DataOps

COMMUNITY DATA OPERATIONS FOR REPRODUCIBLE TLP

Thurston Sexton

Knowledge Extraction and Application for Smart Manufacturing Operations Management

Systems Integration Division

Engineering Laboratory

NIST
National Institute of
Standards and Technology
U.S. Department of Commerce

DISCLAIMER

The use of any products described in any presentation does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that products are necessarily the best available for the purpose.

OVERVIEW

I. **Our Domains**
   A. *Map Data and Domain "pipelines"*
   B. *Immediate needs*

II. **Our TLP Community**
   A. *The Problem*
   B. ***Lessons from the "front lines"***

APPLYING DATA-OPS IN OUR DOMAINS

Example from Maintenance Management

MWO DATA "PIPELINE"

- **E**xtract
- **T**ransform
- **L**oad

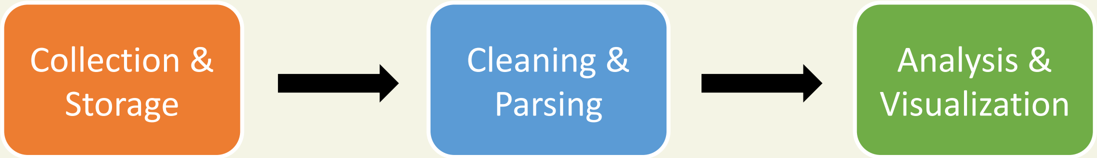
→

- Collection and Storage
- Cleaning and Parsing
- Analysis and Visualization

MWO DATA “PIPELINE”

- Extract
- Transform
- Load

- Collection and Storage
- Cleaning and Parsing
- Analysis and Visualization

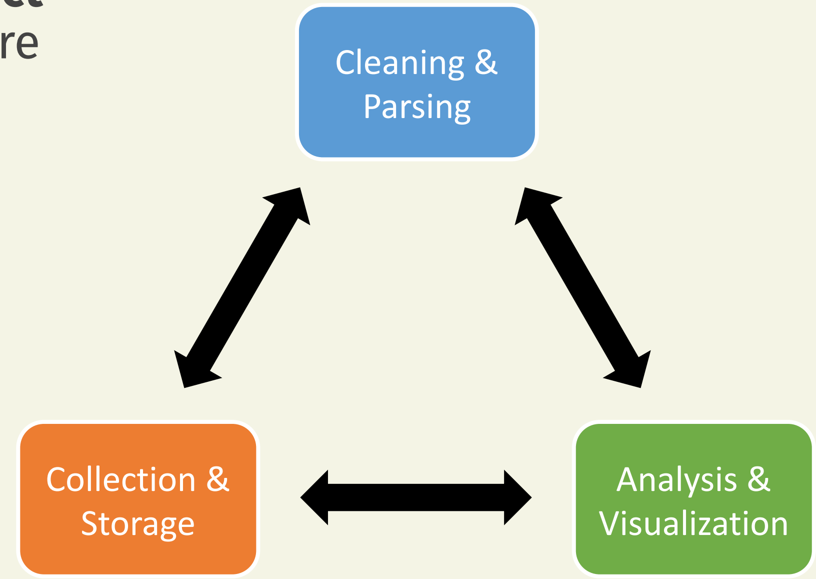


MWO DATA "PIPELINE"

