The OpenFlexure Project. The technical challenges of Co-Developing a microscope in the UK and Tanzania

Julian Stirling
Department of Physics
University of Bath
Bath, UK
j.stirling@bath.ac.uk
ORCID: 0000-0002-8270-9237

Valerian L. Sanga
STICLab
Dar es Salaam, Tanzania
ORCID: 0000-0001-8421-4134

Paul T. Nyakyi
STICLab
Dar es Salaam, Tanzania

Grace A. Mwakajinga
STICLab
Dar es Salaam, Tanzania

Joel T. Collins
Department of Physics
University of Bath
Bath, UK
ORCID: 0000-0002-9382-7511

Kaspar Bumke
Department of Physics
University of Bath
Bath, UK

Joe Knapper
Department of Physics
University of Bath
Bath, UK
ORCID: 0000-0002-5519-1700

Qingxin Meng
Department of Physics
University of Bath
Bath, UK

Samuel McDermott
Cavendish Laboratory
University of Cambridge
Cambridge, UK
ORCID: 0000-0003-2736-5467

Richard Bowman
Department of Physics
University of Bath
Bath, UK
r.w.bowman@bath.ac.uk
ORCID: 0000-0002-0167-4969

***Abstract*—The OpenFlexure Microscope is a 3D-printed laboratory-grade motorised microscope. Over the past 3 years, the microscope has primarily been co-developed between the University of Bath and the Tanzanian engineering company STICLab. We are beginning the process of preparing the microscope for medical certification, as an *in-vitro* diagnostic device. In this paper we detail the technical challenges of remote design of a complex scientific instrument. We believe that identifying and solving these issues is essential if we are to encourage research organisations in the Global North to design instrumentation with Africans, rather than "for Africa".**

***Index Terms*—Microscopy, Co-Design, Distributed manufacturing, Medical Devices**

I. Introduction

The OpenFlexure Microscope (OFM) [1], [2] is a motorised laboratory-grade 3D-printed microscope. The microscope has been engineered for precise motion control and configurable high-performance imaging. As an open source hardware project, the microscope has been used on every continent in environments from rainforest to the Antarctic ice. It has also been modified by other research laboratories for advanced scientific applications such as super-resolution imaging [3]. The OFM forms part of the larger OpenFlexure project which focuses on using low-cost 3D printers for highly-precise positioning mechanisms by designing novel monolithic flexure stages [4].

The World Health Organisation describes manual optical microscopy as "the gold standard" of malaria diagnosis [5]. A trained technician manually checks a stained blood smear under a microscope and counts any malaria-causing *Plasmodium* parasites. In regions with restricted access to conventional microscopes, the OFM offers an accessible, partially automated alternative [1] with sufficient optical performance to identify and quantify these protozoa. Malaria incidence has decreased markedly over recent decades [6], but this makes access to high quality, quantitative diagnosis even more important. "Presumptive treatment" (where any patient with a fever is given antimalarial medicine) is effective when malaria incidence is extremely high, but at lower incidence is likely to result in patients with potentially life-threatening conditions being prescribed incorrect medication. This risks the patient's health and wastes

Funded by Royal Commission for the Exhibition of 1851; Engineering and Physical Sciences Research Council (EP/P029426/1, EP/R011443/1, EP/R013969/1); Royal Society (RGF\EA\181034, URF\R1\180153).

valuable antimalarials, as well as aiding the parasite's development of drug resistance [7]. Well-intentioned donations of scientific and medical equipment often go unused in sub-Saharan Africa, due to a lack of consumables, spare parts, maintenance infrastructure, and training. Local production of open-source hardware goes a long way to addressing these key barriers to the uptake of modern diagnostic technology, by ensuring the availability of both spare parts and competent engineers, as well as enabling products to be tailored to the needs of local situations.

