www.ieee.org/about/corporate/governance/p7-8.html> (Accessed 1 January 2021).
- Soltech (2020) The Cost Of Updating Your Outdated Software, *SOLTECH*, [online] Available at: <https://soltech.net/cost-updating-outdated-software/> (Accessed 1 January 2021).



## 2 Quality Assurance Plan

### 2.1 Product Introduction

The product to be submitted is an academic essay researching ‘The Open Groups Architecture Framework’ (TOGAF); TOGAF is a software engineering practice that aims to “automate and integrate the processes between software development and IT teams, so they can build, test, and release software faster and more reliably” (Capel, 2020). This topic will be researched and critically evaluated in relation to Fred Brooks’ 1987 paper ‘No Silver Bullet’, which argues that despite the advances in hardware and software technology, there is no ‘silver bullet’ or single strategy that can solve all of the issues regarding large scale software development. This paper will, however, shine a light on the development issues that TOGAF does solve, in a bid to evaluate whether or not those solutions support or rebuff Brook’s idea that there is no single solution when attempting to solve issues regarding complexity, conformity, changeability and invisibility in software development.

### 2.2 Process Description

The purpose of the process description is to define the processes that have to be followed in order to produce the product.

GOAL	PROCESS	MEASURE
Conduct appropriate initial reading.	Read the coursework specification, as well as the lectures and given topics.	Ensure that the product requirements and expectations are understood. This can be done by skimming the specification until it is understood and asking peers/tutors for help if needed.
Create an appropriate time management schedule.	Set realistic goals for task allocation by creating a pert chart	Assessment of completion of goals within the given time. This will be measured by whether or not certain tasks are completed by their allocated time slot.

Choose a topic to research between ‘DevOps’ and ‘The Open Group Architecture Framework’.	After reading and understanding the specification, choose a technique to be researched.	Ensure that the topic is well understood and chosen in confidence by making relevant notes based on research.
Include at least one referenced paper from a leading computing journal such as the IEEE and/or Communications of the ACM.	Use the IEEE and ACM search engines to browse for appropriate journal papers.	Ensure this is fulfilled by including at least one journal that can be traced back to any of the leading computing or software engineering journal databases.
Proof-read and assess the final product	After producing a sample paper, self and peer review the final product.	This will be measured by discussing requirements with peers repeatedly and comparing the product with the coursework requirements. Asking the tutor for validation can also ensure that the product is on the correct track.

### 2.3 Product Plans

The purpose of the product plan is to define the criteria that should be included in the product, using the processes mentioned before in the process description.

GOAL	Process	Measure
Satisfactory level of referencing	Use the appropriate credible resources, such as Google Scholar, to find journals and papers in reference.	Ensure there are 5 or more credible academic journals referenced; this can be done by using citing tools such as ‘Cite This for Me’.
Correct referencing format	Use tools mentioned before such as ‘Cite This for Me’ to	Ensure that references are in accordance with the Harvard University of Greenwich

