



# Best practices and lessons learnt from Running NiFi at Renault

Kamelia Benchekroun- Big Data Architect  
Renault

Abdelkrim Hadjidj - Solution Engineer  
Hortonworks

GROUPE RENAULT



# Agenda

- The NiFi journey at Renault
- Best practices for running NiFi in production
- Lessons learnt at Renault
- Questions & answers

# The NiFi journey at Renault

**GROUPE RENAULT**



# About Us

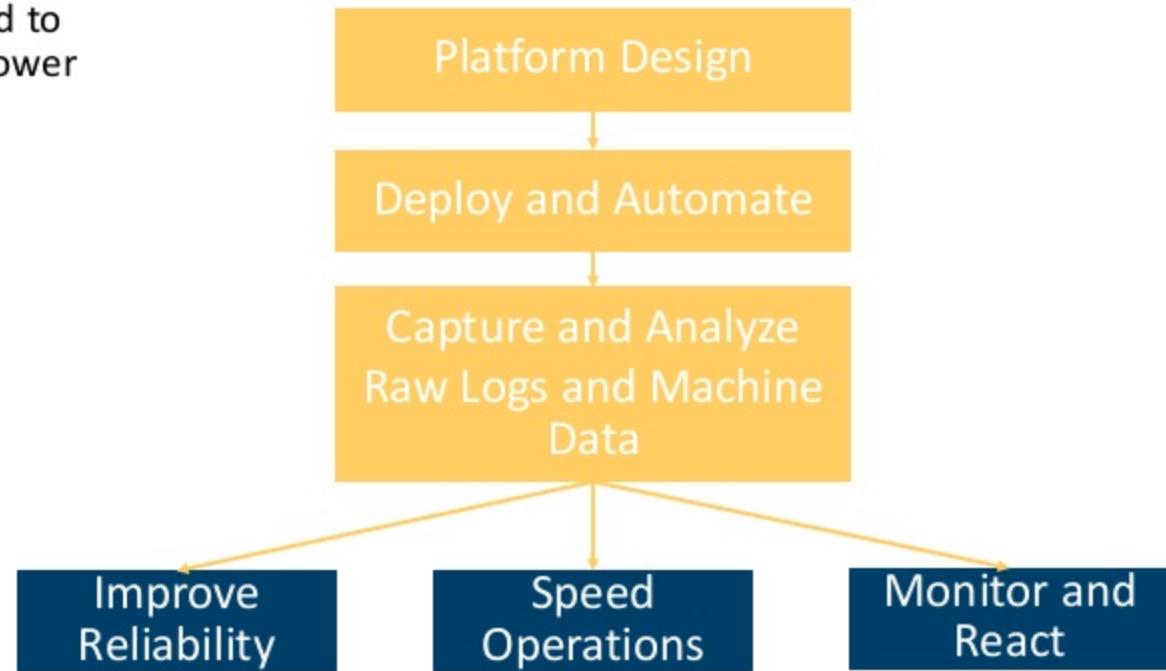
## The Datalake Squad:

- A passionate Team who work really hard to deliver solutions and services and empower Data Initiatives at Renault

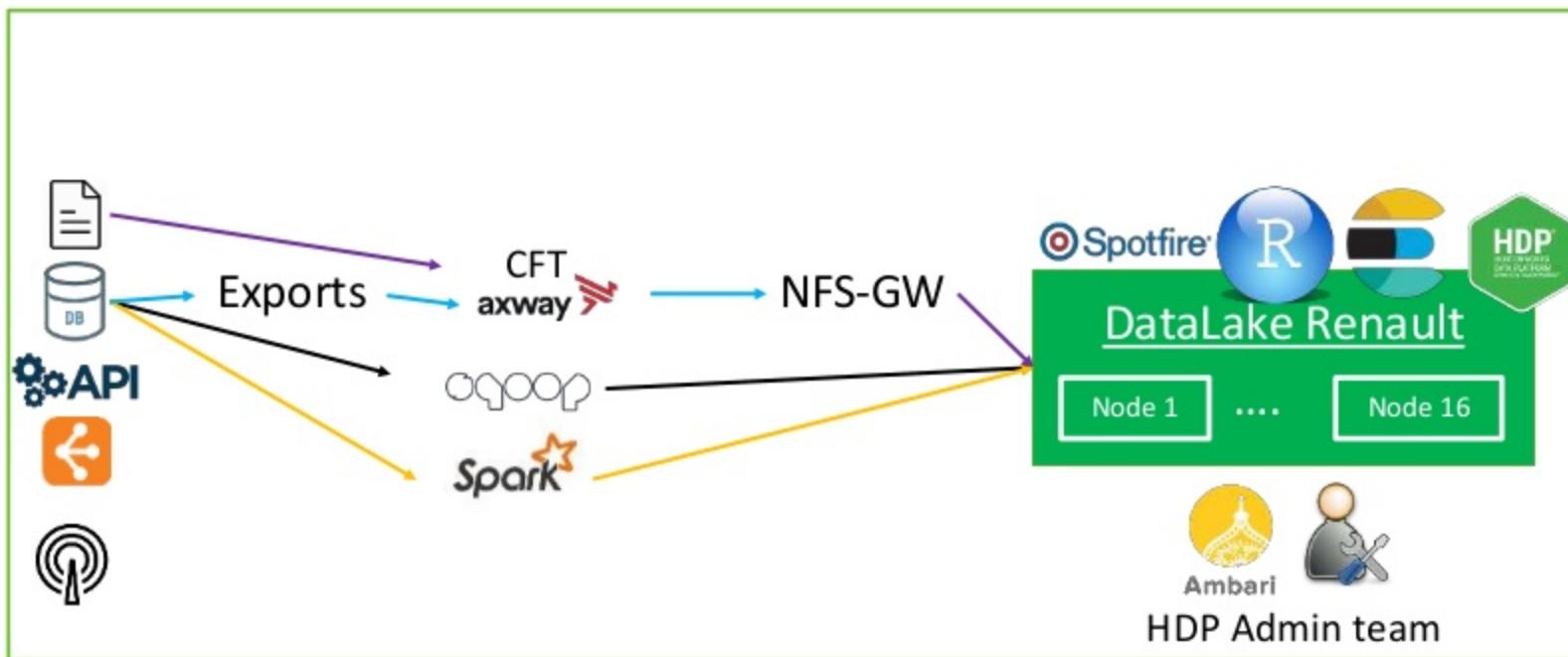
## Datalake Platform Metrics:

450	30	8000
Connected Users	Data Initiatives and use cases	Daily Queries
3500	300TB	
Service Requests	Efficient Storage	