By co-developing the microscope in the UK and Tanzania, we have been able to ensure that the microscope is not only appropriate for use in Tanzania, but also that the components can be reliably sourced in Tanzania and that the microscope can be produced locally. While we have found that co-development is essential to producing a device that is both appropriate for and manufacturable in Tanzania, co-development has not been without its drawbacks in design efficiency. We do not believe that co-design is intrinsically less efficient, but that the tools we would need to do it effectively are not available. Throughout the project, we have utilised software development platforms for our hardware development. In this work we will discuss the workflows we have developed that have enhanced our co-development, issues we have found that have impacted development, and the tools we would need to address these issues.

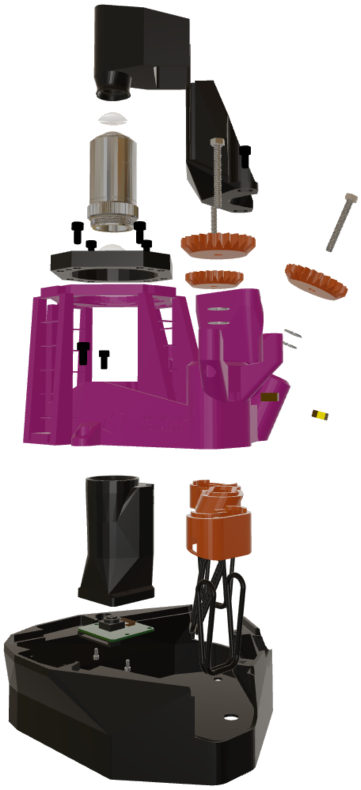

Fig. 1. Exploded view of the OpenFlexure Microscope.

II. Capturing an evolving design

For any complex engineering task, knowledge of the evolution of the design and the reasoning behind design decisions is essential to understanding the project. What is common sense or best practice for one designer may appear inefficient or damaging to another, particularly if there is no record of why each decision was made. This is particularly true for medical devices where design decisions must be recorded for the certification process. To capture the history of the design the OpenFlexure project has used Git, a distributed version control system primarily designed for software engineering. Git allows design snapshots (commits) to be saved along with messages that describe the changes that have been made. It also allows multiple changes to be made by different teams in parallel, which can then be merged together.

The hardware repository for the OFM has over 600 commits [8]. This excludes other repositories for the software [9], custom electronics [10], and calibration procedures [11]. This means that any member of the OpenFlexure project, or any member of the public, can browse, use, and build upon a huge body of collected knowledge in a way that would not be possible if the project was stored in a cloud storage program such as Google Drive or Dropbox. GitLab—the platform we use to share the project—also has project management features allowing the team to discuss changes to the design, problems, and feature requests in a public forum. These "issue threads" allow a distributed team to effectively communicate while producing a permanent and public record of the discussion. As this record includes usernames and dates, this may make people reluctant to publicly ask for explanations or advice for fear of damaging how they are perceived [12]. However, the larger the project gets, the more likely it is that more collaborators may have the same question, and answering enquiries causes the decision makers to consider and justify choices clearly.

In practice, however, we have found that a platform like GitLab can somewhat isolate some members of the team. Subtle differences in key terminology is often difficult for new users to learn, such as the difference between project branches and forks, commits and pushes, clones and checkouts, etc. Furthermore, the simple act of modifying a file can involve multiple steps, including checking for changes in the "master" version of the project, merging these changes into a local copy, modifying the file, syncing this change to a personal copy of the project on the server, and opening a *merge request* to have these changes merged into the main repository.

This workflow is important for members of the team to be able to work independently. The numerous steps, the number of possible options during each step, and the possibility of a merge not being accepted if the often unclear workflow was not followed can be frustrating and off-putting. These workflows require not only the technical understanding of key terminology, but also a grasp of how each project community expects them to be used. As version control creates permanent records, any such mistakes will be visible to collaborators indefinitely. Since most tools for interacting with Git repositories are written for computer programmers, the above workflow is usually performed on a command line interface which can be daunting to new team members. While graphical interfaces such as GitHub Desktop or GitKraken do exist, they are often either too simple for this workflow, or too complicated for inexperienced users. These problems are compounded by the difficulty of resolving conflicts when multiple members of the team edit the same file. This leads to members being reluctant to share their improvements, keeping them locally or sharing them through other mechanisms.

