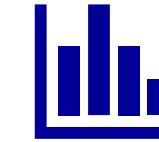


Scientific Computing

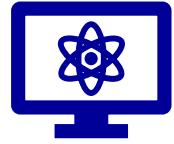
Research Software Engineering Collaborative Open-source Ecosystems in Practice: Research and Business Impact in the age of AI disruption + Python ecosystem (mainly) + Julia ecosystem

Speaker: Dr. **Alexey Neznanov**, PhD
Senior Data Scientist, Associate Professor

2025-09-26 (v.06.02)



Content



Research software engineering

- ✓ Introduction to RSE
- ✓ Trends
- ✓ Communities and societies
- ✓ Main sources
- ✓ RSE in corporations
- ✓ Tools
- ✓ Scientific programming languages and RSE

© 2025, Alexey Neznanov

4

Reproducibility and computational science

- ✓ Reproducibility and beyond
 - ✓ 3R, 4R, 5R, 7R ... How much more?
 - ✓ Code snippets and datasets
- ✓ ACM Artifact Review
- ✓ Some repositories of reproducible research
- ✓ Research software evaluation
- ✓ Soft and data citation

© 2025, Alexey Neznanov

28

Programming-in-the-small/large/many/huge/etc.

- ✓ Programming-in-the-Small/Large
- ✓ Programming-in-the-Many
- ✓ Programming-in-the-Huge
- ✓ Evidence-based software engineering

© 2025, Alexey Neznanov

44



Technologies - platforms

- ✓ Platform oriented approach
- ✓ Infrastructure for RSE
- ✓ Reproducible computational experiments at scale
 - ✓ Orchestration
- ✓ Security

© 2025, Alexey Neznanov

51



Collaborative Technologies

- ✓ Collaboration
- ✓ Collaborative spaces
- ✓ RSE and collaboration
 - ✓ BoKs
 - ✓ Literature programming
 - ✓ Interactive books

© 2025, Alexey Neznanov

61



Interactive reactive notebooks

- ✓ Classical notebooks VS reactive notebooks
- ✓ Types of reactivity
- ✓ Editing and executing
- ✓ Deployment
- ✓ Transform into Web-App
- ✓ The reactive principals in scientific computing for SR

© 2025, Alexey Neznanov

77



Python monolanguage stack architecture

- ✓ DDD
- ✓ Approaches
- ✓ Layers
- ✓ Python ecosystem – foundation
- ✓ [TBD] IS and personal data

© 2025, Alexey Neznanov

100



Data & knowledge modelling

- ✓ Data and their life cycle
- ✓ Knowledge bases (only static examples)
- ✓ DBs
 - ✓ Relational
 - ✓ Non-relational
 - ✓ Semantic

© 2025, Alexey Neznanov



Data storages and ETL

- ✓ External data models
 - ✓ Graphs
 - ✓ Trees
 - ✓ Time series
 - ✓ Grids and efficient
 - ✓ ETL

© 2025, Alexey Neznanov



Dataframe as a heart of common data transformation

- ✓ New generation of Python dataframe libraries
 - ✓ Pandas
 - ✓ Dataframe for Python (DFP)
 - ✓ Dataframe for Spark
 - ✓ Apache Beam
 - ✓ Simple vector DFV

© 2025, Alexey Neznanov



DataViz in Python

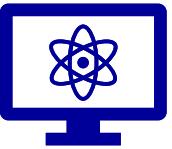
- ✓ DataViz
 - ✓ Matplotlib
 - ✓ Pandas
 - ✓ Seaborn
 - ✓ Plotly
 - ✓ Bokeh

© 2025, Alexey Neznanov

About speaker

- Alexey Neznanov
 - PhD
 - IEEE member
 - Senior Data Scientist at Schlumberger Russia
 - Associate professor at HSE (<http://www.hse.ru/staff/aneznanov>)
 - «Small Guide to Big Data» consultant at PostScience portal (http://postnauka.ru/author/a_neznanov)
 - Author of textbook, instructional guidelines, 11 courses and more than 65 research papers





Research software engineering

- ✓ Introduction to RSE
- ✓ Trends
- ✓ Communities and societies
- ✓ Main sources
- ✓ RSE in corporations
- ✓ Tools
- ✓ Scientific programming languages and RSE

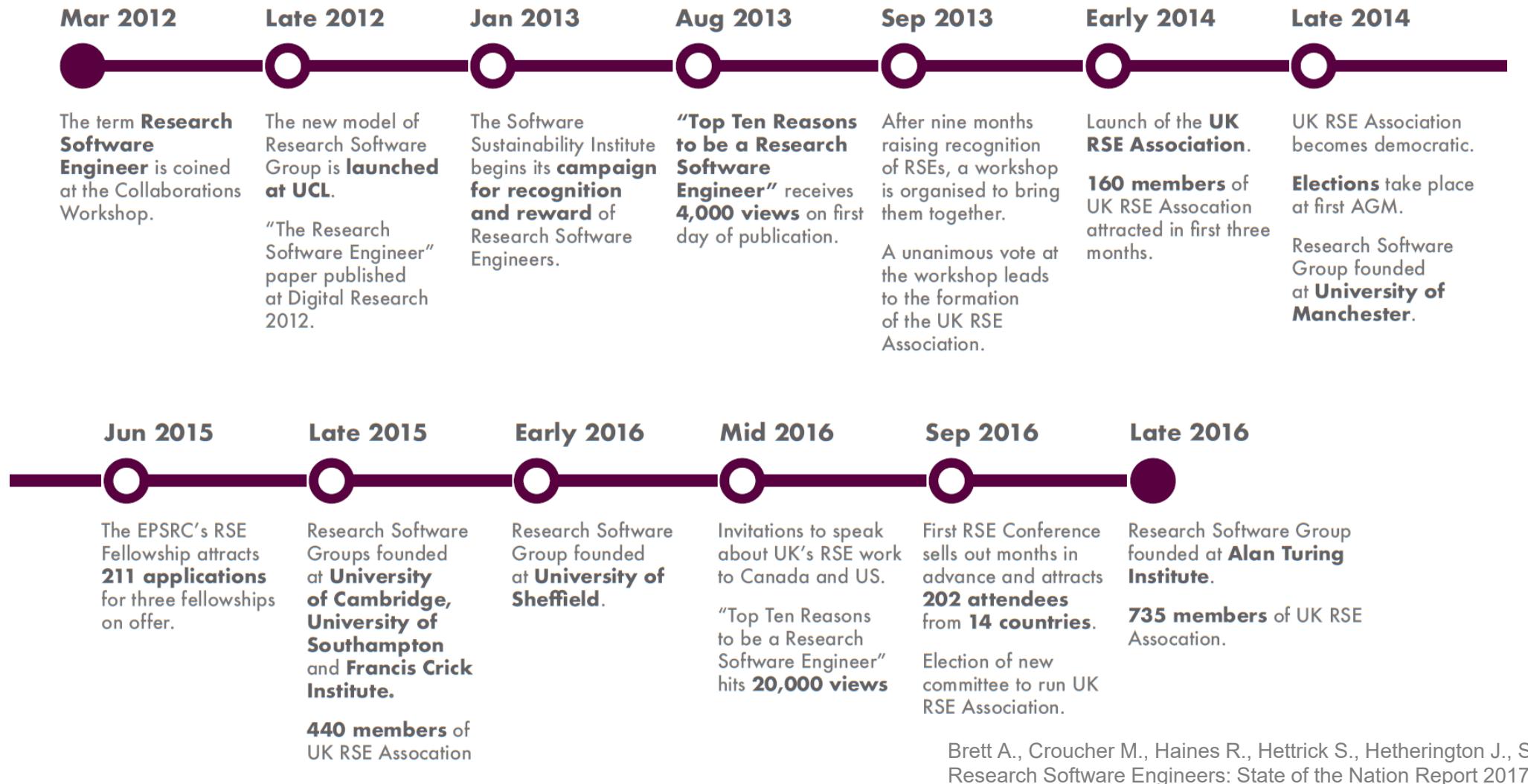
Approaches to sustainable research

- Scientific epistemology and ontology
 - Research programs and evidence-based discovery
- System engineering and software engineering
- Computational experiments and laboratory experiments automation
 - Mathematical models, grounding, and simulation
- Programming-in-the-Small/Large/Many/Huge/...
 - Literature programming → Collaborative interactive research environments
- Data Science
 - Data and knowledge management: FAIR4RS and other frameworks
 - Dataset management and data citation
 - Computational knowledge
- DevOps/DataOps/AIOps/MLOps/SimOps/...
 - Configuration management and artifacts versioning (not only code!)

RSE definition & modern history (1)

- Most comprehensive paper! But the best definition will be cited later...
 - Brett A., Croucher M., Haines R., Hettrick S., Hetherington J., Stillwell M., Wyatt C. **Research Software Engineers: State of the Nation Report 2017**, 2017.
(<http://doi.org/10.5281/zenodo.495360>)
 - Most research would be impossible without software, and this reliance is forcing a rethink of the skills needed in a traditional research group
 - With the emergence of software as the pre-eminent research tool used across all disciplines, comes the realisation that a significant majority of results are based, ultimately, on the skill of the experts who design and build that software
 - The UK has led the world in supporting a new role in academia: the Research Software Engineer (RSE)
 - This report describes the new expert community that has flourished in UK research, details the successes that have been achieved, and the barriers that prevent further progress

Modern RSE timeline (2012 - 2017)



RSE definition & modern history (2)

- **Research Software Engineering** is the use of Software Engineering practices, methods and techniques for research software, i.e. software that was made for and is mainly used within research projects
 - Bernhard Rumpe. Research Software Engineering White Paper. 2024 (<https://www.se-rwth.de/essay/Research-Software-Engineering/>)
- Main content:
 - Since 2010 communities have been created, conferences have been organized, and people exchange their findings on how to overcome the challenges of research software development, i.e., how to overcome the research software crisis
 - Specifics in: Reuse, Modularity, Modelling, Automation with Smart Tools, Versioning, Long-lasting software support, Reproducibility, Framework Development Strategies, Testing, Development Methods, Consulting, ...
 - Research topics, applications, and recommendations

RSE – There is no silver bullet!

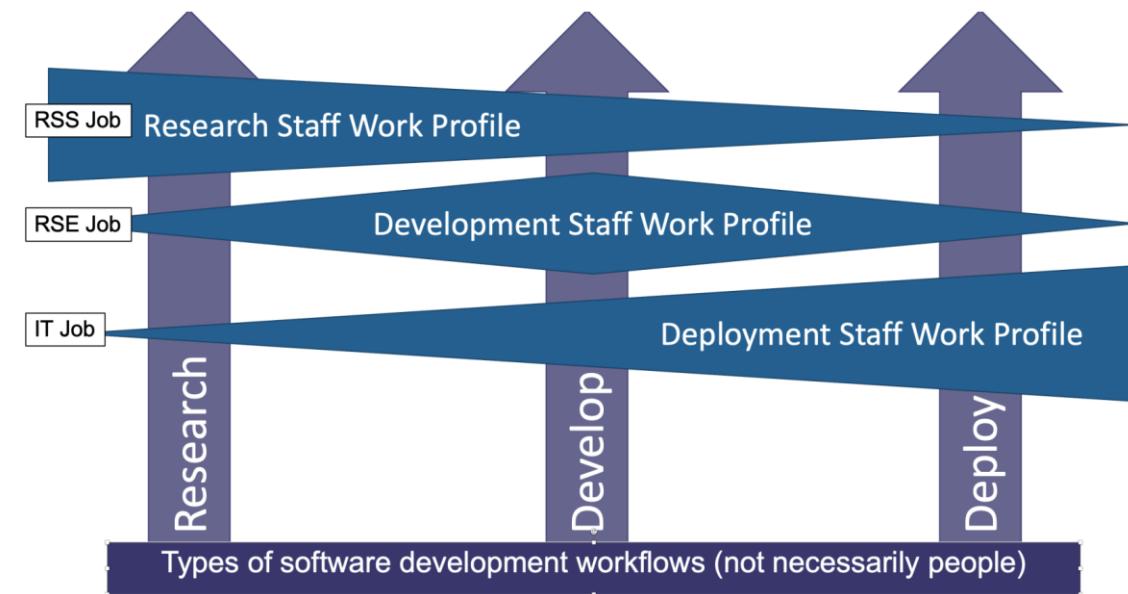
- Recommendations:

1. It is necessary to build better domain-specific tooling to address the domain-specific challenges of research software. Wizard-like smart tools help development amateurs and, to some extent, prevent them from having to focus too much on SE skills themselves.
2. Researchers who create software for a sustainable, long-lasting infrastructure need to be trained in software engineering skills, which drastically differ from mere programming skills.
3. Have one or more software engineers be part of your project to get the software and the technical architecture right, adopt the appropriate tools and quality mechanisms, etc.
4. Start teaching RSE skills to researchers that go beyond mere coding capabilities.
5. Incentives to use RSE best practices must be developed for all stakeholders involved.
6. Establishing RSE principles is a core topic especially for the management, i.e. the professors and the research institutions, that needs to be addressed adequately.
7. Establish a Software Development Guideline for the projects or even for the whole institution, that cover (a) organizational issues, (b) legal issues e.g. with copyrights, and especially (c) a methodological process framework.
8. Establish research software engineering research as a research field over RSE.

RSE – Expanding the impact

- Heroux M.A. **Research software science: Expanding the impact of research software engineering.** Computing in Science & Engineering 24(6), 2022. pp. 22-27. (<http://doi.org/10.1109/MCSE.2023.3260475>)
 - **Research software science (RSS)** – applying the scientific method to understanding and improving how software is developed and used for research

| | | |
|--|---|--|
|  | Applying the Scientific Method to | Use formal observation and experimentation to obtain and disseminate knowledge via repeatable and reproducible processes |
|  | Understanding and Improving | Obtain data to detect correlation; design experiments to identify cause and effect; publish results for broad impact |
|  | How Software is Developed and Used | Combined focus on developers and users; impact individuals, teams, and communities |
|  | For Research | Focus on software used in the pursuit of scientific knowledge and insight |



RSE in 2030

- Katz D.S., Hettrick S. **Research Software Engineering in 2030 // 2023 IEEE 19th International Conference on e-Science, 2023, pp. 1-2.** (<http://arxiv.org/abs/2308.07796>)
 - Software development:
 - Research software developers (including but not limited to RSEs) will be generally aware of research software engineering as a concept and its best practices, at about the same level that software developers today are aware of source control packages like git
 - This will be due to a greater awareness of the role of software in research and changes to community, training, and policy
 - More research software will be sustained because funders will better understand that funding the maintenance of research software will improve the return on their initial investment into it.
 - Community:
 - At least 50 countries globally will have **RSE associations**
 - Most professional scholarly societies will have **research software tracks or themes** in their conferences
 - Training and education:
 - At least one university **Masters of Research Software Engineering (MRSE)** program will exist, and many universities will offer RSE certificates or minors
 - In the US, 75% of R1 universities will have **RSE organizations**
 - In the UK, all research-led universities will host an **RSE Group**
 - Most US universities will have **RSE career paths** that contain multiple levels, allow for both technical and management career progression over 30+ years, and allow RSEs to be PIs
 - ... and other positional statements

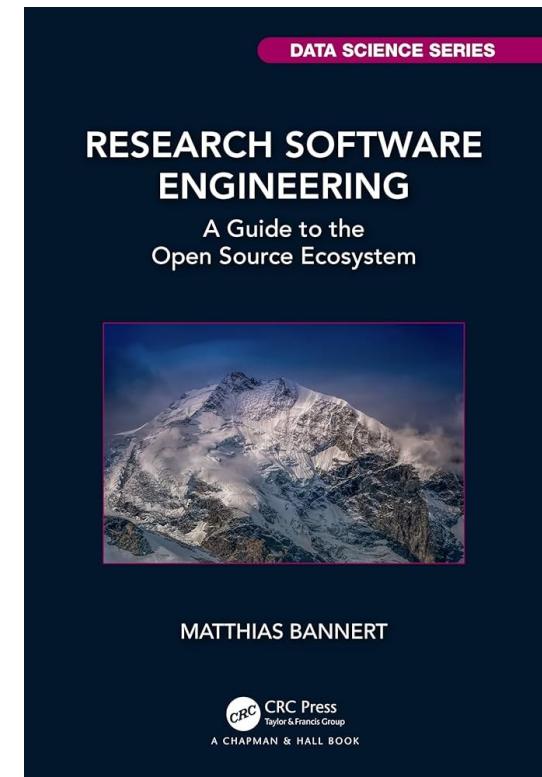
RSE in the world

- Research Software Engineering (RSE): highly developed methodology
 - The Research Software Alliance (ReSA)
(<http://www.researchsoft.org>)
 - The United States Research Software Engineer Association
(<http://us-rse.org>)
 - Society of Research Software Engineering
(<http://society-rse.org>)
 - + RSECon



RSE book

- Research Software Engineering.
A Guide to the Open Source Ecosystem
(<http://rse-book.github.io>)
 - Easy introduction
 - Open source
 - The accent on modern cloud environment
 - Superficial presentation...



And what about openness?

- Licenses for code:
 - OSI Approved Licenses (<http://opensource.org/licenses>)
 - Choose an open source license (<http://choosealicense.com>)
 - Where to host open code?
 - Comparison of source-code-hosting facilities (http://en.wikipedia.org/wiki/Comparison_of_source-code-hosting_facilities)
- The best license series for other artifacts:
 - Creative Commons: When we share, everyone wins (<http://creativecommons.org>)



Role in team

- Research Software Engineer: Overview
(<http://book.the-turing-way.org/collaboration/research-infrastructure-roles/rse.html>)
 - Research Software Engineers (RSEs) are programmers with scientific backgrounds who play increasingly critical roles in the conduct of research and production of research software tools
 - They are **highly skilled team members** who can also conduct their own research as part of their role
- Examples:
 - Princeton Research Computing – Selected Research Software Engineering Projects
(<http://researchcomputing.princeton.edu/services/research-software-engineering/rse-projects>)



RSE and data management

- Most influence declarations:
 - FORCE11 – The Future of Research Communications and e-Scholarship  **FORCE11**
 - FAIR 4 Research Software (FAIR4RS) Working Group at FORCE11
(<http://force11.org/groups/fair-4-research-software-fair4rs-working-group/>)
 - The **FAIR Data Principles** (<http://force11.org/info/the-fair-data-principles/>)
 - Data should be **Findable, Accessible, Interoperable, Re-usable**
 - Data Citation Synthesis Group: Joint Declaration of **Data Citation Principles**.
Martone M. (ed.) San Diego CA: FORCE11; 2014 (<http://doi.org/10.25490/a97f-egyk>)
 - Principles: **Importance, Credit and Attribution, Evidence, Unique Identification, Access, Persistence, Specificity and Verifiability, Interoperability and Flexibility**

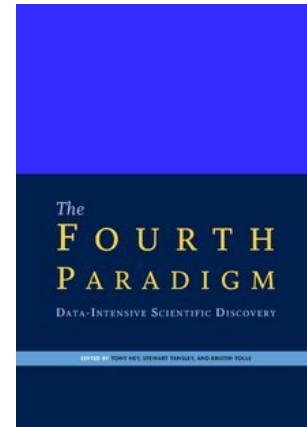
FAIR Implementation Profile

| FAIR principle | Question | FAIR enabling resource types |
|----------------------|---|--|
| F1 | What globally unique, persistent, resolvable identifier service do you use for metadata records? | Identifier service |
| F1 | What globally unique, persistent, resolvable identifier service do you use for datasets? | Identifier service |
| F2 | Which metadata schemas do you use for findability? | Metadata schema |
| F3 | What is the schema that links the persistent identifiers of your data to the metadata description? | Metadata-Data linking schema |
| F4 | In which registry are your metadata records indexed? | Registry |
| F4 | In which registry are your datasets indexed? | Registry |
| A1.1 | Which standardized communication protocol do you use for metadata records? | Communication protocol |
| A1.1 | Which standardized communication protocol do you use for datasets? | Communication protocol |
| A1.2 | Which authentication & authorization service do you use for metadata records? | Authentication & authorisation service |
| A1.2 | Which authentication & authorization service do you use for datasets? | Authentication & authorisation service |
| A2 | Which metadata preservation policy do you use? | Metadata preservation policy |
| I1 | Which knowledge representation languages (allowing machine interoperation) do you use for metadata records? | Knowledge representation language |
| I1 | Which knowledge representation languages (allowing machine interoperation) do you use for datasets? | Knowledge representation language |
| I2 | Which structured vocabularies do you use to annotate your metadata records? | Structured vocabulary |
| I2 | Which structured vocabularies do you use to encode your datasets? | Structured vocabulary |
| I3 | Which models, schema(s) do you use for your metadata records? | Semantic model |
| I3 | Which models, schema(s) do you use for your datasets? | Semantic model |
| R1.1 | Which usage license do you use for your metadata records? | Data usage license |
| R1.1 | Which usage license do you use for your datasets? | Data usage license |
| R1.2 | Which metadata schemas do you use for describing the provenance of your metadata records? | Provenance model |
| R1.2 | Which metadata schemas do you use for describing the provenance of your datasets? | Provenance model |
| R1.3 | Who is the community, and what are their domain-relevant community standards? | FAIR Implementation Profile |

The other view (Data Science?)

- The **Fourth Paradigm**: Data-intensive Scientific Discovery.
Edited by Tony Hey, Kristin Michele Tolle, Stewart Tansley.
Microsoft Research, 2009.

- The paradigms:



- Good comment (by Srikanth Guruswamy, 2016
<http://serc.carleton.edu/earthandmind/posts/4thparadigm.html>):
 - 4 Pillars of DISD (4th Paradigm Science) are:
 1. Data base and Data Life Cycle Management – universally by all data creators and contributors
 2. Scientific Workflow like Taverna, MyExperiment or Microsoft Azure – used across by scientists for consistent, correct and competent methodology (this ensures repeatability)
 3. Analysis methods like R, SciPy – where a continuous set of improved analytics and visualization is provided for entire community
 4. Presentation and Exchange – like arXiv or pubmed – where most if not all scientific results are publicly accessible
 - GeoSciences especially **Petroleum GeoScience** (well funded) is far from following these 4 pillars!
 - It is not about when we started in the right path of science – it is about how we are staying in the right path now
 - It is not about what is accomplished with data – it is about how it is accomplished that determines 4th Paradigm Science

RSE in corporation (1)

- Scientific computing from scientific point of view:
 - Woolston, Chris. Why science needs more research software engineer. Nature Career Q&A, 2022 (<http://www.nature.com/articles/d41586-022-01516-2>)

- RSE is costly (direct and indirect spendings!)

- Infrastructure + Organization + People
 - vs
- Software products functionality and quality
- Openness! Research groups collaboration
 - See next section...



- Investments and Outcomes:

- Deschamps, Joran; Dalle Nogare, Damian; Jug, Florian. Better Research Software Tools to Elevate the Rate of Scientific Discovery – or why we need to invest in research software engineering. arXiv:2307.03934, 2023.
- Sochat, Vanessa. So You Want to Start a Research Software Engineering Group?, 2022 (<http://vsoch.github.io/2022/rseng-group>)

RSE in corporation (2)

RiSE

- Example of Microsoft:
 - **Research in Software Engineering (RiSE)**
(<http://www.microsoft.com/en-us/research/group/research-software-engineering-rise/>)
 - The RiSE group builds the system and foundations of programming through open-ended fundamental research in formal methods, high-performance computing, programming languages, and software engineering
 - We collectively work towards a future where computing systems are provably **correct**, **secure**, and **performant**
 - Our research synergistically combines symbolic reasoning, AI, and human-centric methods to empower every programmer and organization to achieve more



Technical guidelines (in Python ecosystem)

- Popular guidelines:

- D. Irving, K. Hertweck, L. Johnston, J. Ostblom, C. Wickham, and G. Wilson.
Research Software Engineering with Python. Building software that makes research possible.
Chapman and Hall/CRC, 2021. 528 p.
 - + Repo (<http://github.com/merely-useful/py-rse>)
- The Good Research Code Handbook (<http://goodresearch.dev>)
- The Alan Turing Institute – Research Software Engineering with Python
(<http://alan-turing-institute.github.io/rse-course/html/index.html>)

- Language selection?

- **Python** (<http://www.python.org>) V 3.12+
 - Most popular scripting, glue and integration language:
 - High-level
 - Universal
 - Productive
 - With big community and ecosystem



Other languages in modern open ecosystem

- Main candidates:

- **Rust** (<http://www.rust-lang.org>) V 1.87.0

- System programming and optimization
 - Repo (<http://github.com/rust-lang/rust>)



- **R** (<http://www.r-project.org>) V 4.5.0

- Statistics and tabular data analysis



- **Julia** (<http://julialang.org>) V 1.11.5

- Effective scientific programming-in-the-small and programming-in-the-large



Reference languages for scientists

- **Wolfram language**
(<http://www.wolfram.com/language/>)



- **MathWorks Matlab**
(<http://www.mathworks.com/help/matlab/language-fundamentals.html>)



- **Racket** (<http://racket-lang.org>)
 - Functional, descendant of Lisp and Scheme



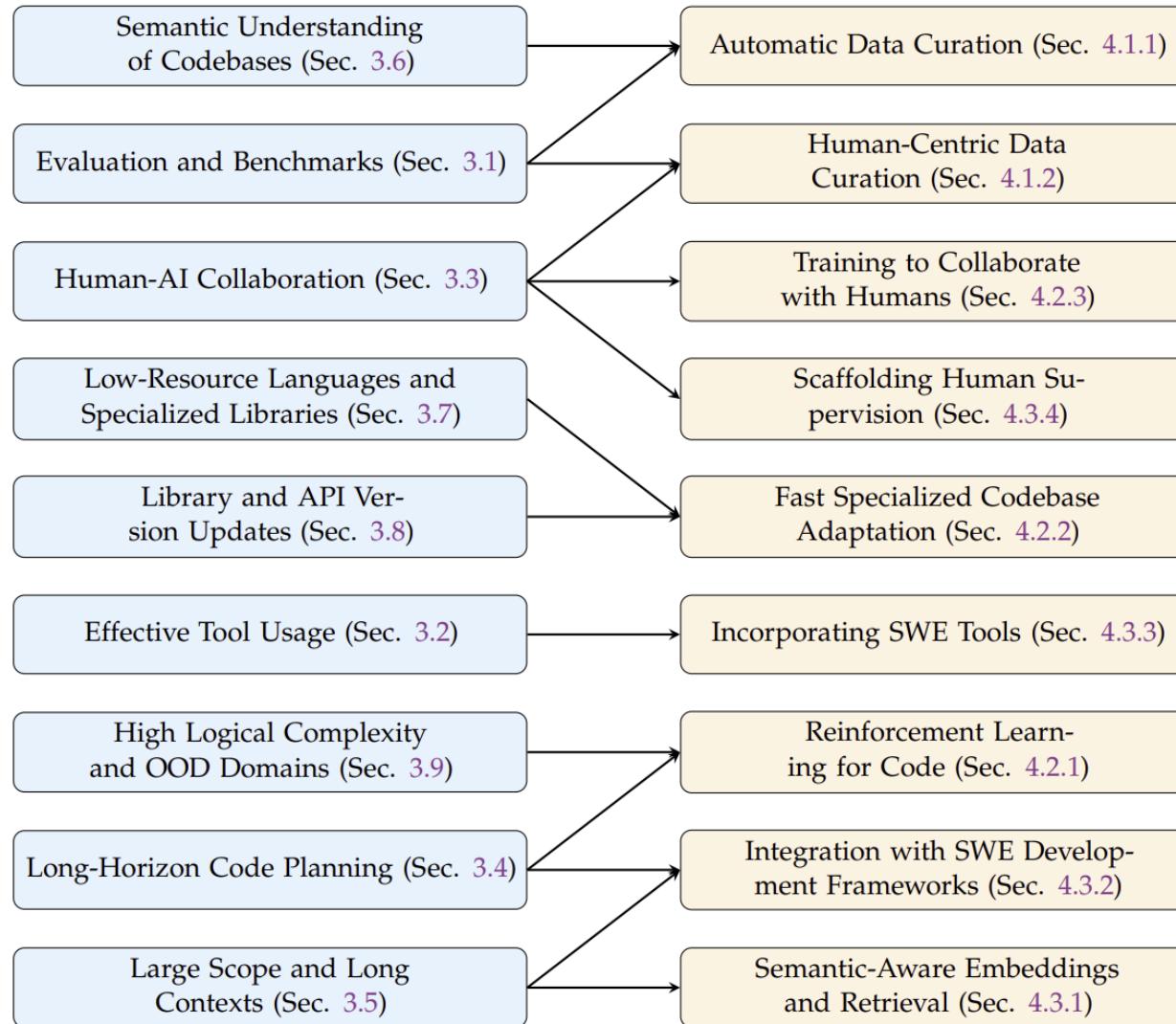
And more languages

- What is a “good” language?
 - What are the best (**productivity-enhancing, well-designed, and concise**, rather than just popular or time-tested) programming languages?
(<http://www.slant.co/topics/5984/~productivity-enhancing-well-designed-and-concise-rather-than-just-popular-or-time-tested-programming-lang>)
 - 94 Options Considered!
 - 3.2K User Recs.

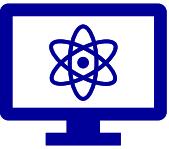
| BEST (PRODUCTIVITY-ENHANCING, WELL-DESIG... | PRICE | CURRENT STABLE VERSION | PLATFORMS |
|--|-------|------------------------|---------------------------------------|
| 62  Nim | - | 2.0.0 | Windows, MacOS/iOS, Linux /Android... |
| 53  Smalltalk | - | 6.1 | - |
| 53  Elixir | - | 1.11.2 | - |
| 51  Common Lisp | - | - | - |
| 51  Rust | - | 1.71.1 | Linux, MacOS X, Windows, BSD |



Challenges and Paths Forward in AI for SE



Gu A., Jain N., Li W.-D., Shetty M., Shao Y., Li Z., Yang D., Ellis K., Sen K., Solar-Lezama A. Challenges and Paths Towards AI for Software Engineering. arXiv:2503.22625, 2025. (<http://arxiv.org/abs/2503.22625>)



Reproducibility and computational science

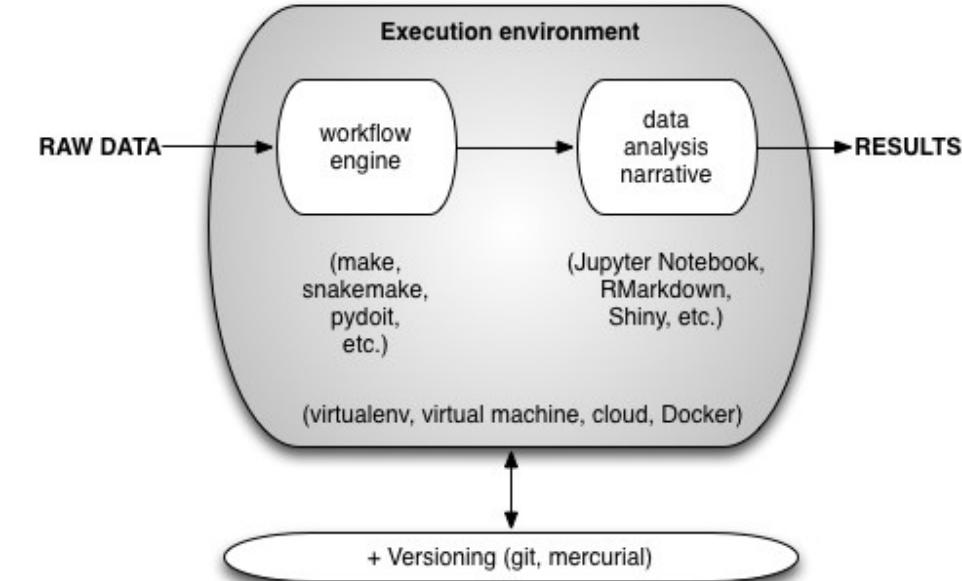
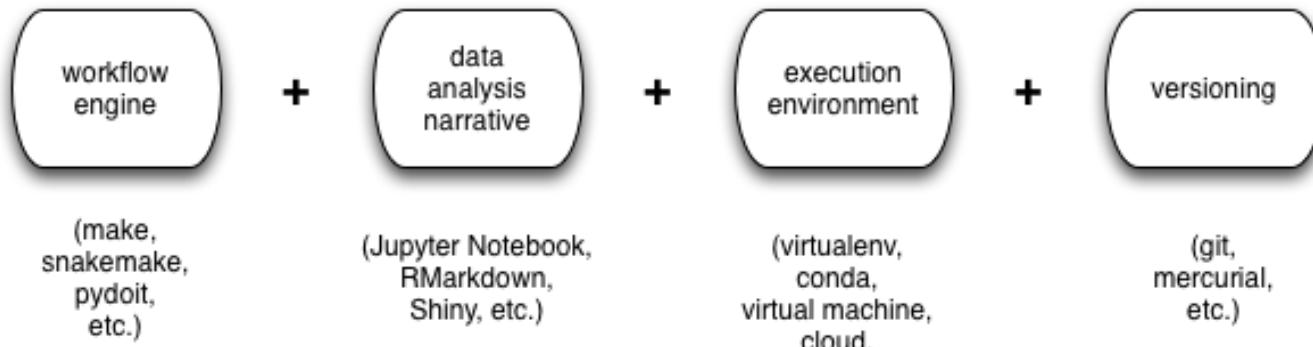
- ✓ Reproducibility and beyond
 - ✓ 3R, 4R, 5R, 7R ... How much more?
 - ✓ Code snippets and datasets
- ✓ ACM Artifact Review
- ✓ Some repositories of reproducible research
- ✓ Research software evaluation
- ✓ Soft and data citation

Algorithms (code) and data as scientific outputs

- Main principals:
 - Interpretability/Explainability
 - Interpretation as a process taking one explanation to another, more understandable explanation
 - From reproducibility up to **5R** or **7R**
 - Re-runnable, Repeatability, Reproducibility, Replicability, Reusability
 - Reusable, Repurposeable, Repeatable, Reproducible, Replayable, Referenceable, Respectful
 - Reenterability
 - RACE
 - Record everything, Automate everything, Contain everything, Expose everything
 - Identifiability
 - Logging and auditing
 - Benchmarking
 - ...

