Simulating Cities: The Spacetime Framework

Crista Lopes, Arthur Valadares, Rohan Achar

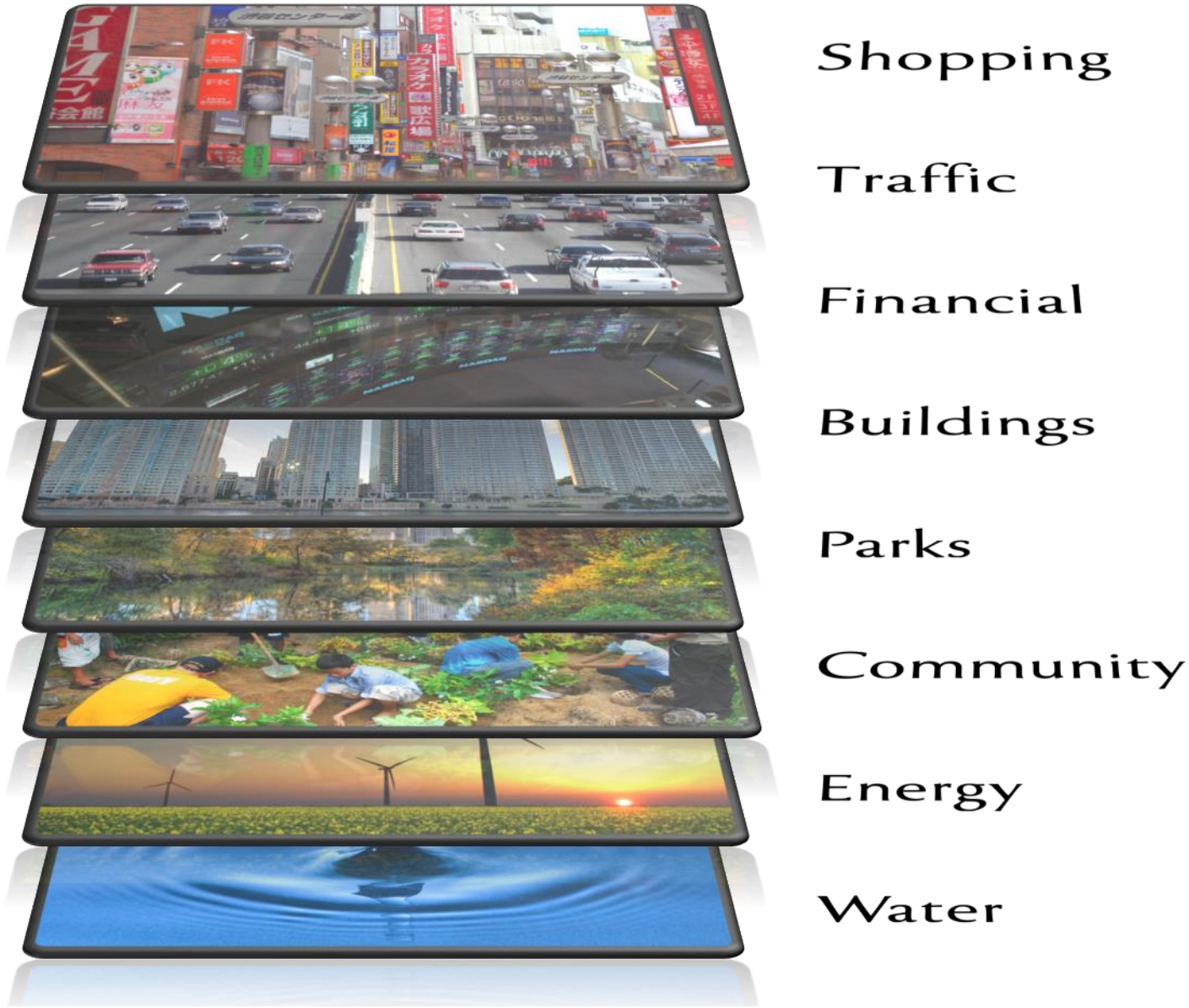
UC Irvine

In collaboration with

4Dialog

The Domain and Its Problems

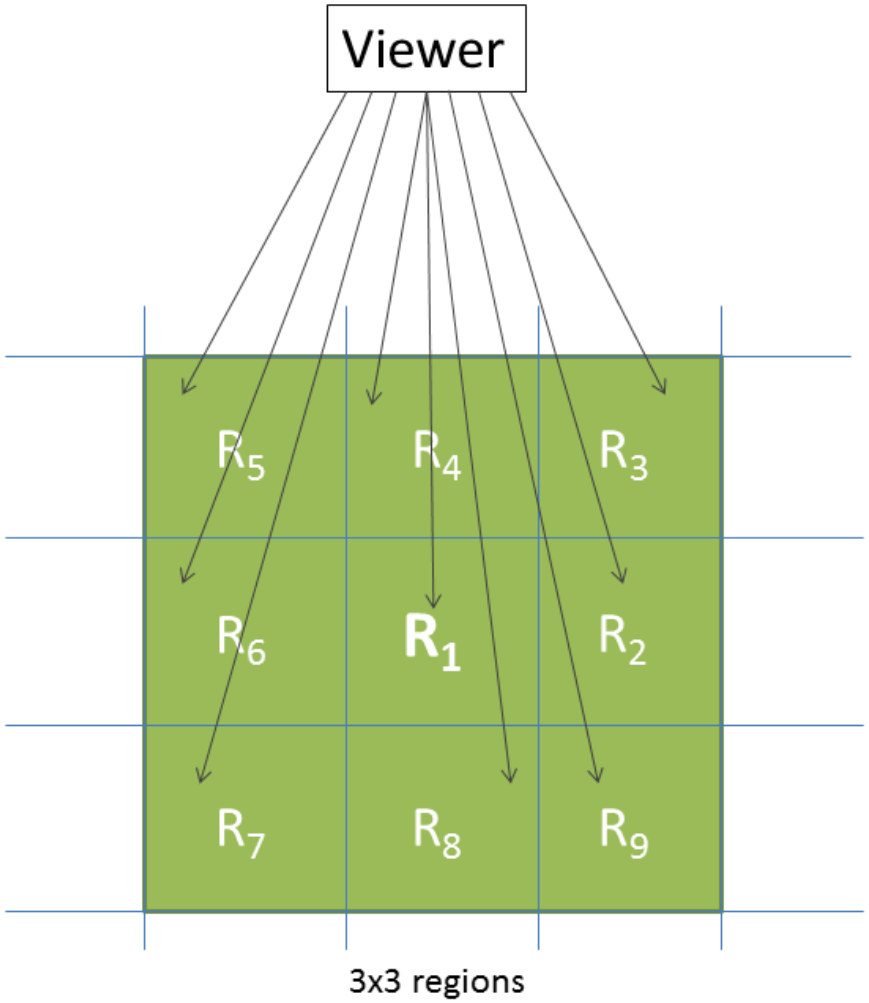
Complex time stepped simulations



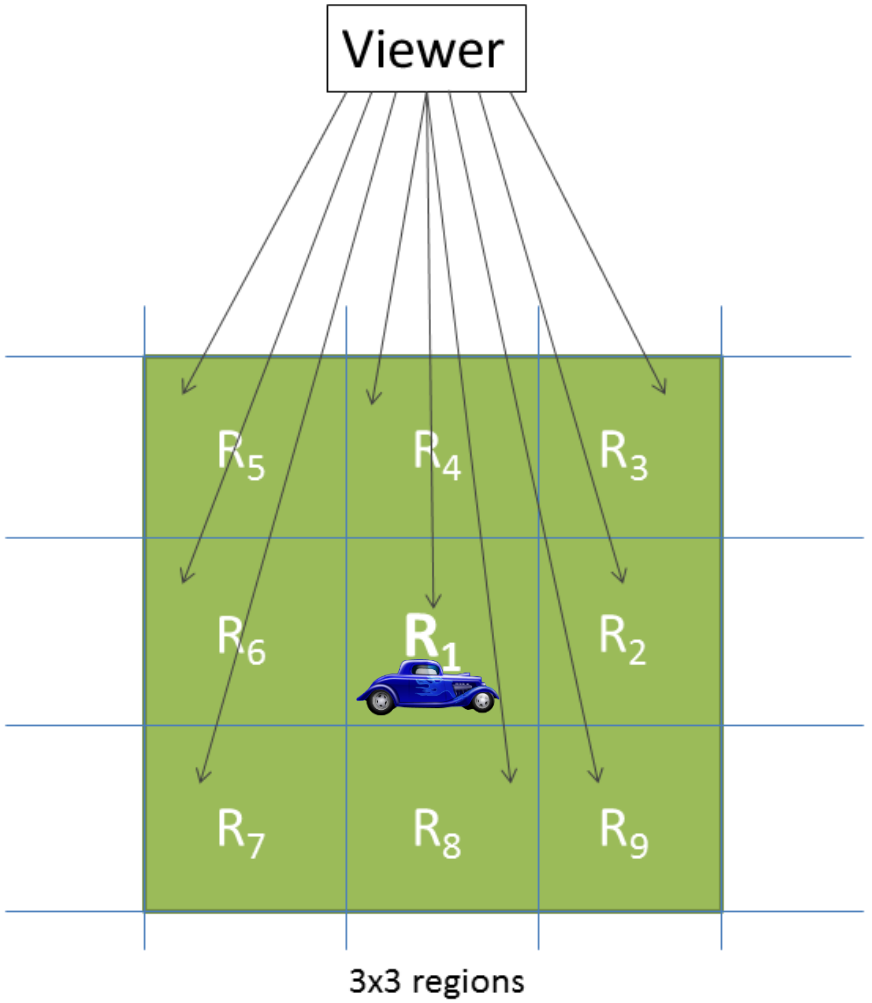
Fujimoto, R. M. Parallel and Distributed Simulation. In Proceedings of the 2014 Winter Simulation Conference (Piscataway, NJ, USA, 2014), WSC '14, IEEE Press, p. 5.

