The Industrial Ecology Digital Laboratory

Konstantin Stadler[1,*)] Radek Lonka[1)] Evert Bouman[1)] Guillaume Majeau-Bettez[1,2)] Anders Hammer Strømman[1)]

1) Industrial Ecology Programme, NTNU, Trondheim, Norway. 2) CIRAIG, École Polytechnique de Montréal, Canada. *) Corresponding and presenting author

1. Introduction

Cutting edge research requires the adoption or even novel development of data management and analysis software. This is mostly done by researchers, either building upon available, more generic software tools or developing new analysis routines. Researchers, however, are seldom trained as software developers; quality, re-usability and transparency of software code and data becomes one of the main obstacles for a smooth scientific development.

Acknowledging this, there is now a variety of offers available to researchers seeking to improve their coding skills. On the most basic level, in two subsequent articles (Wilson et al. 2014; Wilson et al. 2017) Wilson et al. propose a set of guidelines for developing scientific software and managing data. Initiatives like Software Carpentry (SoftwareCarpentry 2017), data carpentry (DataCarpentry 2017) or code refinery (CodeRefinery 2017) provide courses and online education material for researchers. To dive even deeper into the topic, free online course for specific programming languages are available on sites like khan academy (Academy 2017). Following this guidance, researchers can reach expert programming skills with clear benefits for code quality and re-usability. However, almost all these initiatives are targeting individual researchers. This can lead to a skewed distribution of programming skills within an institute. Software tools and scripts developed by these skilled researchers are mostly geared towards their own needs and often know how about how to uses these tools (and in case of close source/unpublished software the tool itself) move with the researcher when they switch institutes. Therefore, although institutes might have researchers with excellent coding skills, their is little build up of a common digital infrastructure for a specific institute. Recognizing this issue, the Industrial Ecology (IE) Programme at the Norwegian University of Science and Technology (NTNU, Trondheim - Norway) recently established a Digital Laboratory to foster a common digital infrastructure for the group.

1.1 The Industrial Ecology Programme at NTNU

IE studies “the flow [and stocks] of material and energy in industrial and consumer activities, the effects of these flows on the environment, and the influences of economic, political, regulatory, and social factors on the flow, use , and transformation of resources” (White 1994). IE brings together environmental, economic, and data research to analyse e.g. the life cycle environmental impacts of individual products, the flows and stocks of materials at different geographical scales, or the social, economic and environmental

footprints of nations. In addition, Industrial Ecologists continuously develop and improve the methods and tools to address these issues. As such, IE's objective is to contribute to the scientific basis for sustainable development.

The Industrial Ecology Programme, NTNU (www.ntnu.edu/indecol) was initiated in 1994 and established the world's first PhD programme in IE (2003). Currently, the IE Programme has 25 PhD students, 8 PostDocs, 2 senior researchers and 6 core faculty members. Given its interdisciplinary nature, the scientific and engineering background of the staff ranges from biology, chemistry and physics to (industrial) economics, energy science, and environmental policy. Current research at the IE programme is centred around the development and application of four main IE methods: life cycle assessment (LCA) and life cycle impact assessment, material flow analysis (MFA), and environmental applications of input-output analysis.

Because of the diverse activities in the IE field and the different backgrounds of the researchers (both scientifically and in terms of IT skills), a plethora of different software tools are used within the IE Programme. This ranges from spreadsheet macros, custom made LCA analysis software, and GIS analysis tools for spatialized impact assessment, to designated analysis software written in Matlab, Python and R. The activities of the IE Programme in the past decades, combined with continuous rotation of scientific staff, have led to the accumulation of software tools varying greatly in quality, complexity, and applicability. Most software was developed for specific projects or tasks, and, being a one-time deliverable, was not developed with re-use and extensive documentation in mind. In general, code reuse and inter-operability between the different areas covered by the IE Programme is currently quite limited. This mirrors the situation of the IE field in general. To address the latter, IE researchers from different institutes recently pushed for more quality and openness in IE software development (Pauliuk et al. 2015). Recognizing the development of the field towards increasingly complex software tools, for the situation inhouse, the IE Programme opted for the establishment of a Digital Laboratory.

