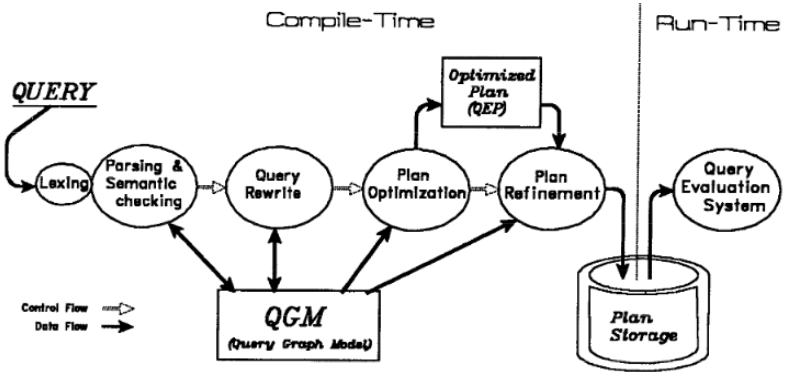
Towards a Modular Data Management System Framework

Haralampos Gavriilidis, **Lennart Behme**, Sokratis Papadopoulos, Stefano Bortoli, Jorge-Arnulfo Quiané-Ruiz, Volker Markl

September 9, 2022

A Brief Retrospect of Data Management Systems



Extensible Query Processing in Starburst

Laura M. Haas, J.C. Freytag[1], G.M. Lohman, and H. Pirahesh
IBM Almaden Research Center, San Jose, CA 95120

- Same architecture since 80s
- Systems are monoliths

Modern Data Management

- Stonebraker:
  “One size does not fit all”
- Observation:
  Modern DMSes share many components

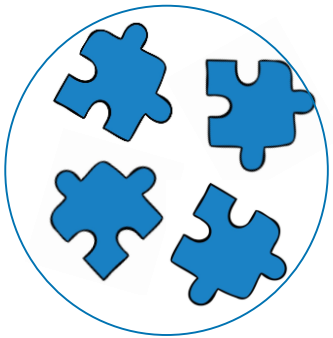
When building systems, we keep reinventing the wheel!

An Example of System Requirements in Industry

- Setup
  - Data in legacy systems
  - Mix of cloud & on-prem systems
- Goal
  - ETL over multiple systems
  - Optimize operator placement
  - Expose SQL interface

Should we build a new system from scratch?

