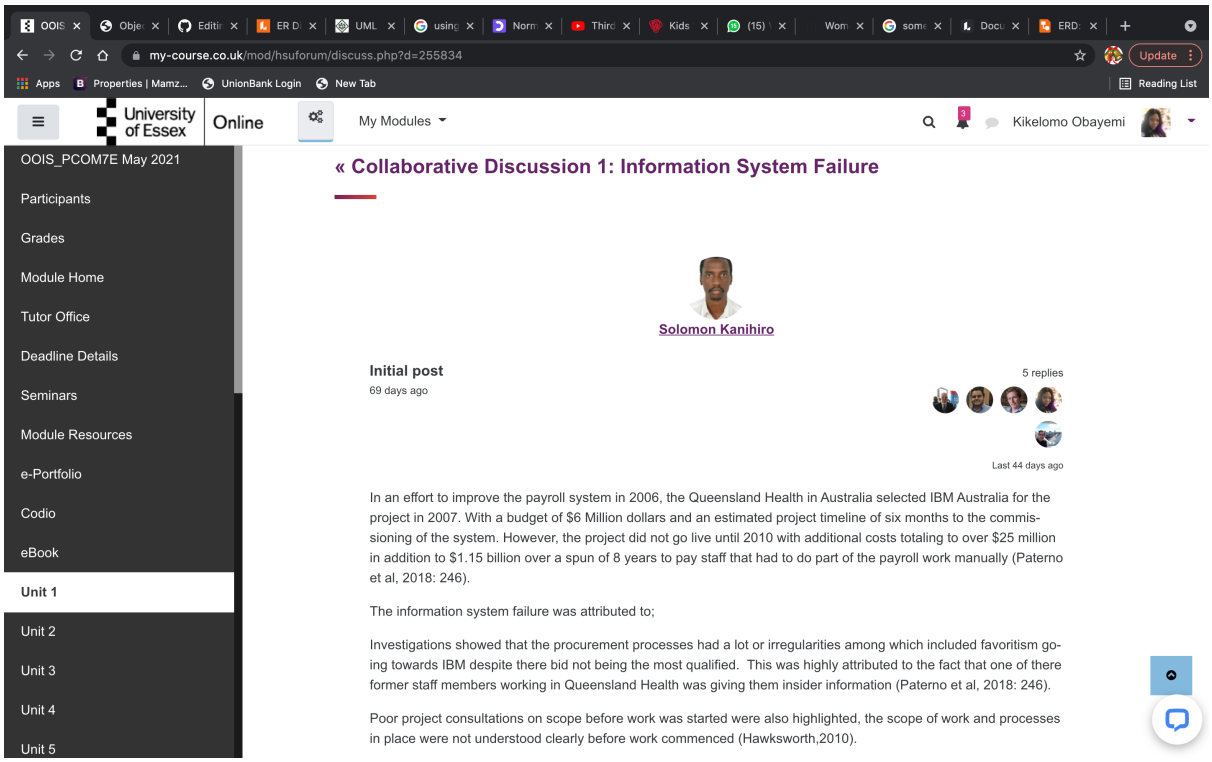
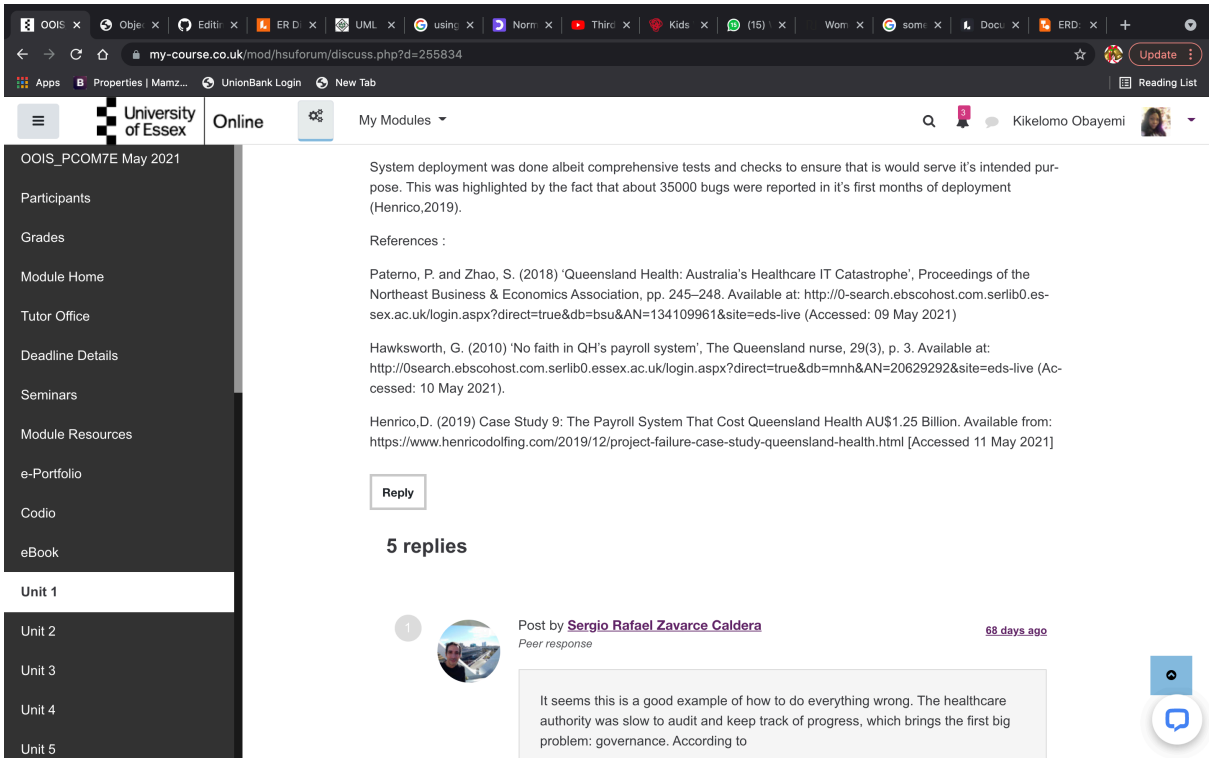