## Datalake Squad Activities



# Data Lake at Renault – the beginning of the story



June 2016

- 10 Users
- Requests by emails, 1 Admin
- Manual provisioning

**GROUPE RENAULT**

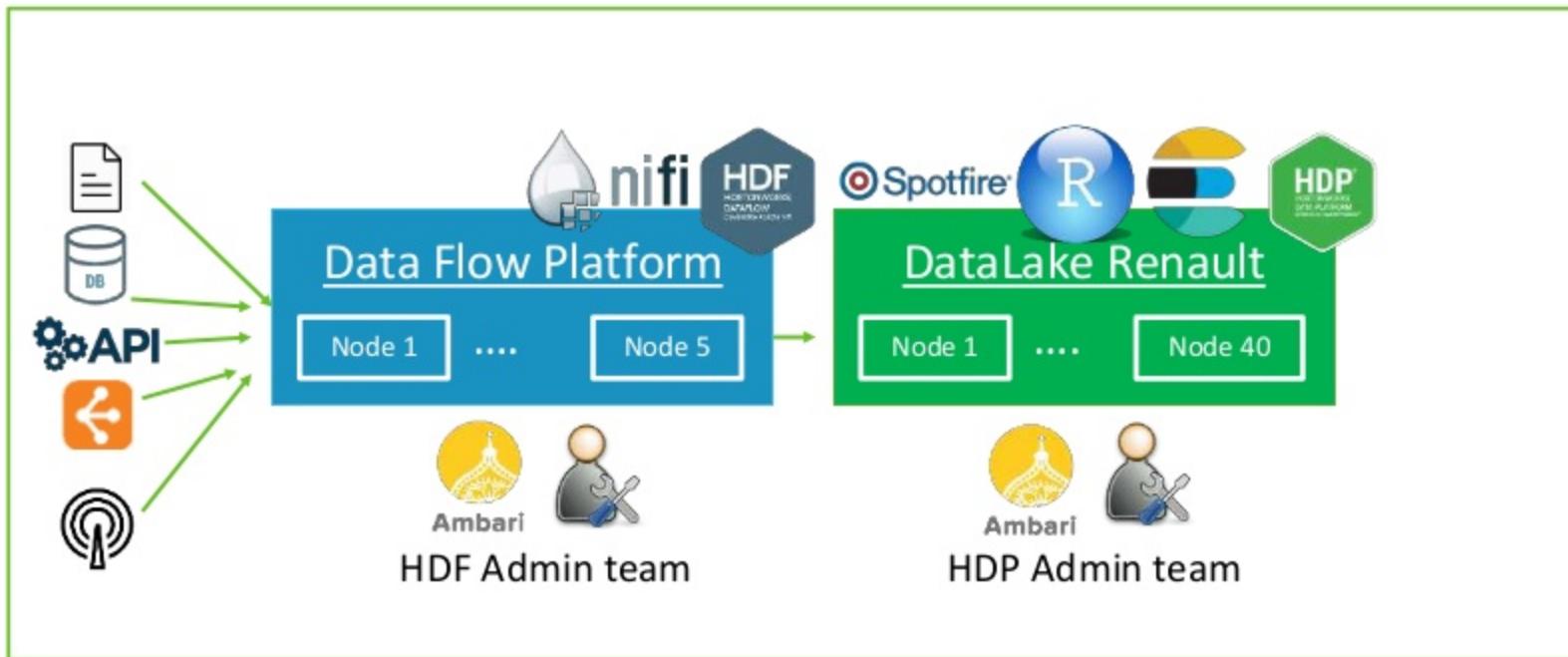


# Big Data Ingestion

GROUPE RENAULT



# HDF came in (Q1 2017)



2016

- 10 Users
- Requests by emails, 1 Admin
- Manual provisioning

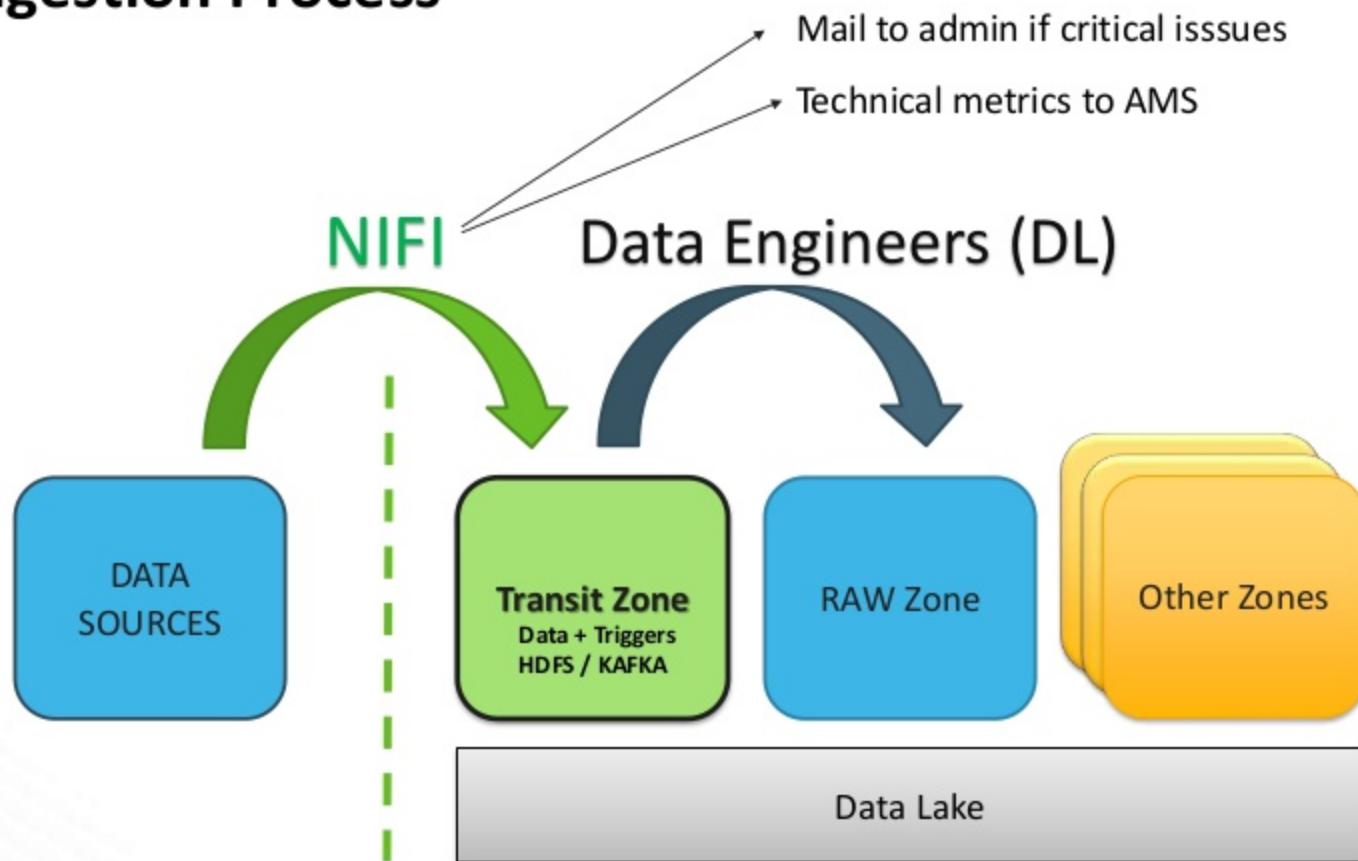
2017

- 200 users
- Jira portal, Admins/devops
- NiFi, Automation (Jenkins for HDP)

Today

- 300 – 500 active users
- 7000 query per day
- 40 data source in production

# Data Ingestion Process



# Data ingestion workflow

NIFI

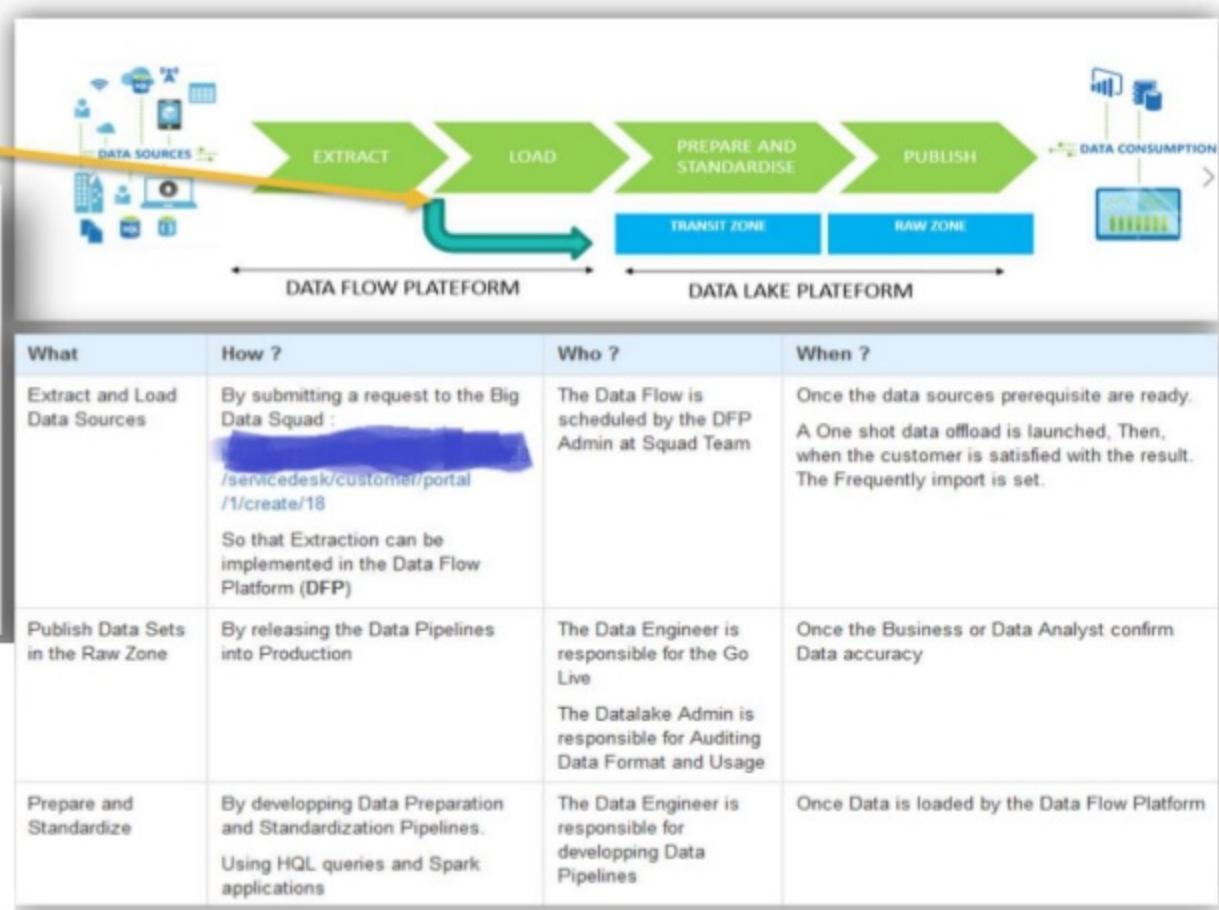
Datalake Core Services

**DLK Platform Support**

Welcome To The Datalake Platform Services! You can raise a request or report an incident from the options provided.

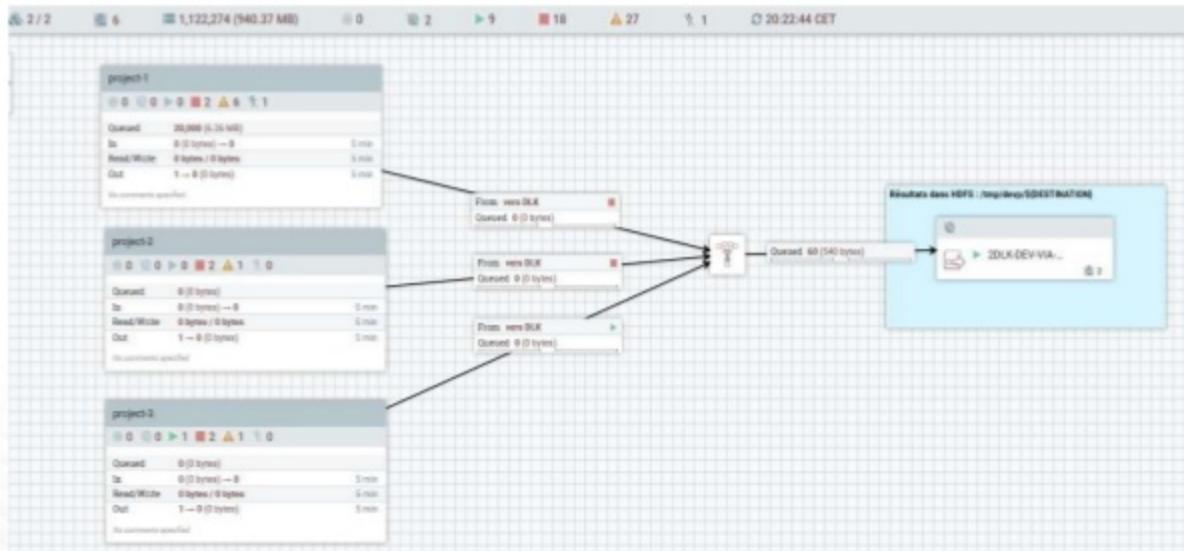
What do you need help with? 

1. Data Access Requests	 Export data from DLK Request data export from datalake
2. Provisioning Requests	 Import data to DLK Request a new data source import
<b>3. Data Flow Requests</b>	 New CFT connection Request for a new CFT connection
4. Common Requests	
5. Reporting an incident	
6. Platform usage	
7. Logins and Accounts	



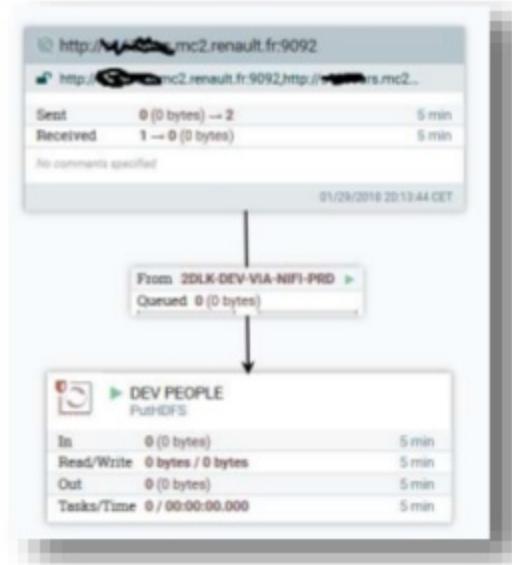
# Dev to prod testing

Dev



S2S  
→

Prod



# NiFi Value for Renault

**90%** of data ingestion since 2016

**One Platform** for all data sources : Files, DBs, API, Brokers, etc.  
**Offload CFT**

**100 active** data flows, + 2000 processors in production,.

Accelerate time to insights, use case development and improve monitoring and governance