2016 #uioCarpentry Oslo workshop on Reproducibility

- The workshop materials for the experimental Data Carpentry/Reproducibility workshop (<http://2016-oslo-repeatability.readthedocs.io>)
 - Make, git, & jupyter notebook + docker and automation:



5R-in-the-small (1)

- Benureau, Fabien C.Y.; Rougier, Nicolas P. **Re-run, Repeat, Reproduce, Reuse, Replicate**: Transforming Code into Scientific Contributions. *Frontiers in Neuroinformatics*, 11, 2018. (<http://www.frontiersin.org/journals/neuroinformatics/articles/10.3389/fninf.2017.00069/full>)
 - What about a **program** as **code snippet**?
 - “A program can **fail** as a **scientific contribution** in many different **ways** for many different **reasons**”
 - Levels: re-runnable (R¹) → repeatable (R²) → reproducible (R³) → reusable (R⁴) → replicable (R⁵)
 - Example on the next slide
 - “There is the belief that an article should suffice by itself: the descriptions of the algorithms present in the paper should suffice to reobtain (to replicate) the published results”

5R-in-the-small (1)

raw code, archive

LISTING 0: Random walk (R⁰)

```
import random

x = 0
for i in xrange(10):
    step = random.choice([-1,+1])
    x += step
print x,
```

LISTING 1: Re-runnable random walk (R¹)

```
# Tested with Python 3
import random

x = 0
walk = []
for i in range(10):
    step = random.choice([-1,+1])
    x += step
    walk.append(x)

print(walk)
# Saving output to disk
with open('results-R2.txt', 'w') as fd:
    fd.write(str(walk))
```

LISTING 2: Re-runnable, repeatable random walk (R²)

```
# Tested with Python 3
import random

random.seed(1) # RNG initialization ✓

x = 0
walk = []
for i in range(10):
    step = random.choice([-1,+1])
    x += step
    walk.append(x)

print(walk)
# Saving output to disk
with open('results-R2.txt', 'w') as fd: ✓
    fd.write(str(walk))
```

LISTING 3: Re-runnable, repeatable, reproducible random walk (R³)

```
# Copyright (c) 2017 N.P. Rougier and F.C.Y. Benureau
# Release under the BSD 2-clause license
# Tested with 64-bit CPython 3.6.2 / macOS 10.12.6
import sys, subprocess, datetime, random

def compute_walk():
    x = 0
    walk = []
    for i in range(10):
        if random.uniform(-1, +1) > 0: ✓
            x += 1
        else:
            x -= 1
        walk.append(x)
    return walk

# If repository is dirty, don't run anything
if subprocess.call(("git", "diff-index",
                    "--quiet", "HEAD")):
    print("Repository is dirty, please commit first")
    sys.exit(1)

# Get git hash if any
hash_cmd = ("git", "rev-parse", "HEAD")
revision = subprocess.check_output(hash_cmd)

# Unit test
random.seed(42) ✓
assert compute_walk() == [1,0,-1,-2,-1,0,1,0,-1,-2]

# Random walk for 10 steps
seed = 1
random.seed(seed)
walk = compute_walk()

# Display & save results
print(walk)
results = [
    "data" : walk,
    "seed" : seed,
    "timestamp": str(datetime.datetime.utcnow()),
    "revision" : revision,
    "system" : sys.version]
with open("results-R3.txt", "w") as fd:
    fd.write(str(results))
```

LISTING 4: Re-runnable, repeatable, reproducible, reusable random walk (R⁴)

```
# Copyright (c) 2017 N.P. Rougier and F.C.Y. Benureau
# Release under the BSD 2-clause license
# Tested with 64-bit CPython 3.6.2 / NumPy 1.12.0 / macOS 10.12.6
import sys, subprocess, datetime, random
import numpy as np

def _rng(seed): ✓
    """Return a numpy random number generator
    initialized with seed as it would be with
    a python random generator.
    """
    rng = random.Random()
    rng.seed(seed)
    _, keys, _ = rng.getstate()
    rng = np.random.RandomState()
    state = rng.get_state()
    rng.set_state((state[0], keys[-1], state[2],
                  state[3], state[4]))
    return rng

def walk(n, seed):
    """Random walk for n steps"""
    rng = _rng(seed)
    steps = 2 * (rng.uniform(-1, +1, n) > 0) - 1
    return steps.cumsum().tolist()

def compute_results(count, x0=0, step=1, seed=0): ✓
    """Compute a walk and return it with context"""
    # If repository is dirty, don't do anything
    if subprocess.call(("git", "diff-index",
                        "--quiet", "HEAD")):
        print("Repository is dirty, please commit")
        sys.exit(1)

    # Get git hash if any
    hash_cmd = ("git", "rev-parse", "HEAD")
    revision = subprocess.check_output(hash_cmd)

    # Compute results
    walk = compute_walk(count=count, x0=x0,
                        step=step, seed=seed)
    return {
        "data" : walk,
        "parameters": {"count": count, "x0": x0,
                      "step": step, "seed": seed},
        "timestamp": str(datetime.datetime.utcnow()),
        "revision" : revision,
        "system" : sys.version}

if __name__ == "__main__":
    # Unit test
    assert (walk(n=10, seed=42) ==
            [1,0,-1,-2,-1,0,1,0,-1,-2])

    # Random walk for 10 steps, with seed=1
    seed = 1
    path = walk(n=10, seed=seed)

    # Save & display results
    results = {"data": path, "seed": seed}
    with open("results-R5.txt", "w") as fd:
        fd.write(str(results))
    print(path)

    # Simulation parameters
    count, x0, seed = 10, 0, 1
    results = compute_results(count, x0=x0, seed=seed)

    # Save & display results
    with open("results-R4.txt", "w") as fd:
        fd.write(str(results))
    print(results["data"])
```

LISTING 5: Replicated random walk (R⁵)

```
# Copyright (c) 2017 N.P. Rougier and F.C.Y. Benureau
# Release under the BSD 2-clause license
# 64-bit CPython 3.6.2 / NumPy 1.12.0 / macOS 10.12.6
import random
import numpy as np

def _rng(seed): ✓
    """Return a numpy random number generator
    initialized with seed as it would be with
    a python random generator.
    """
    rng = random.Random()
    rng.seed(seed)
    _, keys, _ = rng.getstate()
    rng = np.random.RandomState()
    state = rng.get_state()
    rng.set_state((state[0], keys[-1], state[2],
                  state[3], state[4]))
    return rng

def walk(n, seed):
    """Random walk for n steps"""
    rng = _rng(seed)
    steps = 2 * (rng.uniform(-1, +1, n) > 0) - 1
    return steps.cumsum().tolist()

def compute_results(count, x0=0, step=1, seed=0): ✓
    """Compute a walk and return it with context"""
    # If repository is dirty, don't do anything
    if subprocess.call(("git", "diff-index",
                        "--quiet", "HEAD")):
        print("Repository is dirty, please commit")
        sys.exit(1)

    # Get git hash if any
    hash_cmd = ("git", "rev-parse", "HEAD")
    revision = subprocess.check_output(hash_cmd)

    # Compute results
    walk = compute_walk(count=count, x0=x0,
                        step=step, seed=seed)
    return {
        "data" : walk,
        "parameters": {"count": count, "x0": x0,
                      "step": step, "seed": seed},
        "timestamp": str(datetime.datetime.utcnow()),
        "revision" : revision,
        "system" : sys.version}

if __name__ == "__main__":
    # Unit test checking reproducibility
    # (will fail with Python<=3.2)
    assert (compute_walk(10, 0, 1, 42) ==
            [1,0,-1,-2,-1,0,1,0,-1,-2])

    # Simulation parameters
    count, x0, seed = 10, 0, 1
    results = compute_results(count, x0=x0, seed=seed)

    # Save & display results
    with open("results-R5.txt", "w") as fd:
        fd.write(str(results))
    print(results["data"])
```

Program example from Hughes, B. D. (1995). *Random Walks and Random Environments*. Oxford; New York, NY: Clarendon Press; Oxford University Press

Code is Science Manifesto

- Code is Science Manifesto (<http://codeisscience.github.io/manifesto/manifesto.html>)
 - Open over closed
 - Incorrect code results in incorrect science
 - Code for the future
 - Availability over perfection
 - Code deserves credit
 - Soft and data citation!
- Code is Science works with the scientific coding community to champion Open Source code, so we can ensure that code is peer reviewed and FAIR



Reproducibility of computational research

- Sandve G.K., Nekrutenko A., Taylor J., Hovig E.

Ten Simple Rules for Reproducible Computational Research. PLoS Computational Biology 9(10), 2013 (<http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1003285>)

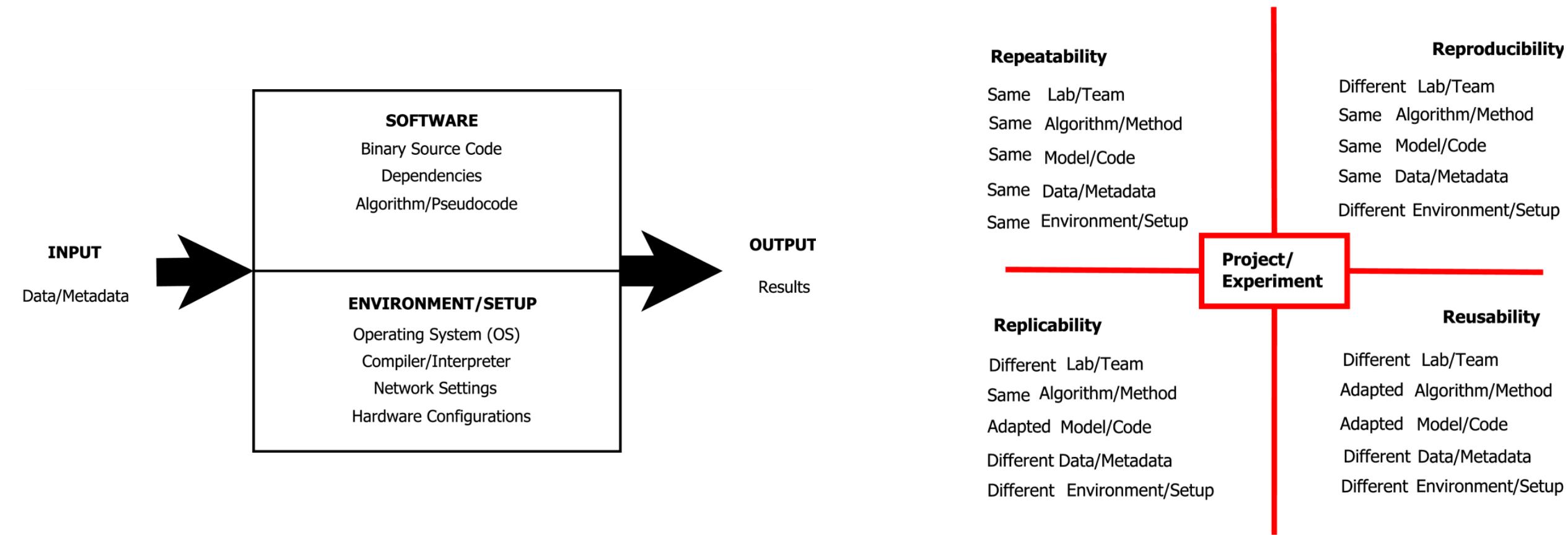
1. For every result, **keep track** of how it was produced
2. **Avoid manual** data manipulation steps
3. Archive the exact **versions** of all external programs used
4. **Version control** all custom scripts
5. Record all intermediate results, when possible in standardized formats
6. For analyses that include randomness, note underlying **random seeds**
7. Always store **raw data** behind plots
8. Generate hierarchical analysis output, allowing layers of **increasing detail** to be inspected
9. **Connect** textual **statements** to underlying **results**
10. Provide **public access** to scripts, runs, and results

Discussion on reproducibility

- There are context specific conditions and constraints!
 - Daniel S. Katz. Is software reproducibility possible and practical? 2017 (<http://danielskatzblog.wordpress.com/2017/02/07/is-software-reproducibility-possible-and-practical/>)
 - “A method to get to reproducibility and reusability while not dramatically reducing our ability to innovate is potentially to overcome software collapse by using flexible or fuzzy APIs between underlying layers, though much computer science research is needed to enable this”
 - Jesus M. Gonzalez-Barahona, Gregorio Robles. Revisiting the reproducibility of empirical software engineering studies based on data retrieved from development repository. Information and Software Technology 164, 2023 (<http://www.sciencedirect.com/science/article/pii/S0950584923001738>)
 - Method for assessing the reproducibility of studies based on mining software repositories (MSR studies)
 - Rasmussen L.V., Whitley E.W., Welty L.J. Pragmatic reproducible research: improving the research process from raw data to results, bit by bit. J Clin Invest. 2023 Aug 15;133(16):e173741 (<http://pmc.ncbi.nlm.nih.gov/articles/PMC10425207/>)
 - “Software tools can help but are not required”
 - Antunes B., Hill D.R.C. Reproducibility, Replicability and Repeatability: A survey of reproducible research with a focus on high performance computing. Computer Science Review, 53, 2024. (<http://www.sciencedirect.com/science/article/abs/pii/S157401372400039X>)

4R in publications

- Hernández J.A., Colom M. **Repeatability, Reproducibility, Replicability, Reusability (4R)** in Journals' Policies and Software/Data Management in Scientific Publications: A Survey, Discussion, and Perspectives.
arXiv:2312.11028v1, 2023 (<http://arxiv.org/abs/2312.11028v1>)



ACM artifact review and badging (1)

- Source:
 - Artifact Review and Badging Version 1.1 – 2020-08-24 (<https://www.acm.org/publications/policies/artifact-review-and-badging-current>)
 - Inspired by the International Vocabulary for Metrology (VIM)
- **Repeatability** (same team, same experimental setup)
 - The measurement can be obtained with stated precision by the same team using the same measurement procedure, the same measuring system, under the same operating conditions, in the same location on multiple trials
 - For *computational experiments*, this means that a **researcher** can reliably **repeat her own computation**
- **Reproducibility** (different team, same experimental setup)
 - The measurement can be obtained with stated precision by a different team using the same measurement procedure, the same measuring system, under the same operating conditions, in the same or a different location on multiple trials
 - For *computational experiments*, this means that an **independent group** can **obtain the same result** using the **author's own artifacts**
- **Replicability** (different team, different experimental setup)
 - The measurement can be obtained with stated precision by a different team, a different measuring system, in a different location on multiple trials
 - For *computational experiments*, this means that an **independent group** can **obtain the same result** using artifacts which they develop **completely independently**

ACM artifact review and badging (2)

- ACM recommend that three separate badges related to artifact review:

- Artifacts Evaluated

- Functional
 - Reusable



- Artifacts Available



- Results Validated

- Reproduced
 - Replicated



Research output with code and data

- Some repositories:

- Papers With Code (<http://paperswithcode.com>)
 - The latest in Machine Learning
 - **12648** benchmarks | **5313** tasks | **150,258** papers with code



- F1000Research

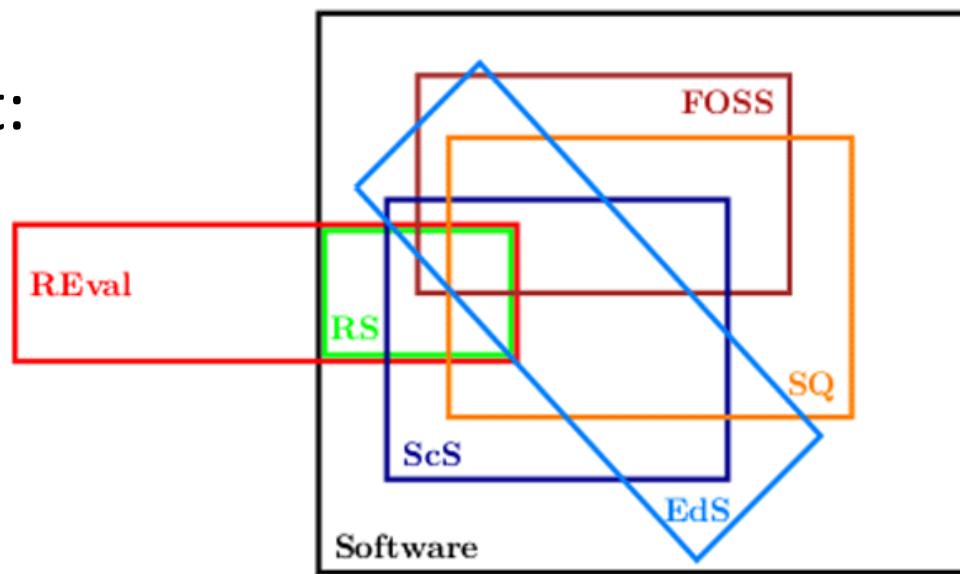
(<http://f1000research.com/collections/reproducible-research-software-data>)

- Collection “Reproducible Research Data and Software”

F1000Research

Research software evaluation

- Gomez-Diaz T and Recio T. On the evaluation of research software: the CDUR procedure. F1000Research 2019, 8:1353 (<http://f1000research.com/articles/8-1353>)
 - CDUR comprises four steps introduced as follows: **C**itation, to deal with correct RS identification, **D**issemination, to measure good dissemination practices, **U**se, devoted to the evaluation of usability aspects, **R**esearch, to assess the impact of the scientific work
- Interrelations between different software concept:

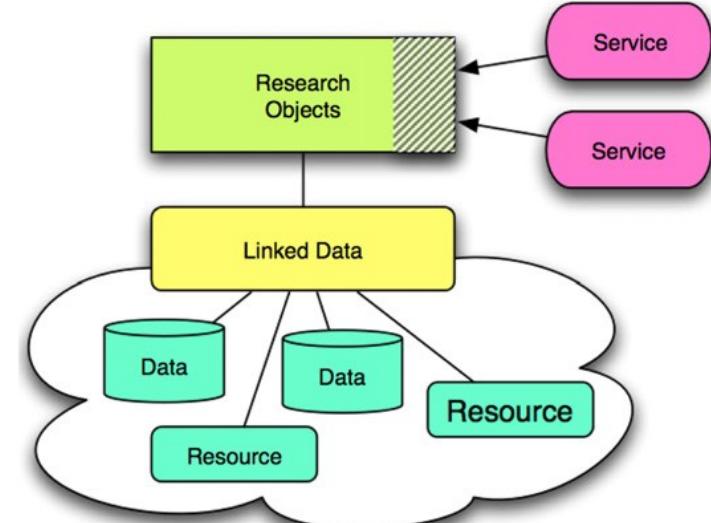


Labels:

- **REval:** Research Evaluation
- **RS:** Research Software
- **ScS:** Scientific Software
- **EdS:** Education Software
- **FOSS:** Free/Open Source Software
- **SQ:** Software Quality

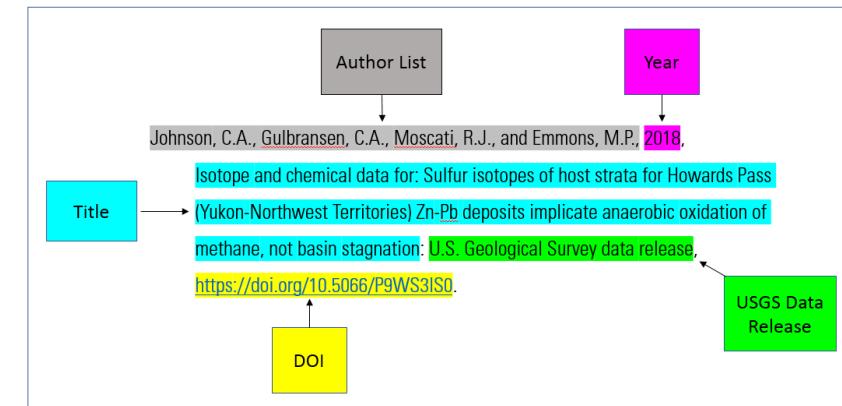
Research Objects and 7R

- The notion of **Research Objects** (ROs), semantically rich aggregations of (potentially distributed) resources that provide a layer of structure on top of information delivered as Linked Data
 - Bechhofer S., Buchan I., et al. Why Linked Data is Not Enough for Scientists / 2010 IEEE Sixth International Conference on e-Science, Brisbane, 2010, pp. 300-307 (<http://dl.acm.org/doi/10.1016/j.future.2011.08.004>)
- 7R of RO:
 - Reusable
 - Repurposeable
 - Repeatable
 - Reproducible
 - Replayable
 - Referenceable
 - Respectful



Data citation

- **Data citation** – reference to the source of the data, preferably in both bibliographic and technical senses, considering data management and data versioning aspects
 - [2014] FORCE11 – Joint Declaration of Data Citation Principles (<http://www.force11.org/datacitationprinciples>)
 - DataCite – Cite Your Data (<http://datacite.org/cite-your-data.html>)
 - Elsevier – Data citation (<http://www.elsevier.com/open-science/research-data/data-citation>)
 - USGS data management – Data Citation (<http://www.usgs.gov/data-management/data-citation>)
 - Digital Object Identifiers!



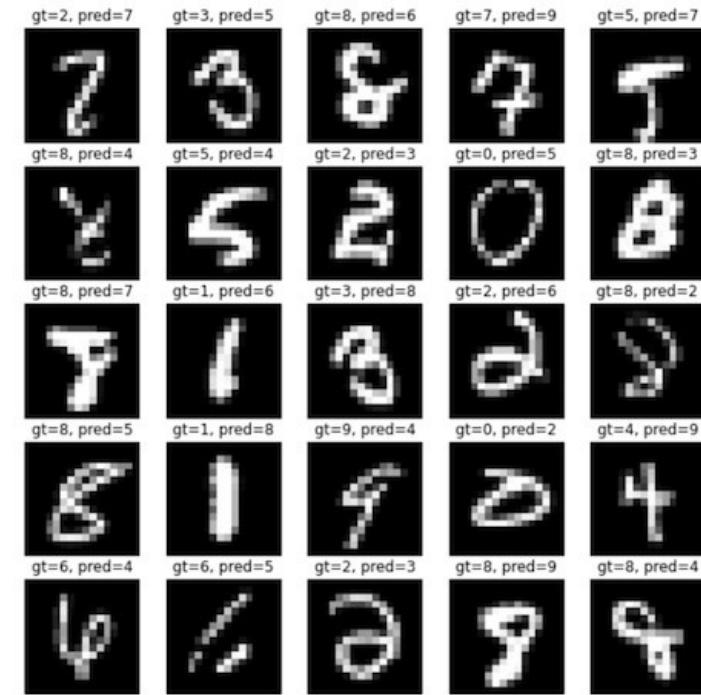
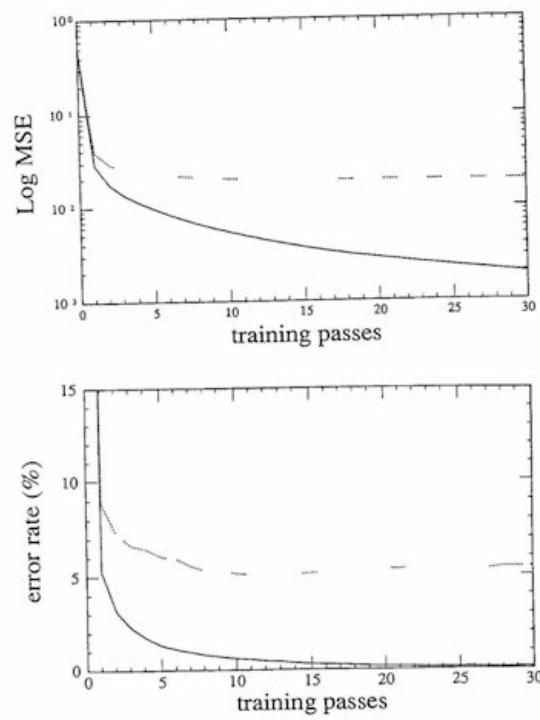
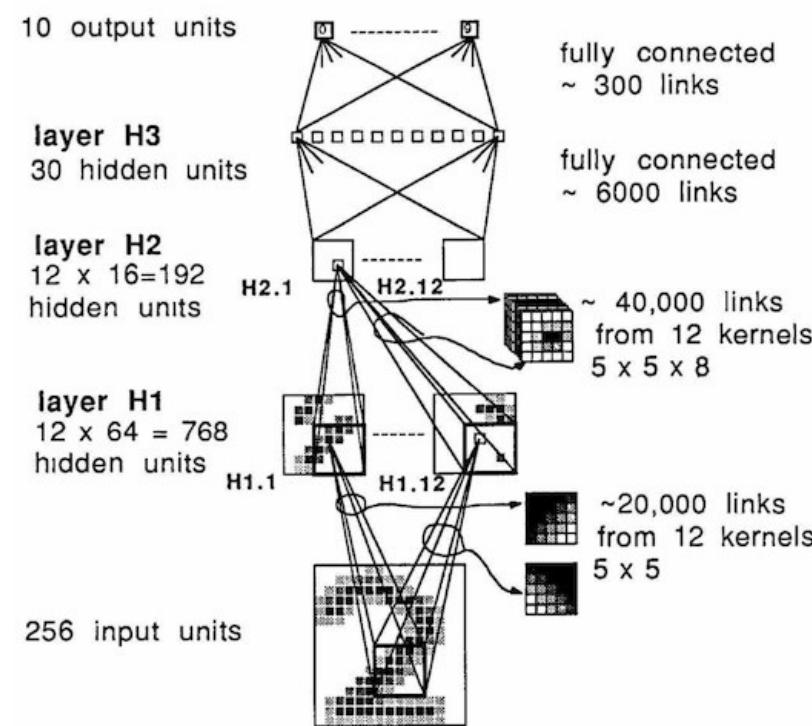
Software citation

- **Software citation** – reference to the source of the program code, preferably in both bibliographic and technical senses, considering dependencies and code versioning aspects
 - [2016] FORCE11 – Software Citation Principles
(<http://force11.org/info/software-citation-principles-published-2016/>)
 - **duecredit** (<http://github.com/duecredit/duecredit>)
 - Python library, V 0.10.2
 - duecredit is being conceived to address the problem of inadequate citation of **scientific software** and methods, and limited visibility of donation requests for open-source software
 - +
 - Citation File Format (**cff**) (<http://citation-file-format.github.io>)



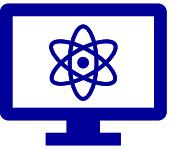
Example of reproducibility: lecun1989-repro

- Reproducing the 1989 Yann LeCun et al. paper: Backpropagation Applied to Handwritten Zip Code Recognition (<http://github.com/karpathy/lecun1989-repro>)
 - Python + Jupyter Notebook
 - Open questions!



RSE and modern AI tools

- Explicit knowledge and ontology engineering (separate report)
- Specific ML models as RSE tools
 - AlfaFold (<http://alphafold.ebi.ac.uk>)
 - ...
- LLMs in RSE
 - Vibe coding as “anti-7R” warfare?
 - BMAD-METHOD™ (<https://github.com/bmad-code-org/BMAD-METHOD>)
 - Agentic Planning + Context-Engineered Development
 - Crowson M.G., Celi L. **Academic Vibe Coding: Opportunities for Accelerating Research in an Era of Resource Constraint.** arXiv:2508.00952, 2025 (<http://arxiv.org/abs/2508.00952>)
 - Sokana AI (<http://sakana.ai>)
 - The AI Scientist Generates its **First Peer-Reviewed Scientific Publication** (<http://sakana.ai/ai-scientist-first-publication>)
 - Beel J., Kan M.-Y., Baumgart M. **Evaluating Sakana’s AI Scientist for Autonomous Research: Wishful Thinking or an Emerging Reality Towards ‘Artificial Research Intelligence’ (ARI)?** arXiv:2502.14297, 2025. (<http://arxiv.org/abs/2502.14297v2>)



Programming-in-the-small/large/many/huge/...

- ✓ Programming-in-the-Small/Large
- ✓ Programming-in-the-Many
- ✓ Programming-in-the-Huge
- ✓ Evidence-based software engineering

Programming-in-the-Small/Large/Many/Huge/...

- Programming-in-the-Large VS Programming-in-the-Small

- Most cited paper:

- F. DeRemer and H. H. Kron, "Programming-in-the-Large Versus Programming-in-the-Small," in IEEE Transactions on Software Engineering, vol. SE-2, no. 2, pp. 80-86, June **1976**
 - "... structuring a large collection of modules to form a "system" is an essentially distinct and different intellectual activity from that of constructing the individual modules. That is, we distinguish programming-in-the-large from programming-in-the-small "
 - Example of **MIL 75** language
- C.V. Ramamoorthy, V. Garg and A. Prakash, "Programming in the large," in IEEE Transactions on Software Engineering, vol. SE-12, no. 7, pp. 769-783, July **1986**
 - "... how research in automatic programming, knowledge-based systems, metrics, and programming environments can make a significant difference in the ability to develop large systems"

```
system Input
author 'Sharon Sickel'
date 'July, 1974'
provides Input_parser
consists of
root module
originates Input_parser
uses derived Parser, Post_processor
uses nonderived Language_extensions
subsystem Scan
must provide Scanner
subsystem Parse
must provide Parser
has access to Scan
subsystem Post
must provide Post_processor
```

PitSLMH – People and resources (Many ☺)

- ... vs Programming-in-the-Many

- Lars Bendix, Programming-in-the-Large versus Programming-in-the-Many, Informatik Berichte Nr. 93-05, Fachgruppe Informatik, Universität-GH-Siegen, Germany, July **1993**

- Conceptual framework:

| | | |
|----------|---------------------------|---|
| PitLarge | | <ul style="list-style-type: none">- Description<ul style="list-style-type: none">- Internal- External- Versioning repository- Selection mechanism- Configuration engine- Merging |
| PitMany | PM - personnel management | <ul style="list-style-type: none">- Method- Security |
| | RM - resource management | <ul style="list-style-type: none">- Heterogeneousness- Distribution |

PitSLMH – Next era

- Software engineering advances...
 - W. Cazzola, A. Savigni, A. Sosio and F. Tisato, "A fresh look at programming-in-the-large," Proceedings. The Twenty-Second Annual International Computer Software and Applications Conference (Compsac '98) (Cat. No.98CB 36241), Vienna, Austria, **1998**, pp. 502-506
- Dynamic Architectural Programming-in-the-Large (DAPIL)
 - Implicit architectural problems
 - Computation-in-the-large (execution of the program-in-the-large) VS computation-in-the-small (execution of the individual **components**' programs)
 - Components reusability
 - Components are architecture-transparent (do not embed architectural assumptions)
 - Components are driven by connectors
 - The PIL virtual machine "executes" the topology and strategy

PitSLMH – Additional approaches

- Evidence-based software engineering

- Barbara Kitchenham, Tore Dybå, M. Jorgensen. Evidence-based Software Engineering. 26th International Conference on Software Engineering (ICSE 2004), 2004.
 - "... evidence-based software engineering is a worthy goal for researchers interested in empirical software engineering and practitioners faced with decisions about the adoption of new software engineering technologies."
- Andreas Stefik, Stefan Hanenberg. The Programming Language Wars: Questions and Responsibilities for the Programming Language Community. ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software, 2014.
 - "We need to think more deeply about the programming language wars, before we leave a mess for the next generation to clean up."

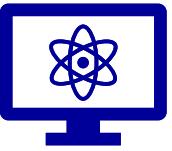
Huge and other words...

- Evolution and revolution in programming
 - Karne, Ramesh K.; Wijesinha, Alexander L.; Ford, George H. Opinion: stay on course with an evolution or choose a revolution in computing. SIGARCH Comput. Archit. News, 2008.
- Logical Semantics for CafeOBJ (abstract), 1998
[\(http://jot.theta.ro/cafeobj/abstracts/Logical-Semantics-of-CafeOBJ.html\)](http://jot.theta.ro/cafeobj/abstracts/Logical-Semantics-of-CafeOBJ.html)
 - Programming “in the **small**”
 - Programming “in the **large**”
 - Programming “in the **huge**”
 - Environment!
 - “**in the small** refers to collections of program statements (as obtained by flattening the individual modules, programming **in the large** to the module interconnection system, programming **in the huge** to the software system composition, and the **environment** to the set of tools (including methodologies)”
 - 3rd principle: “Develop all ingredients (concepts, results, etc.) at **the highest appropriate level of abstraction**”

Programming-in-the-small and AI

- The field of “sport” programming
 - Programming-in-the-very-small-but-algorithmically-hard ☺
- ICPC 2025 (<http://worldfinals.icpc.global/2025/>)
 - OpenAI's models (?) successfully solved all **12 problems** – a milestone akin to achieving a gold medal at the highest level of achievement! (<http://worldfinals.icpc.global/2025/openai.html>)
 - An advanced version of **Google Gemini 2.5 Deep Think** has achieved gold-medal level performance at the 2025 International Collegiate Programming Contest (ICPC) World Finals
(<http://deepmind.google/discover/blog/gemini-achieves-gold-level-performance-at-the-international-collegiate-programming-contest-world-finals/>)
 - Gemini solved **10 of 12 problems**, achieving gold-medal level
 - If the best AI and human solutions in the competition were combined, all 12 problems would have been solved completely and correctly



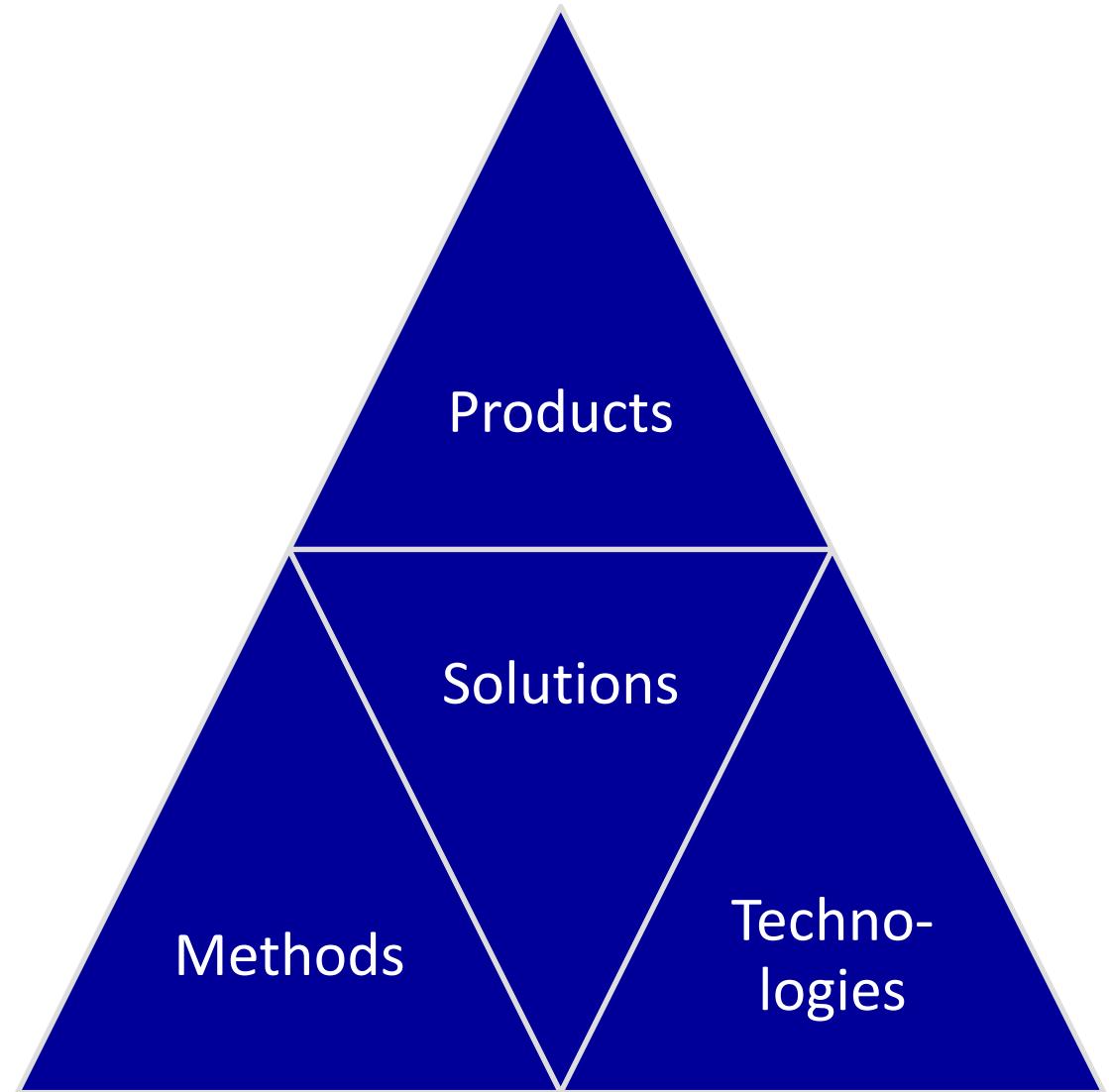


Technologies – platforms

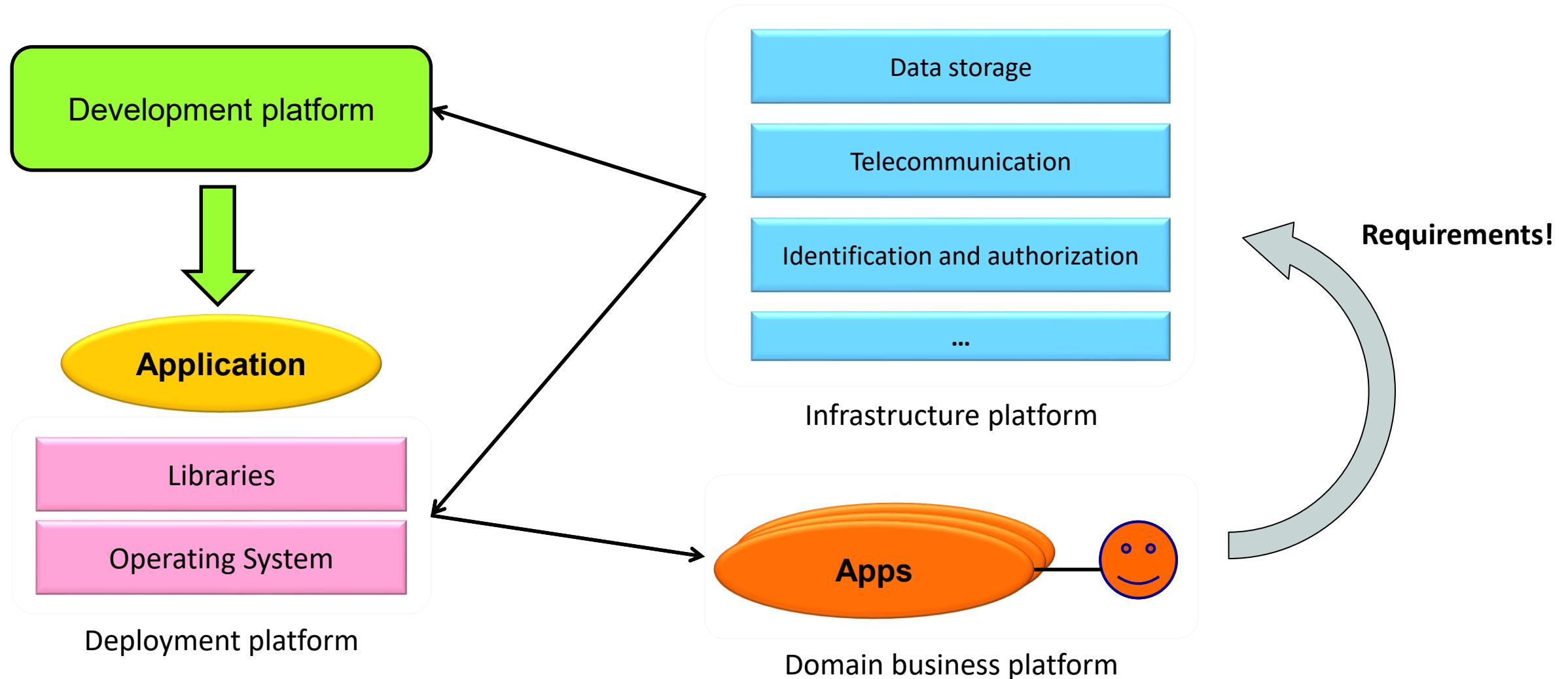
- ✓ Platform oriented approach
- ✓ Infrastructure for RSE
- ✓ Reproducible computational experiments at scale
 - ✓ Orchestration
- ✓ Security

IT platforms and services

- Platforms:
 - Infrastructure
 - Development
 - Deployment
- Interoperability
- Services
- Efficacy/performance
- Security
- ...



