Technische Universität Berlin

Open Source Hardware Seminar – Class 4

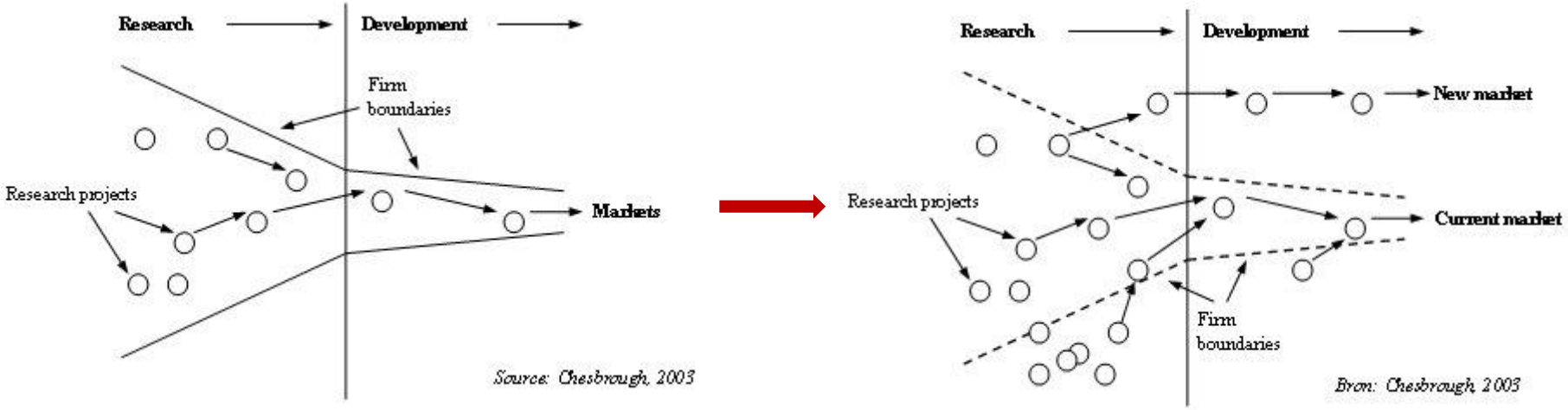
Julieta Arancio, Robert Mies
Technische Universität Berlin, Institute for Machine Tools and Factory Management, Chair of Quality Science

TOC

- Background: open innovation, open design
- Practices in the OSH community
- Version control systems
- Modular design
- OSH specifics
  - Bills of Materials
  - Sharing source files

Background
Open innovation

Opening the firm boundaries to knowledge flows



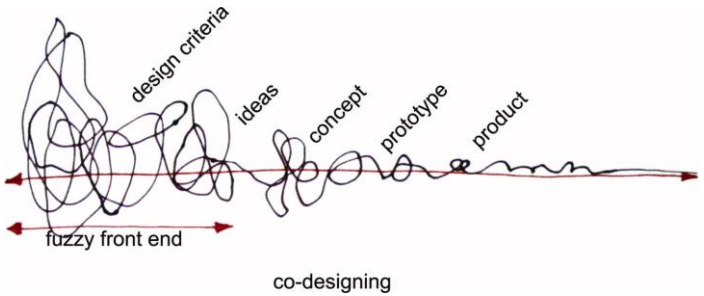
Chesbrough, H. (2003), "Open Innovation: The New Imperative for Creating and Profiting from Technology", Harvard Business School Press.

Background
Open design

Seizing collective creativity

Motivations

- Designing future experiences for people, community and cultures
- Users are informed and connected as never before
- Roles in the design process are changing
  - Users become co-designers
  - Researchers become facilitators
  - Professional designers facilitate the workflow towards production



