Customizing Storm Components for Better Reliability

Swetha Kolalapudi
CO-FOUNDER, LOONYCORN

www.loonycorn.com

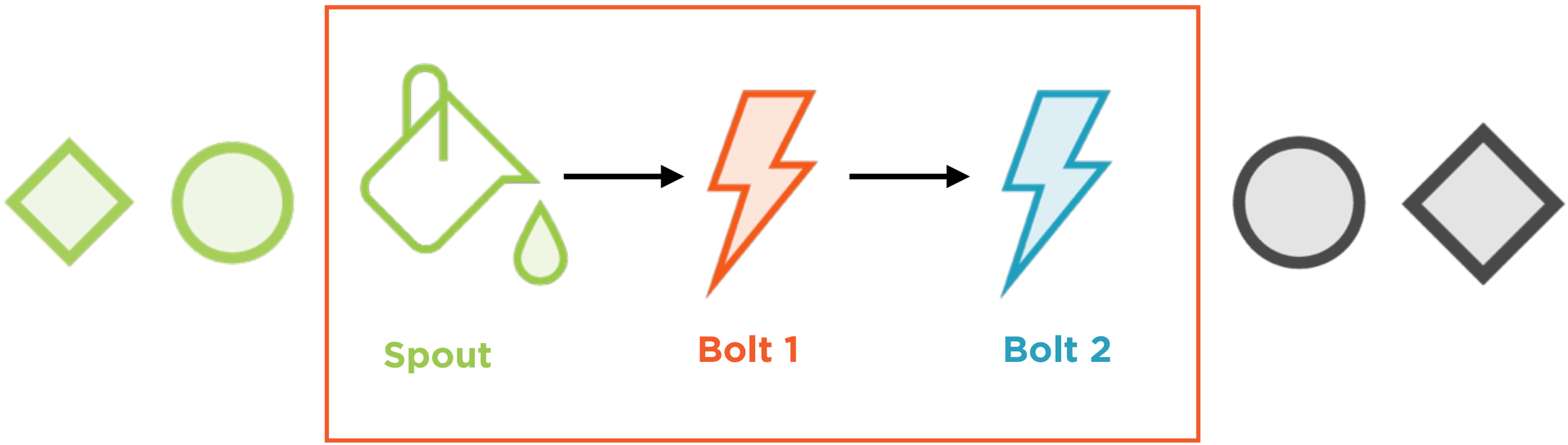
Overview

