Lightning Talk: HydroShare – A Case Study in Software Engineering Best Practices and Culture Change for Developing Sustainable Community Software

Ray Idaszak[1], David G. Tarboton (PI)[2], Hong Yi[1], Michael Stealey[1], Pabitra Dash[2], Alva Couch[3], Daniel P. Ames[4], Jeffery S. Horsburgh[2], Tony Castronova[2], Jon Goodall[5], Mohamed Morsy[5], Venkatesh Merwade[6], Mauriel Ramirez[2], Tian Gan[2], Drew (Zhiyu) Li[4], Jeff Sadler[4], Shawn Crawley[4], Zhaokun Xue[3], Lan Zhao[6], Carol Song[6], Christina Bandaragoda[7]

[1]RENCI, University of North Carolina at Chapel Hill, Chapel Hill, North Carolina, USA, Email: {rayi, hongyi, stealey}@renci.org
[2]Utah State University, Logan, Utah, USA, Email: {dtarb, pabitra.dash, jeff.horsburgh, tony.castronova, mauriel.ramirez, tian.gan}@usu.edu
[3]Tufts University, Medford, Massachusetts, USA, Email: {alva.couch, zhaokun.xue)@tufts.edu
[4]Brigham Young University, Provo, Utah, USA, Email: {dan.ames, zhiyu.li, jeffrey.sadler, shawn.crawley)@byu.edu
[5]University of Virginia, Charlottesville, Virginia, USA, Email: {goodall, mmm4dh}@virginia.edu
[6]Purdue University, West Lafayette, Indiana, USA, Email: {venvmerwade, lanzhao, cxsong}@purdue.edu
[7]University of Washington, Seattle, Washington, USA, Email: cband@uw.edu

***Abstract*—Applying modern software engineering to scientific software development has many challenges. These include lack of time or incentives to learn software engineering best practices, a lack of understanding or appreciation of the value of modern software engineering, and a shortage of mechanisms to more broadly change the software engineering culture of a community of researchers in a singular concerted effort. The HydroShare project is a large distributed research software development project that has made significant inroads on these challenges. As the HydroShare project enters its fifth year of NSF funding, we discuss how research scientists and research software engineers from the ten collaborating institutions consistently produce high-quality HydroShare code releases every 2-3 weeks that are formally reviewed and tested.**

I. Introduction

Applying modern software engineering to scientific software development has many challenges. These include lack of time or incentives to learn software engineering best practices, a lack of understanding or appreciation of the value of modern software engineering, and a shortage of mechanisms to more broadly change the software engineering culture of a community of researchers in a singular concerted effort. The HydroShare project is a large distributed research software development project that has made significant inroads on these challenges [1]. HydroShare is a hydrology community open-source cyberinfrastructure project supported by the National Science Foundation (NSF) through its Software Infrastructure for Sustained Innovation program (SI2) [2, 3]. Domain scientists, professional software engineers, and academic software developers from ten academic, research, and development organizations located across the United States collaborate to develop HydroShare - an online, collaborative system supporting the open sharing of hydrologic data, analytical tools, and computer models.

At the onset of the HydroShare project, most of the research scientist collaborators on the project were not clear on the difference between software development and software engineering, and they were not familiar with concepts such as iterative software development, test-driven development, code reviews, and continuous integration. Now, as the HydroShare project enters its fifth year of NSF funding, research scientists and research software engineers from the ten collaborating institutions consistently produce high-quality HydroShare code releases every 2-3 weeks that are formally reviewed and tested. The HydroShare team now understands and embraces the value of modern software engineering; indeed, they understand the time savings of producing high-quality sustainable code at the onset as enabling more time spent on their research and not on the time-consuming alternative of managing potentially poor quality code had they not embraced modern software engineering.

II. Software Engineering Best Practices and Culture Change for Developing Sustainable Community Software

The HydroShare team has achieved a community rhythm in the continual deployment of high-quality community code releases of HydroShare. The community visibility of this rhythm has served as an incentive for community culture change in that team members are pleased with the resulting code, its evolving significant capabilities, and the efficiency by which new features are tested and integrated as contributed by the collaborating team members. This culture change has naturally incented collaborating researchers to take the time to teach modern software engineering best practices to their graduate and postdoctoral students that time has shown have now also embraced these practices. While host universities have provided the requisite training in computer programming to HydroShare researchers, they have not provided the accompanying instruction on software engineering best practices. However, the HydroShare project has

provided numerous faculty, graduate students, postdoctoral students, and even undergraduate students the opportunity to learn modern software engineering best practices first-hand in the absence formal classroom instruction of same.

