Energon: A Data Acquisition System for Portable Building Analytics

Fang He[1], Yang Deng[1], Yanhui Xu[1],
Cheng Xu[2] , Dezhi Hong[3], Dan Wang[1]

[1] The Hong Kong Polytechnic University
[2] Johnson Electric [3] University of California, San Diego

Building Analytics

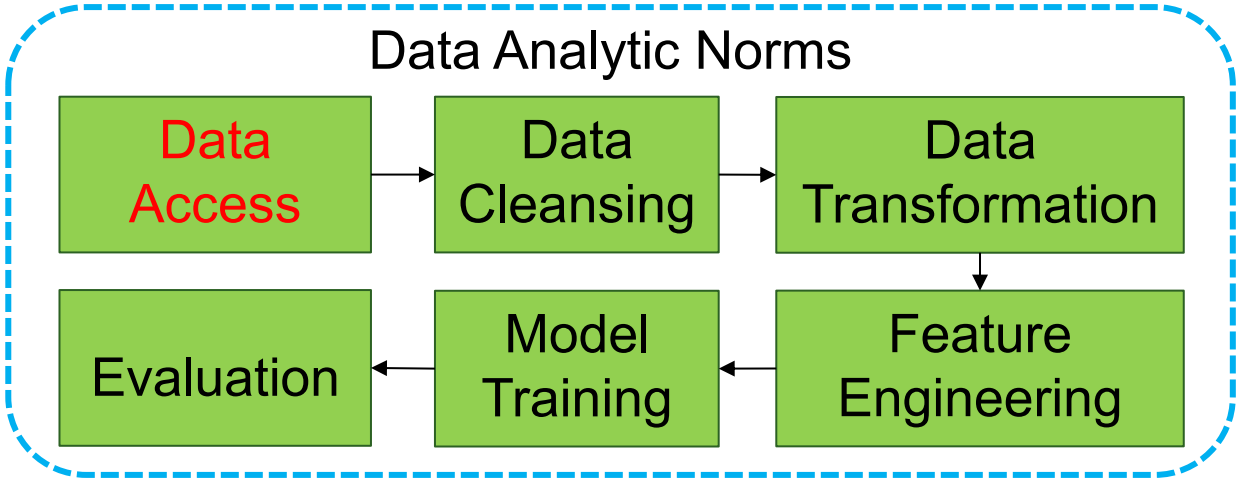
Building analytics—i.e. using data to develop machine learning (ML)-based methods for the operation and control of building systems.

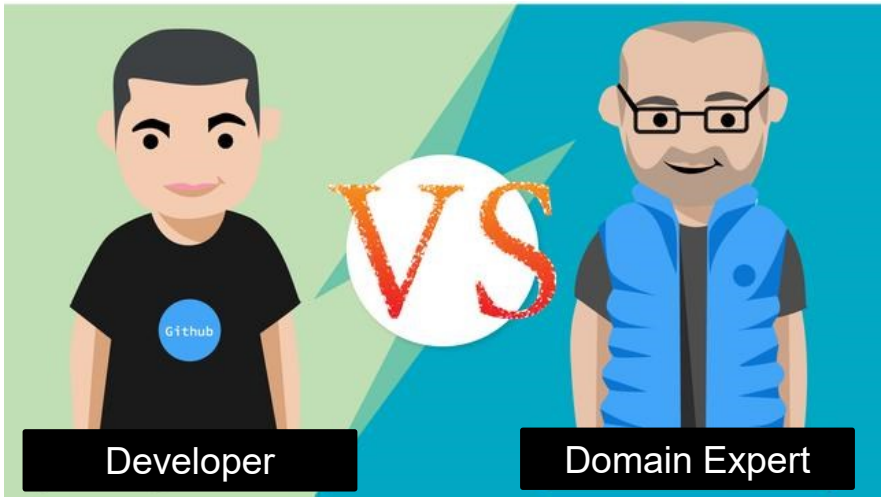
have proven to be effective in reducing energy footprints and operational costs, and improving the maintenance efficiency of buildings.

An increasing number of data-driven analytics being developed for buildings[1][2][3]

Requirements of developing data analytics

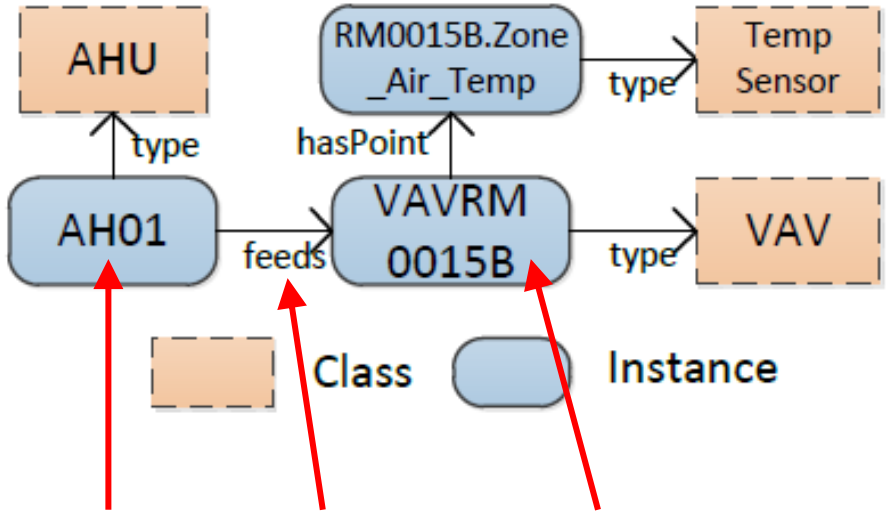
- Expertise about building analytics
  - Algorithm fits
  - Analytic norms
- Building domain knowledge
  - Data required
  - Data resource
  - The entities data describe





Standardized Management of Resources and Data in Buildings

Brick[4]

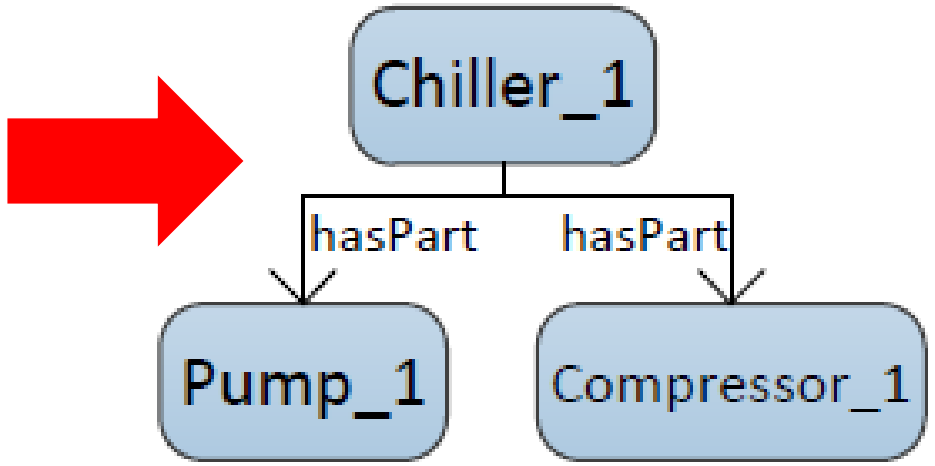


A RDF triple: <AH01, feeds, VAVRM0015B>

Building Resources and relationships are defined as RDF triples in Brick.

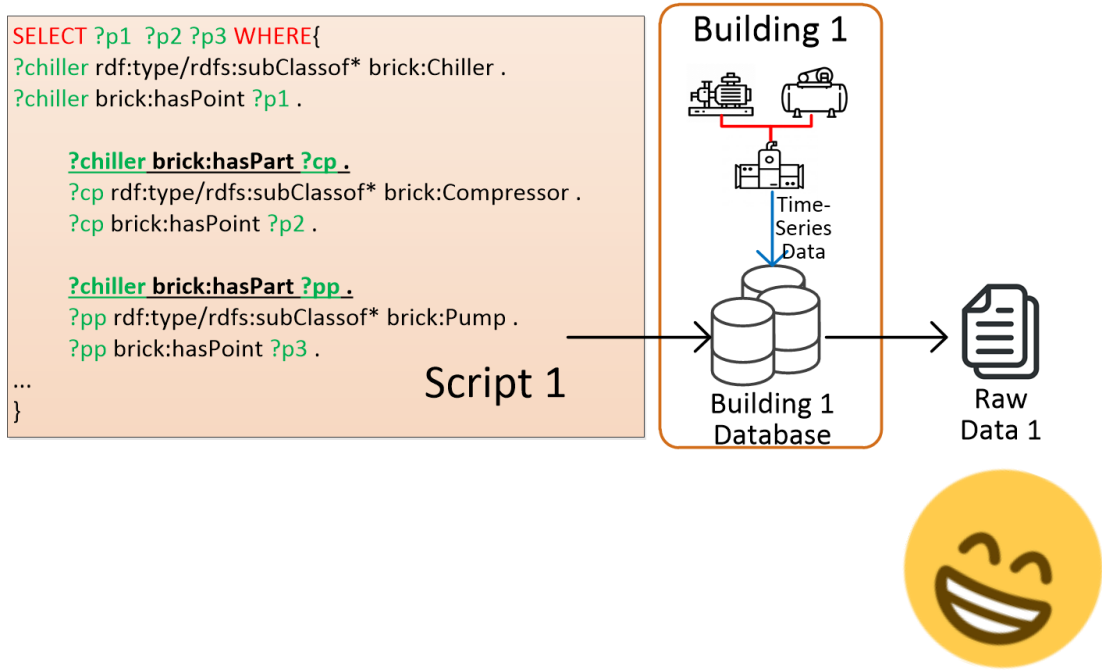
Brick: An uniform metadata schema for building

```
SELECT ?cps ?pps ?cpps WHERE {
    ?chiller rdf:type/rdfs:subClassOf* brick:Chiller .

    ?chiller brick:hasPoint ?cps .
    ?cps rdf:type/rdfs:subClassOf* brick:Power_Sensor .

    ?chiller brick:hasPart ?pp .
    ?pp rdf:type/rdfs:subClassOf* brick:Pump .
    ?pp brick:hasPoint ?pps .
    ?pps rdf:type/rdfs:subClassOf* brick:Power_Sensor .

    ?chiller brick:hasPart ?cp .
    ?cp rdf:type/rdfs:subClassOf* brick:compressor .
    ?cp brick:hasPoint ?cpps .
    ?cpps rdf:type/rdfs:subClassOf* brick:Power_Sensor .
}
```