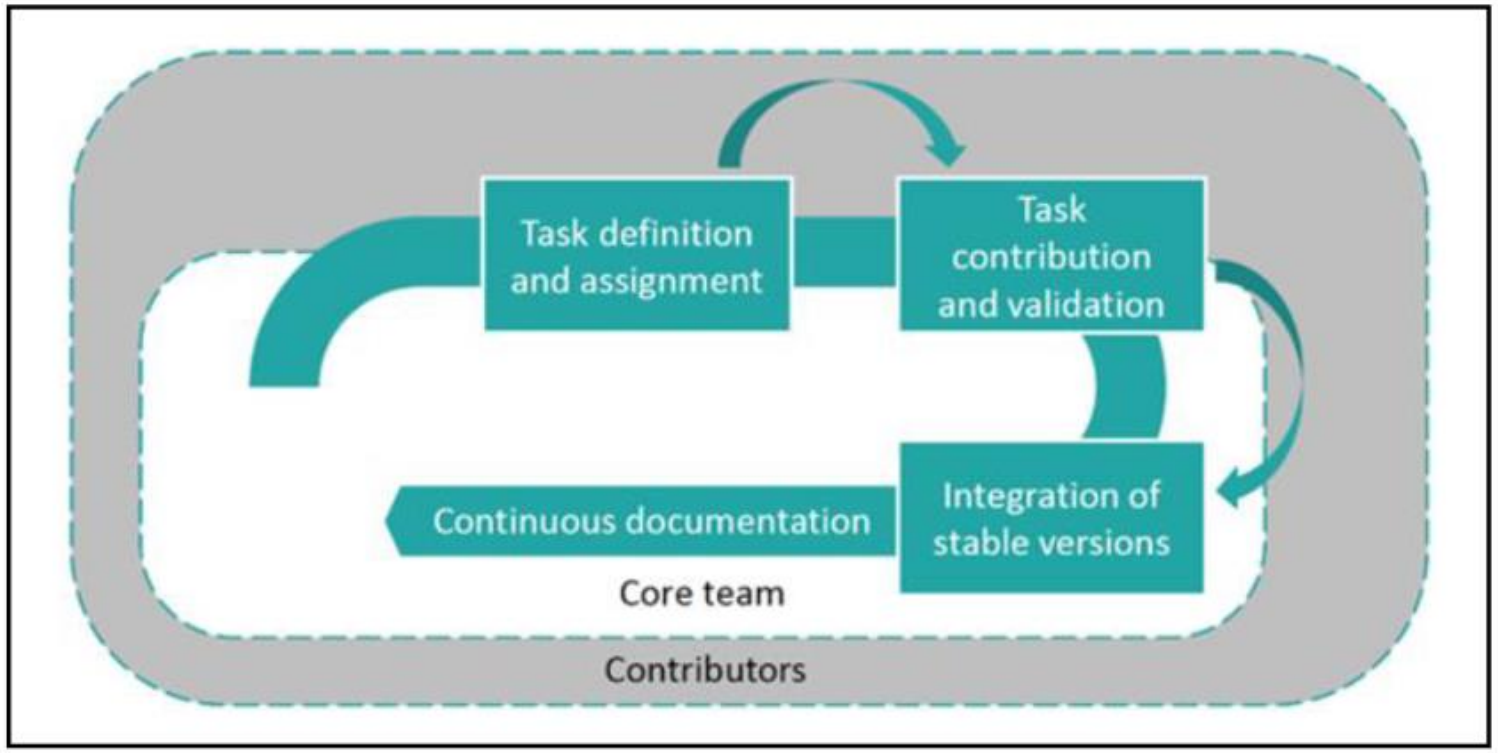
Four levels of creativity identified by Sanders et al (2008):

Level	Type	Motivated by	Purpose
4	Creating	Inspiration	'express my creativity'
3	Making	Asserting my ability or skill	'make with my own hands'
2	Adapting	Appropriation	'make things my own'
1	Doing	Productivity	'getting something done'

Sanders, E. B. N., & Stappers, P. J. (2008). Co-creation and the new landscapes of design. Co-design, 4(1), 5-18. https://doi.org/10.1080/15710880701875068

Background
Collaborative capacity in OSH

The open source product development process



Mies, R., & Bonvoisin, J. (2022, June). **Collaborative capacity of open source hardware communities: A cross-sectional study.** In 2022 IEEE 28th International Conference on Engineering, Technology and Innovation (ICE/ITMC) & 31st International Association For Management of Technology (IAMOT) Joint Conference (pp. 1-9). IEEE.

