Review of The Process and Value of Reprogramming a Legacy Global Hydrological Model

Article by Nyenah et. al. Review by Rolf Hut

The authors present their work on reprogramming the WaterGAP model from its legacy code in C/C++ into python. The main purpose of this activity is to enhance the reproducibility of science done with WaterGAP, to make the science more transparent (FAIR) and overall reduce the effort required to maintain such a large code-base.

This is a worthwhile effort that will greatly help the (hydrological) scientific community in general and the WaterGAP user base in particular. The python version of WaterGAP has potential for bigger uptake, easier collaboration and is generally a better piece of research software than the legacy C/C++ version.

I do, however, struggle with this publication and its place in the academic literature. I find it hard both to judge what the intention of the authors is with the publication and if they success in that.

Sidenote: reporting on the progress of software projects within academia is a struggle. The classic "report on results of work done so others can build on it"-structure of academic articles doesn't fit on reporting on developing new software, because the software in and of itself is not a scientific result. I would argue, however, that in the current age where academic credit is almost solely awarded based on "publications" a form of reporting on important software projects is needed. Both for informing the academic community on the new software (availability) and for rewarding / acknowledging those working on building that software. We had exactly the same problem when writing our eWaterCycle papers, where the first eWaterCycle paper never made it past peer review because it lacked "scientific results". In the second paper we focused on providing use cases to illustrate the platform and did manage to publish the work. I think this illustrates that GMD as a journal is accepting more and more software-like contributions.

Below I will list the different purposes I identified in the manuscript and provide feedback and tips for each different purpose to optimize the paper towards that purpose. I leave it to the combined team of authors and the editors of GMD to decide on which purpose they want to prioritize in the manuscript (or maybe even split in different manuscripts, for different audiences, using different platforms?)

Announcing reprogrammed version of WaterGAP for potential users

The new WaterGAP seems to me like an amazing tool for hydrologists to work with. Sections 2.1 (Model description), 5 (architecture), 6 (eval against sustainability criteria) , 7 (Difference between output of C/C++ and Python versions) are essential for communicating this. For this focus I strongly suggest to add a few case studies that demonstrate the capabilities and user friendly-ness of the new WaterGAP.

Reporting on the process of reprogramming a legacy model

For those that contemplate reprogramming a legacy model, lessons learned from the transition from C/C++ version of WaterGAP to Python are very valuable. Section 3.1 on software evaluation against (FAIR) criteria and section 4 on the reprogramming process are very valuable here. I would add a paragraph on "lessons learned" that give pointers for others that set out to undertake a similar effort.

Reporting on user experience with the new WaterGAP codebase

The overview gathered from the survey conducted at EGU (section 3.3 and 8) gives some preliminary info on the perception of (potential) users of the new WaterGAP code towards its quality. This is in principle a valuable addition to the literature, but the width and execution of the survey is slim for a stand-alone publication. The selective response and low number of respondents make generalizing claims from this survey hard. I would strongly advice to present the results in terms of absolute numbers instead of percentages, so "10 people thought it was easy" versus "15% of people thought it was easy". For a full report on how outside users experience the new WaterGAP I suggest additional work, including for example a focus group session where users working with the new WaterGAP are observed.

Minor remarks

Independent of the direction chosen I have a few smaller remarks on the current version of the text:

- Line 61, add a citation to Wilkinson 2016 for FAIR
- Line 65: the unsuspecting reader might conclude that "to improve … … were reprogrammed" was done as part of the aforementioned eWatercycle project
- Line 181 and continuing: I would avoid as much as possible using URLs as citations and use bibtex citations to websites (preferably with a "last accessed" field)
- Section 4.1.2. I always learned that agile is a project management tool for when you have a fixed amount of hours, or people, to work on something, but the end goal is not fixed, just "create as much value as quickly as possible" (ideal for start-up culture). This is ideal when project goals are allowed to fluctuate during the project. In most scientific large projects the end goal is quite fixed (reprogram this model into python). Therefore, I would invite the authors to highlight why they settled on Agile as a project management practice for this project (there might well be very good reasons!)
- Line 288: "senior developers" are introduced as a role, but not previously explained.
- Line 291: same for "software development advisor"

Concluding

I really like the new python version of WaterGAP. (I like it so much that I would invite the authors to work together to add support for WaterGAP in eWaterCycle to make it even more accessible to hydrologists!). I hope the above suggestions will help in choosing a focus direction for the manuscript and emphasizing those parts throughout. I am happy to review an updated version of this manuscript.