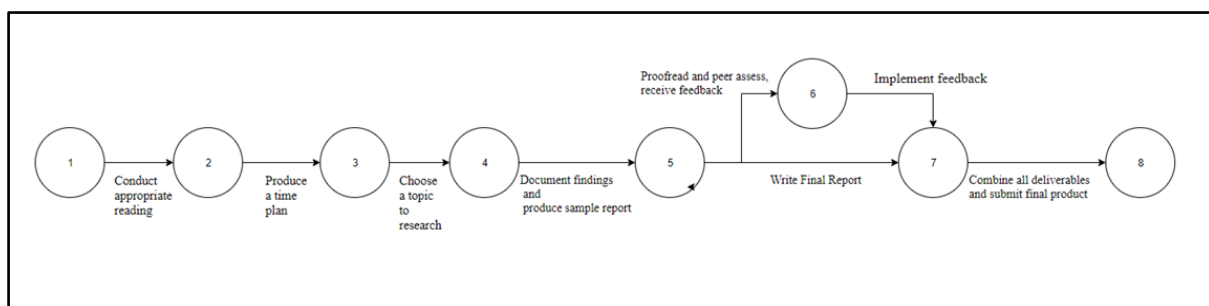
	<p>automate references and avoid formatting errors.</p>	<p>referencing system. Compare the paper to previous examples.</p>
<p>Reliability</p>	<p>Make reputable academic statements while writing the paper. Double check resources to make sure that they are credible as well.</p>	<p>This will be measured by evaluating how strong the statement made is, in addition to how strong the reference is (in terms of credibility). A more credible source means that the argument made based on the specific source is stronger. Once again, peer and tutor review will be used as a measure of understanding.</p>
<p>Correct document length</p>	<p>Make sure to acknowledge the word count requirements while writing the document. When using Google Docs, it is best to highlight individual sections and use the word count tool to verify the document length.</p>	<p>This will be measured by repeatedly using the word count tool to “test” the length of the document during and after the completion of the essay. the document should be between 1,500 and 2,000 words. Failure to reach the benchmark will result in the necessary changes being made (adding/subtracting words from count).</p>
<p>Readability</p>	<p>Make sure to proofread the final paper multiple times during and after the production of the final essay. Using the already given tools such as autocorrect and punctuation tools in order to catch often overlooked mistakes.</p>	<p>This will be measured by proofreading the document by myself and by using peers as well. Funnily enough, not all mistakes are caught by these tools, especially if different tools are being used (Google Docs vs. Microsoft Word). Mistakes can still be overlooked even after using these tools, so the best remedy is to repeatedly look over</p>

		the final product, with the help of multiple peers.
--	--	---

## 2.4 Time Management

The easiest and most practical time beginner management tool is a pert chart; it allows for the developer to create task allocation goals and objectives without any specific timelines to go along with them.