Background
Collaborative capacity in OSH

The open source product development process: for companies

Poor practices	Good practices
Exclusivity vs. reciprocity	
! Hold knowledge as an advantage over others ! Close up knowledge on technology ! Giving away company IP or divulging competitive intelligence to competitors	✓ Treat knowledge as a shared resource ✓ Seek collaborative advantage through the creation of synergies ✓ Codetermine OSH project aims on an equal footing within OSH communities
Mass vs. niche focus	
! Unrealistic expectations of rapid scaling	✓ Engage in a focus strategy that defines aligned business activity sets ✓ Reach customers through highly personalized products

Source: Mies, R., Bonvoisin, J., & Jochem, R. (2019). Harnessing the Synergy Potential of Open Source Hardware Communities. In: Redlich T., Moritz M., & Wulfsberg J. (Eds.). Co-Creation. Management for Professionals, pp. 129-145. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-97788-1_11

Background
Collaborative capacity in OSH

The open source product development process: for companies

Poor practices	Good practices
Hierarchical vs. self-governed workflows	
! Unilateral agenda setting ! Hierarchical chain of commands ! Unfashionable project management practices	✓ Resource deployment by means of a swarm organisation ✓ "Release early" of designs, ideas, documents ✓ Switch from managing to coaching
Control vs. self-propelled learning	
! Extrinsic incentivization (such as monetary rewarding) ! Try to get control over others	✓ Rely on intrinsic motivations ✓ Treat failure as an integral part of a learning- and feedback-based process ✓ Leave the needed room in projects for organic progress to occur

Source: Mies, R., Bonvoisin, J., & Jochem, R. (2019). Harnessing the Synergy Potential of Open Source Hardware Communities. In: Redlich T., Moritz M., & Wulfsberg J. (Eds.). Co-Creation. Management for Professionals, pp. 129-145. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-97788-1_11

Background
Collaborative capacity in OSH

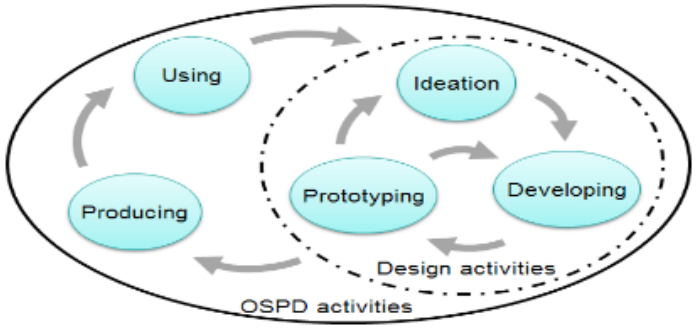
Degrees of adoption

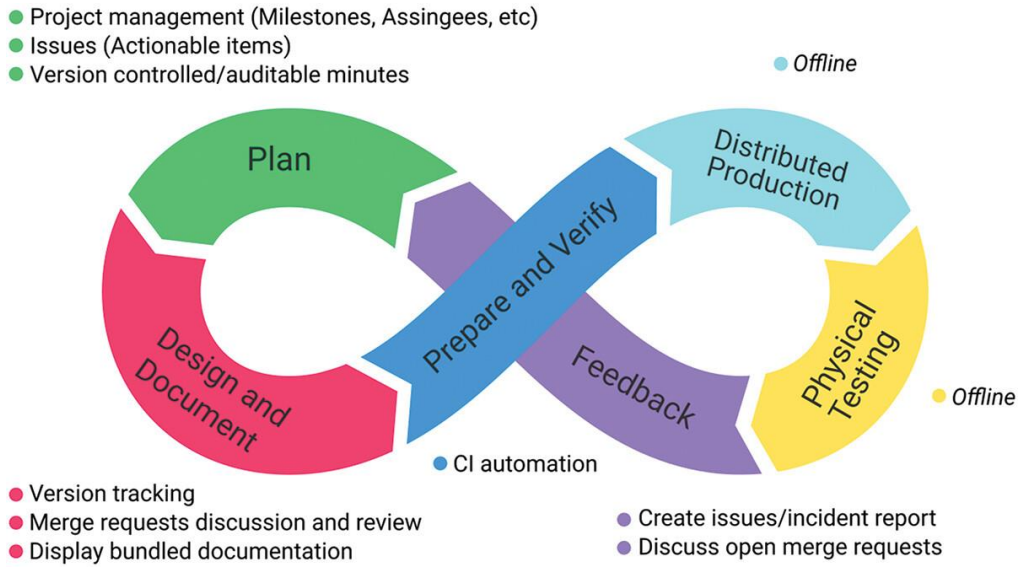
Degree of adoption	Practices
Widely adopted	- Separated, distributed tasks - Reaching collective understanding (trust, goals) - Task definition/assignment is modular - Sharing source files in editable format
Less adopted	- Issue tracking - Contribution guides - Core team meetings - Sharing between contributors - Self-initiation - Review of contributions - Use of VCS
Minimum adoption	- Active recruitment, physical meetings - Task assignment to contributors - Regular updates on tasks

Mies, R., & Bonvoisin, J. (2022, June). Collaborative capacity of open source hardware communities: A cross-sectional study. In 2022 IEEE 28th International Conference on Engineering, Technology and Innovation (ICE/ITMC) & 31st International Association For Management of Technology (IAMOT) Joint Conference (pp. 1-9). IEEE.

