**During the first 3 weeks of the module, we contributed to group discussions about a case of failure of an Information System. I choose to look into an example where the Royal Bank of Scotland had a “meltdown” that had large ramifications for its users in the United Kingdom. Find the discussion of this topic bellow, please note I did not include references here to aid in readability.**

*Case description*

*In 2012 the Royal Bank of Scotland (RBS) has a system "meltdown" causing payments between accounts to be delayed for four days.*

*Reasons for failure*

*RBS did not disclose the cause, but it is speculated that a batch schedular that is used to process routine jobs failed due to an update error. After the 2008 financial crash, RBS sought to minimize costs and choose to outsource many IT roles. As such many of the engineers who were familiar with maintaining the systems were let go and engineers with less experience took over. A former employee said "Backing out of a failed update to CA-7 really ought to have been a trivial matter for experienced operations and systems programming staff, especially if they knew that an update had been made. That this was not the case tends to imply that the criticisms of the policy to "off-shore" also hold some water."*

*I think it is important to note how much valuable knowledge employees who have experience with software and systems they develop/maintain have that is hard to pass on. Good documentation can go a long way, but the scale of redundancys here would have made the handover difficult.*

*Impact of the failure*

*The outage meant that many transfers, including wages, tax credits and disability living allowance were delayed for nearly a week.*

*RBS was fined £56m pounds in 2014 by Britsh regulators and consumer confidence was knocked.*

**Alice responded:**

*Hi Adam!*

*Thank you for bringing this case. Here is my contribution:*

*Stephen Hester, CEO of the RBS Group, said that the problem was caused by a software upgrade.The Royal Bank of Scotland (RBS) has issued an apology to the*

*17.5 million customers that are thought to have been affected by the banking group's computer woes. (Daum, 2012)*

*The tech problems at the RBS banking group were caused by a failure in a piece of batch scheduling software. And at least some of the support staff for that software have been outsourced to India. As Worstall (2012) observed, a company needs to work out what its core business is. What a bank really is these days is a computer system. Thus, outsourcing of computing is the one thing that a bank really should not be doing.*

*Banks usually adopt new technologies early on and technological changes in recent years have occurred very fast. As the pace of technological change and the related launch of new technologies speed up, the ability of the state and businesses to keep pace is challenged. It would be presumptuous to claim to know the trajectory and best policy responses to what might comprise the most formidable challenges and significant opportunities of our times. (Körner, 2018)  "Client expectations will keep increasing," says Sergio P Ermotti, group CEO of UBS. "Younger generations will expect new products, better services and faster delivery—preferably at a lower cost." And this, he argues, is what the continuing digitalisation of bank services will achieve.( Ermotti, 2019)*

*Information System failures is a complex phenomenon that requires in-depth analysis.  According to a 2017 report from the Project Management Institute (PMI), 14% of IT projects fail. However, that number only represents the total failures. Of the projects that didn't fail outright, 31 % didn't meet their goals, 43 % exceeded their initial budgets, and 49 % were late. Even with appropriate measures to prevent failure of the information system project, they may still occur. When a system fails, it's necessary to understand the underlying reasons so we can be learning from it.*

**To which I responded:**

*Hi Alice,*

*Thank you for your comment, I was particularly interested in the insight into technology adoption in banking.*

*I decided to look into some of the technology selection used in banking and found Goldsmith-Sachs to be a very intriguing case. Goldsmith-Sachs have actually been using their own proprietary programming language called "Slang". Slang is a programming language similar to Python but suited specifically to the needs of its domain. Slang is a single-threaded language. I have heard from a colleague who worked on software for a submarine that that project had also used single-threaded programming languages. For the submarine project, obviously reliability and security were paramount and a single-threaded language was chosen to help avoid* *multithreading issues such as deadlocks. I wonder if single-threaded languages are common across critical applications such as this banking example.*

*There have been positives and negatives of Goldsmith-Sachs use of Slang. On the plus side, the language was designed to make common uses such as loading market data very simple, this has made it easier for less advanced users to get value from the language.*

*On the flip side, developers outside of GS are not familiar with Slang and the slang language can not use any of the libraries for other languages like python. The developers at GS have to spend a large amount of time developing tools and libraries that would "come for free" with other languages or technologies. GS have recently started using more Java in their codebase because this helped attract younger developers who were not interested in using Slang.*

**To which I responded:**

*Hi Adam,*

*I'm also interested in banking technology, especially Cognitive RPA. This case you posted is very interesting. I didn't know about this issue on Slang. Thank you very much for posting this.*

