Self-Optimizing Big Data Processing

IT4BI-DC Doctoral Colloquium

Sergi Nadal

Supervised by Alberto Abelló, Oscar Romero and Stijn Vansummeren

Also in collaboration with Panos Vassiliadis

06/07/2017

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

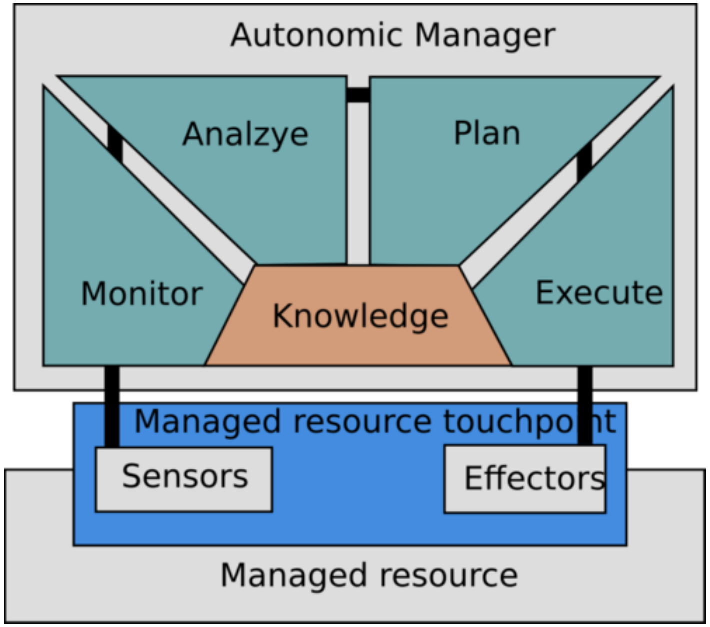
DTIM

Autonomic Computing

- IBM’s vision of autonomic computing

“a computing environment with the ability to manage itself and dinamically adapt to change in accordance with business policies and objectives”

- The MAPE-K adaptation control loop



[Kephart et al., 2003]

On the need of autonomic Big Data computing

- Big Data ecosystems store and process
  - Stationary data → batch (e.g., transactional)
  - Situational data → real-time (e.g., social networks)
- Situational data is highly heterogeneous and dynamic by nature
  - Changes in the arrival rate, schema, data distribution, …
  - Direct impact on the system's performance
- Self-adaptation is highly needed
  - Self-healing, self-configuration, self-protection, …
  - We focus on self-optimization

[Löser et al., 2009]

Metadata – the cornerstone for self-optimization

- A Big Data architecture with the *Knowledge* component, capable of answering:
  - What are my data sources?
  - Which schema are they providing?
  - How frequent data are arriving?
  - How are my data changing? (...)
- Make the architecture aware of "what's going on"
  - Semantic awareness
  - Machine-readable metadata to provide (partial) automation of data definition and exploitation

Today's overview

- A software reference architecture for semantic-aware Big Data systems
- Self-optimization techniques
  - An integration-oriented ontology to govern evolution
  - Intermediate results materialization selection for data-intensive flows
  - A management system for distributed CER
- Conclusions & publication plan
- References

A SOFTWARE REFERENCE ARCHITECTURE FOR SEMANTIC-AWARE BIG DATA SYSTEMS

Requirements for a semantic-aware Big Data architecture

- 5 dimensions, 15 functional requirements
  - A SLR on Big Data architectures

Custom Architectures	*Volume*			*Velocity*		*Variety*			*Variability*			*Veracity*			
	R1.1	R1.2	R1.3	R2.1	R2.2	R3.1	R3.2	R3.3	R4.1	R4.2	R4.3	R5.1	R5.2	R5.3	R5.4
A1 CQELS (Phuoc et al., 2012)	✗	✓	✗	✓	✓	✗	✗	✓	✓	✗	✓	✗	✗	✗	✗
A2 AllJoyn Lambda (Villari et al., 2014)	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗
A3 CloudMan (Qanbari et al., 2014)	✓	✓	✓	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗
A4 AsterixDB (Alsubaiee et al., 2014)	✓	✓	✗	✓	✗	✓	✗	✓	✓	✓	✓	✓	✗	✗	✗
A5 M3Data (Ionescu et al., 2014)	✓	✓	✓	✓	✗	✓	✗	✓	✗	✗	✗	✗	✗	✗	✓
A6 (Twardowski and Ryzko, 2014)	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗
A7 λ-arch. (Marz and Warren, 2015)	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗
A8 SOLID (Martínez-Prieto et al., 2015)	✗	✓	✗	✓	✓	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗
A9 Liquid (Fernandez et al., 2015)	✗	✗	✗	✓	✓	✓	✓	✗	✗	✗	✗	✓	✗	✗	✗
A10 RADStack (Yang et al., 2015)	✓	✓	✗	✓	✓	✓	✗	✓	✗	✗	✗	✗	✗	✗	✓
A11 (Kroß et al., 2015)	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗
A12 HaoLap (Song et al., 2015)	✓	✓	✗	✗	✗	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗
A13 (Wang et al., 2015)	✓	✓	✓	✗	✗	✓	✓	✗	✗	✗	✗	✓	✓	✗	✓
A14 SHMR (Guo et al., 2015)	✓	✓	✗	✗	✗	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗
A15 Tengu (Vanhove et al., 2015)	✓	✓	✓	✓	✓	✓	✗	✓	✗	✗	✓	✗	✗	✗	✗
A16 (Xie et al., 2015)	✓	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗	✓	✗
A17 (e Sá et al., 2015)	✓	✓	✓	✗	✗	✓	✗	✓	✗	✗	✗	✗	✗	✗	✓
A18 D-Ocean (Zhuang et al., 2016)	✓	✓	✗	✗	✗	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗
Software Reference Architectures	*Volume*			*Velocity*		*Variety*			*Variability*			*Veracity*			
	R1.1	R1.2	R1.3	R2.1	R2.2	R3.1	R3.2	R3.3	R4.1	R4.2	R4.3	R5.1	R5.2	R5.3	R5.4
A19 NIST (Grady et al., 2014)	✓	✓	✓	✗	✗	✗	✗	✓	✗	✗	✓	✗	✓	✓	✓
A20 (Pääkkönen and Pakkala, 2015)	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✓
A21 (Geerdink, 2015)	✓	✓	✓	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗
Bolster	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Conclusions of the SLR