Practices in OSH
Version Control Systems (VCS)

A process approach to OSH projects



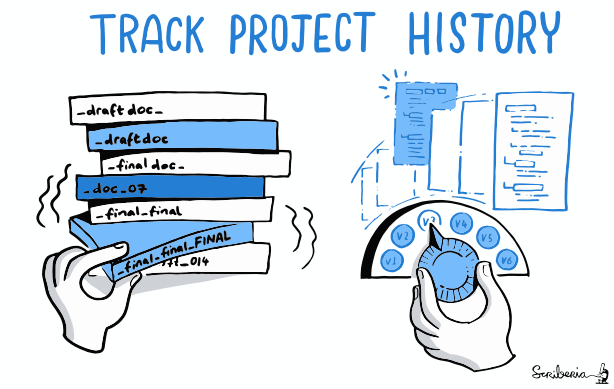


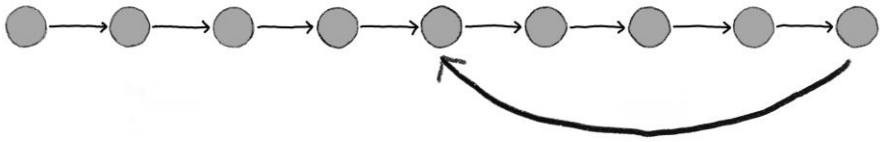
Left: Boujut, J. F., Pourroy, F., Marin, P., Dai, J., & Richardot, G. (2019, July). Open source hardware communities: investigating participation in design activities. In Proceedings of the Design Society: International Conference on Engineering Design (Vol. 1, No. 1, pp. 2307-2316). Cambridge University Press.

Right: Stirling, J., Bumke, K., Collins, J., Dhokia, V., & Bowman, R. (2022). HardOps: utilising the software development toolchain for hardware design. International Journal of Computer Integrated Manufacturing, 35(12), 1297-1309

Practices in OSH
Version Control Systems (VCS)

Definition and motivations for using VCS





Without VCS	With VCS
Several copies of the same file scattered around	Latest version with accessible history allowing traceability
Inefficient collaboration	Notifications when other users update same files
Overwriting	Conflict management

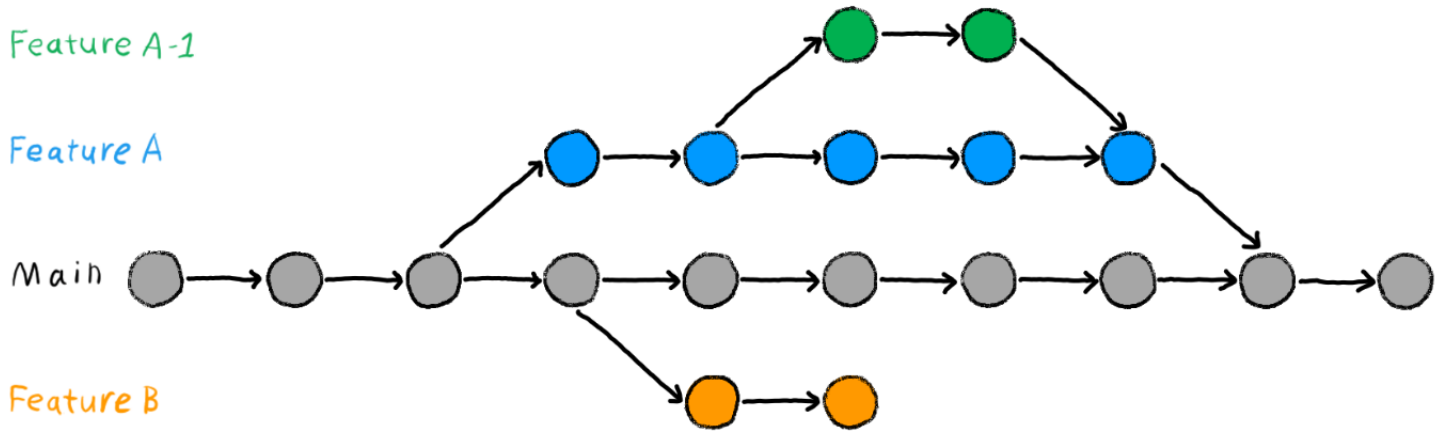
The Turing Way Community, Becky Arnold, Louise Bowler, Sarah Gibson, Patricia Herterich, Rosie Higman, Anna Krystalli, Alexander Morley, Martin O'Reilly, & Kirstie Whitaker. (2019). The Turing Way: A Handbook for Reproducible Data Science (v0.0.4). Zenodo. https://doi.org/10.5281/zenodo.3233986