### PERT Chart

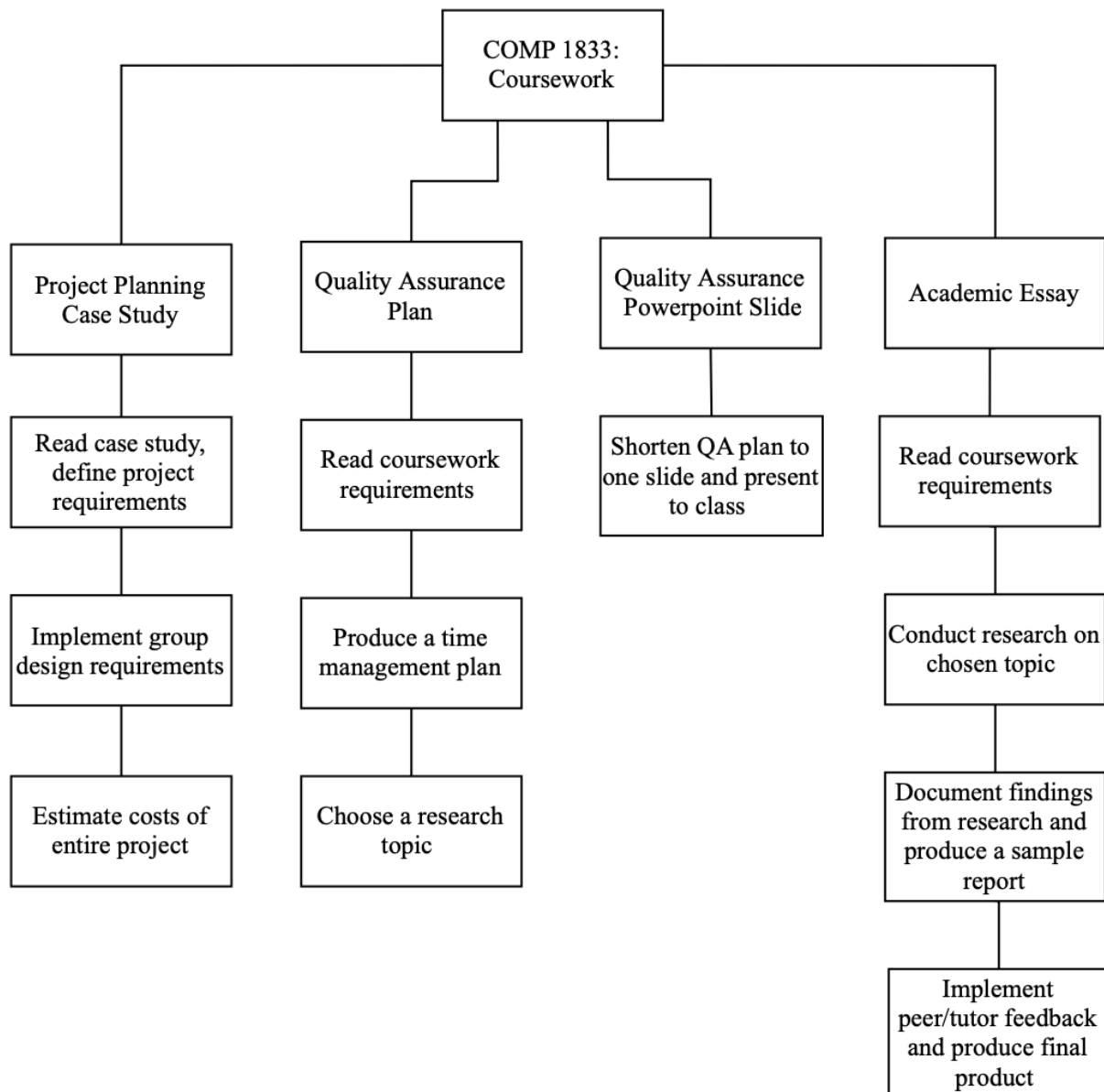


As the coursework material was finalized and further talked about in lecture, a more detailed Gantt chart and work breakdown structure with measurable goals was developed to show the full-time management plan. The plan starts as late as it does due to the fact that I switched into the module at a later date; having more time would have allowed for a better time management plan as well as more time allotted on the Gantt chart.

### GANTT Chart

Task	Timeline							
	Nov 2	Nov 9	Nov 16	Nov 23	Nov 30	Dec 7	Dec 14	Dec 18
Read coursework requirements								
Produce time management plan								
Choose a research topic								
Document findings, produce pre-report								
Receive peer and tutor feedback								
Combine deliverables and submit report								

## Work Breakdown Structure



## 2.5 Risk assessment

Five potential risks of different types have been assessed to depict an adequate representation of considerations made toward topics of risk identification, analysis and management.

### Risk Identification

Risk	Affects	Description	Risk Type
Lack of credible research papers	Product	Not being able to find suitable/adequate	Tools

		research papers	
Incorrect file format when submitting	Developer	Submitting the assignment in the wrong file format, clearly going against the rules in the specification	Technology
Change of specifications	Product	Changes in topic selection between DevOps and TOGAF.	Organizational
Insufficient time management	Product	Underestimating the time needed to produce the final product can lead to less than satisfactory work being submitted.	Estimation
Change of requirements	Developer, Product, Project	Change in coursework requirements due to the university not approving the coursework yet.	Requirements

### Risk Analysis

Risk	Probability	Effects
Lack of credible research papers	Low	Catastrophic
Incorrect file format when submitting	Moderate	Tolerable
Change of specifications	Moderate	Tolerable
Insufficient time management	High	Serious
Change of requirements	Low	Tolerable

### Risk Management

Risk	Strategy
Lack of credible research papers	Consider changing the topic as soon as possible and inform all relevant parties of the change. The effects of such a change are tolerable if handled in due time.

