www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Multi-tenant Pub/Sub Processing for IoT Data Streams

Álvaro Villalba - alvaro.villalba@bsc.es

(22/06/2016)

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

INTRODUCTION

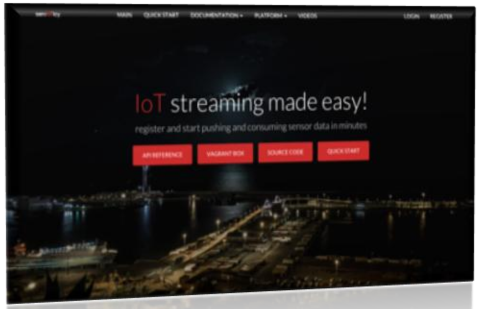
Introduction

- IoT + Big Data
- Cisco has predicted that we will have over 50 billion connected devices by 2020.
- IDC estimates 212 billion "things" globally by the end of 2020. This will include 30.1 billion installed "connected (autonomous) things" in 2020.
- 7 billion of cellphones active already.

http://www.eetimes.com/document.asp?doc_id=1321229

http://www.businesswire.com/news/home/20131003005687/en/Internet-Poised-Change-IDC

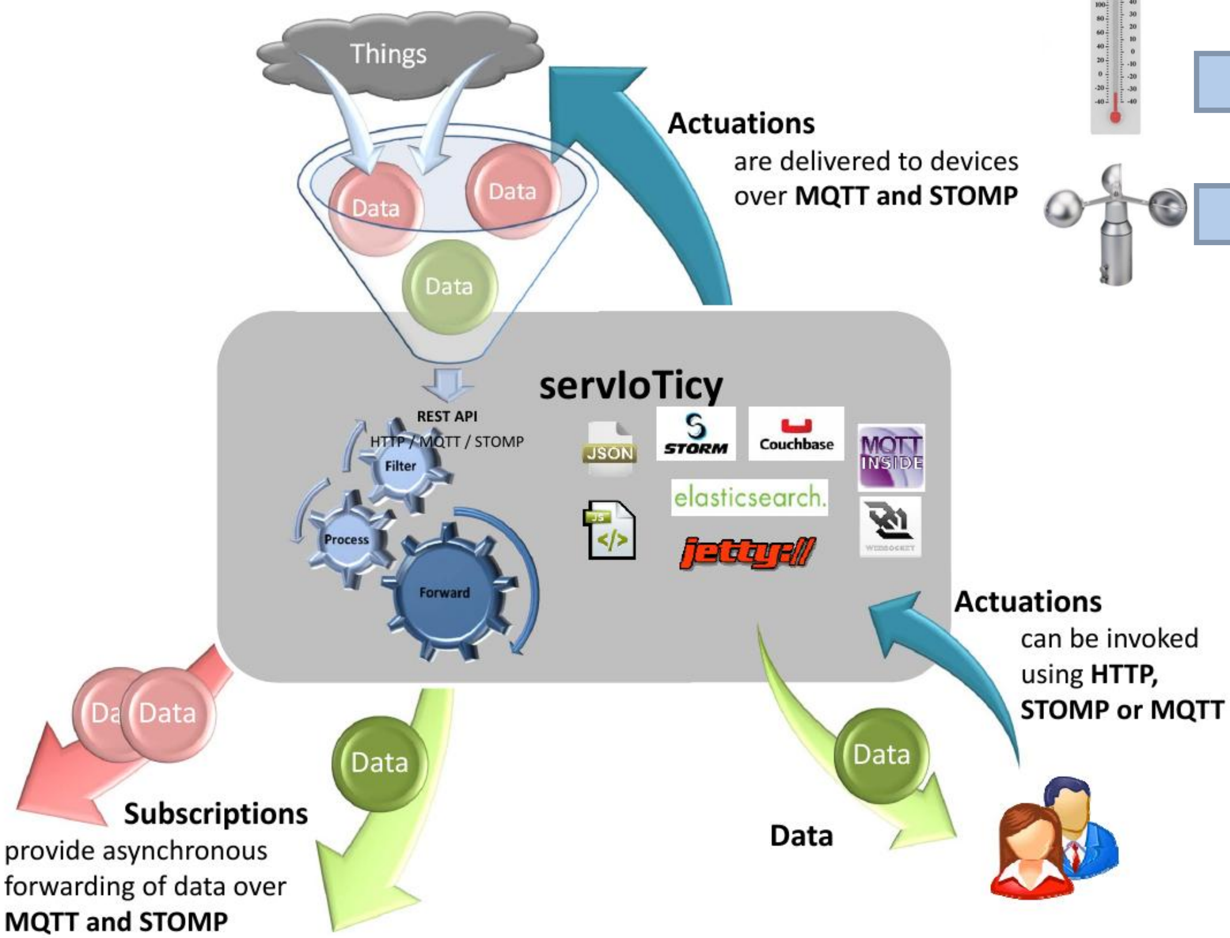
Motivation

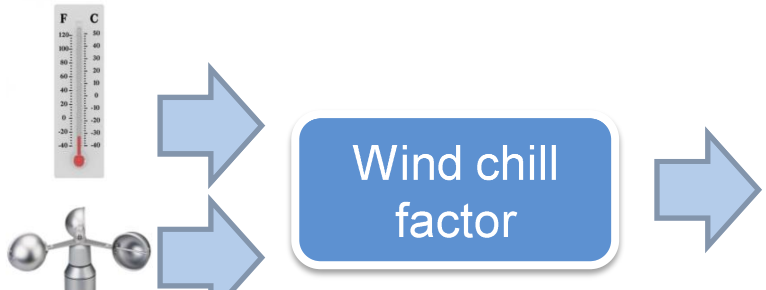


IoT Data Access → Stream Processing → Multi Tenancy

- Initiative to produce an advanced Stream Processing Platform for the IoT
- Providing means for data aggregation and processing
- Scalable runtime and multi-tenancy support
- Stream sharing
- Dynamic pipeline
- Added value chain for data
- http://serviotic y.com

