A Common Machine-Readable Vocabulary and Knowledge Base Supporting Multiple Programming Models

March 16th, 2021

CASC
Center for Applied Scientific Computing

ASCR Workshop on Reimagining Codesign

Chunhua “Leo” Liao

LLNL-PRES-820348

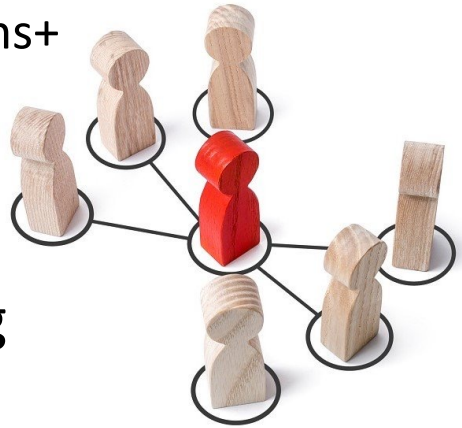
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

The Need

Programming Models

- Goal: providing high-level abstractions for programmers to express algorithms+ generating correct, fast and efficient executables
- Components: languages + compilers + runtime systems + performance and correctness tools …



A large body of knowledge is required to build efficient programming models

- Applications: domain-specific optimization knowledge, performance models
- Programming Language: semantics for language constructs
- Compilers: eligibility and profitability of optimizations
- Libraries: standard and user-defined semantics, such as read-only, aliasing, continuous memory
- Hardware: architectural features and corresponding optimization opportunities

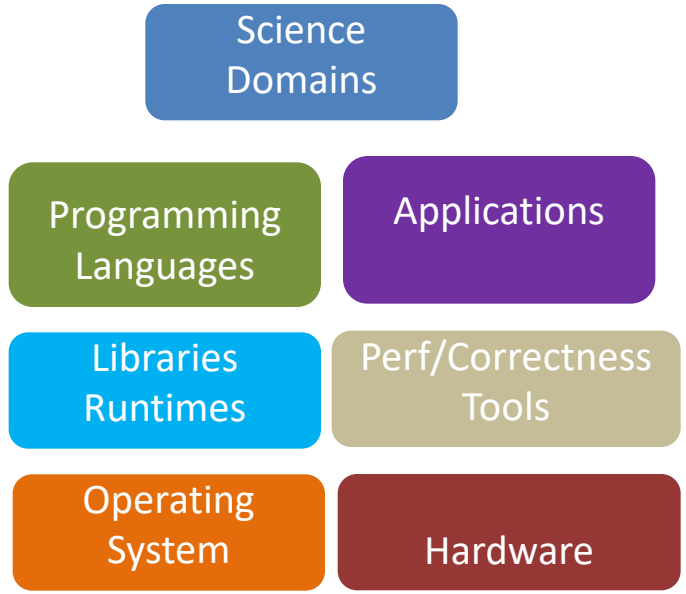
The State of The Art

Knowledge needed for programming models is implicitly assumed or represented using ad-hoc approaches

- Vocabularies: different communities may talk about the same concepts, using very different terms. How to collaborate then?
- Informal: not accessible, not interoperable, no verifiable, not scalable, ….
- Information gets lost across different layers
- Not reusable across different programming models