Incorrect file format when submitting	Make sure that the correct file format is chosen when first saving the file, this avoids problems later on in the life cycle of the project. Make sure to also check the submission format ahead of time.
Change of specifications	Ensure that the same level of effort and preparation is applied to the second topic as was to the first topic. Having a general understanding of both topics can lessen the impact of having to change topics later on.
Insufficient time management	Develop and follow suitable time management models and practices. Pert and Gantt charts will help in this regard, as well as making sure to prepare the relevant change control plans if needed.
Change of requirements	Implement the change control plans mentioned previously and start researching the new topic/assignment as soon as possible. The effects of such a change are tolerable if handled in due time. Ask the lecturer for guidance if needed.

## 2.6 Change control

One of the risks mentioned in section 2.5 has been identified as a suitable indicator of change control for the QA plan.

**Risk/Scenario:** Change of specification/change in research topic

### 1. Document the change request:

Ensure that the 'developer' is well informed regarding the specification change. Make an informal assessment regarding the potential difficulties that can arise due to the change (time management etc.).

### 2. Formal assessment:

Risk assess the potential difficulties defined in step 1 of the process; Consider the possible implications of those difficulties. For example, time management: will there be

enough time to restart and finish the project in time? Will there be a drop in quality?

Brainstorm the potential changes that might have to be made.

**3. Planning:**

Devise a plan and prepare for the upcoming changes to the project. Changes will also have to be made to the QA plan and reading material. However, there might be parts of both specifications that do not change; if this is the case, make sure to compare both documents ahead of time to avoid doing more work than needed, and in turn, saving much needed time.

**4. Designing and testing:**

Using the unchanged parts of the QA plan, write up a sample paper based on the new specification. Collect feedback by self and peer assessing the work, as well as receiving feedback from the tutor. Ensure that feedback is being received from reliable sources.

**5. Implementation and review:**

After receiving the proper feedback, rewrite the sample paper with both the proper feedback and the new specification requirements.

**6. Final assessment:**

Evaluate the success of the change of change control plan. Document any problems with the process and amend them for the next time.

