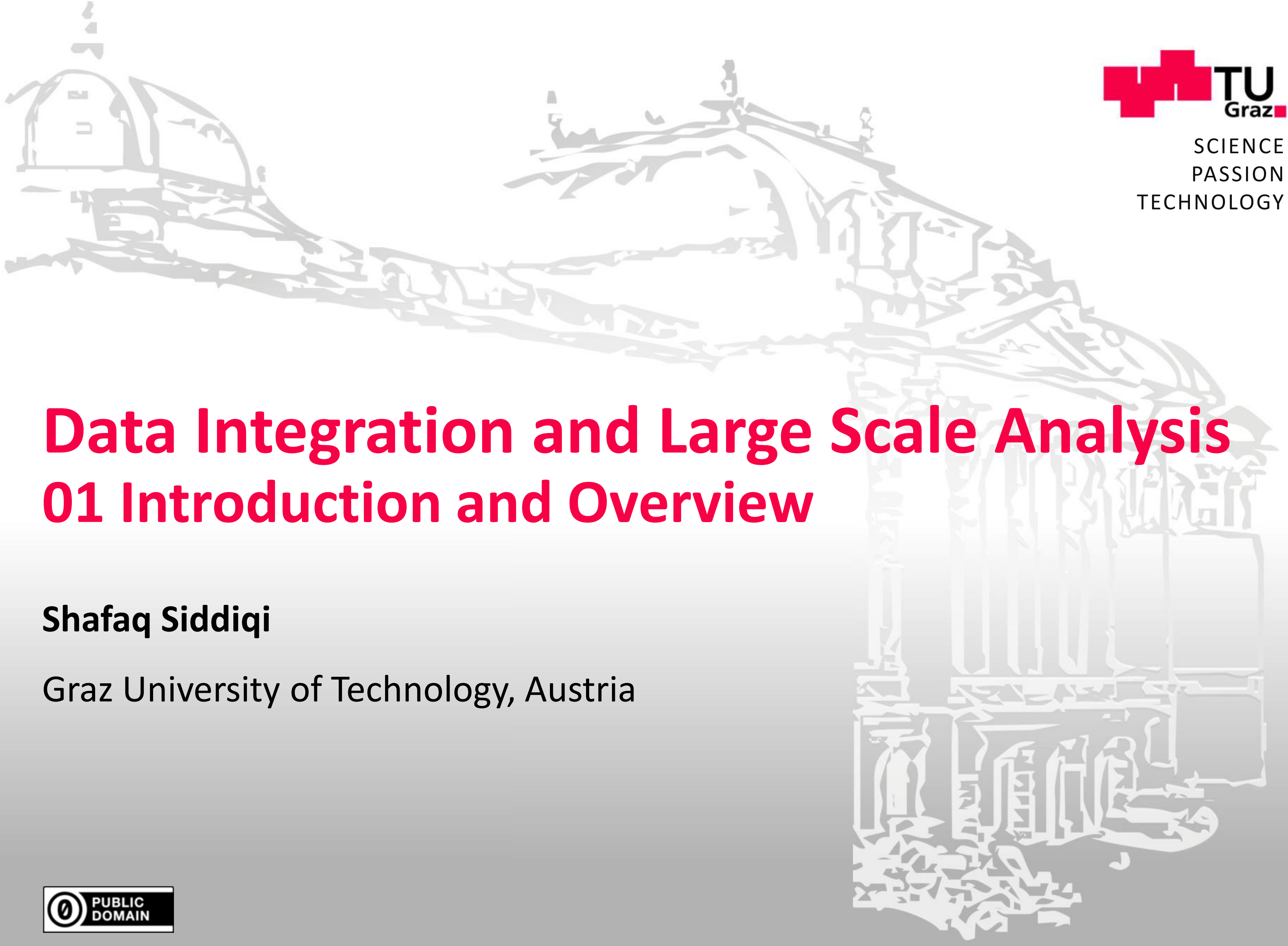


Data Integration and Large Scale Analysis
01 Introduction and Overview

Shafaq Siddiqi

Graz University of Technology, Austria

Last update: Oct 07, 2022

Announcements/Org

- **#1 Video Recording**
  - Link in **TUbe** & **TeachCenter**
  - Optional attendance (independent of COVID)
  - **Hybrid**, in-person but video-recorded lectures
    - **HS i5** or Webex: https://tugraz.webex.com/meet/shafaq.siddiqi

- **#2 Course Registration** (as of Oct 07)
  - **Data Integration and Large-Scale Analysis** (DIA)

WS20/21:	**96 (2)**
WS21/22:	**122 (4)**
WS21/22:	**134 (2)**

Agenda

- **Course Organization**
- **Course Outline and Projects**
- **Course Motivation and Goals**
- **Excursus: Apache SystemDS**

About Me

- **09/2019 TU Graz**, Austria
  - Teaching Assistant, TU Graz
  - **Institute of Interactive Systems and Data Science, CSBME** (ML systems internals, end-to-end data science lifecycle)

    https://github.com/apache/systemds

- **2017-2019 Sukkur IBA University**
  - Lecturer (Computer Science)
  - Teaching and supervising FYPs in Bachelor programs

- **2020 PhD Student TU Graz**, Austria
  - Data preprocessing for Heterogeneous Large Scale Data
  - Generation and Optimization of Data Cleaning Pipelines
  - https://damslab.github.io/

Data Management group

Course Organization

Basic Course Organization

- **Staff**
  - Lecturer: M.Sc. Shafaq Siddiqi, ISDS
- **Language**
  - Lectures and slides: **English**
  - Communication and examination: **English**
- **Course Format**
  - VU 2/1, **5 ECTS** (2x 1.5 ECTS + 1x 2 ECTS), bachelor/master
  - **Weekly lectures** (**Fri 3pm**, including **Q&A**), **attendance optional**
  - **Mandatory exercises or programming project** (2 ECTS)
  - **Recommended papers** for additional reading on your own
- **Prerequisites**
  - **Preferred:** course Data Management / Databases is very good start
  - **Sufficient:** basic understanding of SQL / RA (or willingness to fill gaps)
  - Basic programming skills (Python, R, Java, C++)

Course Logistics

- **Website**
  - https://shafaq-siddiqi.github.io./dia2022.html
  - All course material (lecture slides) and dates
- **Video Recording Lectures (TUbe)?**
- **Communication**
  - **Informal language** (first name is fine)
  - Please, **immediate feedback** (unclear content, missing background)
  - Newsgroup: N/A – email is fine, TeachCenter forum for discussions
  - **Office hours:** Tuesday 11:00 am – 12:00 pm & by appointment or after lecture
- **Exam**
  - **Completed exercises or project**
  - **Final written exam** (oral exam if <25 students take the exam)
  - **Grading** (30% project/exercises completion, 70% exam)

Course Logistics, cont.