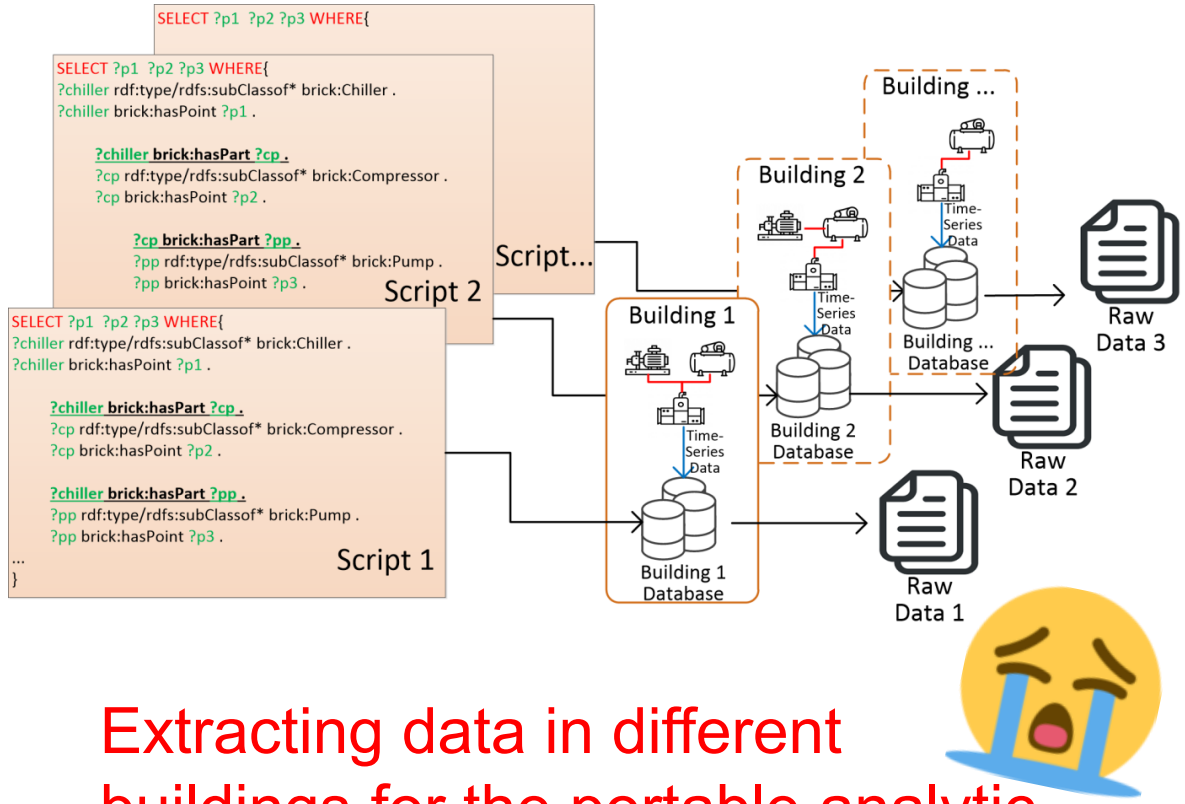


A SPARQL Query

Extract data for portable building analytics

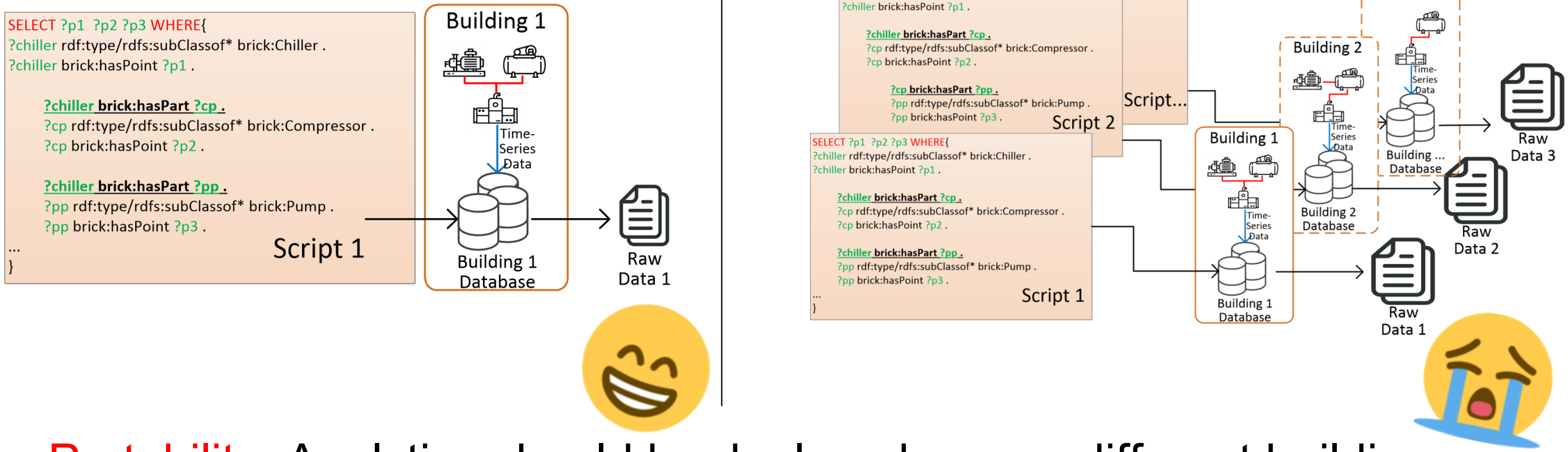


Extracting data in a single building



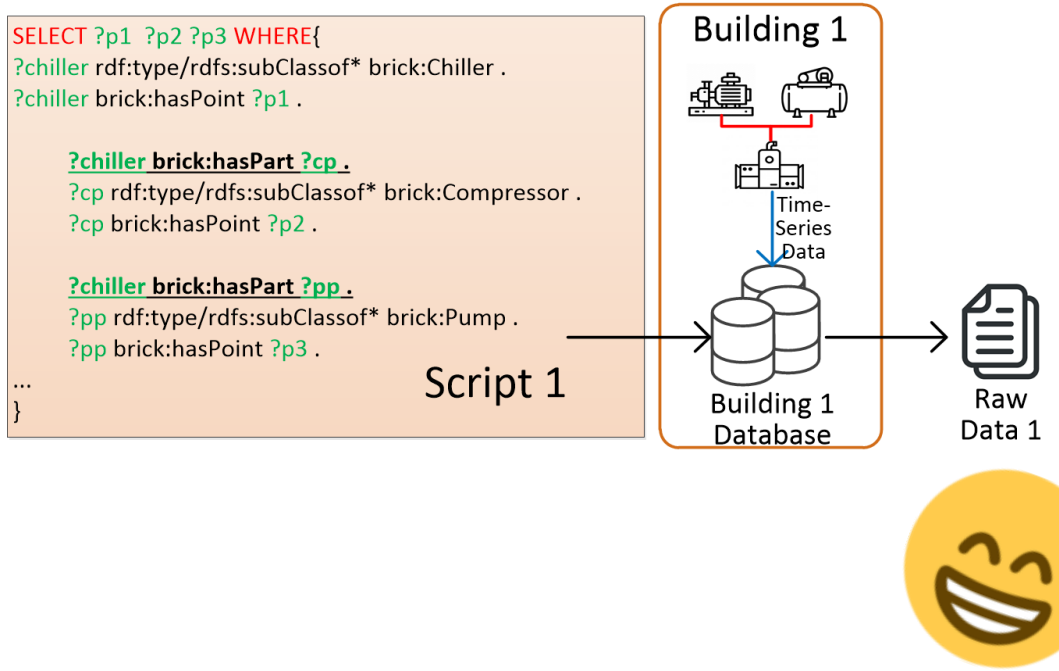
Extracting data in different buildings for the portable analytic