- Two major families of architectures

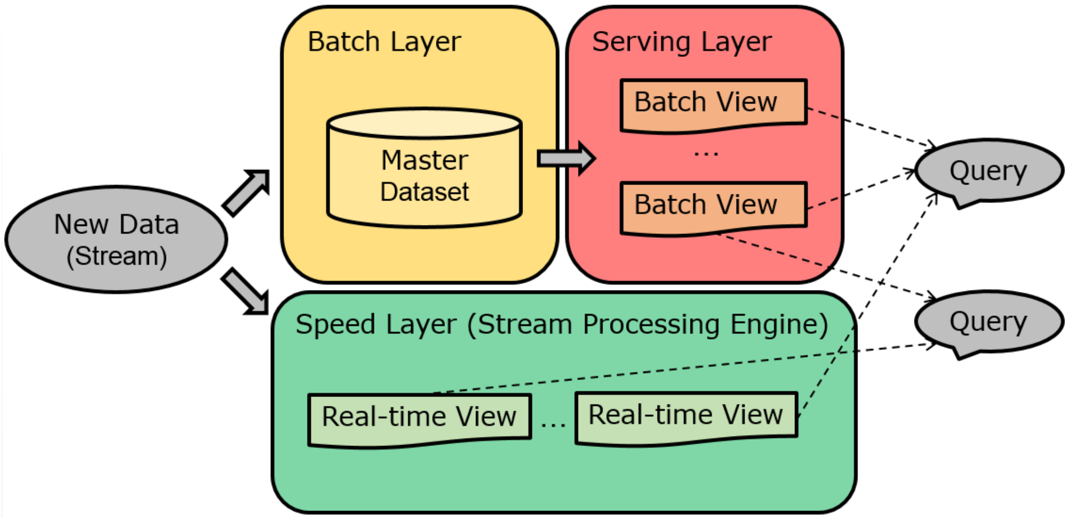
Family	Volume	Velocity	Variety	Variability	Veracity
The λ-architecture and evolutions					
Semantic Web principles & technologies					

Fulfils requirement
Partially fulfils requirement
Does not fulfil requirement

- No architecture satisfies the sought requirements
- Focused on performance-oriented aspects
- *Semantic-awareness* is poorly covered

Bolster

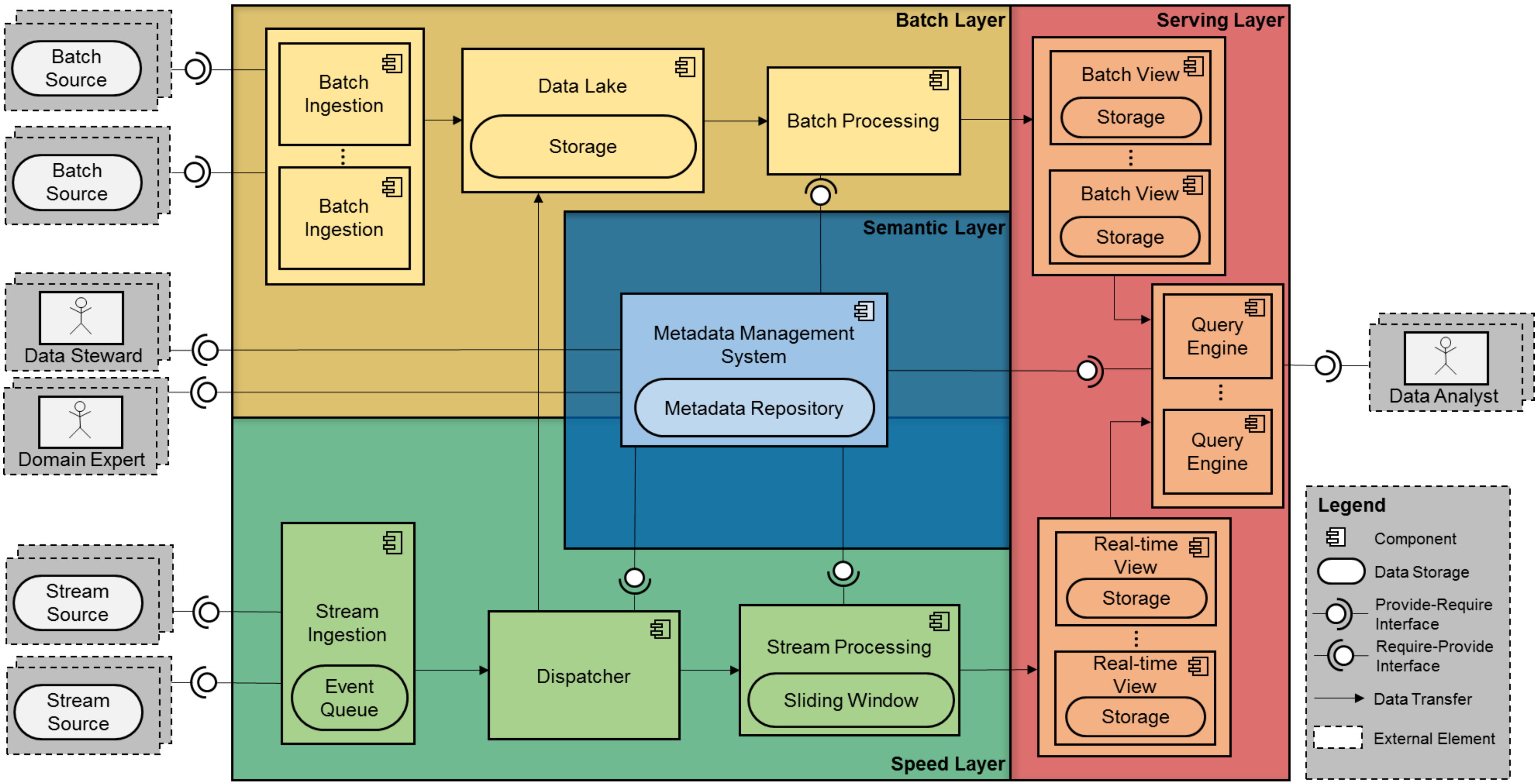
- A SRA for semantic-aware Big Data systems
- Combination of the two major families
  - Based on the λ-architecture

