Project Large-scale Data Engineering (LDE) Kick-off Meeting

Dr.-Ing. Patrick Damme
Technische Universität Berlin
Berlin Institute for the Foundations of Learning and Data
Big Data Engineering (DAMS Lab)

Last update: Apr 26, 2026

Announcements/Org

- **Hybrid Setting with Optional Attendance**
  - In-person in MAR 0.008
  - Virtual via zoom
    https://tu-berlin.zoom-x.de/j/67376691490?pwd=NmlvWTM5VUVWRjU0UGI2bXhBVkxzQT09

- **Delayed Start of Introductory Lectures**
  - Due to participation in **Dagstuhl Seminar** on
    “Managing Vector Data for Retrieval Augmented Generation”
  - **But:** **No impact** on your total working time, submission deadlines, etc.

About Me

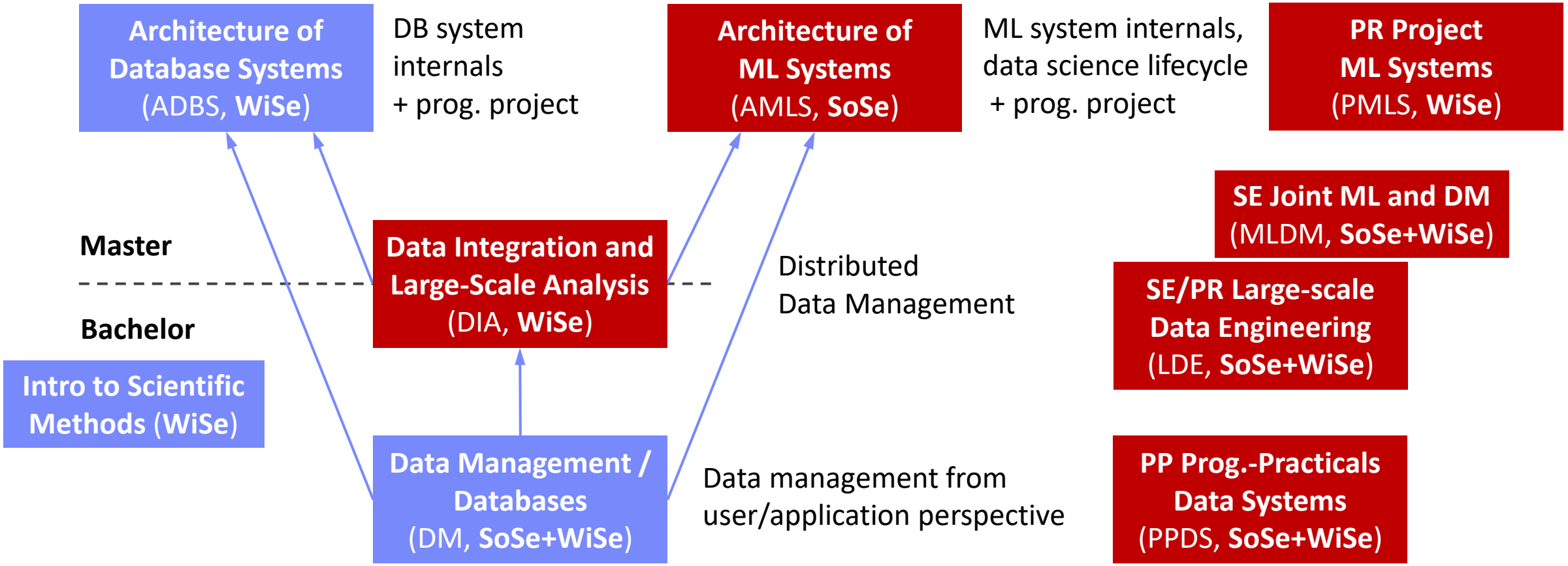
- **Since 10/2022: Postdoc at TU Berlin**, Germany
  - FG Big Data Engineering (DAMS Lab) headed by Prof. Matthias Böhm
  - Continuing work on integrated data analysis pipelines
  - Research interests in the fields of database and ML systems (especially compiler & runtime techniques, extensibility)

