Track 1 Paper: Good Usability Practices in Scientific Software Development

Francisco Queiroz\textsuperscript{1} Raniere Silva\textsuperscript{2} Jonah Miller\textsuperscript{3}
Sandor Brockhauser\textsuperscript{4} Hans Fangohr\textsuperscript{5}

\textsuperscript{1}Tecgraf Institute, PUC-Rio — Department of Arts & Design, PUC-Rio — Rio de Janeiro, Brazil

\textsuperscript{2}Software Sustainability Institute — United Kingdom
School of Computer Science — University of Manchester — Manchester, United Kingdom

\textsuperscript{3}Perimeter Institute of Theoretical Physics — Waterloo, Canada
University of Guelph — Guelph, Canada

\textsuperscript{4}European XFEL GmbH — Schenefeld, Germany

\textsuperscript{5}University of Southampton — United Kingdom

6 September 2017
Usability

- Attribute of software sustainability[1]
- Overlooked aspect of scientific software[2]
- Unique requirements in computational science[3]
A) Think Beyond Graphical User Interfaces (GUIs)

- Command-Line Interfaces allow quick repetition of tasks and scriptability.
- Can be more productive than GUIs depending on the task.
- GUIs can be cumbersome on distributed infrastructures.
Good Practices

B) Keep UI Code Separate From Scientific Calculation

- Keeps software accessible through alternative interfaces, locally or remotely
- Makes software easier to reconfigure, customize[6] and integrate
- Rule applies to other types of software
C) Keep the Configuration in a File

- Storing parameters can be practical
- Good for reproducibility
- Good for accuracy
Good Practices

D) Design for Small, Incremental Changes

- Allows for extensibility
- Allows for emergent requirements
- Stays attuned to users needs and working habits

map3D. Pop-up menu
Good Practices

E) Facilitate and Register User Activity and Environment

- List of recently entered commands (time saving)[4]
- Quick access to frequent commands (ease of use)[7]
- Activity log (might support reproducibility)[8]
- Environment log (further support to reproducibility)
Good Practices
F) Learn About How Users Work

- Learn meanderings of scientific work[1] and its environment[5]
- Evaluate existing tools[9] and industry standards
- Start by addressing specific user bases[10, 11]
- Allow for user customization[11, 12]
- Engage in participatory design[13]/co-design[14, 15, 16]
- Turn users into developers[17]
G) Be Minimalistic, but Look Out for Exceptional Needs

- Make metadata readable and easy to access[18, 19, 20, 4, 15, 21]
- Mimic real-world counterparts of virtual instruments (for familiarity) when necessary[22]
- Use minimalism to emphasize critical information[14]
Good Practices

G) Be Minimalistic, but Look Out for Exceptional Needs

- Make metadata readable and easy to access[18, 19, 20, 4, 15, 21]
- Mimic real-world counterparts of virtual instruments (for familiarity) when necessary[22]
- Use minimalism to emphasize critical information[14]
Good Practices

G) Be Minimalistic, but Look Out for Exceptional Needs

- Make metadata readable and easy to access\cite{18, 19, 20, 4, 15, 21}
- Mimic real-world counterparts of virtual instruments (for familiarity) when necessary\cite{22}
- Use minimalism to emphasize critical information\cite{14}
H) Design for Precision

- Constrain or limit user input
- Give continuous feedback to user[20]
- Devise multiple input methods: one for speed, another for precision[23]
Good Practices

1) Contextualize User Actions

- Facilitate access to functions that are relevant for the task at hand (and prevent access to inadequate ones)[7]
- Separate tasks across individual screens[2]
Good Practices

1) Contextualize User Actions

- Facilitate access to functions that are relevant for the task at hand (and prevent access to inadequate ones)[7]
- Separate tasks across individual screens[2]
Summary

- Good practices address challenging aspects of scientific software development:
  - Lack of attention to software engineering
  - Need for reproducibility
  - Handling of large amounts of data
  - Complexity of actions and parameters involved
  - Frequent changes in requirements
  - Particularities of scientific work and its environment
  - Need for accessing and responding to critical information
  - Need for precision

- Adoption of presented practices should help delivering usable, robust and appropriate tools
This project began during the Fourth Annual Workshop on Sustainable Software for Science: Practices and Experiences (WSSSPE4), where four of the authors established the Working Group on Software best practices for undergraduates [24]. The authors would therefore like to thank the organizers of WSSSPE4 for facilitating this conversation. J. Miller and F. Queiroz would like to thank travel grants from the National Science Foundation of the USA and The Gordon and Betty Moore Foundation, which made attendance possible. J. Miller also acknowledges support from the Natural Sciences and Engineering Research Council of Canada and from the National Science Foundation of the USA (OCI 0905046, PHY 1212401). Research at Perimeter Institute is supported by the Government of Canada through the Department of Innovation, Science and Economic Development and by the Province of Ontario through the Ministry of Research and Innovation. F. Queiroz acknowledges support from PUC-Rio and Tecgraf Institute. H. Fangohr and R. Silva acknowledges support from the Software Sustainability Institute and Engineering and Physical Sciences Research Council (EPSRC) in the UK.