2. The Industrial Ecology Digital Laboratory

The Industrial Ecology Digital Laboratory (IEDL) was established in July 2016. It currently consists of two full time positions, a Lead Researcher/Manager and a research software engineer, with additional short time employees for specific programming task. The main objectives of the IEDL are to

(a) consolidate available digital infrastructure and ease the integration of newly developed tools by providing code and data exchange standards across the group

(b) develop novel software based on common needs across the IE programme, thereby building an IE software and data toolbox

(c) explore synergies with other research groups by connecting to similar initiatives in the sustainability and environmental research community

The main premise of the IEDL organization is, that research software can be categorized into three layers (see Figure 1). The layers structure help to specify the tasks of the IEDL. For layer 1, generic tools, the main task of IEDL is to inform researchers about available resources (tutorials, seminars) and to integrate them into the available digital infrastructure of the IE Programme. In addition, there might be some development by IEDL (cf. the web_trawler, see Appendix). Building upon these generic tools, most

of the novel development executed by IEDL will happen in regard to the IE specific tools of layer 2. This layer provides the backbone of the digital infrastructure of the IE Programme. Cutting-edge research, however, requires the development of novel tools that are often quite specific to a certain research topic or project task. Not everything can be generalized. The specific tools of layer 3 will mostly be developed by researchers working on a specific research topic; IEDL will help and guide the development of these tools, ensure that they fully leverage the code-base of IE tools of Layer 2, and help integrating the new code into the underlying layers if a more general use appears.

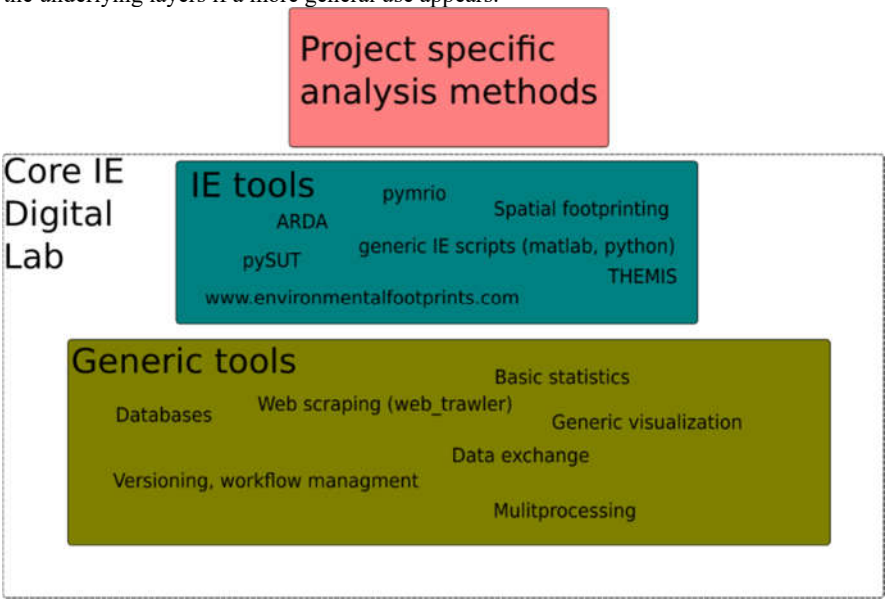

Fig 1. Pyramid of digital tools in IE research. Tools are categorised in three layers: (1) tools which are generic to all research fields, (2) IE specific tools and (3) tools developed for one specific projects /research task. IEDL is involved differently depending on the layer: (1) IEDL mainly provides tutorials (2) main development focus of IEDL, (3) IEDL helps/guides development and, if potentially useful for subsequent projects, merges into layer (2). Some of the tools mentioned here (pymrio, data visualization, web_trawler) are already available as open source packages - see Appendix.

2.1 Steps by step guide for setting up a Digital Laboratory

Here we give a short summary of our activities in the first year of the IEDL. The main topic of the first year was standardization and harmonization of knowledge exchange, code and data. The following steps give a comprehensive and reusable description of how to set-up a Digital Laboratory.