- **2021-2022: Postdoc at TU Graz & Know-Center GmbH**, Austria
  - Data Management group headed by Prof. Matthias Böhm
  - Started work on integrated data analysis pipelines

- **2015-2020: PhD student at TU Dresden**, Germany
  - Dresden Database Research Group headed by Prof. Wolfgang Lehner
  - PhD thesis on making complex analytical database queries more efficient through lightweight compression of intermediate results

FG Big Data Engineering (DAMS Lab) – Teaching

Successfully Established TUB Teaching Portfolio (modules, slides)



Agenda

- Course Organization, Outline, and Deliverables
- How to Approach the Project
- Projects in DAPHNE, Apache SystemDS, and TerseTS
- List of Project Topics

Course Organization, Outline, and Deliverables

Large-scale Data Engineering: Module Overview

20 seats in total

master + bachelor

#41086: LDE Seminar + Project (12 ECTS)

10 students

10 students

#41095: Seminar LDE (3)

#41183: Project LDE (9 ECTS)

11 students

bachelor-only

bachelor-only

Mon, 14:00-16:00
MAR 0.008 & zoom

Seminar LDE

- Reading & writing scientific papers
- Giving presentations on papers
- Summary paper
- Presentation
- Lecturer & seminar mentor

Project LDE

- Building & evaluating prototypes
- Giving presentations on prototypes
- Prototype design/impl/tests/doc/eval
- Presentation
- Project mentors

Mon, 16:00-18:00
MAR 0.008 & zoom

→ In the context of systems for data engineering, data management, machine learning

→ In combination: Ideal preparation for a bachelor/master thesis with our group

Course Organization

- **General Contact Person**
  - Dr.-Ing. Patrick Damme (patrick.damme@tu-berlin.de)
- **Course Website**
  - https://pdamme.github.io/teaching/2026_summer/lde/lde_summer2026.html
  - One site for seminar and project
  - All material, schedule, **deadlines**
- **ISIS course**
  - https://isis.tu-berlin.de/course/view.php?id=46572
  - Announcements, discussion forum, topic selection poll, submission of summary paper and presentation slides
- **Language**
  - Lectures and slides: **English**
  - Communication: **English**/**German**
  - Submitted paper and presentation: **English**
  - **Informal language** (first name is fine), immediate feedback is welcome

Semester Schedule & Deadlines

- **Kick-off Meeting Apr 27** (optional)
- **Recommended Introductory Lecture** (optional)
  - May 11, 14:00: Experiments, Reproducibility, and Giving Presentations
- **Self-organized Project Work**
  - Consultation hours for any questions (optional)
- **Intermediate Presentations** (prerequisite)
  - Jun 22, 16:00-18:00: All students
- **Final Presentations** (mandatory)
  - Aug 03, 14:00-18:00: All students

- **List of Project Topics**
  - Presented today, take your time to select afterwards
- **Topic Selection**
  - **Deadline: May 04, 23:59** (in 1 week)
  - Ranked list of **5 topics** via poll on the ISIS course + pref on individual/team work [+ team members]
  - Global topic assignment based on preferences
  - **Notification of assigned topics: May 11** (in 2 weeks)
- **Submission of Initial Prototype** (prerequisite)
  - Implementation and tests
  - **Deadline: Jun 21, 23:59** (in 8 weeks)
  - As a pull request on GitHub (exceptionally by email)
- **Submission of Final Prototype** (mandatory)
  - Implementation, tests, docs, experiments
  - **Deadline: Aug 02, 23:59** (in 14 weeks)
  - As a pull request on GitHub (exceptionally by email)
- **Submission of Pres. Slides (Intermediate & Final Pres.)**
  - **Deadline: The day before the presentation, 23:59**
  - Upload PDF in the ISIS course