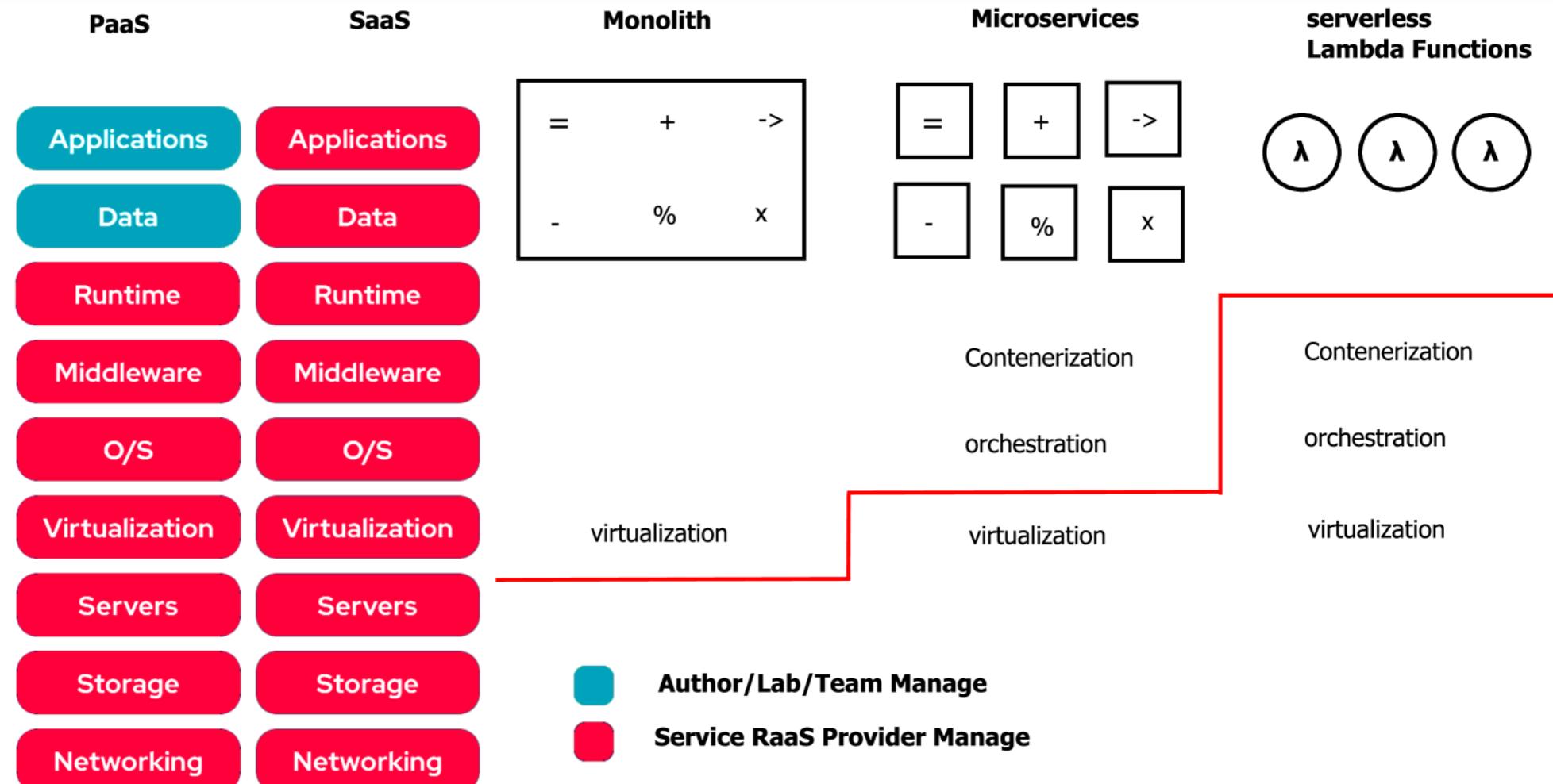
Platforms and domains



RSE infrastructure?

- Infrastructure of RSE as specific infrastructure platform
 - RSE Book, Part 7 – Infrastructure (<http://rse-book.github.io/infrastructure.html>)
- Modern aspects:
 - Virtualized computational and storage resources
 - Infrastructure as Code
 - Orchestration
 - Scheduling
 - Managed storages + ETL

Reproducibility as a Service (RaaS)

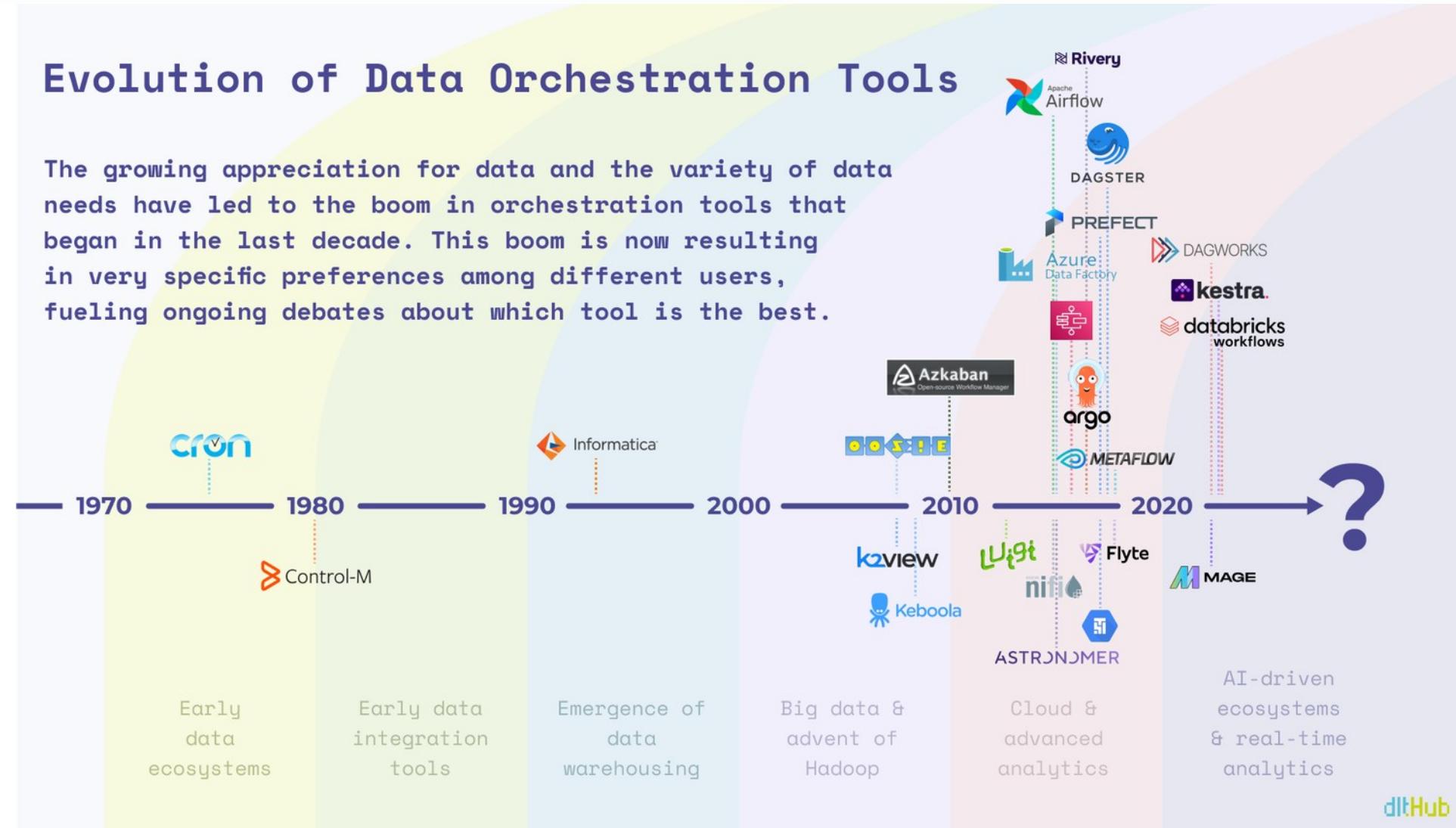


Representation of a RaaS-managed cloud infrastructure (<http://arxiv.org/html/2312.11028v1>)

Orchestration and scheduling

- DataCamp – Orchestration vs. Scheduling | Theory
(<http://campus.datacamp.com/courses/understanding-modern-data-architecture>)
- A Brief History of Workflow Orchestration. 2022
(<http://www.prefect.io/blog/brief-history-of-workflow-orchestration>)
- Anuun Chinbat. On Orchestrators: You Are All Right, But You Are All Wrong Too. 2024 (<http://dlthub.com/blog/on-orchestrators>)

Orchestrators evolution



Variant of orchestrators classification

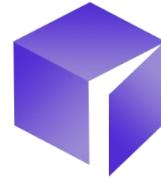


ditHub

© Anuun Chinbat, 2024

Security

- Identity and Access Management platforms
 - Catalogs, federation and multi-layer authorization
 - Keycloak (<http://www.keycloak.org>)
 - Casdoor (<http://github.com/casdoor/casdoor>)
 - An open-source **UI-first** Identity and Access Management
 - Authentik (<http://goauthentik.io>)
- Roles
 - Collaborators
 - Guests as viewers
 - Auditors
 - ...
- Simple access to shared resources
 - Including ETL

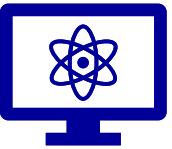


How to select?

- Example of comparison
(<http://goauthentik.io/#comparison>)

- Deployment
- Cloud infrastructure
- Integration
- Scripting support

| | authentik | Keycloak | Microsoft ADFS | Azure/Entra ID | Okta | Duo | Authelia |
|---|-----------|----------|----------------|----------------|------|-----|----------|
| Capabilities included | | | | | | | |
| Self-host anywhere | ✓ | ✓ | ⚠ | ✗ | ✗ | ✗ | ✓ |
| MFA ⓘ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ |
| Conditional Access | ✓ | ✓ | ✓ | ⚠ | ✓ | ⚠ | ✓ |
| Open-source/Source available | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Application Proxy ⓘ | ✓ | ⚠ | ⚠ | ✓ | ✗ | ✗ | ✗ |
| Enterprise support ⓘ | ✓ | ⚠ | ✗ | ✓ | ✓ | ✓ | ✗ |
| Protocol Support (as a provider) | | | | | | | |
| SAML2 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| OAuth2 and OIDC | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| SCIM | ✓ | ⚠ | ✗ | ✓ | ✓ | ✗ | ✗ |
| LDAP | ✓ | ✗ | ⚠ | ⚠ | ✓ | ⚠ | ✗ |
| RADIUS | ✓ | ⚠ | ✗ | ⚠ | ⚠ | ⚠ | ✗ |
| Federation support | | | | | | | |
| SAML2 | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ |
| OAuth2 and OIDC | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ |
| OAuth1 | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ |
| LDAP | ✓ | ✓ | ✓ | ⚠ | ✓ | ✓ | ✓ |
| SCIM | ✓ | ⚠ | ✗ | ✓ | ✓ | ✗ | ✗ |
| Kerberos | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ |
| Use cases | | | | | | | |
| Authentication | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Enrollment | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Self-service | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |



Collaborative Technologies

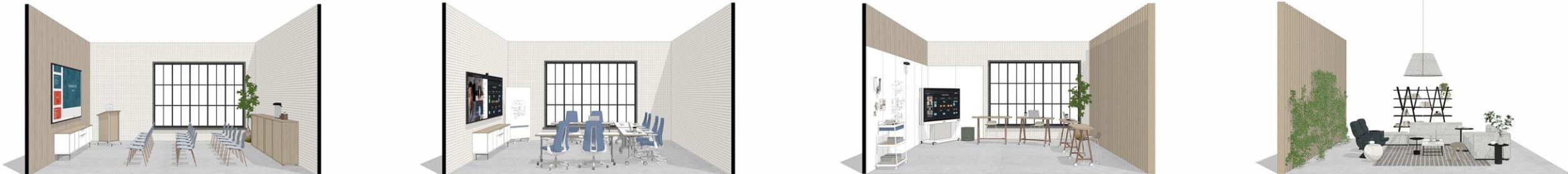
- ✓ Collaboration
- ✓ Collaborative spaces
- ✓ RSE and collaboration
 - ✓ BoKs
 - ✓ Literature programming
 - ✓ Interactive books

Collaborations and collaborative tools

- 1960s: **Engelbart D.C.** – Father of Groupware
 - And the author of "The Mother of All Demos" (<http://douengelbart.org/content/view/209/448/>)!
 - Engelbart D.C. Collaboration Support Provisions in AUGMEN // AFIPS Office Automation Conference, 1984 (<http://www.douengelbart.org/pubs/oad-2221.html>)
 - Engelbart D.C. Toward High-Performance Organizations: A Strategic Role for Groupware // GroupWare '92 Conference, 1992 (<http://www.douengelbart.org/pubs/augment-132811.html>)
- **Collaboration** – interactive process that engages two or more participants who work together to achieve outcomes they could not accomplish independently
 - In electronic collaboration, the “interactive process” is conducted using information and communications technologies (ICT)
 - Salmons J., Wilson L. Handbook of Research on Electronic Collaboration and Organizational Synergy (2 Volumes), 2008, 1026 p.

Collaborative space evolution

- Meeting room → Virtual room → Virtual creative space → Metaverse
- Yu, R.; Gu, N.; Masoumzadeh, S. Exploring the Impact of Digital Technologies on Team Collaborative Design. *Buildings* 14, 2024. 3263.
- Leminen, S.; Vita, K.; Westerlund, M.; Ritala, P. Places and spaces of collaborative R&D and innovation: navigating the role of physical and virtual contexts. *R&D Management* 54(2), 2024. pp. 201-213.
- 4 Types of Collaborative Spaces (<http://www.haworth.com/na/en/spark/articles/2019/4-types-of-collaborative-spaces.html>):



Digital collaborative space

- Technology stack:
 - Infrastructure platform for distributed computing
 - Identity Management System
 - Distributed data storage with versioning
 - Collaborative activity support system
 - Omnichannel communication subsystem
 - Subsystems of collaborative work with artifacts in real time
 - By media and type
 - Integration components
 - Integrated space
 - Artifacts import/export
 - Publishing
 - Client components (“automated workplaces”)
 - For BYOD paradigm

The vision of enterprise metaverse

- Digital twins: The foundation of the enterprise metaverse

(<https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/digital-twins-the-foundation-of-the-enterprise-metaverse>)

- Digital twins will power the enterprise metaverse
- Immersive experiences, enabled by augmented (AR) and virtual reality (VR), will allow employees to gain real-world product design experience and training from their desks as they manipulate 3-D digital replicas of equipment
- Mass simulations and AI technologies will use data streams from across and beyond the enterprise to help senior executives predict what's next with tremendous precision and prescribe the best course of action in even the most turbulent of times
- R&D organization will work with 3-D replicas, and its impact on customers and existing processes is automatically simulated
- ...



Why do companies use collaboration software?

- What Is Collaboration Software & Why Your Organization Needs It?
(<http://www.slingshotapp.io/blog/what-is-collaboration-software>)
 - **Improved communication** – Collaboration software allows team members to communicate in real time, making it easier to ask questions, share information, and resolve issues quickly
 - **Enhanced collaboration** – Online collaboration tools enable team members to work together seamlessly, regardless of their physical location, improving teamwork and increasing efficiency
 - **Increased productivity** – Collaboration software provides a centralized location for team members to access project information and collaborate on tasks, reducing the time and effort required to complete projects
 - **Project management** – On top of it all, collaboration software provides tools for tracking tasks, setting deadlines, and monitoring progress, making it easier to manage projects and keep everyone on track

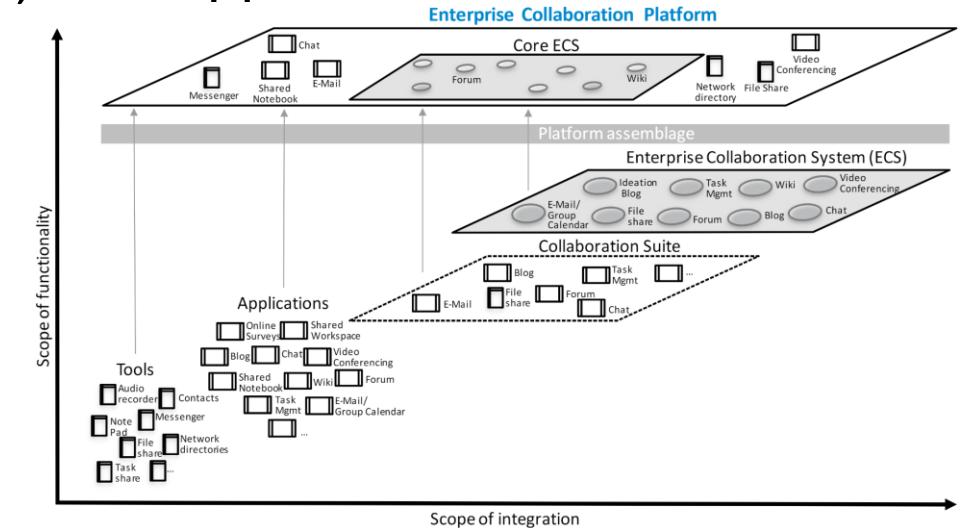


Collaborative tools and peoples

- Pros & Cons for people in organization
 - Artifacts sharing tools
 - Microsoft – Content collaboration – People experiences and Adoption Score (<http://learn.microsoft.com/en-us/microsoft-365/admin/adoption/content-collaboration>)
 - Social media tools
 - Lane, J.N.; Leonardi, P.M.; Contractor, N.S.; DeChurch, L.A. Teams in the Digital Workplace: Technology's Role for Communication, Collaboration, and Performance. *Small Group Research* 55(1), 2024. pp. 139-183. (<http://journals.sagepub.com/doi/10.1177/10464964231200015>)
 - Specific digital collaborative R&D/ Academic/... tools
 - Collaborative Research Environment (CoRE)
 - Microsoft **Teams** Education (<http://learn.microsoft.com/en-us/training/educator-center/product-guides/teams>)
 - **Foldercase** – Revolutionizing Collaborative Research (<http://www.foldercase.com>)
 - **Authorea** – Open Research Collaboration and Publishing (<http://www.authorea.com>)
 - **Mendeley** – Reference Management Software (<http://www.mendeley.com>)
 - **Zotero** – Your personal research assistant (<http://www.zotero.org>)
 - ...

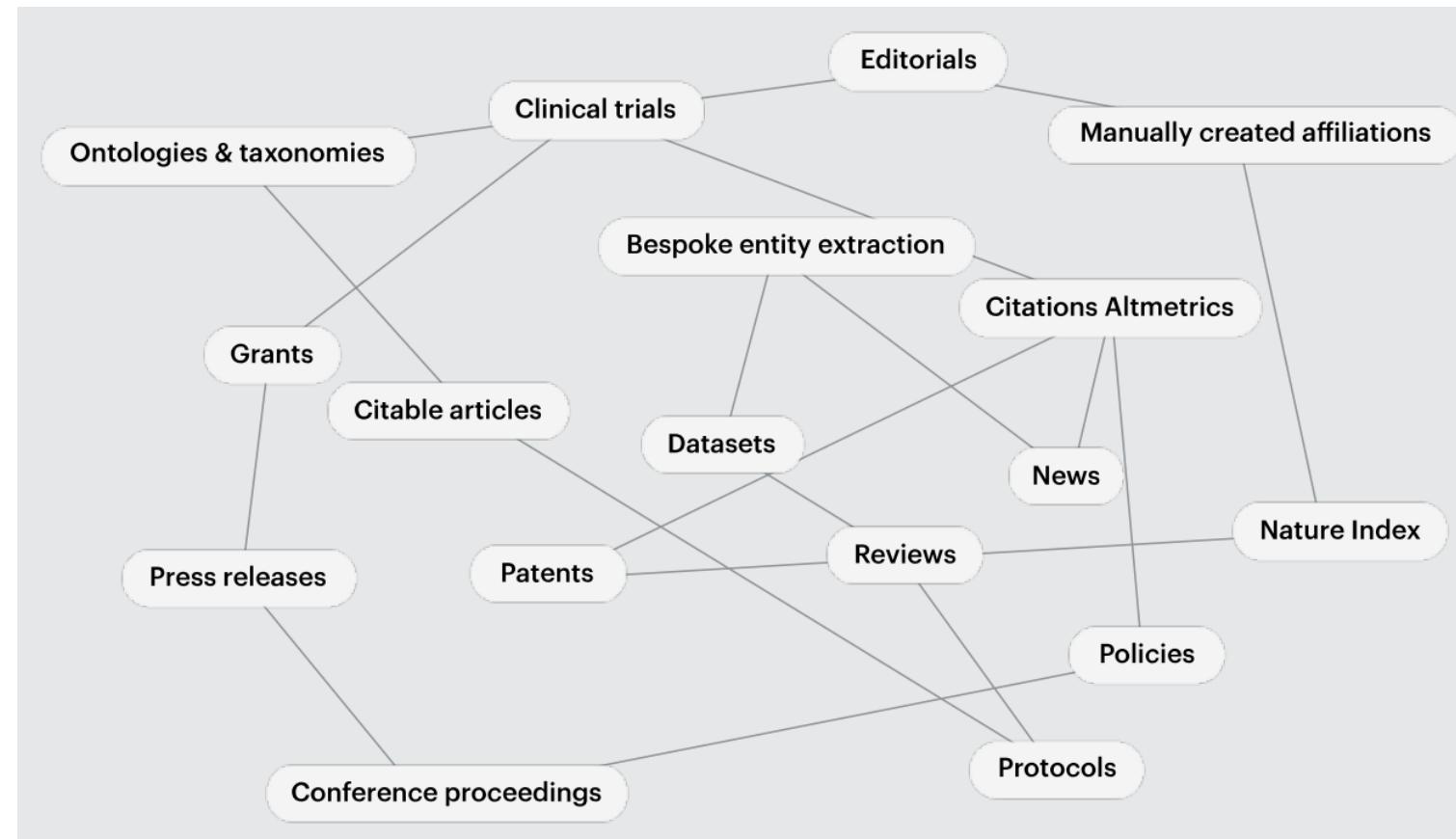
And what about R&D departments?

- Many sociological and organizational studies!
 - Enterprise Collaboration Platforms: An Empirical Study of Technology Support for Collaborative Work. Procedia Computer Science 196, 2022. pp. 305–313.
 - Levels and capabilities
- Buntz B. Is your R&D in the ‘Valley of Death’? 2024
(<http://www.rdworldonline.com/is-your-rd-in-the-valley-of-death/>)
 - Align teams, data, and technology!



Example of guidelines

- Building a Collaborative Research Ecosystem: A Guide for Research Managers
 - (<http://www.nature.com/research-intelligence/file/fda50840-1aff-48e8-b577-e0c4c9e6d9ee>)



Some actualized bodies of knowledge

- **Systems Engineering Body of Knowledge (SEBoK)**
 - SEBoK - Guide to the Systems Engineering Body of Knowledge (<http://www.sebokwiki.org>)
- **Industrial Engineering Body of Knowledge (ISEBoK)**
 - IIE - Body of Knowledge (<http://www.iise.org/Details.aspx?id=43631>)
- **Software Engineering Body of Knowledge (SWEBOK)**
 - SWEBOK wiki (<http://swebokwiki.org>)
- **Data Management Body of Knowledge (DAMA DMBOK®)**
 - DAMA International - Body of Knowledge (<http://www.dama.org/cpages/body-of-knowledge>)
- **Information Management Body of Knowledge (IMBoK)**
 - Information Management Body of Knowledge (<http://www.imbok.info>)
- **Enterprise Information Technology Body of Knowledge (EITBoK)**
 - EITBoK wiki (<http://eitbokwiki.org>)
- **Enterprise Architecture Body of Knowledge (EABOK)**
 - EABOK Glossary (<http://www2.mitre.org/public/eabok/pdf/glossary.pdf>)
- **Project Management Body of Knowledge (PMBOK®)**
 - PMBOK® Guide and Standards (<http://www.pmi.org/standards/pmbok>)
- **APM Body of Knowledge**
 - APM Body of Knowledge online (<http://www.apm.org.uk/body-of-knowledge/>)
- **Business Architecture Body of Knowledge® (BIZBOK® Guide)**
 - Business Architecture Guild - Public Resources (<http://www.businessarchitectureguild.org/page/002>)
- **Business Analysis Body of Knowledge (BABOK®)**
 - IIBA - A Guide to the Business Analysis Body of Knowledge® (BABOK® Guide) (<http://www.iiba.org/standards-and-resources/babok/>)
- **Innovation Management Body of Knowledge (IMBoK)**
 - GIM INSTITUTE - THE INNOVATION MANAGEMENT BODY OF KNOWLEDGE (<http://www.giminstitute.org/index.php/the-innovation-management-body-of-knowledge/>)



Literature programming and its mutations

- 1984 – Donald Knuth
 - Knuth, Donald E. "Literate Programming". *The Computer Journal*, 27(2), 1984. British Computer Society: 97–111.
 - Knuth, Donald E.; Levy, Silvio. *The CWEB System of Structured Documentation*, Version 3.0. Addison-Wesley, 1994.
 - WEB is a software system that facilitates the creation of readable programs
 - CWEB is a version of WEB for documenting C and C++ programs
- Literate Programming (<http://www.literateprogramming.com>)

Some environments for literature programming

● Oberon (<http://oberon.org>)



- Blackbox – environment with dynamic loading of modules (compiled into machine code) and garbage collection, i.e. provides its own component object model
- Writing, compiling, executing, testing, ...

● Pharo (<http://pharo.org>)

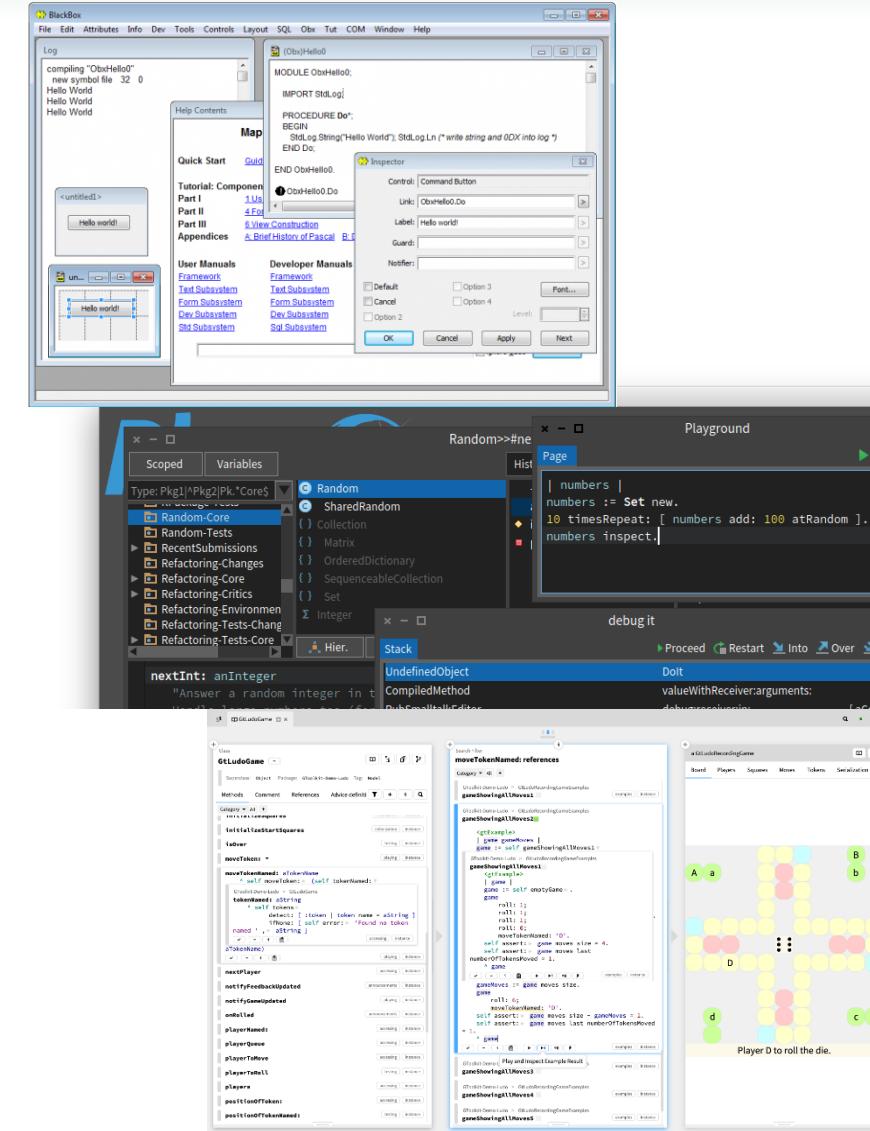


- A pure object-oriented programming language and a powerful environment, focused on simplicity and immediate feedback (think IDE and OS rolled into one)

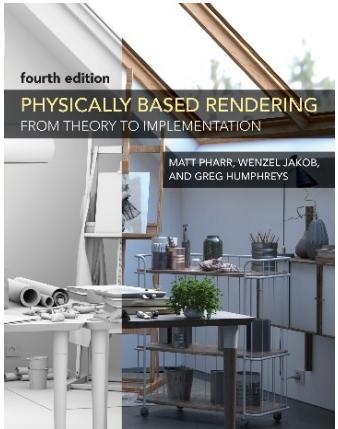
● Gtoolkit (<http://gtoolkit.com>)



- Moldable Development Environment
- Lepiter: Knowledge Management + Multi-language Notebooks + Moldable Development
- Primarily implemented in Pharo



Example in books: Physically Based Rendering



- Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically Based Rendering: From Theory To Implementation*. 2023 (<http://pbr-book.org>)
 - Describes both the mathematical theory behind a modern photorealistic rendering system as well as its practical implementation
 - Free online interactive version!

● “This book has deservedly won an Academy Award.
I believe it should also be nominated for a Pulitzer Prize”
— Donald Knuth



9.2 Diffuse Reflection

One of the simplest BRDFs is the Lambertian model, which describes a perfect diffuse surface that scatters incident illumination equally in all directions. It is a reasonable approximation to many real-world surfaces such as paper or matte paint. The Lambertian model captures the behavior of such diffuse materials relatively well, though the approximation tends to perform worse for light arriving at a grazing angle, where specular reflection causes a noticeable deviation from uniformity. (Microfacet models such as those presented in Section 9.6 can account for such effects.)

It is interesting to note that surfaces created from polytetrafluoroethylene (PTFE) powder are known to be particularly good Lambertian reflectors. They are commonly used to calibrate laboratory equipment for this reason.

```
<<DiffuseBxDF Definition>>=
class DiffuseBxDF {
public:
    <<DiffuseBxDF Public Methods>> ▾
private:
    SampledSpectrum R;
};
```

The constructor takes a reflectance spectrum R with values in the range $[0, 1]$ that specify the fraction of incident light that is scattered.

```
<<DiffuseBxDF Public Methods>>=▼
DiffuseBxDF(SampledSpectrum R) : R(R) {}
```

The reflection distribution function is just a constant, though it requires a normalization factor equal to $\frac{1}{\pi}$ so that the total integrated reflectance equals R .

$$\int_{H^2(\mathbf{n})} f_r(p, \omega_o, \omega') \cos \theta' d\omega' = \int_0^{2\pi} \int_0^{\frac{\pi}{2}} \frac{R}{\pi} \cos \theta' \sin \theta' d\theta' d\phi' = R.$$

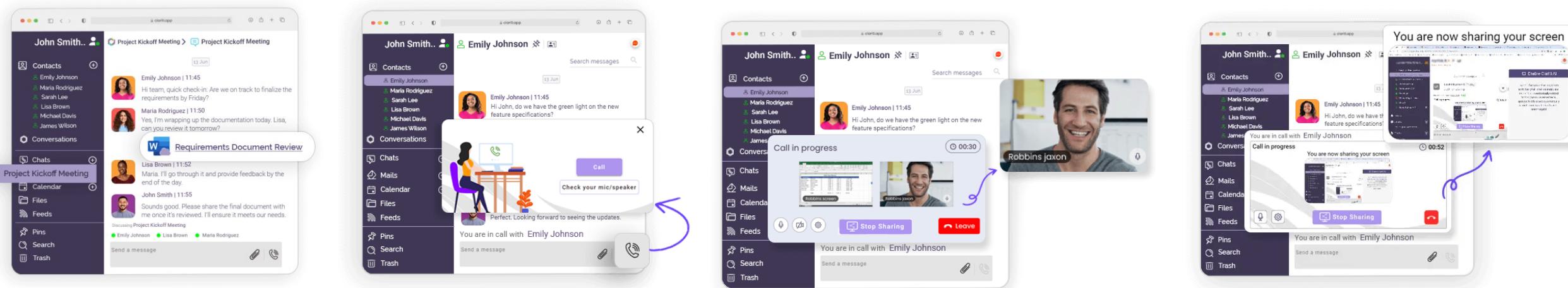
With this correction, the $f()$ implementation is given by

```
<<DiffuseBxDF Public Methods>>+▲▼
SampledSpectrum f(Vector3f wo, Vector3f wi, TransportMode mode) const {
    if (!SameHemisphere(wo, wi))
        return SampledSpectrum(0.f);
    return R * InvPi;
}
```

The sampling function returns an invalid sample if the caller specified that reflection components of

Collaborate with AI agents

- Main collaborative platforms are actively adding AI agents
 - Microsoft 365, Google Workspace, ...
- Clarity (<http://clariti.app>)
 - AI-powered team communication for your warriors
 - Messaging, emailing, voice/video calling, screen sharing, etc. with AI-driven context suggestions
 - Integration with other tools



Microsoft Office 365 AI Tools

- Assist (universal helpers)
 - Reinvent productivity with Microsoft 365 Copilot (<http://www.microsoft.com/en-us/microsoft-365/copilot>)
 - Convergence of tools, for example, Researcher → Copilot
 - Copilot (<http://copilot.microsoft.com>)
- Design
 - AI PowerPoint presentation generator (<http://www.microsoft.com/en-us/microsoft-365/powerpoint/ai-powerpoint-generator>)
 - Designer (<http://designer.microsoft.com>)
- Process text
 - Generative AI in Word (<http://www.microsoft.com/en-us/microsoft-365/word/word-ai>)
 - Grammar, format & rephrase + conciseness
 - Summarization
- Translate
 - Translate text into a different language (<http://support.microsoft.com/en-us/office/translate-text-into-a-different-language-287380e4-a56c-48a1-9977-f2dca89ce93f>)
- Transcribe
 - Transcribe your recordings (<http://support.microsoft.com/en-us/office/transcribe-your-recordings-7fc2efec-245e-45f0-b053-2a97531ecf57>)
- Read aloud
 - Listen to your Word documents (<http://support.microsoft.com/en-us/office/listen-to-your-word-documents-5a2de7f3-1ef4-4795-b24e-64fc2731b001>)
- ...

Microsoft Word - Text tools example

A large language model (LLM) is a language model consisting of a neural network with many parameters (typically billions of weights or more), trained on large quantities of unlabeled text using self-supervised learning or semi-supervised learning.^[1] LLMs are capable of performing a wide range of tasks. This has shifted the focus of natural language processing from a paradigm of training specialized supervised models to a paradigm of training general-purpose models that can perform a wide range of tasks.^[2]

Though the term large language model has no strict definition, it generally refers to a model having a parameter count on the order of billions. These models are designed to excel at a wide range of tasks, as opposed to being specialized for a single task like machine translation, named entity recognition, or mathematical problem solving. They can also handle complex tasks, such as generating human-like text, translating between languages, answering questions, and even solving math problems. The range of tasks at which they are capable of performing well is vast, from simple text generation to complex reasoning tasks.