Also used **export data** from Data Lake to other systems/sites/Cloud

Enables **new use cases** connected plants, package tracking, IoT

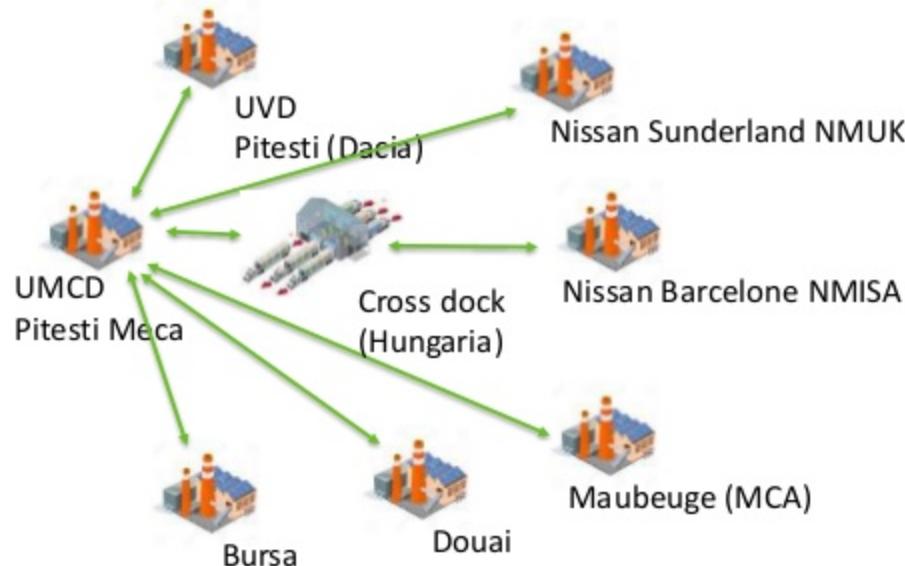
# Packaging Traceability

**GROUPE RENAULT**



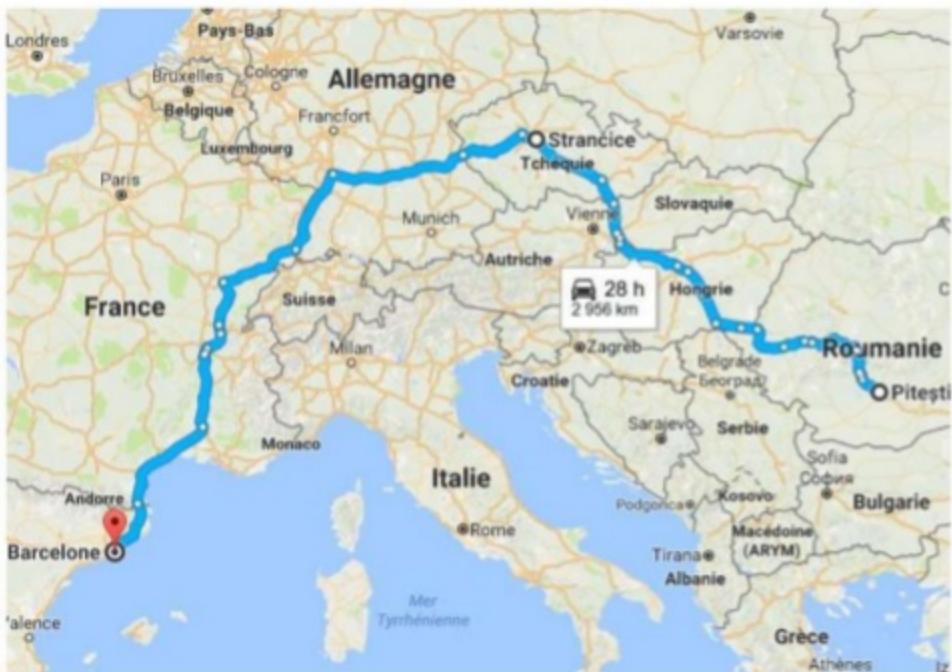
# Packaging Traceability

- POC: 2500 Packages running in the Loop
  - 1 package lost = 800 € (= iPhone)
- If the solution is generalized : 600k package
- 2016 Status
  - 400 K€ packaging re-investment due to packaging losses
- 2017 Status
  - 100 K€ cardboard in January
- Test Expectations:
  - Reduce cardboard costs and packaging losses
  - Test LoRa technology in an industrial context and Renault activities
  - Validate operational added Value of LoRa technology

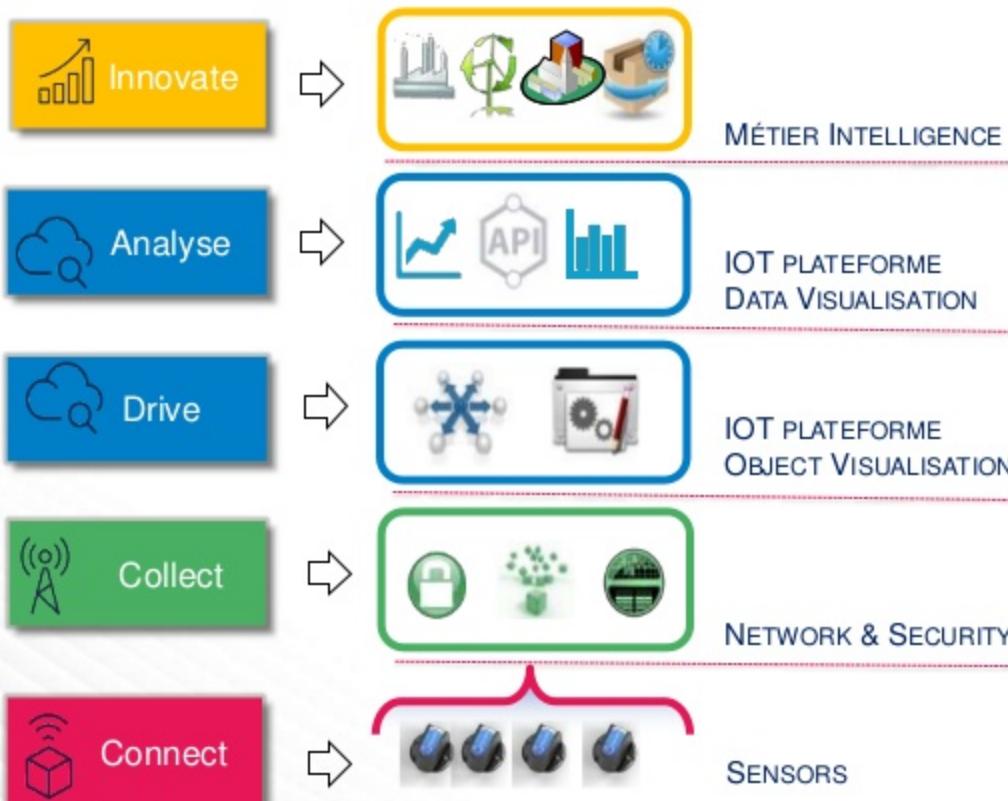


# Example flow

- Pitesti -> Barcelona (full)
  - 3 transports/week
  - 1 truck each time
  - Lead Time : 6 days
- NMUK -> Pitesti (Empty)
  - once a week
  - 1 truck each time
  - Lead Time : 6 days
  - Cross Dock in Strančice (Tchek Rep.)



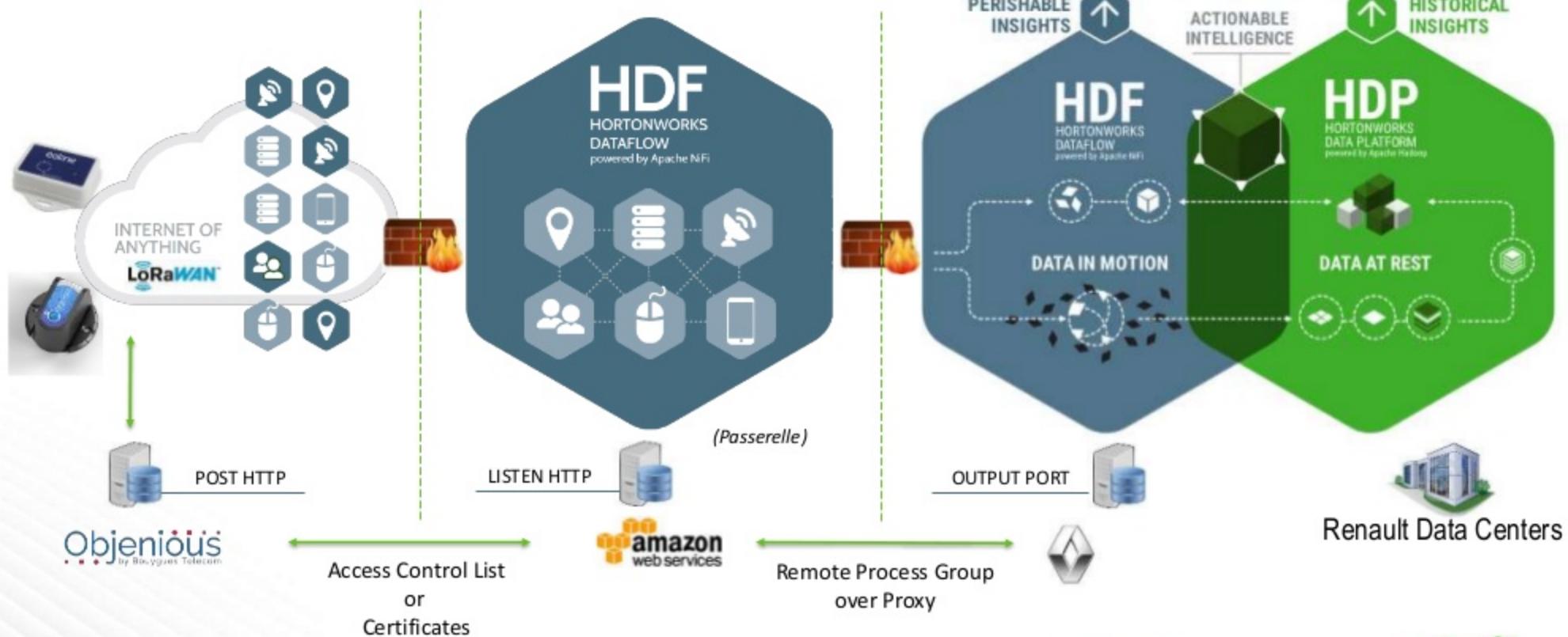
# Use case scope



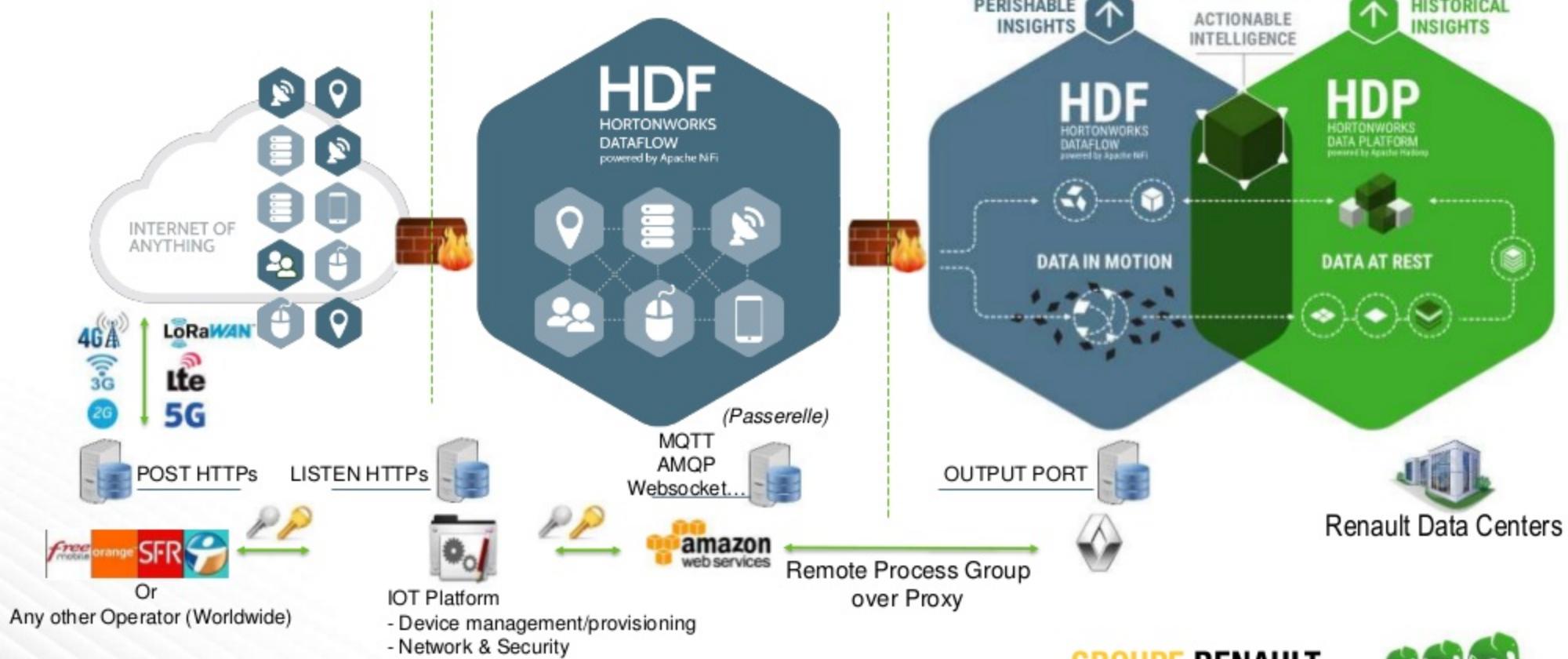
	2017	S1 2018	S2 2018	WTB
Manual Treatment				
Objenious		Renault IT (DLK + DFP + IS integration + Analytics)	Renault IT (DLK + DFP + IS integration + Analytics)	
Objenious		Objenious	Objenious	Telecom
Objenious				Sensor