Understand how Storm guarantees message processing

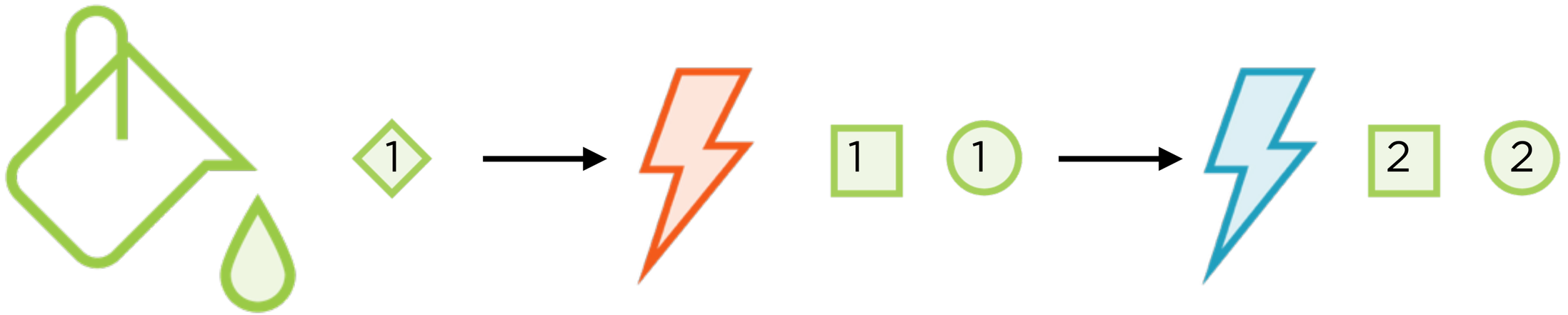
Manage reliability within a spout

Manage reliability within a bolt

Storm Topology