Portfolio Exam Registration

- **Registration: May 11 – Jun 08**
  - Binding registration in Moses/MTS
  - Including selection of seminar presentation date (first-come-first-serve)
- **De-registration**
  - **Until 3 days before the first graded exam part**
    - De-register yourself in Moses/MTS
    - Modules “LDE”/”Seminar LDE”: until **Jun 12**
    - Module “Project LDE”: until **Jul 30**
  - **With sufficient reason: Until the day of the exam**
    - In case of sickness etc.
    - Modules “LDE”/”Seminar LDE”: until **Jun 15/Jun 29/Jul 06**
    - Module “Project LDE”: until **Aug 02/Aug 03**
- **Missing Deadlines/Exam Without De-registration**
  - Zero points in the respective exam part (!)
  - **Approach us early in case of problems**
- **If You Don’t Want to Take LDE Anymore**
  - Let me know asap to give students in the queue a chance to fill in

LDE Project Goals and Mindset

- **Goals**
  - **Design/implement a prototype** in DAPHNE/SystemDS/TerseTS
  - **AND prove it is a valuable contribution** to the system/library (tests, documentation, experiments)
  - **Present and defend your work** in a presentation & discussion

  } **High-quality and convincing contribution to an open-source system**

- **Focus on Methodology**
  - LDE as a preparation for a bachelor/master thesis at the DAMS Lab
- **Mindset: Be Open Learn New Things and to Work on Any Part of the System**
  - Whatever it takes to fulfill the task
  - Self-guided acquisition of required technical skills
- **Grading Criteria**
  - Design/implementation (functionality)
  - Code quality + tests + documentation
  - Experiments

LDE Project Characteristics

- **Individual/Team Project Work**
  - Teams of up to 3 students **strongly encouraged**
  - Unique topic for each individual/team
- **Ambitious Projects**
  - 9 ECTS (~270 h of work)
  - **≈6.75 weeks of full-time work**
- **Potential for Impact**
  - Real open issues in existing systems
  - If successful: meaningful contributions that will be used by others

- **Remarks on Topic Descriptions**
  - Many open topics in DAPHNE, SystemDS, and TerseTS
  - Initial topic descriptions of varying level of detail
  - During topic selection: Approach project mentor directly if interested in more details
  - After topic assignment: More detailed descriptions where necessary
  - We're open to alternative topic proposals

Project Deliverables: Initial Prototype & Intermediate Presentation

- **Introduced in Response to Students' Feedback (Course Evaluation)**
- **Initial Prototype**
  - ≈67% functionally complete prototype including good set of test cases
  - Basis for further improvements driven by experiments and feedback
- **Intermediate Presentation**
  - Slide presentation of 5-10 min per individual/team
  - Briefly **present the problem** you work on **to your fellow students**
  - Give an **overview of your initial prototype** (concepts and crucial changes to the code base)
  - Outline your **planned experiments**
  - Should be the result of **prior discussions with your project mentor**

Ungraded prerequisites for the portfolio exam
to be allowed make mistakes and learn from them

- **Benefits for You**
  - **Improved time management** (retain enough time for experiments)
  - **Exchange with the other students** in the project
  - **Get feedback** by project mentors and other students for improving the **quality of your prototype**

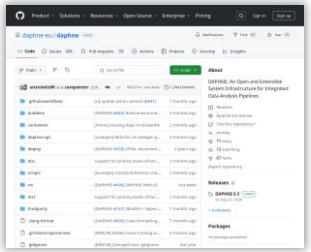
Project Deliverables: Final Prototype & Final Presentation

- **Final Prototype**
  - 100% functionally complete prototype including good set of test cases
  - Efficiency confirmed by experiments
- **Final Presentation**
  - **Summarize the problem** and give an **overview of your final prototype** **to your fellow students**
  - Present your **experimental results**
  - **1 student: 10 min talk + 5 min discussion = 15 min**
  - **2 students: 13 min + 7 min = 20 min**
  - **3 students: 16 min + 9 min = 25 min**
  - **Audience: Engage in the discussion**
- **Academic Honesty / No Plagiarism**
  - Develop **your own** coding/experimentation skills
  - Use of **generative AI** (ChatGPT etc.) **is prohibited**
- **Graded Portfolio Exam**
  - **#41086 (seminar + project)**
    - 25 pts: summary paper
    - 15 pts: presentation
    - 50 pts: design/impl/tests/doc
    - 10 pts: presentation
  - **#41183 (project-only)**
    - 85 pts: implementation/tests/documentation
    - 15 pts: presentation