Though trained on simple tasks along the lines of predicting the next word in a sentence, neural language models with sufficient training and parameter counts are found to capture much of the syntax and semantics of human language. In addition, large language models demonstrate considerable general knowledge about the world, and are able to "memorize" a great quantity of facts during training.^[2]

Editor Score: 83 %

Vocabulary: More specific adjectives are clearer and add impact

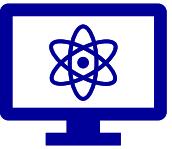
Formal writing: How formal

Corrections:

- Spelling: 3
- Grammar: 2

Refinements:

- Clarity: ✓
- Conciseness: ✓
- Formality: ✓



Interactive reactive notebooks

- ✓ Classical notebooks VS reactive notebooks
- ✓ Types of reactivity
- ✓ Editing and executing
- ✓ Deployment
- ✓ Transform into Web-App
- ✓ The reactive principals in scientific computing for 5R

Notebooks – Open source implementations

- **JupyterHub** – multi-user Hub that spawns, manages, and proxies multiple instances of the single-user Jupyter notebook server
(<http://github.com/jupyterhub/jupyterhub>)
 - **Nebari** – Your open source data science platform. Built for scale, designed for collaboration
(<http://github.com/nebari-dev/nebari>)
 - Example:
 - R. Castilla, M. Peña. Jupyter Notebooks for the study of advanced topics in Fluid Mechanics, Comput. Appl. Eng. Educ. 2023; 31: 1001–1013 (<http://onlinelibrary.wiley.com/doi/full/10.1002/cae.22619>)
- **Pluto** – Simple, **reactive** programming environment for the Julia Language
(<http://plutojl.org>)
 - We are on a mission to make scientific computing more accessible and fun.
- **Marimo** – an open-source **reactive** notebook for Python — reproducible, git-friendly, executable as a script, and shareable as an app (<http://marimo.io>)

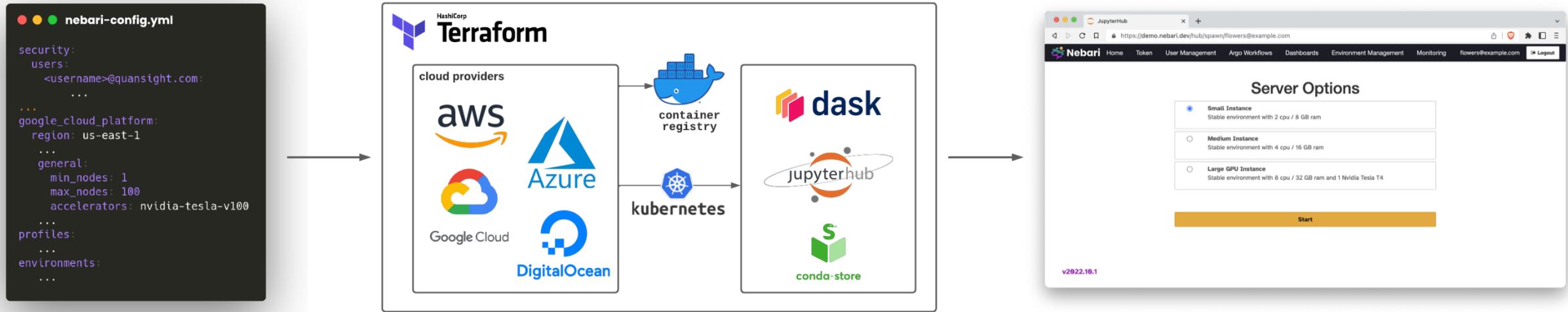
Notebooks – References

- **Wolfram Notebooks** – the preeminent environment for any technical workflows—data science, modeling, research and education—empowering users with literate programming, easy-to-author interactivity and universal deployment across all platforms (<http://www.wolfram.com/notebooks/>)



JupyterHub

- Best deploy – Nebary Terraform 😞



Marimo reactive notebook

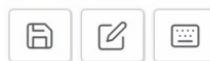
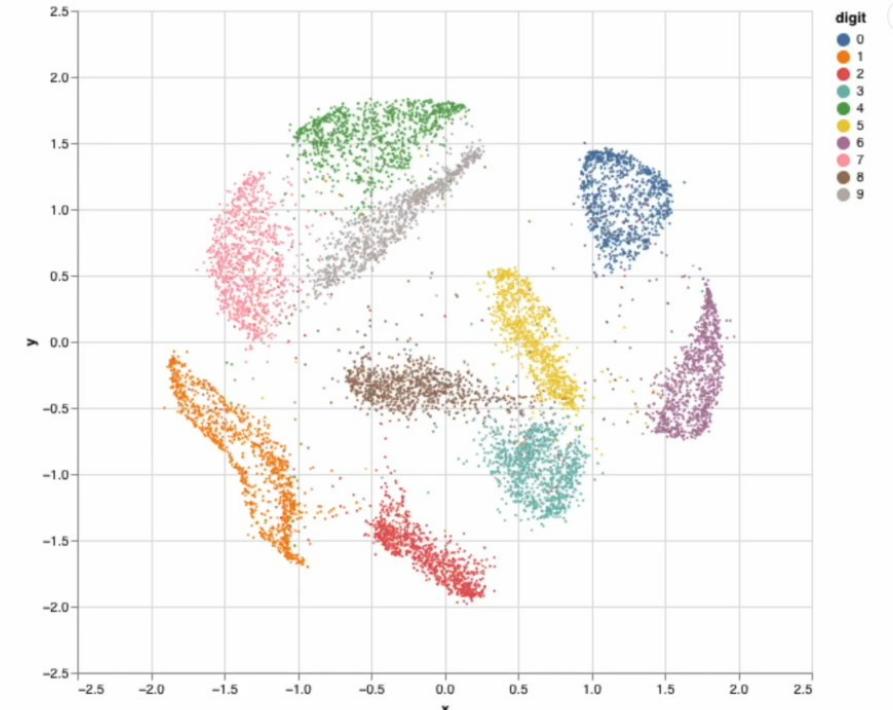


Embedding Analysis

Here's an embed INIST: each point represents a digit, with similar digits close to each other.

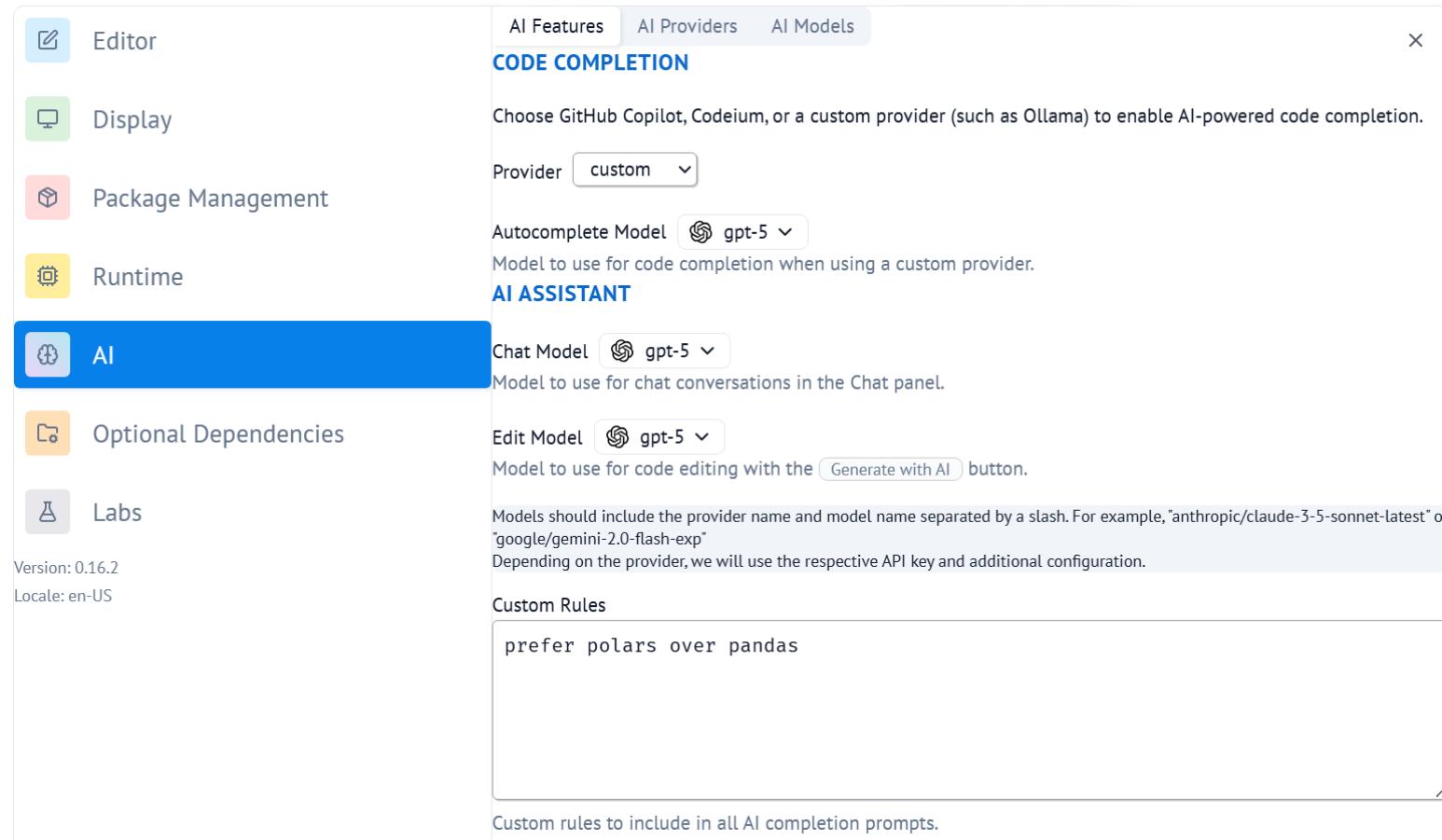
n items

There are **13108** data points in our embedding.



Marimo AI settings

- AI as an assistant at:
 - Code completion
 - Code editing
 - With errors handling!
 - Chat
 - “Add to Notebook” button!



Marimo AI example

The screenshot shows the Marimo AI interface. On the left, a 'CHAT' window displays a conversation about connecting to the OpenAI GPT-5 API. It includes steps for installing the openai package, setting the LLM_TOKEN environment variable, and writing Python code to connect to the API. A red arrow points from the 'Add to Notebook' button in the CHAT window to the 'Add to Notebook' button in the code editor.

CHAT

Provide connection to OpenAI GPT5 API in python. Use token from environment variable `LLM_TOKEN`.

To connect to the OpenAI GPT-5 API in Python using a token stored in an environment variable named `LLM_TOKEN`, you can follow these steps:

1. Install the `openai` package if you haven't already:
sh
1 `pip install openai`
2. Set the `LLM_TOKEN` environment variable in your system or within your script. For demonstration purposes, let's assume you're setting it in your script:
python
1 `import os`
2 `os.environ['LLM_TOKEN'] = 'your_openai_api_token_here'`
3. Write the Python code to connect to the OpenAI API using the token:
python
1 `import openai`
2 `import os`
3
4 `# Load the token from the environment variable`
5 `api_token = os.getenv('LLM_TOKEN')`
6
7 `# Initialize the OpenAI client`
8 `openai.api_key = api_token`
9
10 `# Test the connection by sending a request`
11 `response = openai.Completion.create(`
12 `engine="text-davinci-003", # Replace with the correct engine if needed`
13 `prompt="Hello, world!",`
14 `max_tokens=50`
15 `)`
16
17 `print(response.choices[0].text.strip())`

Here's a breakdown of the code:
We import the necessary modules: `openai` for interacting with the OpenAI API and `os` for accessing environment variables.

Type your message...
gpt-5

mo_test_llm.py

Generate with AI, @ to include context
Start coding or generate with AI.
gpt-5 Include all code

OpenAI connection test

```
1 import marimo as mo
2
3 # Generate with AI, @ to include context
4
5 # Start coding or generate with AI.
6
7
8
9
10
11
12
13
14
15
16
17
```

PYTHON MARKDOWN SQL GENERATE WITH AI

Monolanguage problems (python)

- Python ecosystem!
 - See next parts for systematic review
- Problems:
 - Reusing as **component** and **end-user tool**
 - Context governing
 - Development jactation and legacy code support
 - Example: ISO 8601 support and time zone awareness
 - Packages hell
 - Recipe: **uv** package manager (see below)
 - Resources-on-import
 - Example: pytorch memory consumption 😞
 - Recipe: More **rust-based** libraries!
 - Logging
 - Recipe: **structlog** – the production-ready logging solution for Python (<http://www.structlog.org>)

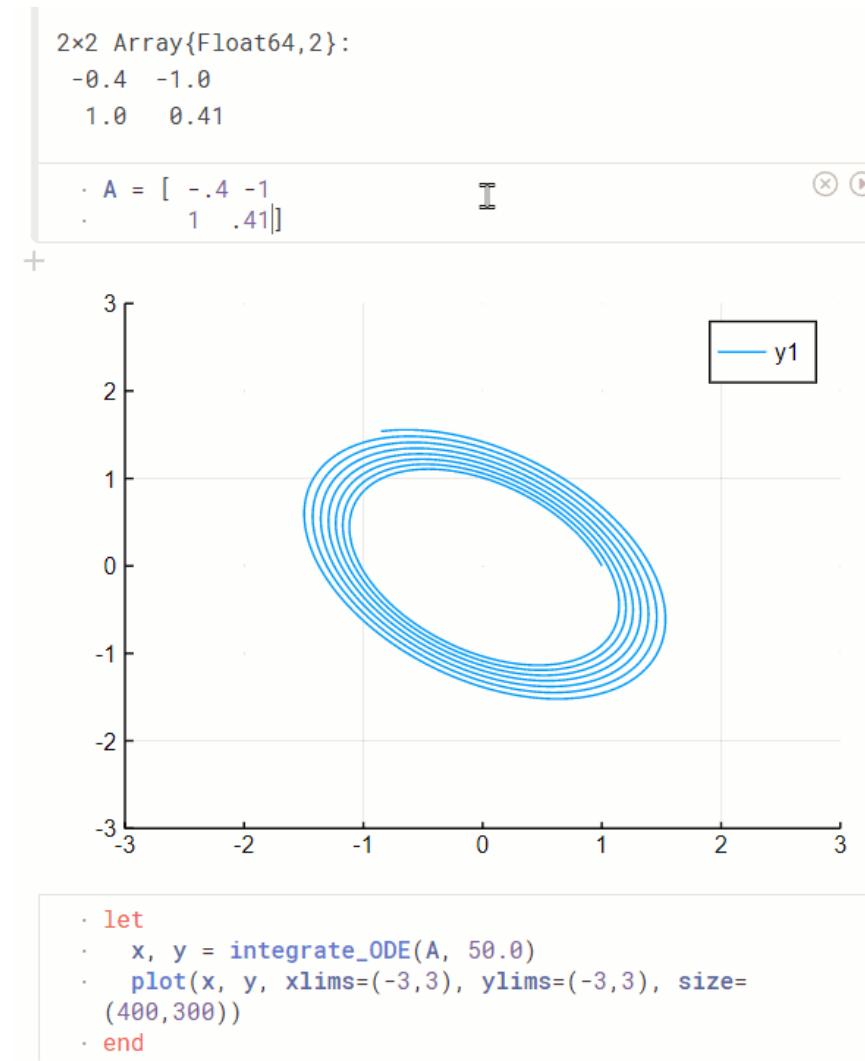
Julia world – Pluto

- Methodology and language support
 - **Julia** (<http://julialang.org>)
- Experimentation with Pluto and Makie
 - **Pluto** (<http://plutojl.org>)
 - Reactive notebooks for science
 - **Makie** (<http://docs.makie.org>)
 - Data visualization ecosystem for the Julia programming language
 - High performance and extensibility
- Experience in simulation tasks -> Efficient implementation in Pluto notebooks

Pluto JL

- It outperforms Jupiter notebooks:

- Fully interactive and reactive
- Base collection of UI widgets
Checkbox, number/text field, dropdown lists, slider, file pickers, etc.
- Markdown & LaTeX
Text & equations & code
- More convenient for researcher than VSCode
Calculator, hypotheses tester, simulator
- Good for teaching
<https://github.com/JuliaPluto/PlutoTeachingTools.jl>



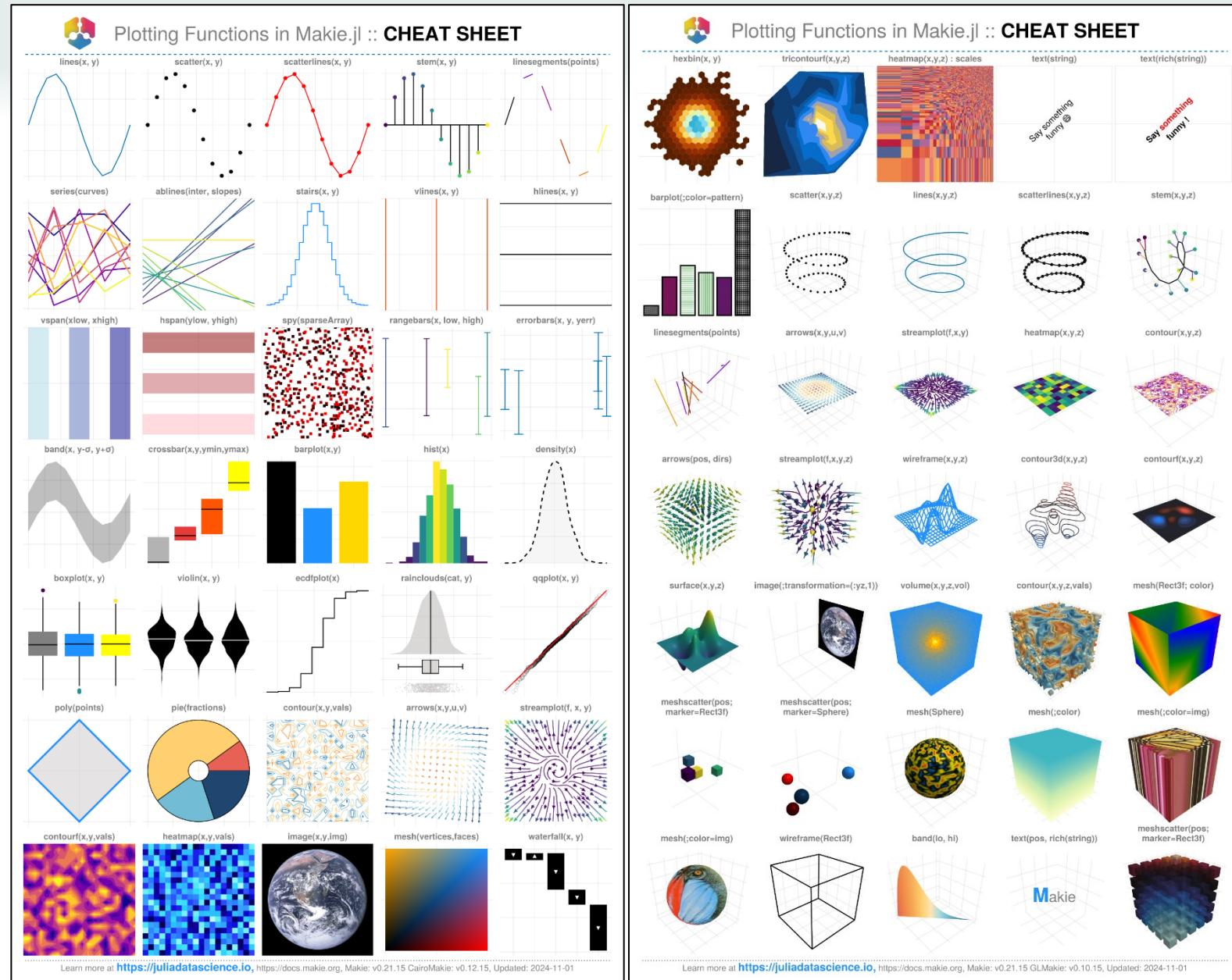
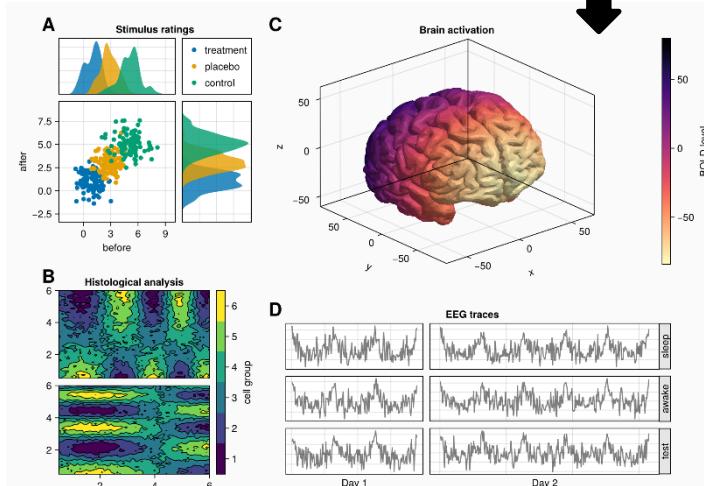
Pluto cells represent functions of many variables defined in other cells, i.e.,

x, y, z
 $f(x, y, z \dots)$
 $g(x, y, z; f \dots)$

You cannot define one variable in two or more cells!

Makie

- Nice-looking plots
- Easy to code
- Clear documentation site
- Rich collection of plots
- Vector graphics for 2D (CairoMakie)
- OpenGL for 3D (GLMakie)
- Complex plot layouts



Julia for teaching - example

- Introduction to Computational Thinking
(<http://computationalthinking.mit.edu>)

- Julia: A Fresh Approach to Computing

- This class uses revolutionary programmable interactivity to combine material from three fields creating an engaging, efficient learning solution to prepare students to be sophisticated and intuitive **thinkers, programmers, and solution providers** for the modern interconnected online world
- Upon completion, students are well trained to be scientific “trilinguals”, seeing and experimenting with mathematics interactively as math is meant to be seen, and ready to participate and contribute to **open source development of large projects and ecosystems**

1.2 Abstraction
Homework 1: Images and Arrays
1.3 Automatic Differentiation
1.4 Transformations with Images
Homework 2: Convolutions
1.5 Transformations II: Composability, Linearity and Nonlinearity
1.6 The Newton Method
Homework 3: Structure and language
1.7 Dynamic Programming
1.8 Seam Carving
1.9 Taking Advantage of Structure
Homework 4: Dynamic programming

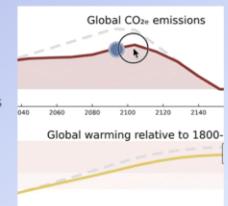
MODULE 2: SOCIAL SCIENCE & DATA SCIENCE

2.1 Principal Component Analysis
2.2 Sampling and Random Variables
Homework 5: Structure
2.3 Modeling with Stochastic Simulation
Homework 6: Probability distributions
2.4 Random Variables as Types
2.5 Random Walks
2.6 Random Walks II
2.7 Discrete and Continuous
Homework 7: Epidemic modeling I
2.8 Linear Model, Data Science, & Simulations
2.9 Optimization
Homework 8: Epidemic modeling II

Highlights

Real-world problems

We will take applications such as climate change and show how you can participate in the big open source community looking to find solutions to challenging problems with exposure to github and parallel computing.



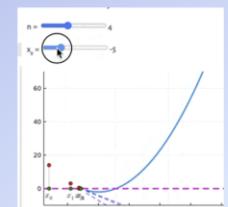
Corgi in the washing machine

You will learn mathematical ideas by immersion into the mathematical process, performing experiments, seeing the connections, and seeing just how much fun math can be.



Revolutionary interactivity

Our course material is built using real code, and instead of a book, we have a series of interactive notebooks. On our website, you can play with sliders, buttons and images to interact with our simulations. You can even go further, and modify and run any code on our website!



Learning Julia

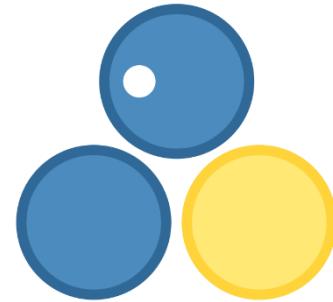
In literature it's not enough to just know the technicalities of grammar. In music it's not enough to learn the scales. The goal is to communicate experiences and emotions. For a computer scientist, it's not enough to write a working program, the program should be **written with beautiful high level abstractions that speak to your audience**. This class will show you how.

Exercise 3: Spheres
Now that we know how to bounce light around things more interesting. Let's define a Sphere

```
struct Sphere <: Object
    center::Vector{Float64}
    # Radius
    radius::Real
    # Index of refraction
    ior::Real
end
```

Julia and Python – Most popular library

- **PythonCall & JuliaCall** (<http://github.com/JuliaPy/PythonCall.jl>)
 - Bringing Python® and Julia together in seamless harmony!



Calling Python from Julia

The figure displays a pairplot for the Iris dataset, showing the distribution of each variable and the pairwise relationships between Sepal.Length, Sepal.Width, Petal.Length, and Petal.Width across three species: setosa, virginica, and versicolor.

Calling Julia from Python

```
[1]: import numpy as np, matplotlib.pyplot as plt
from juliacall import Main as jl, convert as jlconvert

[2]: def batch(n):
    x = np.sort(np.random.uniform(-1, 1, (1,n)))
    y = np.sin(x*10) + np.random.normal(0, 0.1, (1,n))
    return x, y

[3]: jl.seval("using Flux")
model = jl.Chain(
    jl.Dense(1, 10, jl.relu),
    jl.Dense(10, 10, jl.relu),
    jl.Dense(10, 10, jl.relu),
    jl.Dense(10, 1),
)
loss = jl.seval("m -> (x, y) -> Flux.Losses.mse(m(x), y)")(model)

[4]: jl.Flux.train_b(
    loss,
    jl.Flux.params(model),
    jlconvert(jl.Vector[jl.Tuple], [batch(100) for _ in range(2000)]),
    jl.ADAM(0.01),
)
)

[5]: x, y = batch(400)
plt.scatter(x[0], y[0], label="truth")
yhat = model(x)
plt.plot(x[0], yhat[0,:], c="k", label="model")
plt.legend()
print("loss =", loss(x,y))

loss = 0.019229781082918906
```

Julia and Python – Bridging worlds

- Osborne, Ianna; Pivarski, Jim; Ling, Jerry. Bridging Worlds: Achieving Language Interoperability between Julia and Python in Scientific Computing.
arXiv:2404.18170, 2024 (<http://arxiv.org/abs/2404.18170v1>)

- Awkward Array (<http://github.com/JuliaHEP/AwkwardArray.jl>)

- Library for manipulating large-scale arrays of nested, variable-sized data in Python, using array-oriented idioms: like NumPy, but for any JSON-like data
 - Zero-copy share data between the two languages!

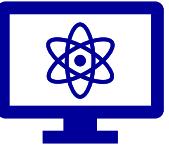


Requirements

- Deployment
 - Redeploy
 - Upgrade
 - Kernels and environments
- User management
 - Quota management
 - Code access (GitLab)
 - Data access (DBMSs)
 - + Data sharing
 - ETL access
- Scheduling of orchestration tasks
- Knowledge base for “juniors”
 - Shared code documentation
 - MkDocs (<http://github.com/mkdocs/mkdocs>)
 - PlantUML server (<https://github.com/plantuml/plantuml-server>)
 - ...
- **ML Workflows!**

Deploy examples

- JupyterHub
 - Zero to JupyterHub with Kubernetes (<http://z2jh.jupyter.org>)
 - Nebari – Custom configurations and overrides
(<http://www.nebari.dev/docs/explanations/custom-overrides-configuration/>)
- Marimo
 - Marimo – Deploying (<http://docs.marimo.io/guides/deploying/index.html>) – any ASGI/WSGI server
- Pluto?
 - PlutoSliderServer.jl (<http://github.com/JuliaPluto/PlutoSliderServer.jl>)
 - Pluto – deploy julia-pluto notebook
(<http://gist.github.com/sdeoras/4592c64ca48573a0d7a3aabdc86a235a>)
 - Kubernetes in Julia with Kuber.jl (<http://www.juliabloggers.com/kubernetes-in-julia-with-kuber-jl/>)

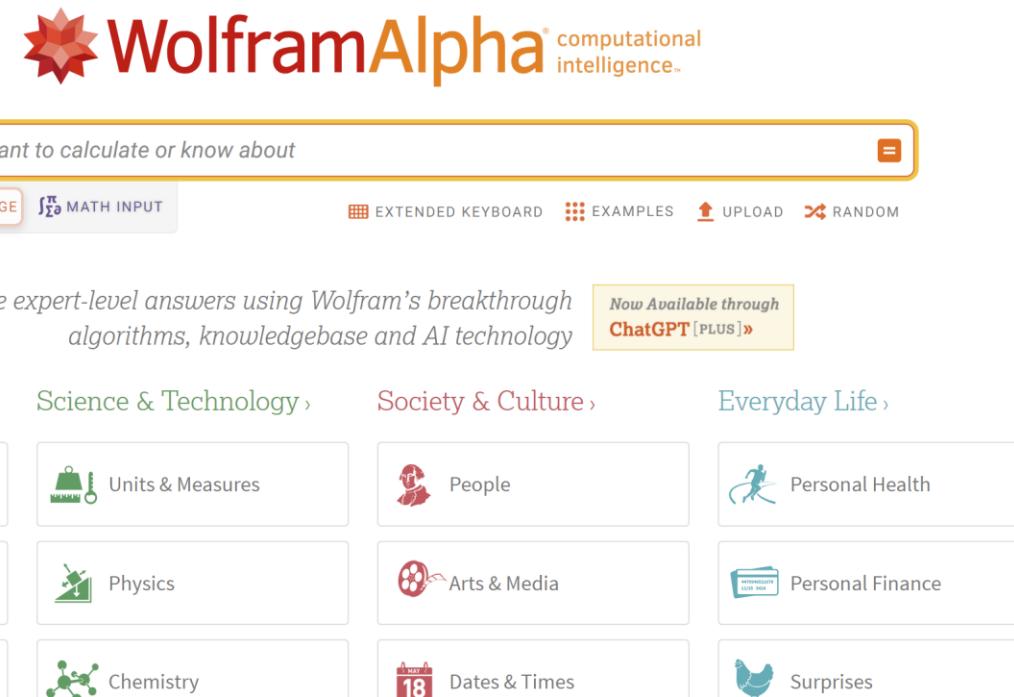


AI next steps?

- ✓ Explicit knowledge as “packed knowledge”
 - ✓ Ontologies + logics
- ✓ LLMs
 - ✓ Reasoning + hybridization
- ✓ New tools for RSE
 - ✓ AI4RSE
- ✓ AI research with RSE

Explicit knowledge and ontologies

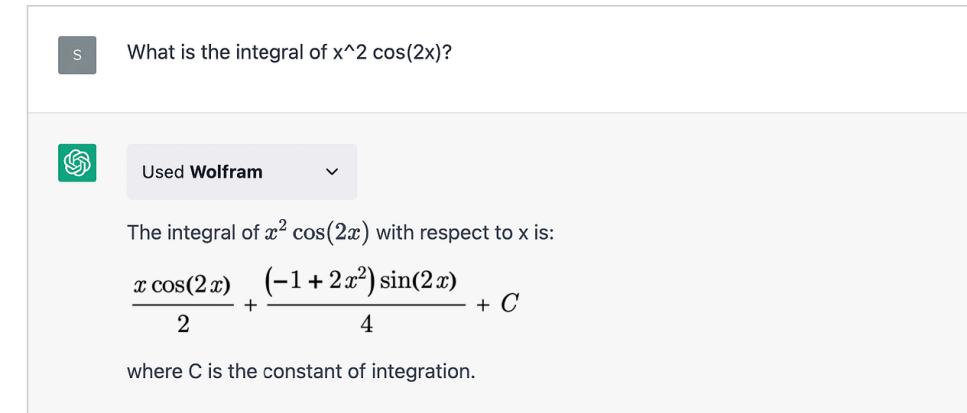
- Wolfram Alpha (<http://www.wolframalpha.com>)
 - Alpha appends Microsoft Bing
 - Announcing the next wave of AI innovation with Microsoft Bing and Edge (<http://blogs.microsoft.com/blog/2023/05/04/announcing-the-next-wave-of-ai-innovation-with-microsoft-bing-and-edge/>)
 - And available in OpenAI ChatGPT



Hybridization of AI tools

- *ChatGPT Gets Its “Wolfram Superpowers”!*
(<http://writings.stephenwolfram.com/2023/03/chatgpt-gets-its-wolfram-superpowers/>)

The screenshot shows a ChatGPT interface. A user asks: "How deep is the ocean at the antipodal point to London?". Below the question, there is a "Used Wolfram" button. The response states: "The antipodal point to London is located at approximately 51° 30' S, 179° 53' E. This point is in the southern Pacific Ocean. The local depth at this location is about 16,699 feet (5,090 meters)."

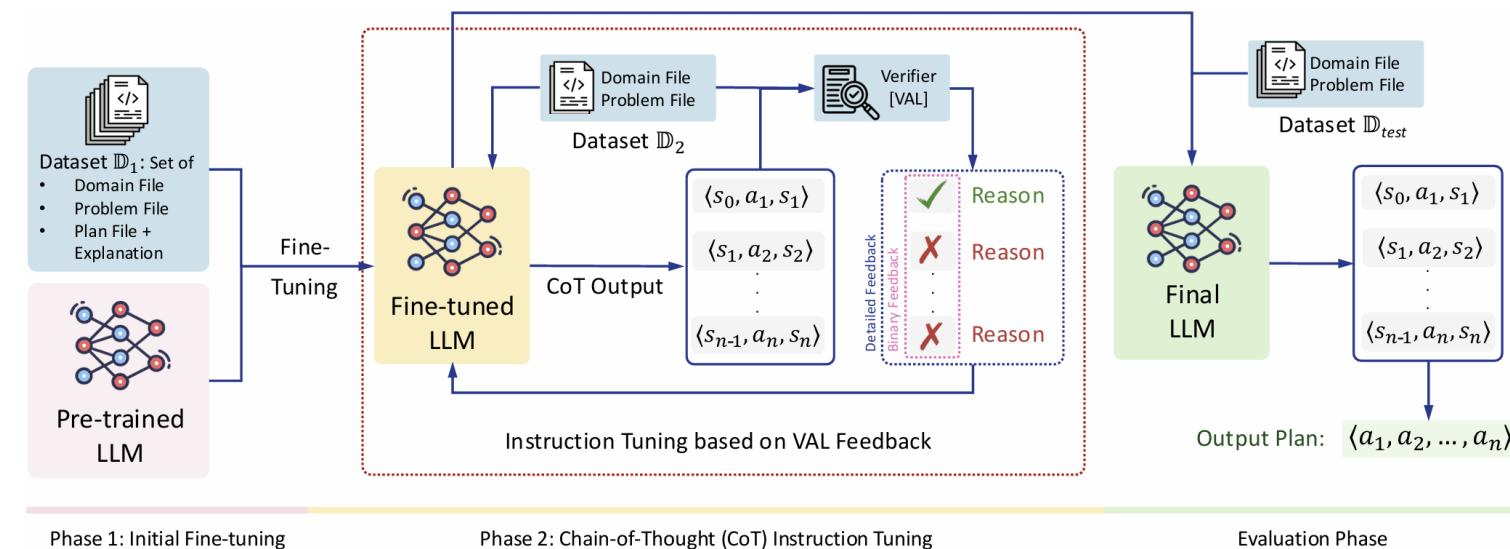


The screenshot shows a ChatGPT interface. A user asks: "What is the integral of $x^2 \cos(2x)$?". Below the question, there is a "Used Wolfram" button. The response provides the integral formula: "The integral of $x^2 \cos(2x)$ with respect to x is:
$$\frac{x \cos(2x)}{2} + \frac{(-1 + 2x^2) \sin(2x)}{4} + C$$
where C is the constant of integration."

- *Improve ChatGPT with Knowledge Graphs*
(http://mlabonne.github.io/blog/posts/Article_ Improve ChatGPT with Knowledge Graphs.html)
- *Neo4j – GenAI, Meet Knowledge Graphs – Completeness of Answers. Every Time*
(<http://neo4j.com/generativeai/>)

Instructions and reasoning

- Verma P., La N., Favier A., Mishra S., Shah J.A. Teaching LLMs to Plan: Logical Chain-of-Thought Instruction Tuning for Symbolic Planning. arXiv:2509.13351, 2025 (<http://arxiv.org/abs/2509.13351>)
 - A novel **instruction tuning framework**, PDDL-Instruct, designed to enhance LLMs' symbolic planning capabilities through logical chain-of-thought reasoning
 - Planning Domain Definition Language (PDDL)



Science problems and correctness

- Feldman M., Karbasi A. Gödel Test: Can Large Language Models Solve Easy Conjectures? arXiv:2509.18383, 2025 (<http://arxiv.org/abs/2509.18383>)
 - 3 from 5 properly formulated scientific problems were solved...
- *Llemma – an open language model for mathematics*
(<http://github.com/EleutherAI/math-lm>)
 - Demo: <https://llemma-demo.github.io>
 - Two years ago!

The screenshot shows a web interface for the Llemma sample explorer. At the top, there's a navigation bar with links for arXiv, Blog, Try the Model (7B) (34B), GitHub, Algebra, Counting and Probability, Geometry, Intermediate Algebra, Number Theory, Prealgebra, Precalculus, GSM8k, OpenCourseWare, MMLU STEM, and SAT. The Algebra tab is selected.

The main content area displays a math problem:

PROBLEM SOURCE: MATH Algebra Level 3

QUESTION: Find the value of $\sqrt{12 - \sqrt{12 - \sqrt{12 - \dots}}}$.

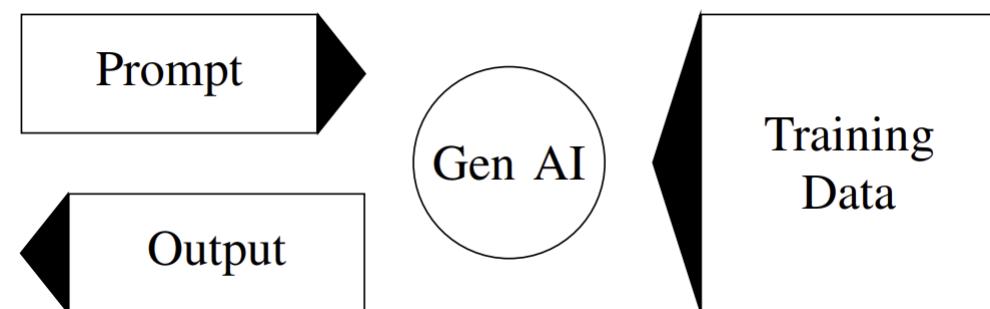
LLEMMA 34B: Answers sampled using self-consistency with $k = 256$. The answer is $x = 3$.

EVALUATION RESULT: Correct

REFERENCE ANSWER: Letting $x = \sqrt{12 - \sqrt{12 - \sqrt{12 - \dots}}}$, we have $x = \sqrt{12 - x}$. Therefore, $x^2 = 12 - x$, so $x^2 + x - 12 = 0$, or $(x + 4)(x - 3) = 0$. Clearly, x must be positive, so $x = 3$.