Practices in OSH
Version Control Systems (VCS)

Managing contributions

Advantages

- Git-based workflows are more efficient for managing contributions in large projects
- The core team can approve/reject changes
- Contributors can work without affecting the main branch



The Turing Way Community, Becky Arnold, Louise Bowler, Sarah Gibson, Patricia Herterich, Rosie Higman, Anna Krystalli, Alexander Morley, Martin O'Reilly, & Kirstie Whitaker. (2019). The Turing Way: A Handbook for Reproducible Data Science (v0.0.4). Zenodo. https://doi.org/10.5281/zenodo.3233986

Practices in OSH
Version Control Systems (VCS)

Software implementations

	git	GitHub	GitLab
Interface	Command-line program	Online platform + Desktop	Online platform + Desktop
Main focus	All features of VCS	All features of VCS + Project management	All features of VCS + Continuous integration / DevOps
Main uses	Created in 2005 to provide VCS to the Linux project	Full toolchain for software development management	Fully automating backend software processes

The Turing Way Community, Becky Arnold, Louise Bowler, Sarah Gibson, Patricia Herterich, Rosie Higman, Anna Krystalli, Alexander Morley, Martin O'Reilly, & Kirstie Whitaker. (2019). The Turing Way: A Handbook for Reproducible Data Science (v0.0.4). Zenodo. https://doi.org/10.5281/zenodo.3233986

Practices in OSH
Version Control Systems (VCS)

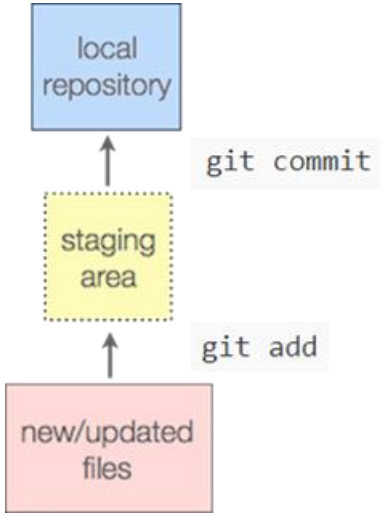
git advantages and limitations

Advantages

- All features of VCS
- Local work is ideal for distributed hardware teams

Limitations

- Significant learning curve
- Works best with text files
- Not suitable for big datasets >1TB



The Turing Way Community, Becky Arnold, Louise Bowler, Sarah Gibson, Patricia Herterich, Rosie Higman, Anna Krystalli, Alexander Morley, Martin O'Reilly, & Kirstie Whitaker. (2019). The Turing Way: A Handbook for Reproducible Data Science (v0.0.4). Zenodo. https://doi.org/10.5281/zenodo.3233986

Practices in OSH
Modular design

What is modularity by design and why it is important?

Definitions

- Modular design is a design principle that subdivides a system into smaller parts called modules.
- Modules can be independently created, modified, replaced, or exchanged between different systems

	No Modules	Modular design
Costs	Very expensive to customize, requires overhaul	Reduced as changes happen only in one part of the system
Interoperability	Non standardized, less interoperable	Facilitates reuse through standard adoption in modules
Learning curve for contributors	Created in 2005 to provide VCS to the Linux project	Less investment, contributors learn about the module they want to modify
Flexibility	Hard to update or augment functionality	Easy to customize and update
Obsolescence	Hard to repair	Easier to isolate malfunction / obsolete component