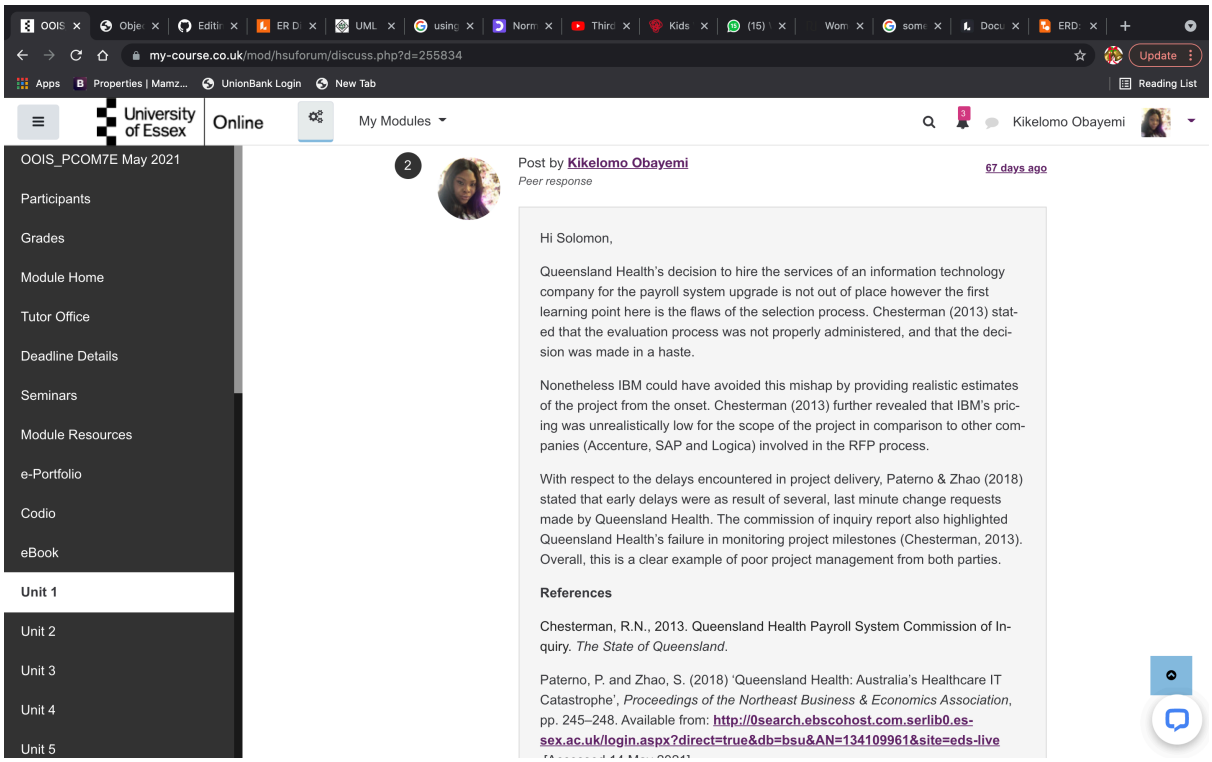
Collaborative Discussion: IS Failure