Problems with the Git workflow can lead to certain files being lost from the project, becoming out of date, or having misleading history information (e.g. hiding a team member's contribution). This workflow often only records successful modifications. This means that contributors can repeat previously attempted but undocumented modifications, rediscovering the same problems. Even issue threads can be problematic, as they are not easy to access through a phone, and only notify participants by email. From a university laboratory it may be simple to open a web-browser when something goes wrong, but from a workshop in Dar es Salaam one may have to find a laptop and tether it to a phone before being able to interact with the issue thread. This extra inconvenience drives conversation to platforms such as WhatsApp, which are neither public nor simple to archive and search. To understand the difference in workflow between the teams, we have found that regular and extended visits between members of both teams are essential to build understanding of each other's workflow.

Using a software development platform such as GitLab has also had benefits for members of the team less comfortable with programming. By using the Continuous Integration (CI) features, we have been able to automatically generate documentation and publish to our website, automatically build and zip the 3D design files ready for printing from the CAD code, package the microscope software into an installable application, and even build an SD card image for the Raspberrry Pi computer that powers the microscope with all necessary software pre-installed.

We have found that underlying technology that powers software development platforms has all the features needed for remote hardware development. What is lacking is the user-interface to make the project more accessible to a wider audience. We would hence advise that members of the open hardware and humanitarian communities that are interested in building development platforms consider building upon open source software development platforms rather than trying to re-invent version control.

III. Conveying essential information

A. Computer aided design

The key design information for the microscope is in computer aided design (CAD) files. The choice of CAD packages is always problematic for any distributed hardware project, as most CAD packages cannot use each other's files. This means that teams that are used to different CAD packages either design in two incompatible packages or need to learn a new package. Interoperability is partially solved by exchange formats such as STEP or STL files, however these only capture the object shape, not the design constraints. As such it is very hard to edit and improve a STEP or STL file.

When running an open source hardware project, especially one with humanitarian goals, it is important to strike a balance between open source purity and pragmatism. Open source CAD packages such as FreeCAD, OpenSCAD or LibreCAD are less advanced than the proprietary options and are often less well known. However, the high price of proprietary options and the number of incompatible proprietary CAD packages means that, whichever option was chosen, a large number of potential collaborators not be using their preferred tool. The OFM primarily uses OpenSCAD, an open source text-based CAD language, for its design. By choosing an open source option, we can at least guarantee that price does not affect availability of the package.

The other big choice is whether to use a text-based or a graphical CAD package. Graphical CAD has the benefit of a faster learning curve, and as most CAD packages are graphical, CAD skills are largely transferable between different packages. Also, graphical CAD does not require programming experience. Text-based CAD, however, has the benefit that it is very simple to compare versions, which is important for collaboration. It also doesn't suffer from topological renaming issues where changes to one part of a structure break parts that were drawn later; this is partially problematic for a very complex monolithic structure like the main body of the OFM. While there are practical reasons for wanting to ensure that the entire design is in a consistent CAD package, this has resulted in certain components designed in other packages not being shared effectively. As a project, we need to find a way to ensure that the consistency that allows us to build the core printable files in the cloud does not stop us capturing work that is done in other packages.

B. Documentation

In addition to the CAD files that are needed to print the components, a large amount of supplementary documentation is also required. This is complicated by the fact that multiple versions of the microscope exist, for example to allow different cameras, or lower cost optics for educational use. As such there is no one single bill of materials either for the printed parts or for other components that need to be purchased. As the design is continuously evolving, it is difficult to ensure that all the instructions for all versions are maintained. In addition to assembly instructions, the project requires documentation for the software and for use of the microscope, as well as documentation about our work flow, how we design and even how we document.

Our documentation is produced in a markup language called Markdown. Markdown can easily be converted into a website and different versions can be compared using the same revision control tools we use for the code. Compared to most markup languages, the syntax is very simple. However, our use of Markdown is yet another barrier to entry for collaborators who are used to standard word processing packages.

