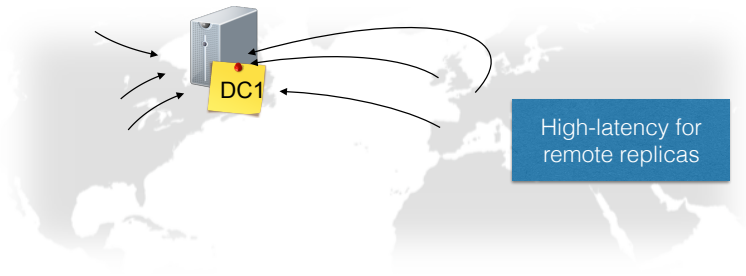
Consistency made easy: building correct-by-design cloud applications

Carla Ferreira
Universidade NOVA de Lisboa

RainbowFS Workshop on Consistency in Distributed Storage Systems
3 May 2017, Paris

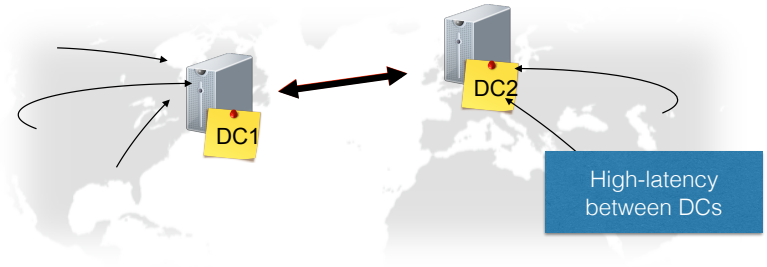
Context

- Internet services are at the core of modern business and organizations.
- Costumers demand quality of service



Geo-replication

- Deploy multiple physical replicas of the service.
- Clients interact with the closest replica.
- Coordinate executions to ensure consistency



Challenges in Data Availability

- Difficult to ensure data invariants and availability at the same time.
- Strong consistency: coordinate execution across replicas.
  - High-Latency, low availability.

Challenges in Data Availability

- Difficult to ensure data invariants and availability at the same time.
- Strong consistency: coordinate execution across replicas.
  - High-Latency, low availability.

- To ensure availability, cloud applications tend to trade strong consistency for weaker models
- These weaker models do not ensure data invariants
  - Solution: Strengthen consistency selectively

Challenges in Data Availability

- Databases with multiple consistency levels
  - Research: Explicit consistency, RedBlue consistency, Pileus
  - Commercial: Amazon DynamoDB, Basho Riak, Microsoft DocumentDB
- Hard to figure out the **minimum consistency** necessary to **maintain global invariants**

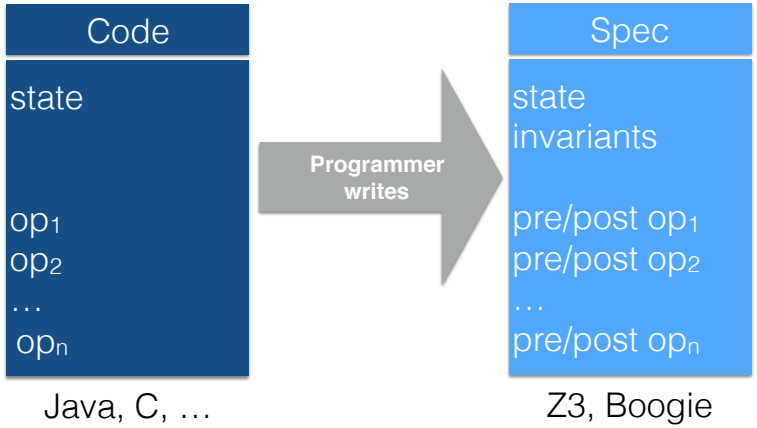
CISE model

- Generic model that expresses most existing consistency models
  - Each operation acquires a set of tokens
  - Conflict relation over tokens
  - Operations with conflicting tokens cannot run concurrently