- **Course Applicability**
  - Bachelor programs computer science (CS), as well as software engineering and management (SEM)
  - Master programs computer science (CS), as well as software engineering and management (SEM)
    - Catalog Data Science: **compulsory** course in major/minor
  - **Free subject course** in any other study program or university

Course Outline and Projects

Part A: Data Integration and Preparation

Data Integration Architectures

- **01 Introduction and Overview** [Oct 07]
- **02 Data Warehousing, ETL, and SQL/OLAP** [Oct 14]
- **03 Message-oriented Middleware, EAI, and Replication** [Oct 21]

Key Integration Techniques

- **04 Schema Matching and Mapping** [Oct 28]
- **05 Entity Linking and Deduplication** [Nov 04]
- **06 Data Cleaning and Data Fusion** [Nov 11]

Part B: Large-Scale Data Management & Analysis

Cloud Computing

- **07 Cloud Computing Foundations** [Nov 18]
- **08 Cloud Resource Management and Scheduling** [Nov 25]
- **09 Distributed Data Storage** [Dec 02]

Large-Scale Data Analysis

- **10 Distributed, Data-Parallel Computation** [Jan 13]
- **11 Distributed Stream Processing** [Jan 20]
- **12 Distributed Machine Learning Systems** [Jan 27]

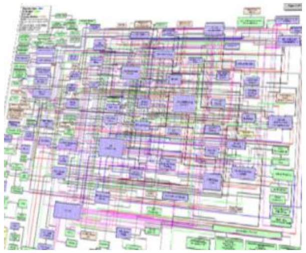
Overview Projects or Exercises

- **Team**
  - **1-3 person teams** (w/ clearly separated responsibilities)
- **Objectives**
  - Non-trivial programming project in DIA context (**2 ECTS → 50 hours**)
  - **Exercise:** Data engineering and ML pipeline
    - Data cleaning and integration of multi-modal data sources
    - ML model training and evaluation
  - **Optional:** Open source contribution to **Apache SystemDS** https://github.com/apache/systemds (from HW to high-level scripting)
- **Timeline**
  - **Oct 21:** Exercise description
  - **Jan 13:** Final project/exercise deadline

Course Motivation and Goals

Data Sources and Heterogeneity

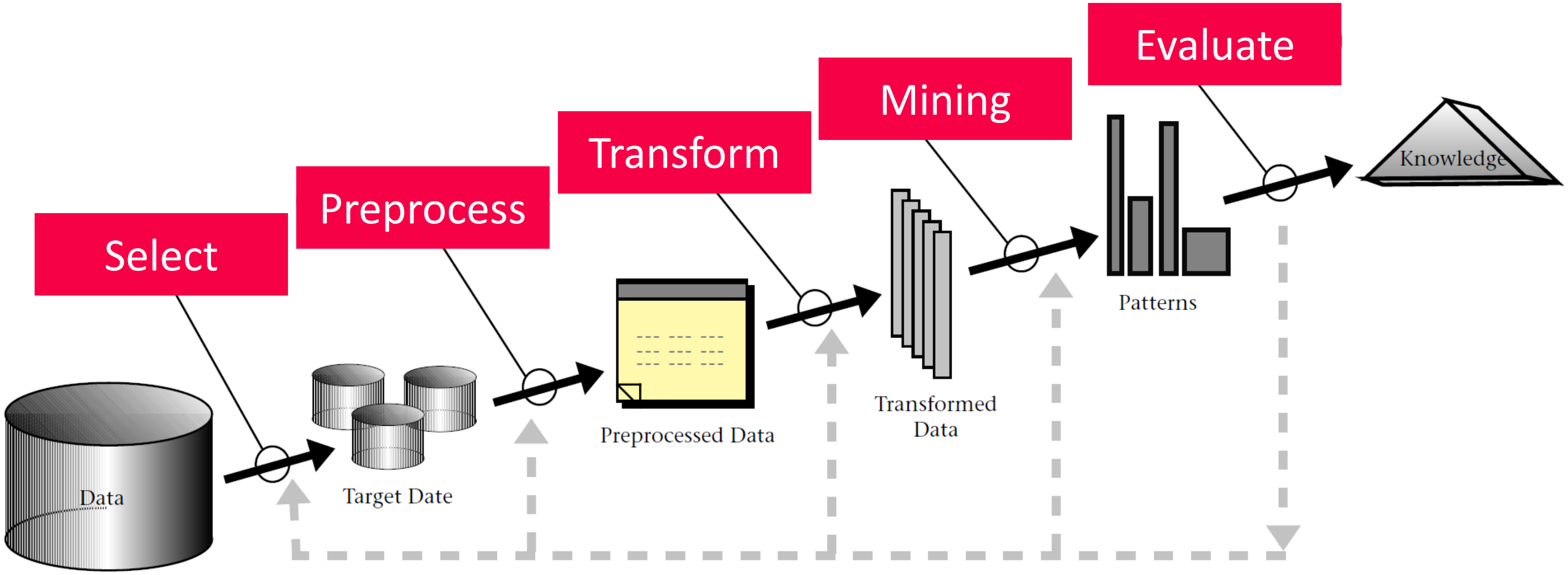
- **Terminology**
  - **Integration** (Latin integer = whole): consolidation of data objects / sources
  - **Homogeneity** (Greek homo/homoios = same): similarity
  - **Heterogeneity**: dissimilarity, different representation / meaning
- **Heterogeneous IT Infrastructure**
  - Common enterprise IT infrastructure contains >100s of **heterogeneous and distributed systems and applications**
  - E.g., health care data management: 20 - 120 systems



- **Multi-Modal Data (example health care)**
  - Structured patient data, patient records incl. prescribed drugs
  - Knowledge base drug APIs (active pharmaceutical ingredients) + interactions
  - Doctor notes (text), diagnostic codes, outcomes
  - Radiology images (e.g., MRI scans), patient videos
  - Time series (e.g., EEG, ECoG, heart rate, blood pressure)