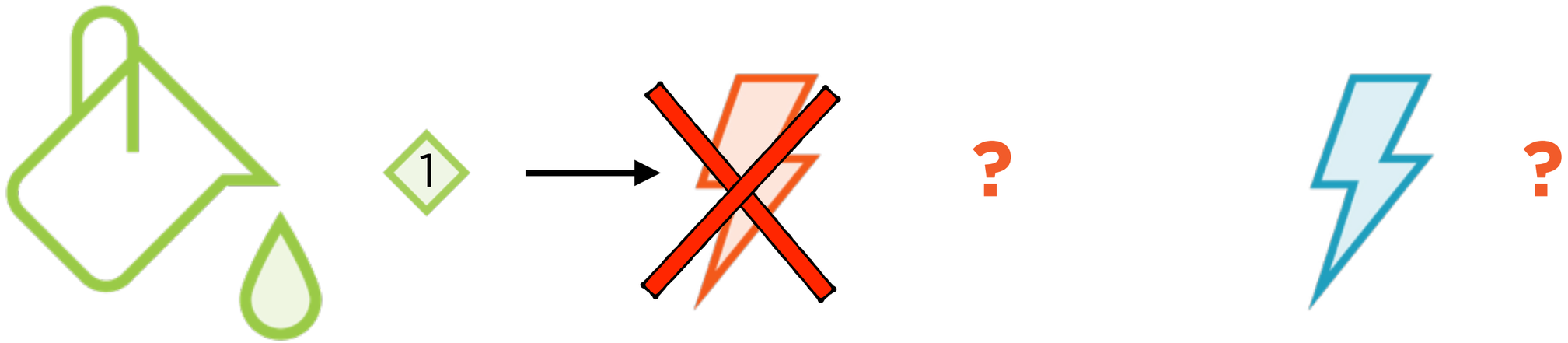


Storm Topology

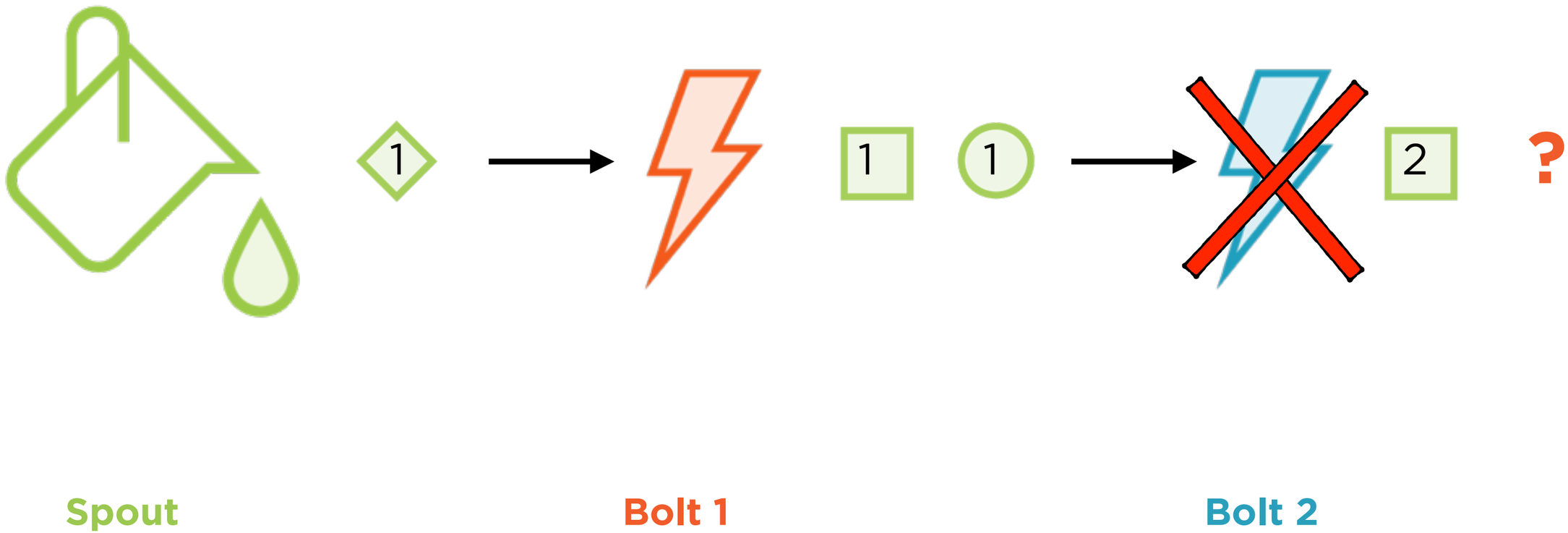


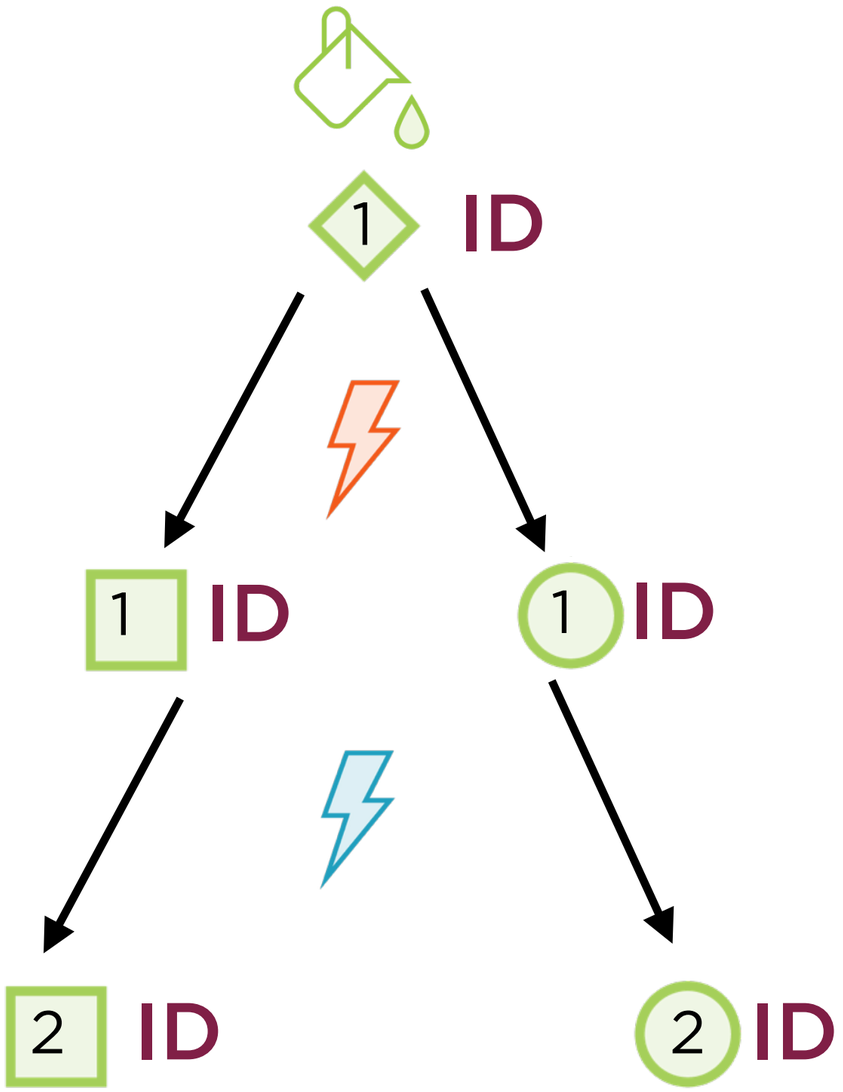
Spout **Bolt 1** **Bolt 2**

Storm Topology