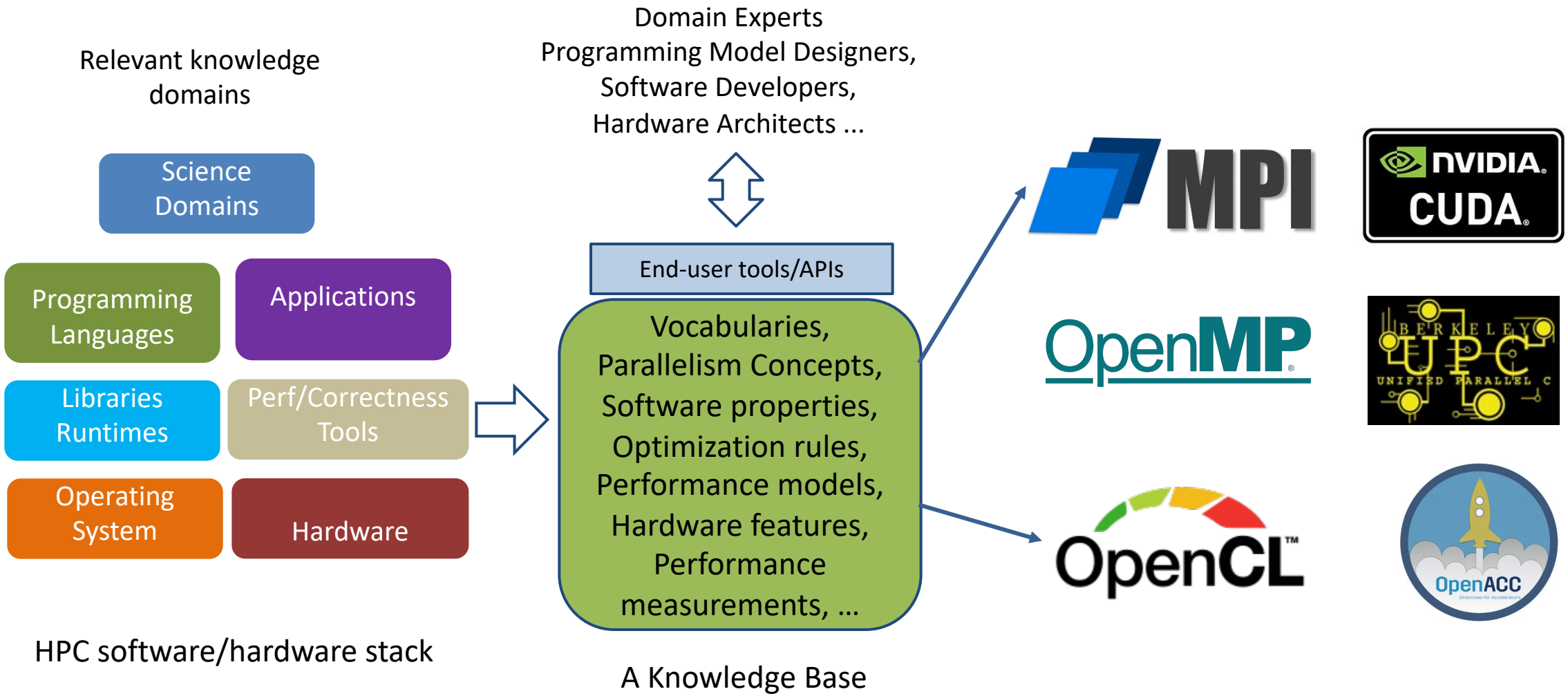
Every programming model, for every domain