The Data Science Lifecycle

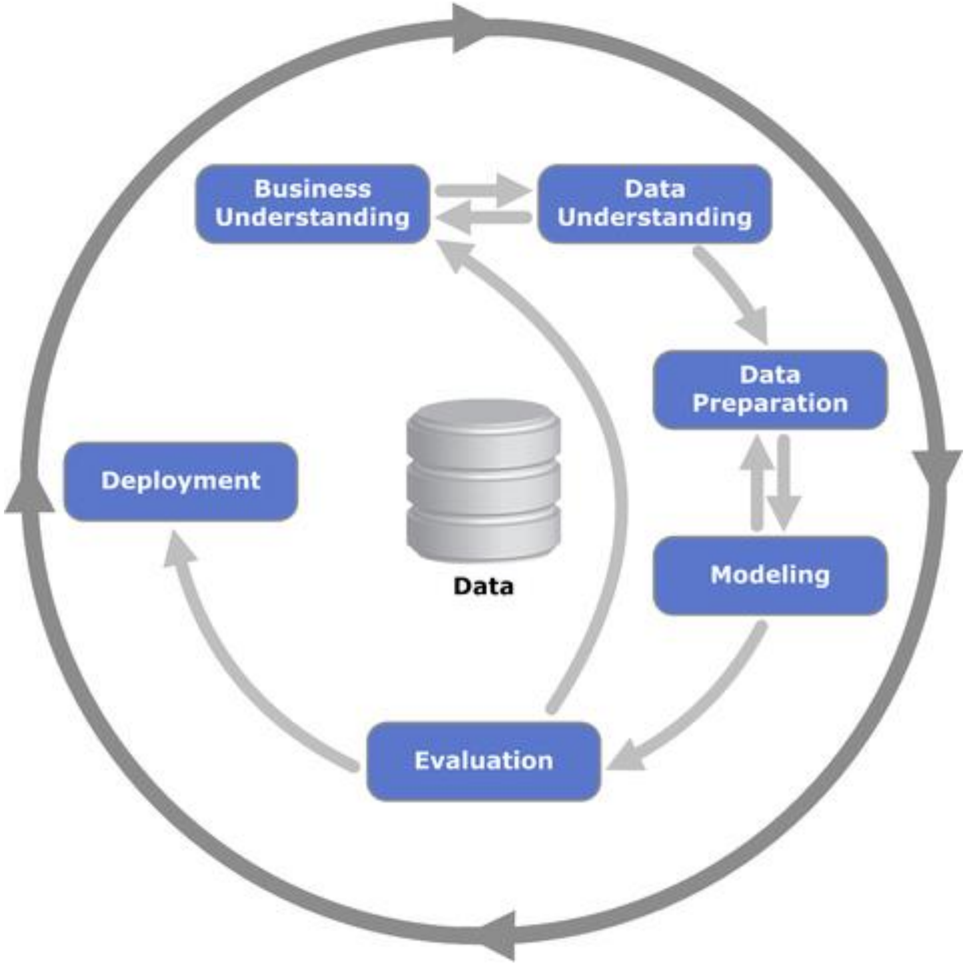
- **Classic KDD Process** (Knowledge Discovery in Databases)
  - Descriptive (association rules, clustering) and predictive



[Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth: From Data Mining to Knowledge Discovery in Databases. **AI Magazine 17(3) (1996)**]

The Data Science Lifecycle, cont.

- **CRISP-DM**
  - **CR**oss-**I**ndustry **S**tandard **P**rocess for **D**ata **M**ining
  - Additional focus on business understanding and deployment



[https://statistik-dresden.de/archives/1128]

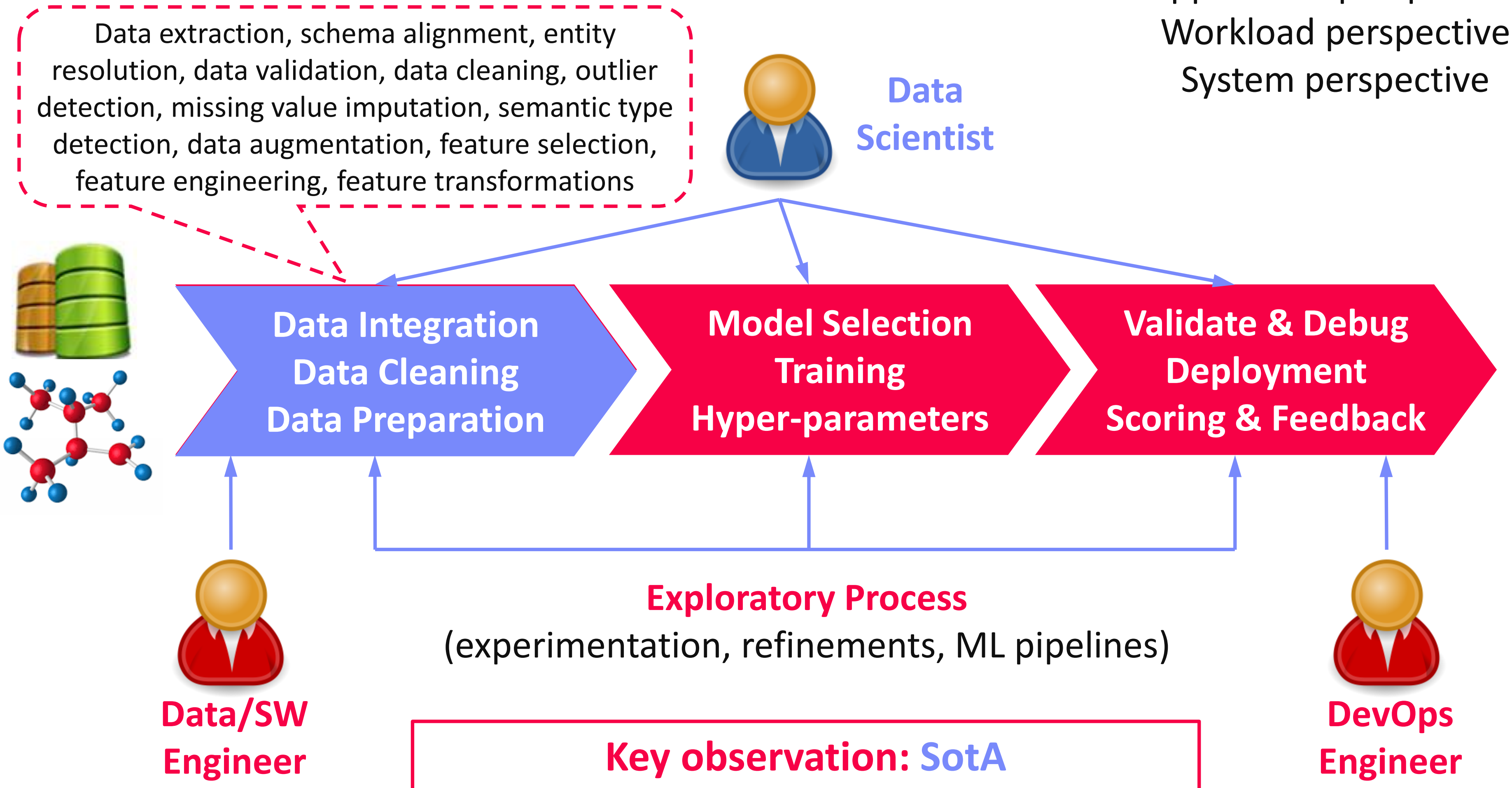
The Data Science Lifecycle, cont.