Decisions made at each stage **will impact** the strategies that are

- Available
- Efficient

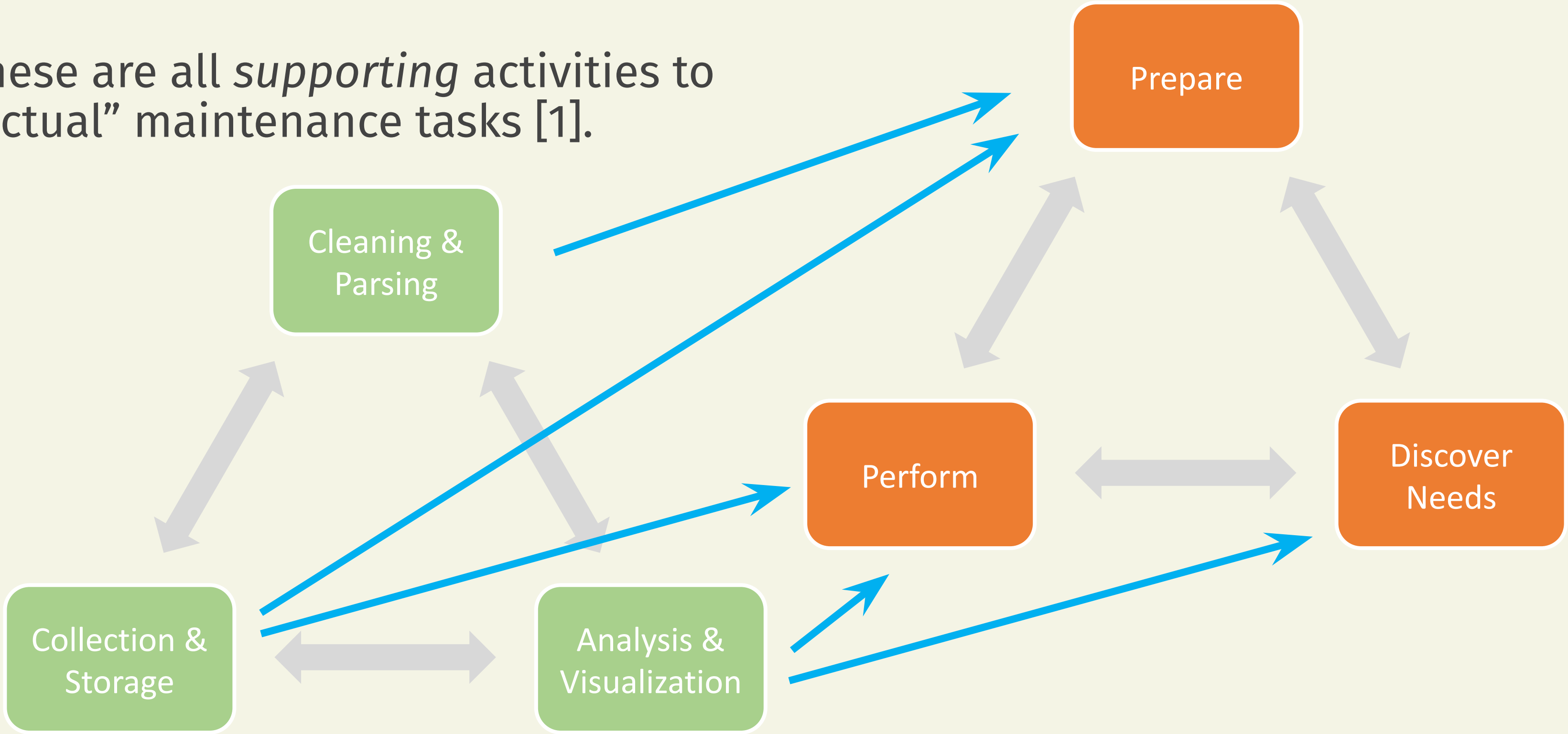
at each other stage.



Keep in mind …

MWO DATA "PIPELINE"

These are all *supporting* activities to "actual" maintenance tasks [1].



[1] Brundage, M. P., Sexton, T., Hodkiewicz, M., Morris, K., Arinez, J., Ameri, F., Ni, J., and Xiao, G. (July 22, 2019). "Where Do We Start? Guidance for Technology Implementation in Maintenance Management for Manufacturing." ASME. *J. Manuf. Sci. Eng*. September 2019; 141(9): 091005.

Needs - Data Collection and Storage

- MWO Terminology Definitions
  What defines its components? Who is involved? What is it recording?

- Atomic data types and formats for information flow in MWOs
  Issue meta-data (dates, descriptions, etc.), personnel, asset IDs