servIoTicy





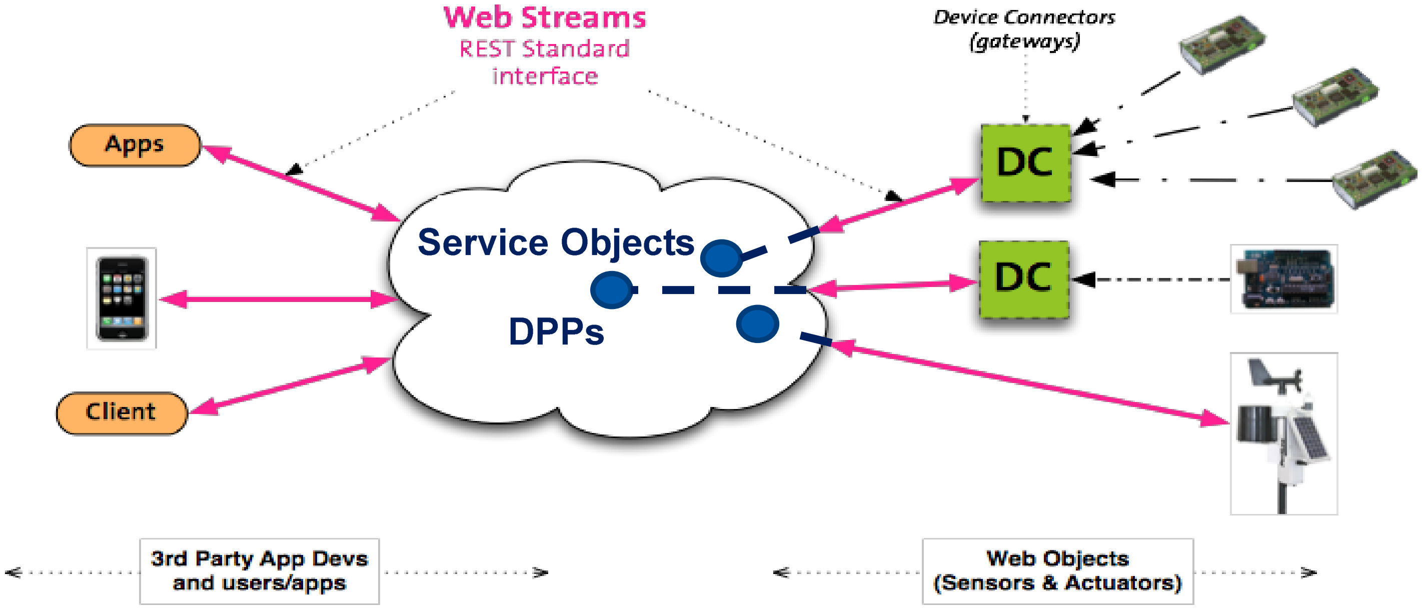
$$T_{\mathrm{wc}} = 13.12 + 0.6215T_{\mathrm{a}} - 11.37V^{+0.16} + 0.3965T_{\mathrm{a}}V^{+0.16}$$

Barcelona
Supercomputing
Center
Centro Nacional de Supercomputación

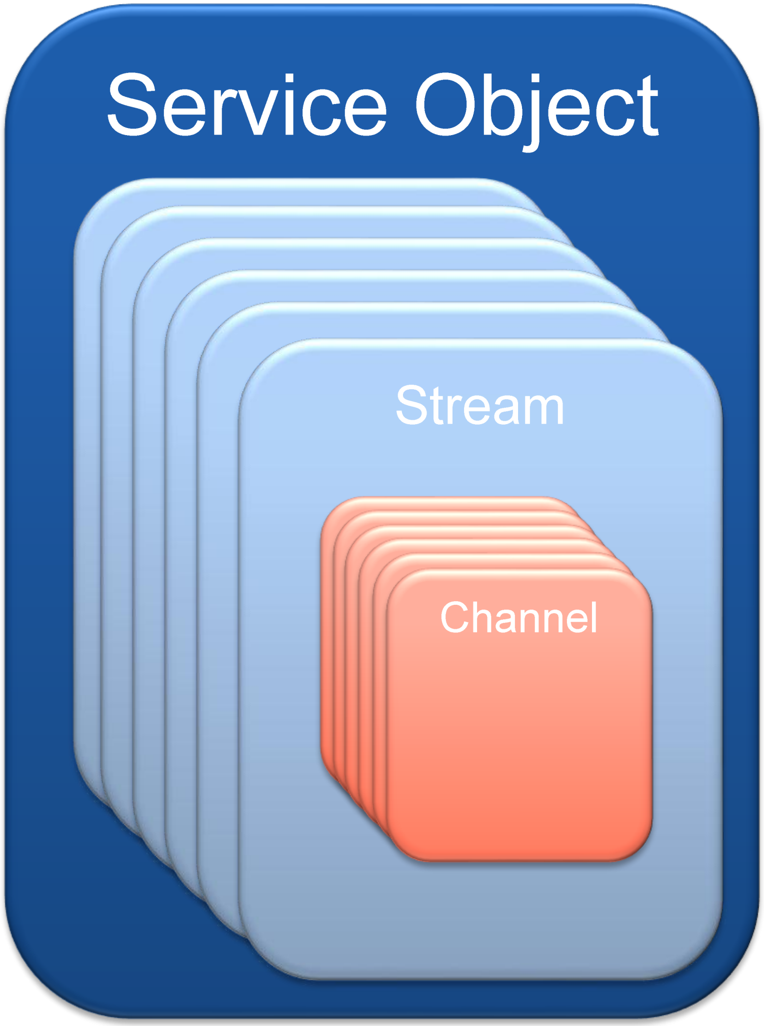
ENTITIES & ARCHITECTURE

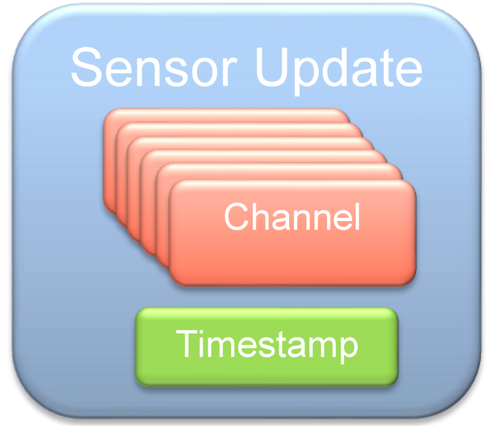
Entities

- WebObject – ServiceObject (SO) – Data Processing Pipe (DPP)



Service Object & Sensor Update structure



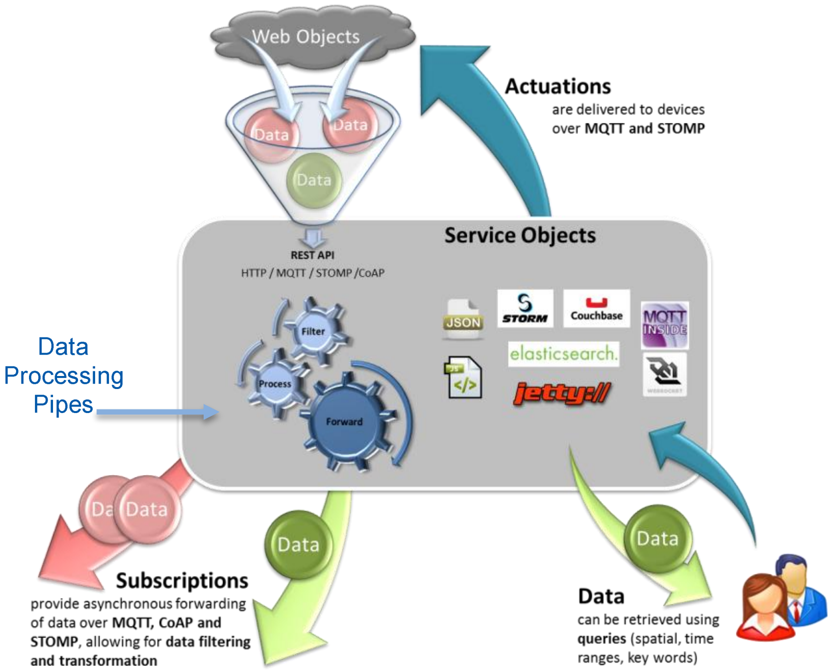


- A device has multiple sensors
- Each sensor has multiple dimensions
- Therefore
  - Service Object → Device
  - Stream → Sensor
  - Channel → Dimension
- Streams can be defined as a function of other streams