Storm Topology





Tuple Tree

Spout assigns a tuple id

Id is carried over to all child tuples

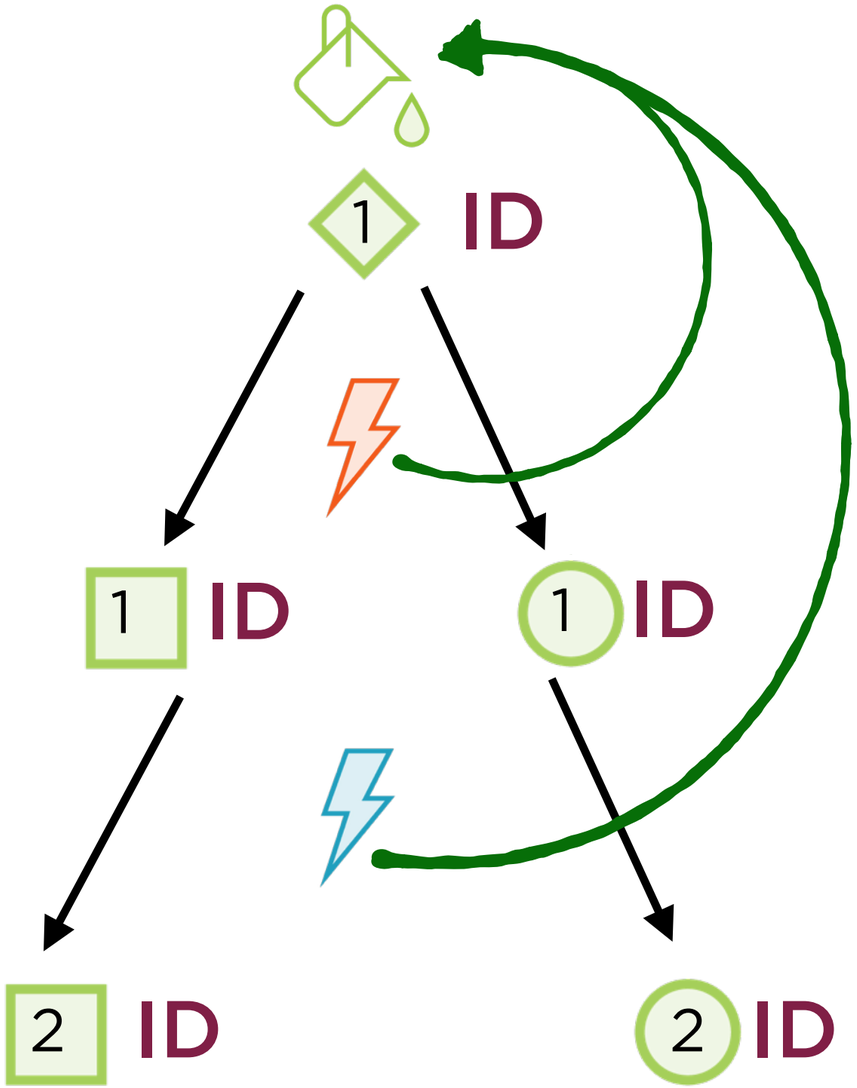
```
collector.emit(new Values("MSFT"), msgId)
```