**Not in Scope of Objenious offer**

# Architecture 1



# Architecture 2



# Reduce cloud communication cost by 50%

## Tag Data Decoding with NiFi ExecuteScript Processor

```
var StreamCallback = Java.type("org.apache.nifi.processor.io.StreamCallback");
var IOUtils = Java.type("org.apache.commons.io.IOUtils");
var StandardCharsets = Java.type("java.nio.charset.StandardCharsets");

var RowFile = session.get();
if (RowFile != null) {
    try {
        // Something that might throw an exception here
        // Create a new StreamCallback, passing in a function to define the interface method
        flowfile = session.writeStreamCallback();
        new StreamCallback(function (inputStream, outputStream) {
            var json = JSON.parse(IOUtils.toString(inputStream, StandardCharsets.UTF_8));
            json.payload = [];
            json.payload.push();
            json.payload[0].timestamp = json.timestamp;
            var splitKey = function (hexValue) {
                //Split each value by 2
                var arr = hexValue.match(/(.{2})/g);
                var result = {};
                var constant = ((parseFloat(3.6) - parseFloat(2.8)) / 255).toFixed(5);
                for (var key in arr) {
                    if (key == '0') {
                        result.battery = parseFloat([parseFloat(constant) * parseInt(arr[key], 16) + parseFloat(2.8)].toFixed(4));
                    }
                    if (key == '1') {
                        result.temperature = parseInt(arr[key], 16);
                    }
                }
            }
            Set empty string
        });
    } catch (e) {
        log.error("Error processing file " + RowFile.getName() + ": " + e.message);
    }
}
```

Configure Processor

SETTINGS SCHEDULING PROPERTIES COMMENTS

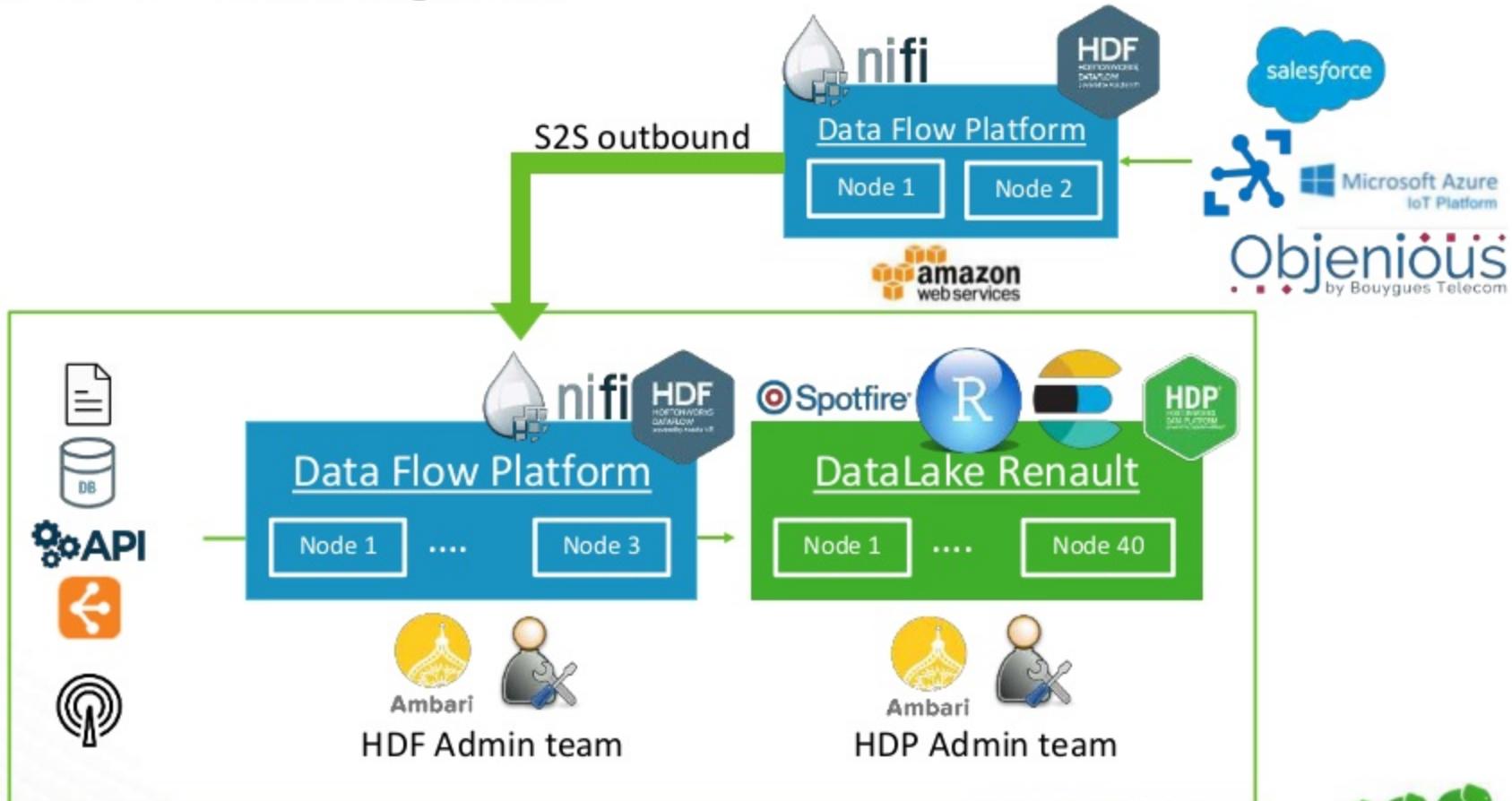
Required field

Property	Value
Script Engine	ECMAScript
Script File	No value set
Script Body	var StreamCallback = Java.type("org.apache.nifi.processor.io.StreamCallback"); var IOUtils = Java.type("org.apache.commons.io.IOUtils"); var StandardCharsets = Java.type("java.nio.charset.StandardCharsets");  var RowFile = session.get(); if (RowFile != null) { try { // Something that might throw an exception here // Create a new StreamCallback, passing in a function to define the interface method flowfile = session.writeStreamCallback(); new StreamCallback(function (inputStream, outputStream) { var json = JSON.parse(IOUtils.toString(inputStream, StandardCharsets.UTF_8)); json.payload = []; json.payload.push(); json.payload[0].timestamp = json.timestamp; var splitKey = function (hexValue) { //Split each value by 2 var arr = hexValue.match(/(.{2})/g); var result = {}; var constant = ((parseFloat(3.6) - parseFloat(2.8)) / 255).toFixed(5); for (var key in arr) { if (key == '0') { result.battery = parseFloat([parseFloat(constant) * parseInt(arr[key], 16) + parseFloat(2.8)].toFixed(4)); } if (key == '1') { result.temperature = parseInt(arr[key], 16); } } } Set empty string }); } catch (e) { log.error("Error processing file " + RowFile.getName() + ": " + e.message); } }
Module Directory	No value set

Raw Data out of sensor : ff1401aa  
Decoded Data out of sensor : [{"data":{"battery":3.6007,"temperature":20}}]