SO-SU definition & features


```
{
  "id": "1398788346015aa59fe5f854b4bda89647b969c51d8a4",
  "userId": "87a69aedb0d5410496167826b4ca2492",
  "public": "false",
  "createdAt": 1398788346015,
  "updatedAt": 1398788346015,
  "groups": {
    "group1": {
      "soIds": [
        "1396461657731411aa73c28444ecf9a8c803e62312fd1",
        "139879170177572ad72be6e67450c9d4d85176e02aeb1"
      ],
      "stream": "data"
    }
  },
  "streams": {
    "fahrenheit": {
      "channels": {
        "temperature": {
          "current-value": "{$group1.channels.temperature.current-value} * 1.8 + 32",
          "type": "number"
        },
        "location": {
          "current-value": "{$group1.channels.location.current-value}",
          "type": "string"
        }
      }
    },
    "aboveSeventy": {
      "channels": {
        "temperature": {
          "current-value": "{$fahrenheit.channels.temperature.current-value}",
          "type": "number"
        },
        "location": {
          "current-value": "{$fahrenheit.channels.location.current-value}",
          "type": "string"
        }
      },
      "post-filter": "{$result.cha
    }
  }
}
```

```
{
  "channels": {
    "location": {
      "current-value": "38.9815891823,-0.731087365108"
    },
    "temperature": {
      "current-value": 39.83
    }
  },
  "lastUpdate": 1398432725
}
```

API (docs.servioticy.com)

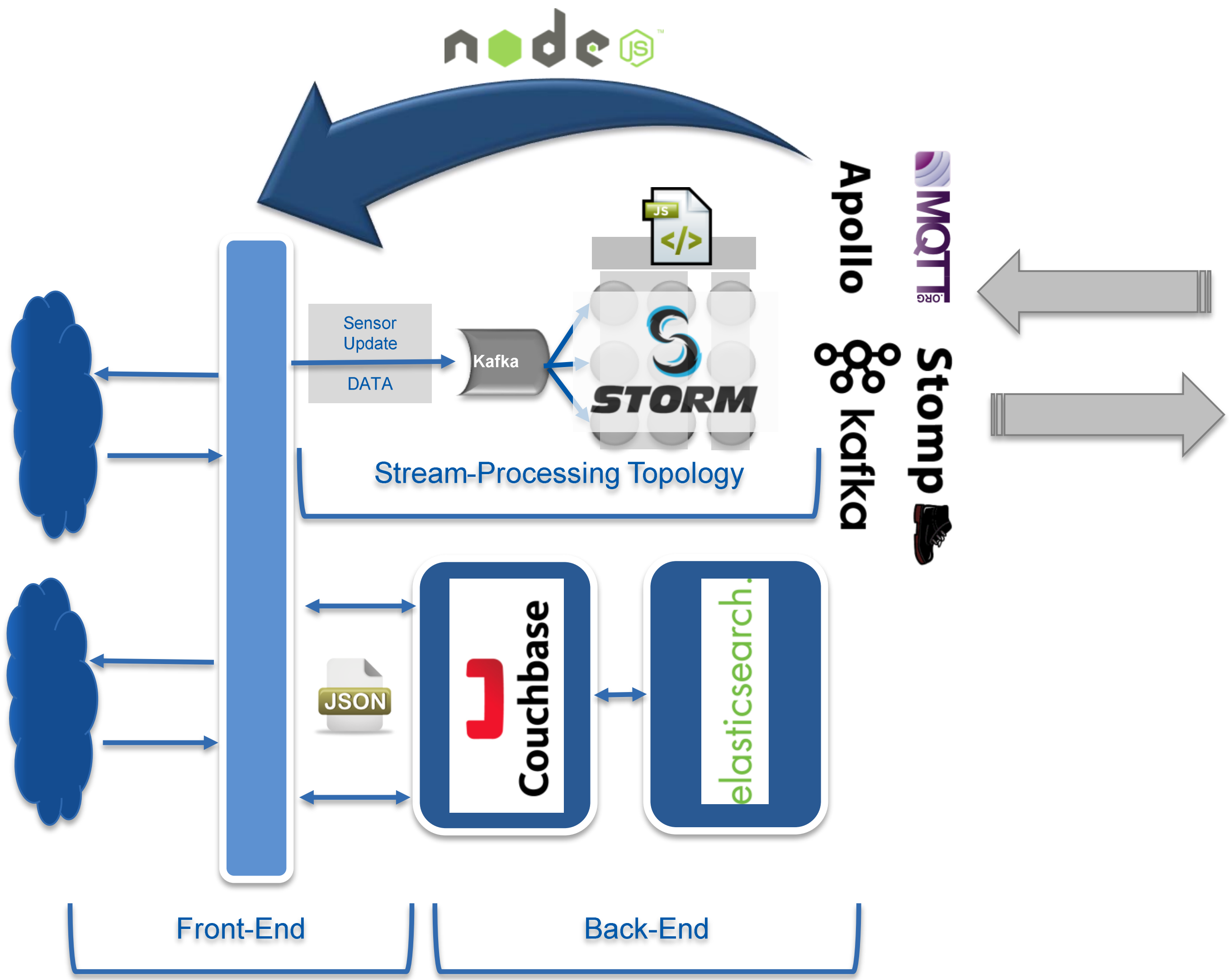
- Service Objects are exposed through a RESTful API.

operation	Target URI	Role
Create	POST /	Create a new SO posting a JSON document.
Retrieve	GET /	Retrieve the list of all the SOs created.
Retrieve	GET /[soId]	Retrieve attributes from the <soId> Service Object.
Update	PUT /[soId]	Modify the <soId> ServiceObject.
Delete	DEL /[soId]	Delete the <soId> Service Object.
Retrieve	GET /[soId]/streams	Retrieve the list of all the SO streams.
Create	POST /[soId]/streams/[streamId]/subscriptions	Subscribe the SO <soId> to a service.
Update	PUT /[soId]/streams/[streamId]	Store <soId> data puting a JSON document.
Retrieve	GET /[soId]/streams/[streamId]	Retrieve the list of all the data of <soId> Service Object.
lastUpdate	GET /[soId]/streams/[streamId]/lastUpdate	Retrieve the last piece of data generated by a <soId> Service Object.