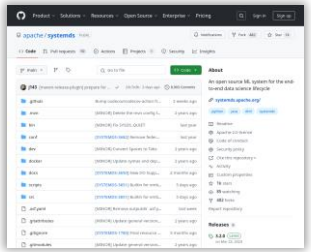
How to Approach the Project

Getting Started: Setting Up Your Development Environment

- **Goals**
  - Build the system from source
  - Successfully run the test suite
- **Navigate to the GitHub Repos**

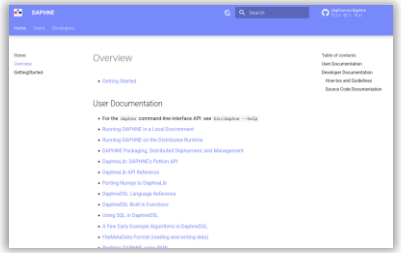


https://github.com/daphne-eu/daphne



https://github.com/apache/systemds

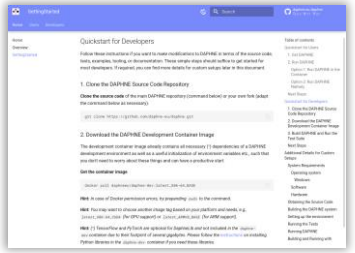
- **Know Where to Find the Documentation**



https://daphne-project.github.io/daphne/

https://apache.github.io/systemds/

- **Clone, Build, Test According to the Documentation**

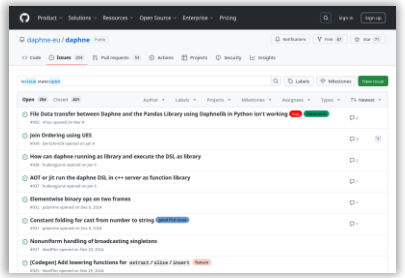


https://daphne-project.github.io/daphne/GettingStarted/#quickstart-for-developers

https://apache.github.io/systemds/site/install.html

Getting Started: Preparing Your First Contribution

- **Goals**
  - Ability to modify the source code and run/test it
  - Initial overview of relevant part of the code base
- **Set Up Your Editor/IDE etc.**
- **Issue Tracking**

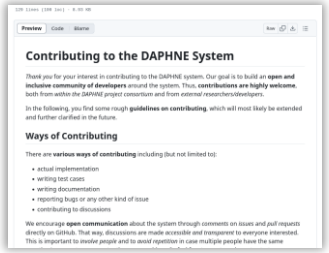


https://github.com/daphne-eu/daphne/issues

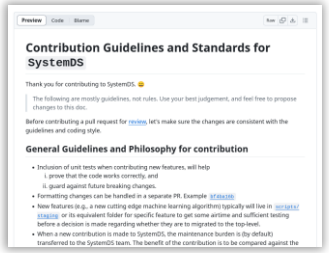
https://issues.apache.org/jira/secure/Dashboard.jspa?selectPageId=12335852

- **Make Your First Modifications to the Code**
  - DAPHNE: “good first issues”
  - SystemDS: Write initial test cases for your task

- **Read the Contribution Guidelines**



https://github.com/daphne-eu/daphne/blob/main/CONTRIBUTING.md



https://github.com/apache/systemds/blob/main/CONTRIBUTING.md

Towards the Initial Prototype: Design/Implementation/Tests (≈1/2 of the semester)

- **Recap: Goals**
  - ≈67% functionally complete prototype including good set of test cases
  - Basis for further improvements driven by experiments and feedback
- **Mindset**
  - **Understand the topic** (task description, mentor, additional material)
  - **Understand the code base** (overview and relevant parts)
  - **Understand the employed libraries/frameworks**
  - **Design and implement** step-by-step
  - Not always “the” right solution: **explore alternatives**
- **Recommendations**
  - **Start as soon as possible**
    → Don’t underestimate the ramp-up effort
  - **Actively approach your project mentor**
    → Your mentor can give you valuable guidance

Towards the Final Prototype: Creating a Convincing Contribution (≈1/2 of the semester)