CANCEL APPLY

# Architecture + Cloud ingestion



# Connected plants

GROUPE RENAULT



# Connected plants

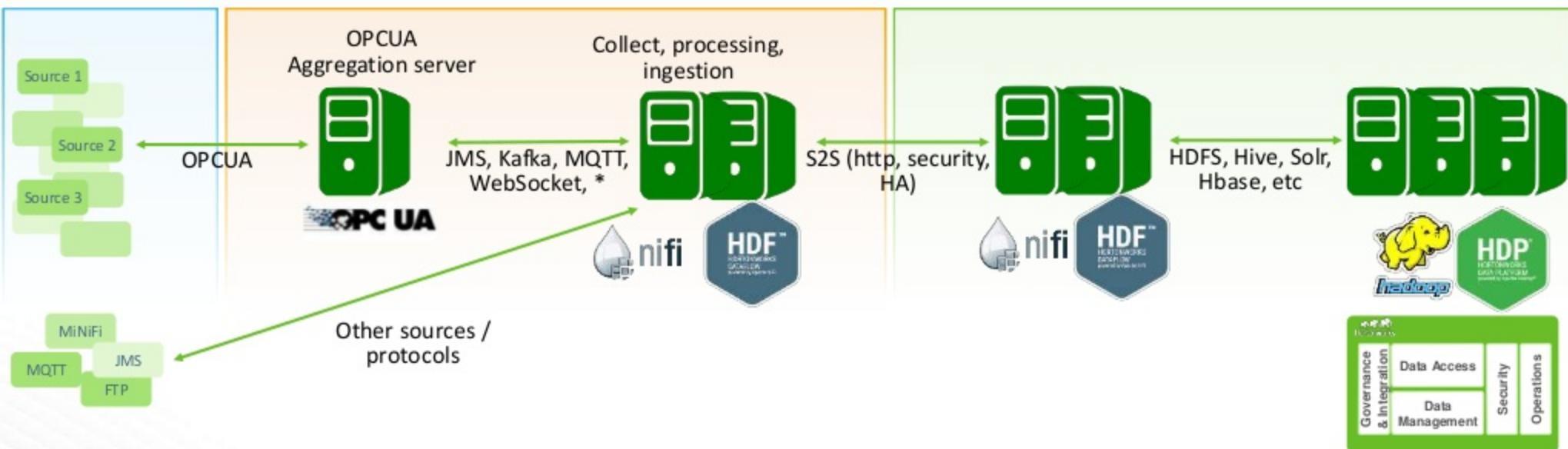
DATA IN MOTION

DATA AT REST

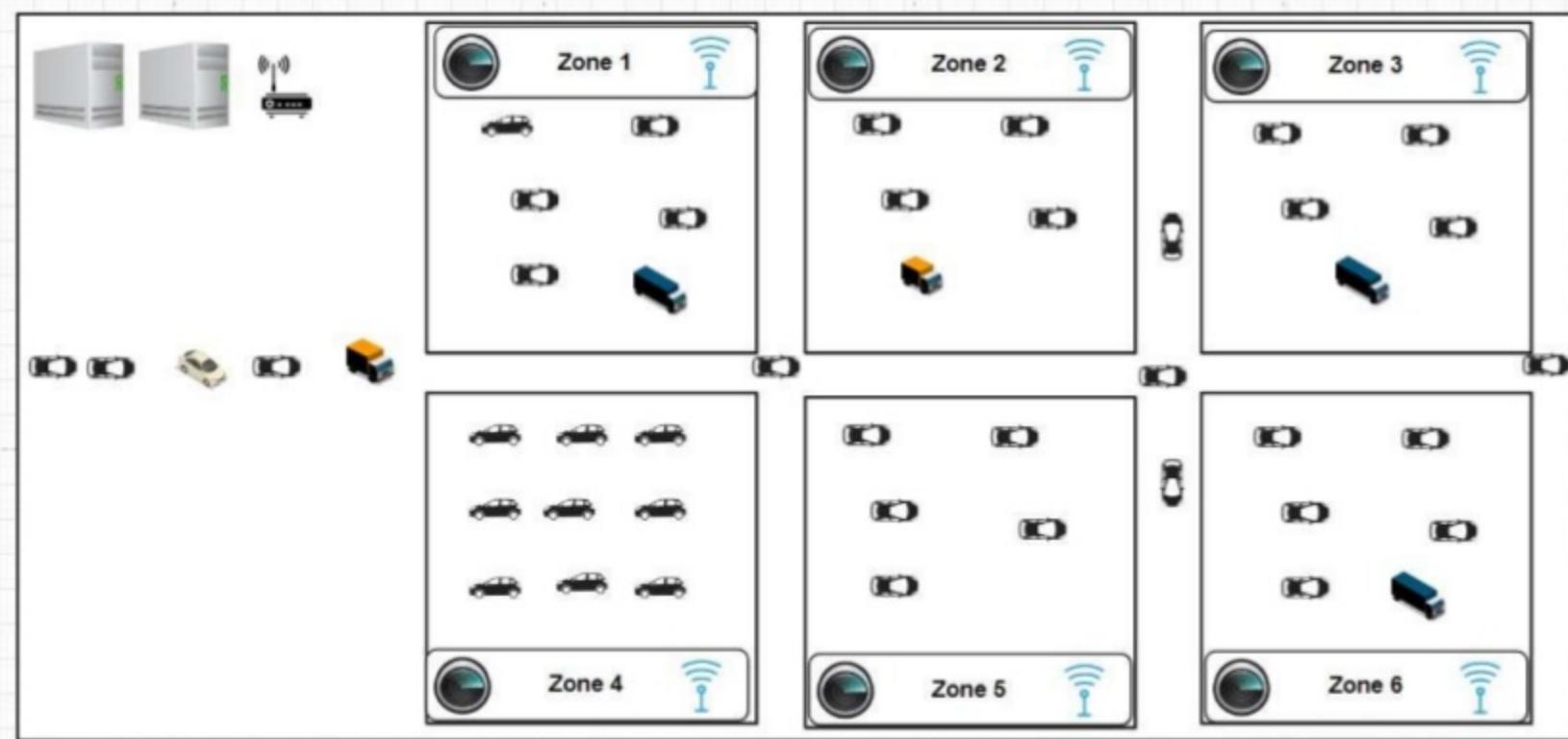
Facility Level

Plant Level

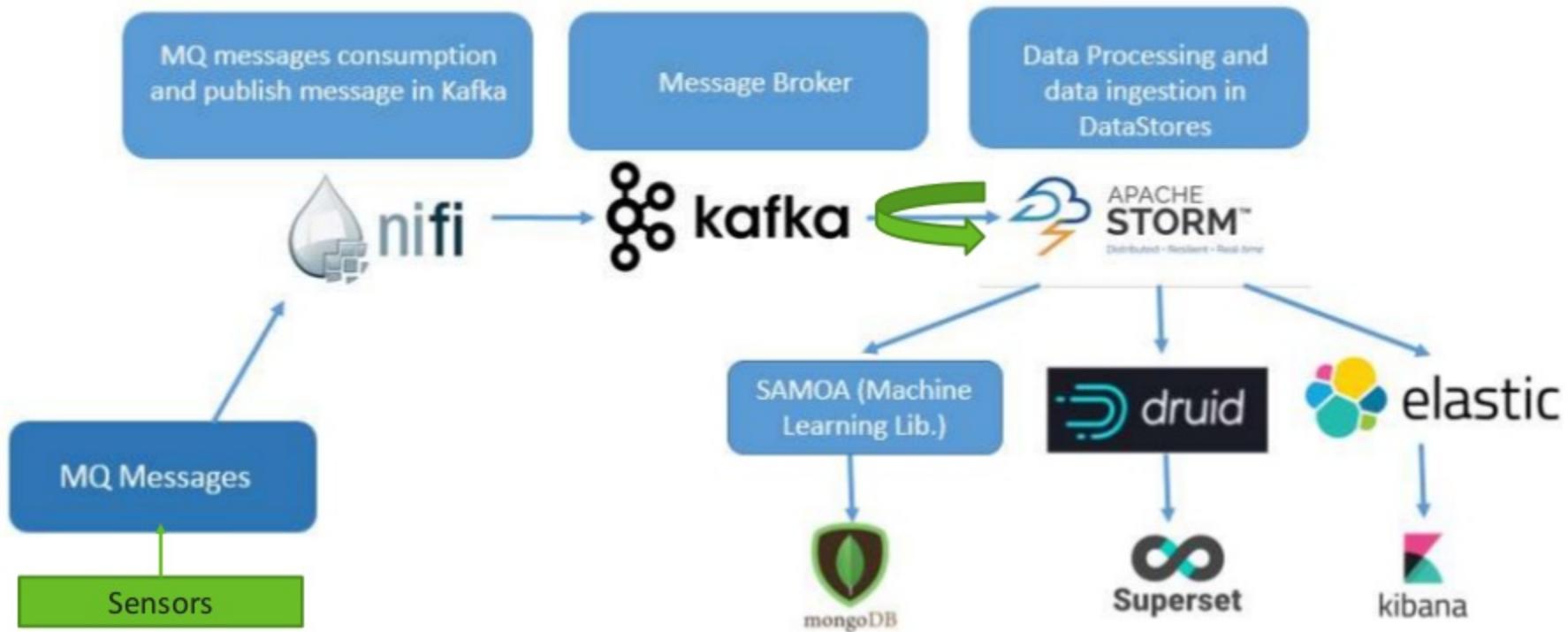
Corporate Level



# Overview of Plant Assembly Factory UC

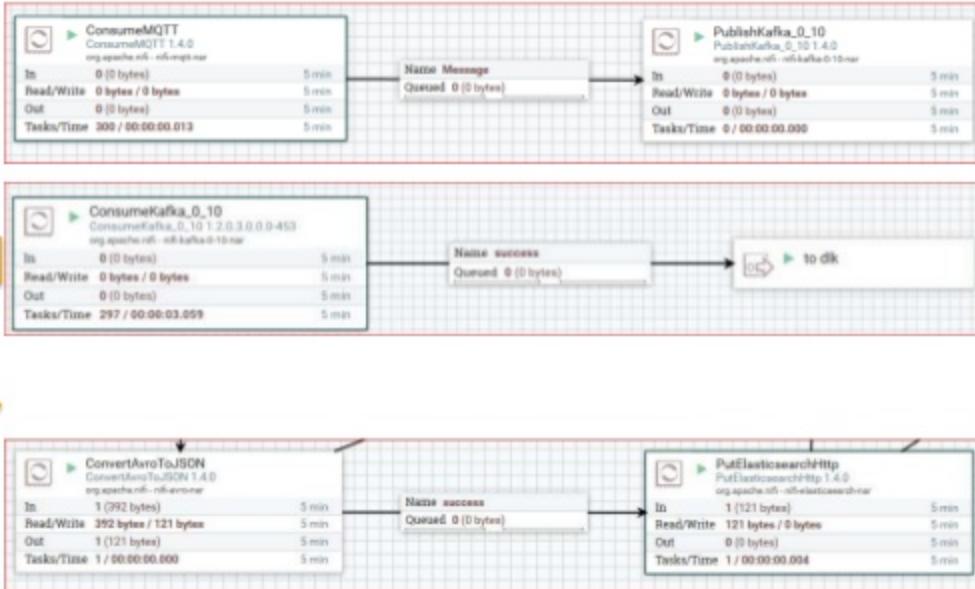


# Full HDF use case (NiFi, Kafka and Storm)



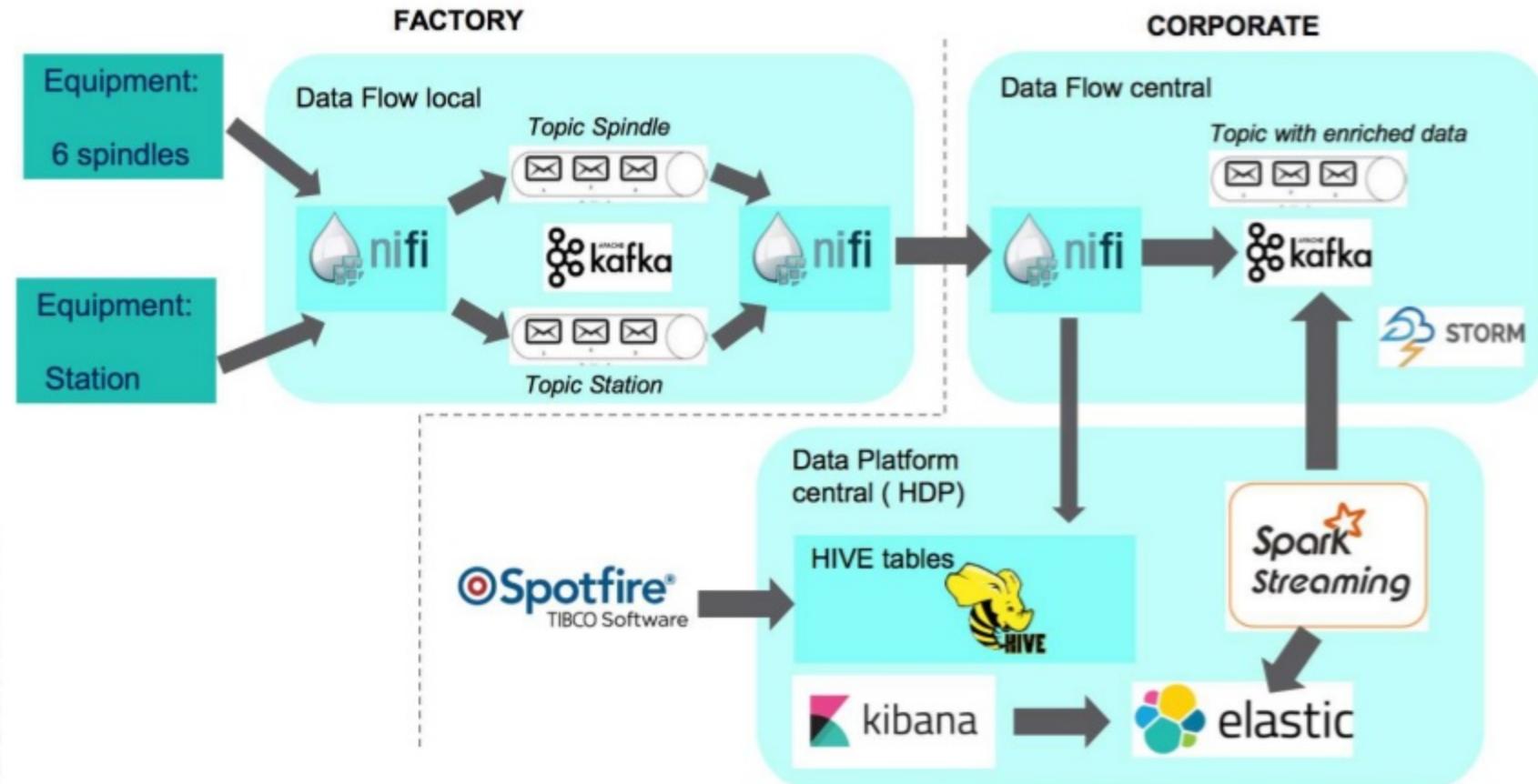
# Connected plants

Site  
To  
Site

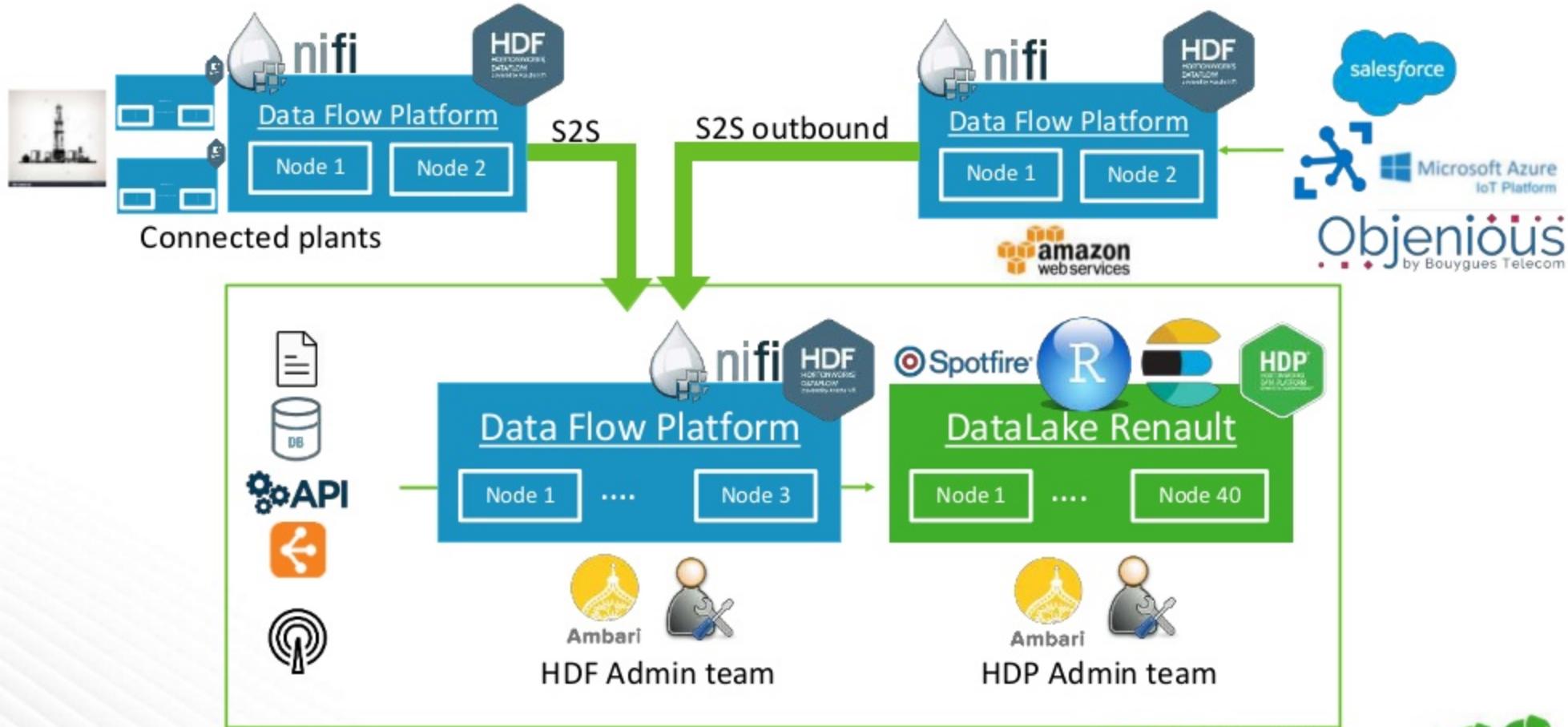


Demo with MQTTTool from  
Hand to Data Lake !  
Handy, MQTT Broker, NiFi  
ConsumeMQTT to Kafka,  
NIFI consume Kafka to  
ElasticSearch and Hive...

# Screwing tools valladolid data analysis



# Architecture + Connected plants



# Best practices for running NiFi in production

GROUPE RENAULT

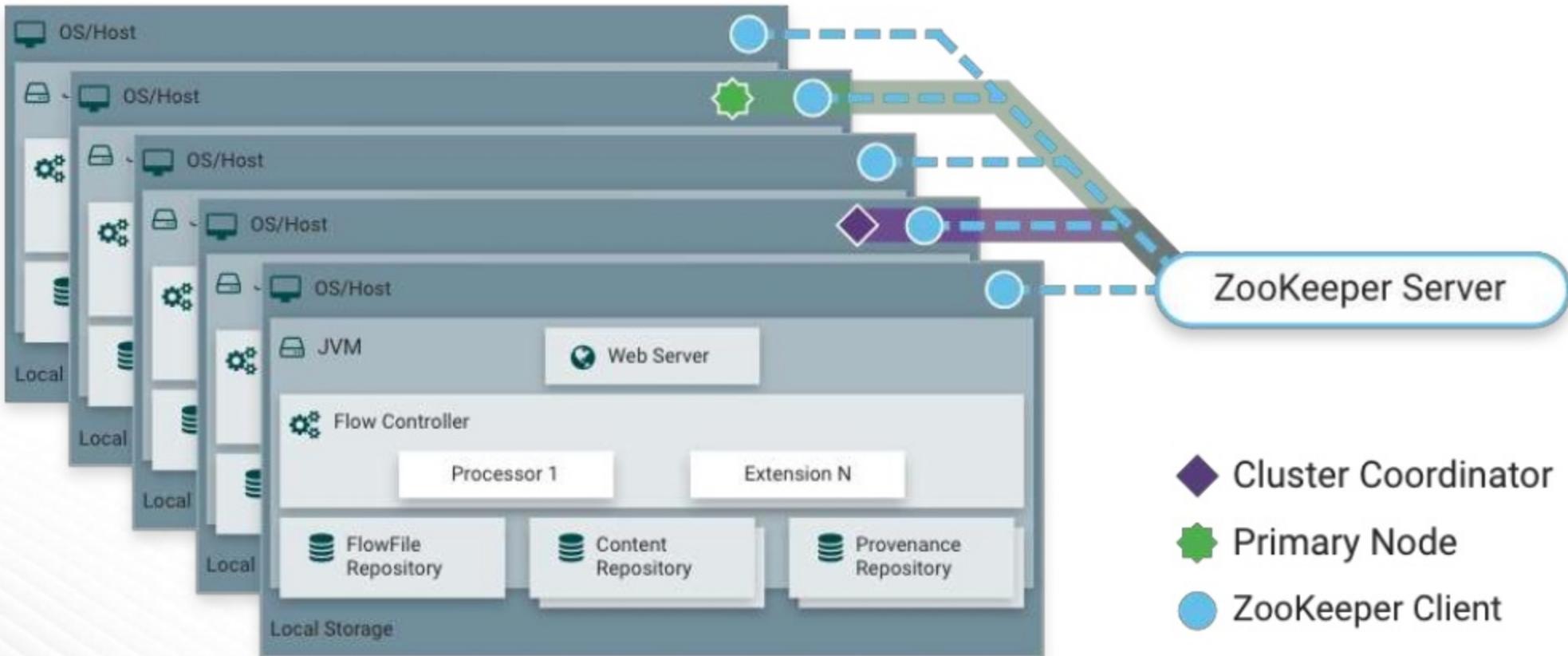


# Architecture & sizing

GROUPE RENAULT



# Typical NiFi Production Cluster Logical View



## Sizing considerations

- There are baselines for NiFi sizing but NiFi is like a “programming language” !!
- NiFi resources usage depends on used processors but it's always IO intensive
  - Use different disks / volumes for the three repositories : flow file, content & provenance