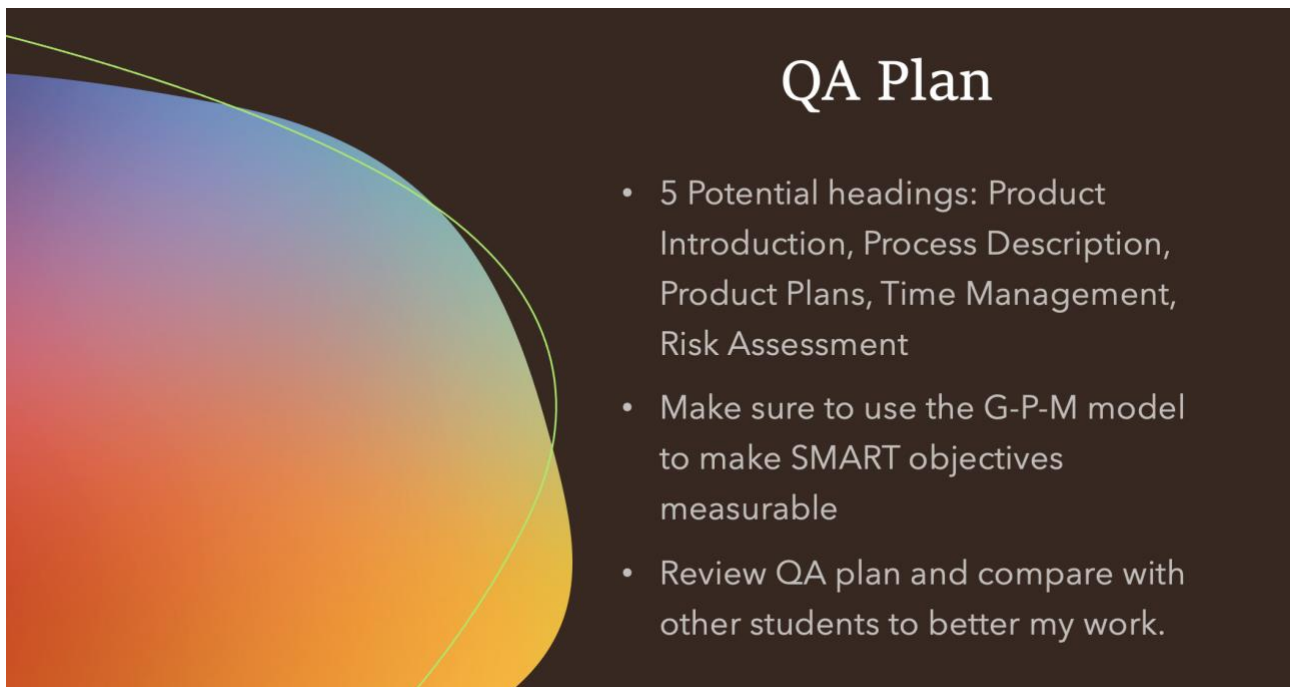
## **2.7 Review**

### **2.7.1 Review of Presentation**

#### **Aim of Presentation**

The purpose of the PowerPoint slide was to provide a short snippet of the ideas that were to be implemented to create a successful quality assurance plan. Essentially, it was created in order to ensure that us as ‘developers’ knew what the coursework task is. The slide was created to be quick to point and shared virtually with the class and lecturer so that we could all bounce ideas off each other. Personally, I felt like the one slide requirement was a bit restricting, as it was quite a large coursework task. However, I still followed through with the task as it helped to prepare myself mentally for the larger coursework task. Shown below is the PowerPoint slide.





## QA Plan

- 5 Potential headings: Product Introduction, Process Description, Product Plans, Time Management, Risk Assessment
- Make sure to use the G-P-M model to make SMART objectives measurable
- Review QA plan and compare with other students to better my work.

### 2.7.2 Review of Final Quality Assurance Plan

The final quality assurance plan was based on the tools and techniques that were taught via the lectures and tutorials, as well as the one slide PowerPoint presentation. Since all students followed the same specification, and were given the same sample paper, it is safe to assume that most students went with the same format when setting up their QA plan. When discussing the product and process descriptions, the Goal-Process-Measure (GPM) model was used to ensure that a significant amount of detail was written for each of the project goals, and that each of those goals was measurable. Time management goals were also supposed to be measurable according to the specification, so a pert chart and work breakdown structure alone would not be enough; to remedy this, a Gantt chart was created as it added a time frame via dates to the goals established in the pert chart. In terms of risk assessment, one of the textbooks that was recommended to us was very helpful in defining industry standards when managing information security. These standards (ISO 27001 and ISO 27005) were then used to develop the risk identification, analysis and management tables shown in chapter 2.5. External research was done to verify this information, as well as to collect information on how to properly document change control plans and requests.

## 2.8 References

- Brooks (1987) No Silver Bullet Essence and Accidents of Software Engineering, *Computer*, 20(4), pp. 10–19.

---

# A Critical Analysis of The Open Group Architecture Framework in Light of Software Engineering's Inherent Problems

---

**Trevor Kiggundu**

001001720

MSc Computing and Information Systems

University of Greenwich

tk9894h@gre.ac.uk

December 2020

## ABSTRACT

Despite the rapid and monumental technological changes that have happened between the 20th century and the modern day, one must not overlook the obstacles that have been hurdled in order to reach those milestones. A 1987 paper written by Fred Brooks titled 'No Silver Bullet. Essence and Accident in Software Engineering' does just this and discusses the challenges and limits regarding the software side of technological advancement. In his paper, Brooks states that in addition to the accidental difficulties that are caused by sources such as human error, there will always be four essential difficulties that plague every software system: complexity, conformity, changeability and invisibility. In addition to this, he also argues that there is no 'silver bullet' or single strategy to relieve software engineering of these problems, nor will there ever be one. This paper aims to challenge that theory by comparing Brooks' paper to The Open Group Architecture Framework (TOGAF), and by analysing its positives, negatives and overall importance within the software development process, a conclusion that either supports or debunks Brooks' view that there will 'never be a silver bullet' will be made.