Many Existing Solutions for DMS Components

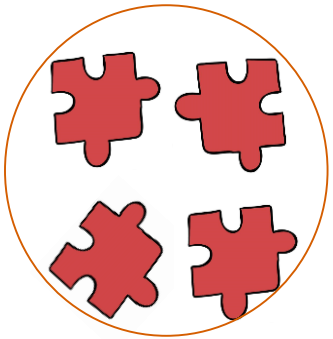


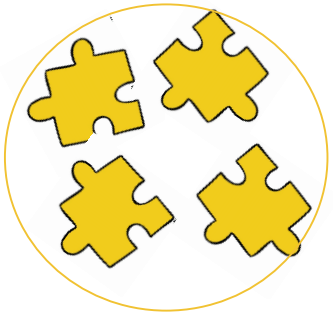
Query Interfaces

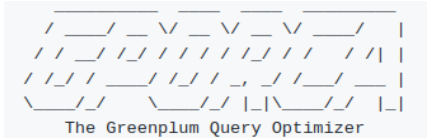
Optimizers

Execution Engines

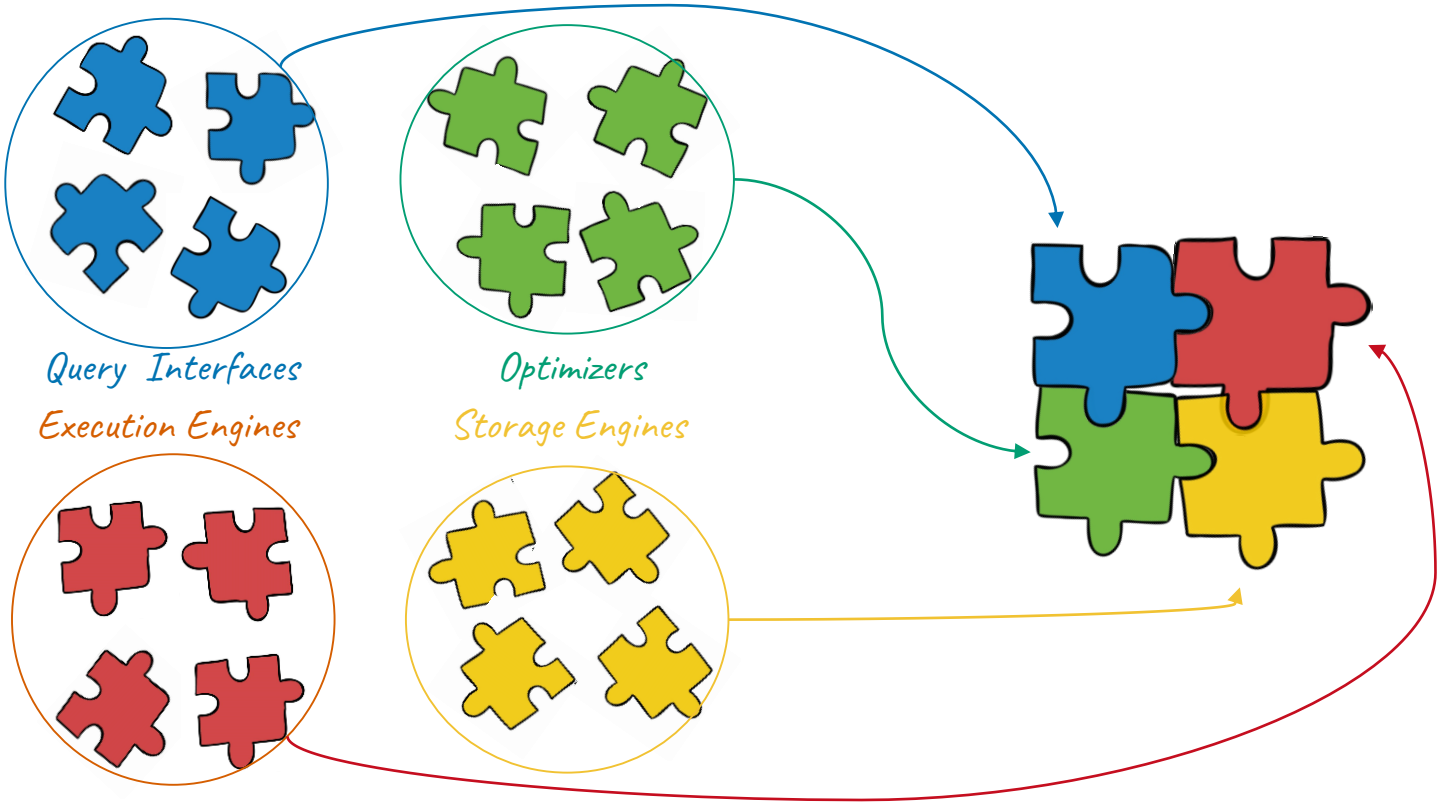
Storage Engines







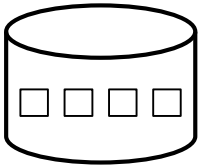
Idea: Compose DMS out of existing Components