What is important to convey is this propagation and adoption of software engineering best practices across the HydroShare team is happening organically without the need to force it. It is a success story in that what is referred to as a "community rhythm" herein is in effect like an engine that, once started, sustains itself in part by visibly promoting its own success and efficiencies such that there is no questioning of its uptake by new team members. In other words, adoption of these software engineering best practices becomes the new norm – there is no alternative that new HydroShare team members are ever exposed to. This is especially important to those faculty and students who are early in their careers as it has proven a viable mechanism of getting these individuals on a sustainable software path early on.

As it is beyond the scope of a lightning talk summary paper to fully describe the mechanism of how the success of HydroShare's software engineering is achieved, we refer the reader to a book chapter titled "HydroShare – A case study of the application of modern software engineering to a large distributed federally-funded scientific software development project" that offers a comprehensive discussion of this work [4]. The book chapter discusses the HydroShare team's use of iterative software development, continuous integration, and DevOps. HydroShare features are positioned as GitHub branches and worked on by subsets of the active HydroShare development community distributed across the ten collaborating organizations. Designs of proposed new features are discussed extensively initially during one of the weekly HydroShare team calls involving hydrology domain researchers, developers, and software engineers from the HydroShare collaborating institutions as well as community stakeholders. Designs are revisited as required so as to adapt to new technologies and/or address changing requirements. Once a design is accepted, unit tests are written and integrated with Jenkins which is an open source continuous integration tool. GitHub commits are made daily, and functional progress with running code demonstrations are reviewed weekly for functionality and usability during the HydroShare weekly team calls on one of the HydroShare pre-release virtual machines. HydroShare GitHub feature branches are rebased regularly with the HydroShare main branch to keep the code from getting out of sync with the HydroShare production release. As with all HydroShare code, code reviews are performed by someone other than the author of the code, and only until a "+1" is given by the code reviewer (via GitHub issue tracking) and all unit tests pass will code be committed incrementally into the HydroShare main branch. The HydroShare main site offers mechanisms for the community at large to comment on issues (including submission of bugs) and contribute suggestions to HydroShare. The HydroShare GitHub site maintains active statistics demonstrating the vibrant, open, and diverse HydroShare research software development activity [5].

A concluding note on the software sustainability of Hydroshare: since 2002, CUAHSI [6] – the primary U.S. hydrology consortium with 130 member universities and international organizations – collaborated in the predecessor to HydroShare called Hydrologic Information System, or HIS [7]. HIS is now maintained by the CUAHSI Water Data Center [8] as its community sustainability model. HydroShare is positioned as the successor to HIS, complementing but not replacing it. When the NSF-funded HydroShare award concludes, it will also be hosted by the CUAHSI Water Data Center – its long-term sustainability ensured by modern software engineering that will readily enable the broader community to continually make novel and useful contributions.

ACKNOWLEDGMENT

This material is based upon work supported by the USA National Science Foundation (NSF) under awards 1148453 and 1148090; any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

REFERENCES

[1] Tarboton, D. G., Idaszak, R., Horsburgh, J. S., J. Heard, Ames, D. P., Goodall, J. L., Band, L., Merwade, V., Couch, A., Arrigo, J., Hooper, R., Valentine D., and Maidment, D. (2014), "HydroShare: Advancing Collaboration through Hydrologic Data and Model Sharing." 7th International Conference on Environmental Modelling and Software. Ed. D. Ames and N. Quinn. San Diego, 2014.

[2] Implementation of NSF CIF21 Software Vision (SW-Vision), http://www.nsf.gov/si2/.

[3] NSF collaborative HydroShare award numbers 1148453 and 1148090, http://www.nsf.gov/awardsearch/showAward?AWD_ID=1148453/ and http://www.nsf.gov/awardsearch/showAward?AWD_ID=1148090/.

[4] Idaszak, R., Tarboton, D.G., Yi, H., Christopherson, L., Stealey, M.J., Miles, B., Dash, P., Couch, A., Spealman, C., Ames, D.P., Horsburgh, J.S. HydroShare – A case study of the application of modern software engineering to a large distributed federally-funded scientific software development project. Accepted for inclusion in: J. Carver, N.P.C. Hong, and G.K. Thiruvathukal (eds.) Software Engineering for Science, ISBN 9781498743853. Taylor&Francis CRC Press; November 2016.

[5] HydroShare GitHub site, https://github.com/hydroshare/hydroshare/graphs/commit-activity.

[6] A Vision for Hydrologic Science Research, Consortium of Universities for the Advancement of Hydrologic Sciences Inc., Technical Report Number 1, http://www.cuahsi.org/publications/cuahsi_tech_rpt_1.pdf.

[7] CUAHSI Hydrologic Information Systems, Consortium of Universities for the Advancement of Hydrologic Science, Inc. Technical Report Number 2 - Hydrologic Information Systems Committee, http://www.cuahsi.org/docs/dois/CUAHSI-TR2.pdf.

[8] https://www.cuahsi.org/wdc.