- **Recap: Goals**
  - High-quality code contribution whose value can easily be appreciated and understood
- **Recommendations**
  - **Submit tidy, well documented, extensively tested code**
  - **View experiments as equally important as features**,
    → Start initial experiments as soon as possible
    → Focus primarily on investigating and improving your initial prototype after the intermediate presentation
    → **Experiments show the value of your contribution**
    → **You need time to incorporate your insights**

    Actively approach your project mentor
    → Your mentor can give you valuable guidance

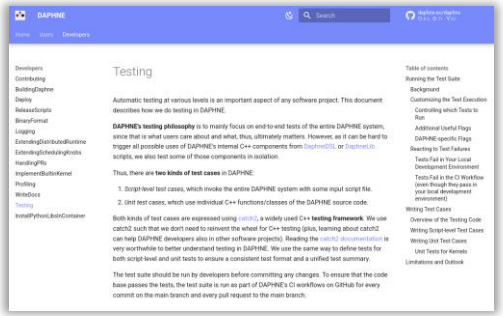
Final Prototype: Code Quality

- **Goals**
  - Make your code easy to read/understand
  - Others will have to maintain it after your contribution is merged
- **General Guidelines**
  - Clearly structure your code into **meaningful units** (classes, functions, etc.)
  - Use **clear yet concise identifiers** (variable/function/class names)
  - Stay **consistent with the existing code base** (e.g., use the same patterns) **OR** **refactor if necessary**
  - Adhere to the code base's **coding style/formatting**
- **Keep Your Pull Request Tidy**
  - Stay focused: **Avoid changes unrelated to your task** (can be contributed as individual small pull requests)
  - **Don't submit anything that's useless for others** (e.g., build artifacts, generated files (e.g., logs), IDE projects)
  - **Exclude specifics of your local setup**: Avoid local paths, usernames, passwords, IP addresses etc.
  - **Read your own pull request on GitHub** (changed files)

Final Prototype: Tests

- **Goals**
  - Show functionally correct behavior
  - Experiments don't make sense
    if prototype doesn't do what it should
- **General Hints**
  - Unit tests and script-level tests
  - **Test cases that should work**
  - **Test cases that should not work**
    (e.g., invalid DSL scripts, invalid input data, …)
  - Construct simple and complex scenarios
  - Think of **corner cases**
  - Small input data is often fine
    (but some bugs only triggered by large inputs)

- **Integrate with the Existing Test Suite**
  - **DAPHNE**
    → directory: `test/`
    → see the documentation on writing test cases



https://daphne-project.github.io/daphne/development/Testing/

  - **SystemDS**
    → directory: `src/test/`

Final Prototype: Documentation

- **Goals**
  - Make your contribution understandable for **users** and **developers**
- **User Documentation**
  - **What's not documented doesn't exist**
  - Add **high-level explanation** of features and concepts behind them
  - Update documentation of **language abstractions** (e.g., new DSL built-in functions, new types, …)
  - Update documentation of **user APIs** (e.g., new command-line arguments)
- **Developer Documentation**
  - **Negative example: "The code is the documentation"**
  - **High-level explanation** of your contribution; justify design decisions
  - **API documentation** of classes, functions, members, etc.
    (integrate with the existing source code documentation of the system, e.g., doxygen or javadoc style)
  - **Comments within function bodies** (e.g., high-level steps of an algorithm)

Final Prototype: Experiments

- **Goals**
  - Understand your prototype to discover potential for improvement (e.g., performance bottlenecks)
  - Demonstrate functional improvements (new features)
  - Showcase non-functional improvements (performance compared to status quo and state-of-the-art baselines)
- **Design Experiments**
  - **Don’t just conduct any random experiments**
  - **Think about which questions you want/need to answer**, **design experiments accordingly**
  - **Types of experiments**: Exploratory, micro benchmarks, benchmarks, end-to-end applications
  - **Aspects of an experiment**: Data, workload, baselines, hardware & software stack, metrics
- **Conduct Experiments**
  - Automate as much as possible for repeatability (shell scripts etc.)
- **Visualize and Interpret the Results**
  - Automate the visualization based on raw experimental data
  - Draw conclusions and react

See 3rd seminar intro lecture on “Experiments, Reproducibility”