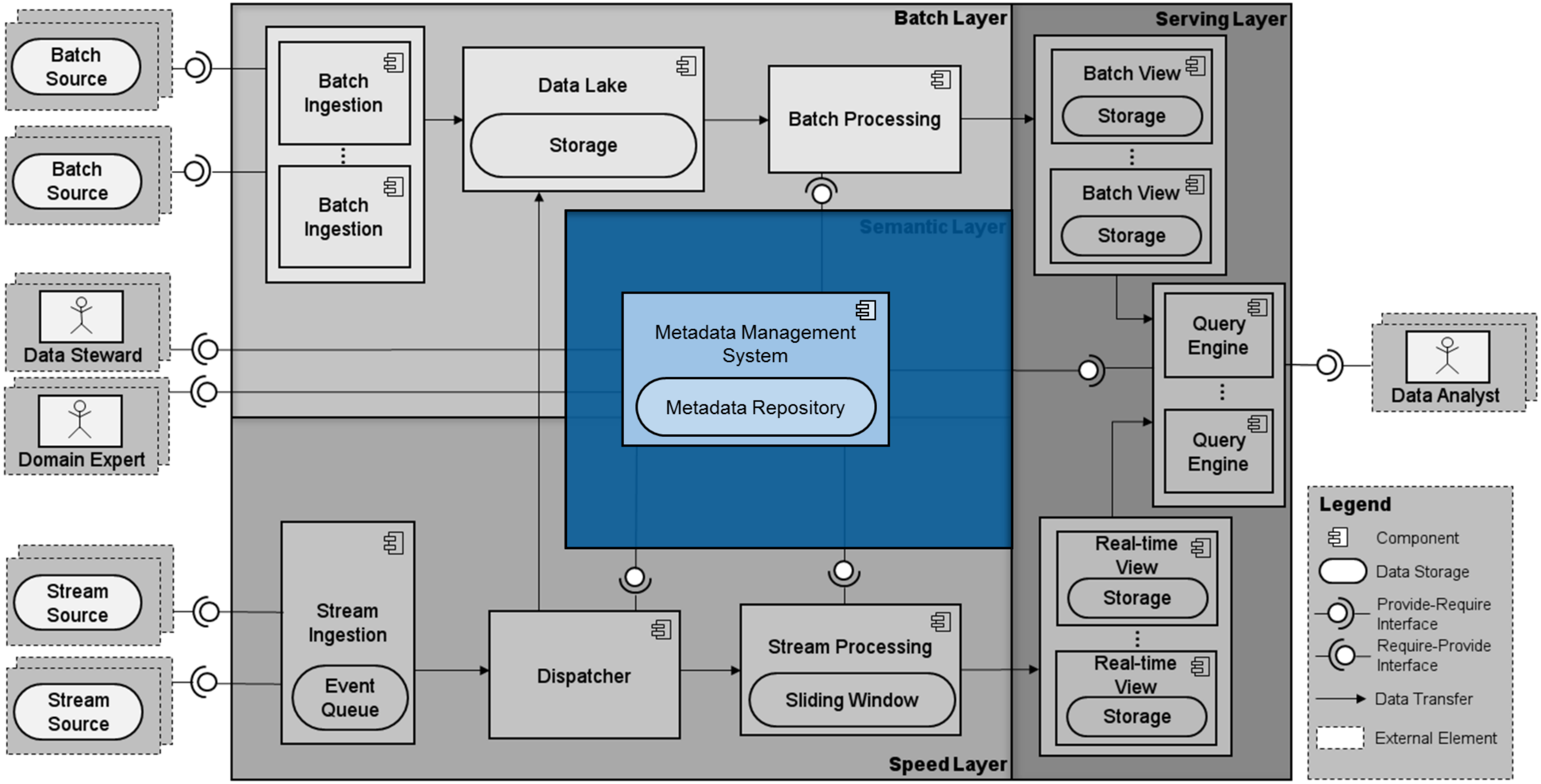


  - Use Semantic Web technologies to represent machine-readable metadata

[Marz et al., 2015]

Bolster conceptual view





AN INTEGRATION-ORIENTED ONTOLOGY TO GOVERN SCHEMA EVOLUTION

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

DTIM

The data variety challenge

- How to provide an integrated view over an evolving and heterogeneous set of data sources?
- Ontologies as a formal tool to provide a shared conceptualization of the domain of interest, formalized by means of Description Logics (DLs)
  - TBox – general properties of concepts and roles
  - ABox – instances of concepts and roles
- Ontology-Based Data Access (OBDA)
  - Allow users to query the ontology ($\mathcal{T}$), and translate such queries to the sources ($\mathcal{S}$) via mappings ($\mathcal{M}$)
  - ABox is in the sources

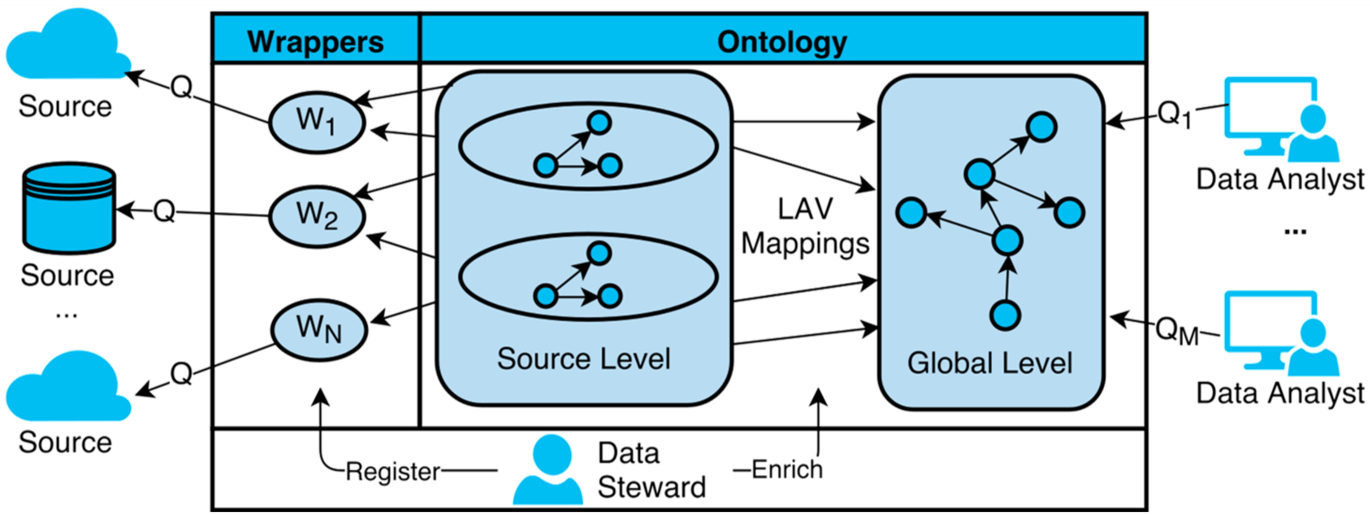
[Horrocks et al., 2016]