Data-centric View:

Application perspective

Workload perspective

System perspective



Data extraction, schema alignment, entity resolution, data validation, data cleaning, outlier detection, missing value imputation, semantic type detection, data augmentation, feature selection, feature engineering, feature transformations

Data Scientist

Data Integration
Data Cleaning
Data Preparation

Model Selection
Training
Hyper-parameters

Validate & Debug
Deployment
Scoring & Feedback

Exploratory Process
(experimentation, refinements, ML pipelines)

Data/SW Engineer

DevOps Engineer

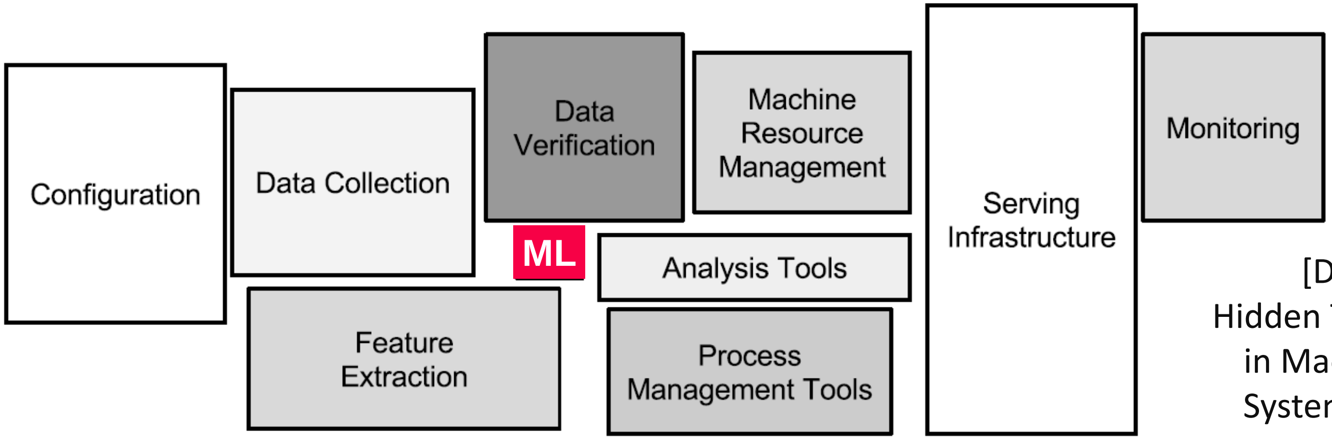
Key observation: SotA data integration/cleaning based on ML

The 80% Argument

- **Data Sourcing Effort**
  - Data scientists spend **80-90% time** on finding, integrating, cleaning datasets

[Michael Stonebraker, Ihab F. Ilyas: Data Integration: The Current Status and the Way Forward. **IEEE Data Eng. Bull. 41(2) (2018)**]

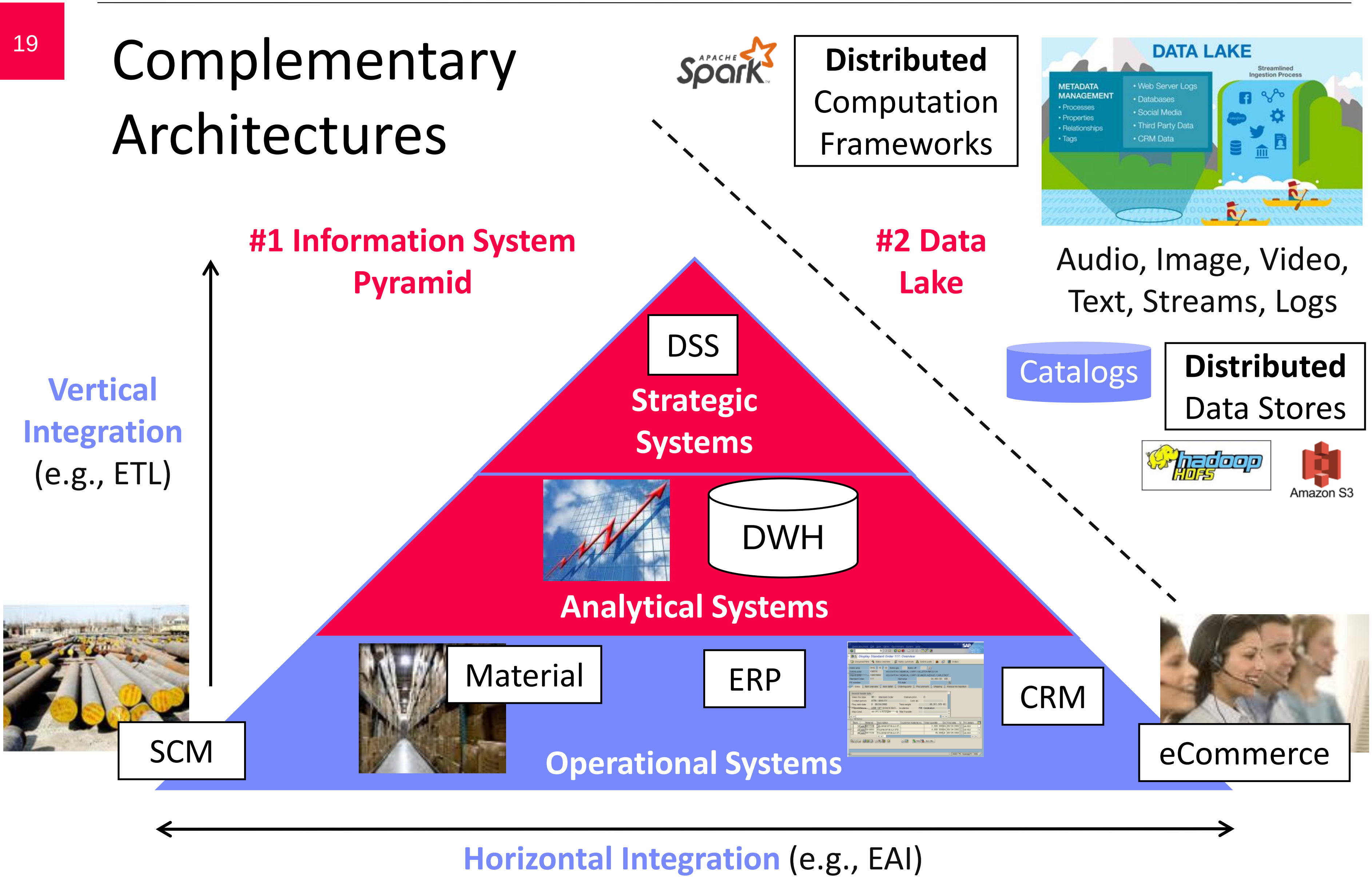
- **Technical Debts in ML Systems**



[D. Sculley et al.: Hidden Technical Debt in Machine Learning Systems. **NIPS 2015**]

