Software: *Open* Questions

A.Matta

Journées Données 16-17th Dec 2024

LPC CAEN

cnrs ENSI CAEN UNICAEN UNIVERSITÉ CAEN NORMANDIE Normandie Université

Delivering software: my personal experience

Who am I?

- Science : nuclear structure and direct reaction since 2008
- nptool develloper since 2008
- CR at LPC Caen since 2016

Delivering software: my personal experience

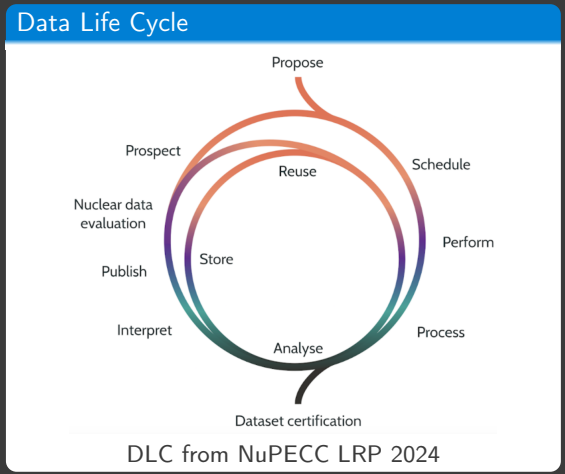
Who am I?

- Science : nuclear structure and direct reaction since 2008
- nptool develloper since 2008
- CR at LPC Caen since 2016

Reflexion on software practices

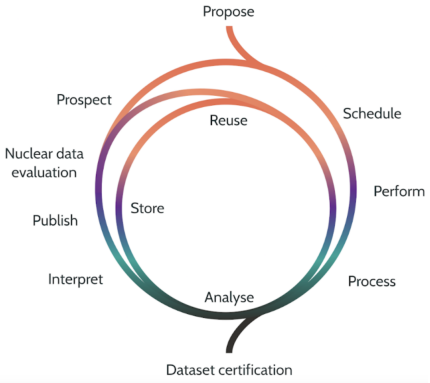
- GANIL ICC (2015)
- GDR RESANET R&D (2018)
- EUROLABS
- DOP2I
- NuPECC LRP 2024
- JENAA WG2 Software

Software devellopment: a central piece of the Data ecosystem



Software devellopment: a central piece of the Data ecosystem

Data Life Cycle



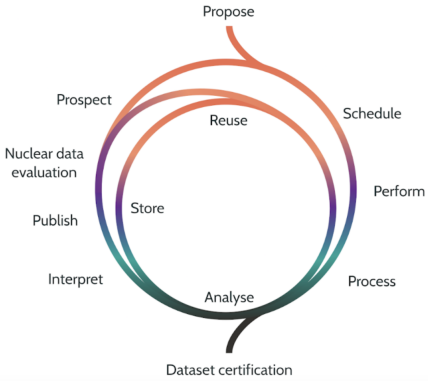
DLC from NuPECC LRP 2024

1st class citizen

- Software produce data
  - Sim, DAQ, log, ...

Software devellopment: a central piece of the Data ecosystem

Data Life Cycle



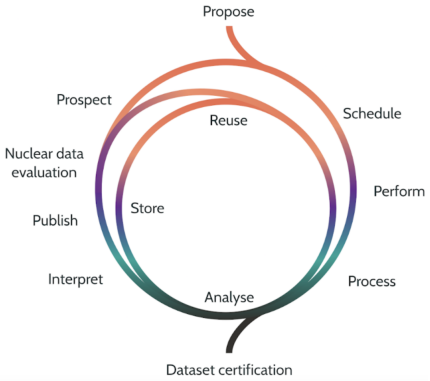
DLC from NuPECC LRP 2024

1st class citizen

- Software produce data
  - Sim, DAQ, log, ...
- Software transform data
  - Conversion, Analysis,...

Software devellopment: a central piece of the Data ecosystem

Data Life Cycle



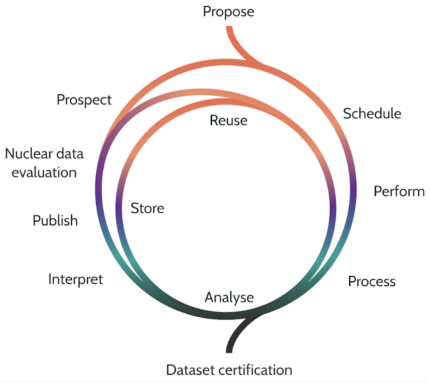
DLC from NuPECC LRP 2024

1st class citizen

- Software produce data
  - Sim, DAQ, log, ...
- Software transform data
  - Conversion, Analysis,...
- Software produce metadata
  - at least it should

Software devellopment: a central piece of the Data ecosystem

Data Life Cycle



DLC from NuPECC LRP 2024

1st class citizen

- Software produce data
  - Sim, DAQ, log, ...
- Software transform data
  - Conversion, Analysis,...
- Software produce metadata
  - at least it should
- Software *is* data
  - and therefore part of the dataset

Which Strategies?