OBDA in our scenario?

- What if $\mathcal{S}$ changes? How are queries on $\mathcal{T}$ affected?
- Traditional OBDA represent schema mappings following the *global-as-view* approach
  - Elements of $\mathcal{T}$ are characterized as queries over $\mathcal{S}$
  - Simple query answering (unfolding), but changes in the sources might invalidate mappings
- We aim for *local-as-view* schema mappings
  - Elements of $\mathcal{S}$ are characterized as queries over $\mathcal{T}$
  - Loosely-coupling between $\mathcal{T}$ and $\mathcal{S}$, but query answering might require reasoning

An RDF-based approach

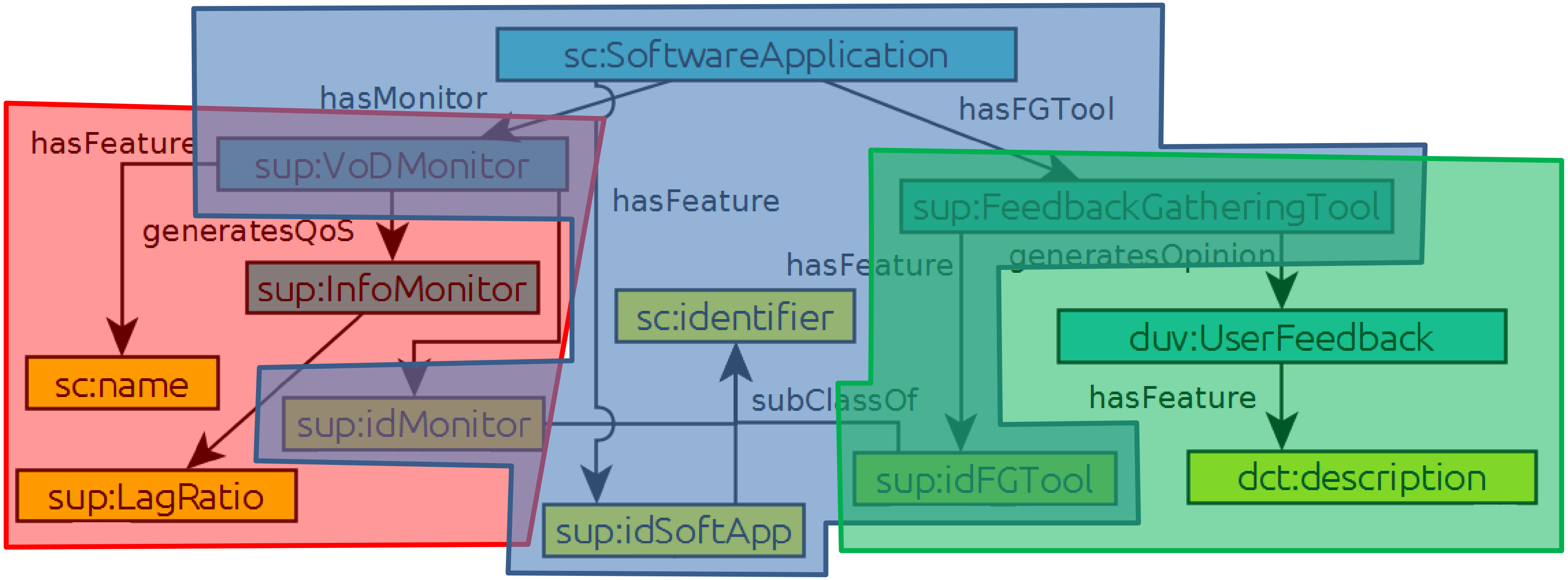
- Global Level $\mathcal{G}$ – integrated view for users to query
- Source Level $\mathcal{S}$ – structure of the data sources
- Mappings $\mathcal{M}$ – LAV mappings between $\mathcal{G}$ and $\mathcal{S}$