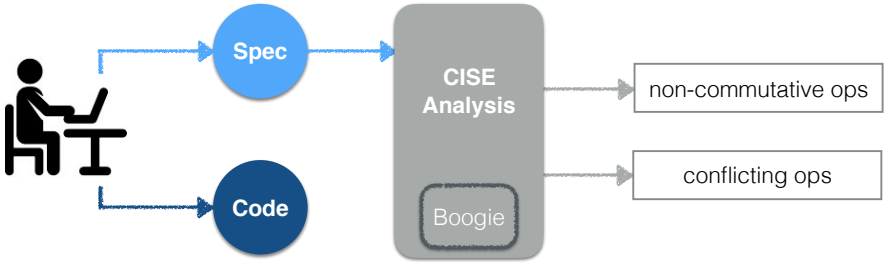
CISE model

- Generic model that expresses most existing consistency models
  - Each operation acquires a set of tokens
  - Conflict relation over tokens
  - Operations with conflicting tokens cannot run concurrently
- First Proof Rule to check correctness of weakly consistent applications (Gotsman et al, POPL'16)
  - Assumes causality and commutativity of non-conflicting operations
  - Polynomial time analysis

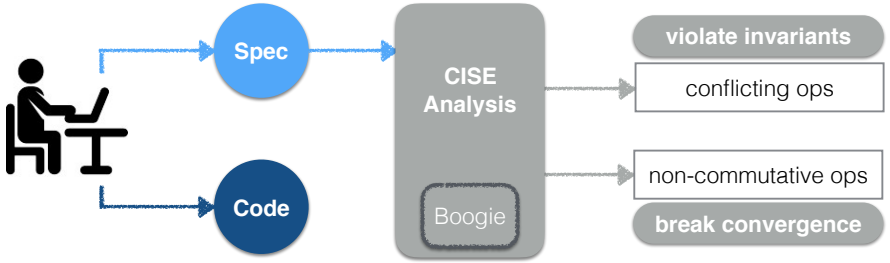
Current CISE tools



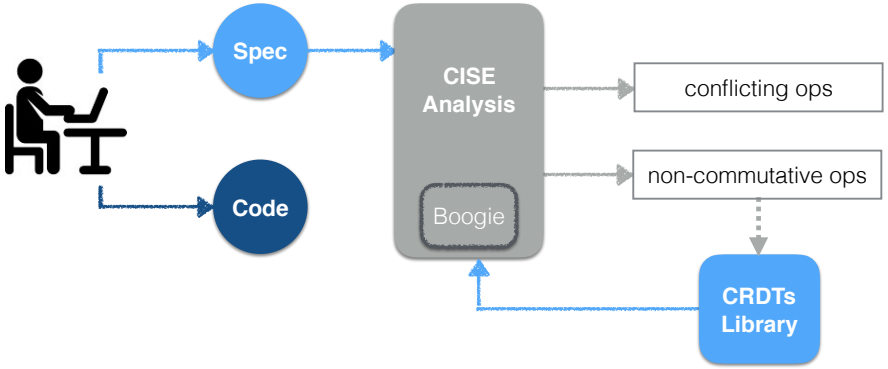
Current methodology



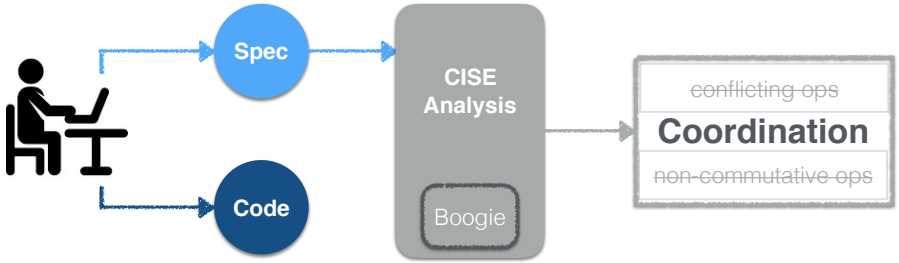
Current methodology



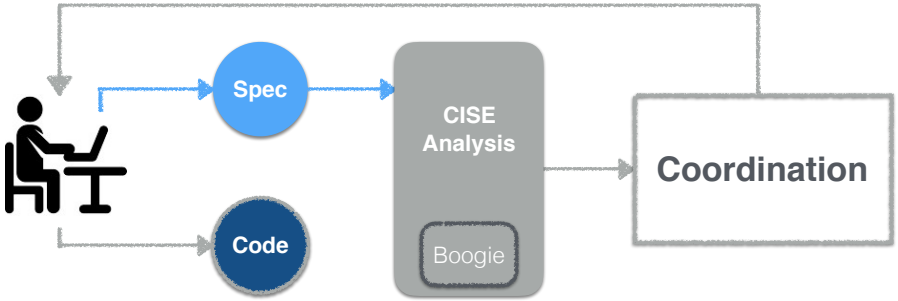
Current methodology



Current methodology



Current methodology



Not yet usable by programmers

Verification tools in industry

- Full automation and integration
- Scalability
- Precision
- Fast reporting
  - Calcagno et al, NFM'15, POPL'09
    INFER: static analysis tool integrated in Facebook's software development process

What is missing for full automation?