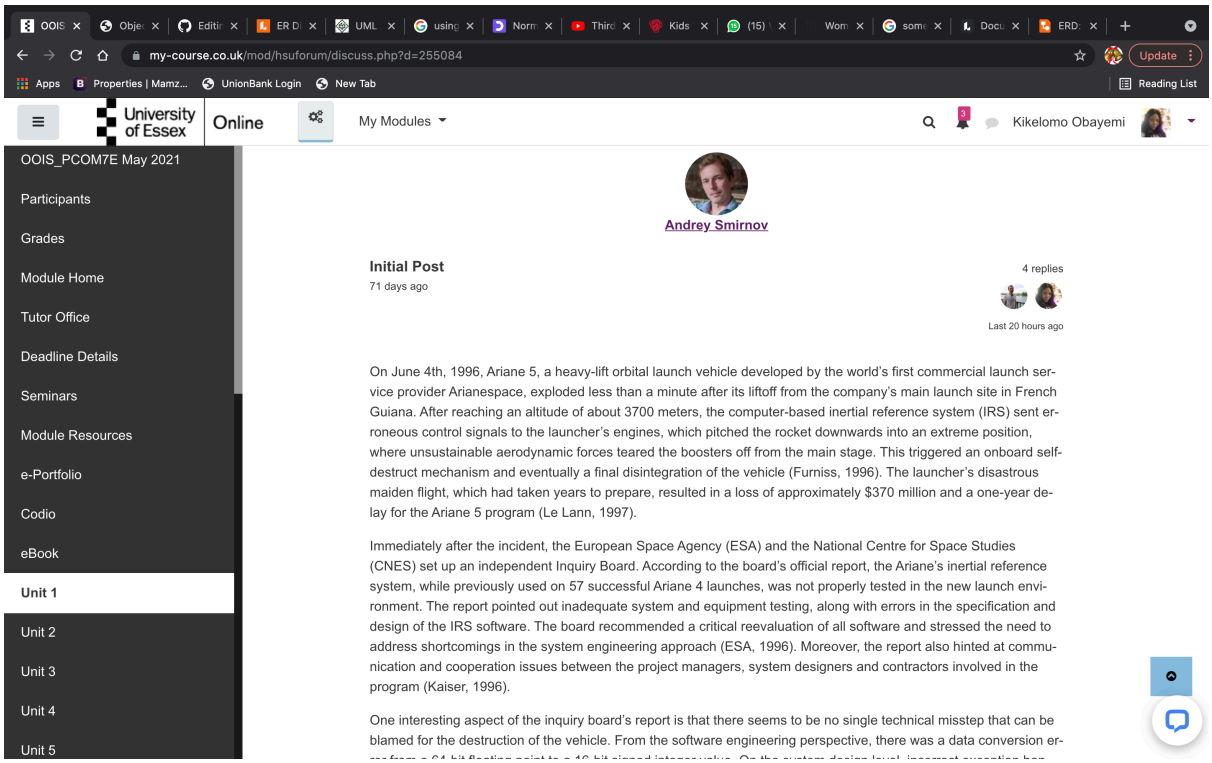
Feedback to Colleague Solomon Kanihiro's Post