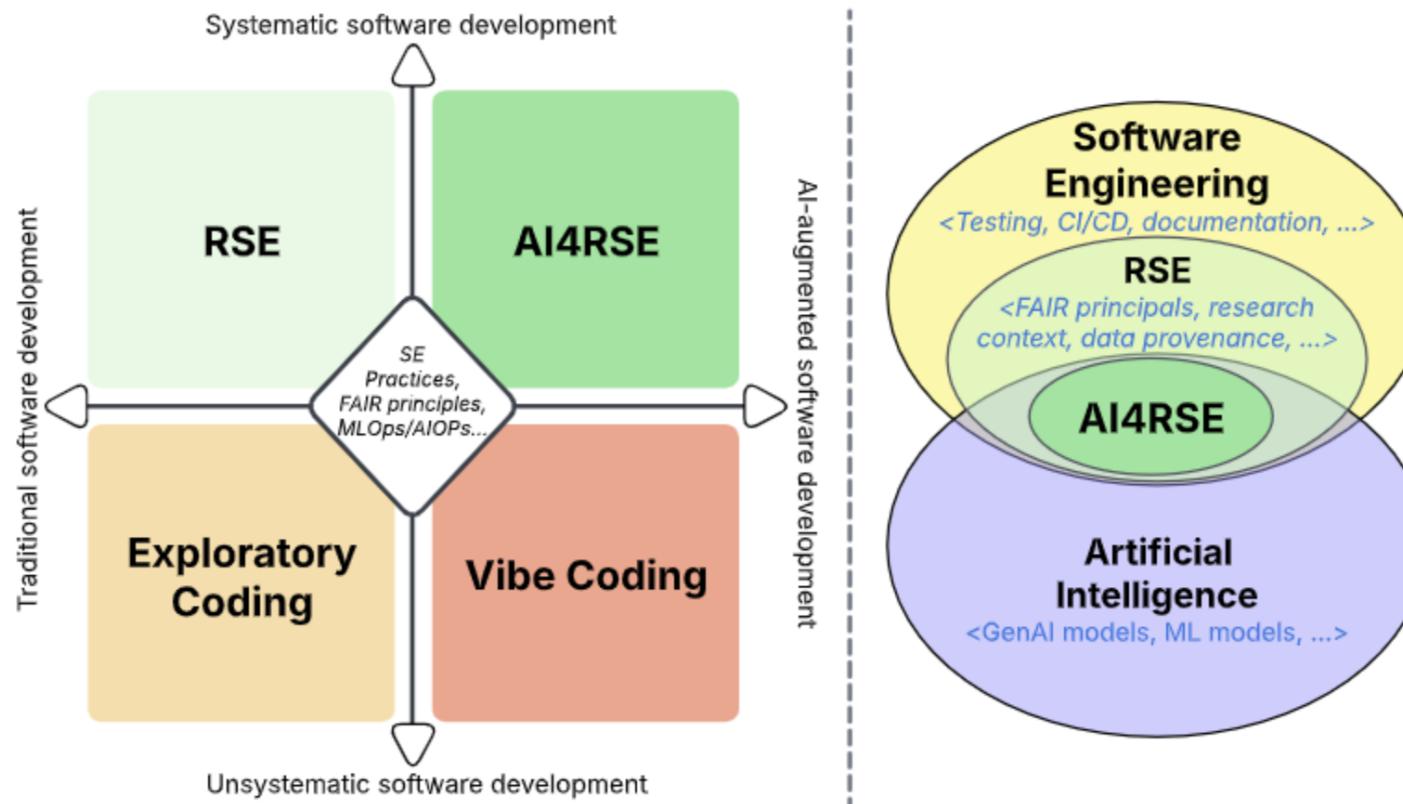
Empirical Software Engineering vs/with RSE

- Treude C., Storey M.-A. [Generative AI and Empirical Software Engineering: A Paradigm Shift](https://arxiv.org/abs/2502.08108v1). arXiv:2502.08108, 2025 ([http://arxiv.org/abs/2502.08108v1](https://arxiv.org/abs/2502.08108v1))
 - New types of data amenable to empirical software engineering research in the era of **generative AI adoption**:
 - Training data
 - Prompts
 - Output

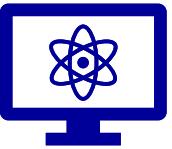


RSE with AI → AI4RSE

- The Software Development Quadrant with two axes:
engineering maturity (unsystematic to systematic) and **AI integration** (traditional to AI-augmented)
- A conceptual Venn diagram illustrating how AI4RSE emerges at the intersection of **Software Engineering**, **Artificial Intelligence**, and **Research Software Engineering**



Advancing Research Software Engineering with AI:
A Research Framework. Automated Software Engineering, 2025, Under review
(<https://www.researchsquare.com/article/rs-7178452/v1>)



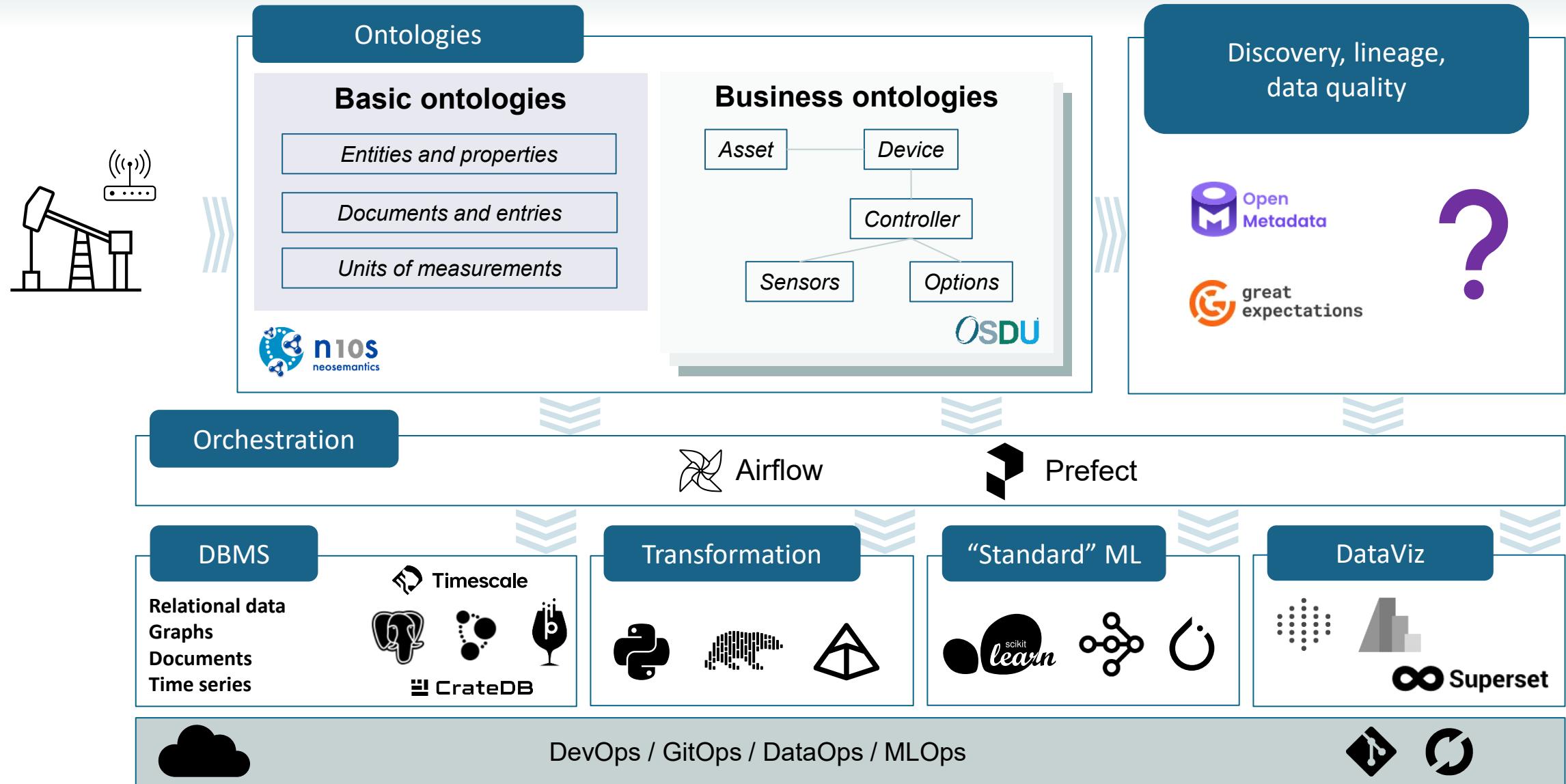
Python monolanguage stack architecture

- ✓ DDD
- ✓ Approaches
- ✓ Layers
- ✓ Python ecosystem – foundation
- ✓ [TBD] IS and personal data

About architecture prerequisites

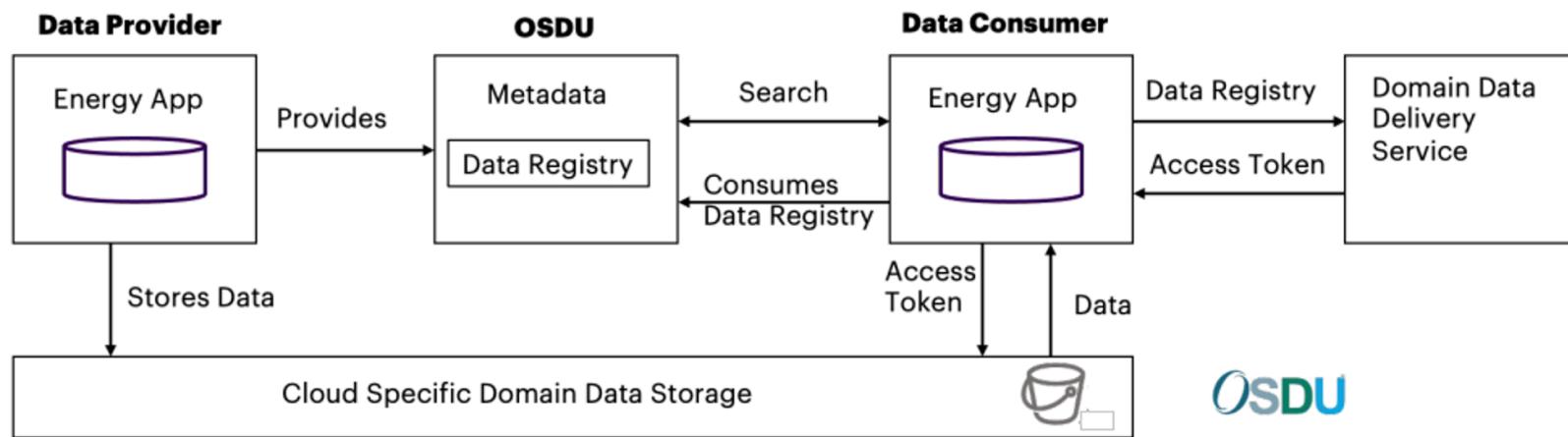
- Data management requirements:
 - Minimalistic
 - Open
 - Open standards and open implementations
 - Durable
 - Knowledge-based (ontology-controlled)
 - Monolanguage
 - Python!
 - With effective data storage, gathering, and transformation
 - Scalable?
 - Secure?

Current minimalistic approach to architecture



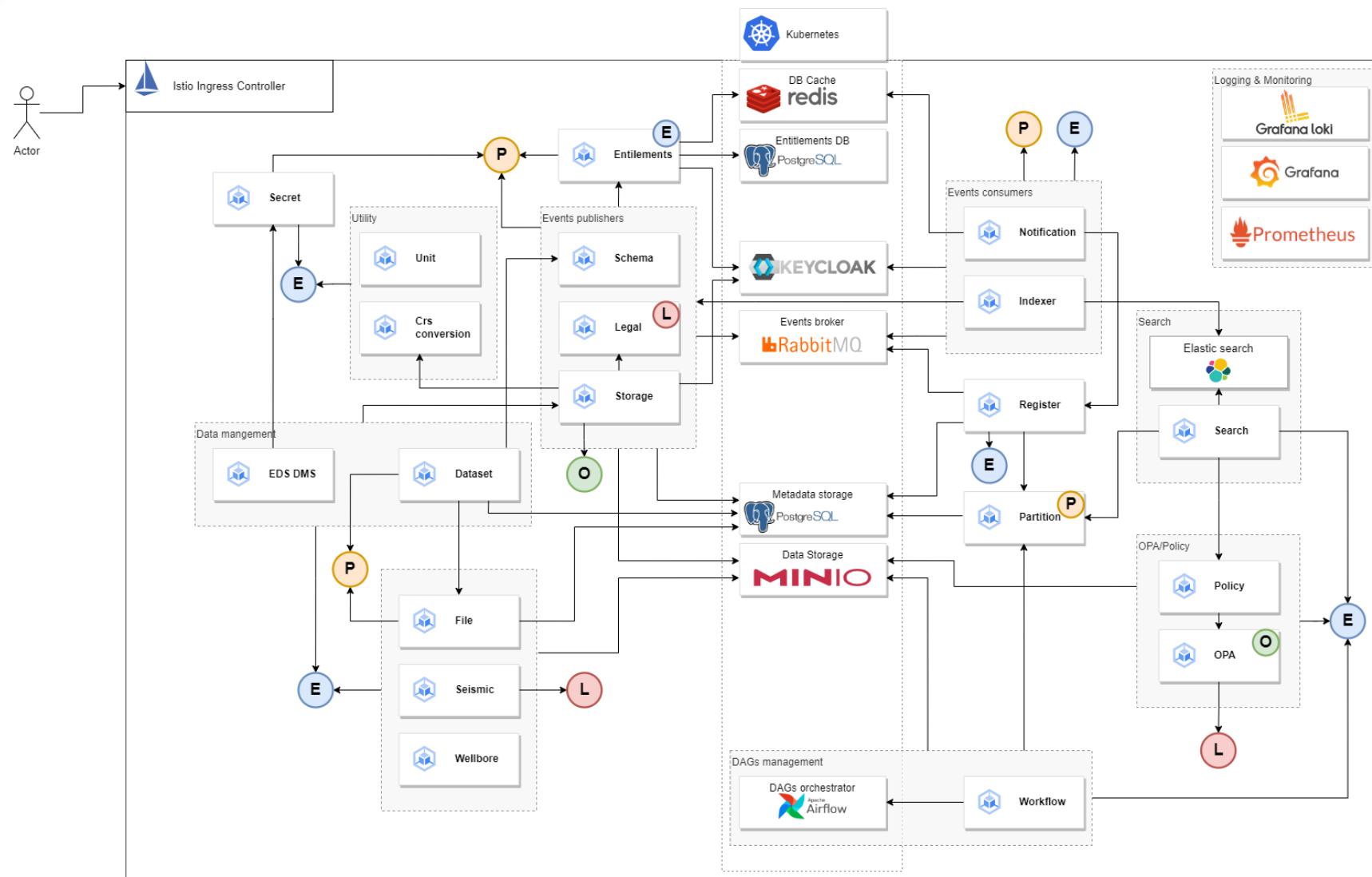
OSDU reference architecture

- Open Subsurface Data Universe (<http://osduforum.org>)
 - OSDU Ontology (<http://github.com/Accenture/OSDU-Ontology>)



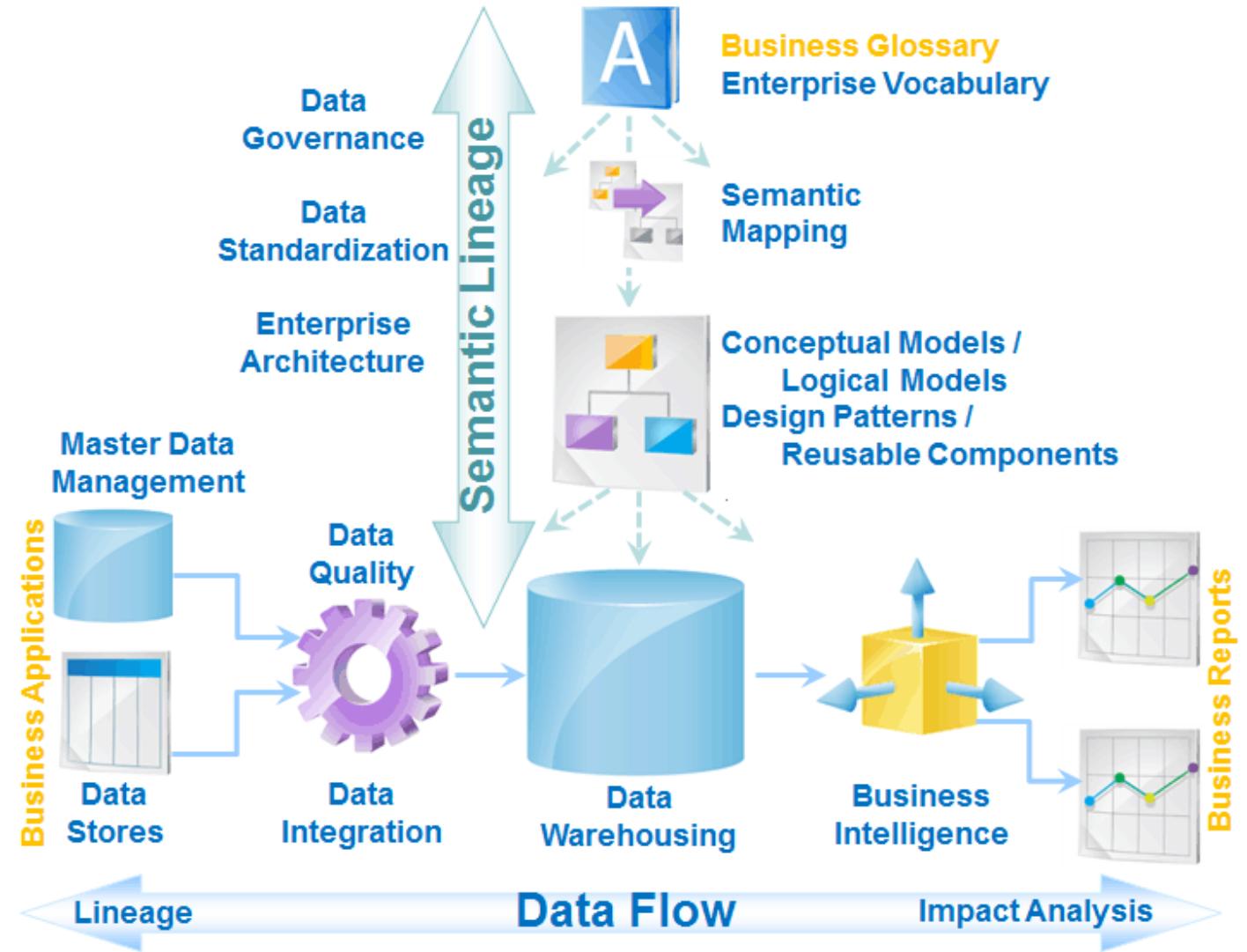
- Open OSDU Deployment Framework
(<http://community.opengroup.org/osdu/platform/deployment-and-operations/infra-gcp-provisioning/>)

Infrastructure of OSDU Baremetal



Example of metadata management

- Meta Integration
(<http://www.metaintegration.net>)
 - Metadata Integration
 - Metadata Harvesting
 - Metadata Management
 - Enterprise Architecture
 - Data Governance

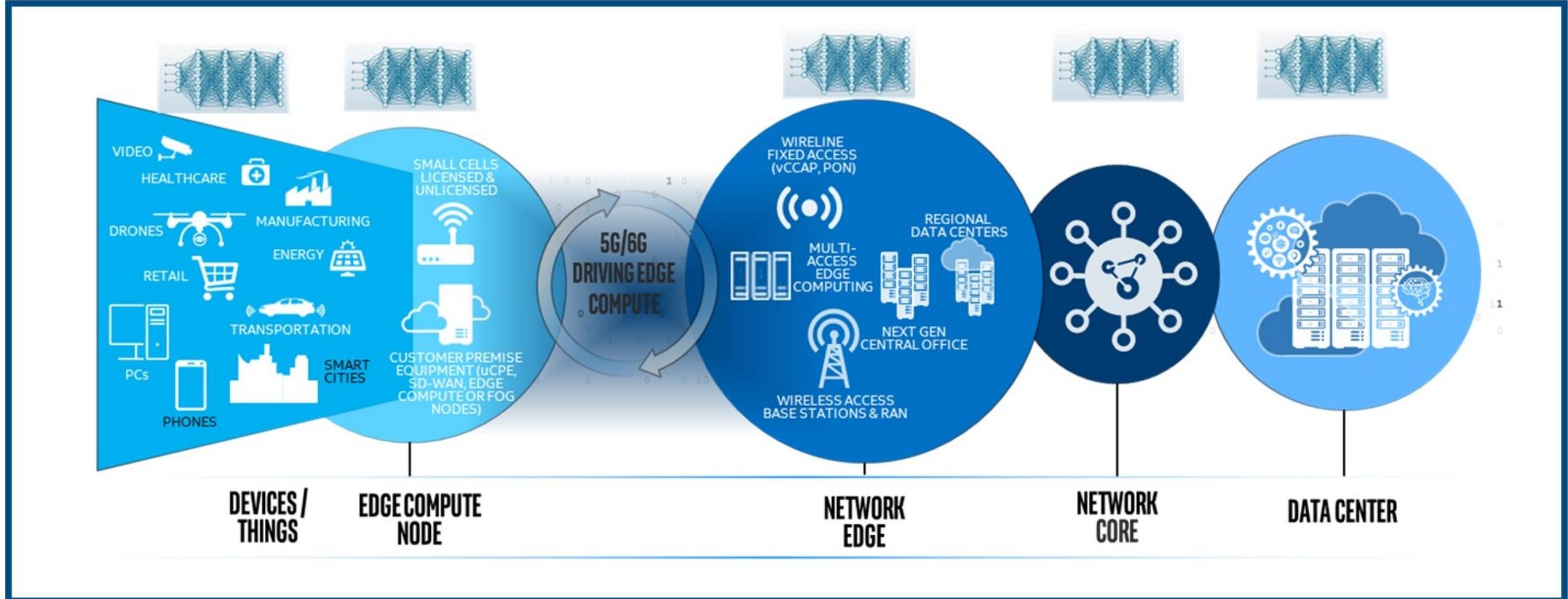


Platform services and PaaS

- Services!
 - Wikipedia – Platform as a service (https://en.wikipedia.org/wiki/Platform_as_a_service)
 - The NIST Definition of Cloud Computing (SP 800-145)
(<https://csrc.nist.gov/publications/detail/sp/800-145/final>)
 - Software, Platform, Infrastructure Model (**SPI** Model)
(<https://www.techopedia.com/definition/14019/software-platform-infrastructure-model-spi-model>)
 - 9 open source PaaS options developers should know in 2024
(<http://www.techtarget.com/searchcloudcomputing/feature/6-open-source-PaaS-options-developers-should-know>)
 - Intel Open End-to-End Platform Infrastructure
(<https://www.intel.com/content/www/us/en/research/news/open-platform-infrastructure.html>)
 - Intel Intent Driven Orchestration - IDO (<http://github.com/intel/intent-driven-orchestration>)



Intel Open End-to-End Platform Infrastructure

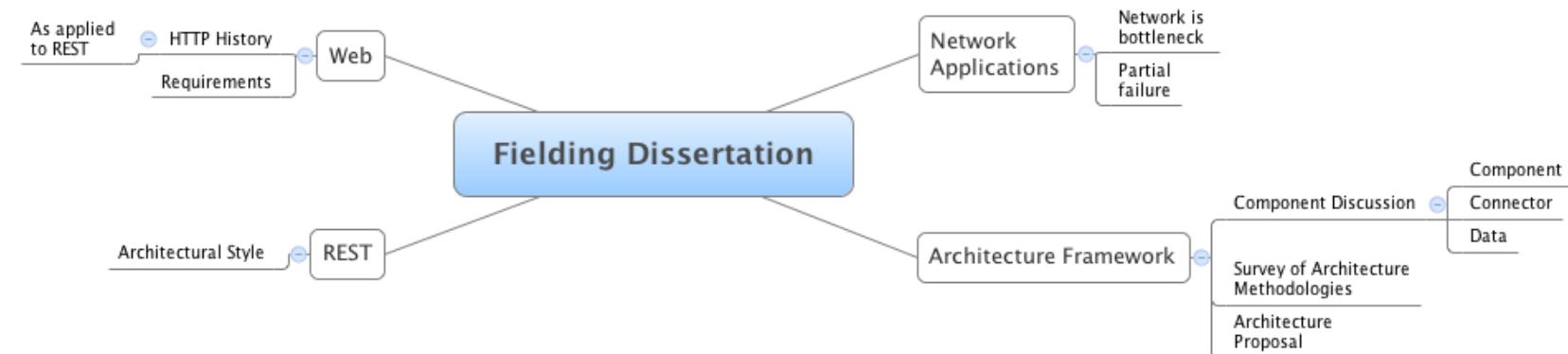


The wide sense of “platforms”

- Simon P., Joel M. The Age of the Platform: How Amazon, Apple, Facebook, and Google Have Redefined Business. Motion Publishing, 2011. 312 p.
- Platform Hunt. The Guide to Platform Business Models
(<https://medium.com/platform-hunt/the-platform-business-model-guide-6e86f5c5679b>)
- Platform Hunt. The 9 Types of Software Platforms. 2016
(<http://medium.com/platform-hunt/the-8-types-of-software-platforms-473c74f4536a>)
- TECHNOLOGY VISION 2016 - PLATFORM ECONOMY
(<https://www.accenture.com/us-en/insight-digital-platform-economy>)

REST – Beginning

- Roy Thomas Fielding – **Representational State Transfer (REST)**
 - *Fielding R.T. Architectural Styles and the Design of Network-based Software Architectures. University of California, Irvine, 2000* (<http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>)
 - *Chapter 5. Representational State Transfer (REST)*
(http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm)
 - Resource oriented Internet services
 - The basis in terms of data access: HTTP + JSON + MIME
- Principles:
 - *Client-Server*
 - *Stateless*
 - *Cache*
 - *Uniform Interface*
 - *Layered System*
 - *Code-On-Demand*



© 2016, W. Watson, <http://blog.vulk.coop/breaking-down-the-rest-dissertation-part-1-overview/>

Modern Python ecosystem

- **Python** (<http://www.python.org>) – a programming language that lets you work quickly and integrate systems more effectively
 - Ver>=3.12 (<http://docs.python.org/3.12/whatsnew/3.12.html>)
- **Pydantic** (<http://pydantic.dev>) – the most widely used data validation library for Python
 - Ver>=2.10 (<http://docs.pydantic.dev/latest/migration/>)
 - **autodoc_pydantic** (<http://autodoc-pydantic.readthedocs.io>) – Seamlessly integrate pydantic models in your Sphinx documentation
 - Entity-Relationship Diagram (<http://autodoc-pydantic.readthedocs.io/en/stable/users/examples.html#entity-relationship-diagram>)
- **Polars** (<http://pola.rs>) – dataframes for the new era
 - Ver>=1.29
 - **polars_ds_extension** (http://github.com/abstractqqq/polars_ds_extension) – Polars for Data Science
- **SQLAlchemy** (<http://www.sqlalchemy.org>) – the Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL
 - Ver>=2.0.30 (http://docs.sqlalchemy.org/en/20/changelog/migration_20.html)
- **Psycopg** (<http://www.psycopg.org>) – the most used, reliable and feature-rich PostgreSQL adapter for Python
 - Ver>=3.2 (<http://www.psycopg.org/psycopg3/>)



Python and Rust – battle for speed and quality

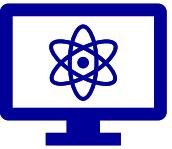
- PyO3 (<http://github.com/PyO3/pyo3>)
 - Rust bindings for Python, including tools for creating native Python extension modules
 - Running and interacting with Python code from a Rust binary is also supported
 - Ver>=0.24
- Other methods of performance rising:
 - CPytnon
 - Numba
 - Taichi
 - ...
 - Mojo?

Revolution in package management

- **uv** (<https://github.com/astral-sh/uv>)

- 🚀 A single tool to replace pip, pip-tools, pipx, poetry, pyenv, twine, virtualenv, and more.
- ⚡ 10-100x faster than pip
- 🐍 Installs and manages Python versions
- 🛠 Runs and installs Python applications
- ⭐ Runs single-file scripts, with support for inline dependency metadata
- 📁 Provides comprehensive project management, with a universal lockfile
- 🔧 Includes a pip-compatible interface for a performance boost with a familiar CLI
- 🏢 Supports Cargo-style workspaces for scalable projects
- 💾 Disk-space efficient, with a global cache for dependency deduplication
- 🔩 Installable without Rust or Python via curl or pip
- 💻 Supports macOS, Linux, and Windows





Data & knowledge modelling

- ✓ Entities and their life cycle
- ✓ Knowledge bases (only some examples)
- ✓ DDD
 - ✓ Validation
 - ✓ Serialization
- ✓ Services construction

Domains and entities

- **Entity** – a something (physical or ideal object, process, phenomenon) that can be identified (distinguished from another, separated from the environment)
 - It follows from the definition that an **identifier** can always be defined for an entity
 - In a trivial case in a data model, an entity and its identifier **can coincide**
 - See term “**Identity**”! → Identity Life Cycle Management
 - John R. Talburt, Yinle Zhou. Entity Information Life Cycle for Big Data: Master Data Management and Information Integration. Morgan Kaufmann, 2015. 254 p.
- **Application domain, subject area** – a set of interrelated entities describing a certain relatively independent part of reality/mental world and considered in the context of solving a certain class of problems
 - Often just a “domain”, but no one should be confused with other “domain” (for example – domain in relational theory)

Basic ontology modelling

- Ontology engineering
 - Keet M. An Introduction to Ontology Engineering. University of Cape Town, 2020.
v1.5. 289 p. (<http://open.umn.edu/opentextbooks/textbooks/590>)
- ***Protégé*** (<http://protege.stanford.edu>)
 - Tudorache T., Nyulas C., Noy N.F., Musen M.A. WebProtégé: A Collaborative Ontology Editor and Knowledge Acquisition Tool for the Web, Semantic Web, 4(1), 2013, pp. 89–99
(<http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3691821/>)
 - Ver>=5.6
 - Manchester Family History Advanced OWL Tutorial
(<http://owl.cs.manchester.ac.uk/publications/talks-and-tutorials/fhktutorial/>)

Protégé – Classes and Individuals

The screenshot shows the Protégé interface with the 'Annotations' tab selected for the 'gravitational acceleration' class. The left sidebar lists various ontology classes. The main panel displays annotations in three languages: English, Dutch, and German. Annotations include:

- rdfs:label [language: en] gravitational acceleration
- rdfs:label [language: nl] valversnelling
- 'alternative label' [language: en] acceleration of free fall
- 'alternative label' [language: nl] gravitatieveldsterkte
- 'alternative label' [language: nl] zwaarteveldsterkte
- symbol g

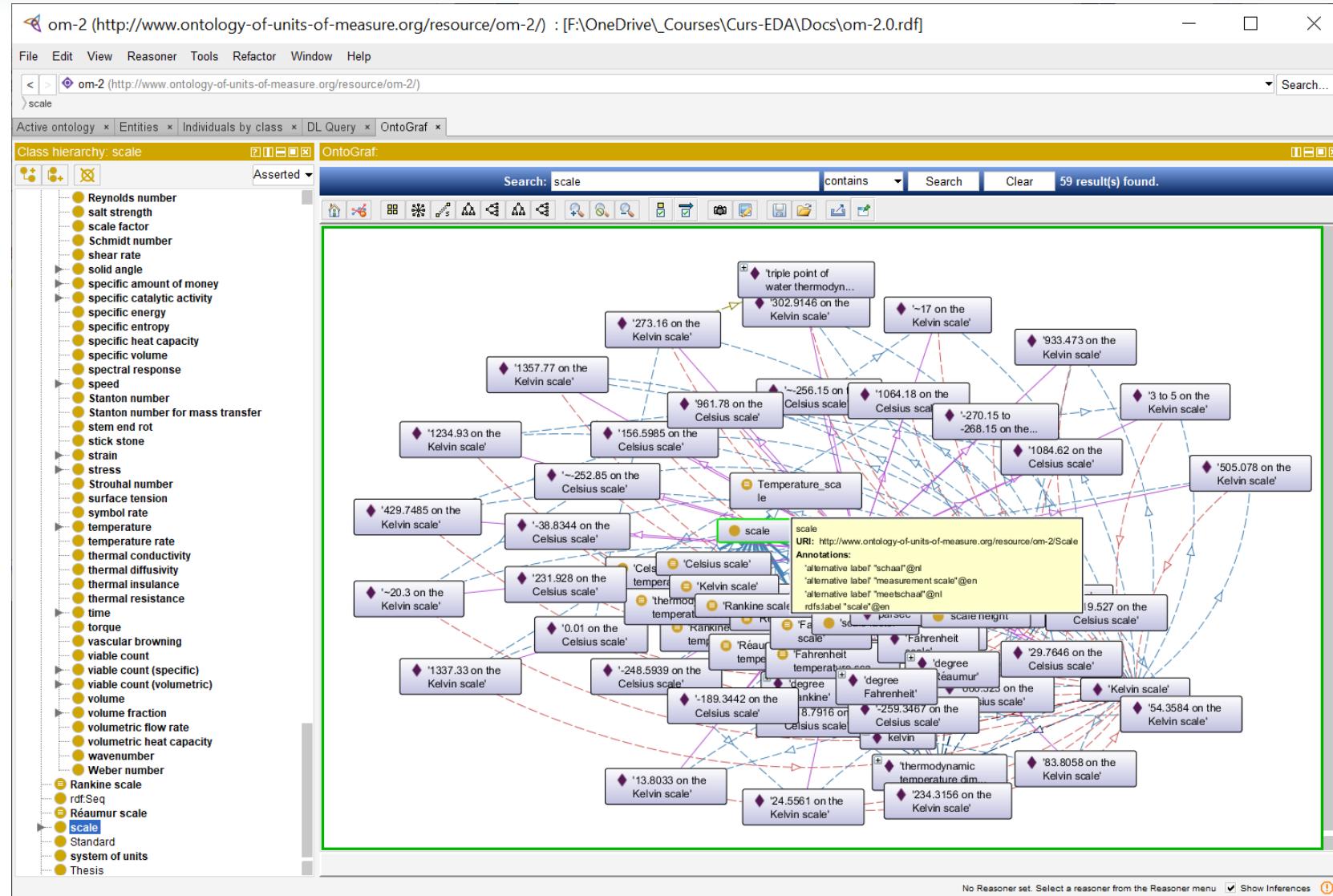
Below the annotations are sections for 'Description' and 'Direct instances'.

The screenshot shows the Protégé interface with the 'Annotations' tab selected for the 'International System of Units' class. The left sidebar lists various ontology classes. The main panel displays annotations in English, Chinese, and Dutch. Annotations include:

- rdfs:label [language: en] International System of Units
- rdfs:label [language: zh] 国际单位制
- rdfs:label [language: nl] Internationale Stelsel van Eenheden
- abbreviation SI

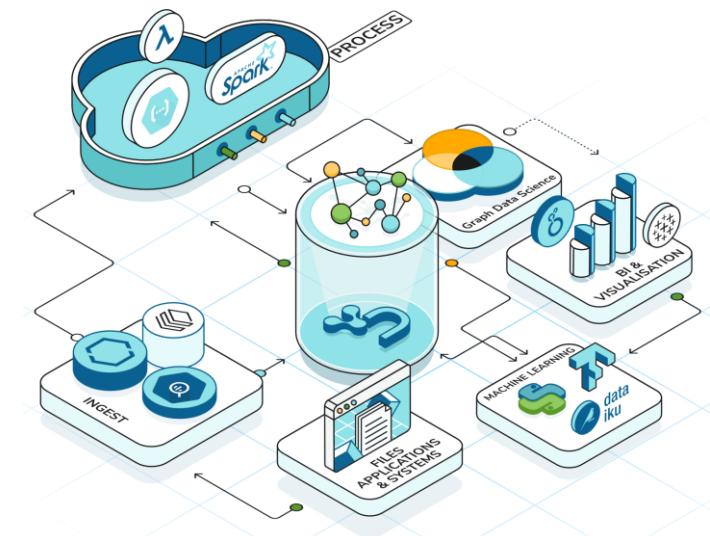
Below the annotations are sections for 'Description', 'Object property assertions', and 'Property assertions'.