- Re-inventing the wheels to bridge the semantics/knowledge gap



HPC software/hardware stack

Reimagining the Ecosystem of Programming Models



Solution: Knowledge Graphs= Vocabularies + Ontologies+…

Knowledge can be modeled using knowledge graphs

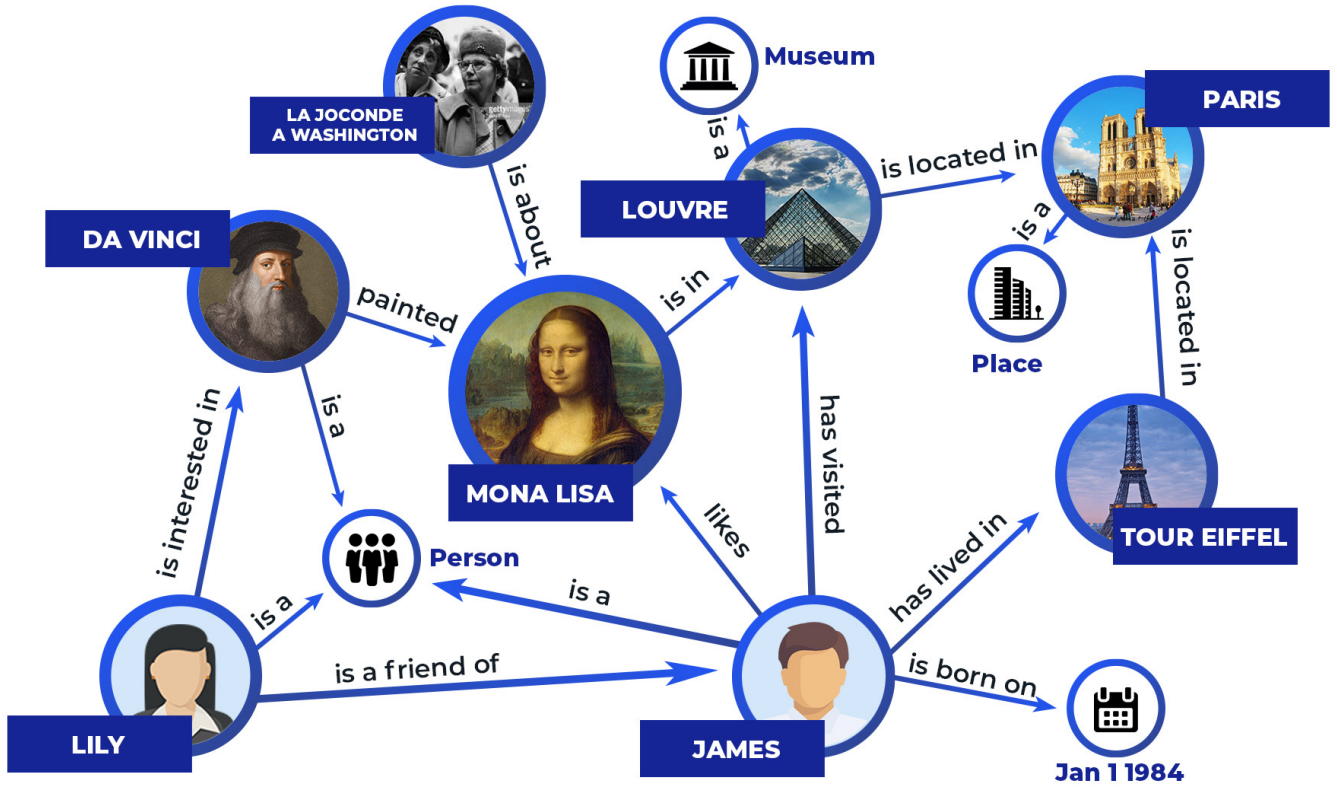
- Nodes: Classes/Objects
- Edges: Properties

Data Model: Resource Description Framework (RDF)

- Subject-Predicate-Object
- :JAMES :isBornOn :Jan-1-1984

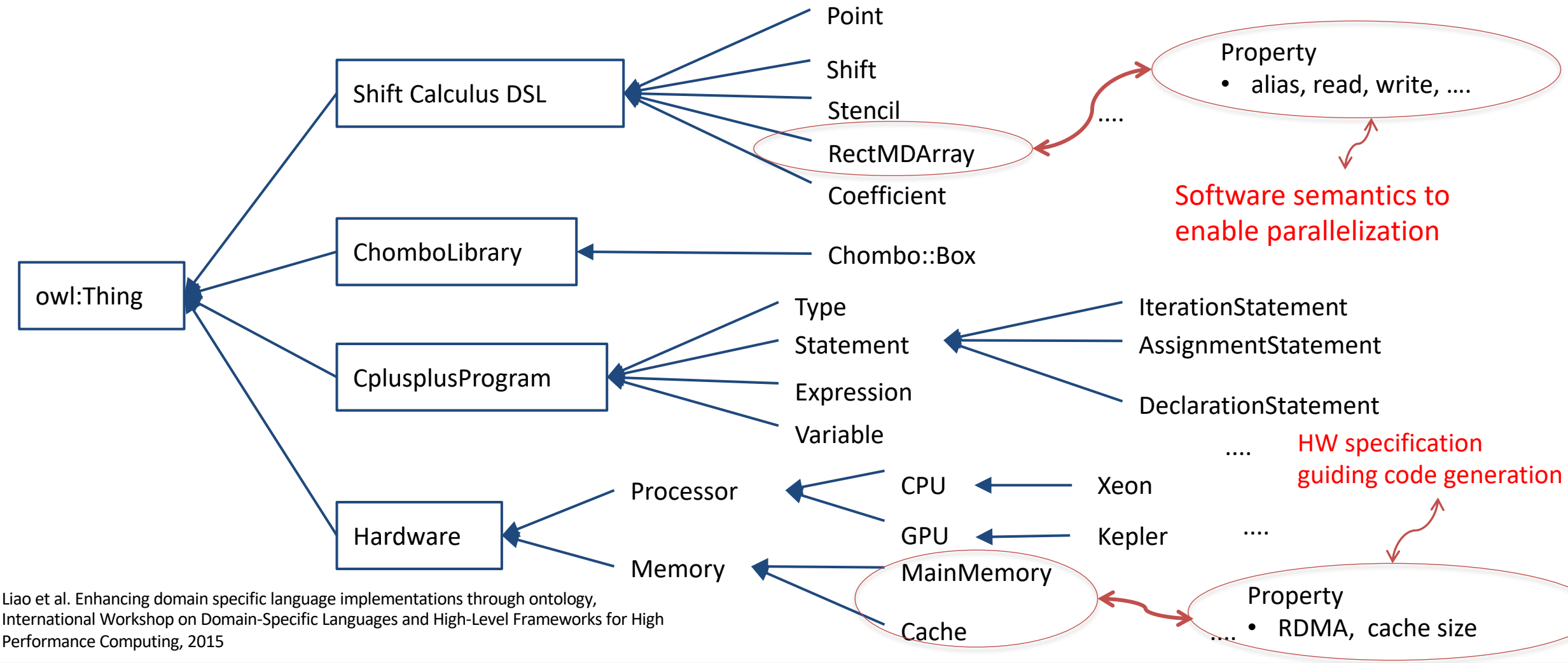
Queried using SPARQL

```
SELECT ?capital ?country
  WHERE
{   ?x ex:cityname ?capital ;
         ex:isCapitalOf ?y .
    ?y ex:countryname ?country ;
        ex:isInContinent ex:Africa . }
```