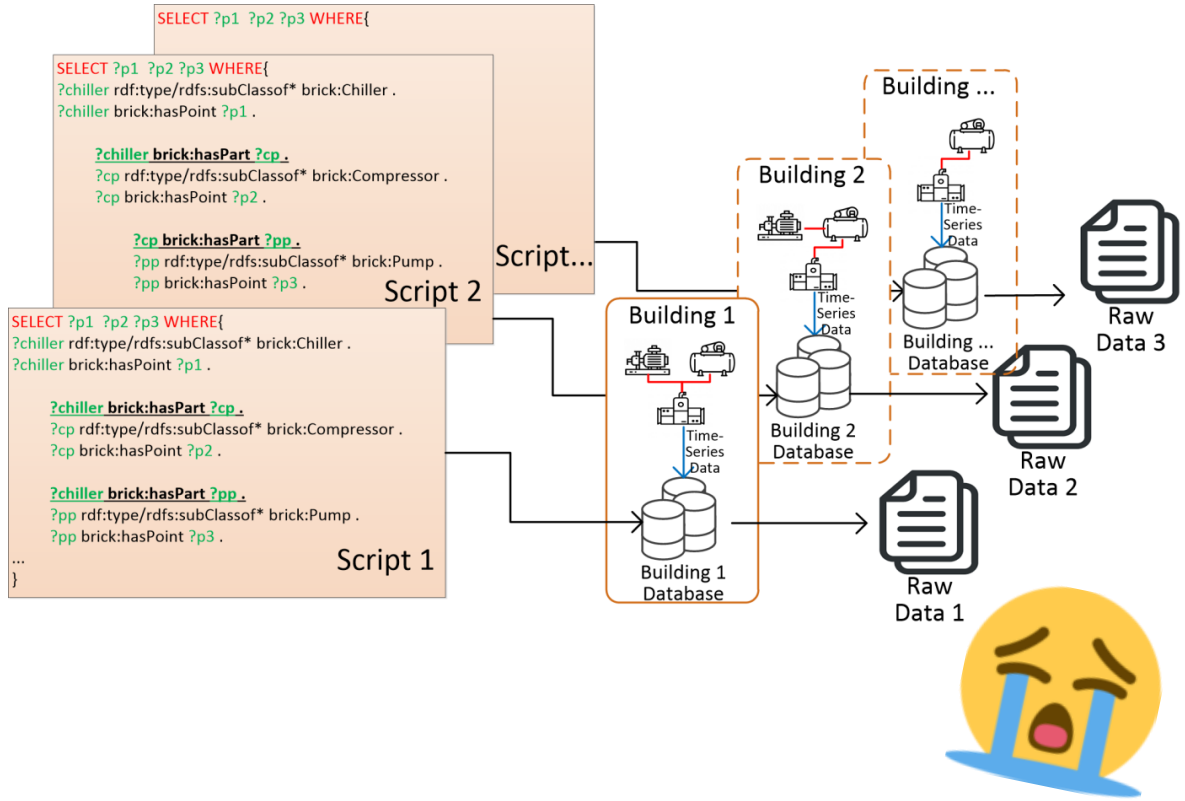
Extract data for portable building analytics



Portability: Analytics should be deployed across different buildings without requiring major changes to the implementation

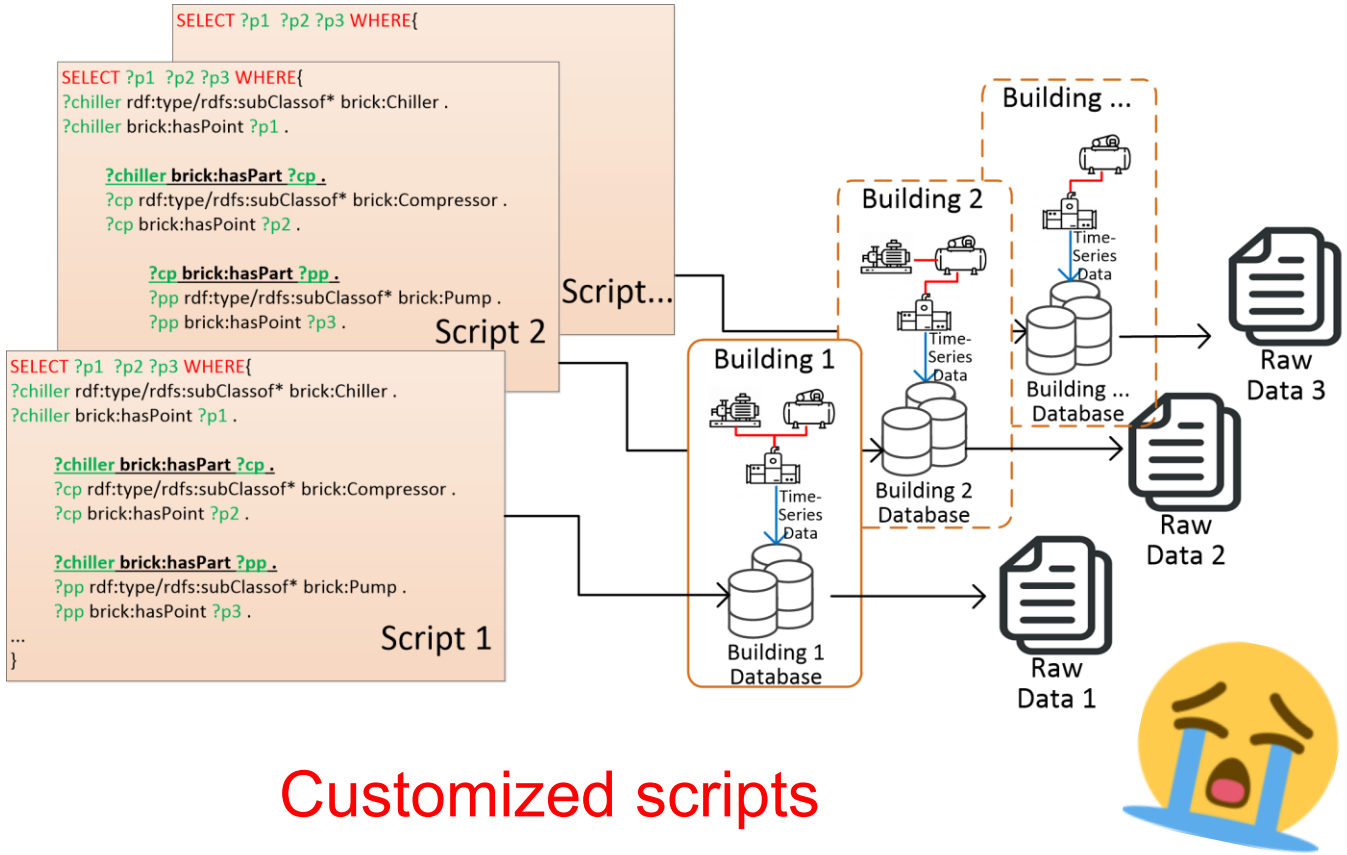
Extract data for portable building analytics



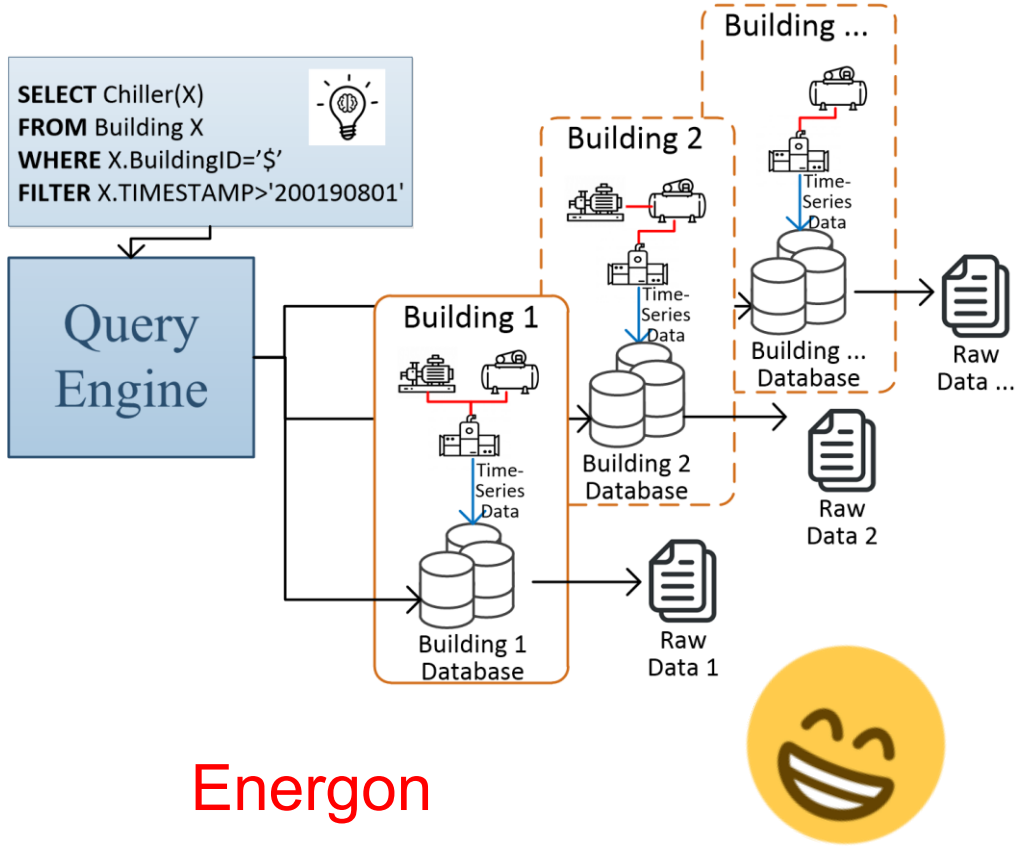


Can we extract data for building analytics in a **building-agnostic** way?