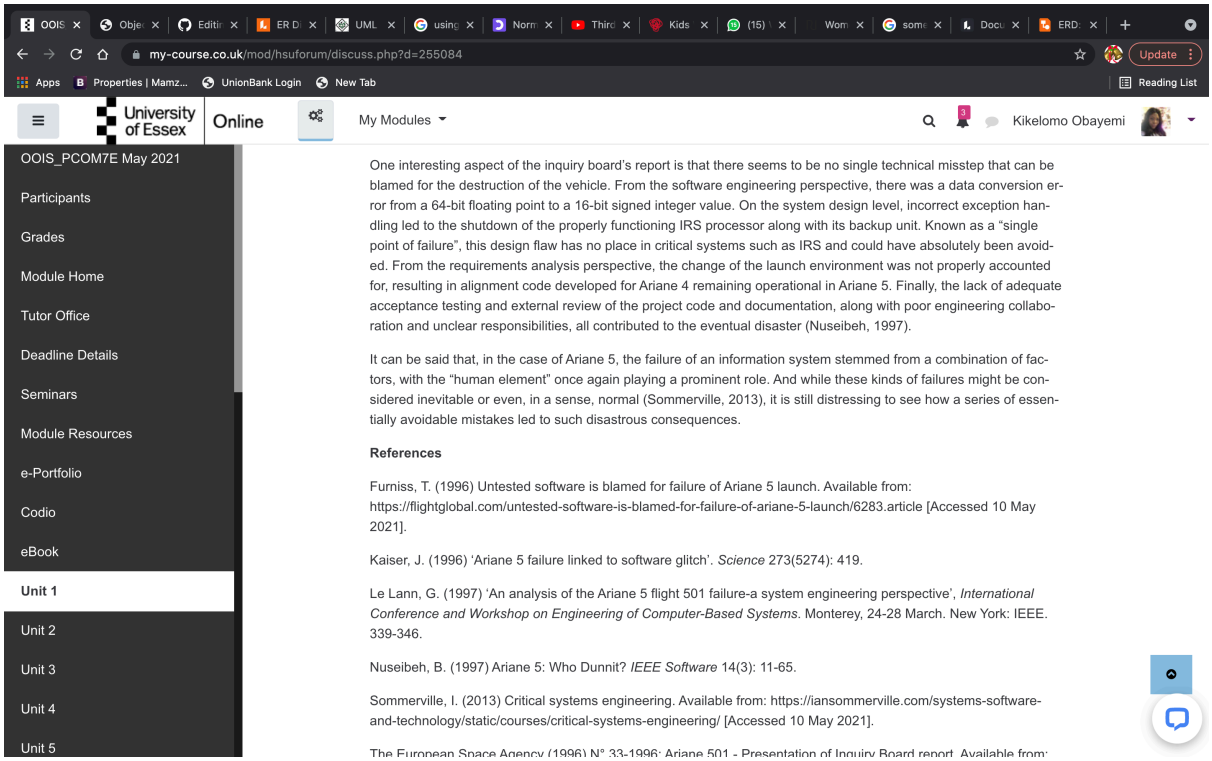






Feedback to Colleague Andrey smirnov's Post





One interesting aspect of the inquiry board's report is that there seems to be no single technical misstep that can be blamed for the destruction of the vehicle. From the software engineering perspective, there was a data conversion error from a 64-bit floating point to a 16-bit signed integer value. On the system design level, incorrect exception handling led to the shutdown of the properly functioning IRS processor along with its backup unit. Known as a "single point of failure", this design flaw has no place in critical systems such as IRS and could have absolutely been avoided. From the requirements analysis perspective, the change of the launch environment was not properly accounted for, resulting in alignment code developed for Ariane 4 remaining operational in Ariane 5. Finally, the lack of adequate acceptance testing and external review of the project code and documentation, along with poor engineering collaboration and unclear responsibilities, all contributed to the eventual disaster (Nuseibeh, 1997).

It can be said that, in the case of Ariane 5, the failure of an information system stemmed from a combination of factors, with the "human element" once again playing a prominent role. And while these kinds of failures might be considered inevitable or even, in a sense, normal (Sommerville, 2013), it is still distressing to see how a series of essentially avoidable mistakes led to such disastrous consequences.

References

Furniss, T. (1996) Untested software is blamed for failure of Ariane 5 launch. Available from: https://flightglobal.com/untested-software-is-blamed-for-failure-of-ariane-5-launch/6283.article [Accessed 10 May 2021].