Protégé - OntoGraf



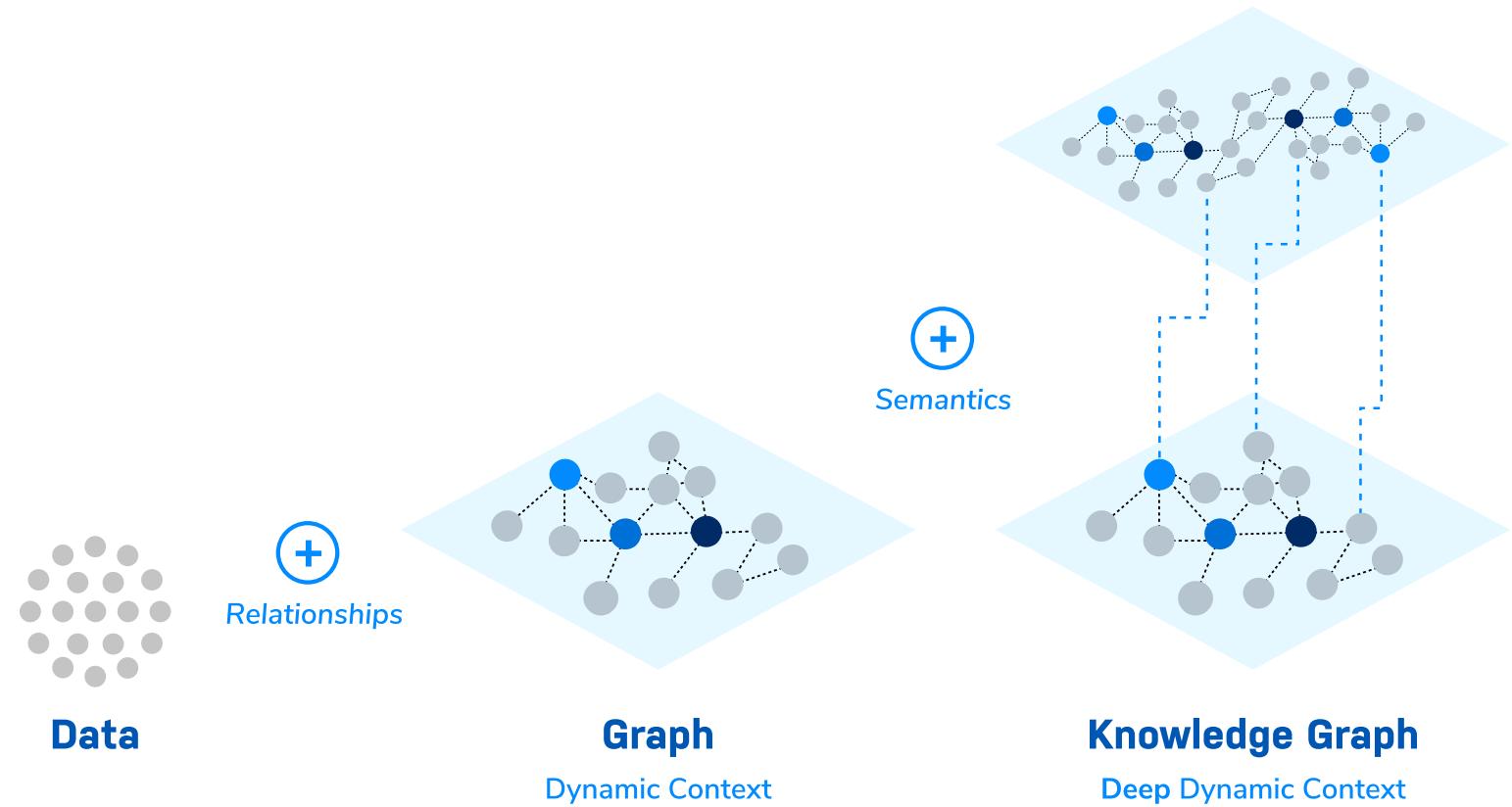
Most popular open graph DBMS

- **Neo4j Graph Database** (<http://neo4j.com/product/neo4j-graph-database/>)
 - **Enterprise Edition** for *scalability, availability, fine-grained access controls!*
- + **Neo4j Graph Data Science** (<http://neo4j.com/product/graph-data-science/>)
 - *Analytics and machine learning (ML) solution that analyzes relationships in data to improve predictions and discover insights*



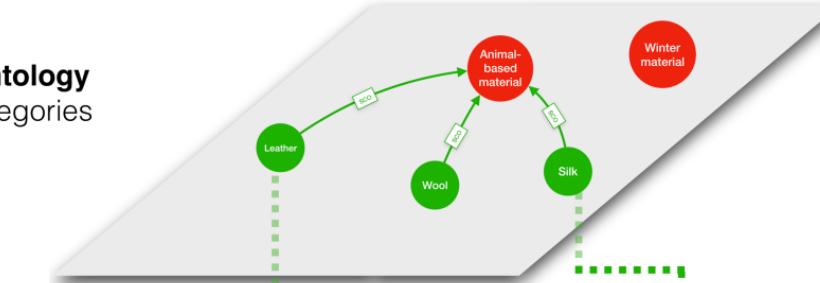
Semantic networks and knowledge graphs

- Formalization, interpretation, visualization, storage
 - Neo4J WP – Knowledge Graphs (<http://neo4j.com/use-cases/knowledge-graph/>)

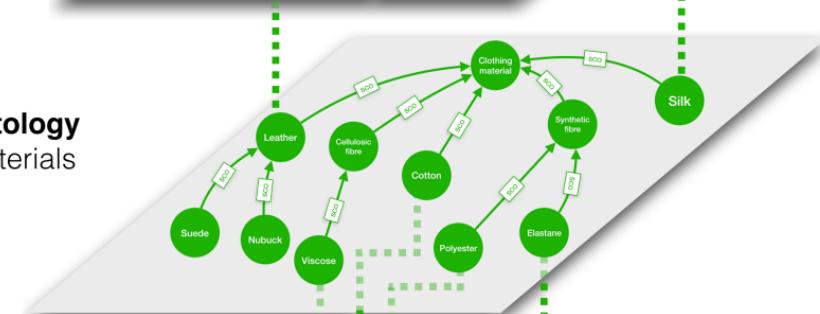


Example from QuickGraph#9

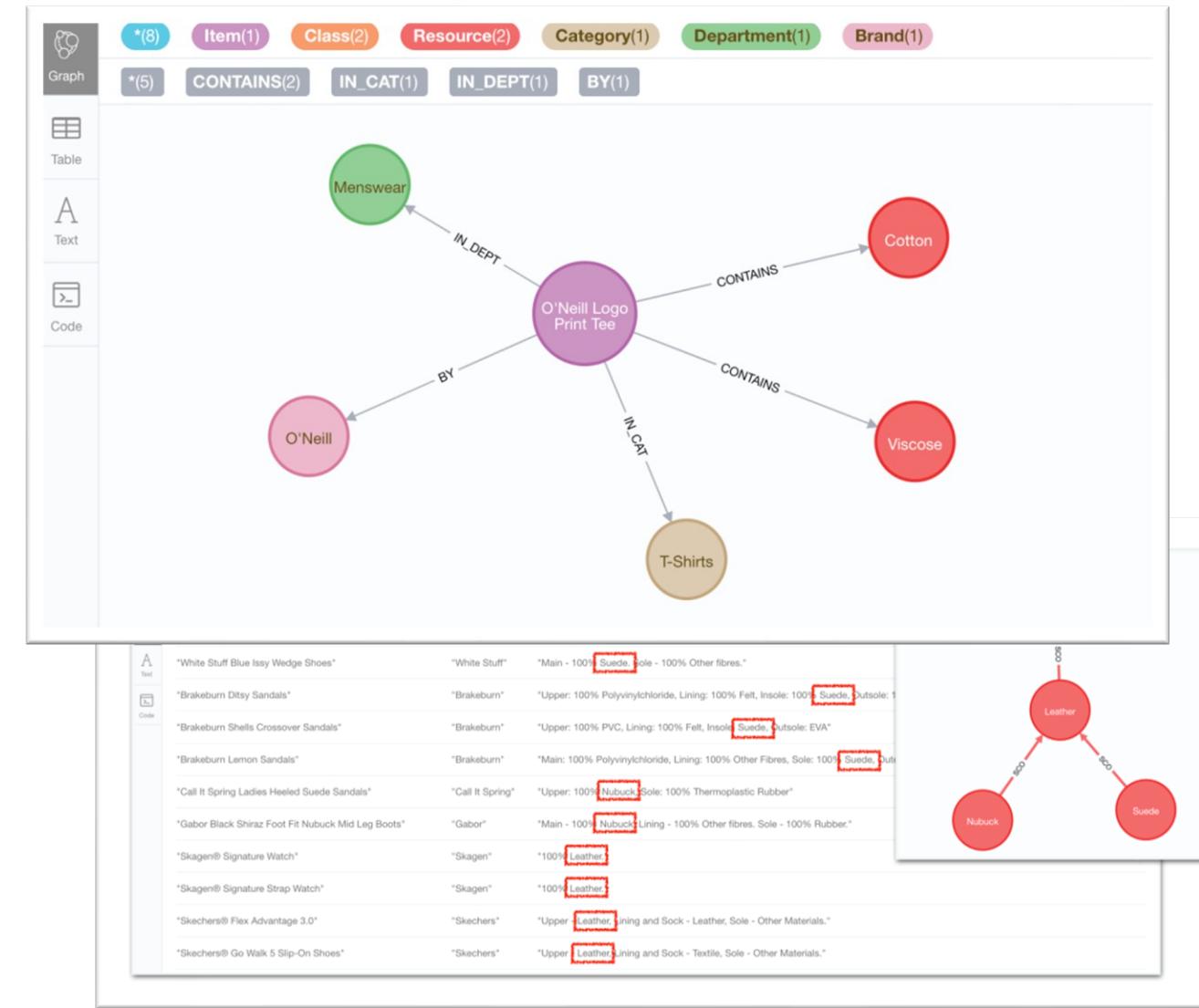
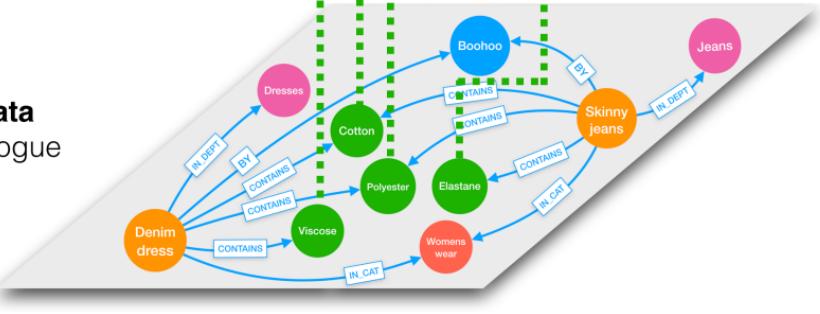
Custom Ontology
Custom Categories



Domain Ontology
Clothing Materials



Instance Data
Product Catalogue



Neo4j Neosemantics

- ***neosemantics (n10s): Neo4j RDF & Semantics toolkit***
[\(http://neo4j.com/labs/neosemantics/\)](http://neo4j.com/labs/neosemantics/)
 - *Plugin that enables the use of RDF and its associated vocabularies like (OWL, RDFS, SKOS and others) in Neo4j*
- Barrasa J. QuickGraph#9 The fashion Knowledge Graph.
Inferencing with Ontologies in Neo4j. 2019
[\(http://jbarrasa.com/2019/11/25/quickgraph9-the-fashion-knowledge-graph-inferencing-with-ontologies-in-neo4j/\)](http://jbarrasa.com/2019/11/25/quickgraph9-the-fashion-knowledge-graph-inferencing-with-ontologies-in-neo4j/)



Metadata management

- **OpenMetadata** (<http://open-metadata.org>)
 - A Single Place to **Discover**, Collaborate and get your Data Right
 - Ver>=1.4
- Business Glossary enhancements:
 - **Custom Properties**: Custom Properties enable teams to enrich data asset metadata by adding additional attributes. Previously, we supported primitive types such as text, integer, and string. In this release, we've expanded the available types to include Enums, **Entity References**, Date, Time, and more
 - These new types allow users to **link other assets** within OpenMetadata or create Enums, providing greater flexibility and depth in metadata management



Validation for OpenMetadata

- **GX – Great Expectations**

(https://github.com/great-expectations/great_expectations)

- A data quality platform designed by and for data engineers
- Ver>=0.18
 - The next step for GX OSS: 1.0 (<http://greatexpectations.io/blog/the-next-step-for-gx-oss-1-0/>)



Data modelling

- **Model** – a simplified description of the system that formalizes information about the system important for modeling purposes and sufficient for solving tasks arising from these purposes
 - If we want to formalize a model, we need a [metamodel](#)!
- **Data model** – model that organizes elements of data and standardizes how they relate to one another for creation, accumulation, processing and management of data
- **Conceptual data model** – data model that formalizes the subject area model regardless of data representation in particular IS
 - Example – entity–relationship model (ER model)
- **Logical data model** – data model that formalizes the subject area model in the form of abstract data structures considering the capabilities of database/knowledge management systems
 - Example – relational model
- **Physical data model** – data model that formalizes the subject area model in the form of specific data structures considering the computational model, system software and hardware
 - Example – direct access files in external memory indexed with B+-trees to access them

Pydantic – common component in data pipelines

- **Pydantic** (<http://pydantic.dev>)

- Ver 2.7+
- V2 is based on Rust
- Required, optional, and nullable fields in V2
(<http://docs.pydantic.dev/latest/migration/#required-optional-and-nullable-fields>)



SQLAlchemy – most popular DAC and ORM

- **SQLAlchemy** (<http://www.sqlalchemy.org>)

- The main goal of SQLAlchemy is to change the way you think about databases and SQL!
- Its object-relational mapper (ORM) is an optional component that provides the data mapper pattern, where classes can be mapped to the database in **open ended**, multiple ways – allowing the object model and database schema to develop in a cleanly decoupled way from the beginning
- Ver 2.1



Integration of Python and SQL

- **SQLModel** (<http://sqlmodel.tiangolo.com>)
 - SQLModel is a library for interacting with SQL databases from Python code, with Python objects
 - It is designed to be intuitive, easy to use, highly compatible, and robust
 - FastAPI integration



Other modern ORM as SQL builder

- **Piccolo** (<http://piccolo-orm.com>)

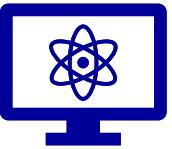
- Powerful async ORM
- Query builder
- Admin GUI
- DBMS dialects:
 - PostgreSQL
 - SQLite
 - CockroachDB

```
# Select:  
await Band.select(  
    Band.name  
)  
.where(  
    Band.popularity > 100  
)  
  
# Join:  
await Band.select(  
    Band.name,  
    Band.manager.name  
)  
  
# Delete:  
await Band.delete().where(  
    Band.popularity < 1000  
)  
  
# Update:  
await Band.update({Band.popularity: 10000}).where(  
    Band.name == 'Pythonistas'  
)
```

Fresh WebAPI ecosystem

- **Litestar** (<http://litestar.dev>)
 - The powerful, lightweight and flexible ASGI framework
 - Migrating to Litestar (<https://docs.litestar.dev/latest/migration/index.html>)





Fundamental data representation checklist

- ✓ Numbers, date/time values, strings, identifiers
- ✓ For binary and textual representation
- ✓ The foundation of 5R

Checklist for binary data representation

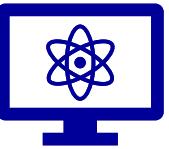
1. Integer numbers — 64 bits = 8 bytes according to **IEEE754**
2. Real numbers — double precision floating point according to **IEEE754**
 - a. Comment: Sources: IEEE 754-2019 – IEEE Standard for Floating-Point Arithmetic (<http://standards.ieee.org/content/ieee-standards/en/standard/754-2019.html>)
3. Date/Time — support all capabilities of **ISO 8601** (conversion to UTC and time zones)
 - a. Comment 1: Interesting choice – Temporenc (<http://temporenc.org>)
 - b. Comment 2: If you select unix time stamp you should use 64 bit!!! See The Year 2038 problem and TimeStamp Converter (<http://www.unixtimestamp.com>)
4. Strings — **UTF-8**
 - a. Comment 1: sources: Section 3.9 in **The Unicode® Standard** (<http://www.unicode.org/versions/Unicode15.0.0/ch03.pdf#G7404>) and former Appendix D in **ISO 10646**, which became Section 9.1 (<http://unicode.org/L2/L2010/10038-fcd10646-main.pdf>)
 - b. Comment 2: see also RFC 3629 (<http://tools.ietf.org/html/rfc3629>)
 - c. Comment 3: you need to consider the requirements for **BOM** (Byte Order Mark)!
5. Unique identifiers — specific unique 128-bit integer value: GUID (Globally Unique Identifier) or UUID (Universally Unique Identifier)
 - a. Comment 1: Universally Unique IDentifiers (UUIDs) **RFC 9562** (<http://datatracker.ietf.org/doc/rfc9562/>) – obsoletes RFC 4122

Checklist for textual data representation (1)

1. **Base encoding — UTF-8**
 - a. Comment 1: sources: Section 3.9 в **The Unicode® Standard** (<http://www.unicode.org/versions/Unicode15.0.0/ch03.pdf#G7404>) and former Appendix D in **ISO 10646**, which became Section 9.1 (<http://unicode.org/L2/L2010/10038-fcd10646-main.pdf>)
 - b. Comment 2: see also **RFC 3629** (<http://tools.ietf.org/html/rfc3629>)
 - c. Comment 3: you need to consider the requirements for BOM (Byte Order Mark)!
2. **Integer numbers** — preferably of arbitrary length or at least 64 bits = 8 bytes
3. **Real numbers** — minimum double precision floating point according to **IEEE754**
 - a. Comment 1: sources: **RFC 6340** (<http://tools.ietf.org/html/rfc6340>)
 - b. Comment 1: if you need **accurate conversion**, consider using Hexadecimal floating literals for C++ (<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2016/p0245r0.html>)
 - c. Comment 3: Check **country/region differences** in formatting: decimal separation, number grouping and separation, etc. (<http://learn.microsoft.com/en-us/globalization/locale/number-formatting>)
4. **Date/Time** — full format with separators according to **ISO 8601**
 - a. Comment 1: sources: ISO 8601-1:2019 Date and time — Representations for information interchange — Part 1: Basic rules (<http://www.iso.org/standard/70907.html>) и ISO 8601-1:2019 Date and time — Representations for information interchange — Part 2: Extensions (<http://www.iso.org/standard/70908.html>)
 - b. Comment 2: Russian GOST 7.0.64-2018 (ISO 8601:2004) «Система стандартов по информации, библиотечному и издательскому делу. Представление дат и времени. Общие требования» (<http://docs.cntd.ru/document/1200159341>)

Checklist for textual data representation (2)

5. **Strings** — with normal Unicode support and escaping
 - a. Comment: sources: **The Unicode® Standard** (<http://www.unicode.org>)
6. **String escaping** — accurate formal description without “gray areas”
 - a. Preferably **strict schema**
 - i. Comment: this allows you to always distinguish between an empty string (for example: "") and an indeterminate (unspecified) value
 - b. Preferably according to Section 7 **RFC 8259** (<http://tools.ietf.org/html/rfc8259>)
 - i. Comment: RFC 4627 (<http://www.ietf.org/rfc/rfc4627.txt>) and RFC 7159 (<http://tools.ietf.org/html/rfc7159>) was deprecated!
But RFC 7159 is a malfeasance :)
7. **Indeterminate values** — blank (really nothing)
 - a. Comment: see sense of strict escaping
8. **Unique identifiers** — full format of GUID/UUID text string with separators without brackets
 - a. Comment 1: sources: Section 4. **UUID Format** in Universally Unique IDentifiers (UUIDs) **RFC 9562** (<http://datatracker.ietf.org/doc/rfc9562/>) – obsoletes RFC 4122
 - b. Comment 2: example: "f81d4fae-7dec-11d0-a765-00a0c91e6bf6"



Data storages and ETL

- ✓ External data models
 - ✓ Relational data
 - ✓ Graphs
 - ✓ Documents
 - ✓ Time series
- ✓ ORM and efficacy
- ✓ ETL

Ideology and sources – the history of warehousing

- Inmon B. Building the Data Warehouse. Wiley, **1992**.
 - 2005 – 4th edition
- Demarest M. Building the Data Mart. DBMS, 7, **1994**.
- Kimball R. The Data Warehouse Toolkit. Wiley, **1996**.
 - 2013 – 3rd editon
- Inmon B. Data Mart Does Not Equal Data Warehouse. DM Review, **1998**.
- **Holy war: Inmon vs Kimball**
- Inmon W.H., Strauss D., Neushloss G. DW 2.0 The Architecture for the Next Generation of Data Warehousing. Morgan Kaufmann, 2008.

DB-Engines Ranking

- DBMS Ranking
(<http://db-engines.com>)

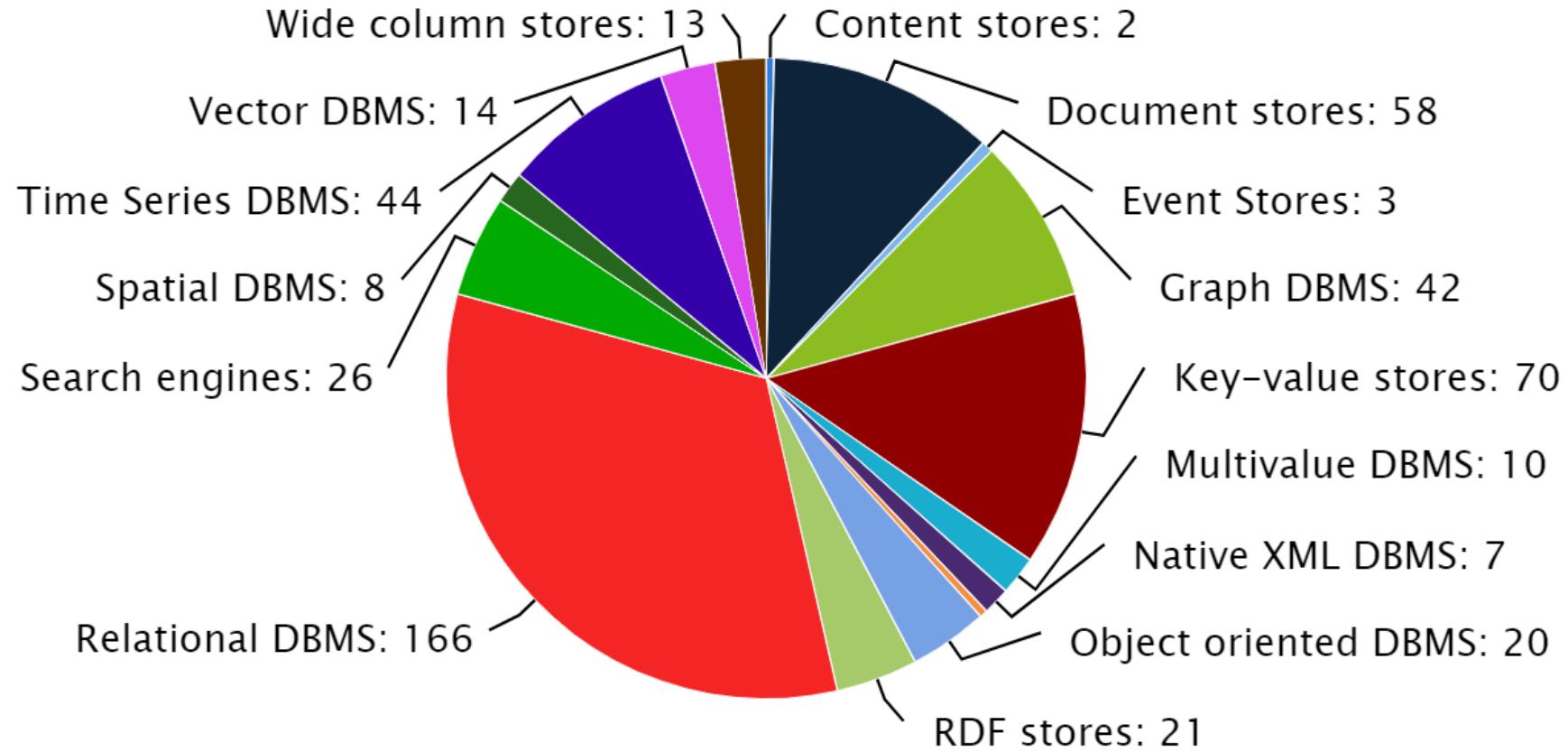
- 405 DMBS!

- Relational DBMS
- Key-value stores
- Document stores
- Time Series DBMS
- Graph DBMS
- Search engines
- RDF stores
- Object oriented DBMS
- Vector DBMS
- Wide column stores
- Multivalue DBMS
- Spatial DBMS
- Native XML DBMS
- Event Stores
- Content stores
- Navigational DBMS

| Rank | Jun 2024 | May 2024 | Jun 2023 | DBMS | Database Model | Score | | |
|------|----------|----------|---|--|--|----------|----------|----------|
| | | | | | | Jun 2024 | May 2024 | Jun 2023 |
| 1. | 1. | 1. | 1. | Oracle + | Relational, Multi-model i | 1244.08 | +7.79 | +12.61 |
| 2. | 2. | 2. | 2. | MySQL + | Relational, Multi-model i | 1061.34 | -22.39 | -102.59 |
| 3. | 3. | 3. | 3. | Microsoft SQL Server + | Relational, Multi-model i | 821.56 | -2.73 | -108.50 |
| 4. | 4. | 4. | 4. | PostgreSQL + | Relational, Multi-model i | 636.25 | -9.30 | +23.43 |
| 5. | 5. | 5. | 5. | MongoDB + | Document, Multi-model i | 421.08 | -0.58 | -4.29 |
| 6. | 6. | 6. | 6. | Redis + | Key-value, Multi-model i | 155.94 | -1.86 | -11.41 |
| 7. | 7. | ↑ 8. | Elasticsearch | | Search engine, Multi-model i | 132.83 | -2.52 | -10.92 |
| 8. | ↑ 9. | ↑ 11. | Snowflake + | | Relational | 130.36 | +9.03 | +16.23 |
| 9. | ↓ 8. | ↓ 7. | IBM Db2 | | Relational, Multi-model i | 125.90 | -2.56 | -18.99 |
| 10. | 10. | 10. | SQLite + | | Relational | 111.41 | -2.91 | -19.81 |
| 11. | 11. | ↓ 9. | Microsoft Access | | Relational | 101.16 | -3.75 | -33.29 |
| 12. | 12. | 12. | Cassandra + | | Wide column, Multi-model i | 98.83 | -3.06 | -9.73 |
| 13. | 13. | 13. | MariaDB + | | Relational, Multi-model i | 91.04 | -2.17 | -6.28 |
| 14. | 14. | 14. | Splunk | | Search engine | 89.10 | +2.65 | -0.35 |
| 15. | 15. | ↑ 18. | Databricks + | | Multi-model i | 81.08 | +2.47 | +15.27 |
| 16. | 16. | 16. | Microsoft Azure SQL Database | | Relational, Multi-model i | 76.78 | -1.20 | -2.18 |
| 17. | 17. | ↓ 15. | Amazon DynamoDB + | | Multi-model i | 74.45 | +0.38 | -5.46 |
| 18. | 18. | ↓ 17. | Hive | | Relational | 59.76 | -1.42 | -15.76 |
| 19. | 19. | ↑ 20. | Google BigQuery + | | Relational | 58.10 | -2.28 | +3.46 |
| 20. | 20. | ↑ 21. | FileMaker | | Relational | 47.91 | -0.29 | -6.47 |
| 21. | ↑ 23. | ↑ 22. | Neo4j + | | Graph | 44.89 | +0.44 | -7.87 |
| 22. | ↓ 21. | ↓ 19. | Teradata | | Relational, Multi-model i | 44.87 | -0.46 | -17.77 |
| 23. | ↓ 22. | 23. | SAP HANA + | | Relational, Multi-model i | 44.27 | -0.42 | -7.14 |

DBMS popularity broken down by database model

June 2024



© 2024, DB-Engines.com

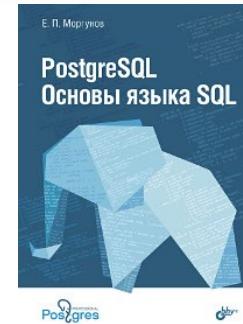
Data storage - main tools

- Most popular open DBMS (mainly relational)
 - PostgreSQL (<http://www.postgresql.org>)
 - Common denominator!
 - Dialect – PostgreSQL + PL/pgSQL (loadable procedural language)
 - SQLite (<http://www.sqlite.org>)
 - Handy embedded DBMS
- OLTP/OLAP worlds!
- Graph DBMSs were already mentioned (neo4j)

PostgreSQL in Russia (на русском)

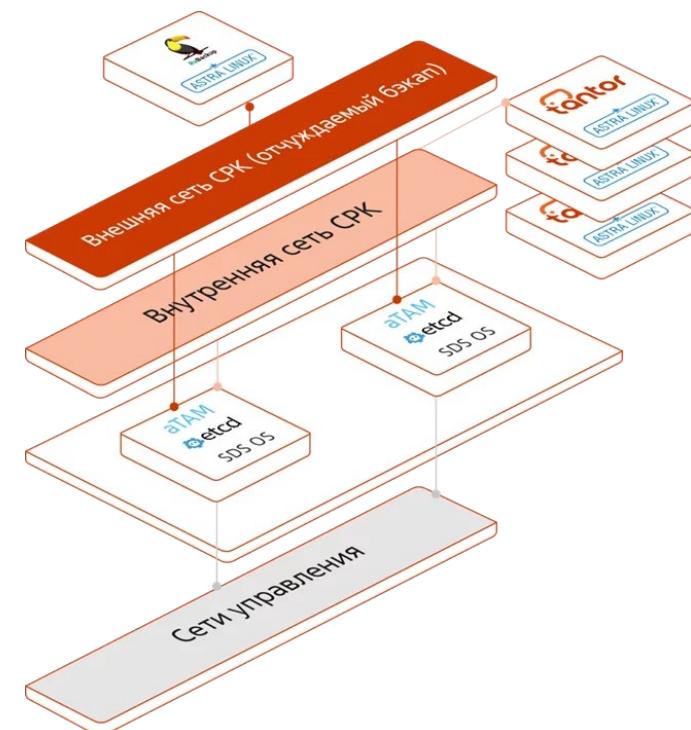
- Postgres Pro (<http://postgrespro.com>)

- Книги (<http://postgrespro.ru/education/books>)



- Tantor (<http://tantorlabs.ru>)

- Российская СУБД и платформа для управления базами данных на основе PostgreSQL
 - XData (<http://tantorlabs.ru/products/xdata>)



Document stores

- **CrateDB** (<https://cratedb.com/solutions/json-database>)

- CrateDB supports complex nested JSON payloads and allows querying of any property without requiring changes to your data schema



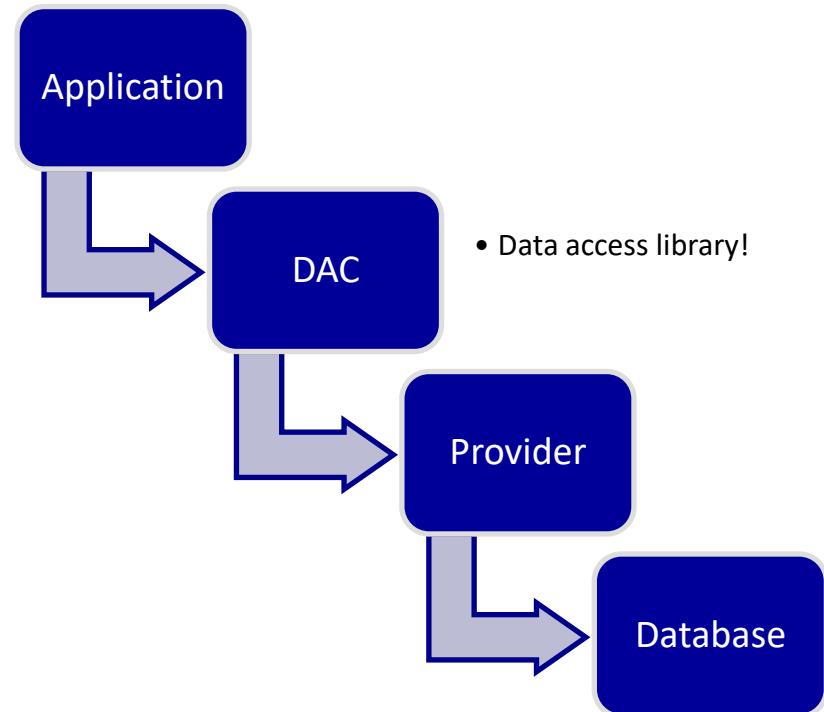
Time series DBMS

- **Timescale** – Postgres for time-series data (<http://www.timescale.com>)
- Apache **Pinot™** – Realtime distributed OLAP datastore (<http://pinot.apache.org>)
 - Timestamp Index (<http://docs.pinot.apache.org/basics/indexing/timestamp-index>)
 - **ThirdEye** – an integrated tool for real-time monitoring of time series and interactive root-cause analysis (<http://github.com/startreedata/thirdeye>)
- InfluxData **InfluxDB** – Time series data with a single, purpose-built database (<http://www.influxdata.com>)
- **OpenTSDB** – The Scalable Time Series Database (<http://opentsdb.net>)



Connection string and parameters of a connection

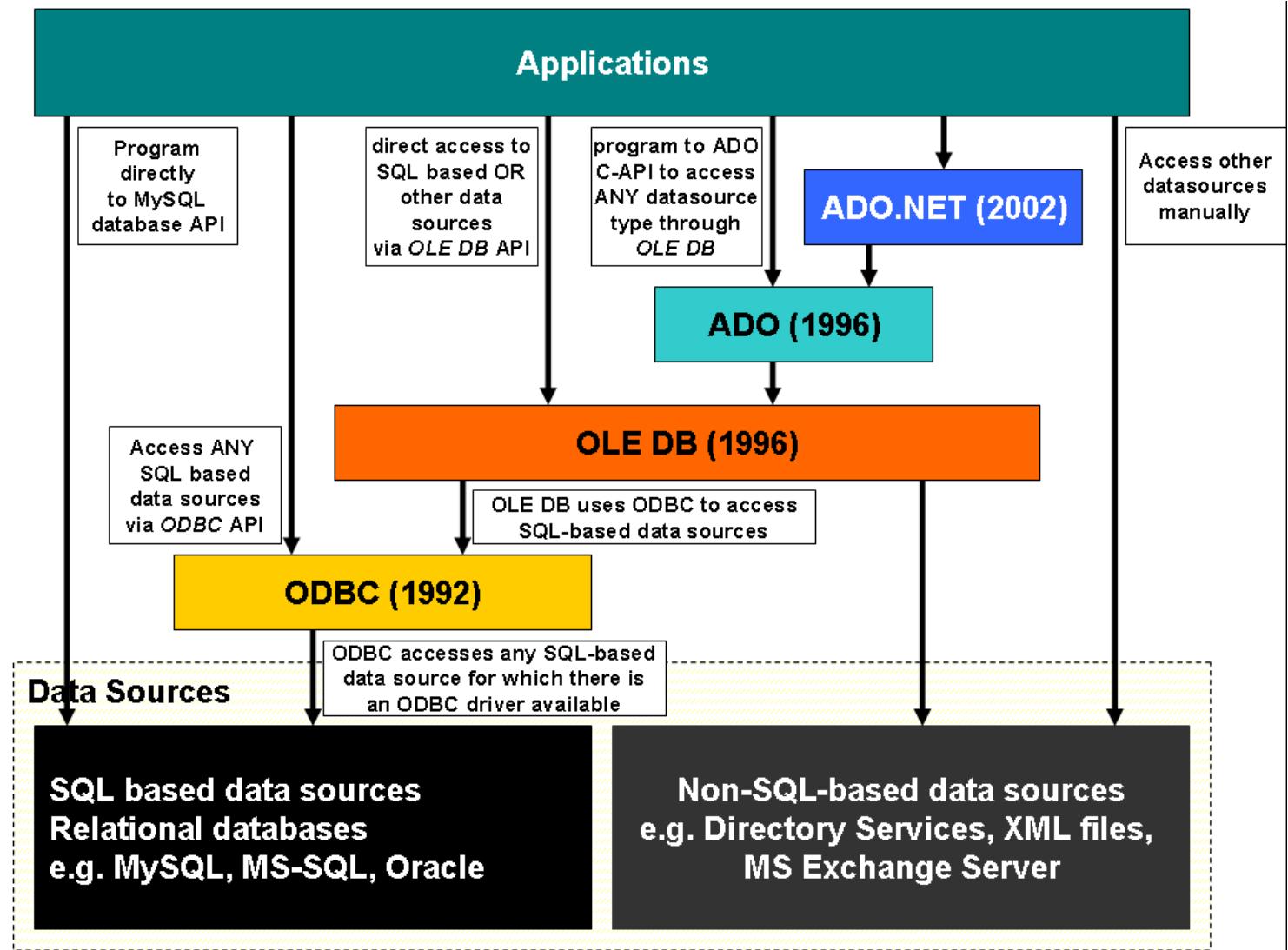
- Connection string!
 - The Connection Strings Reference (<http://www.connectionstrings.com>)



- Examples (ODBC):
 - PostgreSQL: “Driver={PostgreSQL}; Server=IP address; Port=5432; Database=myDataBase; Uid=myUsername; Pwd=myPassword;”
 - MySQL: “Driver={MySQL ODBC 5.2 UNICODE Driver}; Server=localhost; Database=myDataBase; User=myUsername; Password=myPassword; Option=3;”

Note: old good data providers' history

- Remember architecture changes!