Current Composability Challenges

Missing standards for component interfaces

Manual efforts by high-skilled system engineers



Inflexible one-purpose DMS

PolyDMS – Our Vision to Address the Composability Gap

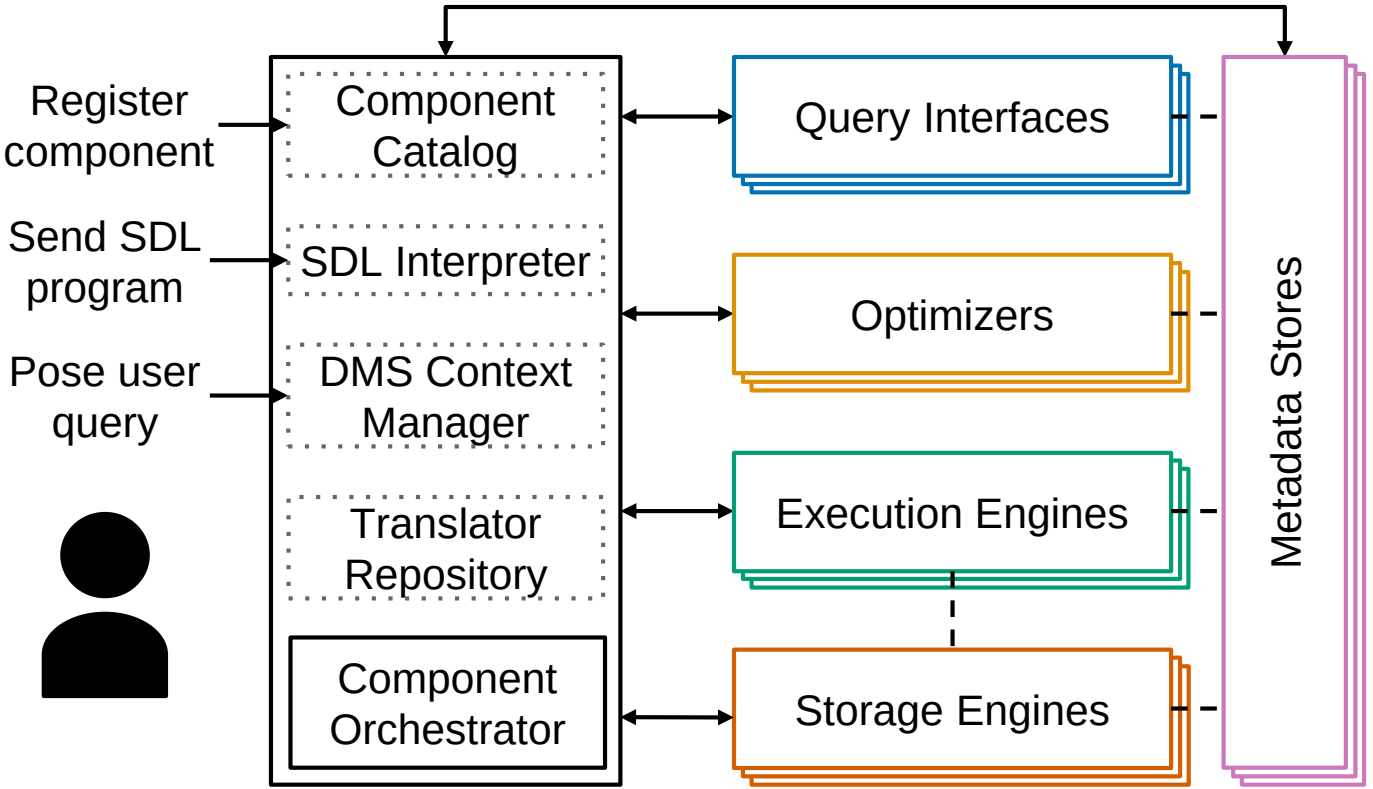
Plug & Play DMS Architecture

Standardized component types & well defined interfaces

System Definition Language (SDL)