**Keywords**— The Open Group Architecture Framework, TOGAF, Complexity, Changeability, Conformity, Invisibility, Silver Bullet, Software Development.

---

## 1 INTRODUCTION

The advances in software development pathways over the last 50 years have led to the development of more innovative, alternative and yet standardized practices when developing software. The Open Group Architecture Framework (TOGAF) is a direct product of some of these advances.

TOGAF is an award-winning enterprise architecture network that was first developed in 1995; the goal of the framework is to help enterprises develop software in the quickest and most cost-effective way possible. Major advancements like these, however, are also the main topics of papers like Fred Brooks' 'No Silver Bullet. Essence and Accident in Software Engineering' (1987). In this paper, he highlights software engineering's inability to be problem-free by stating that the task of software development will always have four essential problems: complexity, conformity, changeability and

invisibility. Brooks' also goes on to reiterate that despite any advances in technology, there will never be a single 'silver bullet' that handles all four problems at once. When combined together, these four essential issues can severely limit the simplicity, consistency and dexterity of the software development process.

## 2 DISCUSSIONS

### 2.1 Complexity

The issue of complexity is defined by how scalable a given project is; modern software projects are as complex as they have ever been and will only continue to become more complex as new strategies, techniques and findings are uncovered. Software entities are also "more complex in construction" regarding their size than "almost any other human construct" (Brooks, 1987). Any additions to these

already large projects will undoubtedly cause issues regarding the “level of communication, understanding, oversight and control” (Harun, 2020) that developers have over the given system. This can make a project a whole lot more difficult and complicated to deal with. This can also cause a skewed, non-linear increase in complexity that is unaccounted or prepared for by a development team, as “most of the complexity in software development is accidental” (Seemann, 2019).

TOGAF helps to remedy issues regarding complexity and project scalability by having a “clear set of standardized rules” (Brown, 2019) and procedures for using the framework. This decreases the complexity of the project, as there is already a strict set of guidelines for the developers to follow and assess their progress with. A potential downfall of not using TOGAF would be having to “interpret, analyse, and make use” (Brown, 2019) of different elements in a business while using different strategies; this would potentially cause issues regarding confusion and time management, as all members of the team would not be on the same page if they were using different strategies to solve the same problem.

The different tasks performed by enterprise architects and software developers can also be extremely complex as well, especially if different members of the team are solving

different problems. TOGAF handles these difficulties by aiding in project organization, helping developers and businesses “establish ROIs and clear connections” between departments in order to carry out a “well-established planning framework” (Brown, 2019). This also aids in task management, as these frameworks help to resist severe project delays and wasted resources, potentially saving the company millions.

Despite all the positives that TOGAF brings to the table, it also sadly brings along a string of ironic problems regarding complexity. TOGAF is relatively new to the software development scene, meaning that it also suffers from issues regarding complexity compared to previous development methods. TOGAF is somewhat complex itself because it requires the individual developer to be well-versed in and “understand enterprise architectures” (Alali, 2016); not all systems operate using enterprise frameworks development cycles. Unfortunate scenarios like this lead to TOGAF being viewed as a better solution of accidental problems rather than the essential problems that Brooks highlights, especially complexity.

## 2.2 Conformity and Changeability

A software entity will always be subject to pressures regarding conformity, as software is developed for one purpose and one purpose only, which is to be used. Conformity refers to the ability of a software system to adapt and conform to older systems and requirements despite being newer and more advanced. An article written by Malepe (2016) regarding conformity states that a software product has to be “interfaced with older systems built before” the current product. This can lead to consumer problems if the software does not run as expected on different platforms.

Changeability runs along the same lines as conformity with only a few minor differences; changeability refers to a software entity’s ability to change based on external factors such as new software requirements, as opposed to conforming to old ones. The current society is a “technology-based” (Susskind, 2017) one, and a software entity must be designed in order to adapt the ever-changing environment of that very society. The inability of a system to do so will result in the consumer recognizing its “incapability” (Malepe, 2016), and eventually looking for other solutions if said problems are not fixed within a reasonable time period.