Projects in DAPHNE, Apache SystemDS, and TerseTS

Overview (1/2): Two Systems Developed at the DAMS Lab

DAPHNE

- **An Open and Extensible System Infrastructure for Integrated Data Analysis Pipelines**
  - Intersection of data management, machine learning, high-performance computing
  - Open-source (Apache v2 license) https://github.com/daphne-eu/daphne
  - Originated from DAPHNE EU-project
  - Written mostly in C++, Python, DaphneDSL
  - Since 2020 (open-source since 2022)

[Patrick Damme et al.: DAPHNE: An Open and Extensible System Infrastructure for Integrated Data Analysis Pipelines. CIDR 2022]

THE ASF Apache SystemDS™

- **A Declarative ML System for the End-to-End Data Science Lifecycle**
  - Data integration/cleaning/prep, model selection/ training/validation/debugging/deployment/scoring
  - Open-source (Apache v2 license) https://github.com/apache/systemds
  - Originated from Apache SystemML (started at IBM)
  - Written mostly in Java, Python, and DML
  - Since 2010 (open-source since 2015) (as SystemML)

[Matthias Boehm et al.: SystemDS: A Declarative Machine Learning System for the End-to-End Data Science Lifecycle. CIDR 2020]

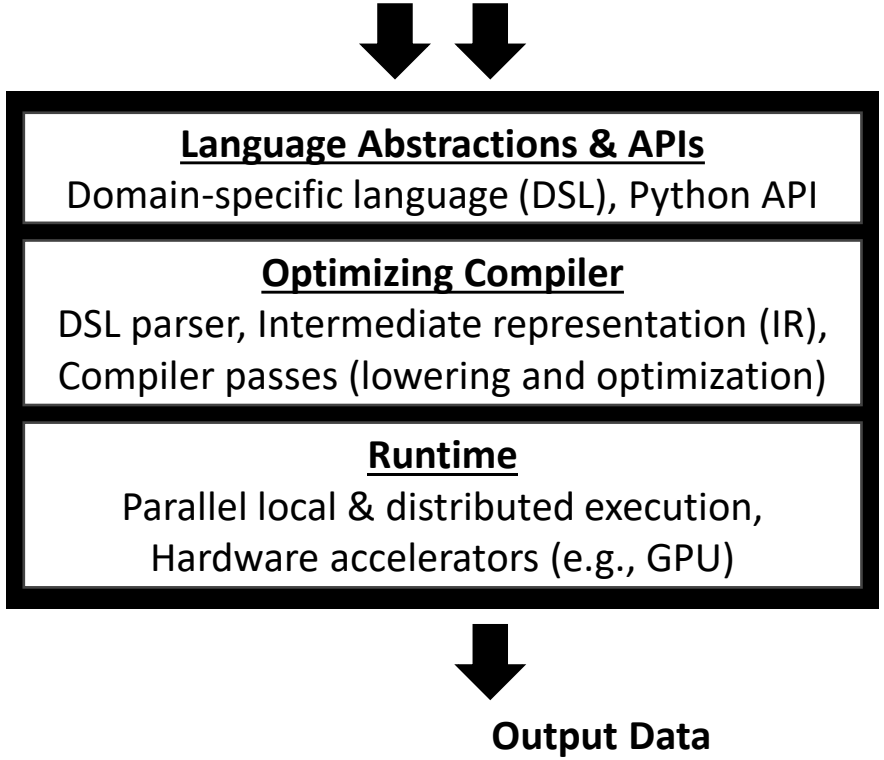
BIFOLD

Simplified High-level Architecture of DAPHNE and SystemDS

clustering … neural networks
regression …
classification …
relational queries

Program
matrix/frame data types
linear/relational algebra ops
complex control flow

Input Data
open file formats
data exchange with other systems/libs



Output Data

Other such systems (examples)