Encitra/4Dialog Urban Simulations

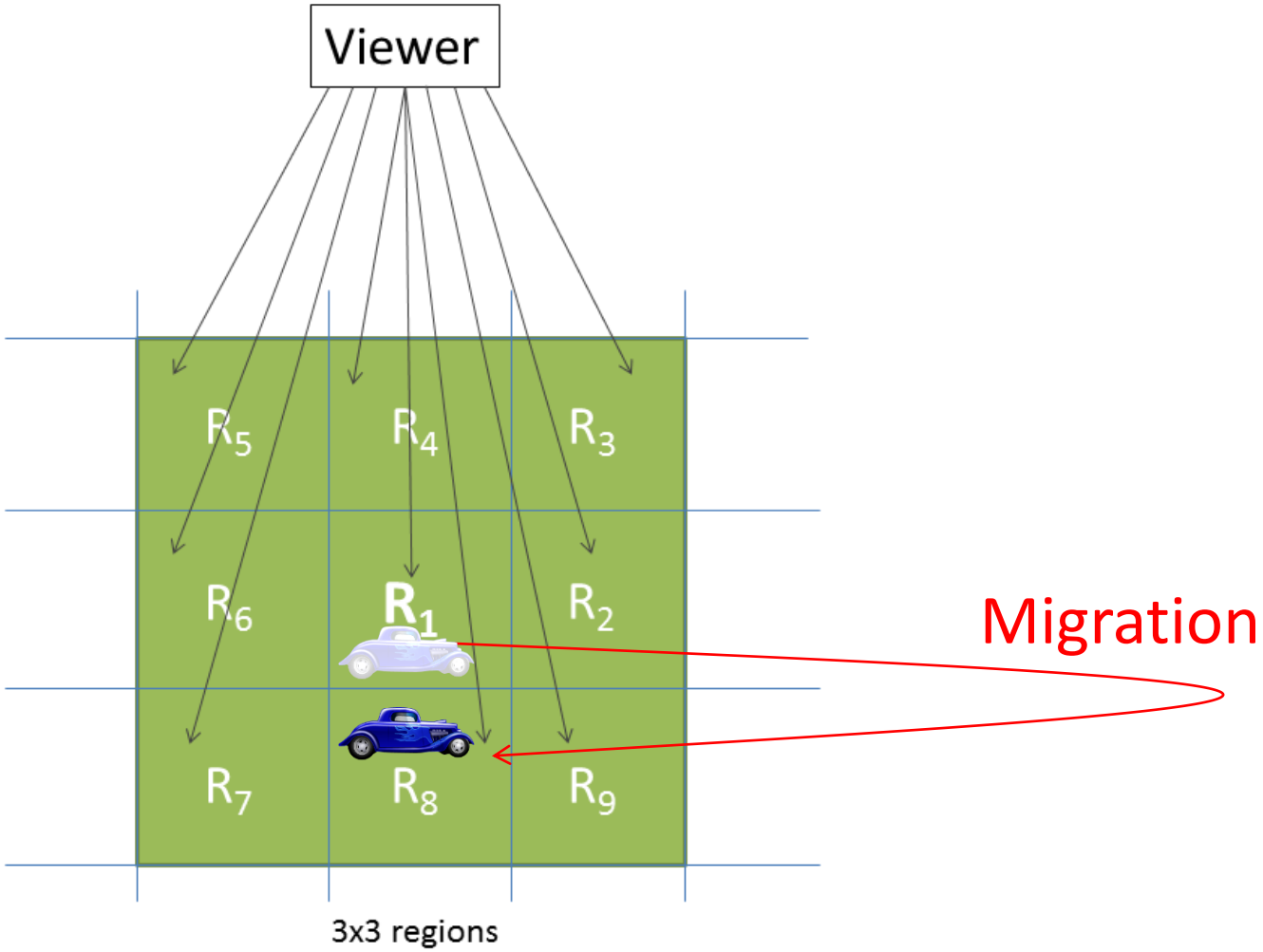
Distributed Simulation with Space Partitions



Distributed Simulation with Space Partitions



Distributed Simulation with Space Partitions



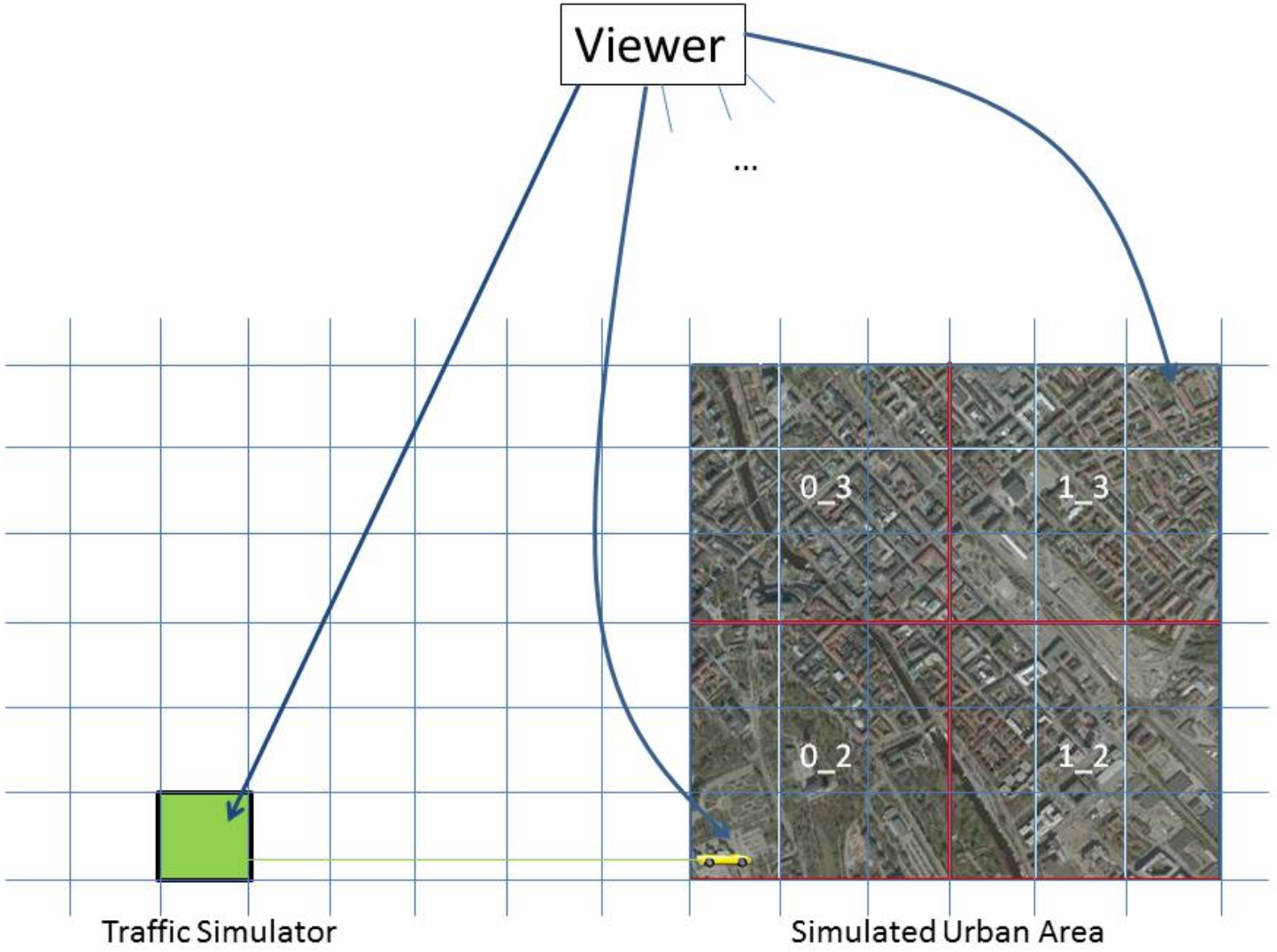
Problems

- Migrations across borders (visual quirks)
- Occasional overload while modelers are modeling
- **No separation of concerns**
  - Work disruption (tear down / boot up traffic sim)

A Different Approach: “Aspect” Partitions

- Separate traffic simulator from world simulator(s)
  - Traffic sim does traffic
  - World sims do the “map”
  - …
- Avoids all the previous problems