Component Orchestrator

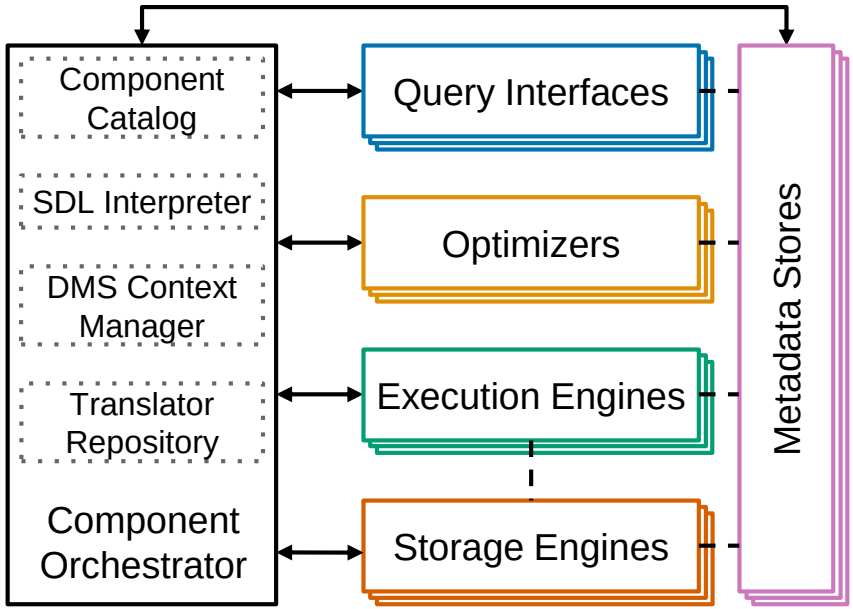
PolyDMS – General Architecture



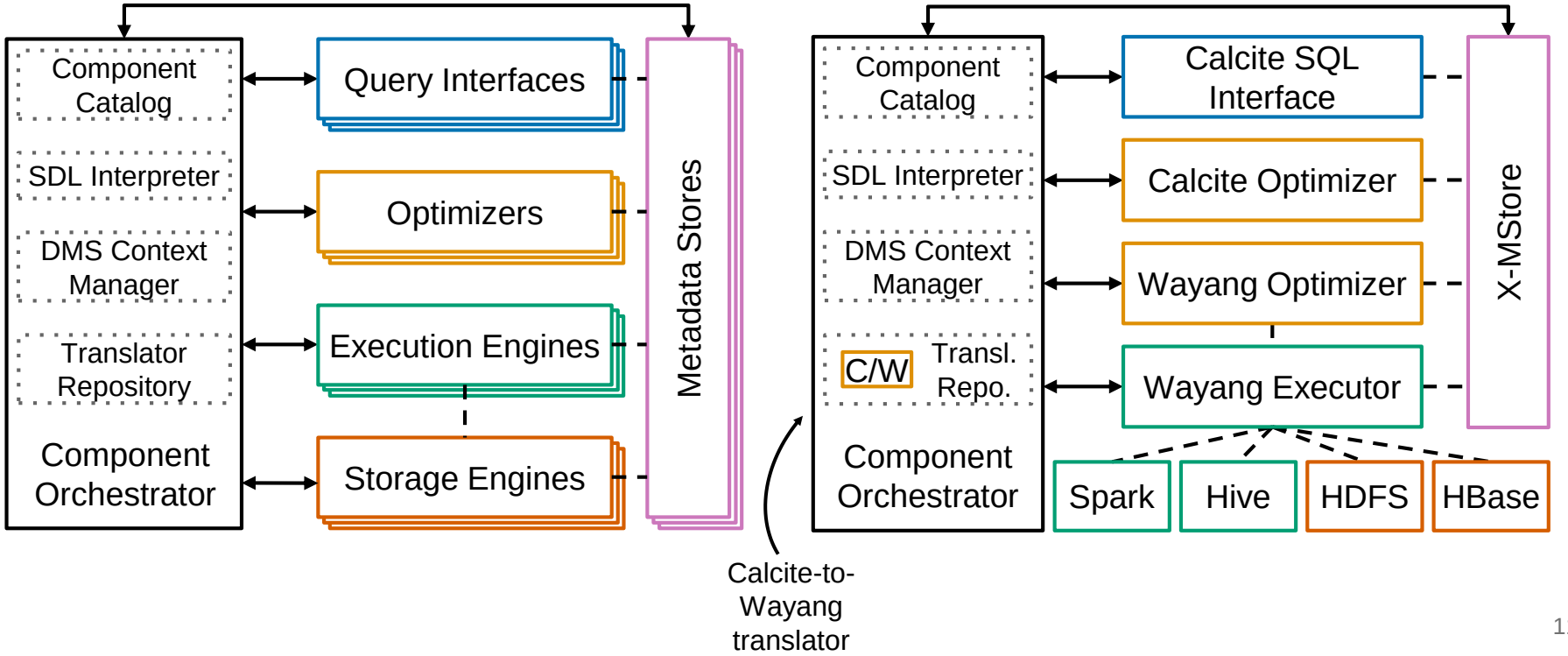
Proof-of-Concept: Combine SQL & Cross-Platform Worlds

Recall requirements:

- SQL user interface
- Optimal operator placement
- Cross-platform execution



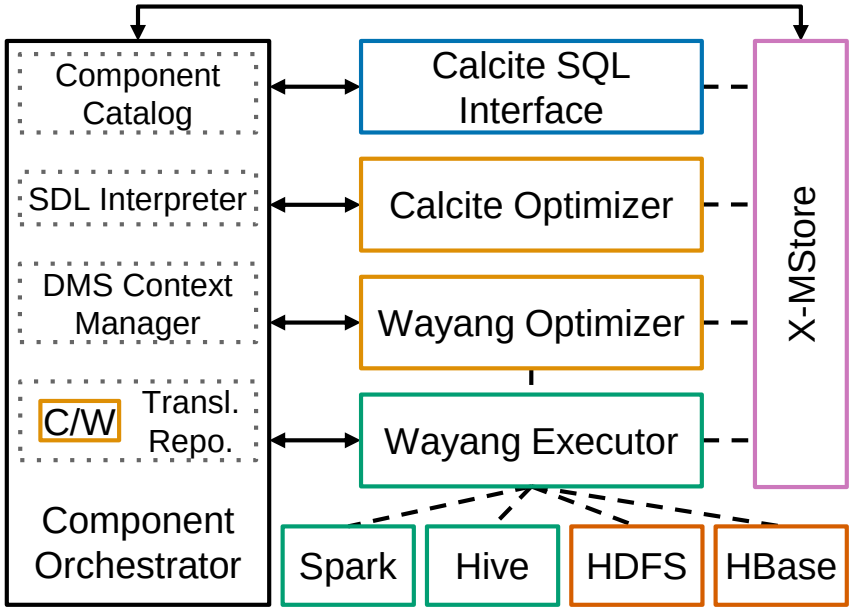
Proof-of-Concept: Combine SQL & Cross-Platform Worlds