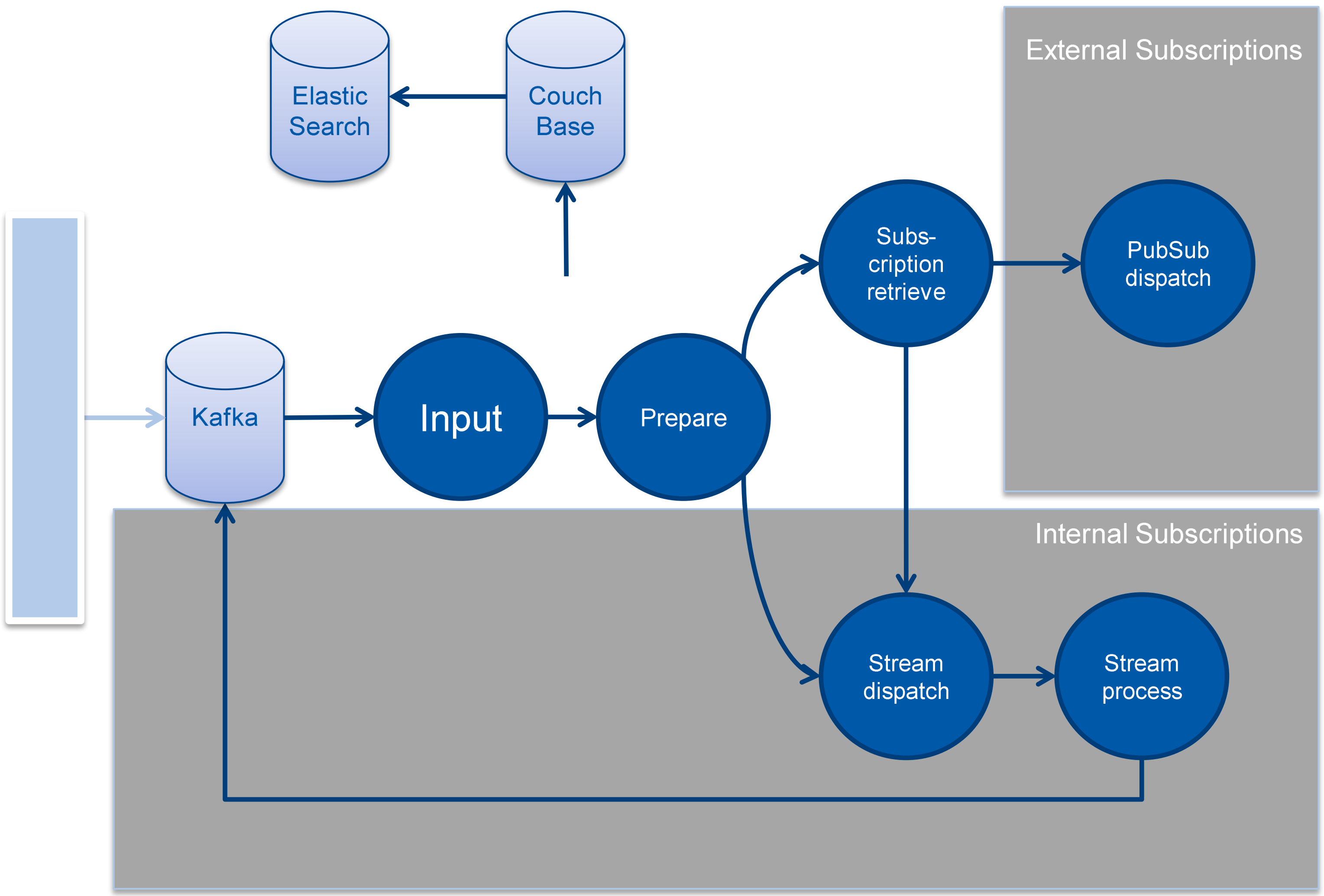
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

+ actuations + searches

Data-Centric Architecture: technologies



Storm topology

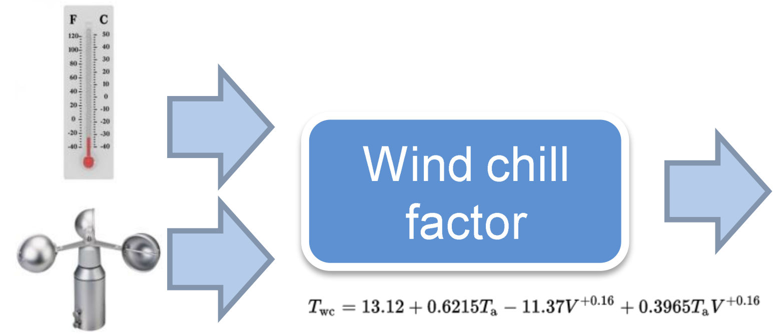


Barcelona Supercomputing Center
Centro Nacional de Supercomputación

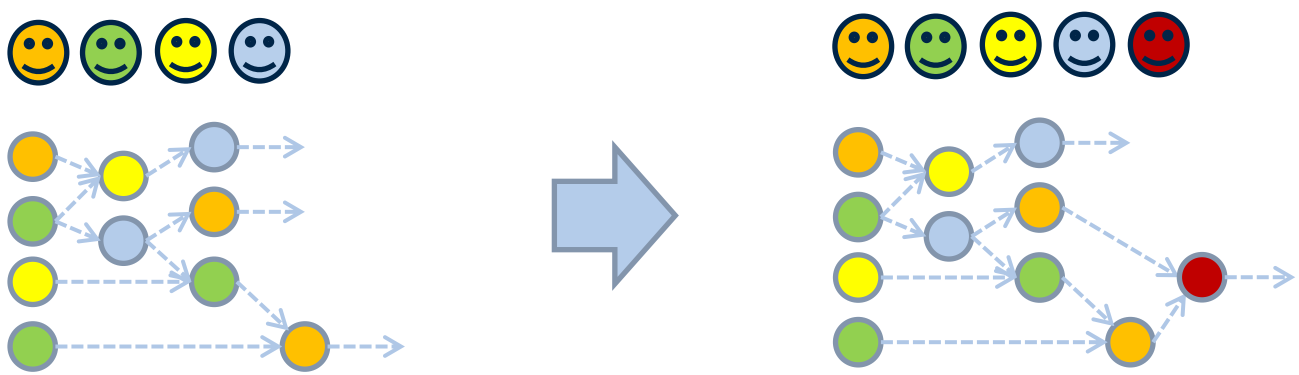
DATA PROCESSING PIPELINES

Data processing pipelines features

- Publisher-Subscriber
- User code
- Expansion on-the-fly
- Reactive computation
- Programing model
- Authorization policies
- Dynamic subscriptions