Arancio, Chagas, Kutschera (2019) Open Hardware Makers Curriculum. Available at https://curriculum.openhardware.space

Practices in OSH
Modular design

Modularity to encourage project reuse

Advantages

- Contributors can quickly identify features to improve or modify
- Standardized modules can be incorporated into forks
- Can contribute towards setting a “de facto” standard

Image: Dr. Adrian Bowyer and the first replication of the RepRap project

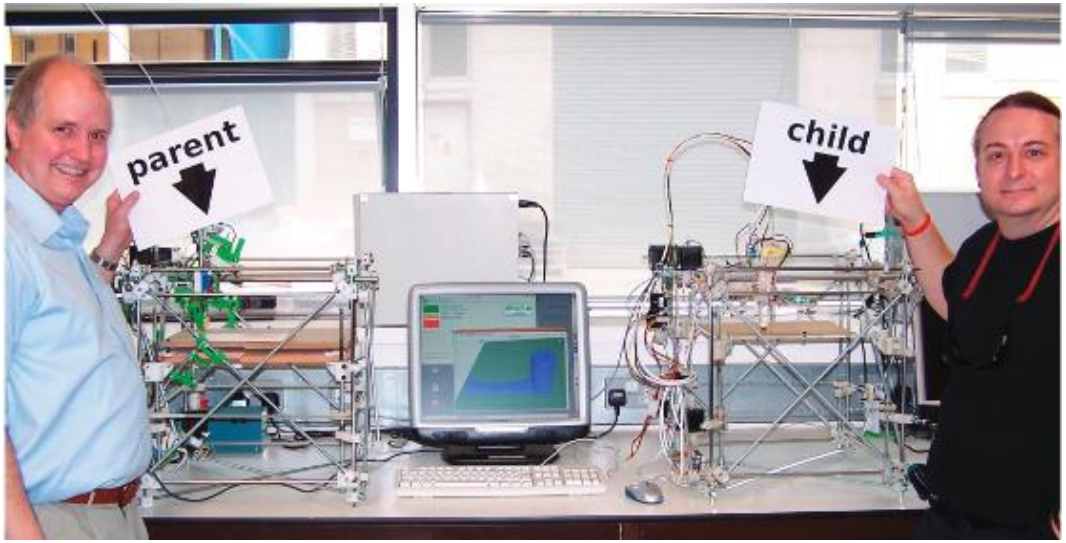
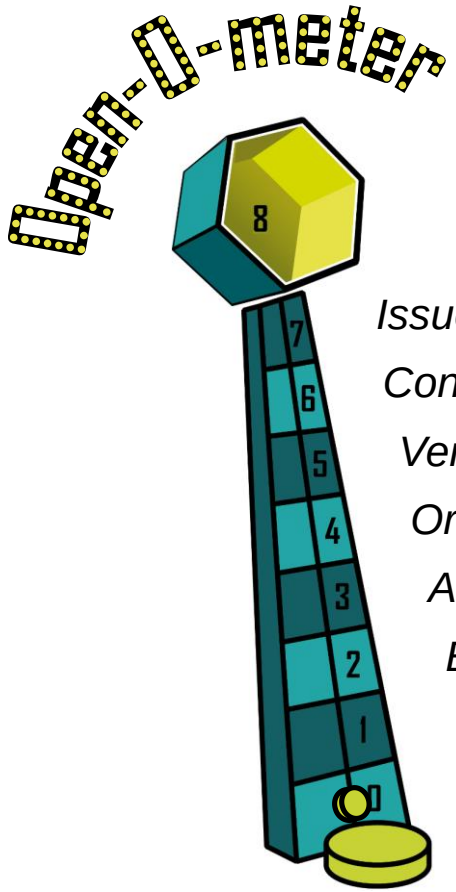


Image source: Wikimedia Commons https://commons.wikimedia.org/wiki/File:First_replication.jpg

Practices in OSH
The specifics of sharing hardware

Beyond software towards materials and designs



Issue tracking system
Contribution guide
Version control system

Process openness

Original files
Assembly instructions
Bill of materials
CAD files
OSH compatible license

Product openness

Bonvoisin, J., & Mies, R. (2018). Measuring Openness in Open Source Hardware with the Open-o-Meter. *Procedia CIRP*, *78*, pp. 388-393.

