Software Engineering for Computational Science

ABSTRACT

- Despite the increasing importance of in silico experiments to the scientific discovery process, state-of-the-art software engineering practices are rarely adopted in computational science.
- To understand the underlying causes for this situation and to identify ways to improve it, we conducted a literature survey on software engineering practices in computational science.
- .We identified 13 recurring key characteristics of scientific software development that are the result of the nature of scientific challenges, the limitations of computers, and the cultural environment of scientific software development. Our findings allow us to point out shortcomings of existing approaches for bridging the gap between software engineering and computational science and to provide an outlook on promising research directions that could contribute to improving the current situation.

EXISTING SYSTEM

- increases the importance of employing sound software engineering practices in the development of scientific software to guarantee reliable and accurate scientific results.
- However, surveys show that state-of-the-art software engineering methods are rarely adopted in computational science.2,3 To understand the underlying causes for this and to identify ways to improve the current situation, in this article, we survey literature on software engineering in computational science and identify key characteristics that are unique to scientific software development

DISADVANTAGES

- increasing capabilities of modern computers, in silicon experiments are becoming more complex and playing a more important role in the scientific discovery process.1 As a consequence, the complexity and lifespan of scientific software are growing, as well as the necessity for its .
- This increases the importance of employing sound software engineering practices in the development of scientific software to guarantee reliable and accurate scientific results output to be reproducible and verifiable.

PROPOSED SYSTEM

- To understand the underlying causes for this situation and to identify ways to improve it, we conducted a literature survey on software engineering practices in computational science.
- We identified 13 recurring key characteristics of scientific software development that are the result of the nature of scientific challenges, the limitations of computers, and the cultural environment of scientific software development.

ADVANTAGES

- basis of this viewpoint, we review publications on case studies and surveys conducted among computational scientists to identify 13 key characteristics of scientific software development that explain why state-of-theart software engineering techniques are poorly adopted in computational science.
- The findings of our literature survey allow us to identify shortcomings of existing approaches for bridging the gap between software engineering and computational science and to provide an outlook on promising research directions that could contribute to improving the current situation.

HARDWARE REQUIREMENTS

- Processor :Intel Pentium IV 1GHz
- RAM :256MB (Min)
- Hard Drive :5GB free space
- Monitor :1024 * 768, High Color inch
- Mouse :Scroll Mouse(Logitech)
- Keyboard :104 keys

SOFTWARE REQUIREMENTS

- OS : Windows XP/7/8
- Front End : Visual Studio 2010/ netbeans 7.1
- Back End : SQL Server 2005/ heidisql 3.2
- Browser : Any Web Browser

CONCLUSION

- On the basis of an examination of the historical development of the relationship between software engineering and computational science (the past), we identified 13 key characteristics of scientific software development by reviewing published literature (the present).
- We found that scientific software development's unique characteristics prevent scientists from using state-of -the- art software engineering tools and methods. This situation created a chasm between software engineering and computational science, which resulted in productivity and credibility crises of the latter discipline.
- We examined attempts to bridge the gap in order to reveal the shortcomings of existing solutions and to point out further research directions, such as the use of DLSs and testing techniques without predefined oracles (the possible future).

REFERENCES

1. D.E. Post, “The Changing Face of Scientific and Engineering Computing,” Computing n Science & Eng., vol. 15, no. 6, 2013, pp. 4–6.
2. D. Heaton and J.C. Carver, “Claims about the Use of Software Engineering Practices in Science: A Systematic Literature Review,” Information and Software Technology, vol. 67, 2015, pp. 207–219.
3. P. Prabhu et al., “A Survey of the Practice of Computational Science,” Proc. Int'l Conf. High Performance Computing, Networking, Storage, and Analysis (SC 11), 2011, p. 19:1.
4. J.E. Hannay et al., “How Do Scientists Develop and Use Scientific Software?,” Proc. Workshop Software Eng. for Computational Science and Eng. (SECSE 09), 2009, pp.1–8.
5. S. Faulk et al., “Scientific Computing’s Productivity Gridlock: How SoftwareEngineering Can Help,” Computing in Science & Eng., vol. 11, 2009, pp. 30–39.