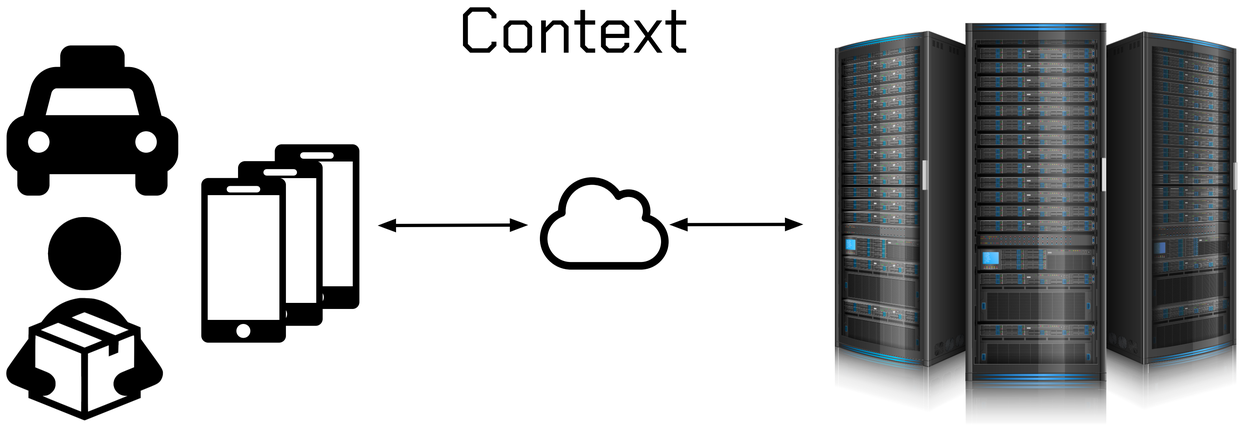
skitter.

A DSL for Distributed Reactive Workflows

Mathijs Saey Joeri De Koster Wolfgang De Meuter

Context



We are producing a lot of data, we need software that **reacts** to this data **instantaneously**

Scale of data forces us to execute on a cluster. Need to deal with **partial failure**, **replication**, **consistency**, …

Problem Statement

We want a programming language which allows one to write **scalable**, **reactive** big data applications from a set of **existing, reactive components**.

Related Work

	Reactive Programming	Stream Processing	Scientific Workflows
Reactive	✓	?	✗
Scalable	✗	✓	✓
Existing Components	?	✗	✓

1 Write Reactive Components

Interface between component and workflow

```
component Distance, in: [p1, p2], out: [distance] do
  react {x1, y1}, {x2, y2} do
    sqrt(square(x2 - x1) + square(y2 - y1)) ~> distance
  end
end
```

Called by runtime when data is present

Publish data to connected components

From scratch

```
component GeoFilter, in: [json], out: [inside, outside] do
  fields area

  init area_json do
    area <~ area_json
  end

  react json do
    loc = ... # extract location from json
    r = System.cmd "in_area", ["--area #{area}", loc]
    if r == "inside", do: json ~> inside, else: json ~> outside
  end
end
```

Track state with fields

Use an existing program

As a *wrapper* for foreign code

Effects

```
component Count, in: [any], out: [current] do
  effect state_change
  fields count

  init _ do
    count <~ 0
  end

  react _ do
    count <~ count + 1
    count ~> current
  end
end
```

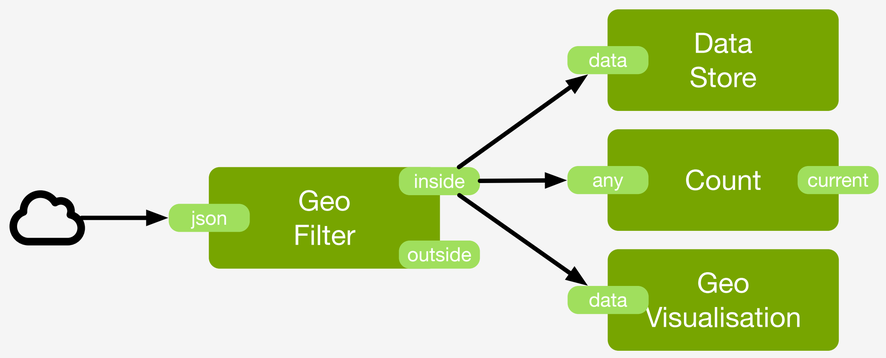
Specify that the component may change its state

<~ primitive can be used while reacting

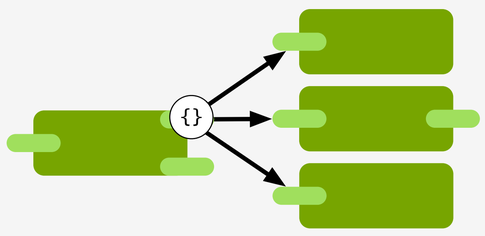
Component with mutable state

Property	Effect	Additional Primitives
Mutable state	`state_change`	`<~`
Foreign process with mutable state	`state_change hidden`	`<~` *create_checkpoint* *restore_checkpoint* *clean_checkpoint*
I/O may occur	`external_effect`	*after_failure*

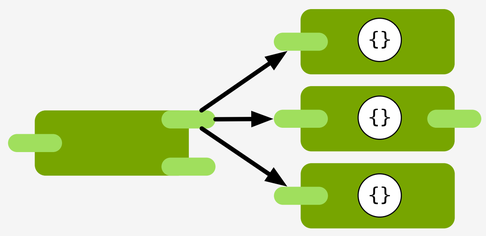
2 Compose Reactive Workflows



3 Execute on a Cluster



Skitter passes messages between connected components and activates `react` and other functions



Separate workflow operations process their data concurrently

Each token entering a workflow is processed concurrently, Skitter will replicate individual components as needed

/	*n*
state_change	1
state_change hidden	1

	/	**external_effect**
/	replay	replay (after_failure)
state_change	restore, replay	restore, replay (after_failure)
state_change hidden	restore checkpoint, replay	restore checkpoint, replay (after_failure)

Skitter automatically handles replication and partial failure handling based on the effects of a component

VRIJE UNIVERSITEIT BRUSSEL

Software Languages Lab

Mathijs Saey, Joeri De Koster, Wolfgang De Meuter
Skitter: A DSL for Distributed Reactive Workflows
Proceedings of the 5th ACM SIGPLAN International Workshop on Reactive and Event-Based Languages and Systems (REBLS '18)

More Information?

soft.vub.ac.be/~mathsaey/skitter/
github.com/mathsaey/skitter/
mathsaey@vub.ac.be