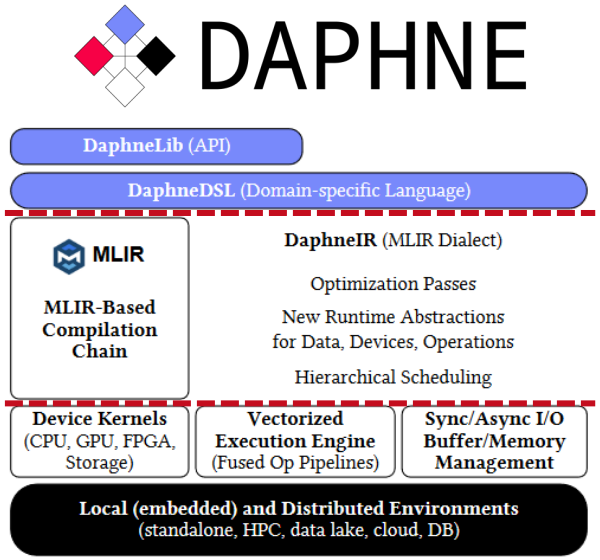
NumPy DASK julia
PYTORCH R TensorFlow

Architecture of ML Systems **(AMLS): Lecture + Exercise by Prof. Matthias Böhm** (every summer semester)

Simplified High-level Architecture of DAPHNE and SystemDS

Program
matrix/frame data types
linear/relational algebra ops
complex control flow

Input Data
open file formats
data exchange with other systems/libs

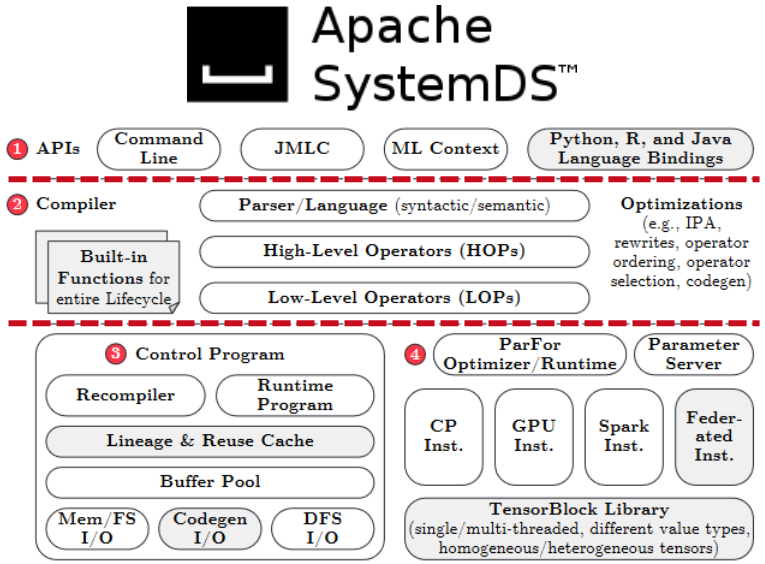


Language Abstractions & APIs
Domain-specific language (DSL), Python API

Optimizing Compiler
DSL parser, Intermediate representation (IR),
Compiler passes (lowering and optimization)

Runtime
Parallel local & distributed execution,
Hardware accelerators (e.g., GPU)

Output Data

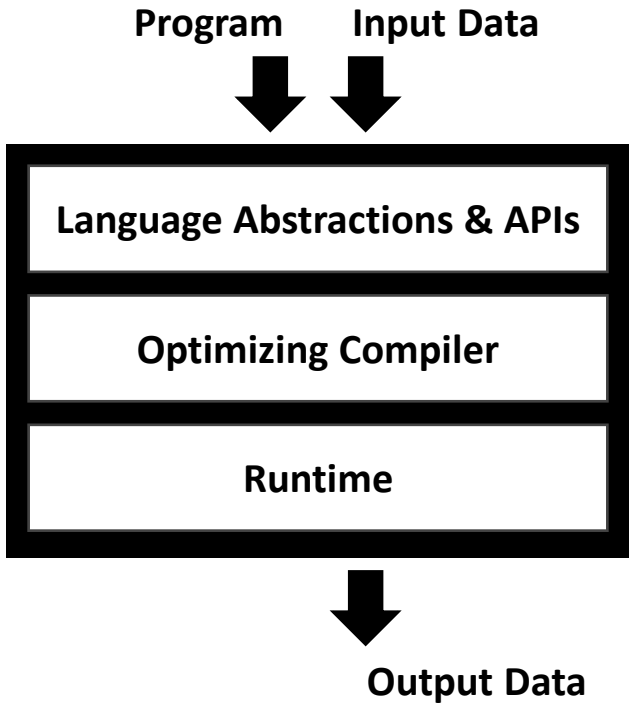


Kinds of LDE Project Topics in Daphne and SystemDS

End-to-end Experiments
typically required in any project topic

External Tooling
- Helpers in the ecosystem
- Any suitable prog. lang.