Knowledge Exchange

The gathering and exchange of knowledge is one of the basic tasks of researchers and research institutes. While for the "classic" scientific content, ready to use frameworks exist to help with this task (reference

managers and note-taking apps, both often implementing shared spaces for collaborations) and are commonly used by researchers, knowledge gathering for used software, data management and other IT topics often happen only ad hoc. Already prior the the IEDL, the IE Programme and its researchers gathered a lot of information about how to use specific IT tools (such as programming language tutorials, HPC usage guides tutorials, tips and tricks collections). These, however, were not structured in any way, often stored in some random subfolder and hard to find again. In order to establish a more organised knowledge exchange regarding digital infrastructure topics, we set-up an internal wiki system. This became the central hub for all information about the digital lab, all IT projects and either directly links or directly contains tutorials and documentations.

We used the DokuWiki (DokuWiki 2017) system at the IEDL. This is simple to set-up and come with a minimal maintenance overhead. The wiki now also includes a summary of persons working at the IE Programme and their competences, as well as discussion and feedback pages.

We established the wiki after a failed attempt to just index all IT related documents on the network drives of the institute using two different open source tools: Open Semantic Search (Mandalka 2017) and Open Semantic Server (OSS 2017). Both tools has some difficulties with setting up and maintenance overhead. It has also too many false positives for many searches. We therefore strongly recommend to used a centralized knowledge exchange platform.

An important point while setting up a new system is to keep the entry point as low as possible: don't enforce any style requirement and the beginning, adding information to the system should be fast and easy. Let the system grow and only fix the top-level structure. However, aim to increase usage of the new system by providing links to answers at the wiki upon email questions.

Code repositories

As for knowledge exchange, a central point for code development should be established to keep history and make collaborating easy. The designated folder on a shared network drive, which was used previously, had difficulties when researchers started collaborate and copy/overwrite changes. In addition, no mechanism for keeping track of issues/todo/enhancements was in place. Therefore, we recommend to use a designated version control system like Github (Gitub 2017), Bitbucket (Bitbucket 2017) or Gitlab (Gitlab 2017). After testing all three services, we settled for Gitlab due to unlimited private repositories for Academia and fine grain access control within a group of projects.

Workflow mapping

Because the aim of the IEDL is to provide digital infrastructure across the IE Programme, we needed detailed knowledge about the data and software needs of every researcher in the Programme. Having that information would than allow us to find commonalities among the needs of the different researchers in the group. In order to do so, we asked each researcher to map out common workflows and tasks. Importantly, we not only wanted to get information about the computation/automated steps but also about manual steps.

This resulted in dozens of workflow diagrams, which we subsequently screened for commonalities. Due to the diverse research topics in the group, most of the workflows were quite unique (corresponding to layer 3 in Fig 1). We found the most overlap in visualization routines and in reformatting (in particular

aggregation) of data. Regarding the manual steps, we discovered that manually downloading data still appears to be a common time consuming step.

Based on the workflow screening, we decided to focus the first development efforts for the IEDL on (interactive) visualization, data handling and retrieval.

Establish documentation standards

The IE Programme at NTNU has a strong background in developing analysis tools in Matlab. In recent years, however, there were some efforts to bring the code base of the IE Programme to a more open programming language (Pauliuk et al. 2015). The current situation is a mixture with a large code base in Matlab, novel code development mainly in Matlab and Python with some additional pieces in GAMS, C and Fortran. We believe, having a diverse ecosystem of tools at hand, allows to choose the best tool for a certain task. We, therefore, do not enforce a specific programming language (most can be called within each other). We are, however, started to suggest a unified documentation standard to be used for all programming languages. We base this standard on the numpy docstring format (numpy 2017) which we also recommend for Matlab and GAMS documentation. Following a common standard enabled us to use a unified parsing routine for all documentation. We are currently building a common documentation hub for all scripts, functions and programmes available at the IE Programme. We take into account, that some people at the IE Programme are not experienced programmers. The entry point for the documentation is one line after a function definition or script start.