  - For content repository, it's recommended to have multiple mount points
  - For content and provenance repositories, SSD can provide the best performances
  - Heap sizing depends on the use case and used processors
  - Depending on the workload, we can scale vertically or horizontally
- Default settings are only for getting started. Tune based on your use case.
  - Thread pool size : Maximum Timer Driven Thread Count
  - Timeouts: nifi.cluster.node.connection/read.timeout, nifi.zookeeper.connect/session.timeout
  - Pluggable modules: WAL Provenance Repository

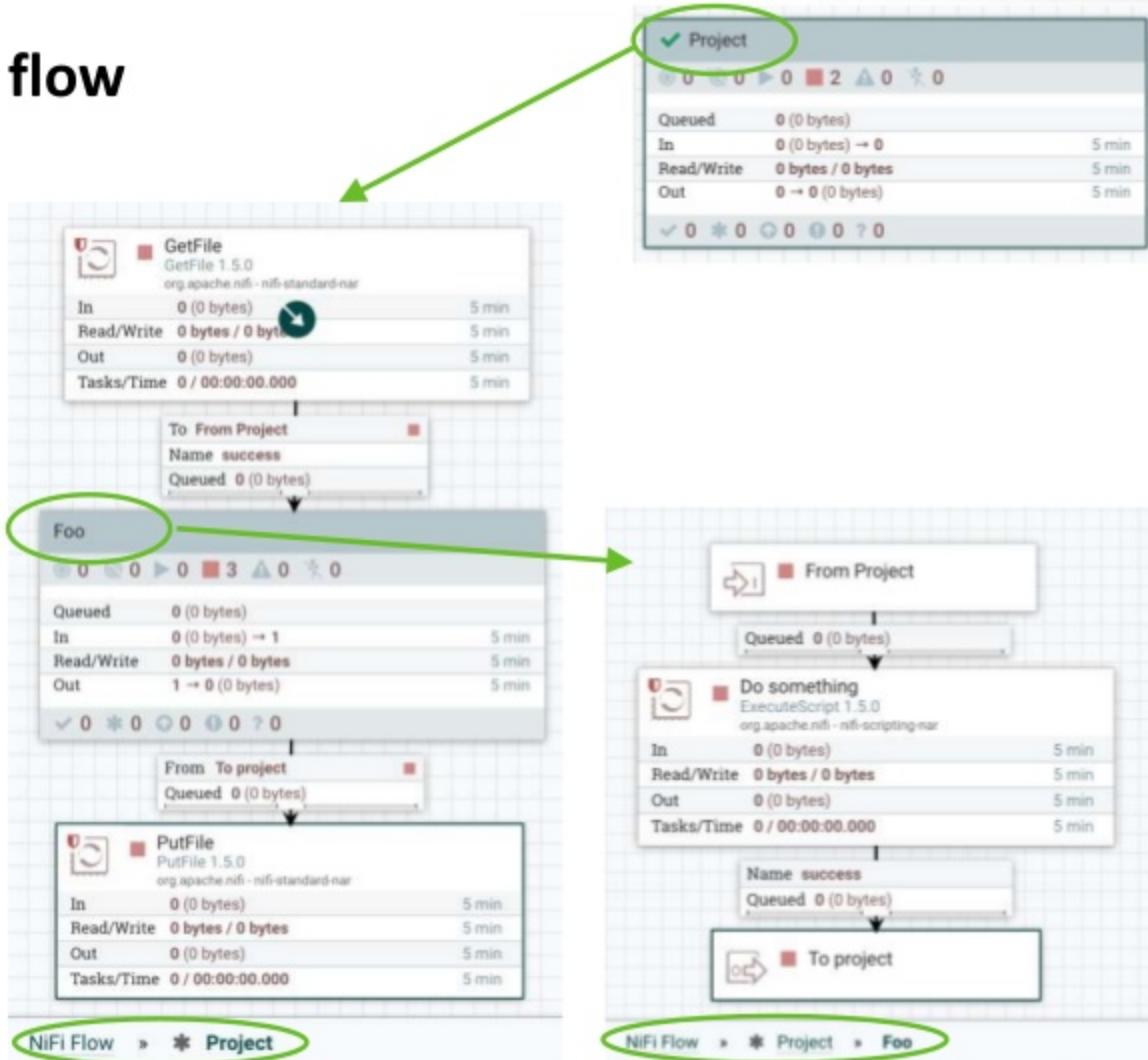
# Development & deployment

GROUPE RENAULT



# Let's consider a simple data flow

```
Public class Project {  
  
    private int foo(int x) {  
        //do something with x -> y  
        return y;  
    }  
  
    public static void main(String [] args) {  
        //some code  
        try {  
            BufferedReader bufferRead = new  
            BufferedReader ...;  
            String data =bufferRead.readLine()  
            data = foo(data);  
            System.out.println(data);  
        } catch ...  
    }  
}
```



# Flow development = software development



## Programming language

- Integrated Development Environment
- Algorithm, code, instructions
- Functions (arguments, results)
- Libraries
- ...

