Initial Post

Initial Post

by Ben Blakemore - Saturday, 8 November 2025, 1:19 PM

Question 1: What do you believe are the three most common reasons for project failure?
Agrawal et al. (2024) found through their primary studies that the majority of failures stems from both knowledge gaps and organisational influences, with Figure 4 in their paper attributing 31% of errors (used synonymously with failures) to each. Following this is 25% of errors being caused by cognitive load - errors due to mental fatigue or stress.

Drawing on personal experience, with 5 years experience as a software engineer, I lack the ability to look back on a lengthy career to relate these figures to, but this broadly matches what I have directly experienced and heard from those I've worked with. Projects I've experienced failure in have primarily been due to mismanagement and evolving milestones and deadlines that often don't have the budget to resolve them. This is supported in Table 7 where the effects of time pressure due to approaching deadlines and declining budgets leads to poor-quality software design resulting in faults.

I feel that the type of project also influences how these 3 factors identified by the authors affects success rates. Large scale projects with multi-disciplinary teams can be more prone to organisational influences, as inefficiencies in one domain can create bottlenecks and therefore have knock-on effects felt within the rest of the project, leading to failure. While smaller projects with a more narrowed scope are more susceptible to knowledge based failures, where a single individual leaving can result in important domain knowledge being lost. Table 7 again supports this with the authors detailing how miscommunication between business functions can result in failures in specifications or processes being properly communicated.

Question 2: Give two examples of failures that support your choices
The rollout of Healthcare.gov in the USA exemplifies how coordinating numerous external businesses and vendors can create conditions for project failure. Intended to reform the way that Americans accessed their healthcare, the result was a disastrously over-budget and non-working website. With 60 contracts being awarded to 33 different businesses, handled by the Centers for Medicare and Medicaid Services (CMS), inter-business management was going to play a key factor in running the project smoothly. However, this didn't happen. A critical breakdown occurred with CGI Federal, the largest contractor awarded over $90 million to develop the front-end marketplace. CMS perceived CGI to be the project's lead integrator but had no written documentation outlining this agreement, while CGI representatives stated they had no such understanding of their role. This fundamental miscommunication about project leadership meant that no single entity was coordinating the integration of the multiple systems, resulting in a fragmented development process where critical integration issues went unaddressed. CMS only raised their concerns about CGI's performance in late 2013, just months before launch, but failed to hold the contractor accountable or take corrective action. They also delayed governance reviews which would have earlier exposed some of the failings of the project, choosing instead to move forward with an on-schedule launch despite identifying 45 critical and 324 severe code defects across the system (Dolfing, 2023).

The Therac-25 case study is often cited due to its numerous shortcomings that lead to abject failure, one of which was the impact of a small development team. A computer-controlled radiation therapy machine, the Therac-25 delivered erroneously high doses to patients, leading to 3 deaths. The project was developed in assembly by a single programmer, who then left the company leaving very little documentation (Leveson and Turner, 1993). When bugs were then discovered, this made the process of finding and fixing them all the more difficult demonstrating the effect that knowledge based failures can have.

References:
Agrawal, T., Walia, G.S. and Anu, V.K. (2024) 'Development of a Software Design Error Taxonomy: A Systematic Literature Review', SN Computer Science, 5(7), p. 467. Springer. Available at: https://doi.org/10.1007/s42979-024-02797-2. [Accessed: 4 Novembe]

Dolfing, H. (2023) *Case Study 17: The Disastrous Launch of Healthcare.gov*. Available at: https://www.henricodolfing.com/2022/12/case-study-launch-failure-healthcare-gov.html (Accessed: 4 N

Leveson, N. G. and Turner, C. S. (1993) *An Investigation of the Therac-25 Accidents*. Available at: https://www.cs.columbia.edu/~junfeng/08fa-e6998/sched/readings/therac25.pdf (Accessed: 6 November 2025).

Permalink Reply

Re: Initial Post

by Lauren Pechey - Monday, 15 December 2025, 12:15 PM

Hi Ben,

Your post offers a strong and reflective analysis that effectively bridges empirical findings from Agrawal et al. (2024) with your professional experience as a software engineer. I particularly agree with your emphasis on organisational influences and knowledge gaps as dominant contributors to failure. This aligns closely with Nizam's (2022) process-based view of software project failure, which highlights how managerial decisions and structural weaknesses propagate downstream into technical defects.

Your discussion of time pressure and declining budgets is especially convincing, as it reflects how organisational constraints amplify cognitive load and degrade design quality. This is consistent with Verner and Sarwar's (2021) findings in NHS IT projects, where unrealistic deadlines and insufficient resourcing directly contributed to poor system outcomes. Your distinction between large-scale, multi-disciplinary projects and smaller, knowledge-concentrated teams is also insightful, as it demonstrates how failure mechanisms vary by project context rather than being uniform.

The Healthcare.gov and Therac-25 examples strongly support your argument by illustrating organisational misalignment and knowledge-based failures respectively. Together, these cases reinforce how socio-technical factors interact to drive software project failure (Agrawal et al., 2024; Nizam, 2022).

References:

Agrawal, T., Walia, G.S. and Anu, V.K. (2024) Development of a software design error taxonomy: A systematic literature review. SN Computer Science, 5: 467. DOI: https://doi.org/10.1007/s42979-024-02797-2

Nizam, A. (2022) Software project failure process definition. IEEE Access, 10: 34428–34441. DOI: https://doi.org/10.1109/ACCESS.2022.3161036

Verner, C.M. and Sarwar, D. (2021) Avoiding project failure and achieving project success in NHS IT system projects in the United Kingdom. International Journal of Strategic Engineering (IJoSE), 4(1): 1–22. DOI: https://doi.org/10.4018/IJoSE.2021010103

Maximum rating: -

Permalink Show parent Reply

◀ Initial Post Summary Post ▶

Chat to us!