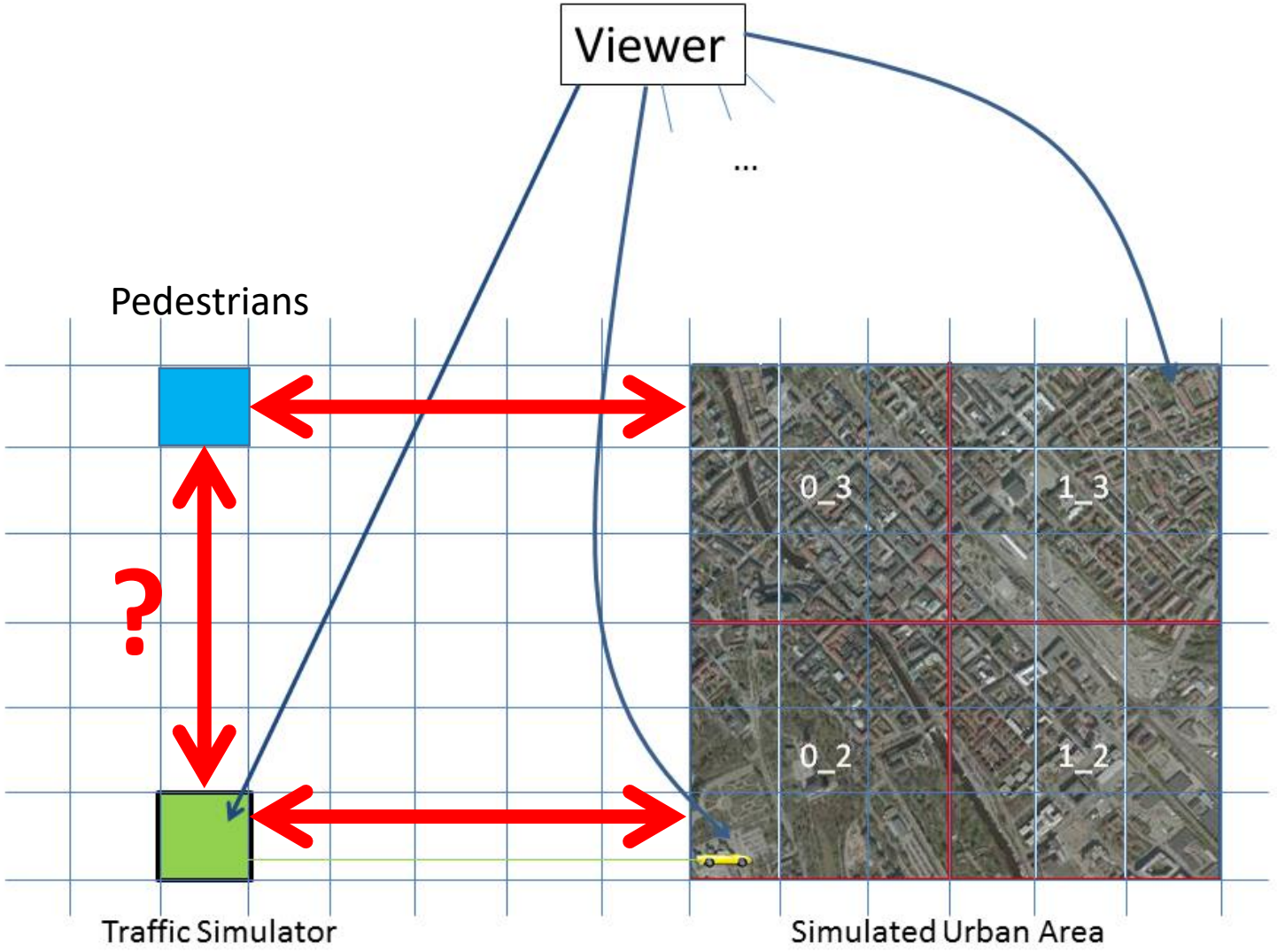
Aspect Partitions



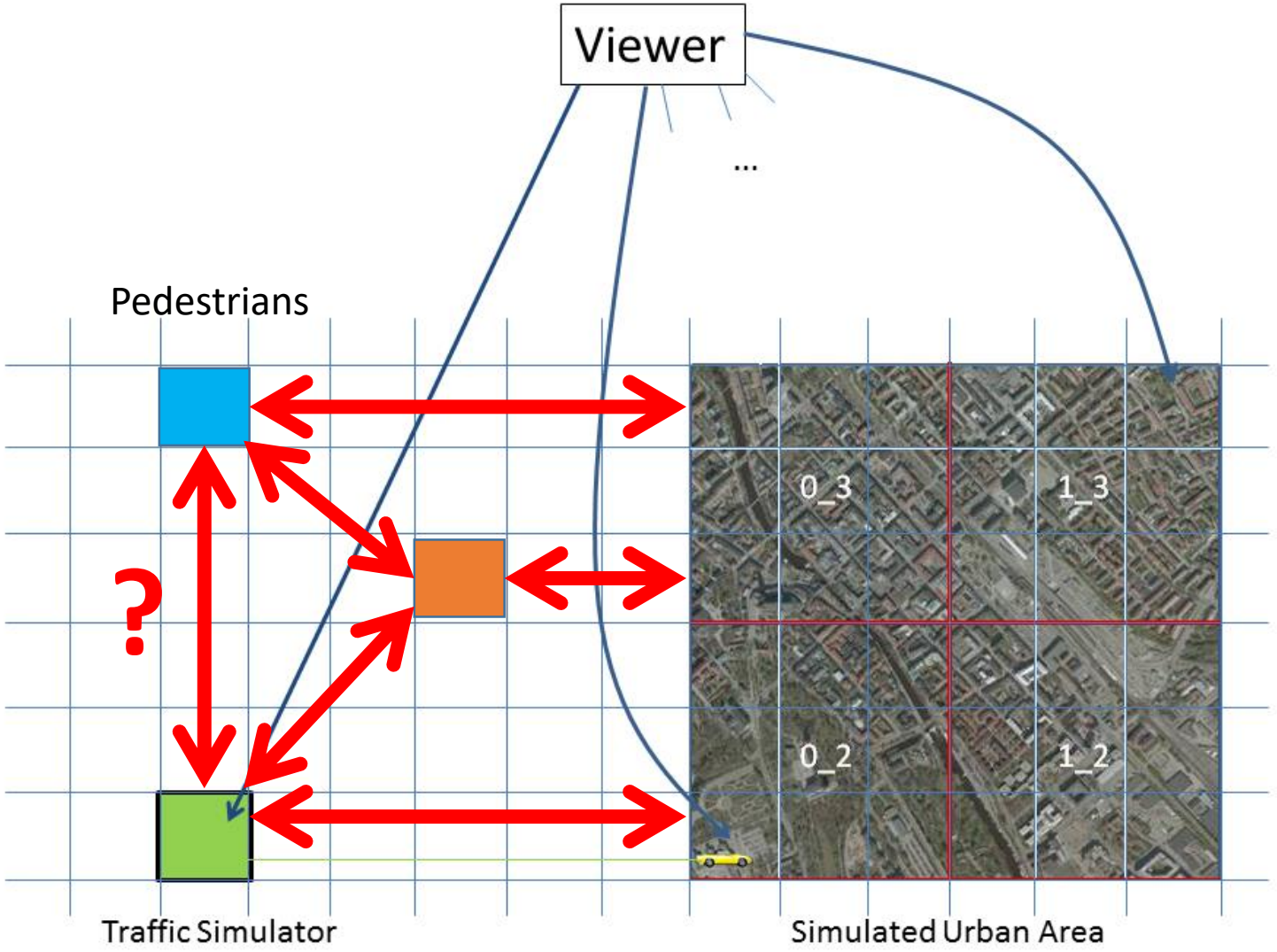
Aspect Partitions: New Problems

- Data sharing and APIs

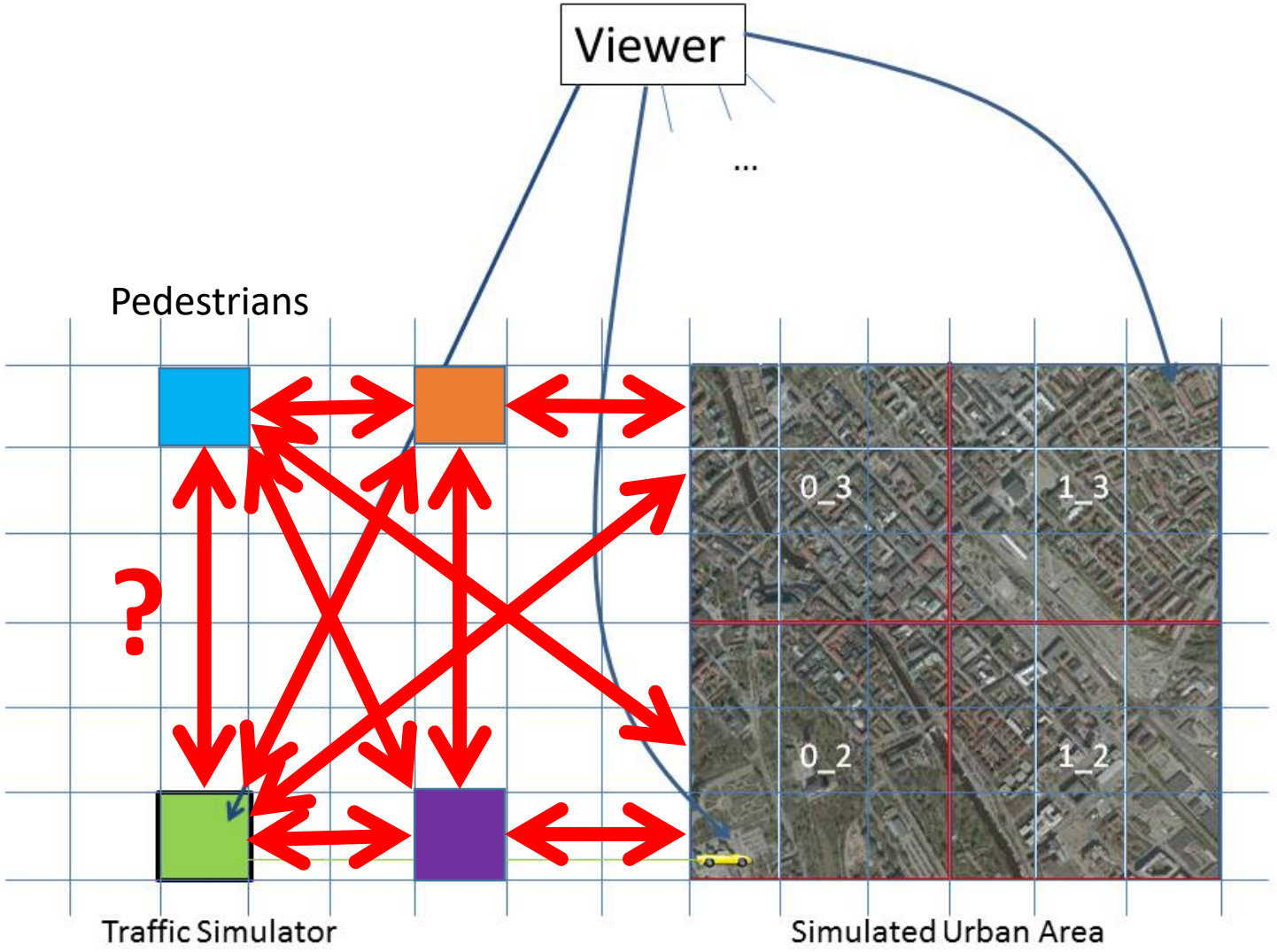
Aspect Partitions



Aspect Partitions



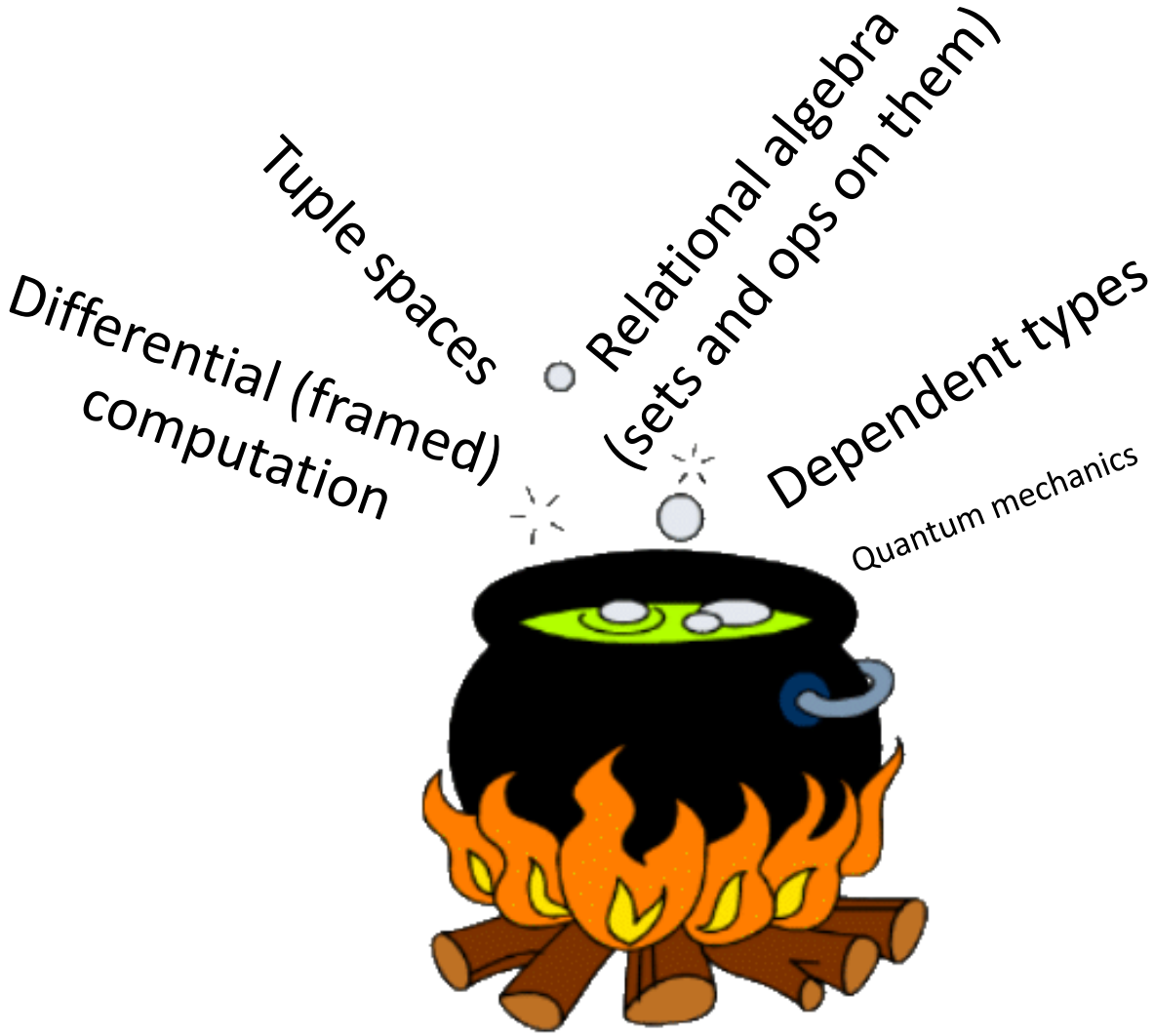
Aspect Partitions



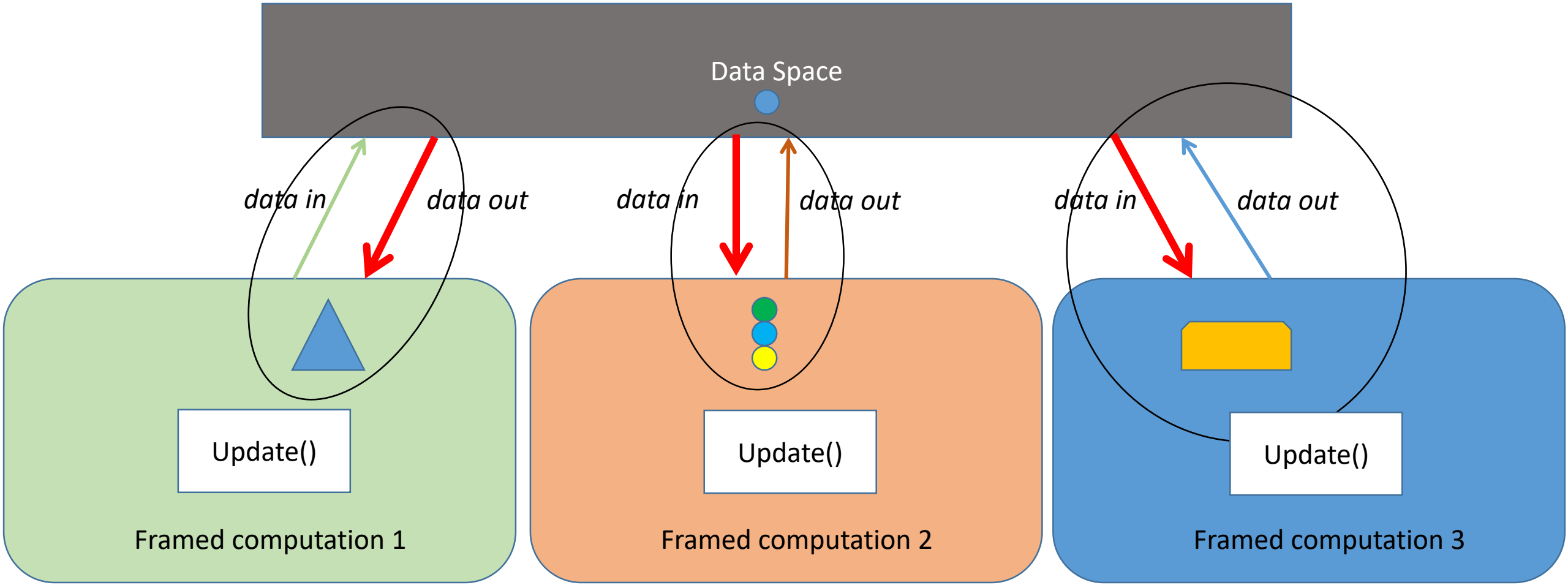
Approach

Data-Oriented Architecture

Ingredients (constraints) of the approach



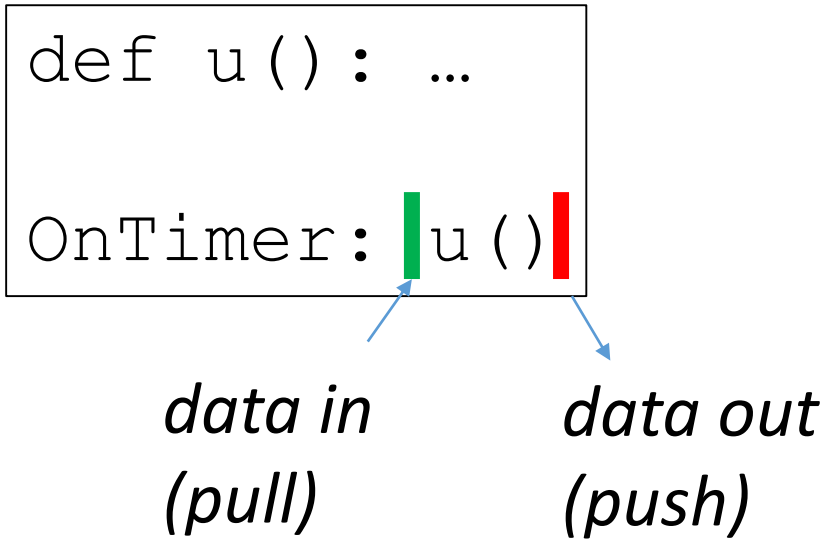
Spacetime Framework



Two Parts (DSLs)

- Framed processing components
  - Declare what objects they need, and how
- Interface with the data space
  - Object frames

Framed Computations



Constraints on u:

- Should not spawn threads
- Must not access shared store explicitly

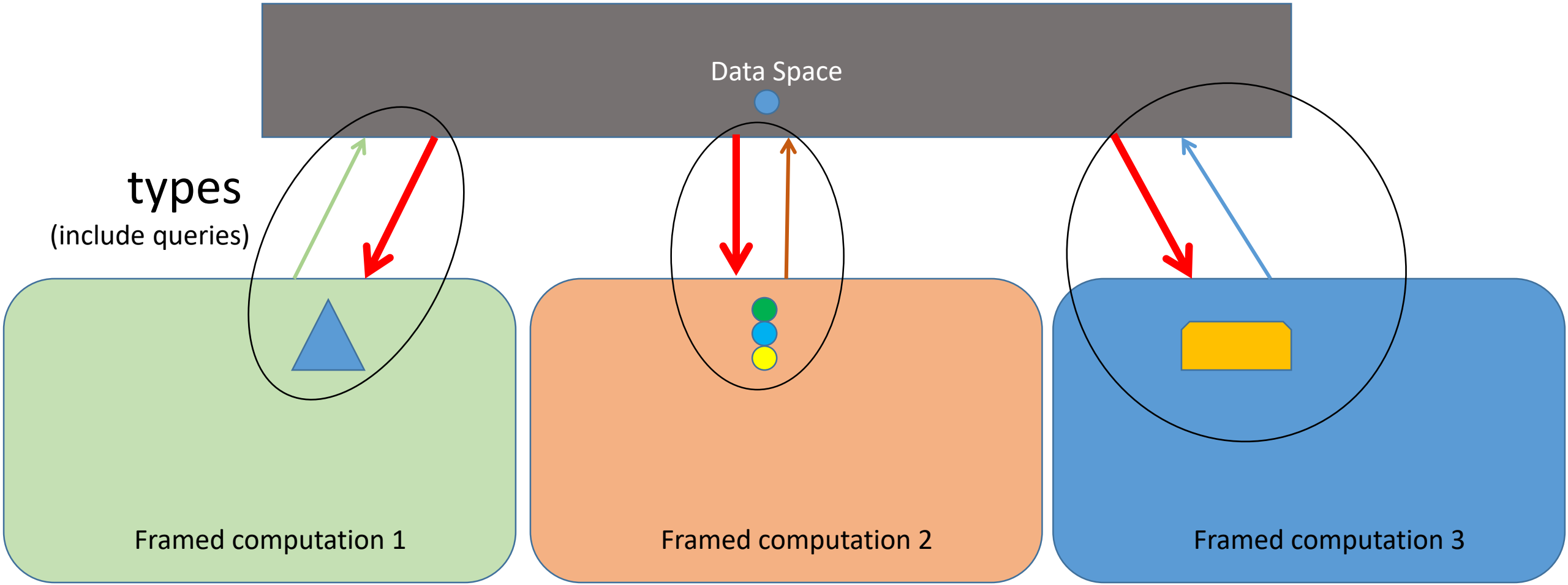
Car Simulation

```
1  @Producer(Car, host = 'http://127.0.0.1:12000')
2  @GetterSetter(InactiveCar, ActiveCar)
3  class TrafficSimulation(IApplication):
4      TICKS_BETWEEN_CARS = 10
5      def __init__(self, frame):
6          self.frame = frame
7
8      def initialize(self):
9          for i in xrange(2):
10             self.frame.add(Car())
11         self.ticks = 0
12         self.cars = self.frame.get(Car)
13
14     def update(self):
15         if self.ticks % self.TICKS_BETWEEN_CARS == 0:
16             inactives = self.frame.get(InactiveCar)
17             if inactives != None and len(inactives) > 0:
18                 inactives[0].start();
19         for car in self.frame.get(ActiveCar):
20             car.move()
21         self.ticks += 1
```

Pedestrian Simulation

```
23  @Producer(Pedestrian)
24
25  @GetterSetter(StoppedPedestrian, Walker, CarAndPedestrianNearby)
26  class PedestrianSimulation(IApplication):
27      TICKS_BETWEEN_PEDESTRIANS = 10
28      def __init__(self, frame):
29          self.frame = frame
30
31      def initialize(self):
32          for i in xrange(5):
33              self.frame.add(Pedestrian())
34          self.pedestrians = self.frame.get(Pedestrian)
35          self.ticks = 0
36
37      def update(self):
38          if self.ticks % self.TICKS_BETWEEN_PEDESTRIANS == 0:
39              inactives = self.frame.get(StoppedPedestrian)
40              if inactives != None and len(inactives) > 0:
41                  inactives[0].move();
42
43          endangereds = self.frame.get(CarAndPedestrianNearby)
44          for car_ped in endangereds:
45              car_ped.move()
46          for pedestrian in self.frame.get(Walker):
47              pedestrian.move()
48          self.ticks += 1
```

Guarantees at the Border