Emit Tuple ID in Spout

```
collector.emit(input, new Values("MSFT"))
```

Link to Parent Tuple in Bolt

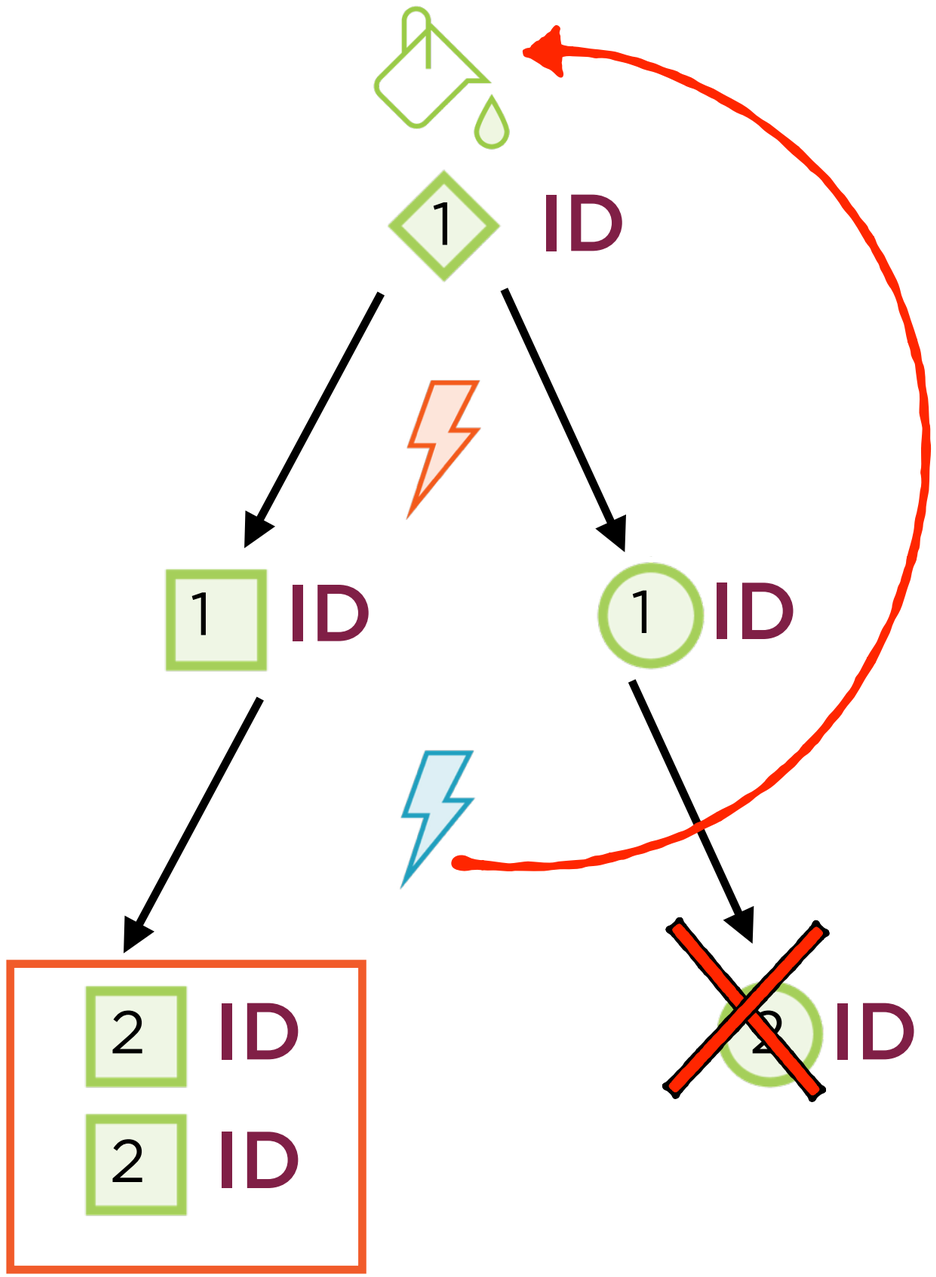
Anchoring



Managing Reliability

Acknowledge successful processing

Spout waits until all child tuples are successfully processed

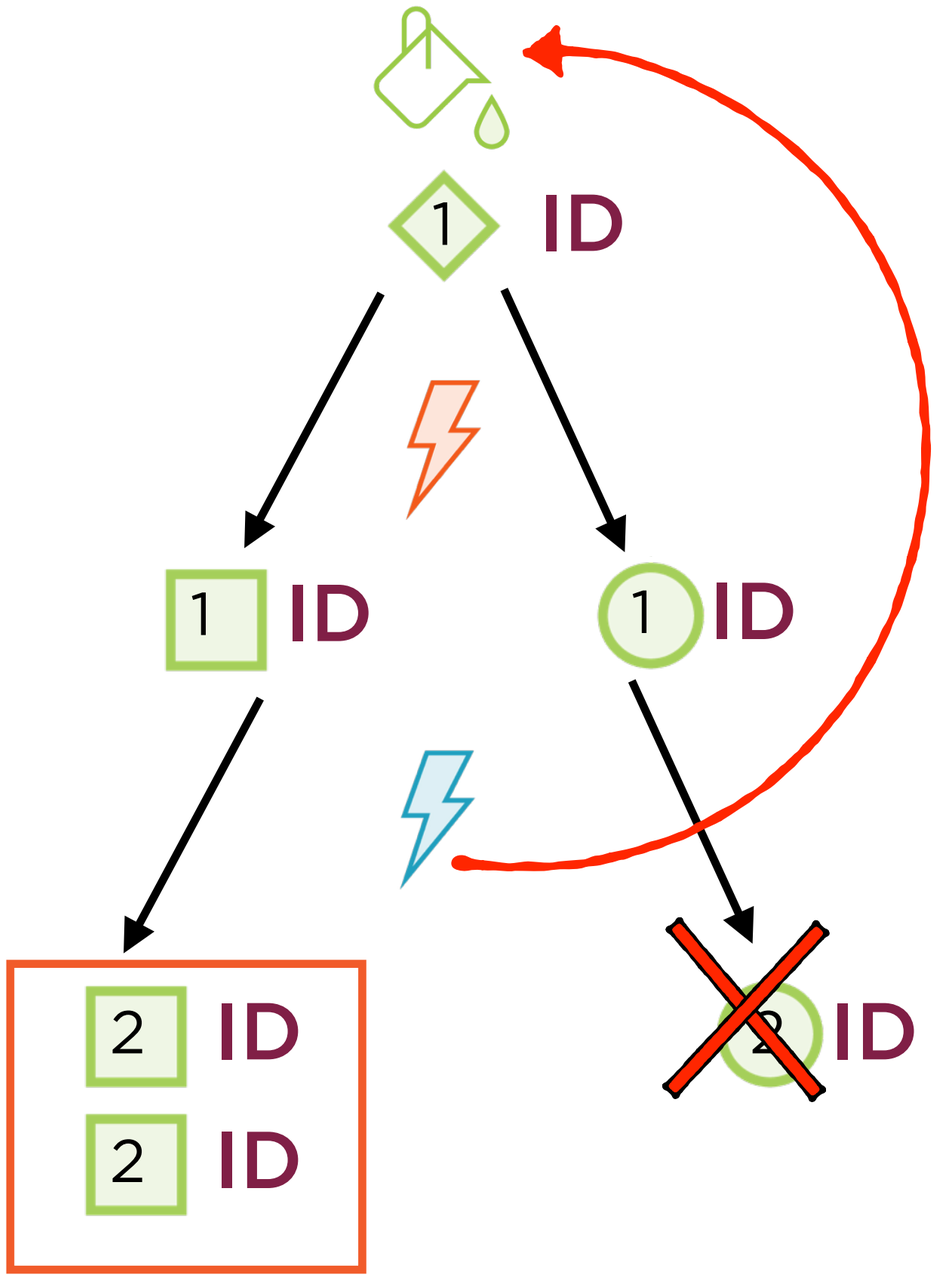


Managing Reliability

If any child tuple fails

Send a failure message

Replay the root tuple



Managing Reliability

If any child tuple fails

Send a failure message

Replay the root tuple

Managing Spout Reliability

Extend the BaseRichSpout class

Emit tuple with ID

Implement ack(), fail()

```
public void open(Map conf, TopologyContext context,
                 SpoutOutputCollector collector) {}

public void nextTuple() {}

public void declareOutputFields(OutputFieldsDeclarer declarer) {}
```

Extending BaseRichSpout

```
public void open(Map conf, TopologyContext context,
                 SpoutOutputCollector collector) {}

public void nextTuple() {}

public void declareOutputFields(OutputFieldsDeclarer declarer) {}

public void ack(Object msgId) {}

public void fail(Object msgId) {}
```

Extending BaseRichSpout

```
public void open(Map conf, TopologyContext context,
                 SpoutOutputCollector collector) {}

public void nextTuple() {}

public void declareOutputFields(OutputFieldsDeclarer declarer) {}

public void ack(Object msgId) {}