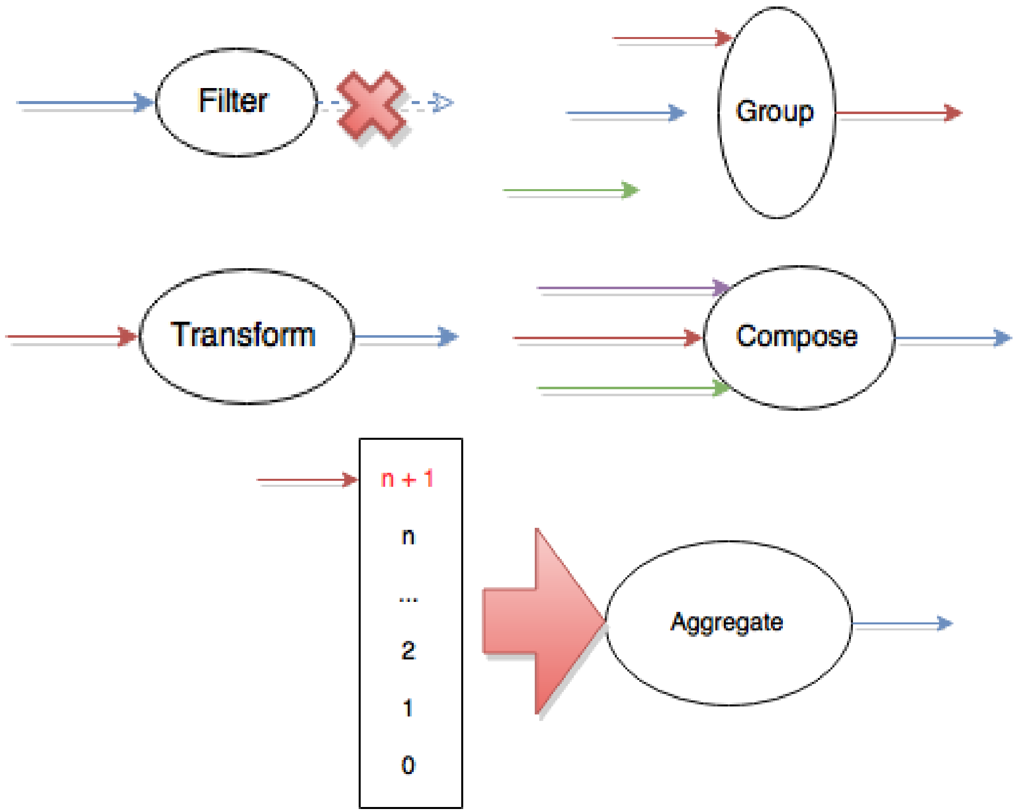


$$T_{wc} = 13.12 + 0.6215T_a - 11.37V^{+0.16} + 0.3965T_aV^{+0.16}$$



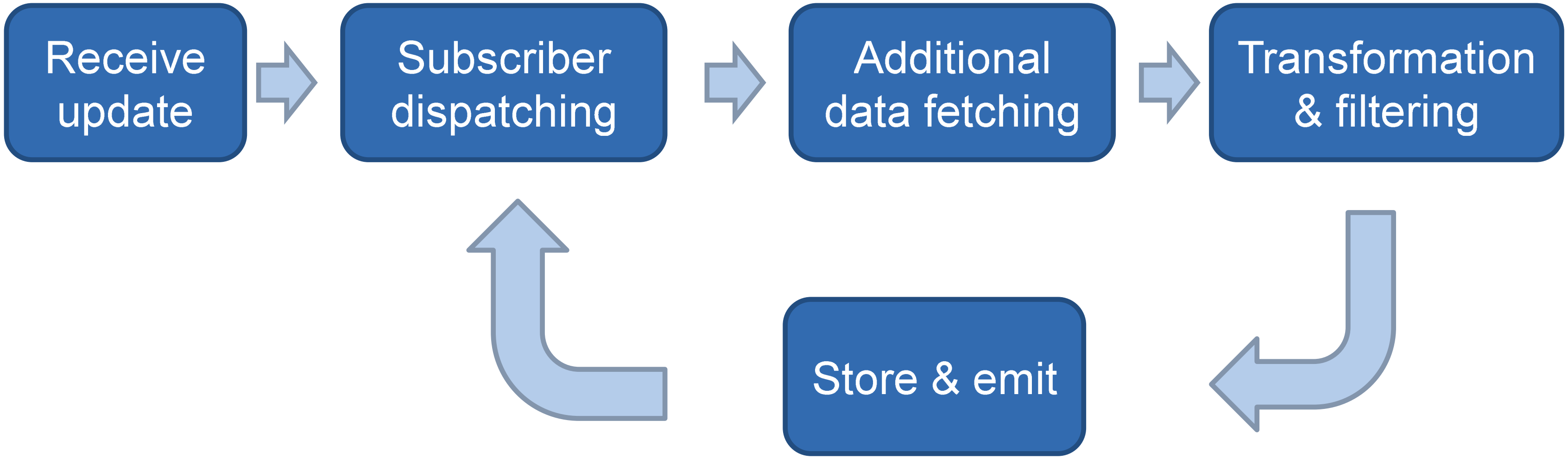
Operations

- Index and query
- Filter
- Transform
- Compose
- Aggregate
- Group



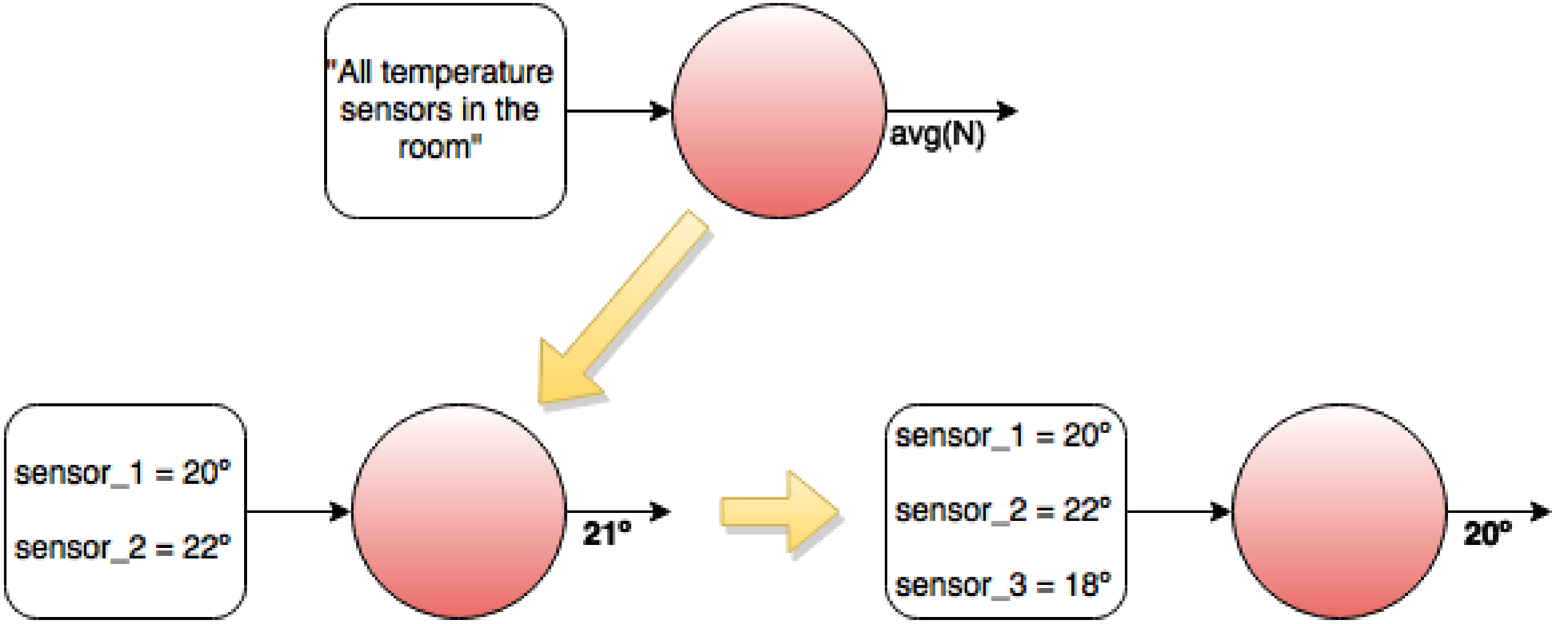
- Pub/sub communication

Data processing pipelines stages



Dynamic subscriptions

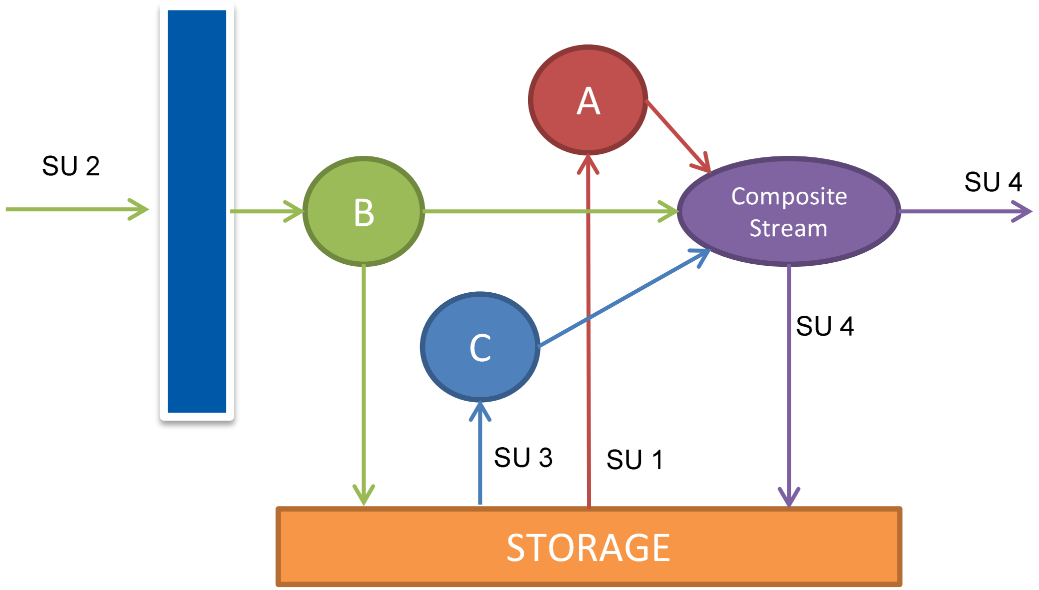
- For groups and aggregations
- Based on stream characterizations