Energon: A Data Acquisition System for Portable Building Analytics

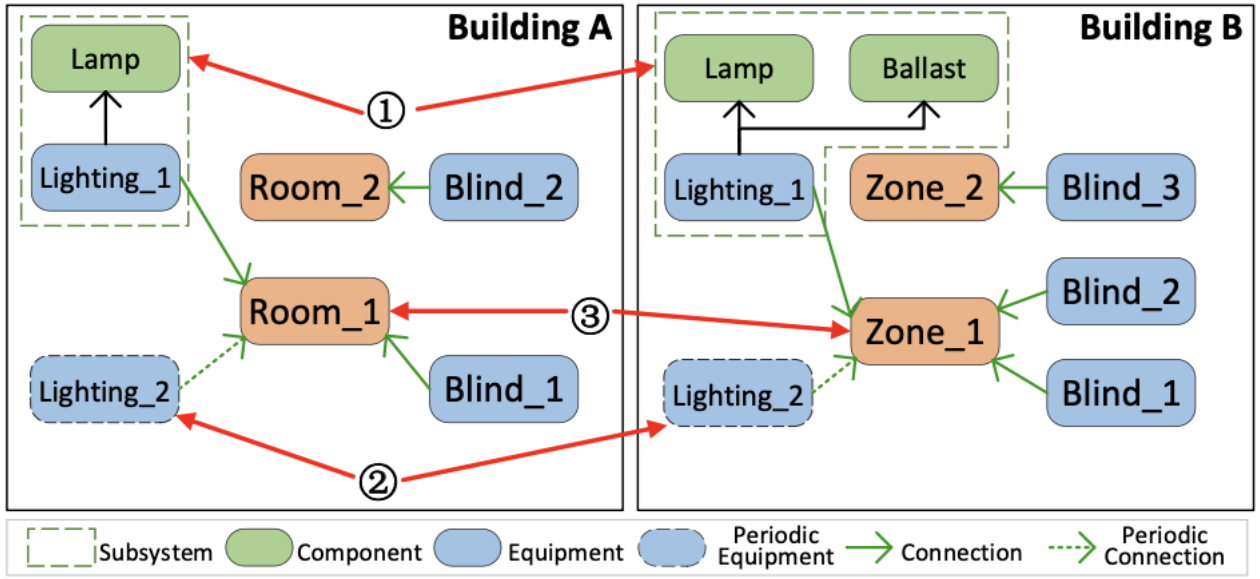


Customized scripts



Energon

Challenge: Building-dependent Auxiliary Knowledge is required



①: Spatial Auxiliary Knowledge
②: Temporal Auxiliary Knowledge
③: Contextual Auxiliary Knowledge

Challenge: Building-dependent Auxiliary Knowledge is required

```
SELECT ?lpsp ?blsp WHERE {
    ?room(?zone) rdf:type/rdfs:subClassof* brick:Room(Zone) .
    ?lighting rdf:type/rdfs:subClassOf* brick:Lighting_System .

    ?room(?zone) brick:isLocationOf ?lighting .

    ?lamp rdf:type/rdfs:subClassOf* brick:Luminance .
    ?light brick:hasPart ?lamp .
    ?lamp brick:hasPoint ?lpsp .
    ?lpsp rdf:type/rdfs:subClassOf* brick:Setpoint .

    ?blst rdf:type/rdfs:subClassOf* brick:Ballast .
    ?light brick:hasPart ?blst .
    ?blst brick:hasPoint ?blsp .
    ?blsp rdf:type/rdfs:subClassOf* brick:Setpoint .
    }
```

Data Tracing Flow

Design Approach: Logic Views

Func. \ Subsys.	AHU system	VAV system	Chiller System	Weather	Zone	Lighting	Blind
Temperature	(3)(5)	(6)	(1)	(1)(2)(3)	(2)(3)		
Humidity	(5)	(6)		(2)			
Pressure	(5)	(3)(6)	(7)				
Flow Rate	(3)(5)	(6)	(1)(2)(7)	(3)	(2)		
Power	(3)	(3)	(1)(3)(7)				
Solar Radiance				(3)(4)			
Solar Angle				(4)			
Control Signal	(5)	(6)					
Setpoint	(5)	(3)(6)				(4)	(4)
Analytics	Profiling	(1) Chiller Profiling					
	MPC	(2) PMV Prediction	(3) ECP	(4) Building Integrated Control			
	FDD	(5) FDD for AHU	(6) FDD for VAV	(7) FDD for Chiller			

Design Approach: Query Language

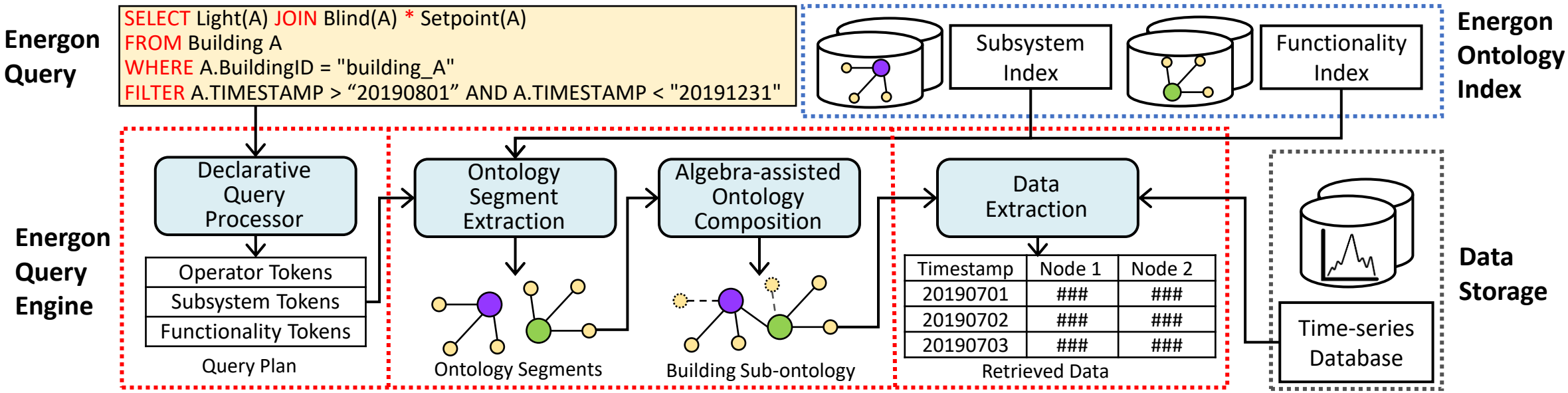
Category / Analytics	*Subsystem*			*Functionality*	Query Operation
	Spatial	Temporal	Contextual		
FDD-Chiler	✓			✓	(1) (3)
FDD-VAV	✓			✓	(1) (3)
FDD-AHU	✓			✓	(1) (3)
CP	✓	✓		✓	(1)(2)(3)
IAQ	✓	✓	✓	✓	(1)(2)(3)(4)
BIC	✓	✓	✓	✓	(1) (3)(4)
ECP	✓		✓	✓	(1) (3)(4)
Query Operation	(1) Select, From, Where; (2) Filter; (3) Union, Difference, Intersect; (4) Join;				

Energon Query for BIC

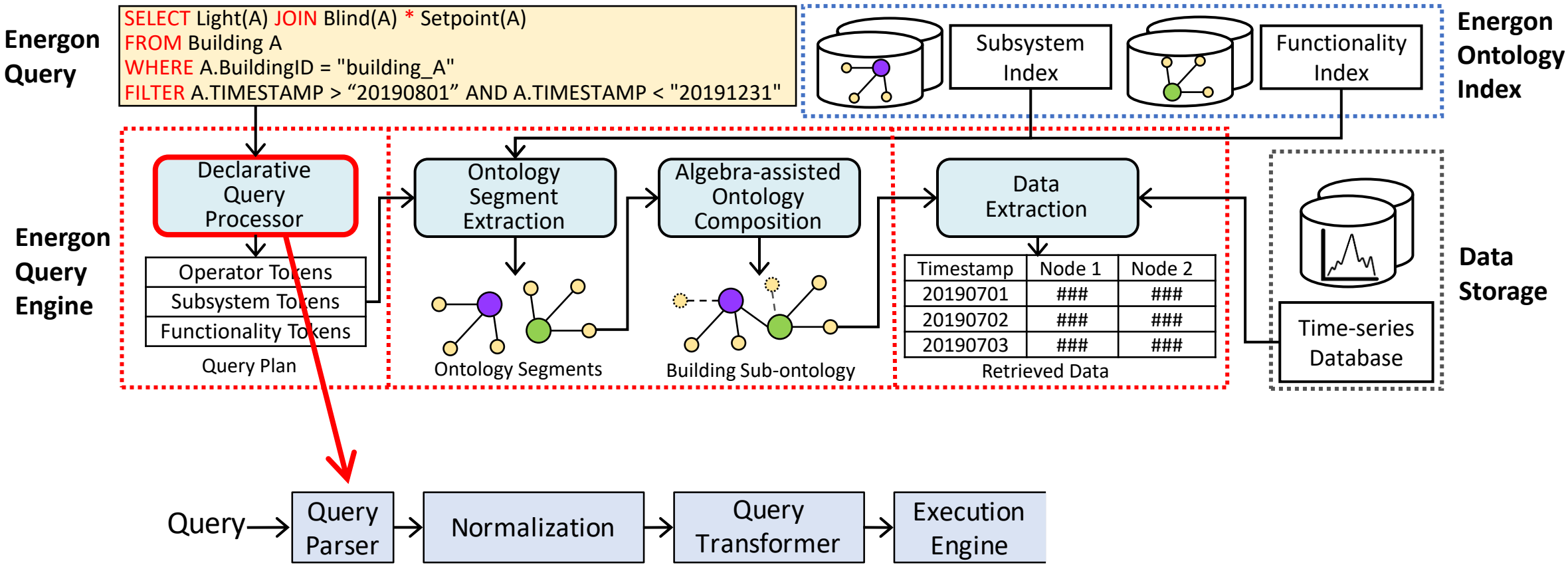
```
SELECT Lighting(B) JOIN Blind(B) *
Setpoint(B)
FROM Building B
WHERE B.BuildingID = ‘PU’
FILTER B.TIMESTAMP > ‘20190801’
```