Credit:https://yashuseth.blog/2019/10/08/introduction-question-answering-knowledge-graphs-kgqa/

Building Vocabulary and Ontology for HPC

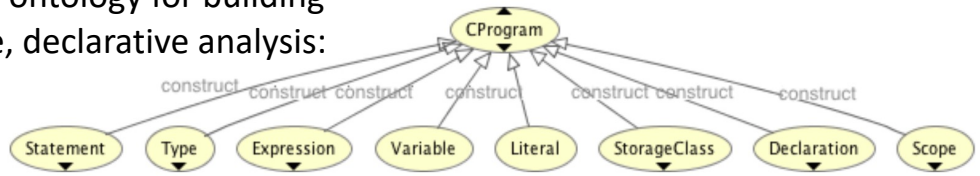


Liao et al. Enhancing domain specific language implementations through ontology, International Workshop on Domain-Specific Languages and High-Level Frameworks for High Performance Computing, 2015

Benefits of a Common Vocabulary and Knowledge Base

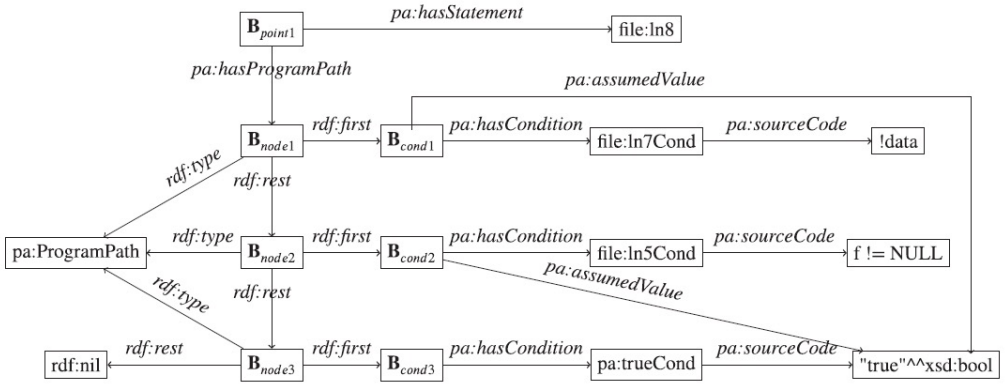
- Synergy between programming models:
  - Share/reuse/combine
    - domain-specific semantics
    - program analysis information
    - optimization rules
    - machine descriptions
  - Connect domain knowledge to programming models
  - Query semantics using standard query languages
- Some relevant studies
  - Zhao et al. PATO: Towards Ontology-Based Program Analysis, The European Conference on Object-Oriented Programming, 2016
  - Atzeni et al. CodeOntology: Querying Source Code in a Semantic Framework. In International Semantic Web Conference, 2017
  - Pattipati et al. OPAL: An Extensible Framework for Ontology-based Program Analysis, Software Practice and Experience 2020

C program ontology for building mergeable, declarative analysis: PATO'16



Connecting parameters to semantics: CodeOntology'17

```
SELECT ? method
WHERE {
? method a : Method ;
: hasParameter /: hasType : Double ;
: associatedWith : Cube_root .
}
```



Path-sensitive analysis using semantics in a knowledge graph: OPAL'20

Why Now?