TOGAF helps to ease the essential difficulties caused by conformity by, once again, being a standardized solution for software development. The fact that TOGAF has been around for 20+ years speaks favourably to its ability to conform and be integrated with older systems, as software development has

become not only a technological problem, but a business one too, encompassing ideas such as “business process management and data analytics” (Watts, 2018). By creating its own rules, as well, the TOGAF architecture has also negated the need for conforming to old systems, as developers and stakeholders have been and continue to amend their systems to fit the frameworks’ model. The TOGAF community also boasts a forum in which members “come together to exchange information and feedback” (TOGAF, 2006). This makes it much easier for the new members to look towards the future, and not where they came from regarding old practices.

Another benefit of using TOGAF as a development model is that it is “customizable to organizational needs”, meaning that there is no obligation to use a structure that “does not serve” (Watts, 2018) the specific business needs of the team. This is a bonus regarding the essential issue of changeability, as it means that the specific developer can change and use the bits and pieces of the to their liking, helping to deter complexity issues too as the software packages will be smaller. Project scalability is also a much smoother process, as these packages can be “successfully be rolled out to other teams without much tweaking” (Watts, 2018)

TOGAF does have its downfalls though regarding the issues of conformity and complexity. The fact that not all organizations use this type of development cycle means that there is still a large percentage of companies out there that would not need to use this tool. Companies that do use the framework successfully however, also admit that failings happen, as in their

own words: “TOGAF cannot be a cure all for enterprise issues” (Watts, 2018). This, unfortunately, directly supports Brook’s view that there will never be a silver bullet for software development, as funnily enough, it does not even solve enterprise issues. Conformity and complexity are issues that will always exist, and it is clear that TOGAF cannot be regarded as a single solution.

### 2.3 Invisibility

Arguably the most difficult essential problem, invisibility sums up the very nature of software engineering as a whole because unlike hardware, which is physical and visible, “software is invisible and unvisualizable” (Brooks, 1987). It is, unfortunately, not an entity that can be recreated using geometric models.

TOGAF helps to make software development that much more visualizable by helping organizations “implement software technology in a structured and organized way” (White, 2018). There is a certain level of clarity that this brings to the development process, especially if some of the processes that are being structured were previously not. This is also the only way to visualize these processes at the physical level, through structure. However, these improvements are unlikely to benefit the software development process as a whole, as “software doesn’t exist meaningfully in static form; it only exists meaningfully when it’s executed” (McConnell, 1999). Due to this, a conclusion can be made that TOGAF is not a silver bullet for software engineering’s invisibility issues, as software will always be somewhat invisible.

## 3 CONCLUSION

The Open Group Architecture Framework has played a large role in changing the landscape of the software development process. Like many other advancements in the technological field, TOGAF does help to decrease some of the accidental and essential issues. It unfortunately, does leave a lot to be desired. At its best, it is an innovative and modernized scalability tool; at its worst, a reminder of the issues that still plague the task of software development thirty-odd years after Brooks’ paper was published.

The concept of TOGAF does not satisfy Brooks’ requirement for a single silver bullet to eliminate all of the essential issues raised. Is TOGAF a silver bullet that can solve all of software engineering’s inherent problems? The answer to that question, sadly, is no. TOGAF on its own cannot be relied on to remedy issues regarding complexity, changeability, conformity and invisibility. Will there ever be a silver bullet for software development’s inherent issues? Only time will tell. There are, however, a multitude of ‘bronze bullets’ that will continue to decrease and eliminate accidental issues, while also slowly chipping at the essential ones in the future.

A silver bullet would have to be able to address all these issues; therefore, TOGAF is not a silver bullet.

## 4 FUTURE WORK

Future work regarding this topic could involve a discussion regarding other software development strategies, and their impact on the inherent issues highlighted by Brooks’ paper.