Main design principles

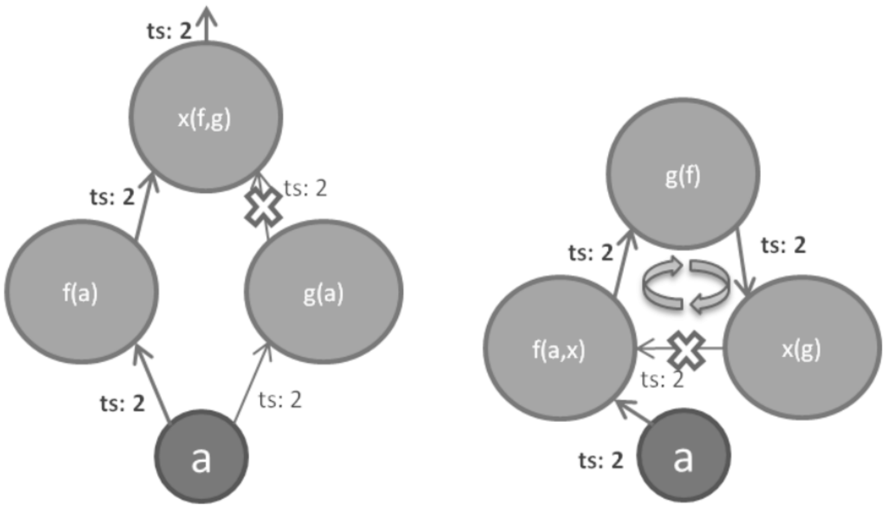
- A stream is **sequential**
  - One update after another in strict order
- Several streams with the same sources can run in parallel
- Priority in current state
- Ideally the computation is faster than the interval between updates in a stream
  - Some updates are discarded otherwise from the most active streams
- Not blocking.
  - If there is an error processing an update and new updates are waiting to be processed, discard the old one
  - Timeouts
  - No blocking 'zip' to compose streams

Non-blocking composition example



Strict order and cycles control

- Input update must be **newer** than last output update
- Output update timestamp is the higher from the inputs
- Avoids cycles and controls strict order



Barcelona Supercomputing Center
Centro Nacional de Supercomputación

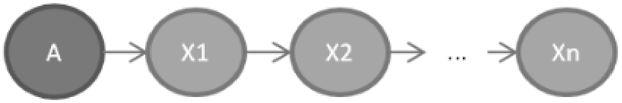
BENCHMARKING

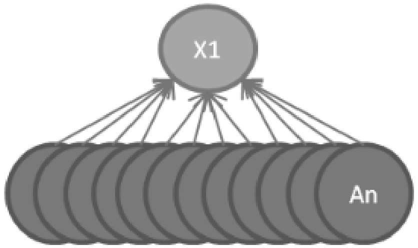
DPP generator

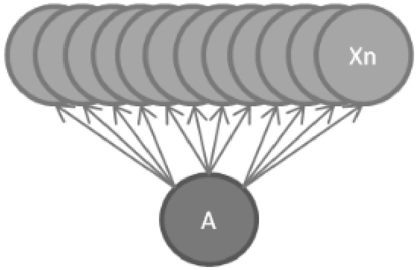
- Python scripts collection
  - NetworkX module
- Configurable:
  - Number of independent topologies
  - Number of SOs
  - Number of streams (initial and composite)
  - Number of channels
  - Number of operands per composite stream
  - Distribution used to select operands
  - Random operands
- All numeric values can be specified as a statistical distribution
- Measuring DPP stages latencies
  - Input and Output
- Measuring source to sink latencies
- `https://github.com/servioticy/servioticy-dispatcher-benchmark`

Significative graphs

- Measuring latency impact on:
  - Length
  - In-degree
  - Out-degree

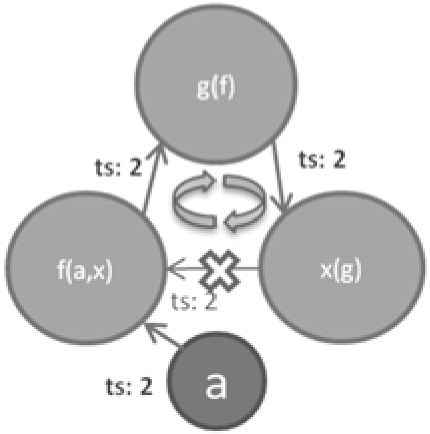


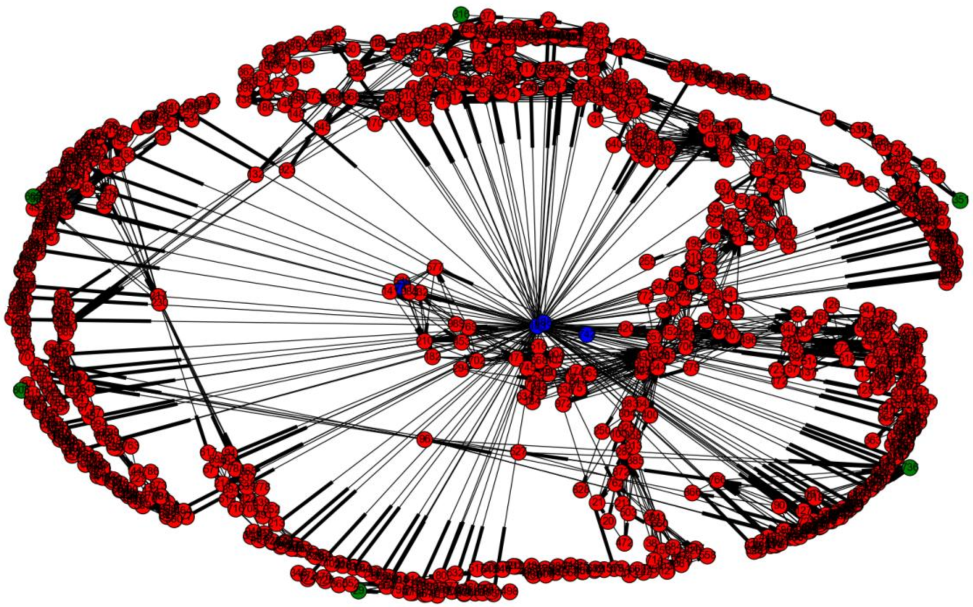




First try – Random graphs with cycles

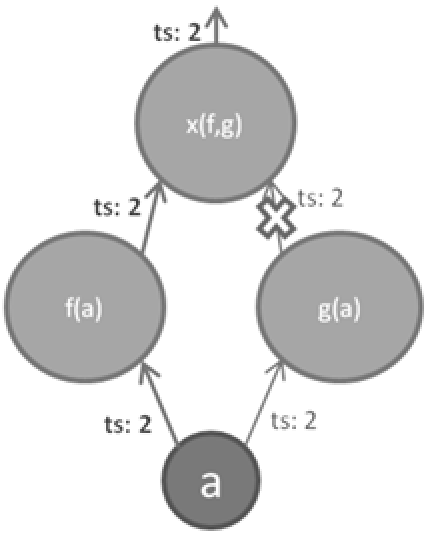
- Big graphs with cycles (+1000 DPPs)
- High latency
- Most computations were noise
  - Updates that did not reach the sink DPPs