- Algebra to perform building traversals
- List of buildings to determine boundings
- Predicate expressions to perform resource selections
- Predicate expressions to perform data selections

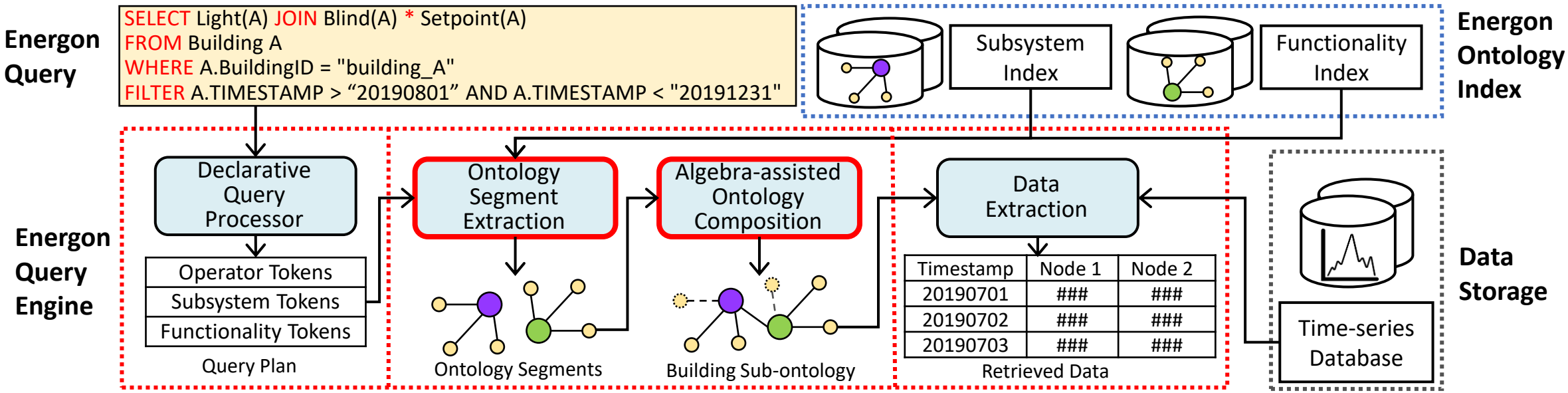
Design Overview of Energon System



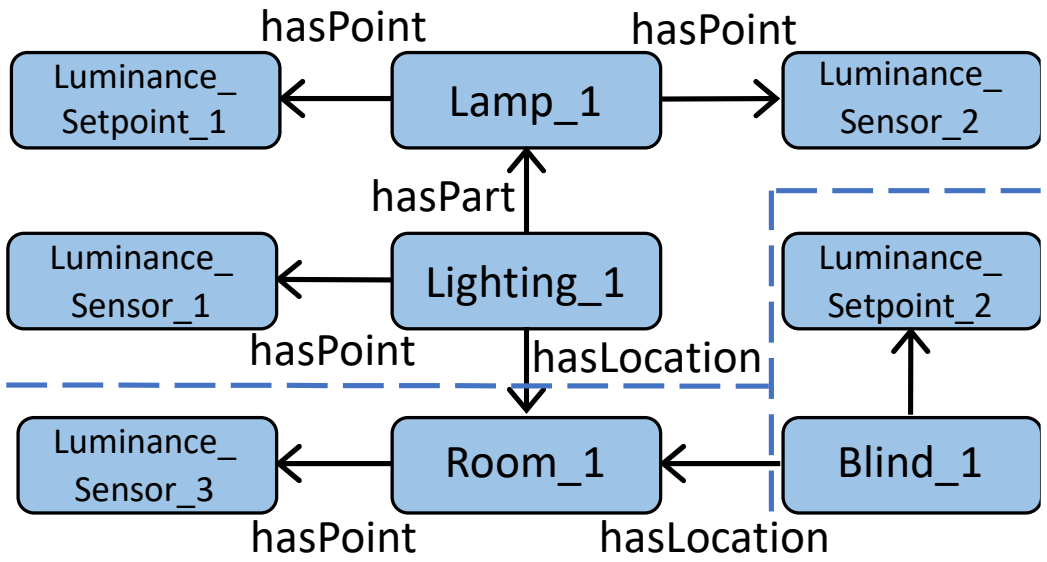
Declarative Query Processor



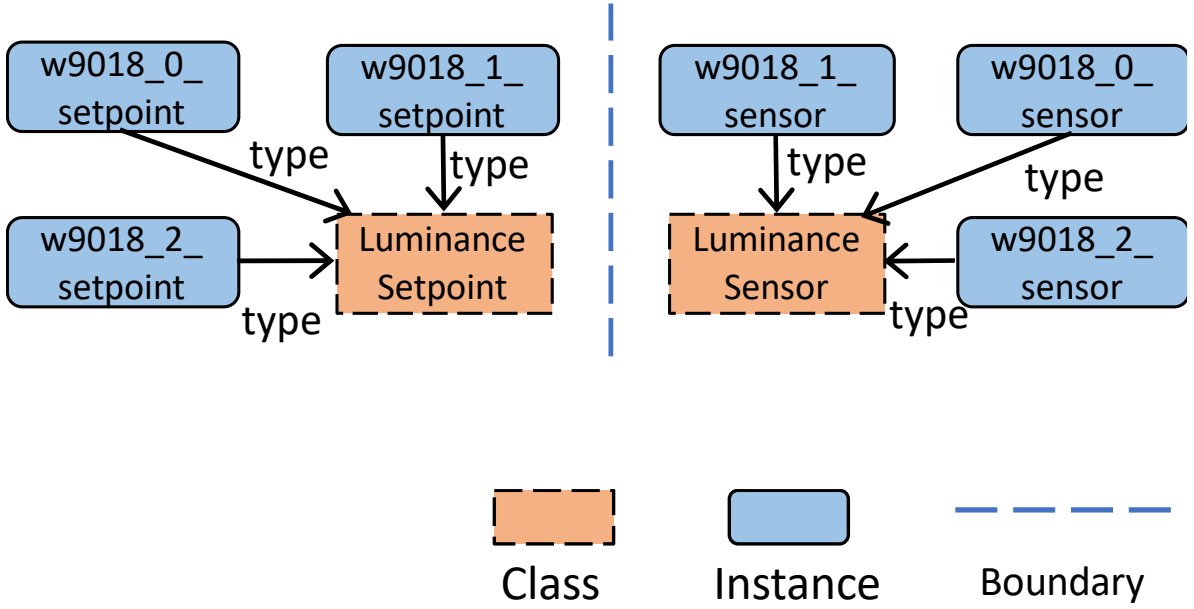
Building Independent Ontology Extraction