Different OS

Researchers and the IE Programme are free to choose their preferred operation system. Hence,MS Windows, various Linux flavours and Mac OS are present at the Programme. The IEDL accommodates tor that fact by (1) make use of cross platform programs as far as possible (python, matlab, libre office; not using MS Office programs) (2) providing access to a Linux based server for heavy computing tasks and Linux specific tools and (3) started using docker containers to enable cross-platforms functionality in a reproducible manner. In addition, the Linux server of the IEDL provides a JupyterHub with various kernels (python3, matlab, octave, bash, javascript, R, julia, c++) which can be used all users of the IE Programme. More kernels will be installed if necessary.

User involvement and researcher - research engineers interactions

Foremost, IEDL aims to provide the infrastructure for the IE Programme. This, however, also includes to train people using and extending the available infrastructure. The main problem doing so are the different background of researchers working at the IE Programme; providing Programme wide seminars are thus not as effective as targeted sessions for a specific user group. We know adopted a project based education framework: When the IEDL takes part in a specific projects, we present the best tools and how to use them to the researchers involved in the project. For the most part, this includes git and gitlab (including how to use remote repositories, branching, how to use the issue tracker, how to do merge requests), code style (including setup unit tests) and review (either within merge requests or through reviewboard (ReviewBoard 2017)).

For project development we adopted an agile development (Beck et al. 2001) like framework. Each research projects with involvement of the IEDL needs to have a main contact person (researcher) which acts as a project/product owner. We do so in order to keep a short communication cycle (daily to weekly) which helps to contentiously re-prioritise requirements and refine implementations. After the first working version of a specific software, all development is based on issues and enhancement requests through the git repository.

Communication channels

Based on the email overload of some researchers we tried to implement alternative communication channels (e.g. IRC, slack, …). This did not get good response from the researchers of the Programme, which felt that it was adding overhead by opening another communication channels besides email. Therefore, we returned to emails (and Programme wide email lists) for communication to researchers. Internally, we now use slack (slack 2017) for short, ad-hoc communcations. All discussions about a specific project or software happen through the issue tracker discussion channel of the specific project (see http://www.yegor256.com/2016/08/23/communication-maturity.html), which we enforce also for outside communication.

Reoccurring problems

During the establishment we face some reoccurring problems which we did not anticipate before.

First, users/researchers are very reluctant to open multiple accounts. Throughout the NTNU system, users only have to use one account to access their email, print, log-in to different workstations, loan from the library, etc. This habit somehow restricts the amount of tools we can use on a day to day basis every new account and password is considered overhead. For common users we now settled for 1. an account for the wiki system, 2. an account for the gitlab repository, 3. an account on the internal server and 4. an account on the code review system. Ideally, all these login could be part of one research infrastructure account. This, however, seems currently out of reach.

The second reoccurring problem was that some people are intimidated by command line tools and scripts in a non-familiar programming language. We found, that documentation is mostly in-sufficient to overcome this fear. Instead, providing step-by-step tutorials (e.g. through interactive Jupyther Notebooks) help to overcome this issue.

3. Discussion and Outlook

In its first year, the IEDL provided the backbone of a digital infrastructure for knowledge and code exchange. We standardised the ways the IE Programme develops codes and found commonalities across the IE Programme as starting point for further development. In the spirit of an open methodology approach, we share the steps and mistakes of the establishment, in order to be reused by other groups aiming for a designated Digital Laboratory.

After setting up the basic infrastructure and identifying the most urgent and promising task, we are now starting to harmonise and advance the core methodologies available for IE research. We are currently setting up a interactive notebook based workflow (based on Jupyter notebooks), which are hosted on the internal server of the IE Programme. Implementing this server based approach not only solves the platform diversity

issue, but also improves the reproducibility of current research as the notebooks can easily be published as supplementary material. In addition, using a common platform for data analysis should allow to streamline data visualisation tasks, one of the most requested features by the IE Programme.

The main development focus of the IEDL in the next year will be to connect the methodologies of IE research. Building on the digital infrastructure already set in place, we will connect LCA , MFA and MRIO methods in order to allow highly detailed analysis of human development and emerging technologies, taking into account the global socio-economic metabolism. This will happen with strong involvement of researchers in the different domains, following the Agile development strategy described above.