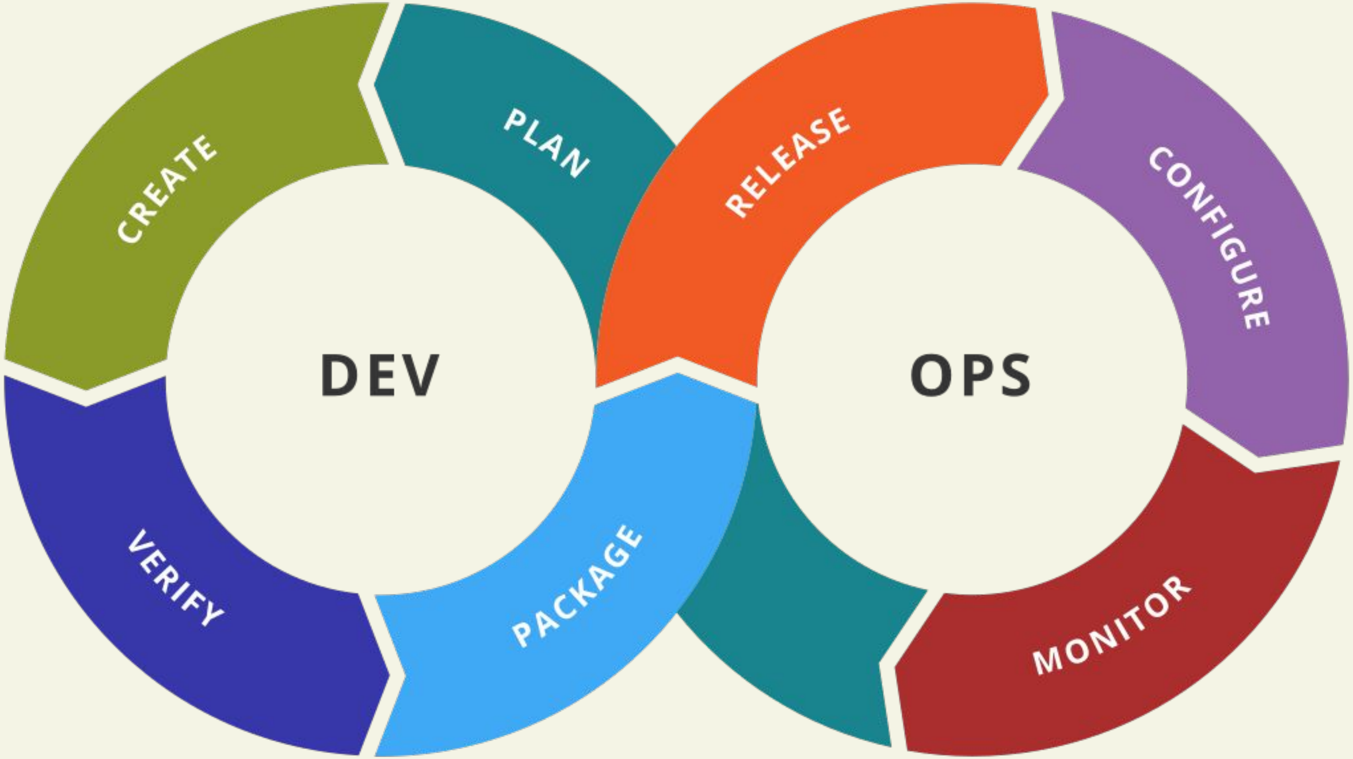
- Adaptive database schemas for storing varied MWO data
  Desirable information will shift over time—what are the core invariable relations?

- Mapping from disparate CMMS solutions into standard data types
  Current software uses proprietary/custom schemas—unification?

OUR COMMUNITY: The Problem

Developers vs “Hackers”

DEALING WITH COMPLEXITY



MOVIES VS REALITY

Programmers?

Researchers

See: *Science as Amateur Software Development*, R. McElreath 2020

So programmers use ***Dev-Ops…***
Science and Research is fueled by ***Data…***

→ ***Data-Ops***

WHAT IS DATA OPS?

*“**DataOps** (data operations) is an approach to **designing**, **implementing** and **maintaining** a distributed data architecture that will **support** a wide range of **open source tools** and frameworks in production.”* - Jack Vaughan

- Establish **progress** and **performance** measurements everywhere
- Abstract **validation** layer: Ensure everyone is
    a. “speaking the same **language**”
    b. **agrees** on what the data (and metadata) **is** and **is not**.

Toph Whitmore, Principal Analyst at Blue Hill Research

WHAT IS DATA OPS? cont'd...

*"**DataOps** (data operations) is an approach to **designing**, **implementing** and **maintaining** a distributed data architecture that will **support** a wide range of **open source tools** and frameworks in production."* - Jack Vaughan

- **Validate** with the "eyeball test":
    a. Include continuous-improvement-oriented **human feedback loops**.
    b. Trust in the data comes from **incremental** validation.
- **Automate** data flow…. As much as possible:
    a. preprocessing
    b. testing
    c. data science
    d. analytics

Toph Whitmore, Principal Analyst at Blue Hill Research

WHAT IS DATA OPS? cont'd...

*"**DataOps** (data operations) is an approach to **designing**, **implementing** and **maintaining** a distributed data architecture that will **support** a wide range of **open source tools** and frameworks in production."* - Jack Vaughan

- Identify **bottlenecks**, then **optimize** for them.
  a. Use performance measurements here!
  b. Investment: hardware, automation, etc.