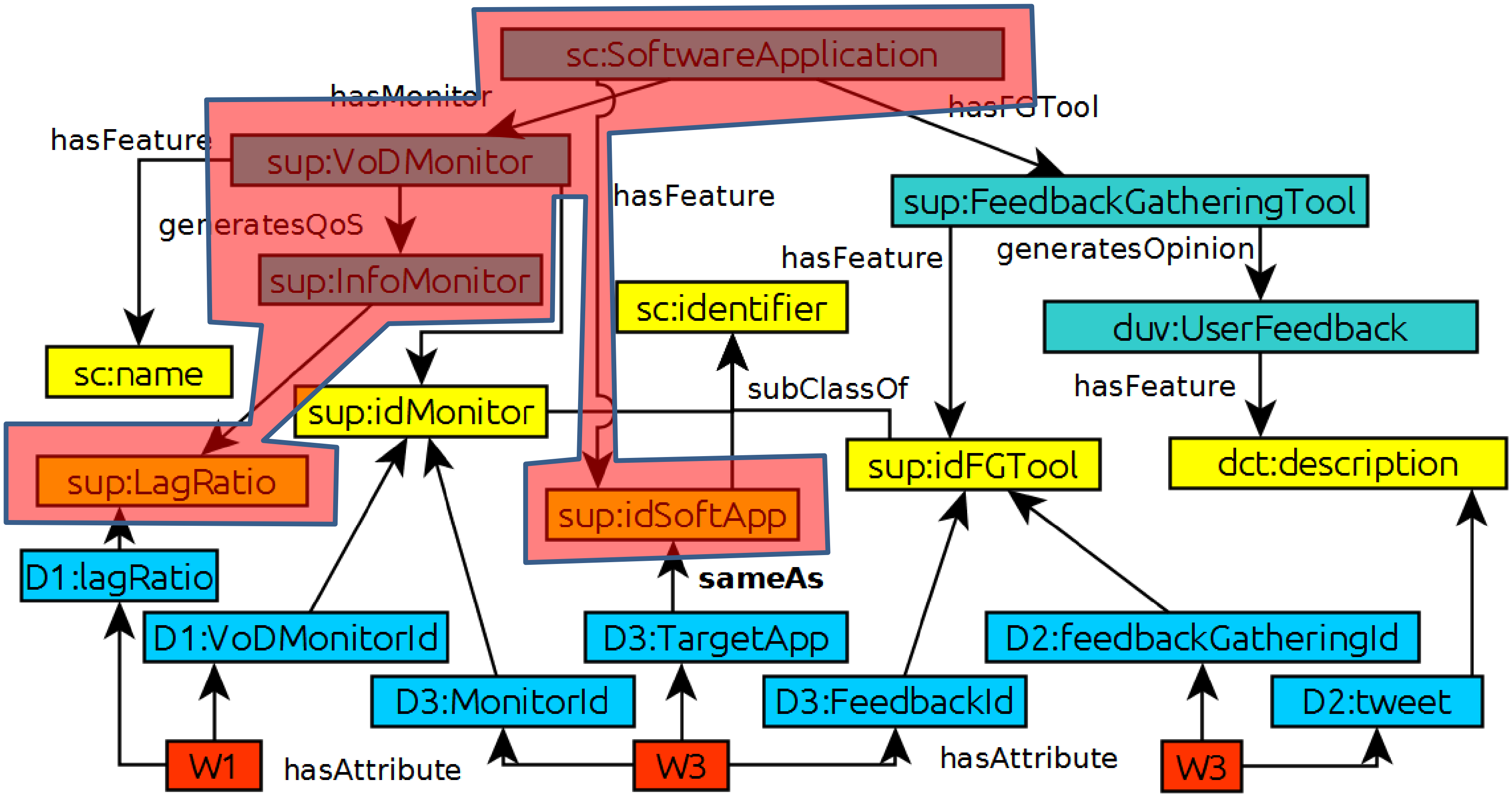
- Given a SPARQL pattern matching over $\mathcal{G}$, return an equivalent walk over $\mathcal{S}$ (chain of joins and projections) using the mappings $\mathcal{M}$ and translate it to a union of CQs over the wrappers.

The Big Data Integration ontology

- $\mathcal{G}$ - Concepts and features of analysis
- $\mathcal{S}$ - Accurate representation of the wrappers
- $\mathcal{M}$ - LAV mappings with named graphs (SPARQL support)



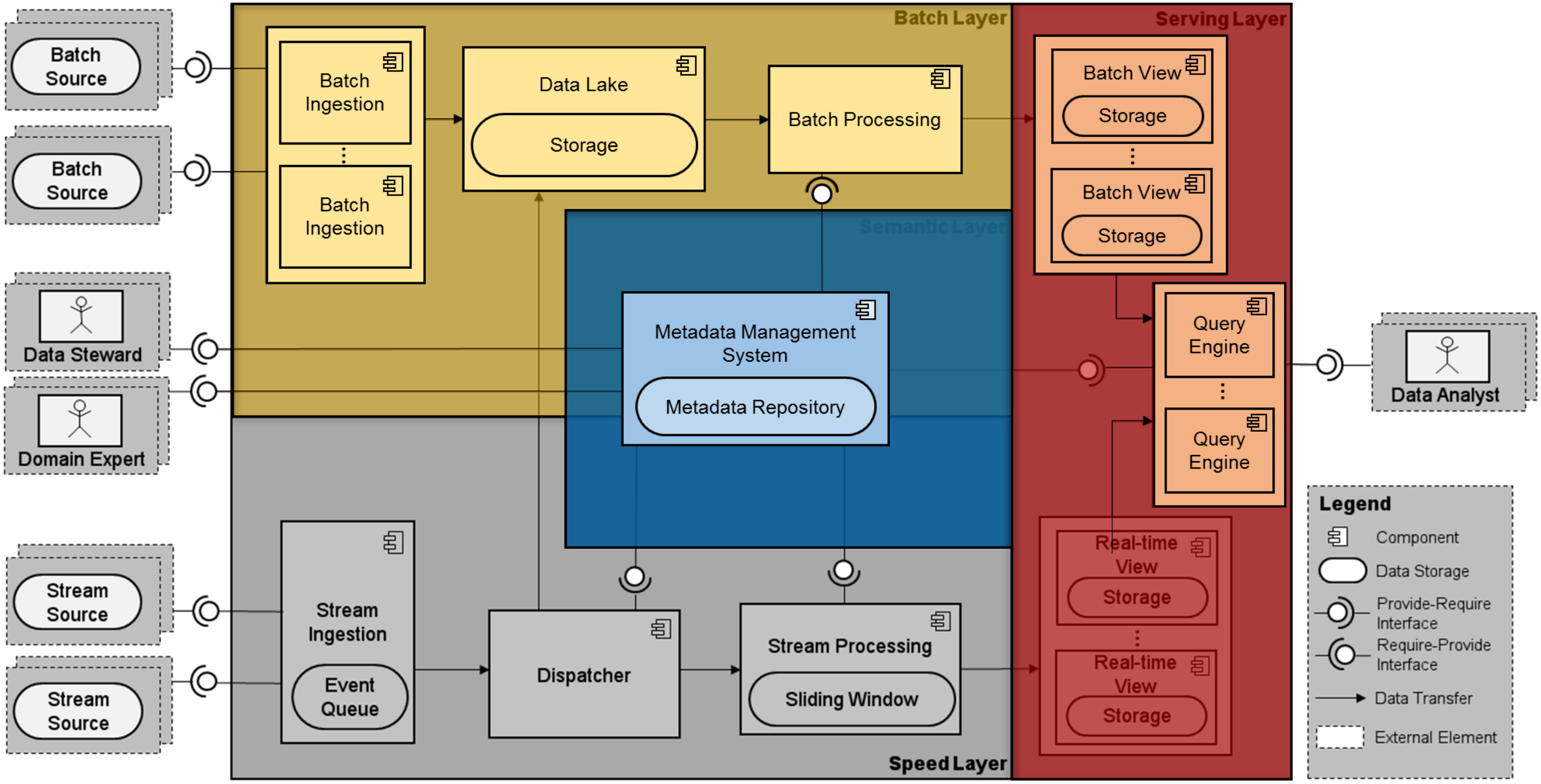
Query answering



$$\Pi_{W_1.lagRatio, W3.TargetApp}(\sigma_{W_1.VoDMonitorId=W_3.MonitorId}(W_1 \times W_3))$$