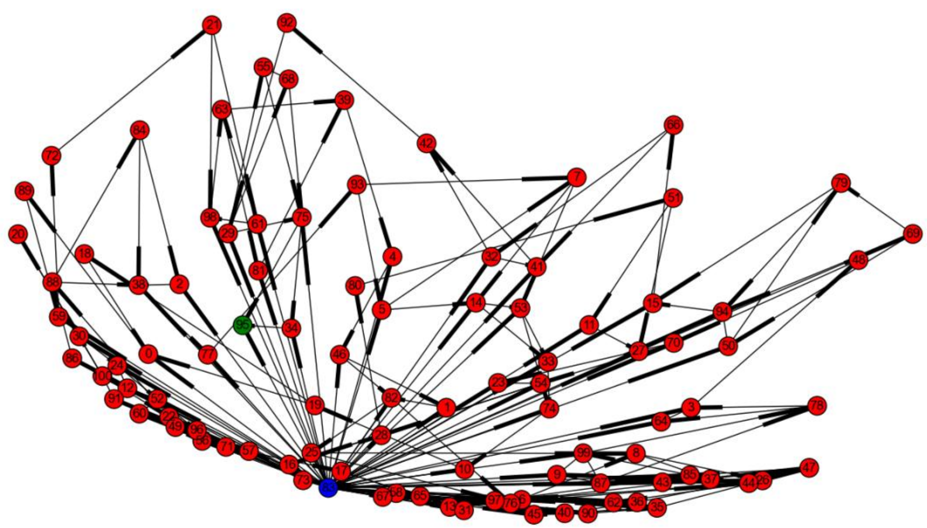




Second try – Random DAGs

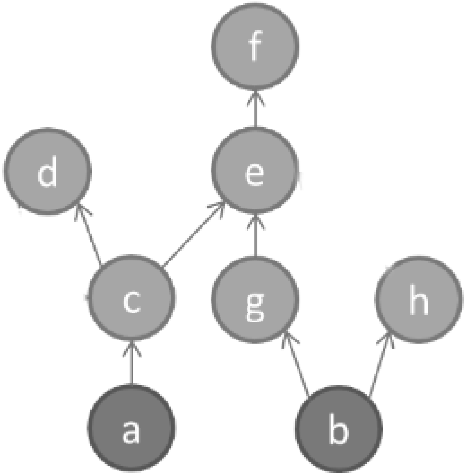
- DAGs with +100 DPPs
- Still high latency
- Most of noise was still there

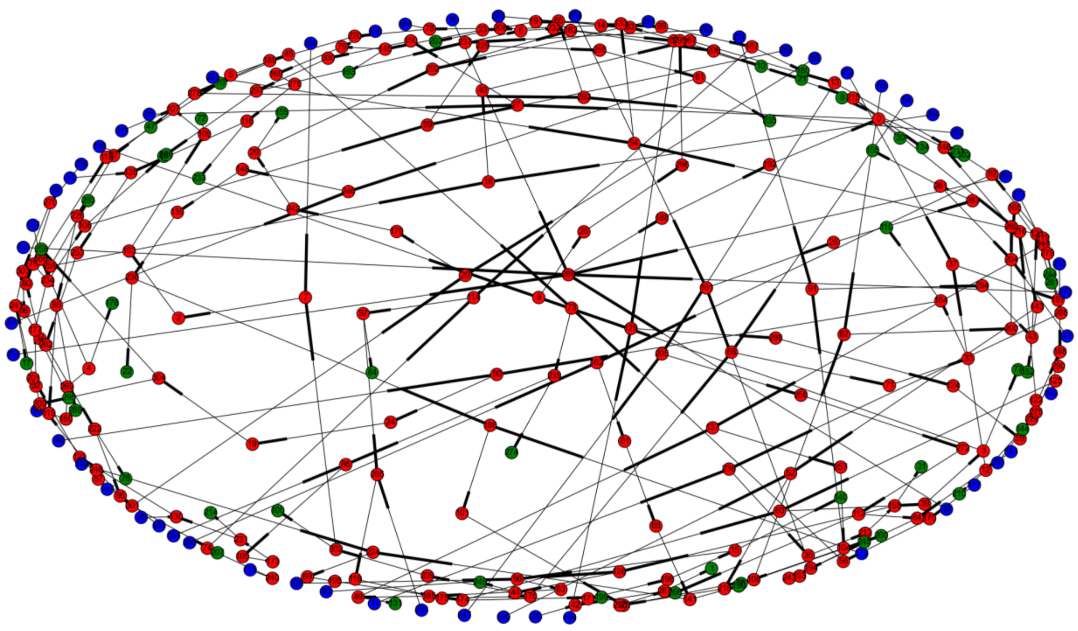




Third try – Random 'merged trees'

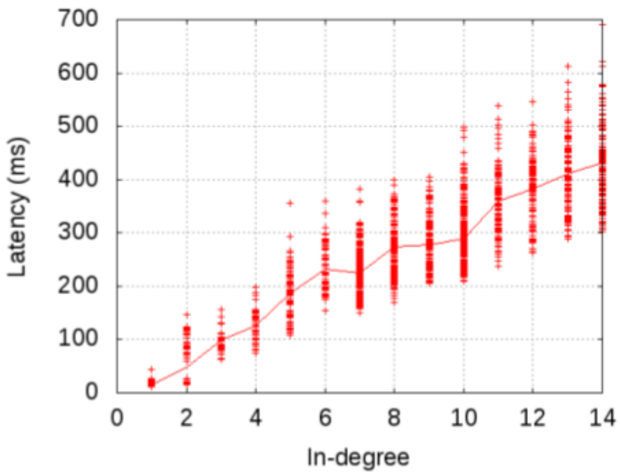
- Graphs with one source-sink path
- Sinks shared between sources
- +100 DPPs
- Without noise