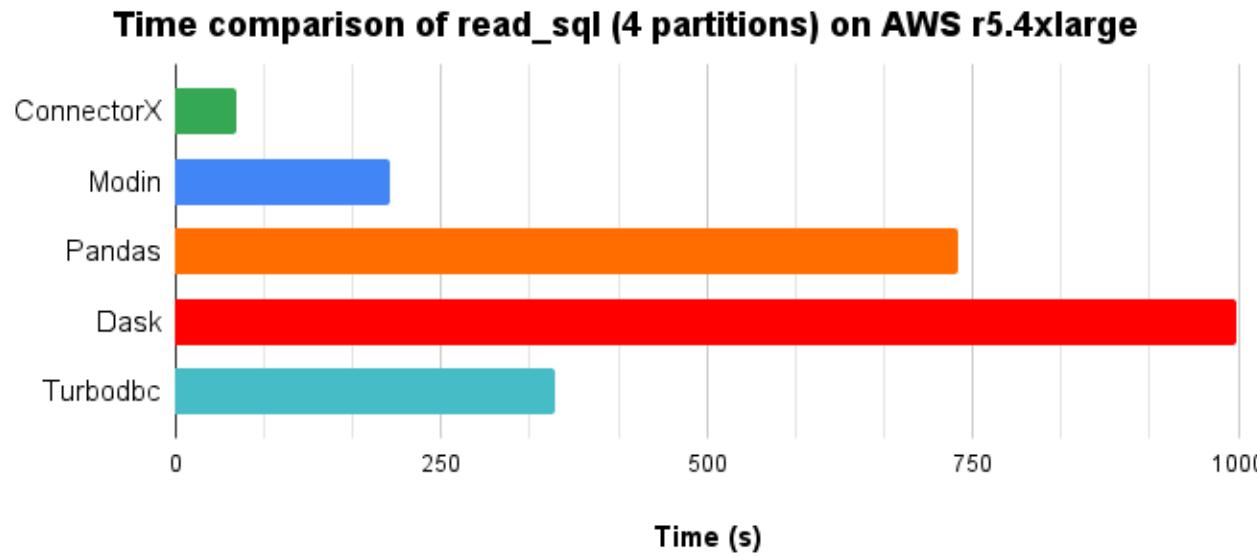
Apache Flight SQL and ADBC

- New era of data providers ☺

- Apache Arrow Flight SQL adapter
for PostgreSQL 0.1.0 - 2023-09-17
(<http://www.postgresql.org/about/news/apache-arrow-flight-sql-adapter-for-postgresql-010-2716/>)
- Introducing ADBC: Database Access for Apache Arrow
(<http://arrow.apache.org/blog/2023/01/05/introducing-arrow-adbc/>)
- [Article] Benchmarking Apache Arrow Flight –
A wire-speed protocol for data transfer, querying and
microservices. **2022**
(<http://dl.acm.org/doi/fullHtml/10.1145/3527199.3527264>)

ConnectorX in Python

- ConnectorX – fastest library to load data from DB to DataFrames in Rust and Python (<http://github.com/sfu-db/connector-x>)
 - ConnectorX enables you to load data from databases into Python in the fastest and most memory efficient way
 - MIT license



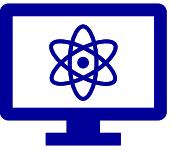
Specific ETL tools

- **dbt** (<http://www.getdbt.com>)

- SQL-based modelling and transformation:

The screenshot shows the dbt Docs interface. On the left, there's a sidebar with 'Overview', 'Projects' (listing 'acme'), and 'mrr' selected. The main area shows a 'mrr view' with tabs for 'Details', 'Description', 'Columns', and 'SQL'. Under 'Details', it shows 'TAGS: untagged', 'OWNER: TRANSFORMER', 'TYPE: view', 'PACKAGE: acme', and 'RELATION: analytics.dbt_claire_playbook.mrr'. The 'Description' section contains two paragraphs about the model. The 'Columns' section lists 'id' (TEXT), 'date_month' (TIMESTAMP_NTZ), and 'customer_id' (NUMBER). To the right is a 'Lineage Graph' showing dependencies between models: 'customer_revenue_by_month' and 'customer_churn_month' both depend on 'mrr'.

The screenshot shows the dbt IDE interface. At the top, there's a header with 'dbt' logo, 'Develop', 'Deploy', 'Documentation', and a GitHub integration button. The main area has a 'File Explorer' on the left listing dbt files like 'orders.sql', 'customers.sql', etc. In the center, there's a code editor with an open file 'orders.sql' containing SQL code for generating 'orders' from multiple sources. Below the code editor is a 'Lineage Graph' showing the relationships between various source tables ('ecommerce_src_orders', 'stg_order_items', etc.) and target tables ('stg_orders', 'stg_order_items', 'stg_products', 'stg_locations', 'stg_supplies').



Orchestration

- ✓ Legacy ☺
- ✓ New generation

Apache Airflow

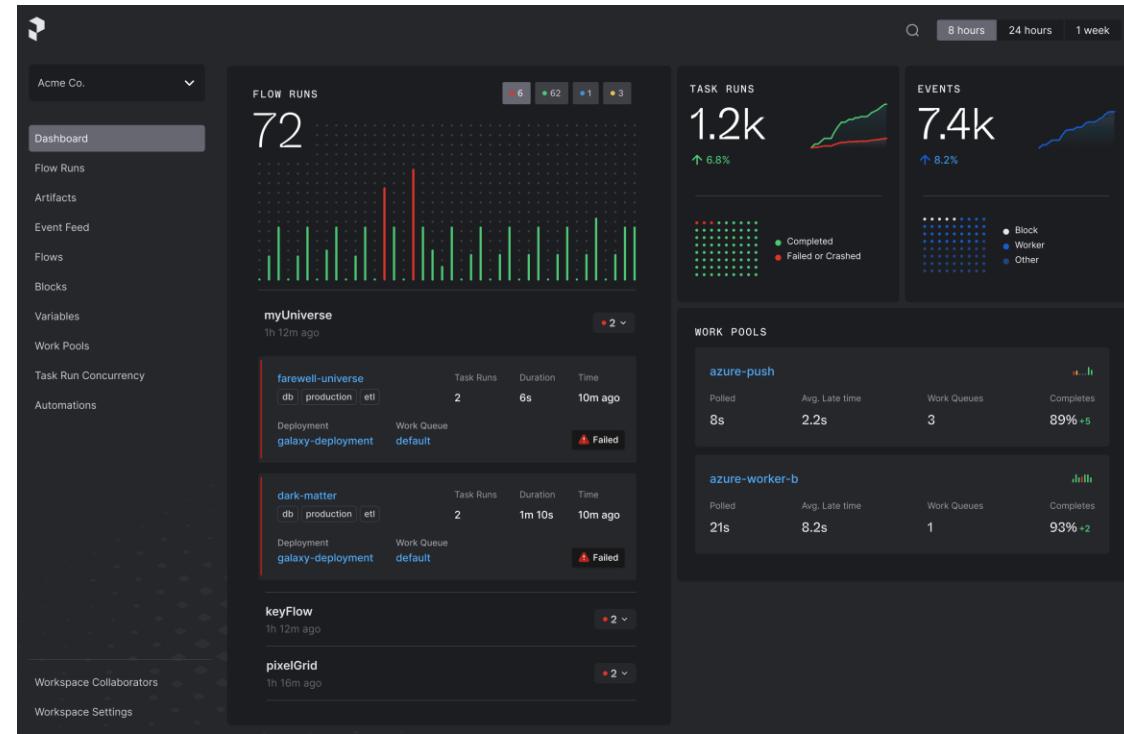
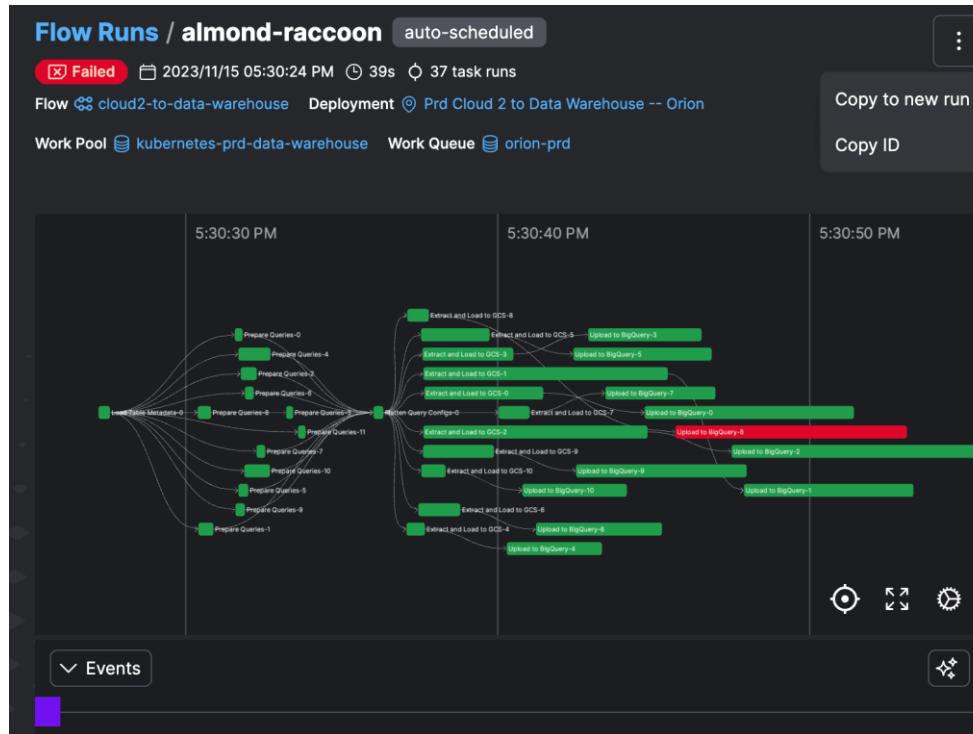
- Apache Airflow (<http://airflow.apache.org>)
 - Java!
 - Airflow Ecosystem (<http://airflow.apache.org/ecosystem/>)
 - Apache Airflow packaged by Bitnami
(<http://github.com/bitnami/charts/tree/main/bitnami/airflow>)
- Apache Airflow and ETL Pipelines with Python (<http://www.krasamo.com/apache-airflow/>)
- Good ETL Practices with Apache Airflow - Analytics Vidhya
(<http://www.analyticsvidhya.com/blog/2021/11/good-etl-practices-with-apache-airflow/>)



Prefect

- Prefect (<http://www.prefect.io>)

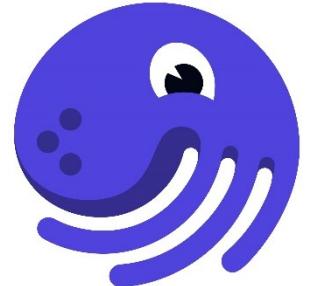
- The orchestrator for people who have done orchestration before!



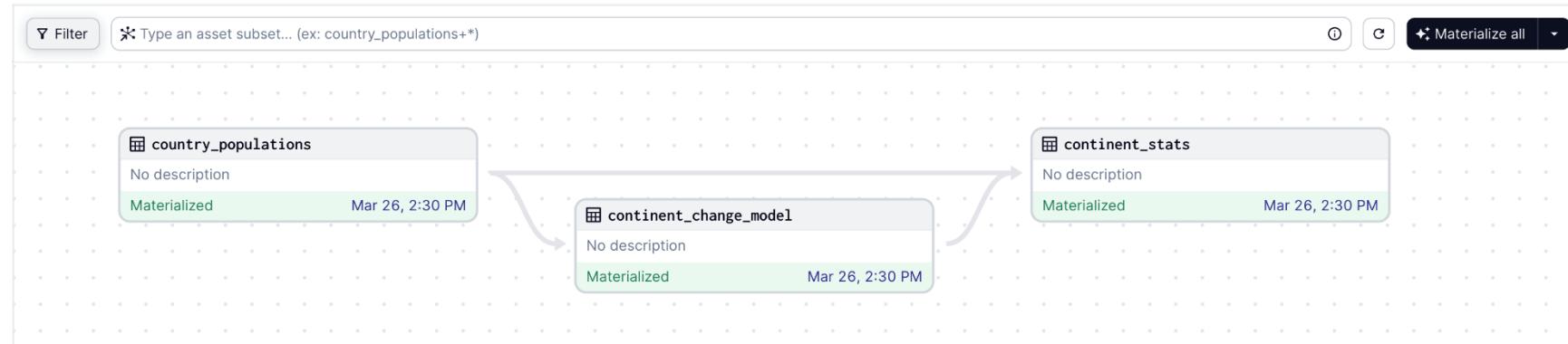
Dagster

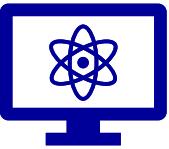
- **Dagster** (<http://github.com/dagster-io/dagster>)

- Ver>=0.17
- Good integration!



- Dagster is a cloud-native data pipeline orchestrator for the whole development lifecycle, with integrated lineage and observability, a declarative programming model, and best-in-class testability



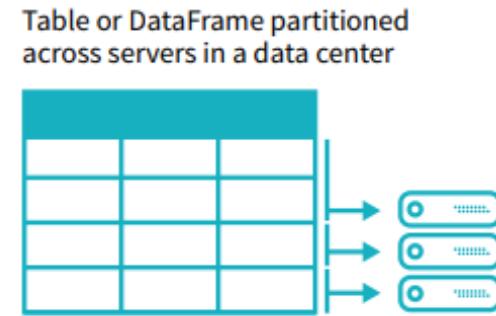
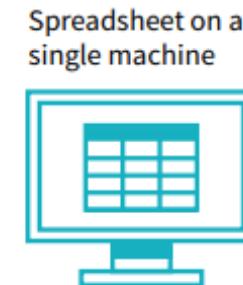


Dataframe as a heart of common data transformation

- ✓ Dataframes and tables
- ✓ New generation of Python dataframe libraries
 - ✓ + Dataframe Interchange Protocol (DIP)
- ✓ Dataframes interoperability
 - ✓ Python, R, ...
- ✓ Simple vector DBMSs

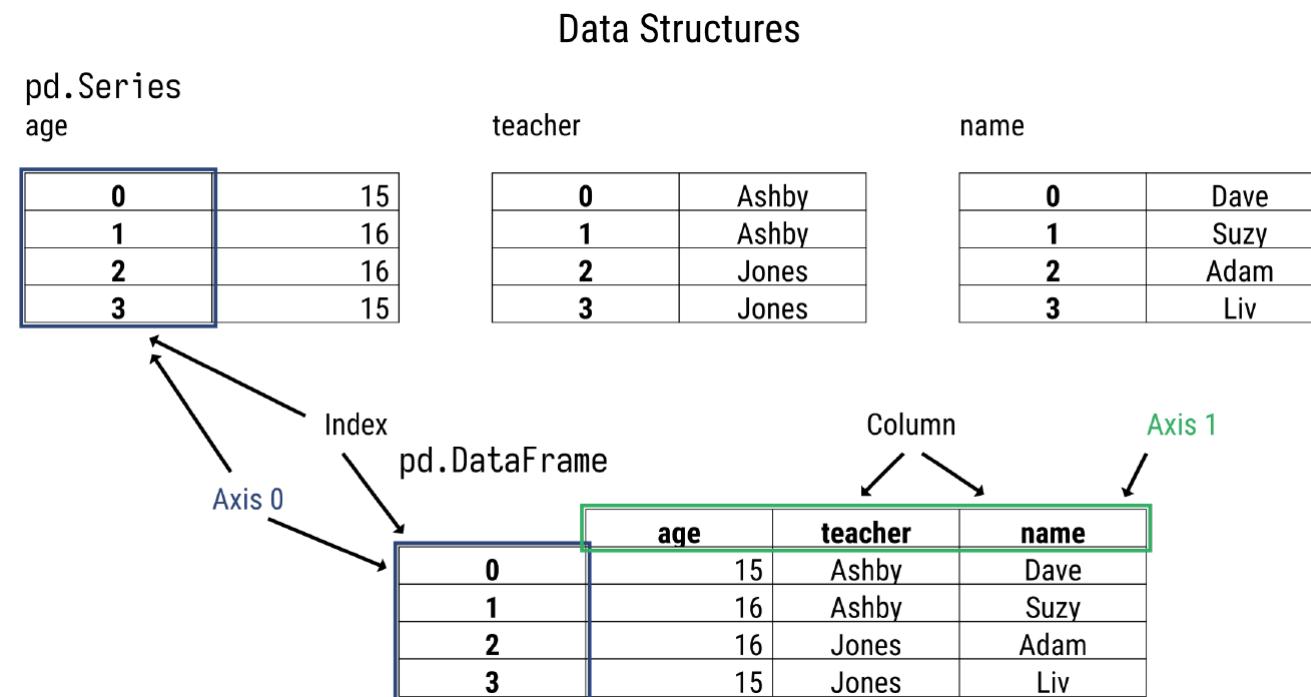
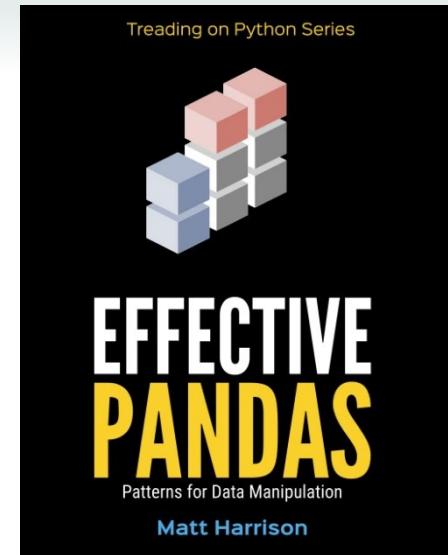
Dataframe

- A **tabular data structure** common to many data processing libraries:
 - pandas and other Python libraries
 - The Dataframe API in Apache Spark
 - Data frames in the R programming language
 - ...
- Databricks – What is a DataFrame?
(<http://www.databricks.com/glossary/what-are-dataframes>)
 - DataFrame contains a blueprint, known as a schema, that defines the name and data type of each column
 - Missing or incomplete values are stored as null values in the DataFrame



Python – Pandas

- Harrison M. Effective Pandas. Patterns for Data Manipulation. 2021. 397 p.
 - Series manipulation
 - Creating columns
 - Summary statistics
 - Grouping, pivoting, and cross-tabulation
 - Time series data
 - Visualization
 - Chaining



Python – Polars

- **Polars** – Lightning-fast DataFrame library for Rust and Python (<https://www.pola.rs>)

How Polars will make your life easier

01

Easy
to use

Write your queries the way they were intended. Polars will determine the most efficient way to execute them using its query optimizer.

02

Embarrassingly
parallel

Complete your queries faster! Polars fully utilizes the power of your machine by dividing the workload among the available CPU cores without any additional configuration or serialization overhead.

03

Apache
Arrow

Polars utilizes the Apache Arrow memory model allowing you to easily integrate with existing tools in the data landscape. It supports zero-copy data sharing for efficient collaboration.

04

Close to
the metal

Polars is written from the ground up, designed close to the machine and without external dependencies. This allows for full control of the ecosystem (API, memory & execution).

05

Written
in Rust

The core of Polars is written in Rust, one of the fastest growing programming languages in the world. Rust allows for high performance with fine-grained control over memory.

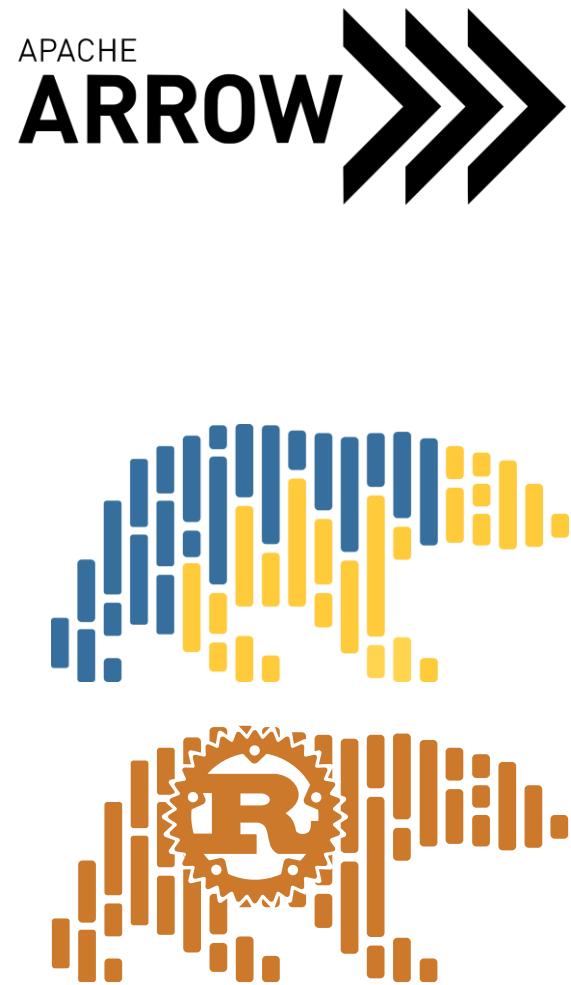
06

Out
of core

Want to process large data sets that are bigger than your memory? Our streaming API allows you to process your results efficiently, eliminating the need to keep all data in memory.

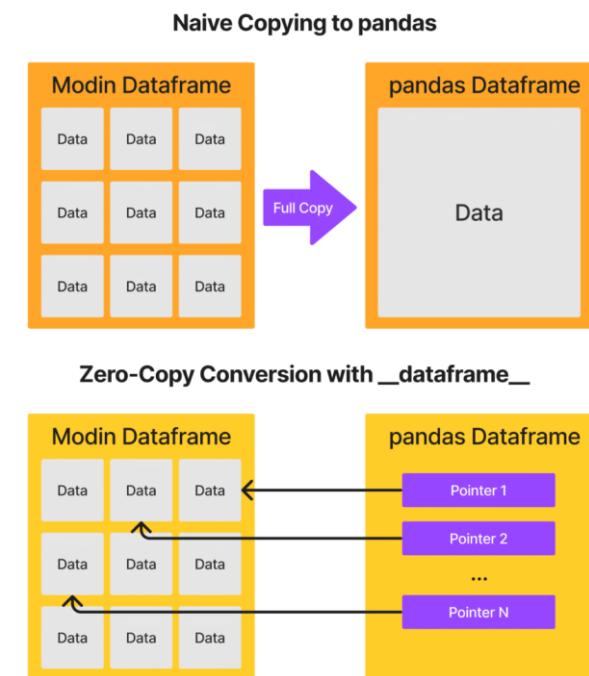
Arrow + Polars (Rust + Python)

- Apache **Arrow** – development platform for in-memory analytics (<http://arrow.apache.org>)
 - Reading and Writing the Apache **Parquet** Format (<http://arrow.apache.org/docs/python/parquet.html>)
- **Polars** – Lightning-fast DataFrame library for Rust and Python (<http://www.pola.rs>)
 - Lightning-fast DataFrame library for Rust and Python
 - Rust / Python
 - Lazy execution, end-to-end optimization
 - SQL-style declarative operations



Python dataframes interoperability

- Consortium for Python Data API Standards
(<http://data-apis.org>)
- Dataframe Interchange Protocol
(http://arrow.apache.org/docs/python/interchange_protocol.html)
- How the Python Dataframe Interchange Protocol makes life better
(<http://ponder.io/how-the-python-dataframe-interchange-protocol-makes-life-better/>)
 - Vaex (PR merged 10/13/21)
 - cuDF (PR merged 11/17/21)
 - Modin (PR merged 2/25/22)
 - pandas (PR merged 4/27/22)
 - Polars (PR merged 1/30/23)
 - ibis (PR merged 6/16/23)



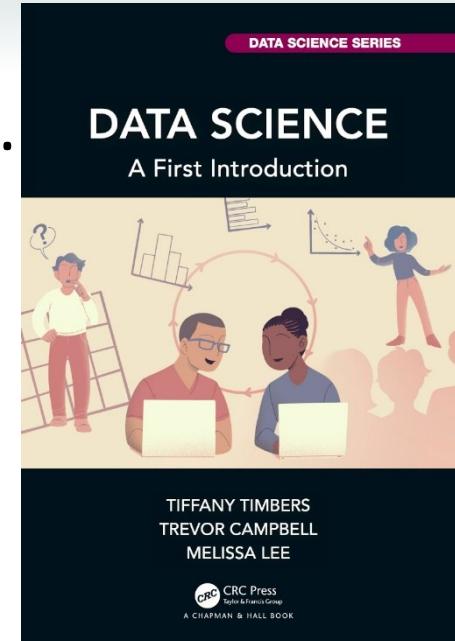
R - data frame

- Timbers T., Campbell T., Lee M. Data Science. A First Introduction.
 - What is a data frame? (<http://datasciencebook.ca/wrangling.html#what-is-a-data-frame>)

| region | year | Variable |
|-----------|------|------------|
| Toronto | 2016 | population |
| Vancouver | 2016 | 1027613 |
| Montreal | 2016 | 1823281 |
| Calgary | 2016 | 544870 |
| Ottawa | 2016 | 571146 |

Observation

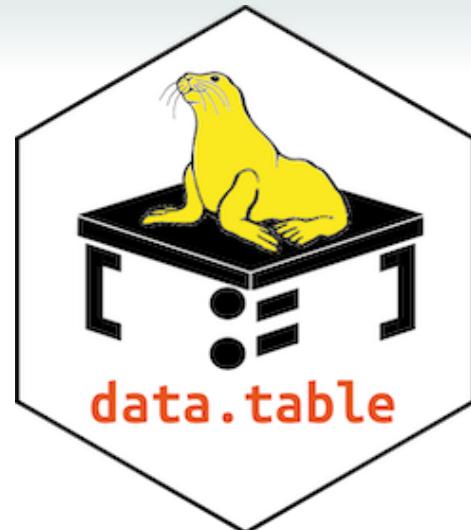
Value



- + Tidyverse – R packages for data science (<https://www.tidyverse.org>)

R – Tables

- `data.table` is an enhanced version of `data.frames`
 - Signorell A. Tables in R – A quick practical overview. 2021
(<http://cran.r-project.org/web/packages/DescTools/vignettes/TablesInR.pdf>)



R: Data Table

DT[i, j, by]



Row filter
~ SQL WHERE clause

Column select/update..
~ SQL SELECT/ UPDATE/
ADD/ DROP etc.

Grouped By
~ SQL GROUP BY

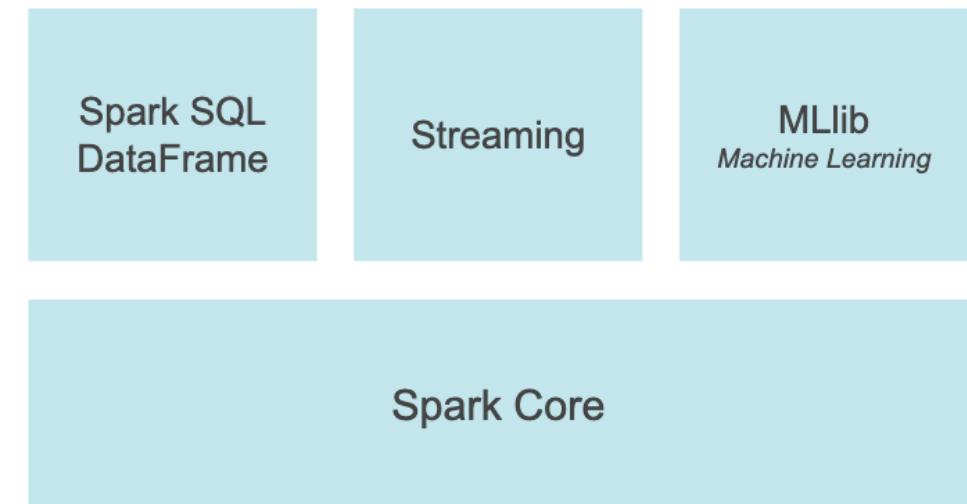
Spark – DataFrame



- Apache PySpark – `pyspark.sql.DataFrame`

(<http://spark.apache.org/docs/3.1.1/api/python/reference/api/pyspark.sql.DataFrame.html>)

- A DataFrame is a Dataset organized into named columns
- It is conceptually equivalent to a table in a relational database or a data frame in R/Python, but with richer optimizations under the hood



Terminology drift

- Preventing the Death of the Dataframe

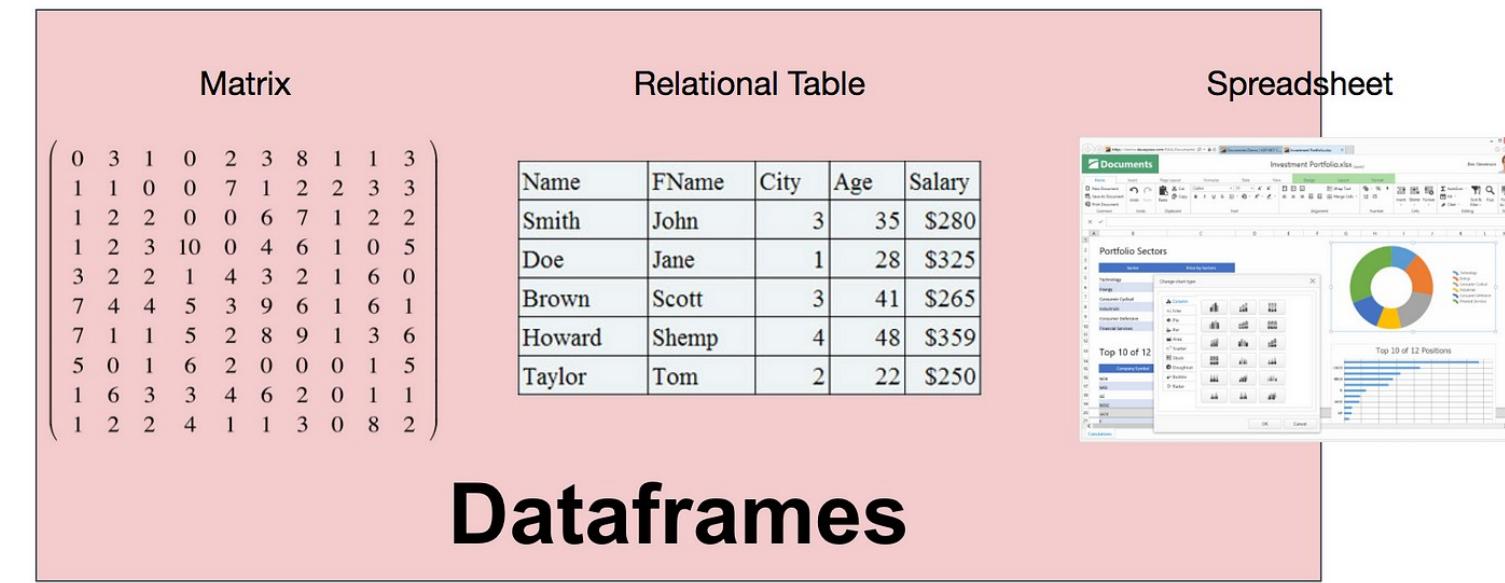
(<http://towardsdatascience.com/preventing-the-death-of-the-dataframe-8bca1c0f83c8>)

- Dataframes are losing their statistical computing and machine learning roots:



- Dataframe supports:

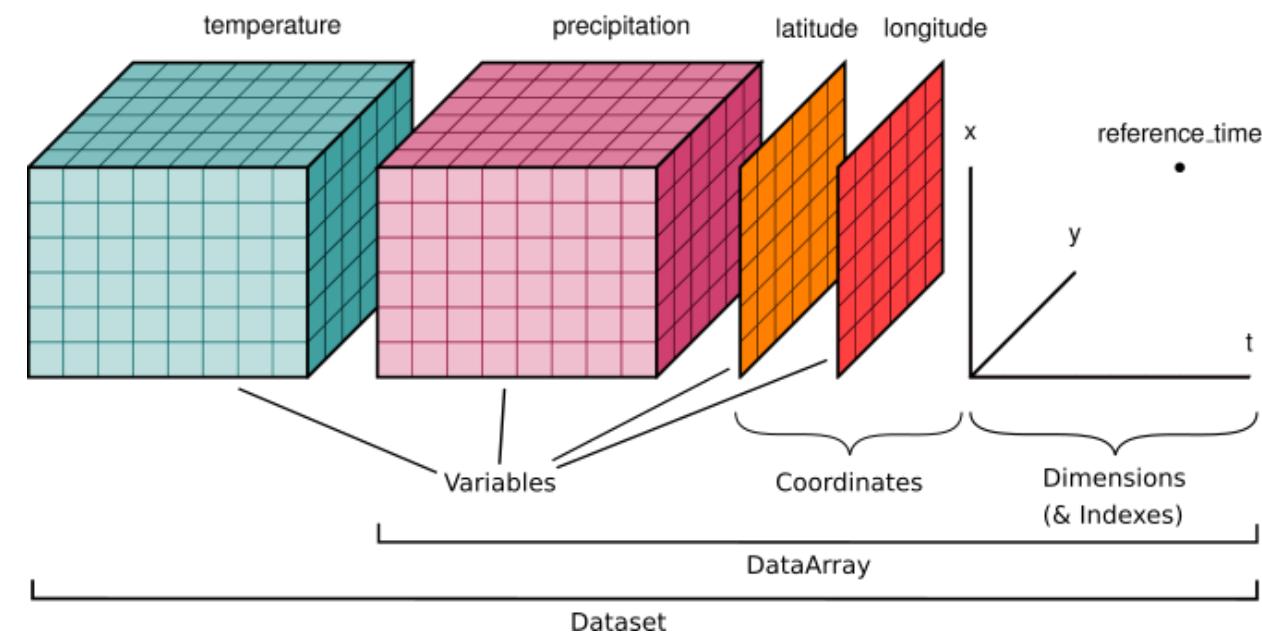
- Linear algebra (matrices!)
- Relational algebra (tables!)
- Some cell formulae (spreadsheets!)



Multidimensional arrays (Xarray)

- **Xarray** (<http://xarray.dev>)

- N-D labeled arrays (tensors) and datasets in Python
- Big ecosystem for ML
 - **DataArray**: a multi-dimensional array with labeled or named dimensions
 - **Dataset**: a dict-like collection of DataArray objects with aligned dimensions
- Built on top of NumPy, Pandas, and Dask
- Supports a wide range of domain specific scientific applications



Multidimensional arrays and more complex structures

- **Taichi** (<http://www.taichi-lang.org>)

- Domain-specific language embedded in Python that helps you easily write portable, high-performance parallel programs





Speed up Python

```
import taichi as ti
ti.init()

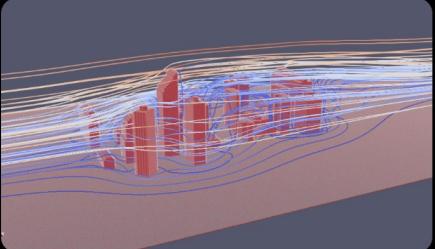
@ti.kernel
def monte_carlo_pi(n: int) -> float:
    total = 0
    for i in range(n):
        x = ti.random()
        y = ti.random()
        if (x ** 2 + y ** 2) < 1.0:
            total += 1
    return 4.0 * total / n
```

Taichi translates the compute-intensive Python code into fast machine code at runtime with a just-in-time compiler, accelerating your Python code.

[Try now >](#)



Numerical simulation



Create an urban airflow simulation using 3D Lattice Boltzmann Method, with spatially sparse data structures provided by Taichi.

[Learn more >](#)



3D rendering



A GPU path tracing voxel renderer made by Taichi! You can easily create inspiring voxel art using this renderer within 99 lines of Taichi code.

[Learn more >](#)



Computer vision



With automatic parallelization and spatially sparse data structures, Taichi powers the fast development of a LiDAR-based SLAM system.

[Learn more >](#)

Some Taichi features

- DSL with intuitive linear algebra constructs
- SNodeTree-based containers optimized for GPU backend
 - Fields (<http://docs.taichi-lang.org/docs/layout>)
 - Spatially Sparse Data Structures (<http://docs.taichi-lang.org/docs/sparse>)

High-performance parallel computing

A one-billion-particle MPM simulation running on a GPU with 80 GB memory - the world's largest-scale single-GPU particle simulation!

Learn more >

Automatic differentiation

Soft robot locomotion control powered by Taichi's handy automatic differentiation system, which is [significantly faster than PyTorch, TensorFlow, and JAX](#).

Learn more >

Flexible SNode system

Row-major Column-major

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$
$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Taichi's SNode system allows quick experimentation with different memory layouts to maximize performance without rewriting computation code.

Learn more >

Spatially sparse data structures

Particles 1x1x1
4x4x4
16x16x16

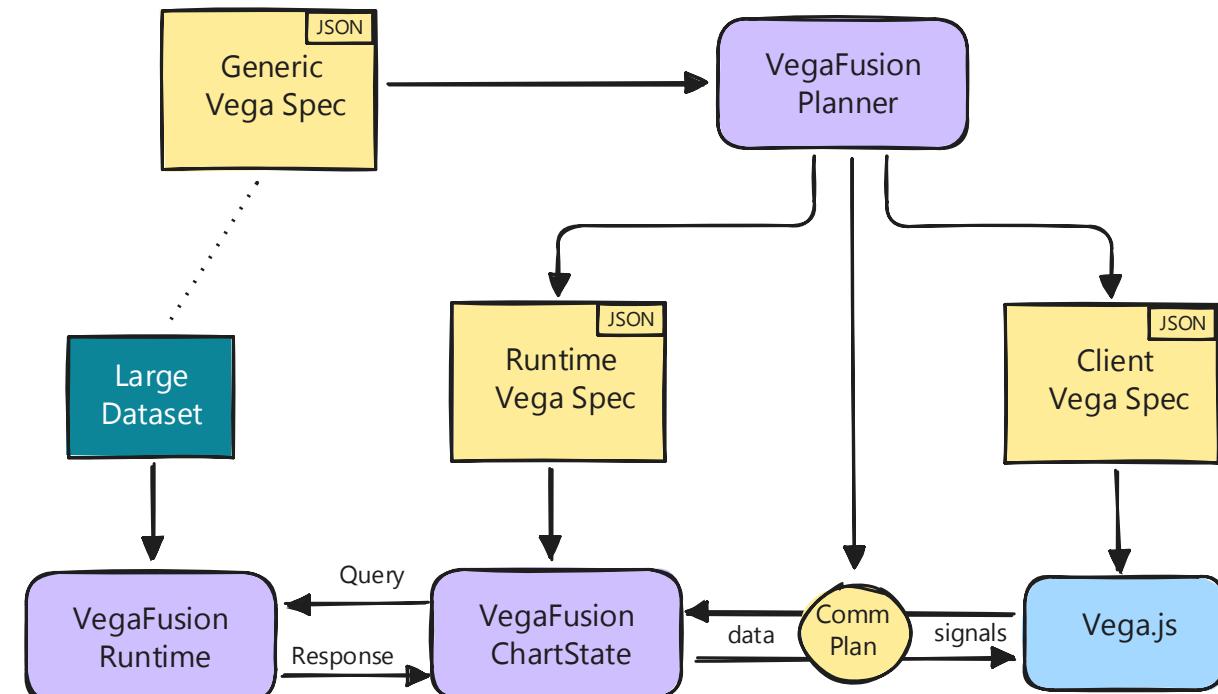
A fluid simulation example where empty regions take no computation nor memory, enabled by spatially sparse data structures in Taichi.