public void fail(Object msgId) {}
```

Manage Spout Reliability

Called when a tuple is fully processed

```
public void open(Map conf, TopologyContext context,
                 SpoutOutputCollector collector) {}

public void nextTuple() {}

public void declareOutputFields(OutputFieldsDeclarer declarer) {}

public void ack(Object msgId) {}

public void fail(Object msgId) {}
```

Manage Spout Reliability

Called when any tuple in the tree fails

Managing Bolt Reliability

Extend the BaseRichBolt class (instead of BaseBasicBolt)

Send acknowledgement/ failure messages

```
public void prepare(Map stormConf, TopologyContext context) {}

public void execute(Tuple input,BasicOutputCollector collector){}
```

Extending BaseBasicBolt

```
public void prepare(Map stormConf, TopologyContext context,
OutputCollector collector) {}

public void execute(Tuple input){}
```

Extending BaseRichBolt

Random Failure Topology

Spout

Bolt

Reads words from a file

Converts to lowercase

Randomly fails 20% of the time

Random Failure Topology

Set up a Spout

Spout reads from a file

Set up the Bolt

Randomly fails 20% of the time

Run the topology

Build the topology

Submit to a local cluster

Demo

Implementing the spout

Demo

Implementing the bolt

Running the topology

Overview

Understand how Storm guarantees message processing

Manage reliability within a spout

Manage reliability within a bolt