- Governance discipline
  a. data ownership & **transparency**,
  b. data lineage tracking

- Design for growth and **extensibility**
  a. Must accommodate volume and variety of data.
  b. Enabling technologies should be priced affordably

Toph Whitmore, Principal Analyst at Blue Hill Research

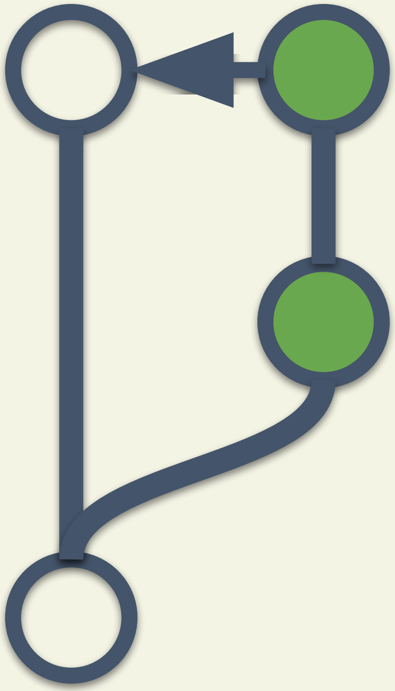
LESSONS WE’VE LEARNED

From the “front lines”

EXPLOIT GIT(FORGE) ARCHITECTURE

Pull(Merge) Requests

- Projects as **iterative** collaborations
  - Start exploration as a branch
  - Can be “empty”
  - Track small commits w/ **conversation**
  - Integrated review, suggestions, @’s
  - Inline change views/comments
- Prototype, test, complete, review, merge
  - All without breaking “main”
  - Can apply to all steps in the pipeline



References:

- [Ten Simple Rules for Taking Advantage of Git and GitHub]()
- [Ask students to iterate on their work with draft pull requests]()

EXPLOIT GIT(FORGE) ARCHITECTURE

Data Science Environments

- Reproducible Compute (e.g. Python?)
  - Jupyter Notebooks + git??? → **Jupytext**
  - *Lightweight* environments? → **miniconda**
  - Simple Packages (w/o setuptools) → **poetry**
- Documentation and Interop.
  - Easier documentation → **mkdocs-material**
  - Use automated docstring extraction
  - Data-oriented programming
  - Unify styles: Type-hinting, functions-first.
  - property-based tests → **Hypothesis**

Also see:

- Tom Augspurger, *Modern Pandas*

EXPLOIT GIT(FORGE) ARCHITECTURE

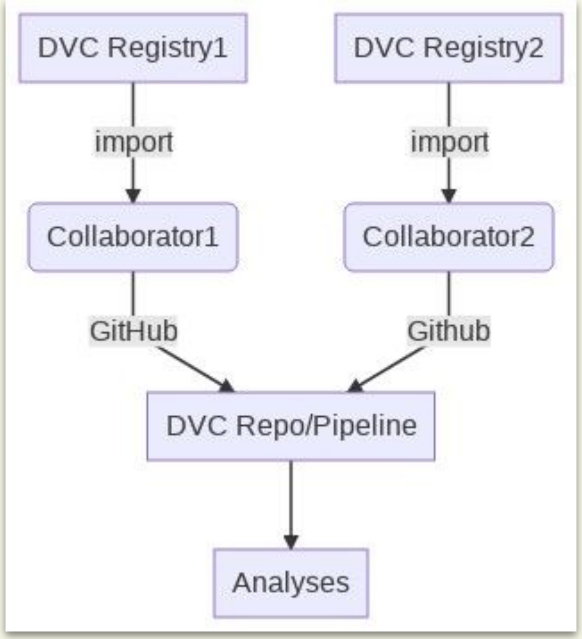
Data *Itself*

- Data-as-Code: makefiles+git=DVC
  - Don't reinvent the wheel, **use git.**
  - Language-agnostic, w/ python API
  - Every step of the **pipeline**, version-controlled with automated cache-updates
  - Make **registries** for your entire community (!) (data is just an "import" away...)

- Validate *all the things*
  - Data shape, types, etc., make *explicit*: **datatest**
  - Schemas once-and-for-all: → **pydantic**

References:

- Ten Simple Rules for Taking Advantage of Git and GitHub
- Ask students to iterate on their work with draft pull requests



THE PLAN

Distributed Collaboration for the TLP CoI

I. GitHub Organization: **TLP-COI**

   A. Documentation - best practices for TLP, theory, etc
   B. Networking - curated list for state-of-the-practice (“awesome-tlp”)
   C. Collaboration - base or forks for open tool repositories

II. Communication:

   A. TLP-COI Slack Workspace - QR code →
   B. Other options? Possible “Discourse”? Webinars? Let us know!

Copy and paste icon to desired slide. To change color, double click on icon, select color from drop down. For consistency, please use colors in the template. *Due to licensing restrictions, you can only use these icons for NIST PowerPoints.*

NEWS
in
You
Tube
NIST