*In my previous post, I mentioned the outsourcing of the bank's IT system. I read the Forbes article you used and the author highlights that the outsourcing of computing is the one thing that a bank really should not be doing. He mentioned that part of the support staff for the software had been outsourced to India. As you said, developers outside of GS are not familiar with Slang, so looks like this fact contributed to the failure.*

*I don't know other banking experiences with outsourcing, but what Worstall said about modern banking in the Forbes article makes sense. What a bank really is these days is a computer system. In fact, banking is being redefined by AI. There is even a new name for it: cognitive baking.*

*By the way, since you mentioned you are interested in banking technologies, I'll post the link of an amazing document from the World Economic Forum I recently read. It's called "The New Physics of Financial Services - Understanding how artificial intelligence is transforming the financial ecosystem".*

Beran responded to the topic suggesting:

Hi Adam,

Good discussion and the point about the general documentation is very important I would strongly recommend you to research about standards for documentation in particular priority and risk management.

**To which I responded:**

Hi Beran,

Software Risk management requires the identification of technical and process risks and the development of a mitigation strategy to handle them. The identified risks must be monitored during the project and if any risks the mitigation must be implemented.

Risk mitigation requires action throughout the Systems Development Life Cycle.

During the planning and requirements phases, the risks/reward of requirements should be evaluated and alternatives can be considered. During the implementation phase, risk mitigations should be implemented and verified. Throughout the project, the project managers should monitor if any of the risks come to fruition and if so implement the mitigation. The National Institute of Standards and Technology (NIST) state that during the operation phase the "Risk management activities are performed for periodic system reauthorization (or reaccreditation) or whenever major changes are made to an IT system in its operational, production environment".

Crucially in the case of RBS, risk management activities should have been performed during the upgrade to the production environment (the batch processor). The risk of a failed update should have been captured and therefore a mitigation should have been detailed. If we assume that the risk was captured and mitigation documented, a possible cause for the failure is that the staff did not have adequate training to implement the mitigation. A Skills/Competence Matrix helps compare the available skills and competencies within a team to the skills required to successfully complete the job. In this case, a Skills/Competence Matrix could have shown that the necessary skills were not met for implementing this mitigation. With the change of team members after the outsourcing, refreshing the Skills/Competence Matrix would be of vital importance. Had the skill gap been identified the issue could have been stopped from occurring.

Interestingly the NIST also discusses the disposal phase of a systems life cycle and the risks involved. Risks such as residual data not being destroyed and the importance of archiving important information are raised and important to consider.

**As a summary I concluded:**

Information systems are large risky projects with an estimated 14% never reaching delivery (PMI, 2017). The story does not end with delivery though, information systems require maintenance and security patches at a minimum and feature updates are almost certain.

In the RBS case study, it has been postulated that the outsourcing of development/maintenance led to a colossal service outage (Worstall, 2012). During further reading, I have come to realize that the core problem here is not unique to this RBS case but could affect any organization. The core issue is that knowledge was lost inside of the organization that was necessary to perform routine operations (updates). For RBS the cause was a dramatic shift towards outsourcing as a means to cut costs but it could also occur simply from natural staff turnover. Further, the large scale departure of engineers from a project is very common with many contractors and temporary staff being let go after delivery. This is due to many temporary staff/contractors leaving once the system has been delivered and only a smaller "skeleton" maintenance crew remaining.

It speaks volumes that former employees at RBS could postulate the cause of the outage correctly, clearly, they wouldn't have made the same mistake.

During my reading, I found that "Skills/Competency" matrix could have been used to help prevent this outcome. Such a document should have been filled with the required skills generated from risk mitigation.

I have read further advice on handling software system handovers and have some more ideas that could have helped RBS and other organizations in such a situation.

Implementing a phased transition period for a system handover can ensure the new team have "back-up" when learning the ropes.

In the case of RBS, the outsourcing team could have begun with a limited period of overlap with the pre-existing team. Over time the original team could be dissolved when there is confidence in the new team's abilities (Etheredge, 2020).

Ensuring that testing/integration environments are set up before the handover is also of paramount importance so that the new team is ready to make modifications without accidentally breaking things.

There is a raft of documentation that becomes so much more important after a handover. Design, discovery and technical documentation should all be in the best shape for a handover. Its important for engineers on the team to not only document what has been done but why. Explaining the thought process behind decisions can help new developers make better-informed decisions.

Lastly, the question must be asked, should RBS have outsourced maintenance at all? As was stated by Forbes " What a bank really is these days is a computer system". In outsourcing maintenance, RBS is in fact giving away control of its key business.

If at a future date RBS realize this loss it will be much harder reclaim this skill with the outsourced team having little incentive to ensure another handover is done properly.

I'd like to say thanks to everyone for the enlightening discussions!