- Better Software
- Long term vision
- Attract and retain talent

Software FAIRness

FAIR4RS article

scientific **data**

Check for updates

OPEN

ARTICLE

Introducing the FAIR Principles for research software

Michelle Barker[1] ✉, Neil P. Chue Hong[2], Daniel S. Katz[3], Anna-Lena Lamprecht[4], Carlos Martinez-Ortiz[5], Fotis Psomopoulos[6], Jennifer Harrow[7], Leyla Jael Castro[8], Morane Gruenpeter[9], Paula Andrea Martinez[10] & Tom Honeyman[11]

Software FAIRness

FAIR4RS article

F: Software, and its associated metadata, is easy for both humans and machines to find.
F1. Software is assigned a globally unique and persistent identifier.
F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.
F1.2. Different versions of the software are assigned distinct identifiers.
F2. Software is described with rich metadata.
F3. Metadata clearly and explicitly include the identifier of the software they describe.
F4. Metadata are FAIR, searchable and indexable.
A: Software, and its metadata, is retrievable via standardised protocols.
A1. Software is retrievable by its identifier using a standardised communications protocol.
A1.1. The protocol is open, free, and universally implementable.
A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
A2. Metadata are accessible, even when the software is no longer available.
I: Software interoperates with other software by exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.
I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
I2. Software includes qualified references to other objects.
R: Software is both usable (can be executed) and reusable (can be understood, modified, built upon, or incorporated into other software).
R1. Software is described with a plurality of accurate and relevant attributes.
R1.1. Software is given a clear and accessible license.
R1.2. Software is associated with detailed provenance.
R2. Software includes qualified references to other software.
R3. Software meets domain-relevant community standards.

Software Quality

Science driven

- Correctness → does what it is suppose to do
- Reliability → does it consistently
- Efficiency → does not wast resources

Operationally driven

- Maintainability → access to source code and necessary skill
- Portability → could run somewhere else
- Usability → *allowed* easy to run
- Reusability → *allowed* and easy to reuse in other context

How to publish your software

Self archiving

Where:

- Zenodo
- Software Heritage

Gain:

- Unique identifier
- Sub id for each version
- Distribution via archive

Drawback:

- Not included in metrics

How to publish your software

Self archiving

Where:

- Zenodo
- Software Heritage

Gain:

- Unique identifier
- Sub id for each version
- Distribution via archive

Drawback:

- Not included in metrics

Physics Journal

Where:

- Journal of Physics G

Gain:

- Well recognised
- High impact

Drawback:

- Ususally lot of work

→ Not for every version

How to publish your software

Self archiving

Where:

- Zenodo
- Software Heritage

Gain:

- Unique identifier
- Sub id for each version
- Distribution via archive

Drawback:

- Not included in metrics

Physics Journal

Where:

- Journal of Physics G

Gain:

- Well recognised
- High impact

Drawback:

- Ususally lot of work

→ Not for every version

Software Journal

Where:

- JOSS

Gain:

- Specific review process

→ easier to write

Drawback:

- Lacking Domain specialist

Carreer profile

RSE

- Research Software Engineer
- Devellopment powerhouse
- Technology expertise
- Architecture vision

Carreer profile

RSE

- Research Software Engineer
- Devellopment powerhouse
- Technology expertise
- Architecture vision

DSSE

- Domain Specific Software Expert
- Software focused physicist
- Relay to the community
- User experience vision

Carreer profile

RSE

- Research Software Engineer
- Devellopment powerhouse
- Technology expertise
- Architecture vision

DSSE

- Domain Specific Software Expert
- Software focused physicist
- Relay to the community
- User experience vision

How to make both path attractive?

Carreer profile

Profiles

How to make both path attractive?

Carreer profile

Profiles

How to make both path attractive?

To a variety of profile!

Conclusion

Open questions

- How to increase the community skill level?
- How to manage our technology debt?
- How to increase software quality?
- How to attract and retain talent?

Conclusion

Open questions

- How to increase the community skill level?
- How to manage our technology debt?
- How to increase software quality?
- How to attract and retain talent?

Strategies

- Guidelines (License, distribution, quality, ...) & Trainning
- Formal software collaboration
- Clear software strategy
- Well identied role in collaboration
- Metric to evaluate software related activities