Ontology Segment Extraction



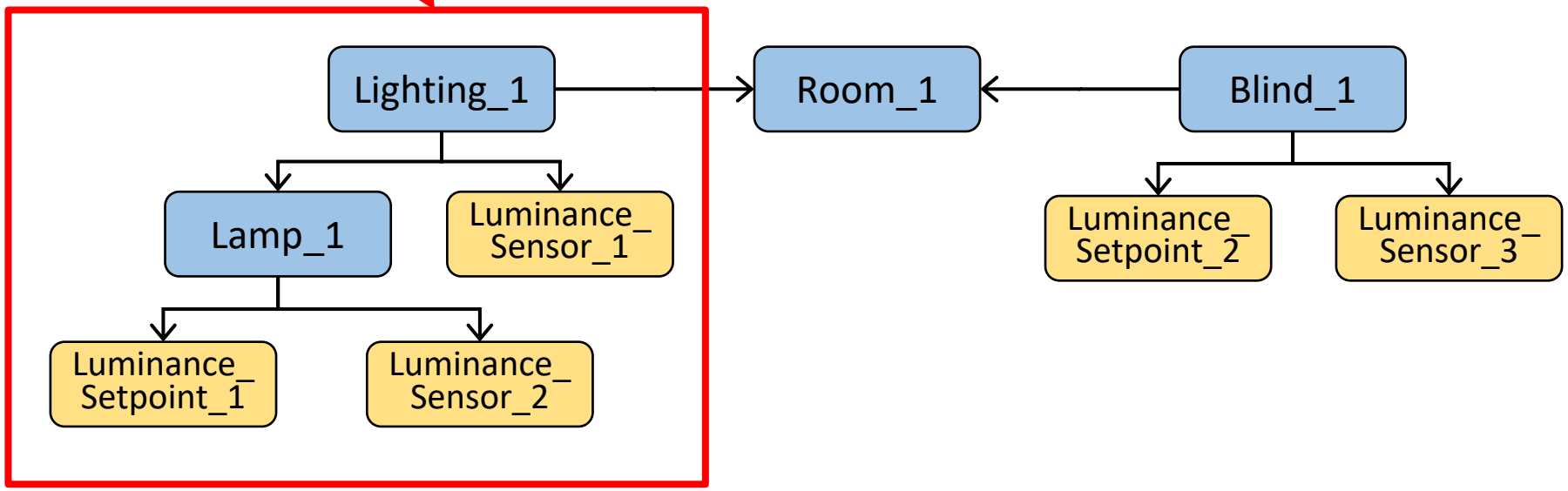
Subsystem Ontology Segment



Functionality Ontology Segment

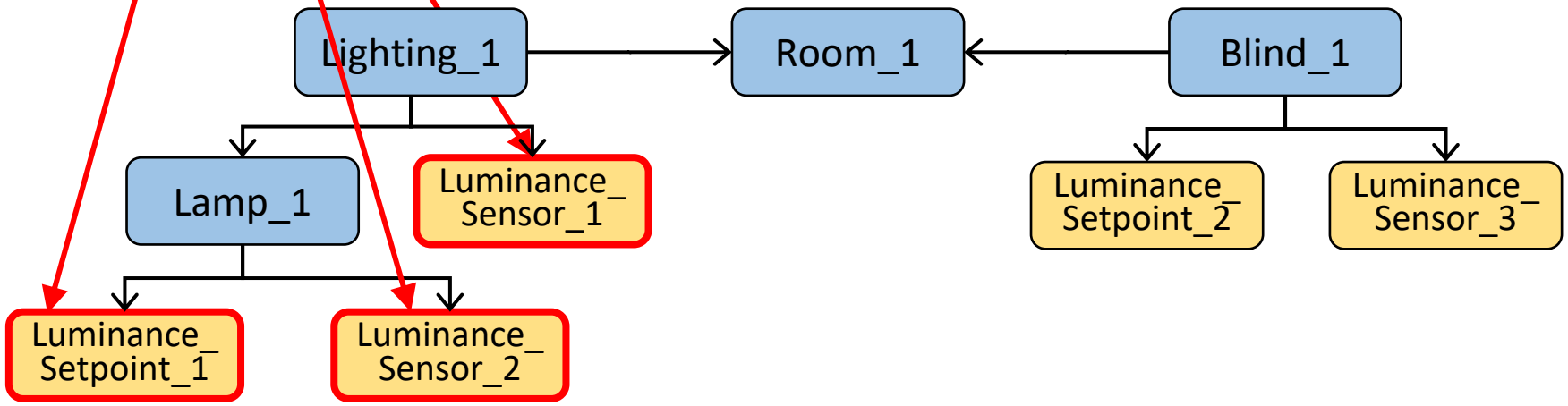
Ontology Segment Extraction for BIC

```
SELECT Lighting(B) * Setpoint(B)
FROM Building B
WHERE B.BuildingID = 'PU'
FILTER B.TIMESTAMP > '20190801'
```



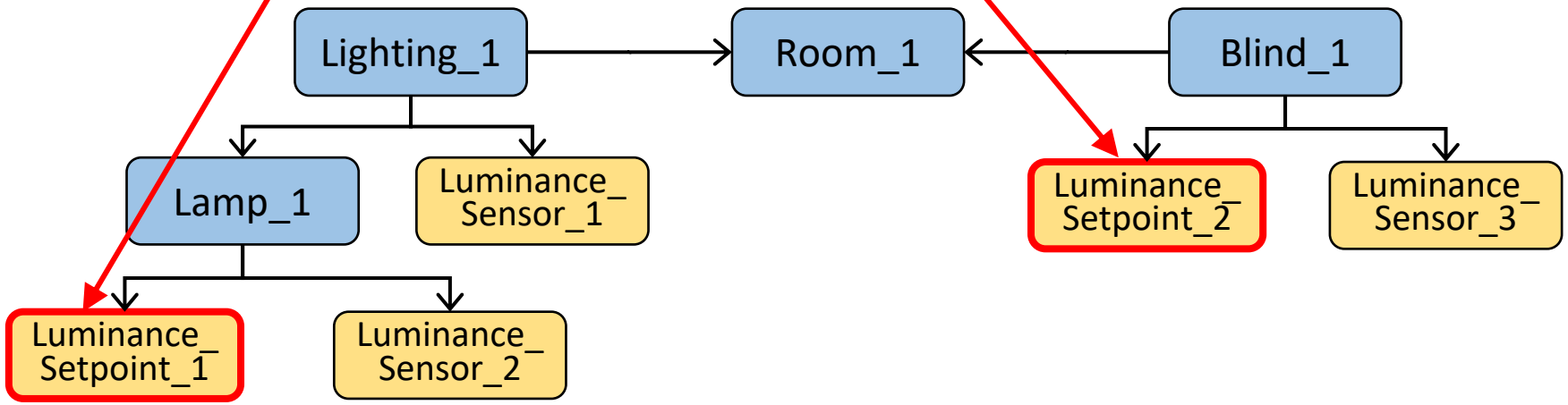
Ontology Segment Extraction for BIC

```
SELECT Lighting(B) * Setpoint(B)
FROM Building B
WHERE B.BuildingID = 'PU'
FILTER B.TIMESTAMP > '20190801'
```



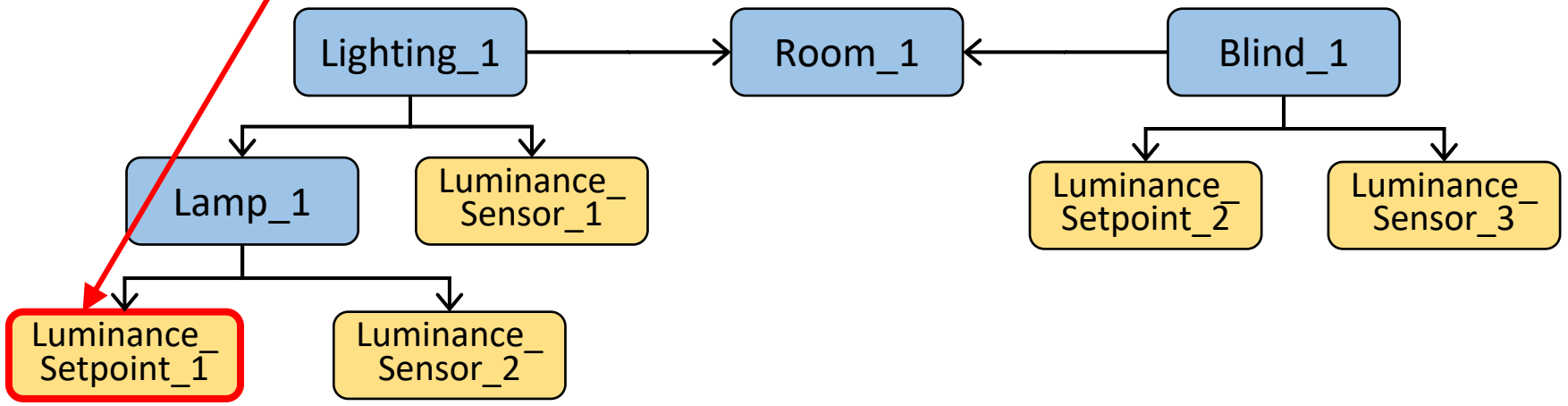
Ontology Segment Extraction for BIC

```
SELECT Lighting(B) * Setpoint(B)
FROM Building B
WHERE B.BuildingID = 'PU'
FILTER B.TIMESTAMP > '20190801'
```



Algebra-assisted Ontology Composition

```
SELECT Lighting(B) * Setpoint(B)
FROM Building B
WHERE B.BuildingID = 'PU'
FILTER B.TIMESTAMP > '20190801'
```



Algebra-assisted Ontology Composition

- **Union** $A + B = \{x \mid x \in A \; or \; x \in B\}$.
- **Intersection** $A * B = \{x \mid x \in A \; and \; x \in B\}$.
- **Difference** $A - B = \{x \mid x \in A \; and \; x \notin B\}$.
- **Join** $A \bowtie B = \{x \mid x = a \; or \; b, where \; a \in A, b \in B \; and \; a \leftrightarrow b)\}$,

" $\leftrightarrow$ " represents a situation where at least one predicate exists between a and b, i.e. a and b are connected in the ontology graph.

Algebra-assisted Ontology Composition

- **Join** $A \bowtie B = \{x \mid x = a \text{ or } b, where\ a \in A, b \in B \text{ and } a \leftrightarrow b)\}$,

" $\leftrightarrow$ " represents a situation where at least one predicate exists between a and b, i.e. a and b are connected in the ontology graph.

Lighting System

Subject	Predicate	Object
Lighting_1	hasLocation	Room_1
Lighting_1	hasPoint	Luminance11
Lighting_1	hasPart	Lamp_1
Lighting_2	hasLocation	Room_1
Lighting_2	hasPoint	Luminance21

Room

Subject	Predicate	Object
Room_1	isLocationOf	Blind_1
Room_2	isLocationOf	Blind_2

Blind System

Subject	Predicate	Object
Blind_1	hasPoint	Slat_Setpoint1
Blind_2	hasPoint	Slat_Setpoint2

Lighting System Join Blind System

Subject	Predicate	Object (Subject)	Predicate	Object
Lighting_1	hasLocation	Room_1	isLocationOf	Blind_1
Lighting_2	hasLocation	Room_1	isLocationOf	Blind_1

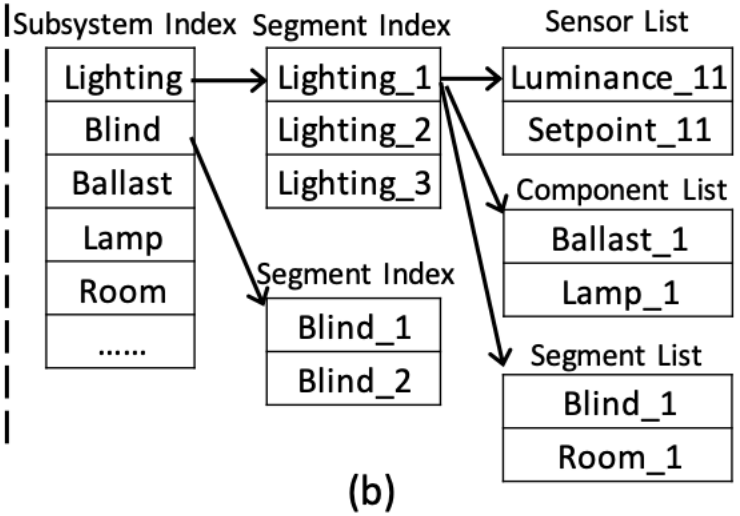
Join Results

Subsystem	Sensors
Lighting_1	Luminance11
Lighting_2	Luminance21
Blind_1	Slat_Setpoint1

Ontology Index

Subject	Predicate	Object
Lighting_1	hasLocation	Room_1
Lighting_1	hasPoint	Setpoint_11
Lighting_1	hasPart	Ballast_11
Lighting_1	hasPart	Lamp_1
Room_1	isLocationOf	Blind_1
.....	.....	.....