Practices in OSH
The specifics of sharing hardware

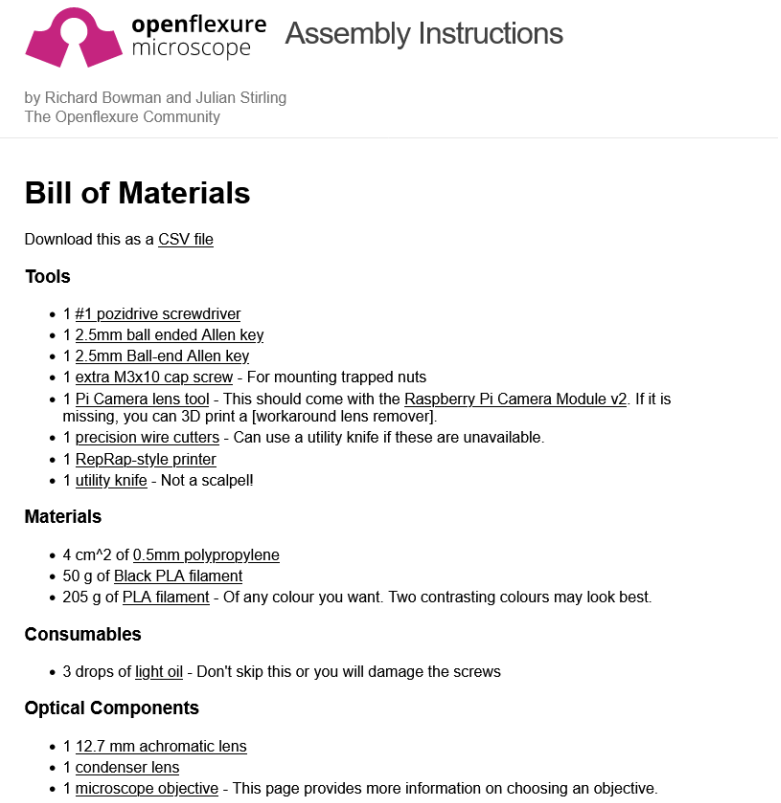
Bill of materials

A bill of materials is a list of the raw materials, sub-assemblies, intermediate assemblies, sub-components, parts, and the quantities of each needed to manufacture a product.

It is commonly named "BOM". Some projects also include bill of tools and bill of skills.

It should answer four main questions:

- What parts are used?
- Where can they be sourced?
- Criteria for use
- Requirements



openflexure microscope Assembly Instructions

by Richard Bowman and Julian Stirling
The Openflexure Community

Bill of Materials

Download this as a CSV file

Tools

- 1 #1 pozidrive screwdriver
- 1 2.5mm ball ended Allen key
- 1 2.5mm Ball-end Allen key
- 1 extra M3x10 cap screw - For mounting trapped nuts
- 1 Pi Camera lens tool - This should come with the Raspberry Pi Camera Module v2. If it is missing, you can 3D print a [workaround lens remover].
- 1 precision wire cutters - Can use a utility knife if these are unavailable.
- 1 RepRap-style printer
- 1 utility knife - Not a scalpel!

Materials

- 4 cm^2 of 0.5mm polypropylene
- 50 g of Black PLA filament
- 205 g of PLA filament - Of any colour you want. Two contrasting colours may look best.

Consumables

- 3 drops of light oil - Don't skip this or you will damage the screws

Optical Components

- 1 12.7 mm achromatic lens
- 1 condenser lens
- 1 microscope objective - This page provides more information on choosing an objective.

Image: BOM of the OpenFlexure microscope project, available at https://build.openflexure.org/openflexure-microscope/v7.0.0-beta1/high_res_microscope_BOM.html

Practices in OSH
The specifics of sharing hardware

Source files: CAD

Why is CAD relevant for OSH?

- Almost all OSH projects produce computer-assisted designs (CAD)
- Significant engagement point for contributors
- Multiple software options generating both source and export files in native formats
- Now available for electronics design: KiCAD

	Export file	Source file
Formats	.stp .stl .pdf	.svg .ods
Size	Light weight	Large files
Editable	No	Yes
Audience	Users	Contributors

Arancio, Chagas, Kutschera (2019) Open Hardware Makers Curriculum. Available at https://curriculum.openhardware.space