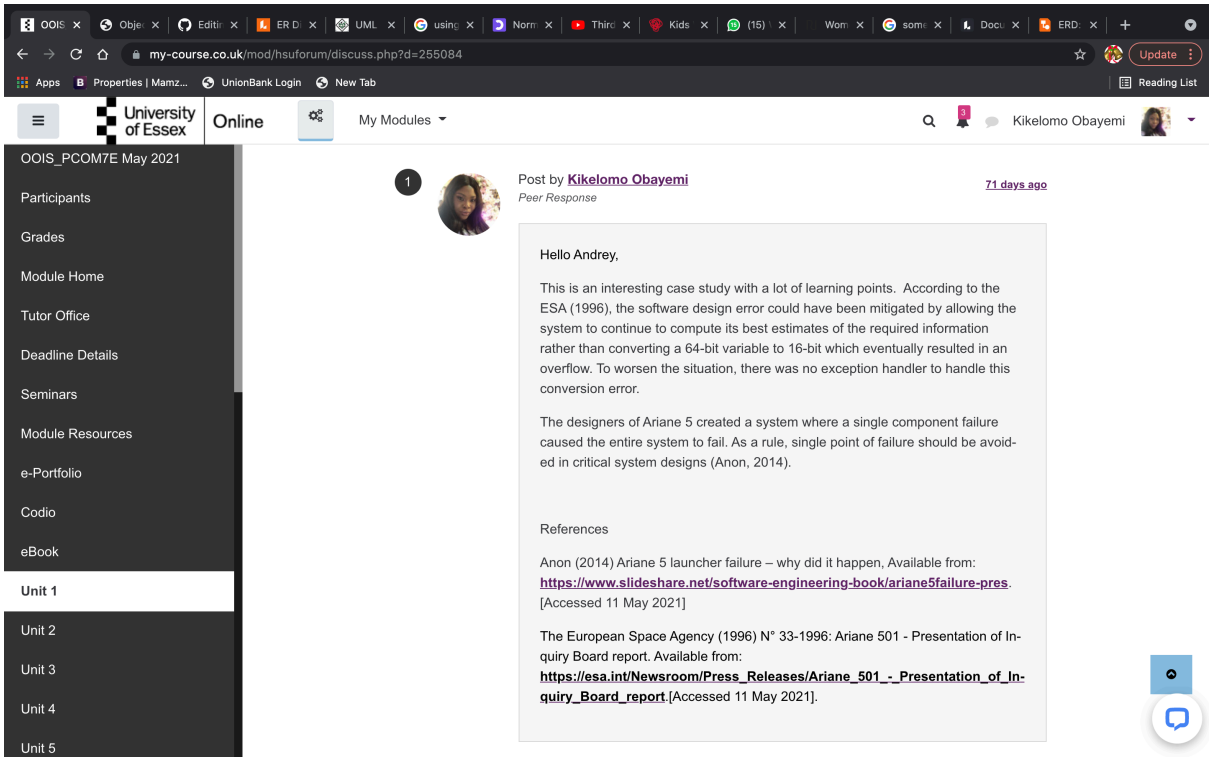
Kaiser, J. (1996) 'Ariane 5 failure linked to software glitch'. *Science* 273(5274): 419.

Le Lann, G. (1997) 'An analysis of the Ariane 5 flight 501 failure-a system engineering perspective', *International Conference and Workshop on Engineering of Computer-Based Systems*. Monterey, 24-28 March. New York: IEEE. 339-346.

Nuseibeh, B. (1997) Ariane 5: Who Dunnit? *IEEE Software* 14(3): 11-65.

Sommerville, I. (2013) Critical systems engineering. Available from: https://iansommerville.com/systems-software-and-technology/static/courses/critical-systems-engineering/ [Accessed 10 May 2021].

The European Space Agency (1996) N° 33-1996: Ariane 501 - Presentation of Inquiry Board report. Available from:



1

Post by **Kikelomo Obayemi**

Peer Response

71 days ago

Hello Andrey,

This is an interesting case study with a lot of learning points. According to the ESA (1996), the software design error could have been mitigated by allowing the system to continue to compute its best estimates of the required information rather than converting a 64-bit variable to 16-bit which eventually resulted in an overflow. To worsen the situation, there was no exception handler to handle this conversion error.

The designers of Ariane 5 created a system where a single component failure caused the entire system to fail. As a rule, single point of failure should be avoided in critical system designs (Anon, 2014).

References

Anon (2014) Ariane 5 launcher failure – why did it happen, Available from: **https://www.slideshare.net/software-engineering-book/ariane5failure-pres**. [Accessed 11 May 2021]

The European Space Agency (1996) N° 33-1996: Ariane 501 - Presentation of Inquiry Board report. Available from: **https://esa.int/Newsroom/Press_Releases/Ariane_501_-_Presentation_of_Inquiry_Board_report**.[Accessed 11 May 2021].