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

DTIM



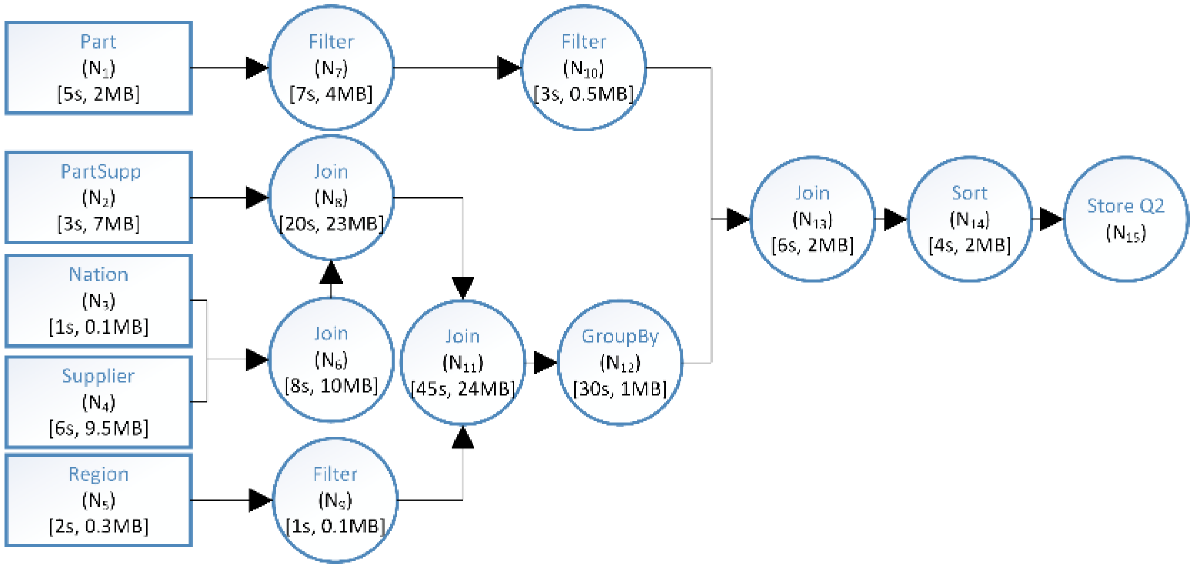
INTERMEDIATE RESULTS MATERIALIZATION SELECTION FOR DATA-INTENSIVE FLOWS

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

DTIM

Reusing intermediate results

- Batch processing is commonly represented by DIFs (e.g., MapReduce or Spark jobs)



- User workloads have high temporal locality
  - 80% will be reused in the range of minutes to hours
- How can I optimize its reuse?

[Chen et al., 2012]

Challenges

- What intermediate results to materialize?
- ~~How to materialize them?~~ (Rana Faisal)
- Materialized view selection in DIFs
  - A cost-based approach driven by SLAs (e.g., optimize query time, storage space, …)
  - Multiple and conflicting objectives
- We provide a local search algorithm that probabilistically selects a set of near-optimal intermediate results to materialize.

The cost model

- Statistics – logical properties of the flow, propagated across operators
  - Selectivity factor, distinct values per attribute, cardinality
- Metrics – engine-specific estimations per node
  - Size of a disk block, memory buffers, size of attributes
  - We estimate execution (disk I/O) and space (blocks)
- Cost functions – composition of metrics to measure a SLA
  - Loading cost, query cost, storage cost
  - Easily extensible: monetary aspects, energy consumption

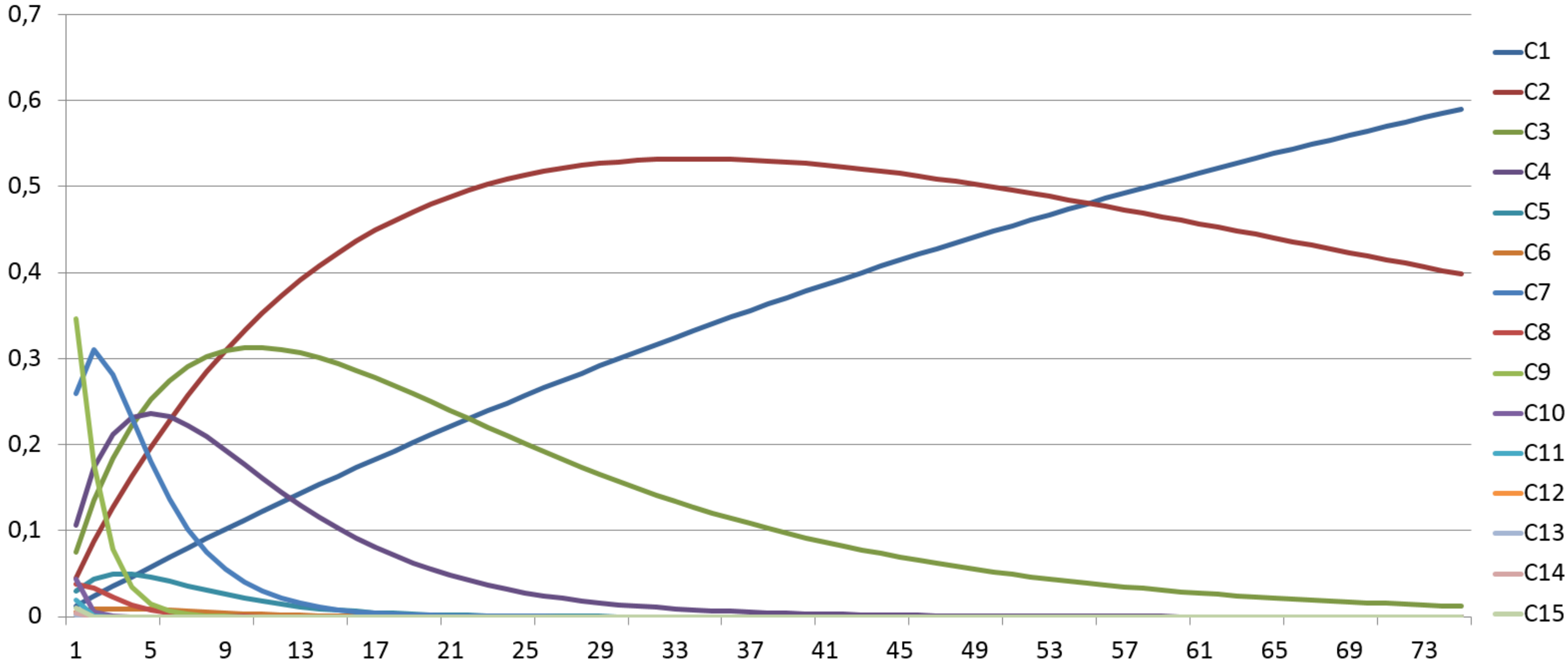
[Nguyen et al., 2012], [Roukh et al., 2015]