  - Glue code, pipeline jungles, dead code paths
  - Plain-old-data types (arrays), multiple languages, prototypes
  - Abstraction and configuration debts
  - Data testing, reproducibility, process management, and cultural debts

Complementary Architectures



#1 Information System Pyramid

Vertical Integration (e.g., ETL)

DSS

Strategic Systems

DWH

Analytical Systems

SCM

Material

ERP

CRM

eCommerce

Operational Systems

Horizontal Integration (e.g., EAI)

#2 Data Lake

Distributed Computation Frameworks

Audio, Image, Video, Text, Streams, Logs

Catalogs

Distributed Data Stores

Course Goals

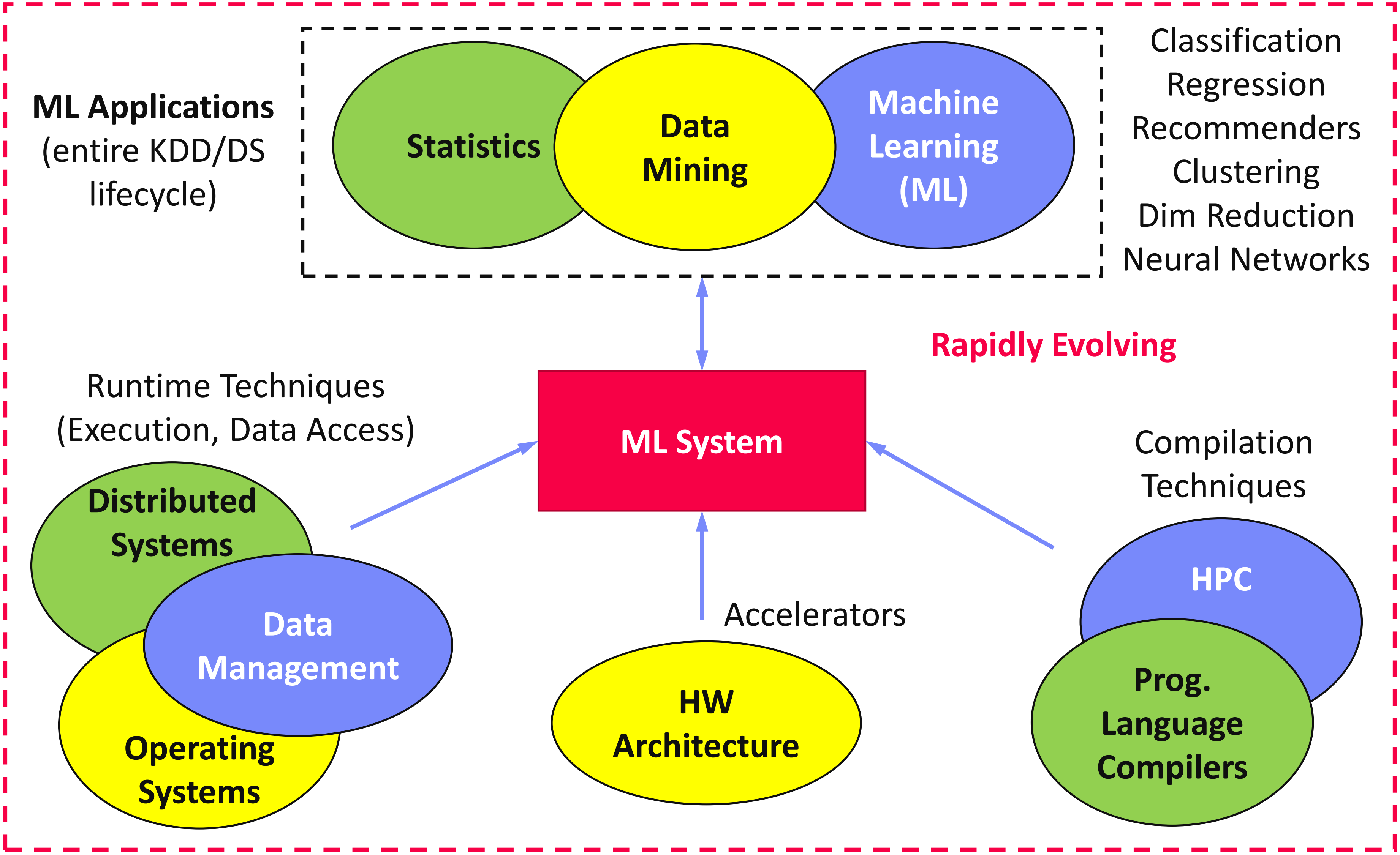
- **#1** **Major data integration architectures**
- **#2** **Key techniques for data integration and cleaning**
- **#3** **Methods for large-scale data storage and analysis**

Apache SystemDS:
A Declarative ML System for the End-to-End Data Science Lifecycle

Background and System Architecture

https://github.com/apache/systemds

What is an ML System?



Landscape of ML Systems

- **Existing ML Systems**
  - **#1 Numerical computing** frameworks
  - **#2 ML Algorithm libraries** (local, large-scale)
  - **#3 Linear algebra ML systems** (large-scale)
  - **#4 Deep neural network** (DNN) frameworks
  - **#5 Model management, and deployment**

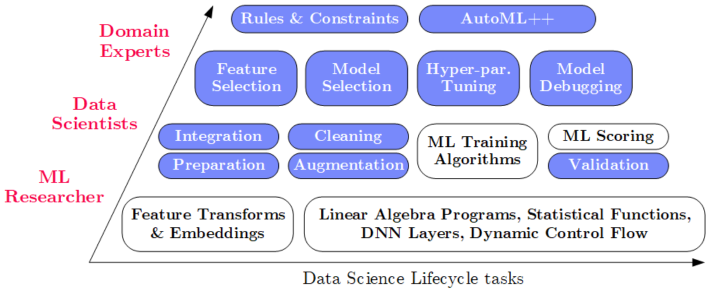
- **Exploratory Data-Science Lifecycle**
  - **Open-ended problems** w/ underspecified objectives
  - Hypotheses, data integration, run analytics
  - **Unknown value** → lack of system infrastructure
    → **Redundancy of manual efforts and computation**

“Take these datasets and show value or competitive advantage”

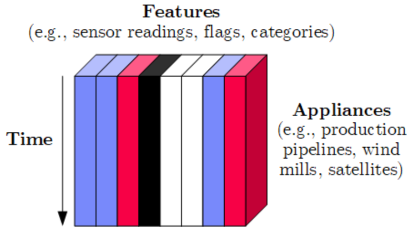
Apache SystemDS Design

Apache SystemML (since 2010)
→ SystemDS (09/2018)
→ Apache SystemDS (07/2020)