Interface with data space: Predicate Collection Classes (PCCs)

- Inspired by dependent types:
- A type that is predicated upon a (dynamic) value

  or

- Static guarantees about dynamic properties of the object

Subset

```
1   class Car:
2     @dimension(int)
3     def ID(self): return self._ID
4
5     @dimension(list)
6     def Position(self): return self._Position
7
8     @dimension(list)
9     def Velocity(self): return self._Velocity
10
11    #...methods of Car...
12
13  @subset(Car)
14  class ActiveCar:
15    @staticmethod
16    def __predicate__(car): return not (car.Velocity == [0,0] or car.Velocity == None)
17
18    def Move(self):
19      self.Position[0] += self.Velocity[0]
20      self.Position[1] += self.Velocity[1]
21      if self.Position == (100, 100): self.Velocity = None
```

Projection

```
1   class Person:
2     @dimension(int)
3     def ID(self): return self._ID
4     @dimension(str)
5     def Name(self): return self._Name
6     # plus 20 other dimensions
7
8   @projection(Person, Person.ID, Person.Name)
9   class PersonInfo:
10    def PrintSummary(self):
11      print "ID=" + str(self.ID) + " Name=" + self.Name
12
13    @Name.setter
14    def Name(self, value): self._Name = value
15    # More fields and methods for PersonInfo
```

But... Imperative Language?