- Spec synthesis
  - Invariant deduction
  - Pre/post conditions synthesis
  - Extensive research on the subject

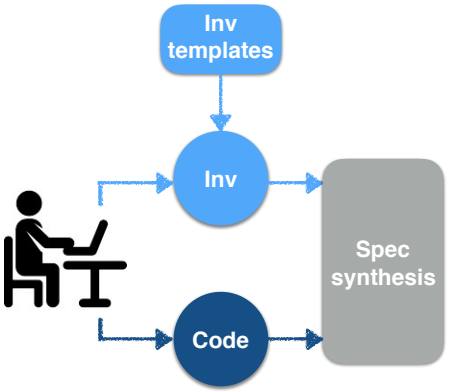
Writing invariants is hard

- Static analysis techniques able to deduce invariants
  - Shape analysis deduce data structure invariants
  - Use of “good” applications traces to extract invariants
- In the end some invariants will have to be written down by the programmer

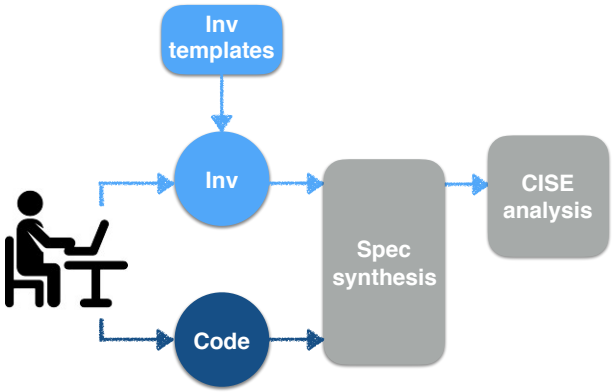
Invariants templates

- Well-known invariants:
  - Relational database integrity constraints
  - Lower and upper bounds to data values
  - Consistency in 3D proposes other classes of invariants (Marc’s morning talk)