Finally, the IEDL aims to connect to similar initiatives, both within the IE field and in the scientific community in general. We already establish contacts with Data Carpentry (DataCarpentry 2017), and currently reaching out to the The Software Sustainability Institute (SSI 2017). In the ideal case, we envisage a network of Digital Labs, sharing information and best practices about infrastructure build up, code development and data organisation.

4. Appendix

We provide a list of open source software the IEDL has published or contributed to since its foundation:

country converter (coco)

A Python package for converting country names between different classification schemes and to match different forms of a certain country name with regular expressions (Stadler 2017).

https://github.com/konstantinstadler/country_converter

Datavis for specific data visualization.

Providing world map, sankey and nested pie graphs

https://datavis.indecol.no/

pymrio

A python module for automating input output calculations and generating reports (Stadler 2015).

https://github.com/konstantinstadler/pymrio

web-trawler

Bulk download files from a given url.

https://gitlab.com/dlab-indecol/web_trawler

5. References

Wilson, G., D.A. Aruliah, C.T. Brown, N.P.C. Hong, M. Davis, R.T. Guy, S.H.D. Haddock, et al. 2014. Best Practices for Scientific Computing. Ed. by Jonathan A. Eisen. *PLoS Biology* 12(1): e1001745. https://doi.org/10.1371%2Fjournal.pbio.1001745.

Wilson, G., J. Bryan, K. Cranston, J. Kitzes, L. Nederbragt, and T.K. Teal. 2017. Good enough practices in scientific computing. Ed. by Francis Ouellette. *PLOS Computational Biology* 13(6): e1005510. https://doi.org/10.1371%2Fjournal.pcbi.1005510.

SoftwareCarpentry, S. 2017. Software Carpentry. http://software-carpentry.org//index.html.

DataCarpentry. 2017. Data Carpentry. http://www.datacarpentry.org/.

CodeRefinery. 2017. Code Refinery. http://coderefinery.org/.

Academy, K. 2017. Khan Academy. http://www.khanacademy.org.

White, R.M. 1994. Preface. In *The Greening of Industrial Ecosystems*, ed. by B.R. Allenby and D.J. Richards. Washington D.C.: National Academies Press.

Pauliuk, S., G. Majeau-Bettez, C.L. Mutel, B. Steubing, and K. Stadler. 2015. Lifting Industrial Ecology Modeling to a New Level of Quality and Transparency: A Call for More Transparent Publications and a Collaborative Open Source Software Framework. *Journal of Industrial Ecology* 19(6): 937–949. https://doi.org/10.1111%2Fjiec.12316.

DokuWiki. 2017. DokuWiki - It's better when it's simple. https://www.dokuwiki.org/dokuwiki#.

Mandalka, M. 2017. Open Semantic Search. https://www.opensemanticsearch.org/.

OSS. 2017. Open Search Server. http://www.opensearchserver.com/.

Gitub. 2017. Github. https://github.com/.

Bitbucket. 2017. Atlassian Bitbucket. https://bitbucket.org/product.

Gitlab. 2017. Code, test, and deploy together with GitLab. https://gitlab.com/.

numpy. 2017. Numpy docstring format specification. https://github.com/numpy/numpy/blob/master/doc/HOWTO_DOCUMENT.rst.txt.

ReviewBoard. 2017. Review Board. https://www.reviewboard.org/.

Beck, K., M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, et al. 2001. The Agile Manifesto. http://agilemanifesto.org/.

slack. 2017. slack. https://slack.com/.

SSI. 2017. The Software Sustainability Institute. https://www.software.ac.uk/.

Stadler, K. 2017. The country converter coco - a Python package for converting country names between different classification schemes. *The Journal of Open Source Software* 2(16). https://doi.org/10.21105/joss.00332.

Stadler, K. 2015. Pymrio - a Python module for automating input output calculations and generating reports. In *EnviroInfo & ICT4S - Adjunct Proceedings (Part 2)*, 235–235.