Future works regarding the study of enterprise frameworks could look into determining which of the currently available frameworks as a whole support the ‘No Silver Bullet’ more than the others, as TOGAF is not the only framework available.

## ACKNOWLEDGEMENTS

I would like to thank Dr. Simon Scola for his feedback and guidance throughout the duration of this coursework. It was not an easy experience starting so late into the year, but I was able to complete the assignment with his help.

## REFERENCES

- [1] Alali (2016) What are the advantages of TOGAF, as compared to other frameworks? - Quora, *Quora.com*, [online] Available at: <https://www.quora.com/What-are-the-advantages-of-TOGAF-as-compared-to-other-frameworks> (Accessed 1 January 2021).
- [2] Brooks (1987) No Silver Bullet Essence and Accidents of Software Engineering, *Computer*, 20(4), pp. 10–19.
- [3] Brown (2019) What is TOGAF? Everything you Need to Know | Good e-Learning, *Good e-Learning*, [online] Available at: <https://www.goodelearning.com/courses/enterprise-architecture/togaf-9-foundation/what-is-togaf> (Accessed 1 January 2021).
- [4] Harun (2020) No Silver Bullet—Essence and Accident in Software Engineering (Fred Brooks, 1986), *Medium*, [online] Available at: <https://medium.com/@markharun/no-silver-bullet-essence-and-accident-in-software-engineering-fred-brooks-1986-41172655e62c> (Accessed 1 January 2021).
- [5] Malepe, K. A. (n.d.) GRIN - Summary of "No Silver Bullet. Essence and Accident in Software Engineering" by Frederick Brooks, 1995, *Summary of "No Silver Bullet. Essence and Accident in Software ..."* [online] Available at: <https://www.grin.com/document/321505> (Accessed 1 January 2021).
- [6] McConnell, S. (1999) Software Engineering Principles, *IEEE Software*, 16(2), pp. 6–8.
- [7] Seemann (2019) Yes silver bullet, *Blog.ploeh.dk*, [online] Available at: <https://blog.ploeh.dk/2019/07/01/yes-silver-bullet/> (Accessed 1 January 2021).

<https://pubs.opengroup.org/architecture/togaf8-doc/arch/>  
(Accessed 1 January 2021).

- [8] Susskind, R. E. and Susskind, D. (2017) *The future of the professions: how technology will transform the work of human experts*, Oxford, United Kingdom, Oxford University Press.
- [9] TOGAF (2006) The Open Group Architecture Framework Version 8.1.1, *Pubs.opengroup.org*, [online] Available at:

## 4 Appendix

### 4.1 Approved Extenuating Circumstances Claim



noreply@greenwich.ac.uk

12/2/20

To: tk9894h@gre.ac.uk >

## Extenuating Circumstance Claim

Dear Trevor Kiggundu

RE: Extenuating Circumstances claim number [20200041879](#)

COMP1430 - COMP 1430 Coursework  
COMP1430 - COMP 1430 Logbook  
COMP1608 - COMP 1608 Case Study Ind Rep  
COMP1618 - COMP 1618 Coursework  
COMP1833 - COMP 1833 Coursework

I am writing to you regarding your application to extenuating circumstances submitted on 27 November 2020.

The Extenuation Panel have considered your claim and have decided that the claim has been given the following outcome(s):

Course Title	Assessment Title	ID	Decision	Outcome
Systems Design & Dev	COMP 1430 Coursework	A01	Accepted	Up to 10 working days
Systems Design & Dev	COMP 1430 Logbook	A99	Accepted	Up to 10 working days
Managing IT Security & Risk	COMP 1608 Case Study Ind Rep	A99	Accepted	Up to 10 working days
Software Tools and Techniques	COMP 1618 Coursework	A99	Accepted	Up to 10 working days
Software Quality Management	COMP 1833 Coursework	A99	Accepted	Up to 10 working days

Regards

Chair of the EC Panel

Faculty of Liberal Arts & Sciences