(a)



(b)

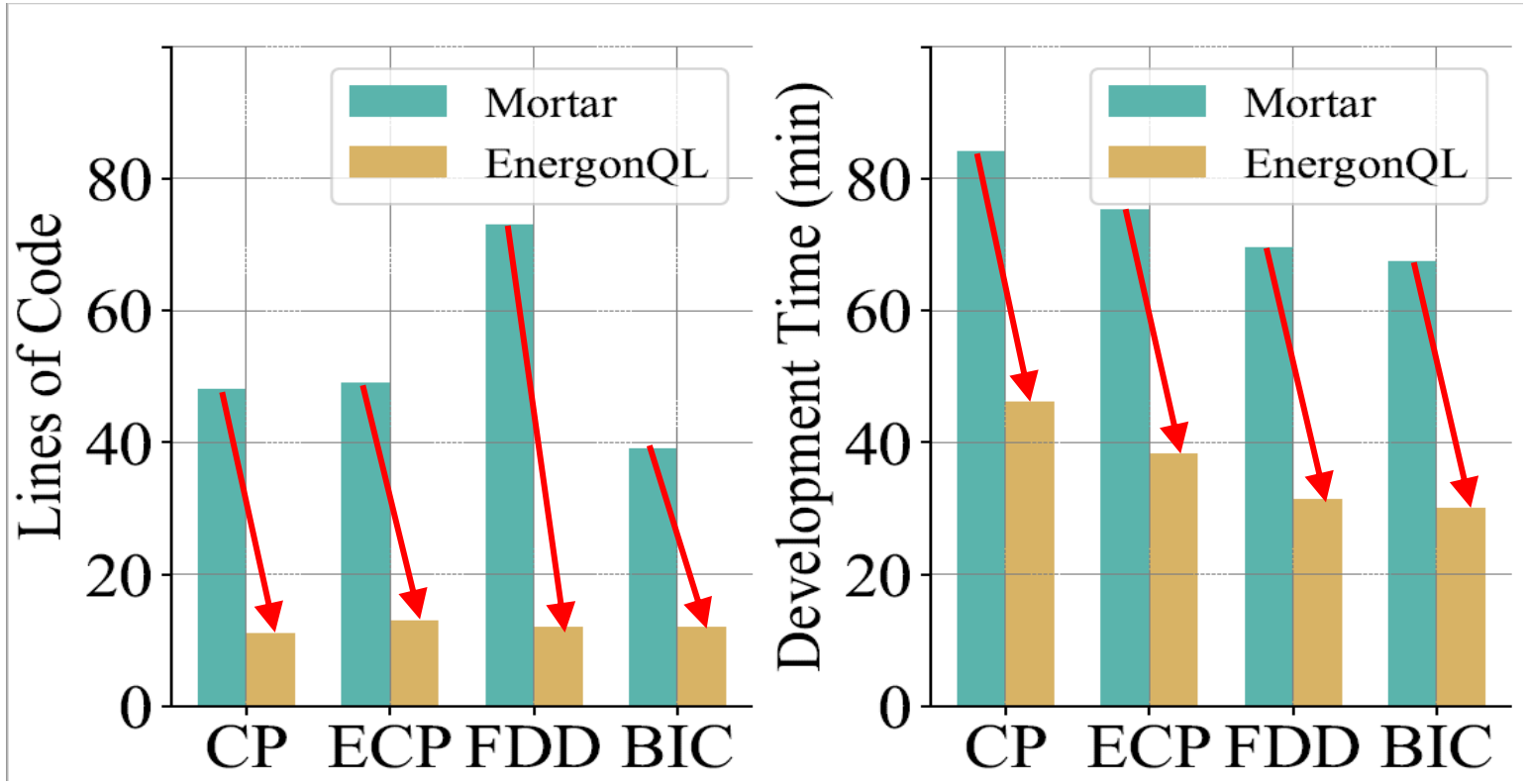
(a) Ontology store in RDF triples;
(b) Energon Ontology Index for optimized performance

Qualitative Evaluation

```
# 1. SPARQL query for ontology extraction
bic_query = '''
    SELECT ?lpsp ?llum ?bsp ?blum ?ans ?srs WHERE {
    ?room rdf:type/rdfs:subClassof* brick:Room .
    ?lighting rdf:type/rdfs:subClassOf* brick:Lighting_System .
    ?blind rdf:type/rdfs:subClassOf* brick:Shading_System .

    ?room brick:isLocationOf ?lighting .
    ?room brick:isLocationOf ?blind

    ?lamp rdf:type/rdfs:subClassOf* brick:Luminance .
    ?lpsp rdf:type/rdfs:subClassOf* brick:Setpoint .
    ?llum rdf:type/rdfs:subClassOf* brick:Luminance_Sensor .
    ?light brick:hasPart ?lamp .
    ?lamp brick:hasPoint ?lpsp .
    ?lamp brick:hasPoint ?lum .

    ?bsp rdf:type/rdfs:subClassOf* brick:Setpoint .
    ?blum rdf:type/rdfs:subClassOf* brick:Luminance_Sensor .
    ?blind brick:hasPoint ?bsp .
    ?blind brick:hasPoint ?blum .

    ?wea rdf:type/rdfs:subClassOf* brick:Weather .
    ?ans rdf:type/rdfs:subClassOf* brick:Angle_Sensor .
    ?srs rdf:type/rdfs:subClassOf* brick:Solar_Radiance_Sensor .
    ?wea brick:hasPoint ?ans .
    ?wea brick:hasPoint ?srs .
    }
    '''

# 2. data extraction and encapsulation
request = pymortar.FetchRequest(
    # Define building 'building_A' as data source
    sites=['building_A'],
    views=[
        pymortar.View(
            name='data_points',
            query=bic_query,
        ),
    ],
    # Data format is omitted here, e.g. time series interval
    # and aggregation method
    ...
    # Define the time window
    time=pymortar.TimeParams(
        start='2019-08-01T00:00:00Z',
        end='2019-12-30T00:00:00Z',
    )
)

result = fetch(request)
data = result['data'][data_list]
```

```
from Energon.EnergonQL import *
from ontology.ontology_bic import global_ontology

# Load the complete building ontology for BIC
global_ontology()

# 1. Energon Query for ontology and data extraction
bic_query = '''
    SELECT Light(A) JOIN Blind(A) * (Luminance(A) + Setpoint(A)) +
          Weather(A) * (Solar_Angle(A) + Solar_Radiance_Rate(A))
    FROM Building A
    WHERE A.BuildingID = 'LightZone' AND A.Source = 'Local'
    FILTER A.TIMESTAMP > '20190801' AND A.TIMESTAMP < '20191231'
    '''
# 2. execute the query to retrieve the data
data = fetch(bic_query)
```