## Apache NiFi

- Apache NiFi UI
- Flows, processors, funnels
- Process groups (input/out ports)
- Templates
- ...

# General guideline

## Principle

- Use separate environments
- No flows at root level: use a PG per department, BL, project, etc
- Break your flow into process groups
- Use a naming convention, use comments (labels, comments)
- Use variable when possible
- Organize your projects into three PGs: ingestion, test & monitoring

## Benefits

- Performance, security and SLA
- Easy to secure, everything in NiFi is attached to a PG, heritage
- Easy to test, update & version
- Very useful for development/monitoring. Ideally use unique names
- Promotion (dev > test > prod), update
- NiFi can generate data for TDD (Test Driven Dev), can collect/parse logs for BAM (Business App Monitoring)

# NiFi organization example

The image shows the NiFi interface with three main sections: Done, Doing, and To Do.

**Done:**

- MKT05\_TwitterMonitoring:** Status: ✓, 0 / 0 / 0 / 0 / 3 / 1 / 0. Metrics: Queued: 0 (0 bytes), In: 0 (0 bytes) → 0, Read/Write: 0 bytes / 0 bytes, Out: 0 → 0 (0 bytes). Last updated: 5 min.
- FSI15\_Customer360:** Status: ✓, 0 / 0 / 0 / 0 / 50 / 7 / 0. Metrics: Queued: 0 (0 bytes), In: 0 (0 bytes) → 0, Read/Write: 0 bytes / 0 bytes, Out: 0 → 0 (0 bytes). Last updated: 5 min.
- R&D01\_AttritionScore:** Status: ✓, 2 / \* 1 / 0 / 0 / 0 / 0 / 0. Metrics: Queued: 0 (0 bytes), In: 0 (0 bytes) → 0, Read/Write: 0 bytes / 0 bytes, Out: 0 → 0 (0 bytes). Last updated: 5 min.

**Doing:**

- FSI01\_FraudDetection:** Status: \*, 0 / 0 / 0 / 0 / 5 / 0 / 0. Metrics: Queued: 0 (0 bytes), In: 0 (0 bytes) → ✎, Read/Write: 0 bytes / 0 bytes, Out: 0 → 0 (0 bytes). Last updated: 5 min.

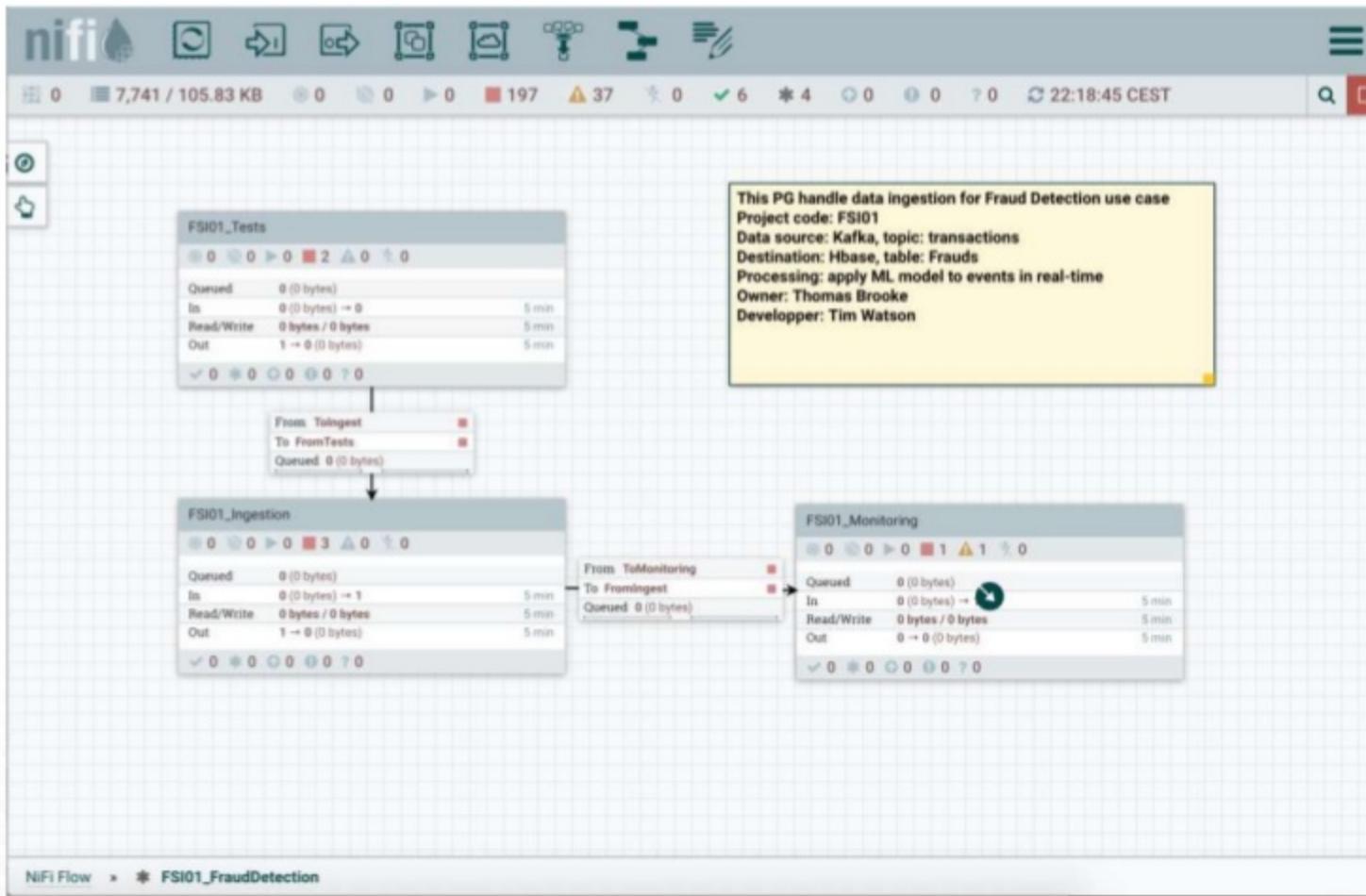
**To Do:**

- IOT01\_DABMonitoring:** Status: 0 / 0 / 0 / 0 / 0 / 0 / 0. Metrics: Queued: 0 (0 bytes), In: 0 (0 bytes) → 0, Read/Write: 0 bytes / 0 bytes, Out: 0 → 0 (0 bytes). Last updated: 5 min.

**Header:** nifi 0 7,741 / 105.83 KB 0 0 0 196 36 0 6 4 0 0 0 22:07:14 CEST

**Bottom:** NiFi Flow

# NiFi organization example



# Know & use NiFi Design Patterns

## Fan IN/OUT

List & Fetch

## Attributes promotion

Extract, Update Attribute

## Throttling

ControlRate, expiration

## Funneling

RPG, RouteOnAttribute

## Error loops

Relations, Counters

etc

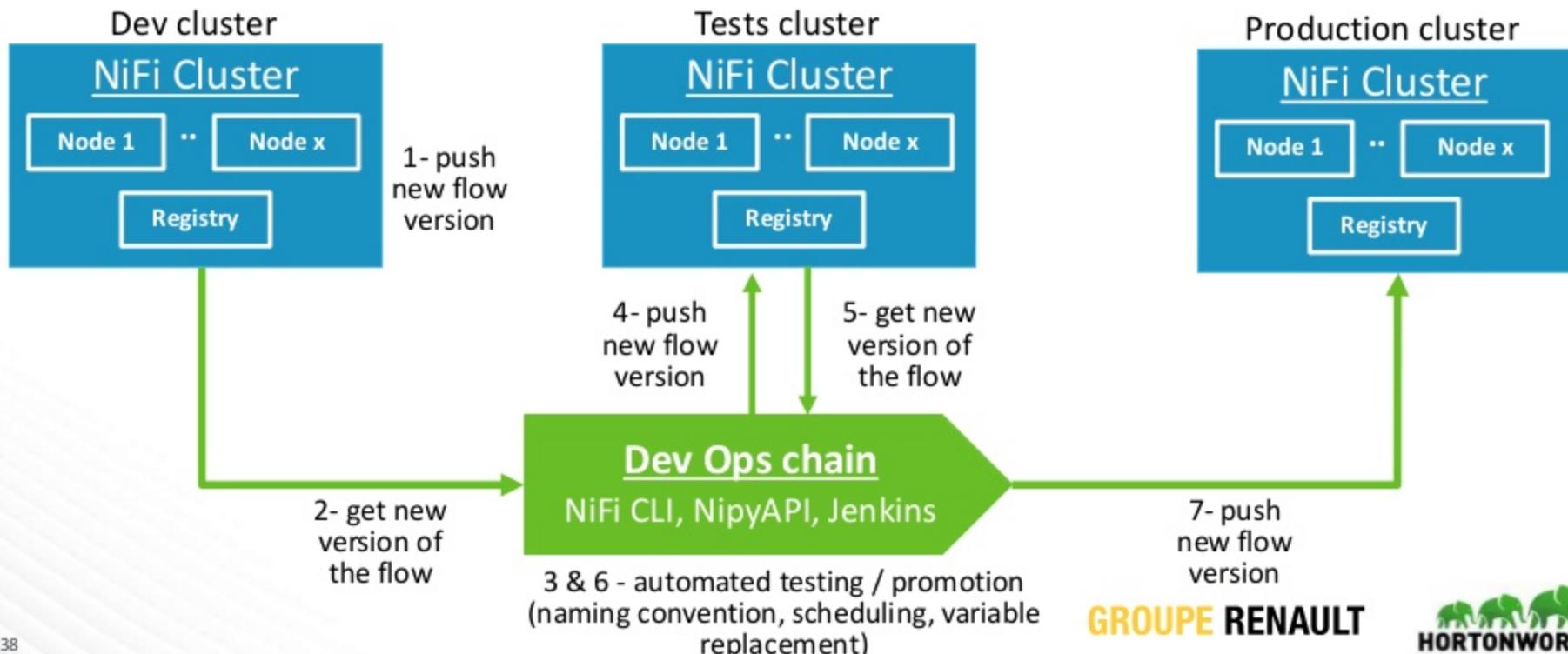
# FDLC: Flow Development LifeCycle



## Forget Duplicating Local Changes: Apache NiFi and the Flow Development Lifecycle (FDLC)

TECHNICAL • Data Processing and Warehousing

Thursday, April 19  
4:00 PM - 4:40 PM  
Room II



# Monitoring

GROUPE RENAULT



# NiFi Monitoring

## Service monitoring

- Is NiFi service running correctly?
- Monitor global system metrics such as threads, JVM, disk, etc
- Monitor global flow metrics such as number of flow files sent, received or queued, processors stopped, etc
- Solutions
  - NiFi UI
  - Reporting tasks
  - Ambari
  - Grafana

## Applications (Flow) monitoring

- Are a particular flow running correctly?
- Monitor per application (flow, PG, processor) metrics such as number of flow files, data size, queues, back pressure, etc
- Solution
  - S2S Reporting tasks
  - Custom flow developments (integrate monitoring and reporting in the application logic)