Maturing techniques and tool chains

- Web Ontology Language
- Resource Description Framework
- Large-scale knowledge graph databases

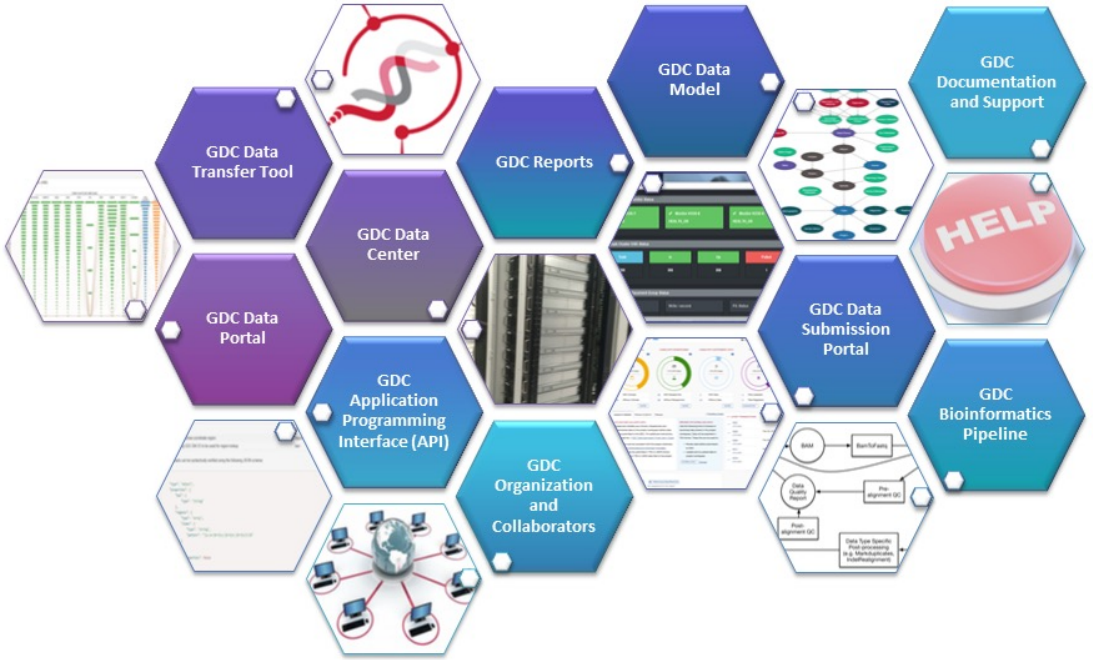
Notable Vocabularies/Ontologies

- Schema.org, Wikidata, Yago 4, Linked Open Vocabularies (LOV)

Promising prior research in ontologies + DSLs/compilers

Notable Applications

- National Cancer Institute: Genomic Data Commons (GDC)



https://gdc.cancer.gov/

Linking knowledge from two domains:

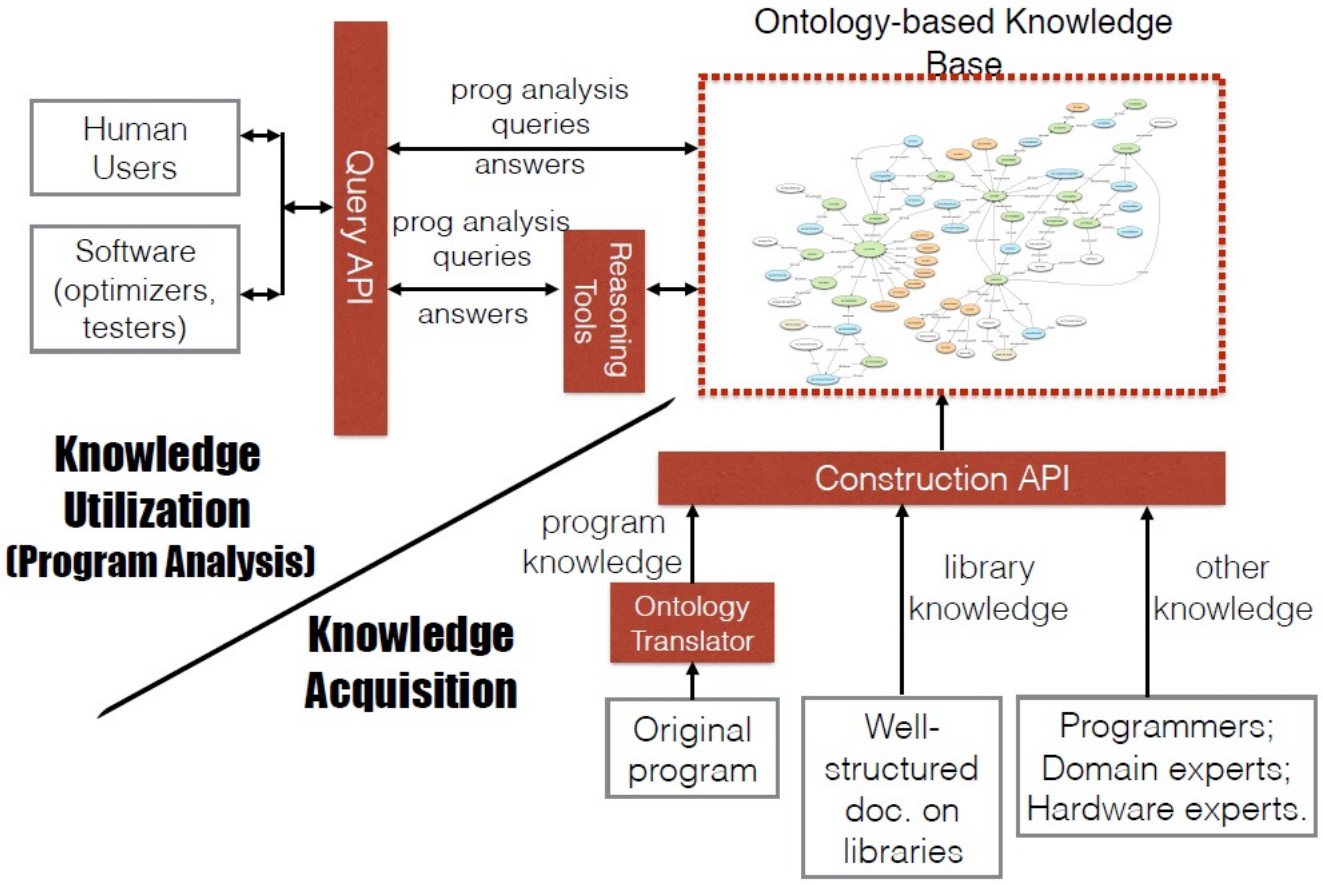
Clinical + Genomic data → Precision Cancer Treatments

1

Summary: CoDesign of Programming Models, powered by Common Vocabularies and Ontologies

HPC Programming models have a huge semantics gap

- Need: Heavily rely on sufficient knowledge of all layers of Software/hardware Stack for performance optimizations
- Situation: Mostly implicit or ad-hoc management of semantics/knowledge
- Solution: Formally manage knowledge across communities
  - Many existing knowledge management techniques available
  - Knowledge engineering: a community effort
  - The right time to address it is Now.



From PATO'16

Disclaimer

CASC

Center for Applied
Scientific Computing

Sponsored by DOE Office of Science and LLNL.

Knowledge: hard to recover, critical to performance

- High-level abstractions in Source Codes written in C++
  - Encouraging code reuse and hiding implementation from interfaces to reduce software complexity
  - Functions, data structures, classes and templates …
  - Could be standard or user-defined/domain-specific
- Semantics of high-level abstractions
  - Critical to optimizations, including parallelization
    - Read-only, data dependence, aliasing, accessed-by, etc.
    - Enabling a wide range of optimizations aimed for improving parallelism and reducing data movement.
  - Hard to be extracted by static/dynamic analysis
    - STL vector implementation ?→? elements are contiguous in memory
- Conventional compilers lose track of abstractions
  - Analyses and optimizations are mostly done on top of middle or low-level IR
  - Hard to trace back to the high-level abstractions represented in source level