Learn more >

Dataframe “analytical” previz engines

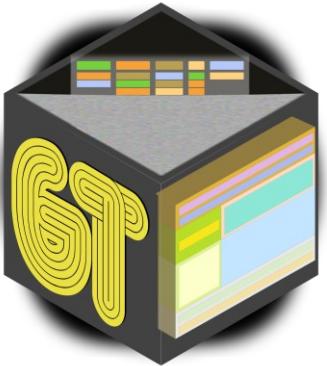
- **VegaFusion** (<http://vegafusion.io>)

- Building blocks for analyzing, accelerating, and scaling Vega visualizations
- Provides Rust, Python, and JavaScript libraries
- Replaces altair-transform

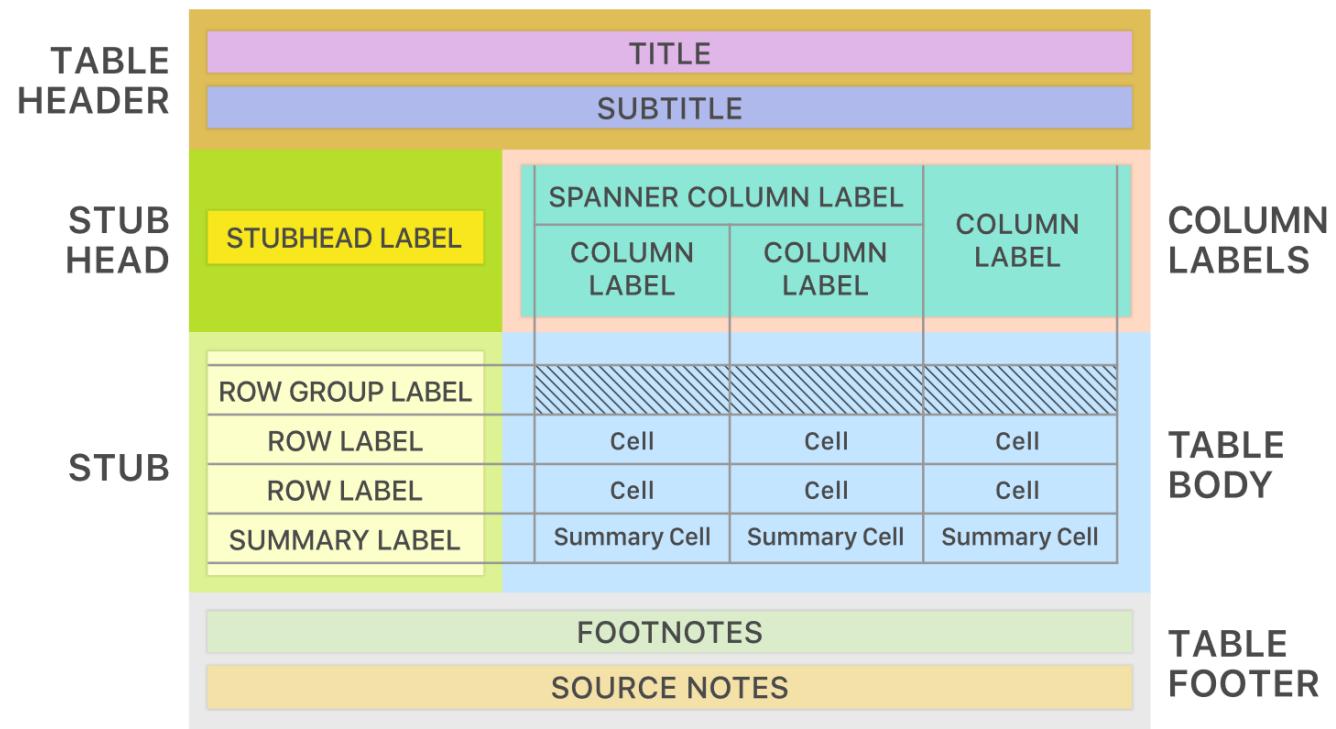


[R] gt (A grammar of tables)

- gt package (<http://gt.rstudio.com>)
 - Constructs a wide variety of useful tables with a cohesive set of table parts:
 - Header, stub, column labels with spanners, body with summary, footer, ...



The Parts of a gt Table



Great Tables

- **Great Tables:** The [Polars DataFrame Styler of Your Dreams](http://posit-dev.github.io/great-tables/blog/polars-styling/)

(<http://posit-dev.github.io/great-tables/blog/polars-styling/>)

- **Title and subtitle:**
User friendly titles and subtitles, describing the data
- **Column spanners:**
Group related columns together with a custom label
- **Styles:**
Highlight rows, columns, or individual cells of data
- + markdown and research-oriented formatting

| New York Air Quality Measurements | | | | | | |
|-----------------------------------|-------|-----|--|---------------------------------|--------------|-------------|
| | | | Daily measurements in New York City (May 1-10, 1973) | | | |
| | | | Time | | | |
| Year | Month | Day | Ozone, ppbV | Solar R., cal/m ² | Wind, mph | Temp, °F |
| 1973 | 5 | 1 | 41.0 | 190.0 | 7.4 | 67 |
| 1973 | 5 | 2 | 36.0 | 118.0 | 8.0 | 72 |
| 1973 | 5 | 3 | 12.0 | 149.0 | 12.6 | 74 |
| 1973 | 5 | 4 | 18.0 | 313.0 | 11.5 | 62 |
| 1973 | 5 | 5 | None | None | 14.3 | 56 |
| 1973 | 5 | 6 | 28.0 | None | 14.9 | 66 |
| 1973 | 5 | 7 | 23.0 | 299.0 | 8.6 | 65 |
| 1973 | 5 | 8 | 19.0 | 99.0 | 13.8 | 59 |
| 1973 | 5 | 9 | 8.0 | 19.0 | 20.1 | 61 |
| 1973 | 5 | 10 | None | 194.0 | 8.6 | 69 |

Title

Column
spanners

Styles

Panel for rich tabular data representation

- **Tabulator** (<http://panel.holoviz.org/reference/widgets/Tabulator.html>)
 - Widget allows displaying and editing a dataframe ([Pandas](#))

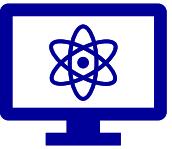
| index | int | float | str | bool | date | datetime |
|-------|-----|--|-----|------|------------|---------------------|
| 1 | 1 | | A | ✓ | 2019-01-01 | 2019-01-01 10:00:00 |
| 2 | 2 | | B | ✗ | 2020-01-01 | 2020-01-01 12:00:00 |
| 3 | 3 | | C | ✓ | 2020-01-10 | 2020-01-10 13:00:00 |

Vector DBMS for tight integration

- LanceDB (<http://github.com/lancedb/lancedb>)



- LanceDB + Polars
(<http://blog.lancedb.com/lancedb-polars-2d5eb32a8aa3/>)
- Benchmarking random access in Lance
(<http://blog.lancedb.com/benchmarking-random-access-in-lance/>)
 - “We’ve been claiming 100x faster random access performance than parquet, but as this benchmark shows, it’s really more like **2000x**”
- Lance v2 is now in Beta
(<http://blog.lancedb.com/lance-v2-is-now-in-beta/>)

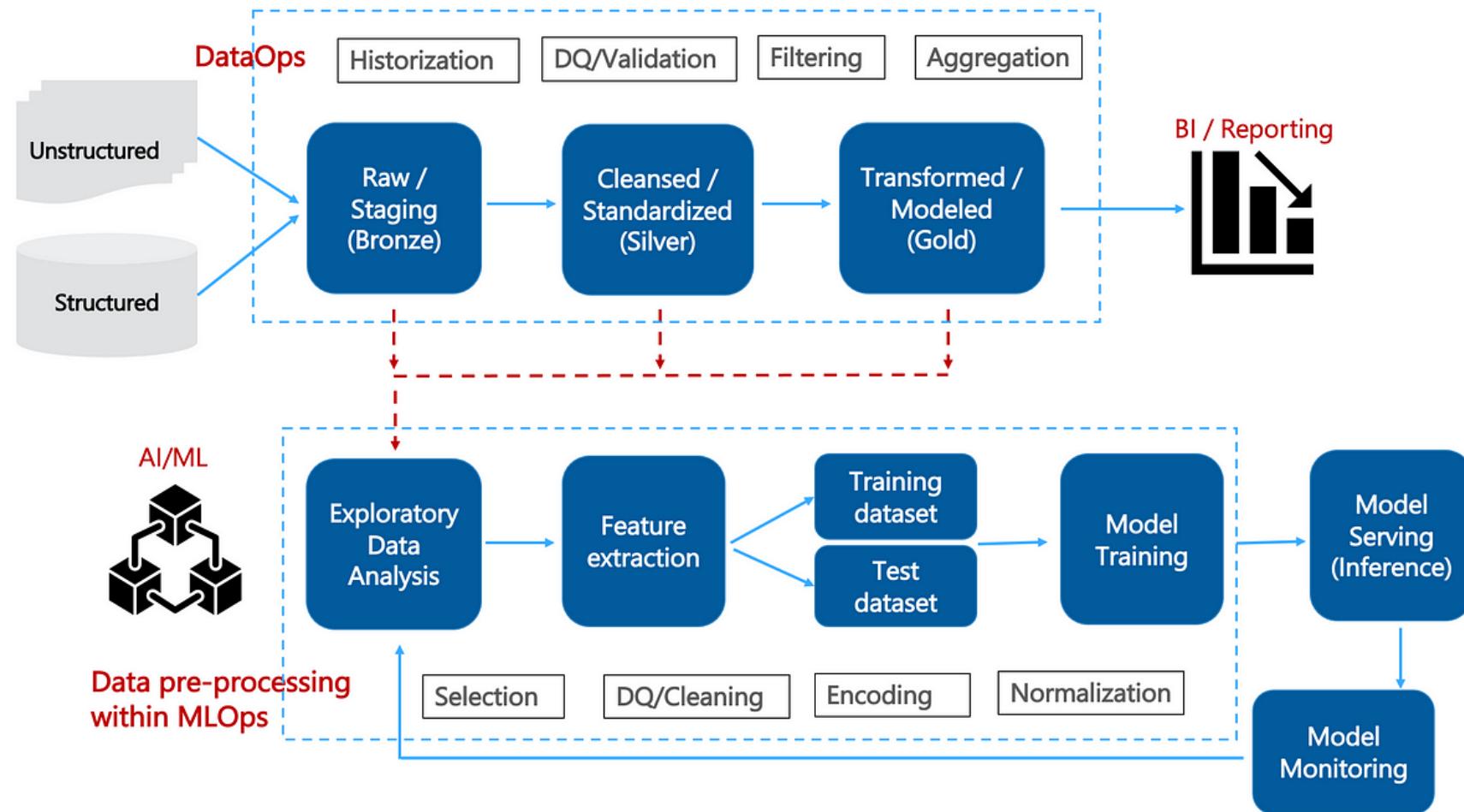


Machine learning

- ✓ MLOps
- ✓ Pythonic way
- ✓ Ray platform

MLOps

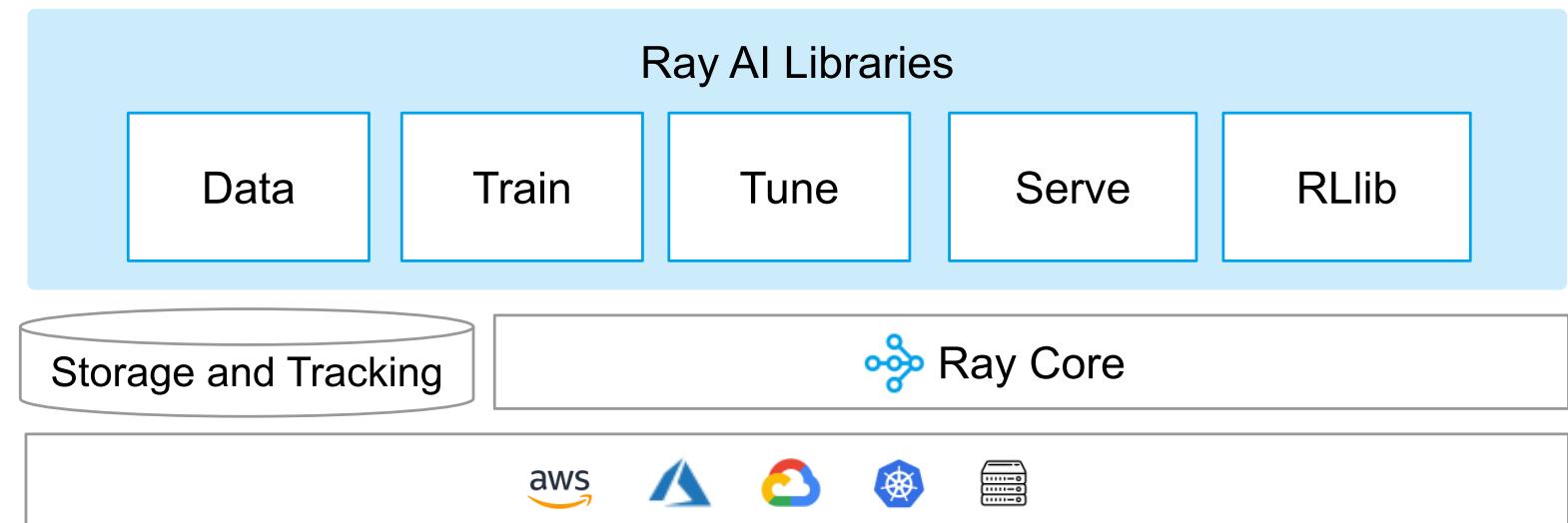
- What are Dataops and mlops? (<http://www.thedataops.org/what-are-dataops-and-mlops/>)



Pythonic MLOps

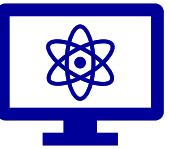
- **Ray** (<http://www.ray.io>)

- An open-source unified compute framework that makes it easy to scale AI and Python workloads — from reinforcement learning to deep learning to tuning, and model serving
- See Ray Use Cases (<http://docs.ray.io/en/latest/ray-overview/use-cases.html>)



Separate session

- Data Science at Scale?
 - Optimal data pipelining and versioning
 - Feature engineering
 - ML methods
 - Tensor processing
 - Security
 - ...
- Some references:
 - **Osipov C.** MLOps Engineering at Scale. Manning, 2022. 340 p.
 - **Pote S.** Machine Learning in Production. Master the art of delivering robust Machine Learning solutions with MLOps. BPB, 2023. 458 p.

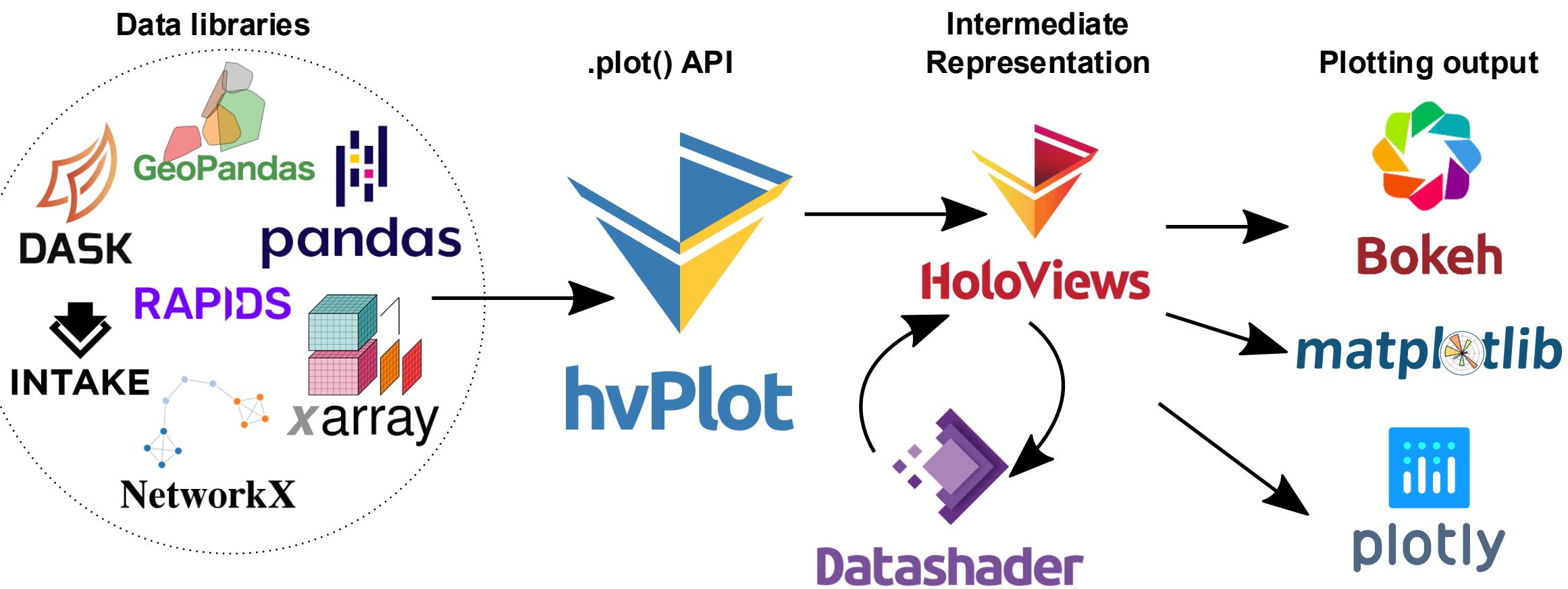


DataViz in Python

- ✓ Ecosystem
- ✓ Functionality, efficiency, and efficacy
- ✓ Charts
 - ✓ Dynamic
 - ✓ Static
- ✓ BI

hvPlot – A familiar and high-level API for data exploration and visualization

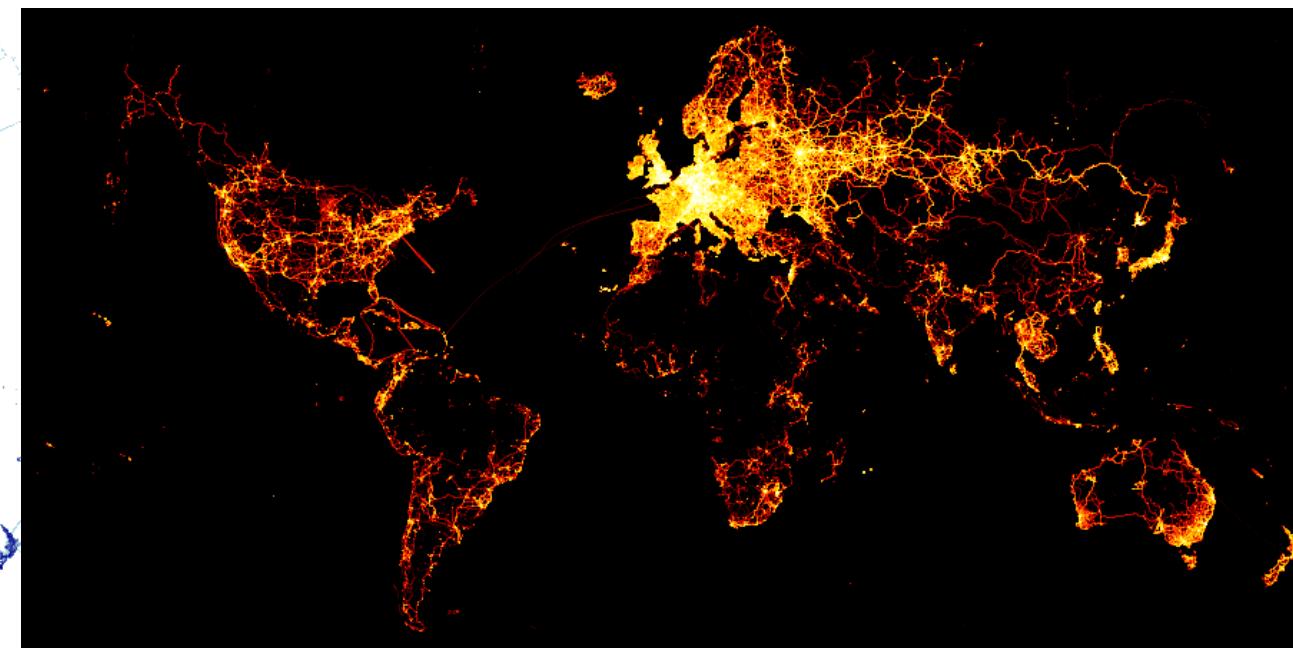
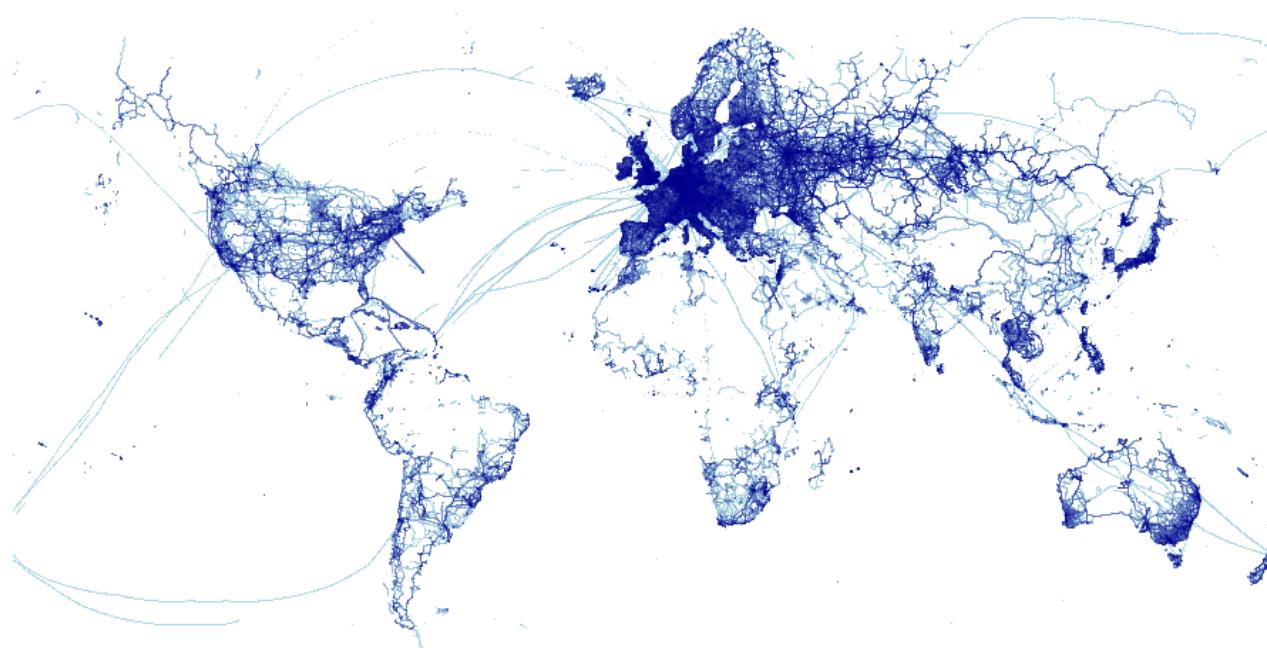
- hvPlot (<http://hvplot.holoviz.org>)
 - hvPlot Explorer – Example (http://panel-gallery.pyviz.demo.anaconda.com/hvplot_explore)



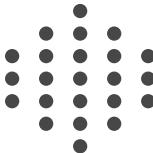
Effective rasterization (Datashader)

- **Datashader** (<http://datashader.org>)

- A graphics pipeline system for creating meaningful representations of large datasets quickly and flexibly
- ✓ DataShader – Plotting Pitfalls (http://datashader.org/user_guide/Plotting_Pitfalls.html)
 - Overplotting, Oversaturation, Undersampling, Undersaturation, Underutilized range, Nonuniform colormapping, ...



Lets-Plot for Python and Kotlin

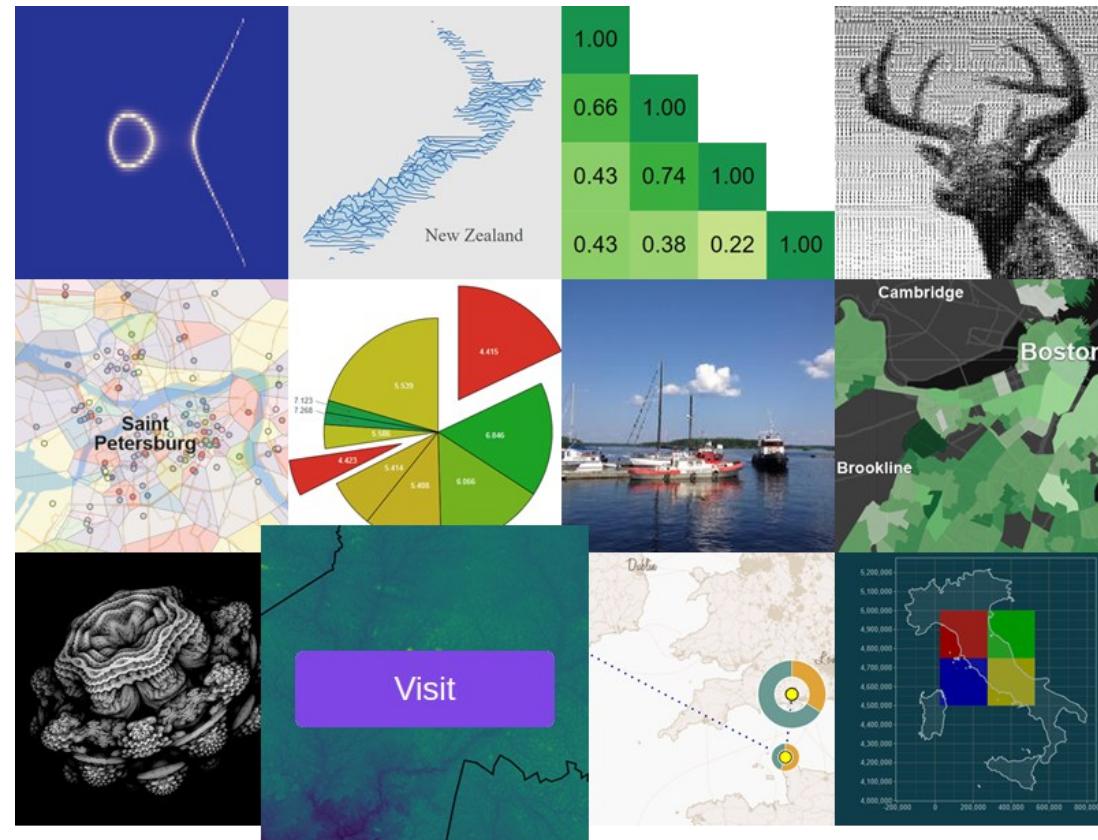
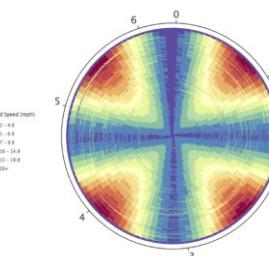
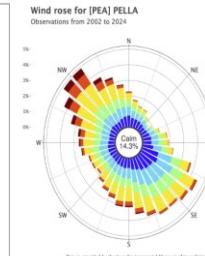
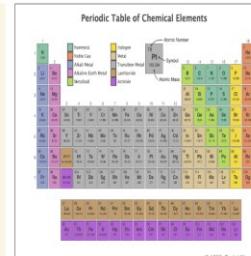
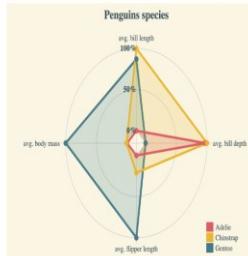
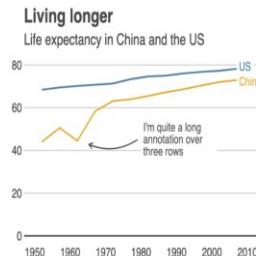


- **Lets-Plot** (<http://lets-plot.org>)
 - Multiplatform plotting library built on the principles of the Grammar of Graphics

- Inspired by **ggplot2**!
 - **Grammar of Graphics** for R



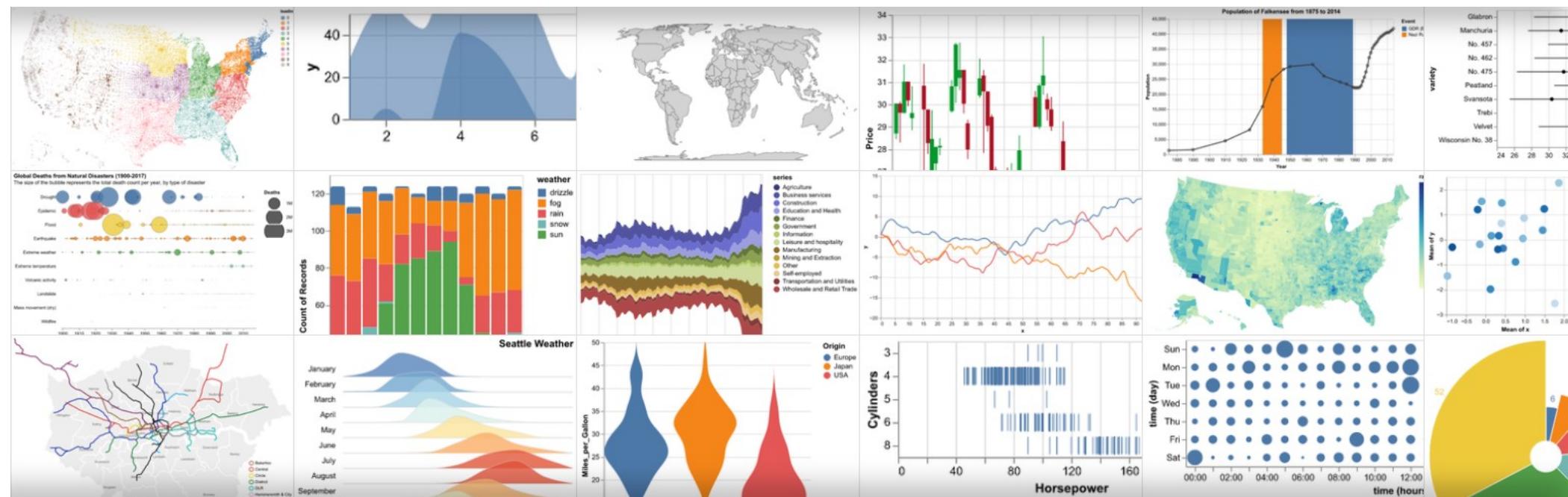
- ## ● Best static charts!



Vega-Altair: Declarative Visualization in Python

- **Vega-Altair** (<http://altair-viz.github.io>)

- Simple, friendly and consistent API, built on top of the powerful Vega-Lite grammar
- **Interactive dynamic** charts!
 - Rich interactive UI with events and selectors



NCVis

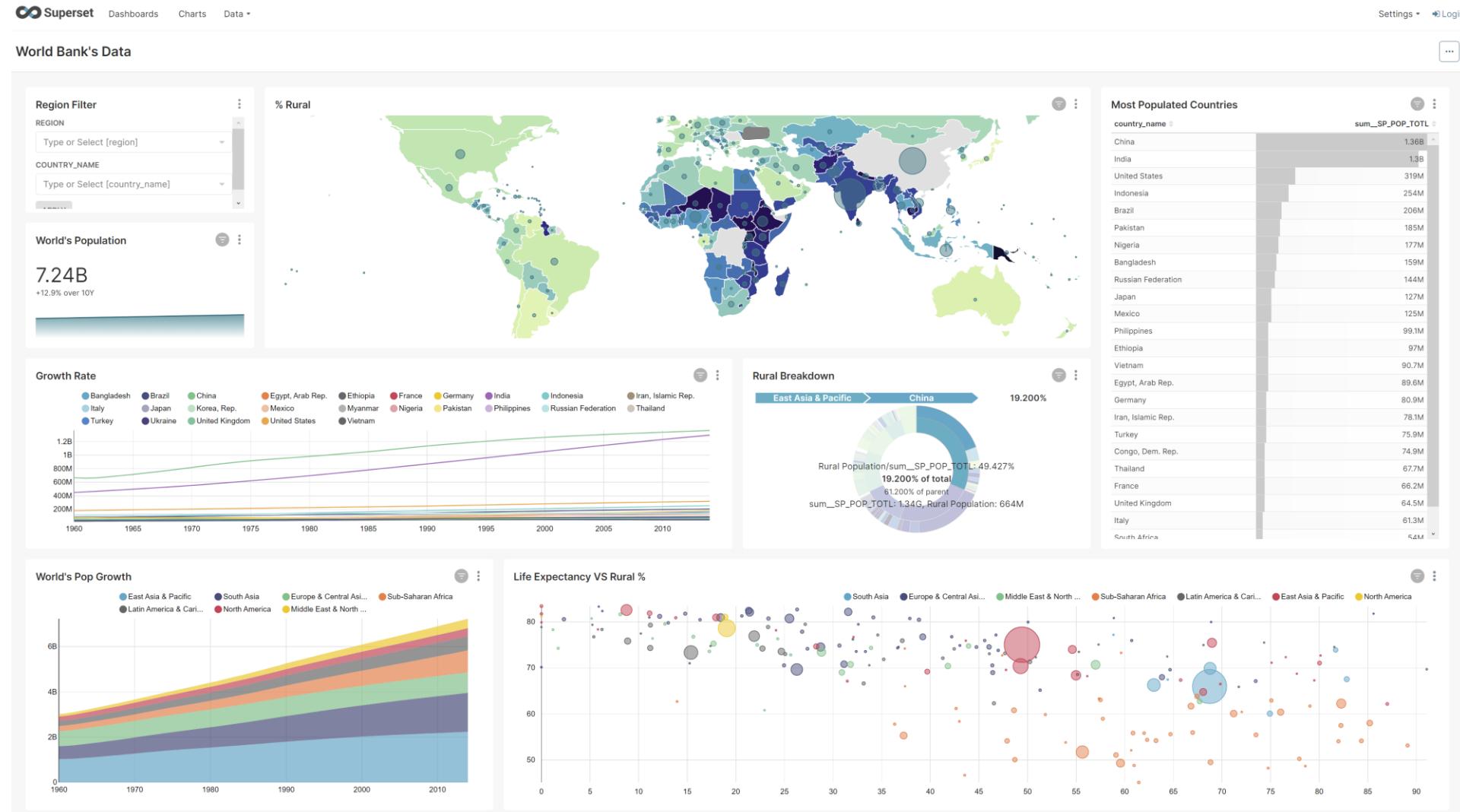
- **NCVis** (<http://github.com/stat-ml/ncvis>)
 - An efficient solution for data visualization and dimensionality reduction
 - Python / C++



Dashboarding

- Apache **superset** (<http://superset.apache.org>)
 - Python flask application with react js components + ECharts (<http://echarts.apache.org>)
 - Deployment: local (Microsoft Windows, Linux), docker, Kuber
 - Simple ideology: Data source – SQL → Dataset – SQL+Jinja → Chart

Official example dashboard of Apache Superset



Conclusion: RSE and open technology stack

- Scientific epistemology and ontology
 - Glossaries/taxonomies/ontologies-based data description **inside** data management substack
- System engineering and software engineering
 - SEBoK and SWBoK 4.0 ++ (with **DevOps/DataOps/AIOps/MLOps/SimOps/...**)
- Computational experiments and laboratory experiments automation
 - Mathematical models, grounding, and simulation **in reactive notebooks** with publishing + **5R**
- Programming-in-the-Small/Large/Many/Huge/...
 - Open ecosystems for development, **testing, and deployment**
 - Collaborative interactive research environments **for new productive languages**
- Data Science
 - Data and knowledge management: FAIR4RS and other frameworks
 - Dataset management and data citation **in all affected scientific artefacts**
 - Computational knowledge **as a kind of enterprise assets**
 - Configuration management and artifacts versioning (code, **data, viz, math, ADRs, DoEs, ...**)
- AI as disruptive toolset
 - AI4RSE + empirical/generative software engineering

The end?

- Questions, remarks, comments?

- E-mail: alex.neznanov@gmail.com
- Telegram: [@Alex_Neznanov](https://t.me/Alex_Neznanov)