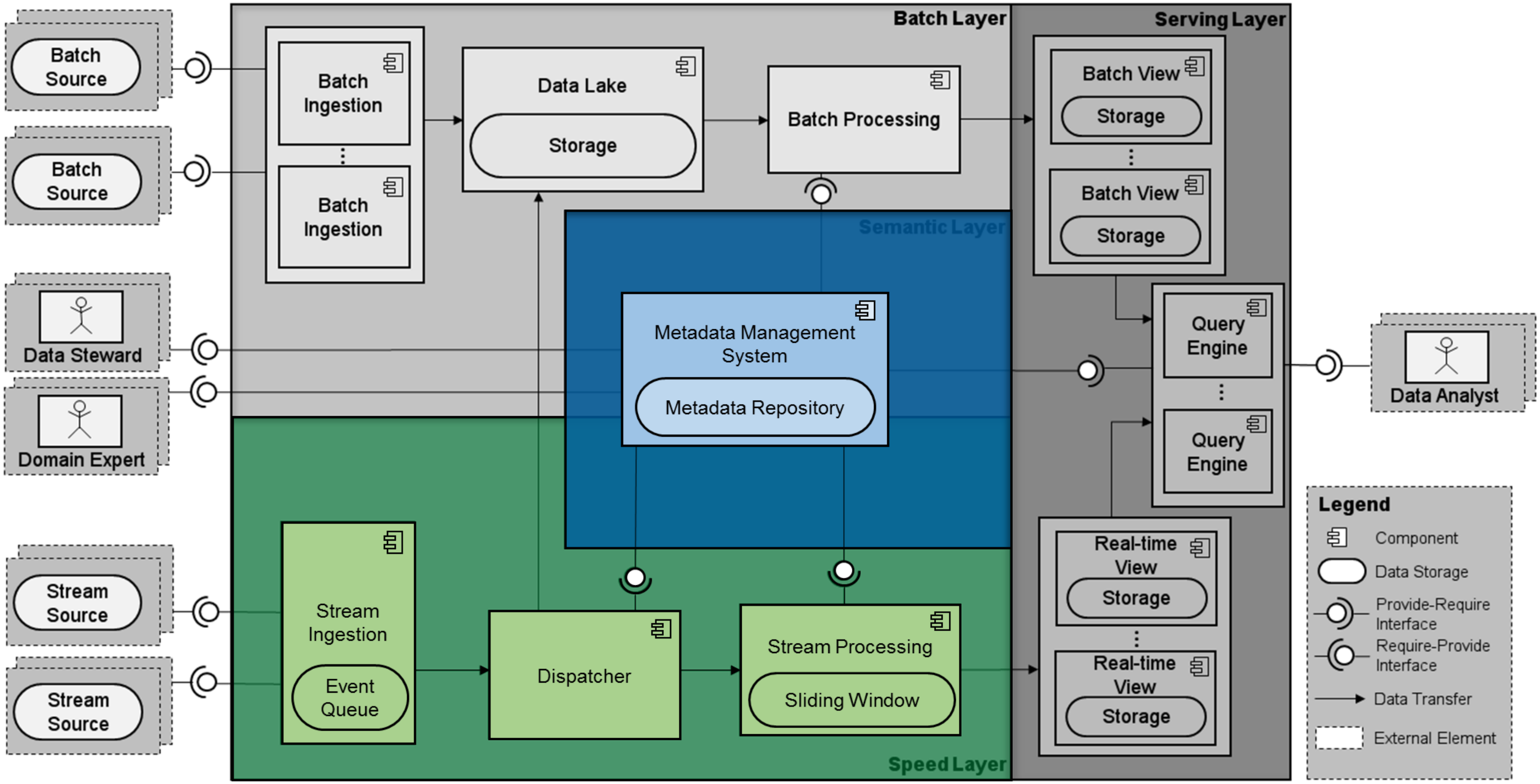
Shotgun hill-climbing

- Design goals (heuristic function)
  - Combination of SLAs (e.g., 75% query, 25% space)
- Hill-climbing – greedy to the best heuristic
- Cost functions are non-monotonic
  - The output will vary with the initial state
- Approach: execute hill-climbing a certain number of iterations
  - Random initial state
  - Keep the best heuristic across iterations

Evaluation

- Evolution of probabilities per number of iterations for each different solution