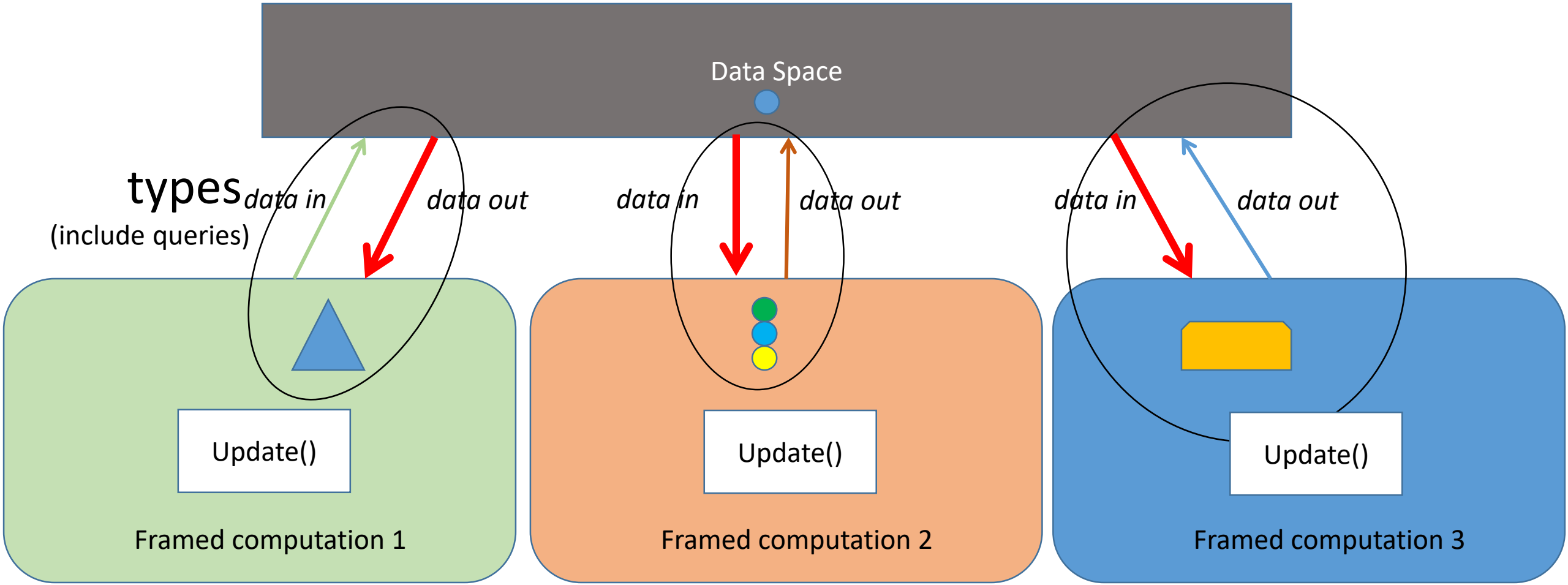
```
f(ActiveCar aCar)
{
    aCar.Velocity = Vector3.Zero;
}
```

Mutable state => local violations

PCCs: Full relational algebra

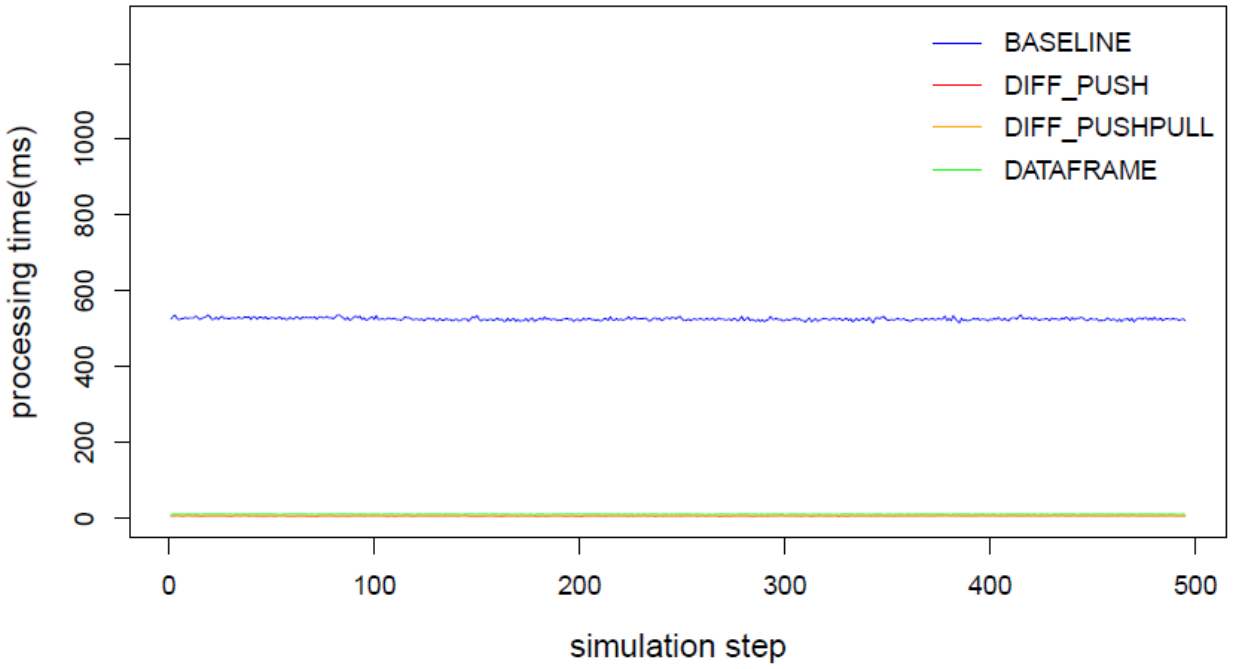
- Subsets
- Projections
- Cross Product (joins)
- Intersections
- Unions
- Foreign keys (object references)

Spacetime Framework

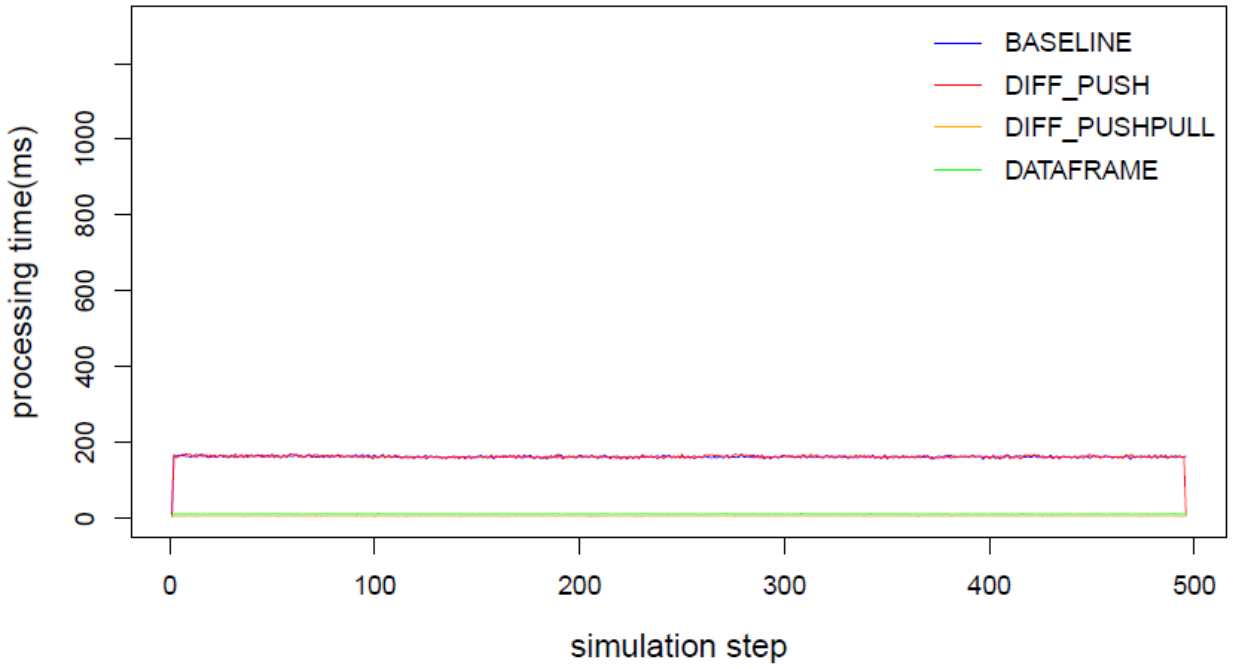


Performance

Set, no updates



(a) Producer



(b) Consumer

Figure 2. Total processing time (push, pull, and update) for producer and consumer in PCC base type scenario.