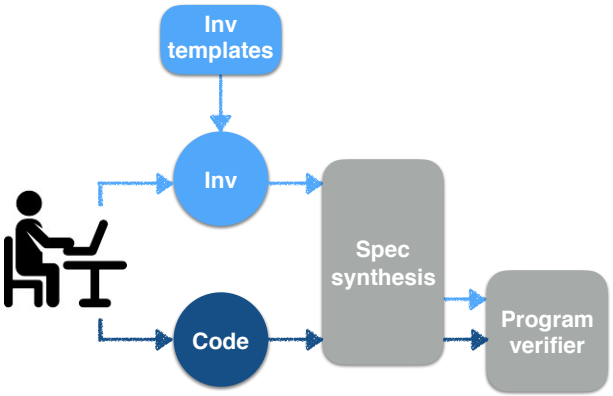
Correct-by-design methodology



Correct-by-design methodology

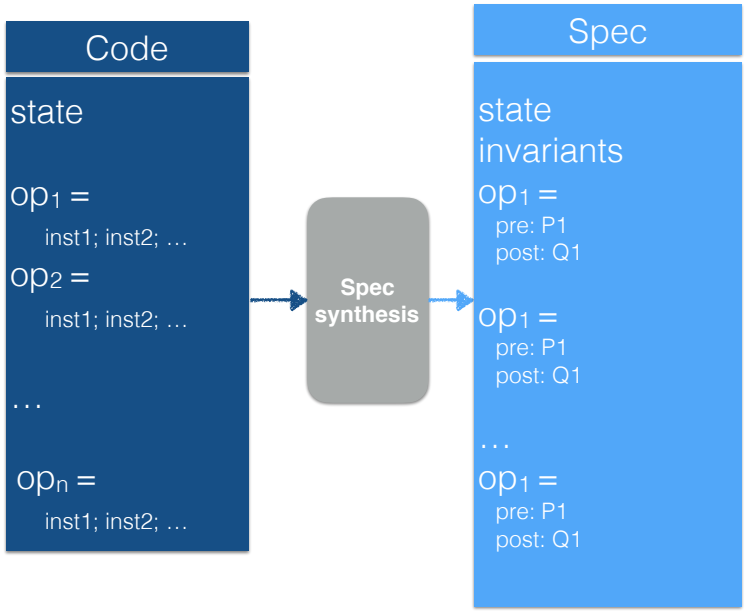


Correct-by-design methodology

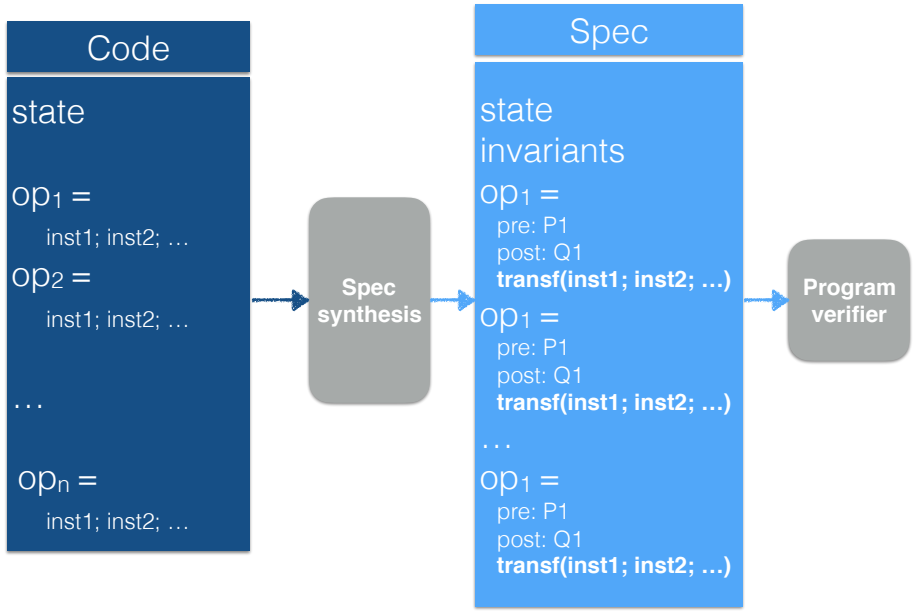


- Sequential reasoning
- no explicit concurrent access to data

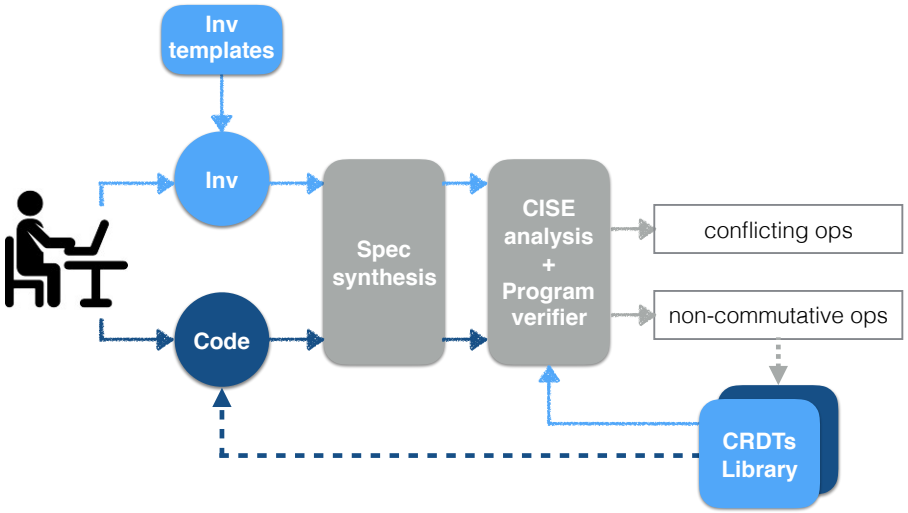
Verification frameworks provide mainstream language APIs



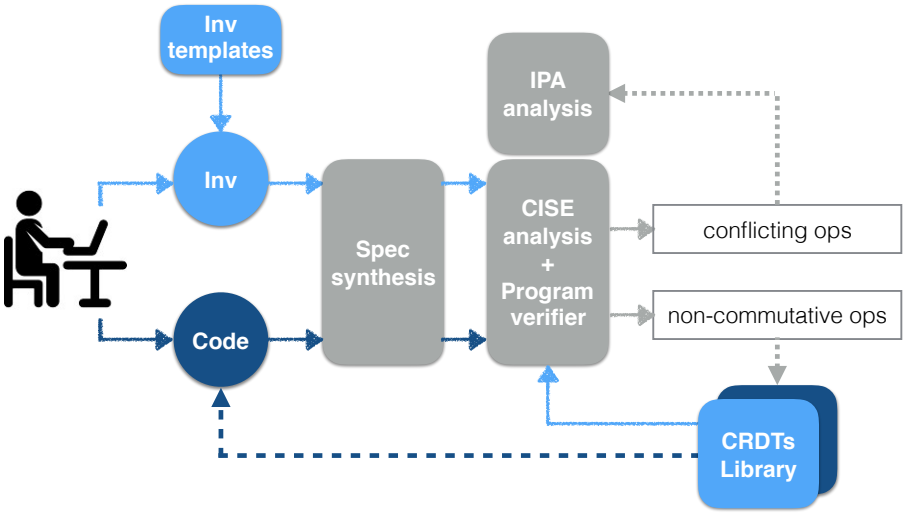
Verification frameworks provide mainstream language APIs



