Collaborative Learning Discussion 2: The Effect of Risk on the SDLC

After reviewing Hijazi et al. (2014) and Roy et al. (2015) and considering the effect of risk on the SDLC, I would suggest five prevalent causes of risk are:

- Time
- Costs
- Funding
- Developers
- Communication

When considering time factors, we can also link this to costs. From Hijazi et al. (2014), inadequate project duration estimation has a knock-on effect. The drain of resources from poor planning can result in insufficient resources for developers and team members. Unrealised schedules and costs would profoundly affect the budget, and funding could be reconsidered, impairing the project's result or success.

Developers can be a potential risk due to numerous factors affecting the project and performance. Depending on the programming language being used, it could be found that developers may be inexperienced (Shahzad et al., 2010), resulting in over-complicated code with errors that would affect performance. Developers may also encounter challenges when implementing or utilising new technologies they are unaware of (Hijazi et al., 2014). This would link to time and cost as improvements would be needed meaning a reduction in productivity.

Communication is another cause identified. If there are breakdowns in information being relayed between end-users, customers, managerial staff or developers, the use or needs of the product may not be adhered to (Huang & Han, 2008). The expectations between parties may be on different levels, leading to conflict, extended project times or project failure.

Potential mitigations for these causes could be:

- Infrastructure management process
- Mapping between design and coding
- Agreements on execution and expectations
- Transparent, open dialogue between all parties
- Training solutions

To mitigate, an effective infrastructure management process (Ross et al., 2016) could support the systems engineering team process. Therefore 'standards of excellence' would ensure optimal performance. Agreements on expectations and executions could ease the process flow, and all parties involved could adhere to which would also support communication. The security aspects of the process could be facilitated with the regular assessment that can be improved through mapping and testing (Rituparna et al., 2016). Mapping the design and coding process thoroughly would support an understanding of duration and costs in the planning implantation cycle. Communication could be further improved with open, transparent internal and external dialogue with third-party providers. This can be supported with data protection, privacy, stakeholder and personnel for all users. Furthermore, developer training solutions could also help technology use and minimise errors in code.

Huang, S.-J. & Han, W.-M. (2008). Exploring the relationship between software project duration and risk exposure: A cluster analysis. *Information & Management,* 45, (3): 175-182.

Rituparna, C., Agostino, C., Khalid, S. & Nabendu, C. (2016). *Advanced Computing and Systems for Security: Volume 2*. New Delhi, Springer.

Ross, R. S., Mcevilley, M. & Oren, J. C. Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems | NIST. 2016.

Shahzad, B., Almudimigh, A. & Ullah, Z. (2010). Risk Identification and Preemptive scheduling in Software Development Life Cycle. *Journal of Computer Science and Technology,* 10, 55-63.