- **Objectives**
  - Effective and efficient **data preparation, ML, and model debugging at scale**
  - High-level abstractions for different lifecycle tasks and users



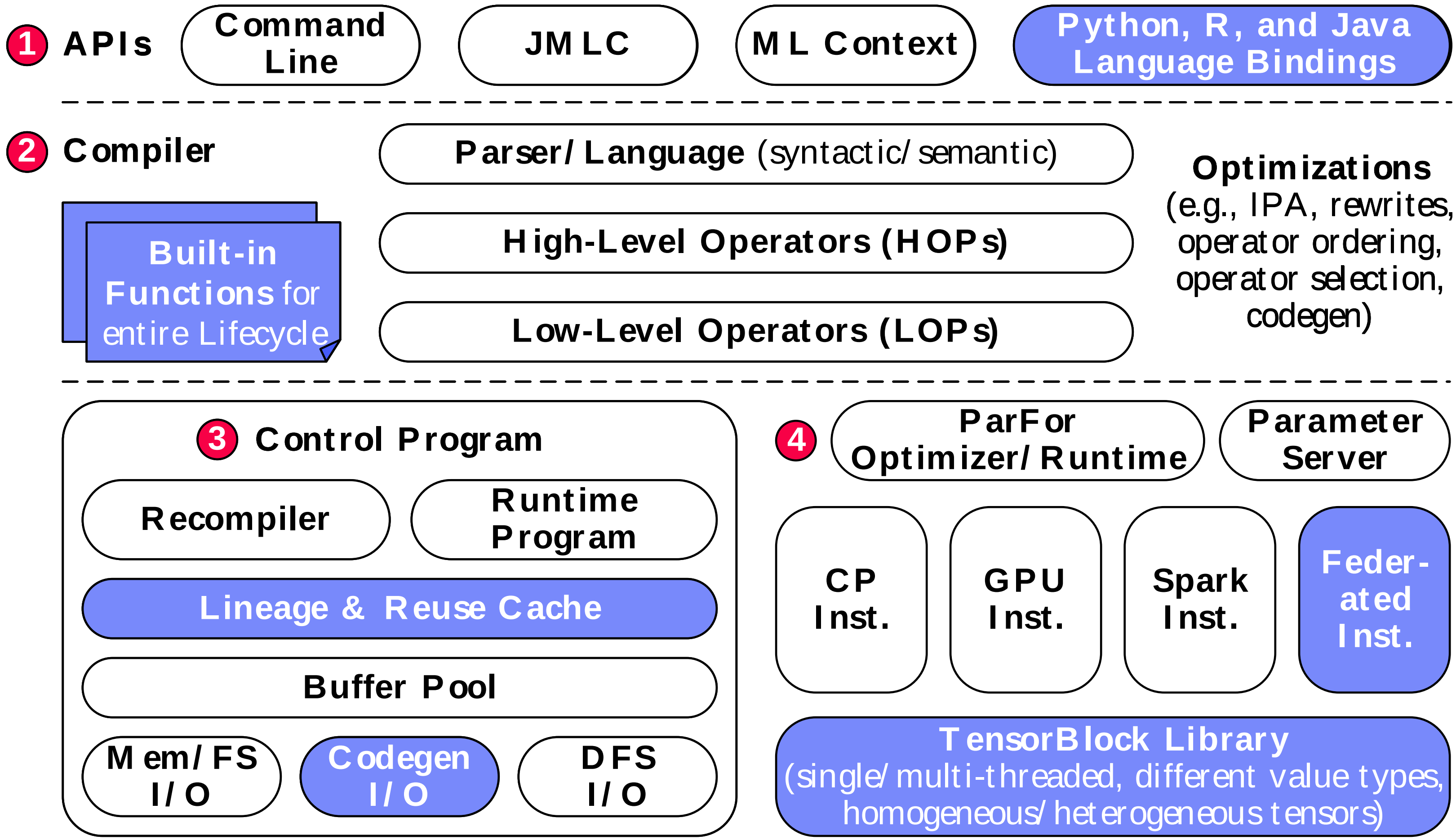
- **#1 Based on DSL for ML Training/Scoring**
  - Hierarchy of **abstractions for DS tasks**
  - ML-based SotA, interleaved, performance
- **#2 Hybrid Runtime Plans and Optimizing Compiler**
  - System infrastructure for diversity of algorithm classes
  - Different parallelization strategies and new architectures (**Federated ML**)
  - Abstractions → redundancy → automatic optimization


Features
(e.g., sensor readings, flags, categories)
Time
Appliances
(e.g., production pipelines, wind mills, satellites)

- **#3 Data Model: Heterogeneous Tensors**
  - Data integration/prep requires **generic data model**

Apache SystemDS Architecture

> 17,500 tests



1 APIs

Command Line

JMLC

MLContext

Python, R, and Java Language Bindings

2 Compiler

Built-in Functions for entire Lifecycle

Parser/Language (syntactic/semantic)

High-Level Operators (HOPs)

Low-Level Operators (LOPs)

Optimizations (e.g., IPA, rewrites, operator ordering, operator selection, codegen)

3 Control Program

Recompiler

Runtime Program

Lineage & Reuse Cache

Buffer Pool

Mem/FS I/O

Codegen I/O

DFS I/O

4 ParFor Optimizer/Runtime

Parameter Server

CP Inst.

GPU Inst.

Spark Inst.

Feder-ated Inst.

TensorBlock Library (single/multi-threaded, different value types, homogeneous/heterogeneous tensors)

[M. Boehm, I. Antonov, S. Baunsgaard, M. Dokter, R. Ginthör, K. Innerebner, F. Klezin, S. N. Lindstaedt, A. Phani, B. Rath, B. Reinwald, S. Siddiqui, S. Benjamin Wrede: SystemDS: A Declarative Machine Learning System for the End-to-End Data Science Lifecycle. **CIDR 2020**]

Basic HOP and LOP DAG Compilation

LinregDS (Direct Solve)