Correct-by-design methodology



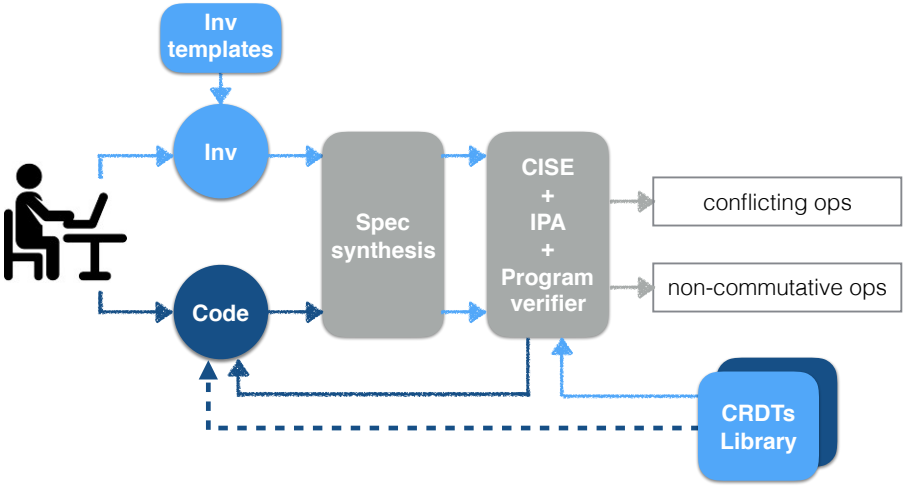
Correct-by-design methodology



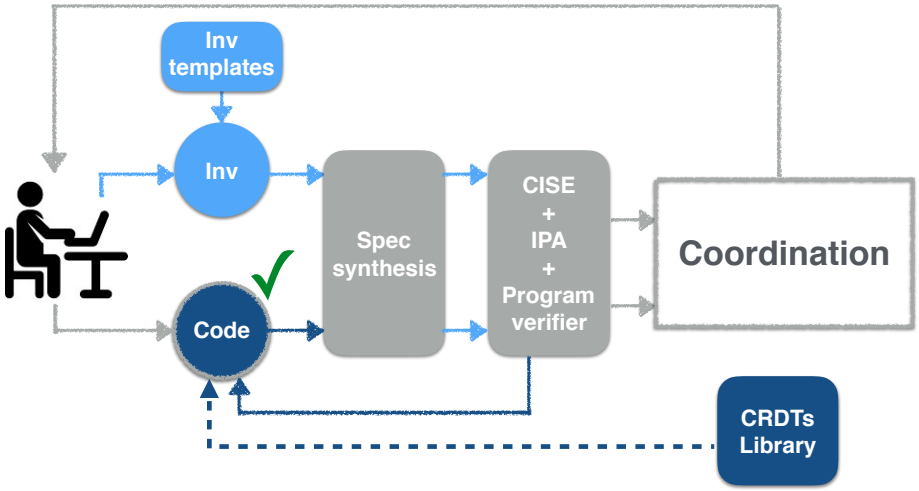
IPA Analysis

- Transforms the application operations so that they become invariant-preserving by design.
- Some invariants can be preserved by modifying the effects of operations.
- Proposes an algorithm that is capable of generating those modifications.
- Maintain the observable effects of each operation.

Correct-by-design methodology



Proposed methodology



Conclusions

- Developing applications for weak consistency is hard.
- CISE analysis allows the programmer to develop applications assuming initially a sequential setting.
  - Analysis detects what are the problematic operations if executed on geo-replicated setting.
  - Assumes casual order and commutativity for non-coordinated operations.
- Need to define a methodology and simple tools to help programmers building correct cloud applications.