Proof-of-Concept: Combine SQL & Cross-Platform Worlds

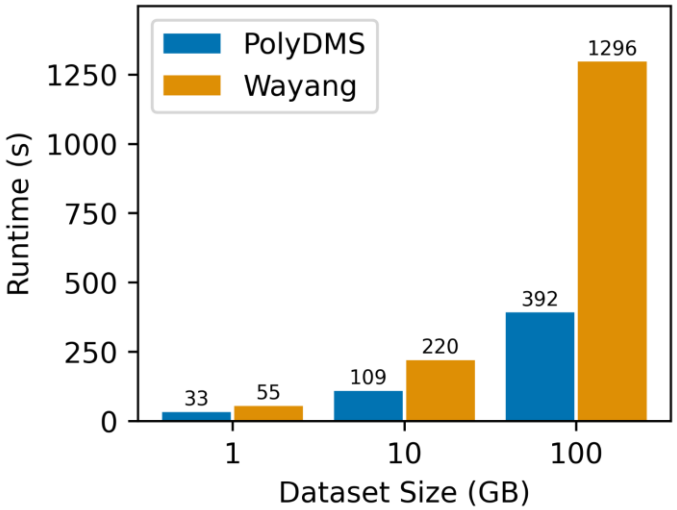
```
def cross_platform_dms(input):
  ctx = PolyDMSContext()
  ctx.md = XMStore()
  ctx.qi = CalciteSqlInterface(input,  ctx.md)
  ctx.l_opt = CalciteOpt(ctx.qi, ctx.md)
  ctx.w_opt = WayangOpt(translate(ctx.l_opt), ctx.md)
  ctx.exec = WayangExec(ctx.cp_opt, ctx.md)

  return ctx.initialize()
```

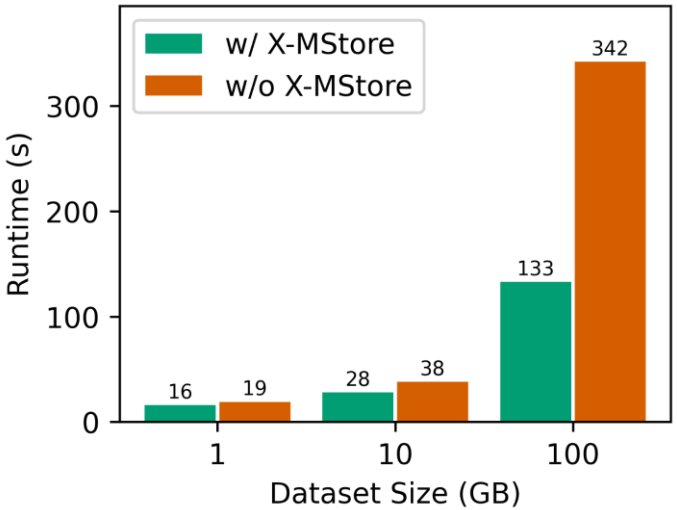


Preliminary Evaluation

Optimization synergies



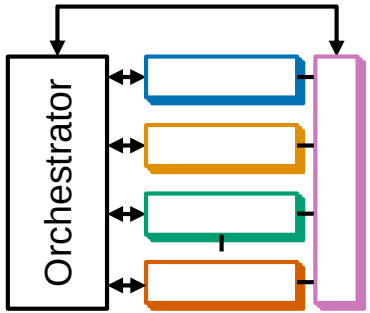
Metadata impact



PolyDMS achieves better performance without altering engines!

Towards a Modular Data Management System Framework

- Goal: Break DMS monoliths
  - Compose DMS out of existing frameworks
  - Eases development efforts for new DMSes
  - Fosters re-usability w.r.t. components



- Next: Refine SDL, intermediate representations, interfaces

Haralampos Gavriilidis
gavriilidis@tu-berlin.de
www.user.tu-berlin.de/harry_g

Lennart Behme
lennart.behme@tu-berlin.de
www.lbeh.me