Qualitative Evaluation: Development Effort



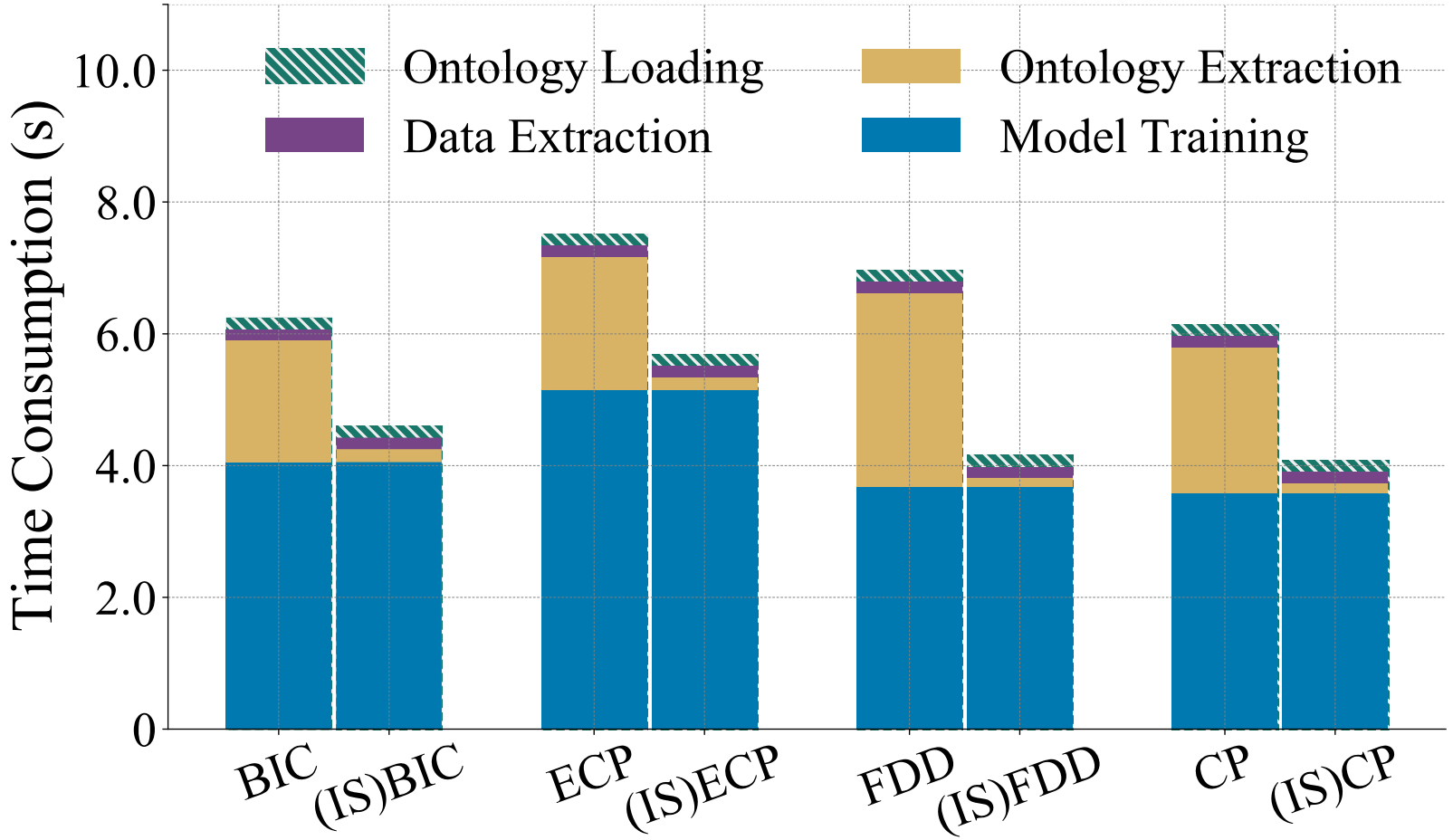
CP: Chiller Profiling

ECP: Energy Consumption Prediction

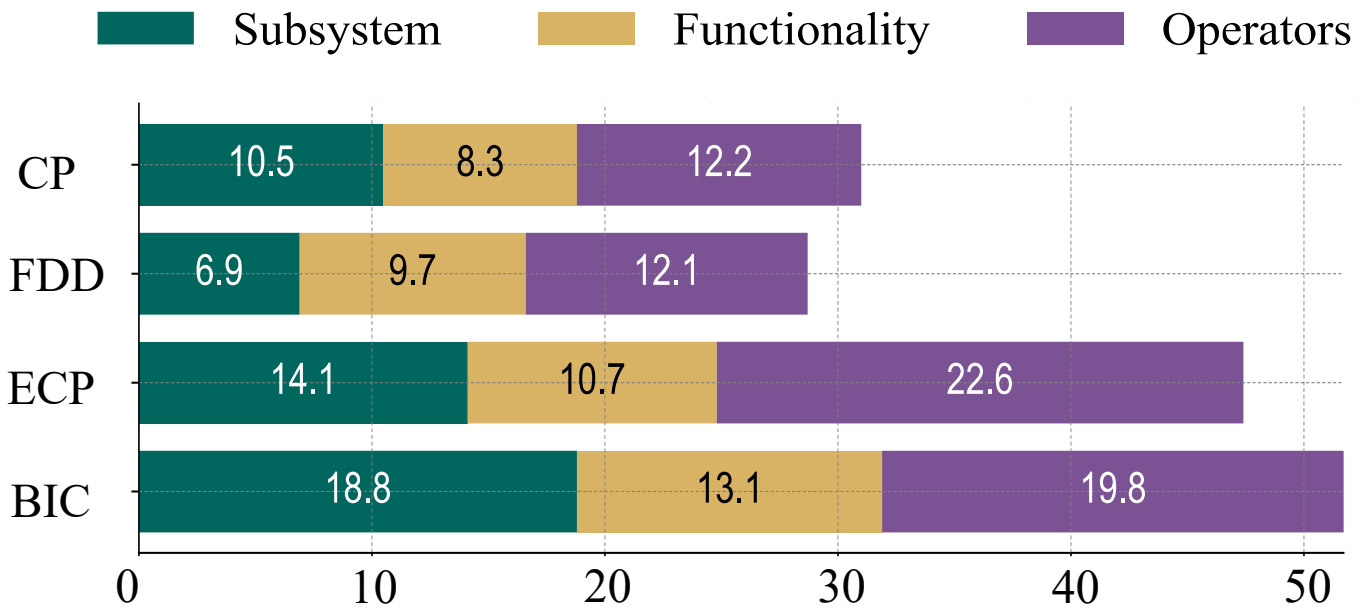
FDD: Fault Diagnosis and Detection

BIC: Building Integrated Control

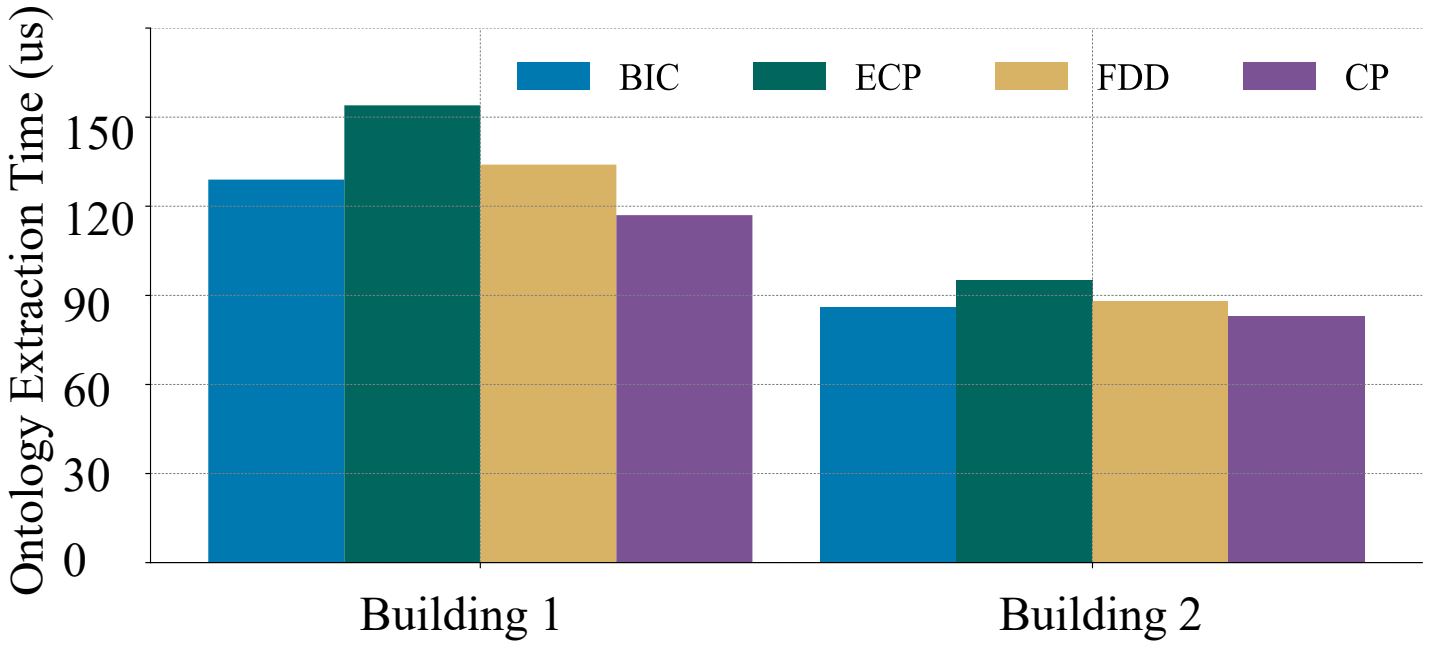
Quantitative Evaluation: Indexing Structure



Quantitative Evaluation: Indexing Structure



Quantitative Evaluation: Across Buildings



Conclusion

- We present a new abstraction of building resources so that developers do not need to have building-specific knowledge when developing building analytics; thus the development of applica- tion can be simplified. The abstraction is based on two kinds of logic partitions over building resources, namely, subsystem and functionality.
- •We present the design and implementation1 of Energon to materialize the abstraction. We develop a declarative query language to evade building-dependent and analytics irrelevant knowledge. We develop a query engine for automatically extracting data by traversing a building ontology that widely exists in buildings. We further develop an indexing structure to optimize query execution time.
- We develop four types of building analytics through Energon, and we qualitatively show that the development process becomes simpler when using Energon.
- We quantitatively evaluate Energon with regards to program length and development time. We evaluate our optimization schemes designed for execution time in Energon.

Reference

[1] Z. Zheng, Q. Chen, C. Fan, N. Guan, A. Vishwanath, D. Wang, and F. Liu. 2018. Data Driven Chiller Sequencing for Reducing HVAC Electricity Consumption in Commercial Buildings. In Proc. ACM e-Energy '18. 236–248.
[2] E. Shen, J. Hu, and M. Patel. 2014. Energy and visual comfort analysis of lighting and daylight control strategies. Building and Environment 78 (2014), 155–170.
[3] M. Najafi, D. M. Auslander, P. L. Bartlett, P. Haves, and M. D. Sohn. 2012. Application of machine learning in the fault diagnostics of air handling units. Applied Energy 96 (2012), 347–358.
[4] B. Balaji, A. Bhattacharya, G. Fierro, J. Gao, J. Gluck, D. Hong, A. Johansen, J. Koh, J. Ploennigs, Y. Agarwal, M. Berges, D. Culler, R. Gupta, M. Kjægaard, M. Srivastava, and K. Whitehouse. 2016. Brick: Towards a Unified Metadata Schema For Buildings. In Proc. ACM BuildSys'16. 41–50.
[5] G. Fierro, M. Pritoni, M. Abdelbaky, P. Raftery, T. Peffer, G. Thomson, and D. Culler. 2018. Mortar: An Open Testbed for Portable Building Analytics. In Proceedings of the 5th Conference on Systems for Built Environments (BuildSys '18). 172–181.

Thank you!

Q&A