Set with updates

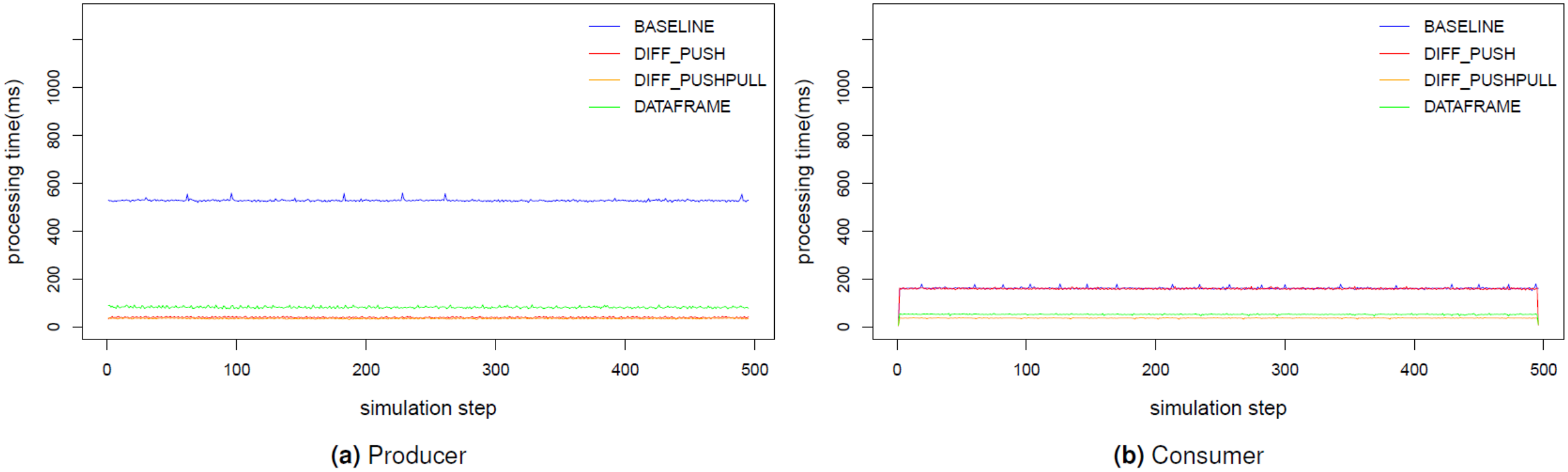
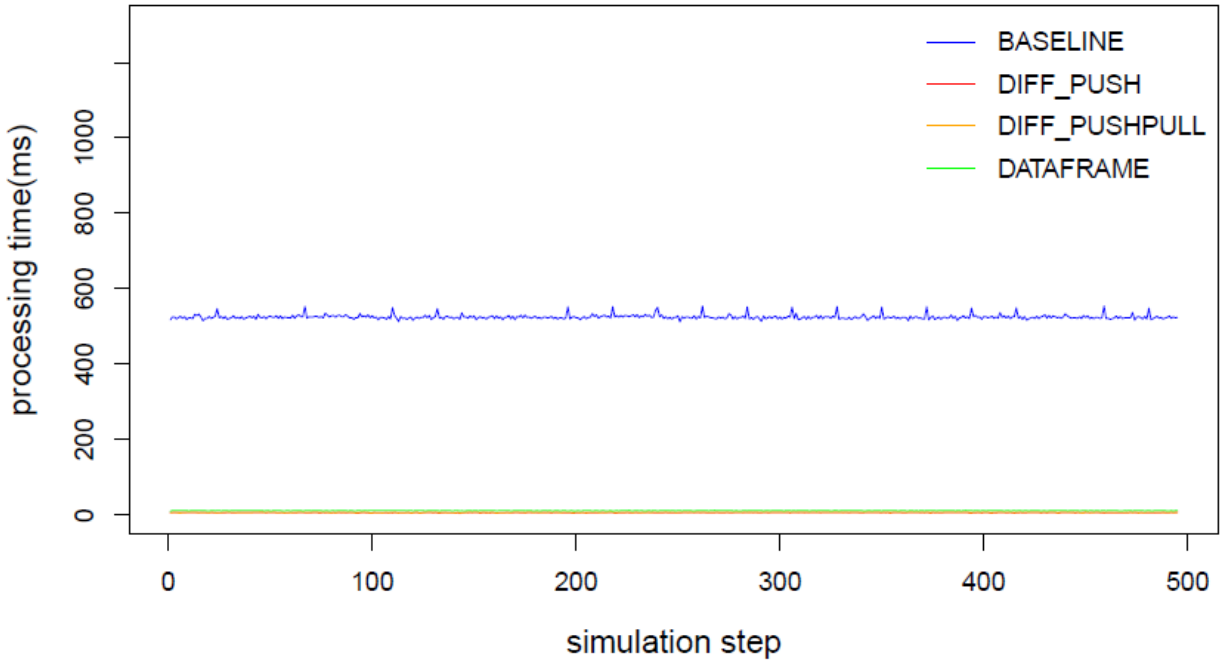
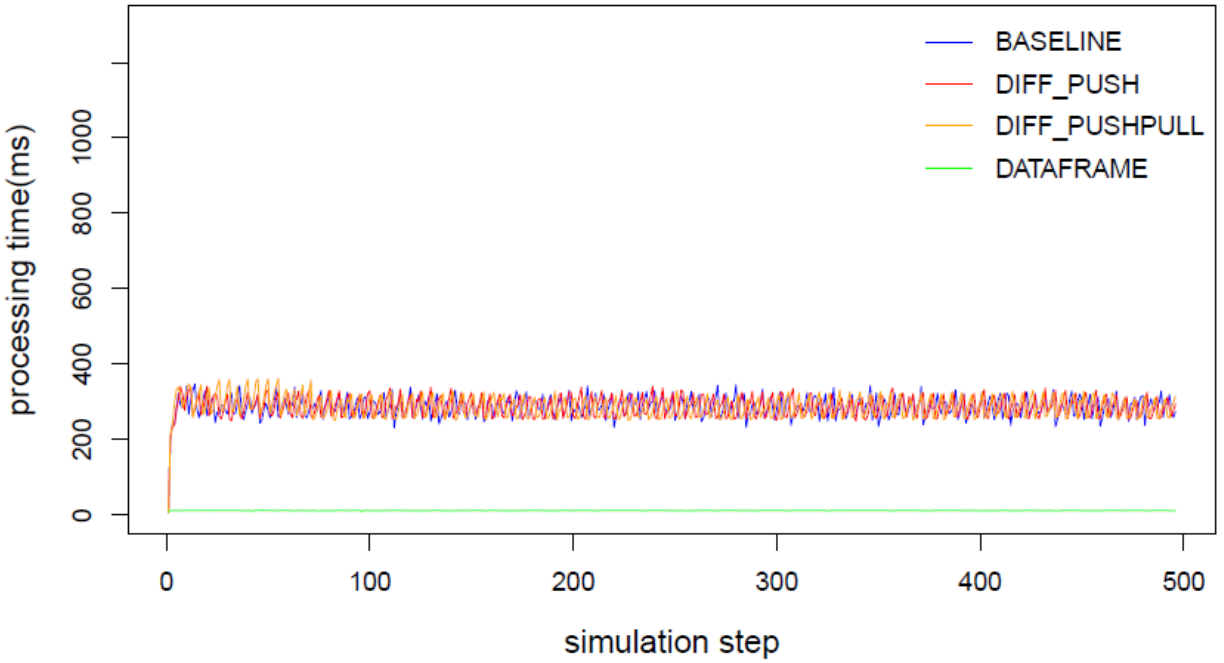


Figure 6. Total processing time (push, pull, and update) for producer and consumer in update scenario.

Projection, no updates



(a) Producer



(b) Consumer

Figure 4. Total processing time (push, pull, and update) for producer and consumer in projection scenario.