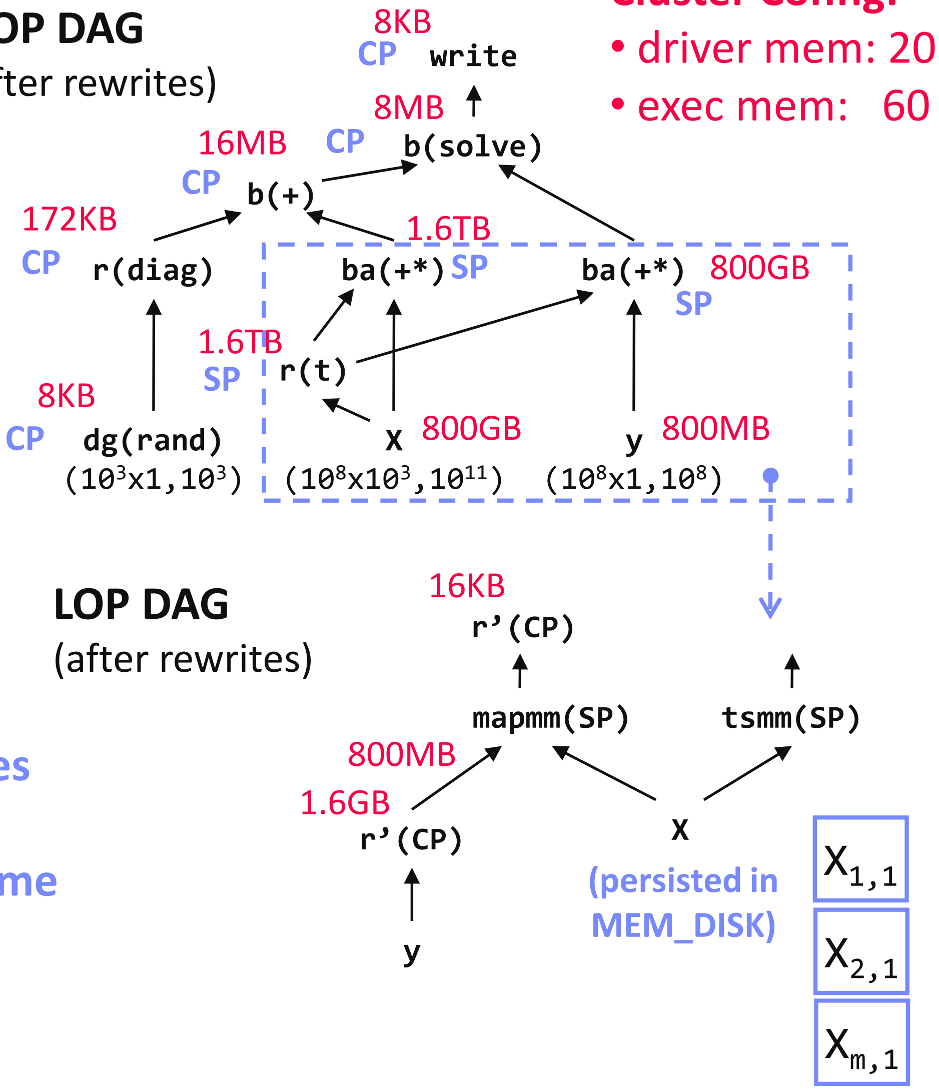
```
X = read($1);
y = read($2);
intercept = $3;
lambda = 0.001;
...
if( intercept == 1 ) {
  ones = matrix(1, nrow(X), 1);
  X = append(X, ones);
}
I = matrix(1, ncol(X), 1);
A = t(X) %*% X + diag(I)*lambda;
b = t(X) %*% y;
beta = solve(A, b);
...
write(beta, $4);
```

Scenario:
X: 10^8 x 10^3, 10^{11}
y: 10^8 x 1, 10^8

Cluster Config:
- driver mem: 20 GB
- exec mem: 60 GB

HOP DAG
(after rewrites)

LOP DAG
(after rewrites)

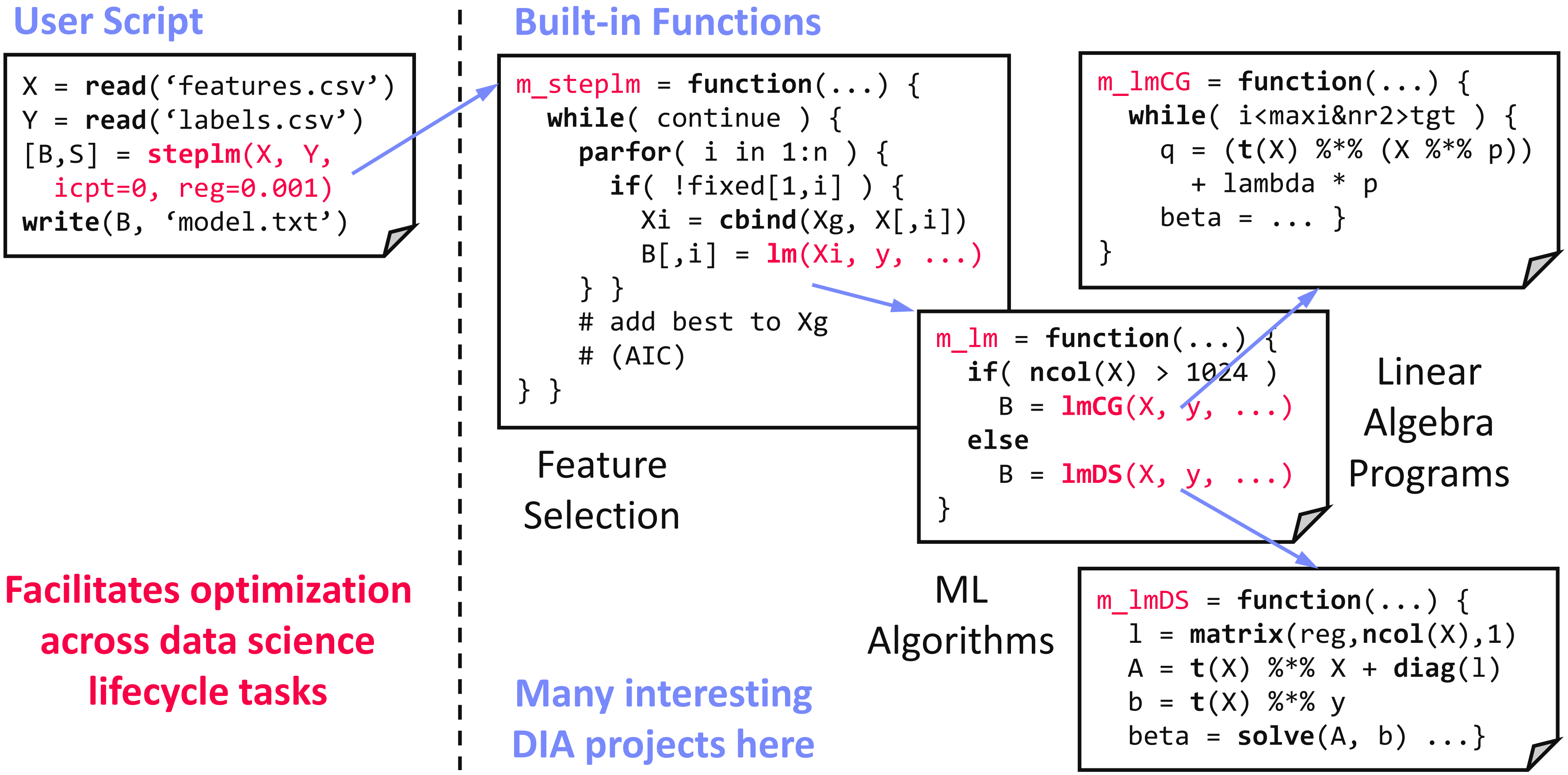


➔ **Hybrid Runtime Plans:**
- **Size propagation / memory estimates**
- **Integrated CP / Spark runtime**
- **Dynamic recompilation during runtime**

➔ **Distributed Matrices**
- **Fixed-size (squared) matrix blocks**
- **Data-parallel operations**

Language Abstractions and APIs, cont.