Openness to All Levels of the Stack
to fulfill the task



Reusable Data Science Primitives
- Data analysis algorithms/applications
- Mostly in DaphneDSL/DML

System Internals
- Optimization and efficient execution of input programs (various aspects)
- Mostly in C++/Java, some also in Python

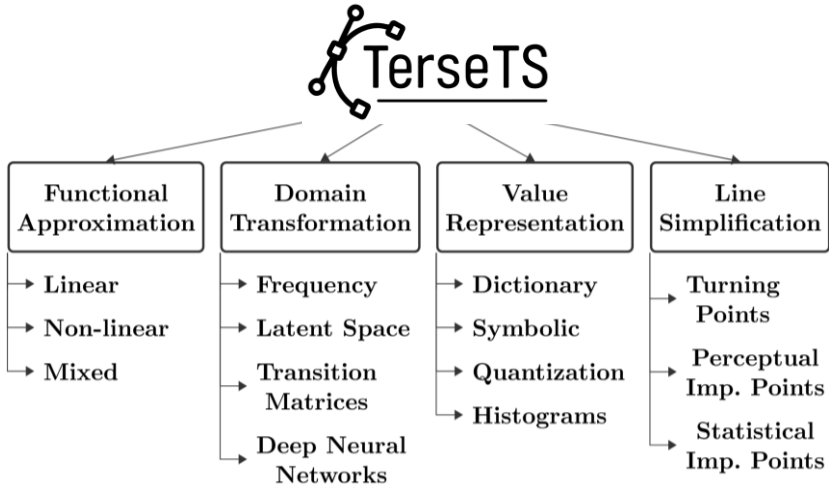
Choose topic based on your interests and technical skills

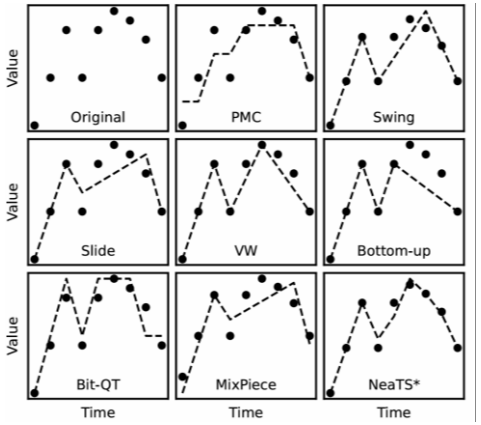
Overview (2/2): A Framework Developed at the DAMS Lab

TerseTS

- **A Framework for Time Series Compression**
  - Native library for efficient time series lossy and lossless compression
  - Lowers the barrier for compression method evaluation, extension, and deployment while improving reproducibility in compression research
  - Open-source (Apache v2 license)
    https://github.com/cmcuza/TerseTS
  - Multi-platform: optimized for portability and efficiency on MacOS, Windows, and Linux
  - Written mostly in Zig, with bindings for C, Julia, Python, and Rust

[Carlos E. Muñiz-Cuza et al.: TerseTS: A Framework for Time Series Compression. EDBT 2026]
EDBT 2026 Best Demo Award Honorable Mention





List of Project Topics (Proposals)

See list at https://pdamme.github.io/teaching/2026_summer/lde/ProjectTopics.pdf

Summary and Q&A

- Course Organization, Outline, and Deliverables
- How to Approach the Project
- Projects in DAPHNE, Apache SystemDS, and TerseTS
- List of Project Topics (Topic Selection by May 04)

- Remaining Questions?

- Reminder: Seminar Introductory Lecture Recommended for the Project
  - 03 Experiments, Reproducibility, and Giving Presentations [May 11, 14:00]

- See you during the consultation hours and intermediate presentations ☺