Join, no updates

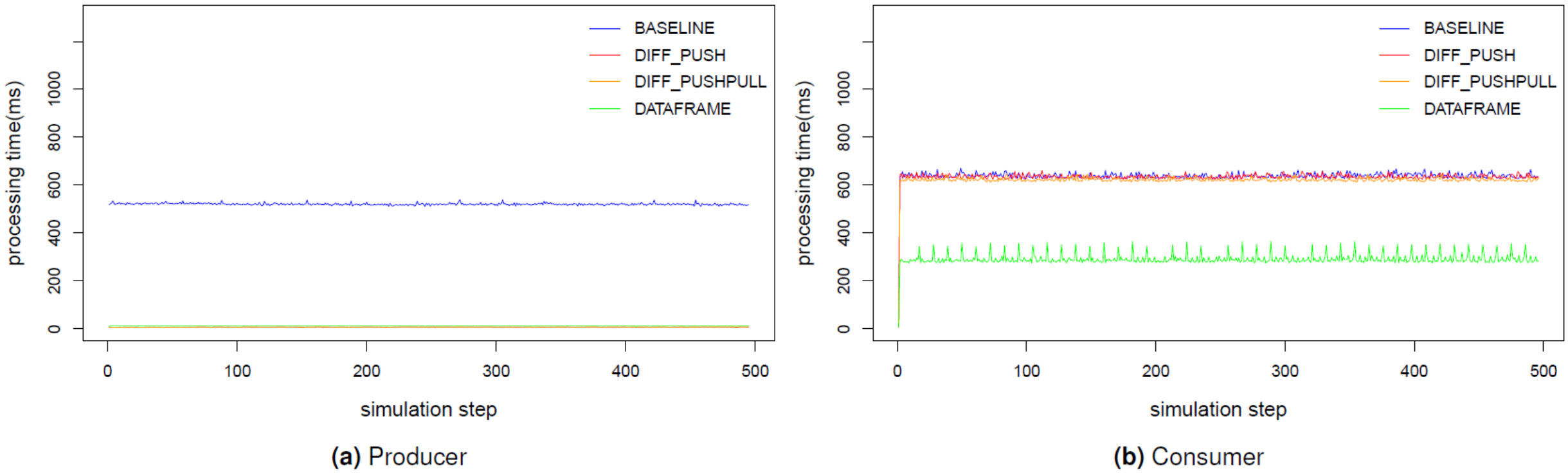
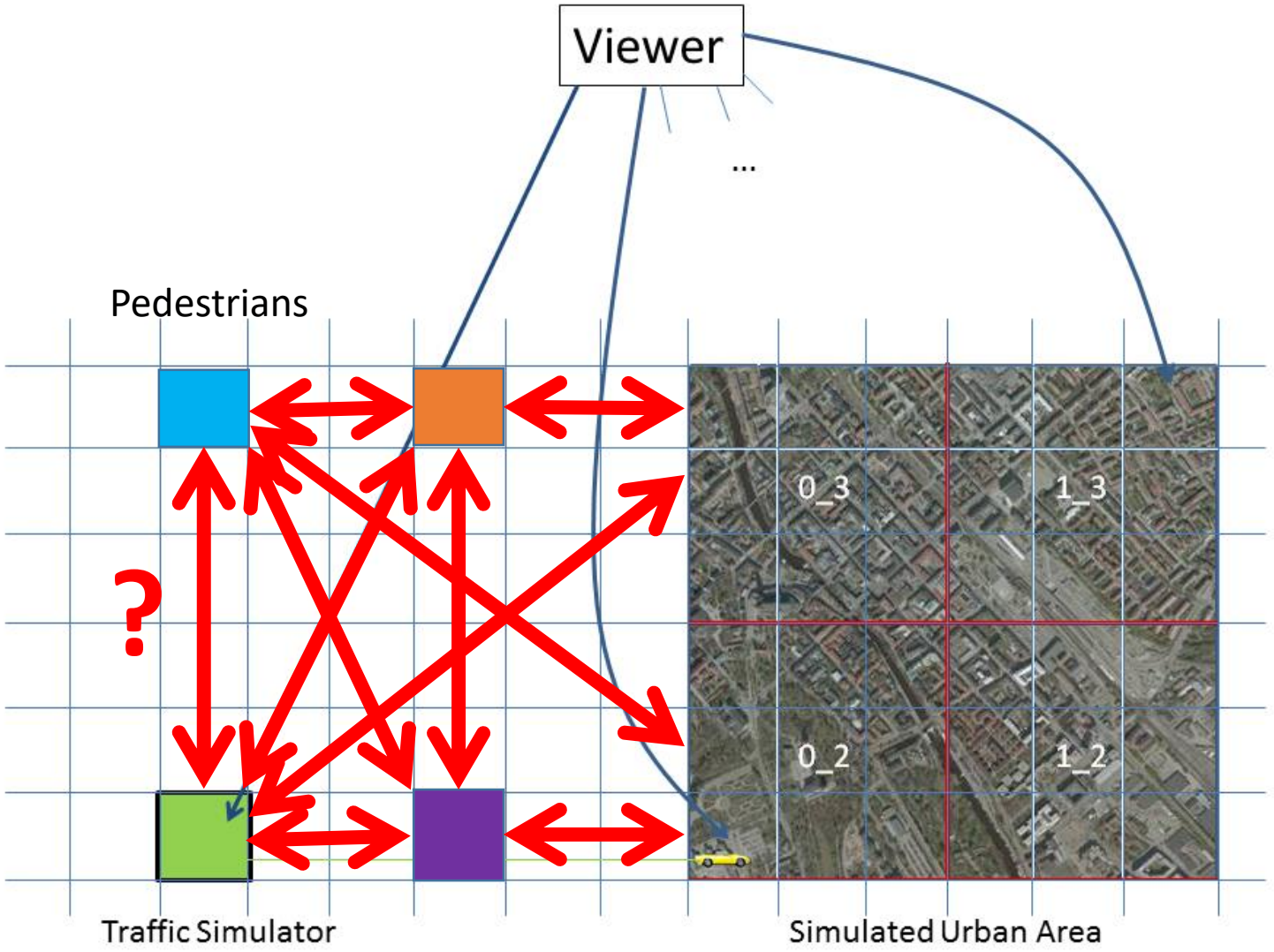


Figure 5. Total processing time (push, pull, and update) for producer and consumer in join scenario.

Experiences

Course on Distributed Simulations

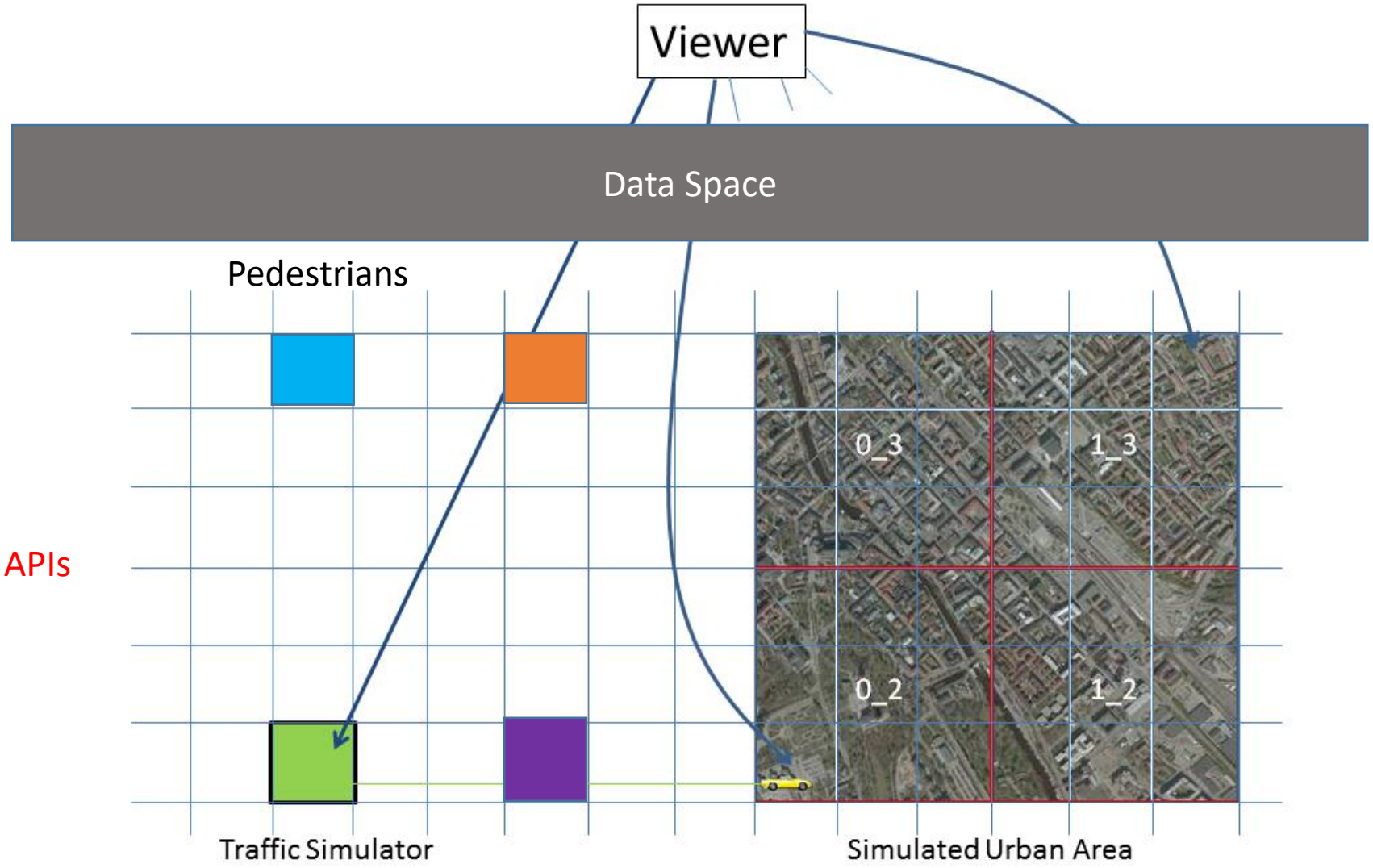
Deployment in 4Dialog



Unmanageable APIs

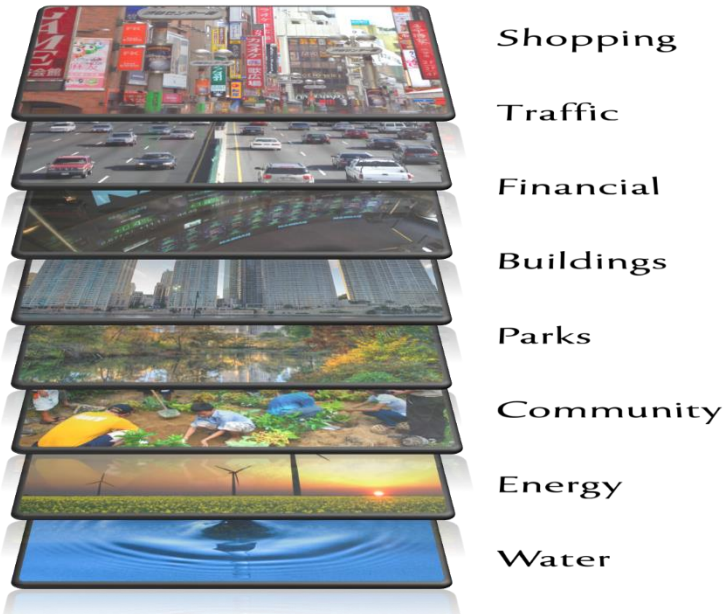
Deployment in 4Dialog

Data-driven APIs



Concluding Remarks

- Architectural separation of concerns: real need in this domain
  - War story: High Level Architecture (HLA)
- Amorphous data store + polymorphic data uses
- Identity (data records), Isolation (copy), and Entanglement (of copies)



https://github.com/Mondego/spacetime

https://github.com/Mondego/pcc

@cristalopes

Rohan Achar

@arthursv