- **Example: Stepwise Linear Regression**



User Script

```
X = read(‘features.csv’)
Y = read(‘labels.csv’)
[B,S] = steplm(X, Y,
  icpt=0, reg=0.001)
write(B, ‘model.txt’)
```

Built-in Functions

```
m_steplm = function(...) {
  while( continue ) {
    parfor( i in 1:n ) {
      if( !fixed[1,i] ) {
        Xi = cbind(Xg, X[,i])
        B[,i] = lm(Xi, y, ...)
    } }
    # add best to Xg
    # (AIC)
} }
```

Feature Selection

```
m_lm = function(...) {
  if( ncol(X) > 1024 )
    B = lmCG(X, y, ...)
  else
    B = lmDS(X, y, ...)
}
```

ML Algorithms

```
m_lmCG = function(...) {
  while( i<maxi&nr2>tgt ) {
    q = (t(X) %*% (X %*% p))
      + lambda * p
    beta = ... }
}
```

Linear Algebra Programs

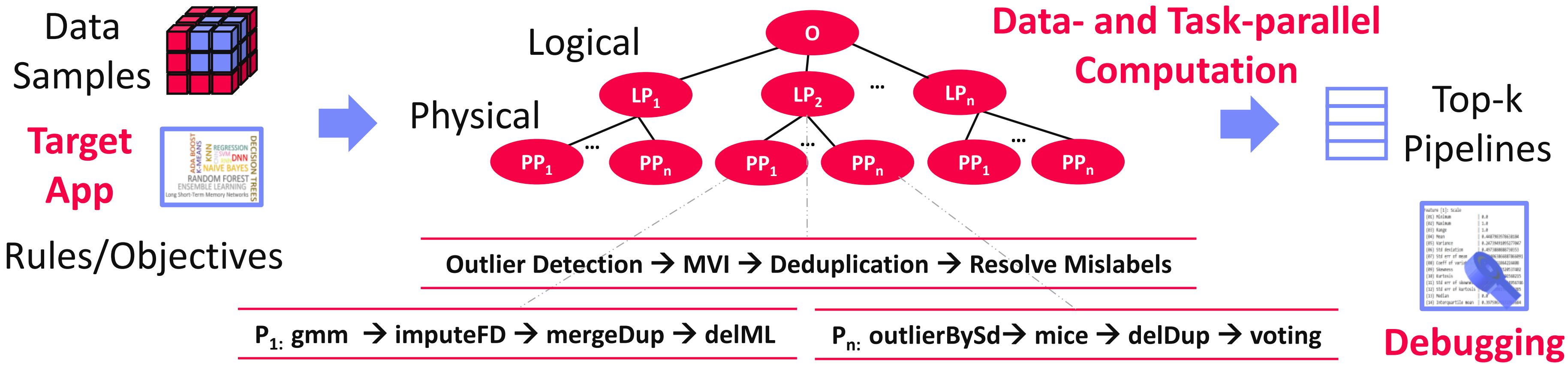
```
m_lmDS = function(...) {
  l = matrix(reg,ncol(X),1)
  A = t(X) %*% X + diag(l)
  b = t(X) %*% y
  beta = solve(A, b) ...}
```

Facilitates optimization across data science lifecycle tasks

Many interesting DIA projects here

Data Cleaning Pipelines

- **Automatic Generation of Cleaning Pipelines**
  - Library of robust, parameterized **data cleaning primitives** (physical/logical)
  - **Enumeration of DAGs** of primitives & **hyper-parameter optimization** (HB, BO)



Outlier Detection → MVI → Deduplication → Resolve Mislabels

$P_{1:}$ gmm → imputeFD → mergeDup → delML

$P_{n:}$ outlierBySd → mice → delDup → voting

University	Country
TU Graz	Austria
TU Graz	Austria
TU Graz	Germany
IIT	India
IIT	IIT
IIT	Pakistan
IIT	India
SIBA	Pakistan
SIBA	null
SIBA	null

Dirty Data

University	Country
TU Graz	Austria
TU Graz	Austria
TU Graz	Austria
IIT	India
IIT	India
IIT	India
IIT	India
SIBA	Pakistan
SIBA	Pakistan
SIBA	Pakistan

After **imputeFD(0.5)**

A	B	C	D
0.77	0.80	1	1
0.96	0.12	1	1
0.66	0.09	null	1
0.23	0.04	17	1
0.91	0.02	17	null
0.21	0.38	17	1
0.31	null	17	1
0.75	0.21	20	1
null	null	20	1
0.19	0.61	20	1
0.64	0.31	20	1

Dirty Data

A	B	C	D
0.77	0.80	1	1
0.96	0.12	1	1
0.66	0.09	17	1
0.23	0.04	17	1
0.91	0.02	17	1
0.21	0.38	17	1
0.31	0.29	17	1
0.75	0.21	20	1
0.41	0.24	20	1
0.19	0.61	20	1
0.64	0.31	20	1

After **MICE**

Notable DAMS Lab Research since 2019

- SystemDS [CIDR'20]
- Multi-Level Lineage Tracing & Reuse [SIGMOD'21]
- Federated Learning [SIGMOD'21]
- Model Debugging (SliceLine) [SIGMOD'21]
- DAPHNE [CIDR'22]
- Feature Transformations [VLDB'22]
- Compression Framework [SIGMOD'23]

https://damslab.github.io/

Summary and Q&A

- **Course Goals**
  - **#1** Major data integration architectures
  - **#2** Key techniques for data integration and cleaning
  - **#3** Methods for large-scale data storage and analysis

- **Exercise/Projects**
  - Exercise on **data integration and ML pipeline**

- **Next Lectures**
  - 02 **Data Warehousing, ETL, and SQL/OLAP** [Oct 14]
  - 03 **Message-oriented Middleware, EAI, and Replication** [Oct 21]