For software development, a number of tools, such as Sphinx for Python, allow documentation to automatically be generated from information embedded in the code. Hardware development requires its own tools that can track the components and tools used in assembly procedures. This would allow multiple variations of a project to generate documentation by selecting from the same set of procedures, helping the documentation stay consistent. We have begun a project called GitBuilding [13], where we are trialing the implementation of these features. The syntax for GitBuilding is a superset of Markdown, which allows it to interact well with revision control tools and simplify development. However, by adding more syntax to hold this meta information, it is possible that we will make the documentation harder to write for some members of the community. For this reason we need to carefully consider the user interface of the software to make it as inclusive as possible.

IV. Conclusion

We have described how the OpenFlexure project has used software development platforms to enable distributed development of a complex scientific instrument in a way that has ensured that it is both of laboratory quality and can be produced locally in Tanzania. While adoption of these platforms has allowed us to curate a rich history of the development and automate a number of difficult procedures, it has not been without its drawbacks. Primarily, the drawback is that these platforms have been designed for software programmers, and the cryptic terminology and intimidating menus create a barrier to entry that can deter collaboration. We believe that these issues largely stem from the user interface of the platform and its associated software rather than being a limitation of the core functionality. We have also described the difficulties of maintaining complete and up-to-date documentation, and the frustrations that stem from incompatible CAD packages. Our team is actively working on an open source solution to a number of these documentation problems, and would welcome external input.

Acknowledgement

The authors would like to acknowledge all members of the community that have interacted with the OpenFlexure project over the years, especially those who submitted bug reports, asked probing questions on issue threads, and contributed code.

References

[1] J. T. Collins, J. Knapper, J. Stirling, J. Mduda, C. Mkindi, V. Mayagaya, G. A. Mwakajinga, P. T. Nyakyi, V. L. Sanga, D. Carbery, L. White, S. Dale, Z. J. Lim, J. J. Baumberg, P. Cicuta, S. McDermott, B. Vodenicharski, and R. Bowman, "Robotic microscopy for everyone: the openflexure microscope," *Biomed. Opt. Express*, vol. 11, no. 5, pp. 2447–2460, May 2020.

[2] J. P. Sharkey, D. C. Foo, A. Kabla, J. J. Baumberg, and R. W. Bowman, "A one-piece 3d printed flexure translation stage for open-source microscopy," *Review of Scientific Instruments*, vol. 87, no. 2, p. 025104, 2016.

[3] S. D. Grant, G. S. Cairns, J. Wistuba, and B. R. Patton, "Adapting the 3d-printed openflexure microscope enables computational super-resolution imaging," *F1000Research*, vol. 8, no. 2003, p. 2003, 2019.

[4] Q. Meng, K. Harrington, J. Stirling, and R. Bowman, "The openflexure block stage: sub-100 nm fibre alignment with a monolithic plastic flexure stage," *Opt. Express*, vol. 28, no. 4, pp. 4763–4772, Feb 2020.

[5] World Health Organization, *Malaria microscopy quality assurance manual-version 2*. World Health Organization, 2016. [Online]. Available: https://www.who.int/malaria/publications/atoz/9789241549394/en/

[6] ——, "World malaria report 2019," 2019. [Online]. Available: https://www.who.int/publications-detail/world-malaria-report-2019

[7] ——, *Guidelines for the treatment of malaria*, 3rd ed. World Health Organization, 2015.

[8] GitLab repository of main OFM hardware https://https://gitlab.com/openflexure/openflexure-microscope.

[9] GitLab organisation that holds the OpenFlexure Project repositories https://gitlab.com/openflexure.

[10] GitLab repository of Sangaboard motor controller https://gitlab.com/bath_open_instrumentation_group/sangaboard.

[11] R. Bowman, B. Vodenicharski, J. T. Collins, and J. Stirling, "Flat-field and colour correction for the raspberry pi camera module," *Journal of Open Hardware*, vol. 4, p. 1, 2020.

[12] A. W. Brooks, F. Gino, and M. E. Schweitzer, "Smart people ask for (my) advice: Seeking advice boosts perceptions of competence," *Management Science*, vol. 61, no. 6, pp. 1421–1435, 2015.

[13] https://gitbuilding.io.