# NiFi UI for service monitoring

**Processor Type**

**Status Indicator**

**Processor Name**

**Bulletin Indicator**

**Active Tasks**

**5-Minute Statistics**

**Component Counts**

**Name**

**Process Group ABC**

**Active Tasks**

**Bulletin Indicator**

**5-Minute Statistics**

**Version State Counts**

**Hover over to see Comments**

**Remote Instance Name**

**Transmission Status**

**NiFi Flow**

**Secure Indicator**

**Remote Instance URL**

**5-Minute Statistics**

**Comments**

**Last Refresh Time**

**NiFi Summary**

**Component Tabs**

PROCESSORS	INPUT PORTS	OUTPUT PORTS	REMOTE PROCESS GROUPS	CONNECTIONS	PROCESS GROUPS	
Displaying 60 of 60						
Filter by name		View: Single node Cluster				
Name	Type	Run Status	In / Size 5 min	Read / Write 5 min	Out / Size 5 min	Tasks / Time 5 min
Base64EncodeCo...	Base64EncodeCo...	Invalid	0 (0 bytes)	0 bytes / 0 bytes	0 (0 bytes)	0 / 00:00:00.000
Capture Network...	ExecuteProcess	Running (5)	0 (0 bytes)	0 bytes / 245.97 MB	2,883 (2)	Go To Processor
Check if Dataset ...	IdentifyMimeType	Running	0 (0 bytes)	0 bytes / 0 bytes	0 (0 bytes)	0 / 00:00:00.000
Decompress Zip...	ContentAttribute	Running	0 (0 bytes)	0 bytes / 0 bytes	0 (0 bytes)	0 / 00:00:00.000
Empty comments...	ReplaceText	Running (2)	11,741,700 (13 GB)	0 bytes / 0 bytes	0 (0 bytes)	Show History
EvaluateJsonPath	EvaluateJsonPath	Running	0 (0 bytes)	0 bytes / 0 bytes	0 (0 bytes)	0 / 00:00:00.000
EvaluateJsonPath	EvaluateJsonPath	Running (4)	11,742,510 (13 GB)	13 GB / 0 bytes	11,742,510	Show Cluster Details
Extract File ID	UpdateAttribute	Running	0 (0 bytes)	0 bytes / 0 bytes	0 (0 bytes)	0 / 00:00:00.000
Extract File ID	UpdateAttribute	Running	0 (0 bytes)	0 bytes / 0 bytes	0 (0 bytes)	0 / 00:00:00.000
ExtractText	ExtractText	Running	1,498,914 (244.45...)	244.45 MB / 0 bytes	1,496,995 (244.25...)	1,500,692 / 00:02:4...

Last updated: 17:45:24 UTC

system diagnostics

Status History

Last updated: 22:46:13 UTC

Bytes Written (3 ...)

Min / Max / Mean  
0.00 bytes / 76.78 MB / 35.89 MB

NiFi

Nodes

Min / Max / Mean  
0.00 bytes / 76.78 MB / 35.89 MB

00:04:00:00

CLOSE

# Tools available for service monitoring

- Bootstrap notifier: send notification when the NiFi starts, stops or died unexpectedly
  - Email/HTTP notification services
- Use reporting tasks: export metrics to your monitoring solution
  - AmbariReportingTask (global, process group)
  - MonitorDiskUsage (Flowfile, content, provenance repositories)
  - MonitorMemory
- Also, monitor inactivity
  - NiFi has a built-in MonitorActivity processor
  - To be used with the S2SBulletinReportingTask
  - You can use InvokeHTTP to call the reporting Rest API

Add Reporting Task

Source	Type	Version	Tags
all groups	Type		
ambari metrics	AmbariReportingTask	1.5.0	ambari, metrics, reporting
garbage collection, gc, garbage	ControllerStatusReportingTa...	1.5.0	stats, log
heap, jvm, jython, log, ha...	DataDogReportingTask	1.5.0	datadog, metrics, reporting
host, memory, metrics	MetricReportingTask	1.5.0	metrics, reporting
monitor, monitoring, python	MonitorDiskUsage	1.5.0	disk, repo, warning, storage, me...
monitor, memory	MonitorMemory	1.5.0	jvm, memory, warning, monitor, ...
reporting	ScriptedReportingTask	1.5.0	log, metrics, provenance, python, ja...
restricted, script, site	SiteToSiteBulletinReportingT...	1.5.0	site, restricted, bulletin, site to s...
site to site, stats	SiteToSiteProvenanceReport...	1.5.0	lineage, site, provenance, restrict...
storage, warning	SiteToSiteStatusReportingTa...	1.5.0	site, metrics, history, status, sit...
	StandardGangliaReporter	1.5.0	stats, ganglia

AmbariReportingTask 1.5.0 - org.apache.nifi - nifi-ambari-nar  
Publishes metrics from NiFi to Ambari Metrics Service (AMS). Due to how the Ambari Metrics Service works, this reporting task should be scheduled to run every 60 seconds. Each iteration it will send the metrics from the previous iteration, and calculate the current metrics to be sent on next iteration. Scheduling this reporting task at a frequency other than 60 seconds may produce a...

CANCEL ADD

# How to achieve granular monitoring?

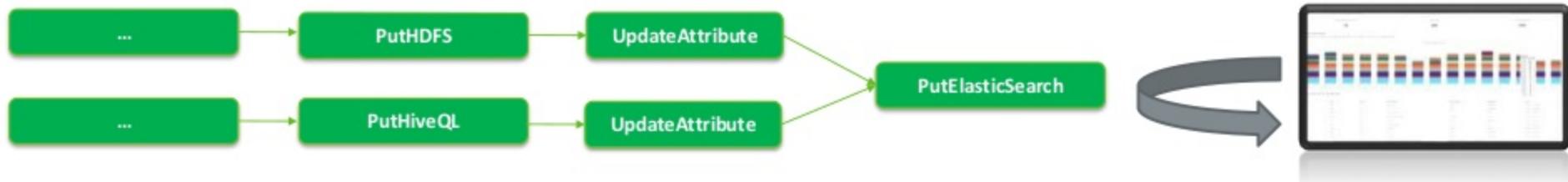
## Monitoring Driven Development (MDD)

- Integrate your monitoring logic in the flow design
  - Count data (lines, tables, events, etc)
  - Extract business metadata from data (table names, project name, source directory, etc)
- Handle different types of errors (connection, format, schema, etc)
- Send extracted KPI to brokers, dashboards, API, files, etc

## S2S reporting tasks

- NIFIception: NiFi can export metrics to another cluster or to itself
- Bulletins provide information on errors
- Status provide metrics on usage
- These reports become data and we can use the power of NiFi to extract our KPI
- **Naming convention is key**

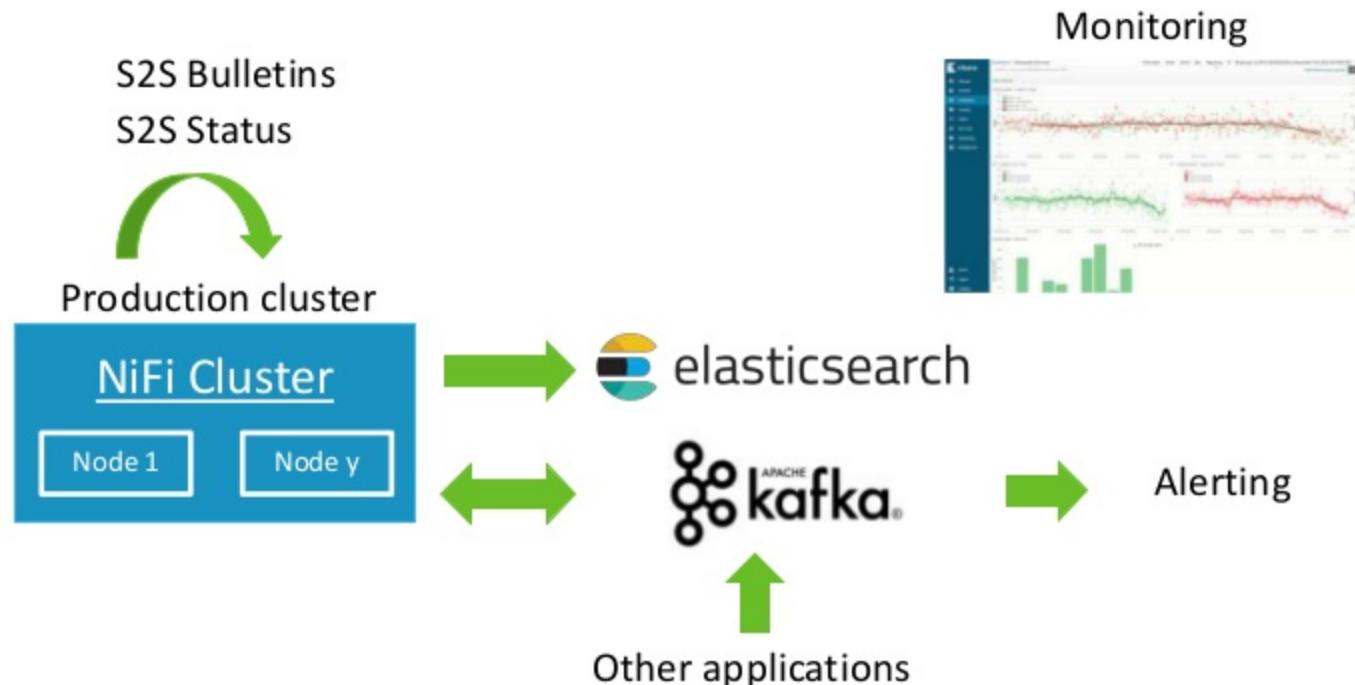
# Monitoring Driven Development



**GROUPE RENAULT**

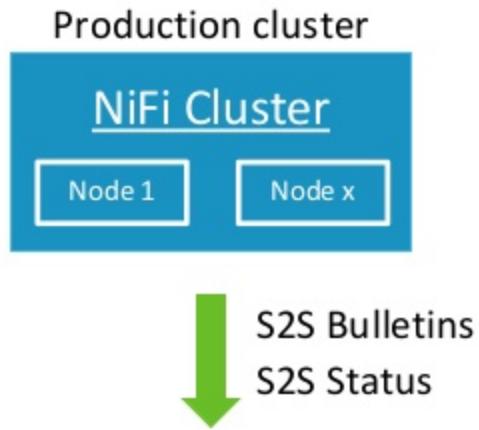
# NiFIception architecture

- Generate status and bulletins
- Send them through S2S to a local input port
- Ingest bulletin and status reports
- Parse reports (JSON) and extract useful KPI
- Send KPI to a dashboard tool (AMS, Elastic, etc)
- Use a broker for alerting (Kafka)
- Can ingest logs from other systems or application



# NiFIception architecture 2

- Ingest data
- Generate status and bulletins
- Send them through S2S to a remote cluster



Monitoring



- Ingest bulletin and status reports
- Parse reports (JSON) and extract useful KPI
- Send KPI to a dashboard tool (AMS, Elastic, etc)
- Use a broker for alerting (Kafka)
- Can ingest logs from other systems or application



# Lessons learnt at Renault

GROUPE RENAULT



## Recommendations from day to day life

- High availability means several weeks to validate before Go live
- Use Ranger authorizations instead of NIFI build in ACL management
- Too much « if then do else »: do the data flow and do not offload other processing solutions
- S2S : Minimize the number of RPG and self-RPG. Use attributes and routing.
- Put hive via Two redundant processors
- Discuss with DBA to better handle impacts (Views VS Tables, Open Sockets ..etc).
- Backup flowfile.xml.gz (users.xml et autorizations.xml) using NiFi itself.

## Recommendations from day to day life

- Success flag can be done through counters instead of InvokHTTP
- In case of CIFS, do not forget to use AUTO MOUNT for NFS client side on NiFi Servers.
- Check '0' size before transmitting into HDFS (especially for IoT use case).
- Configure a TIMER for when we use FAILURE redirections to avoid back-pressure scenario.
- Build separate clusters for separate use cases (Real Time with SSD, Batch, etc ...)
- CA PKI very useful for internal communications, no need to wait Security teams answers but need security skills (SSL) .

# Questions

GROUPE RENAULT