A MANAGEMENT SYSTEM FOR DISTRIBUTED COMPLEX EVENT RECOGNITION

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

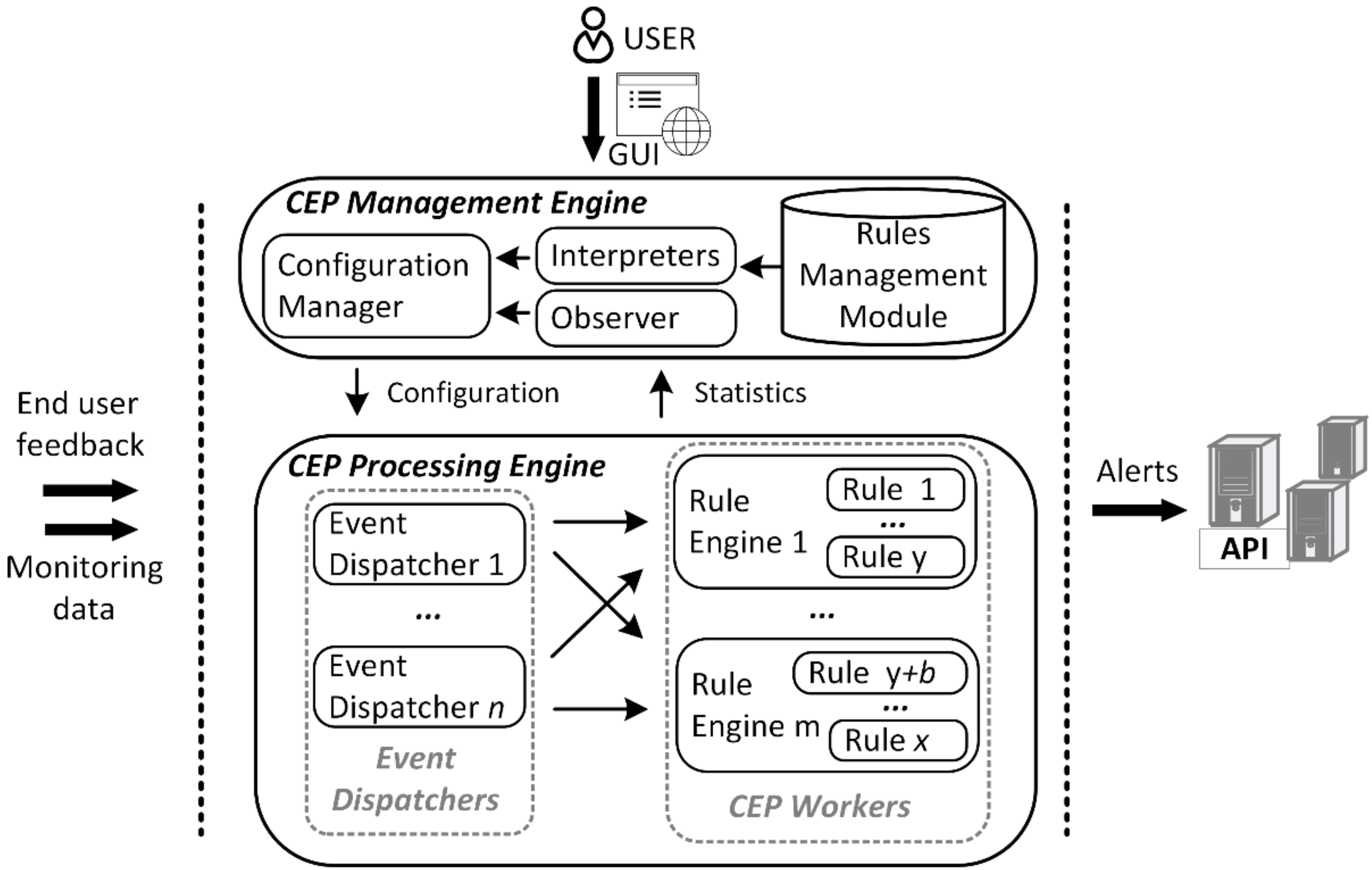
DTIM

CER systems

- Complex Event Recognition (CER) deal with the detection of events in Big Data streams
  - E.g., raise an alert if **A**, no **B** after 5 minutes and 3 times **C** after 15 minutes
- Distributing CER operators is a challenging task
  - Most approaches rely on centralized solutions
- Proposed approach
  - A set of shared nothing CER engines (e.g., Esper)
  - Dynamic event dispatchment and rule placement

[Cugola et al., 2012]

Architecture for distributed CER



Cost-based distribution of events

- We aim for a cost model to decide
  - Rule placement
  - Where are events dispatched
- Rely on an implementation-independent declaration of rules
  - Based on an RDF vocabulary
  - Linked to the BDI ontology
- Annotate it with runtime metadata

CONCLUSIONS & PUBLICATION PLAN

Conclusions

- Autonomic Big Data computing
  - Focusing on self-optimization
- *Bolster*
  - An SRA that includes the *Knowledge* component
- The Big Data Integration ontology
  - LAV mappings for dynamic environments
- SLA-driven materialization of intermediate results
- Distributed CER

References (I) – in order of appearance

- Kephart, J. O., & Chess, D. M. (2003). The vision of autonomic computing. *Computer*, *36*(1), 41-50.
- Löser, A., Hueske, F., & Markl, V. (2009). Situational business intelligence. *BIRTE,* 1-11.
- Marz, N., & Warren, J. (2015). Big Data: Principles and best practices of scalable realtime data systems. *Manning*.
- Horrocks, I., Giese, M., Kharlamov, E., & Waaler, A. (2016). Using semantic technology to tame the data variety challenge. *IEEE Internet Computing*, *20*(6), 62-66.
- Jovanovic, P., Romero, O., & Abelló, A. (2016). A unified view of data-intensive flows in business intelligence systems: a survey. Transactions on Large-Scale Data-and Knowledge-Centered Systems, 29, 66-107.

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

DTIM

References (II) – in order of appearance

- Chen, Y., Alspaugh, S., & Katz, R. (2012). Interactive analytical processing in big data systems: A cross-industry study of mapreduce workloads. PVLDB, 5(12), 1802-1813.
- Nguyen, T. V. A., Bimonte, S., d'Orazio, L., & Darmont, J. (2012, March). Cost models for view materialization in the cloud. In Proceedings of the 2012 Joint EDBT/ICDT Workshops (pp. 47-54). ACM.
- Roukh, A., Bellatreche, L., Boukorca, A., & Bouarar, S. (2015, October). Eco-dmw: Eco-design methodology for data warehouses. In DOLAP (pp. 1-10). ACM.
- Cugola, G., & Margara, A. (2012). Processing flows of information: From data stream to complex event processing. ACM Computing Surveys (CSUR), 44(3), 15.

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

DTIM