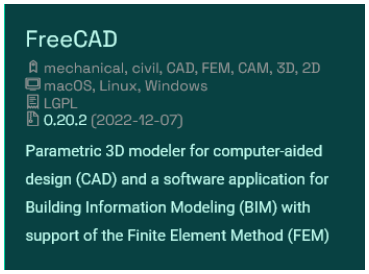
Practices in OSH
The specifics of sharing hardware

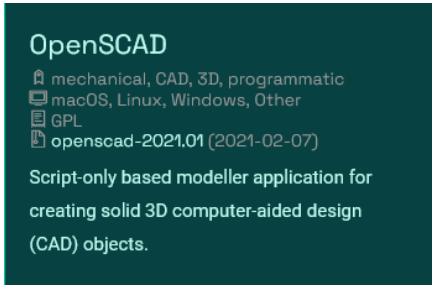
Software for hardware design

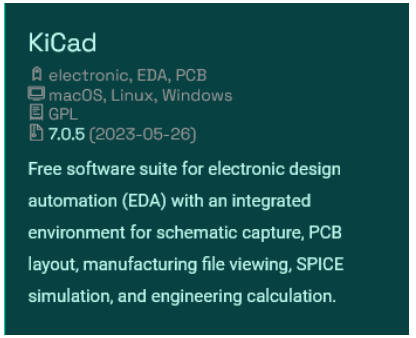
Today there is no 100% open toolchain for hardware design.

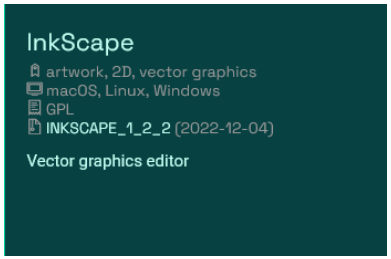
Proprietary vs FOSS

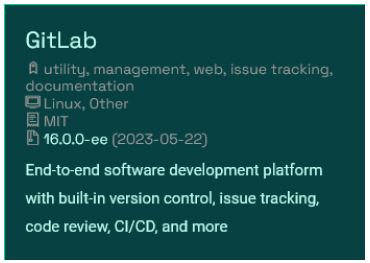
- Proprietary software often offers better functionality and User Experience
- Proprietary software diminishes interoperability due to native file formats & there is risk of lock-in
- FOSS is growing and specific projects improving (FreeCAD, KiCAD)
- Sharing editable formats is always the best option to minimize impact of software preference

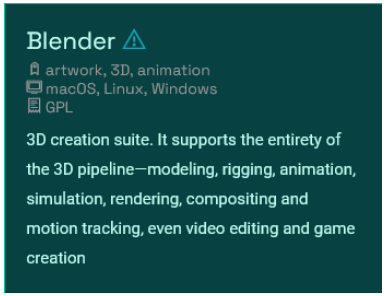








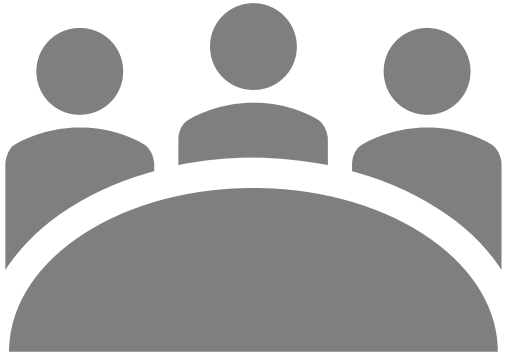




Open Toolchain Foundation, available at https://opentoolchain.org/tools/

Summary

- Open source hardware projects can be understood as Open Innovation activities
- While some of them happen through open design, this is not always the case
- Ideally, OSH project would follow a pattern similar to collaboration in FOSS
- In practice, the OSH community has implemented basic collaboration practices but more is needed
- Version Control Systems are key for enabling distributed collaboration in OSH
- Git is the most popular VCS together with the platforms GitHub and GitLab
- Modular design fosters contribution and is more efficient for maintaining OSH
- There is no open toolchain for OSH but sharing source files is key as a good practice



Julieta Arancio, Guest Researcher
Robert Mies, Research Associate
Technische Universität Berlin
Institute for Machine Tools and Factory Management
Chair of Quality Science
PTZ 3, Pascalstr. 8-9, 10587 Berlin
+49 (30) 314-25930
robert.mies@tu-berlin.de
arancio@campus.tu-berlin.de
www.qw.tu-berlin.de | www.openmake.de | www.opennext.eu

Thank you for your attention!