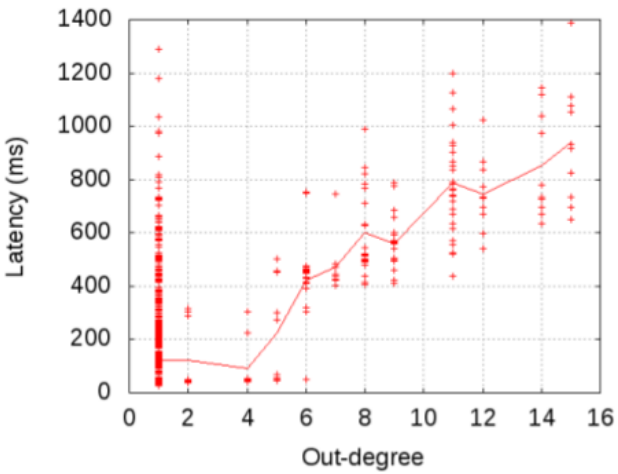




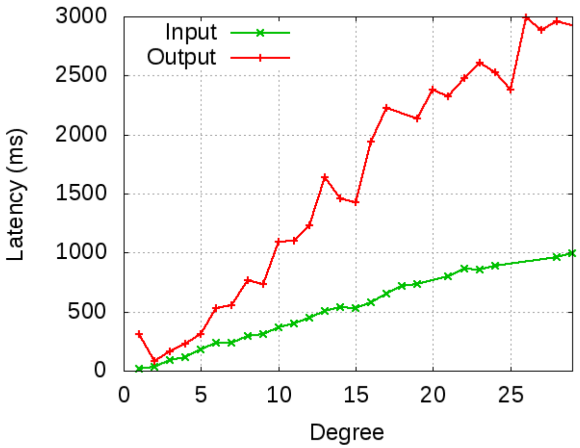
Stages latency

- 149 edges
- 42 nodes

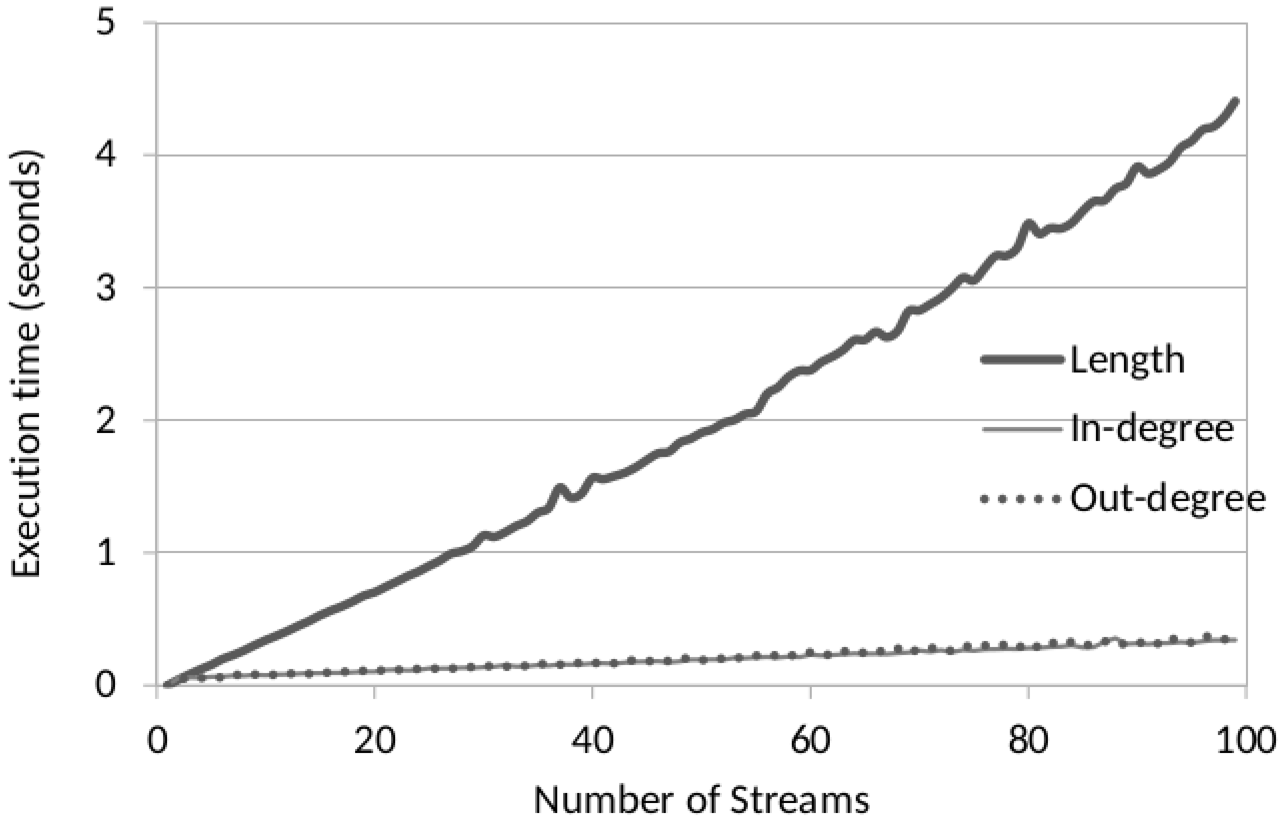




- 523 edges
- 80 nodes

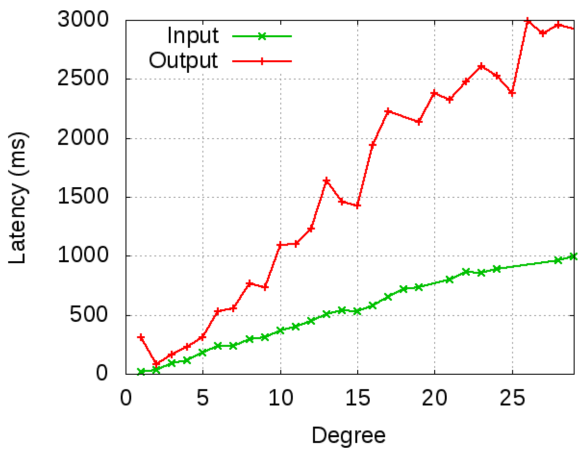


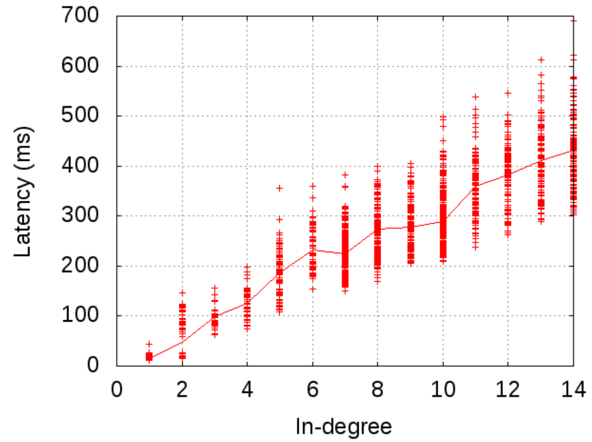
Significative graphs latencies

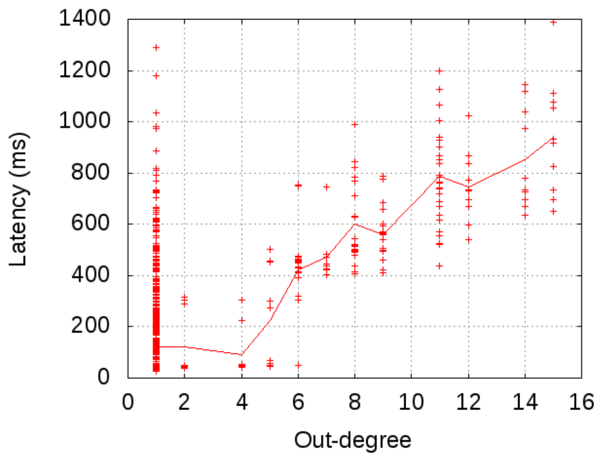


Completed work

- ServIoTicy[1]
  - Multi-tenant and centralized IoT platform
  - Event-driven and non-blocking







- A. Villalba, D. Carrera and J. L. Perez, "*ServIoTicy Data Pipelines: Multi-Tenant Data Processing Topologies for the Internet of Things*", submitted in *Cloud Computing for IoT Special Issue*, IEEE Internet of Things Journal.

Barcelona Supercomputing Center

Centro Nacional de Supercomputación

Thank you!

For further information please contact me

alvaro.villalba@bsc.es