

Centre de Calcul
de l'Institut National de Physique Nucléaire
et de Physique des Particules

Running event-driven workflows with dCache storage events

Bastien Gounon

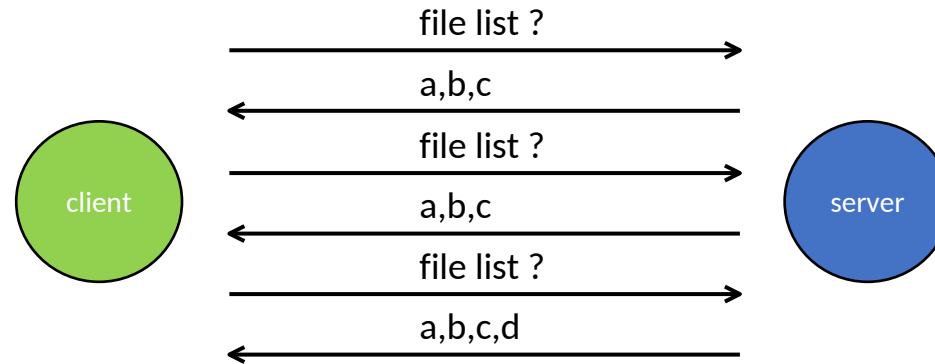
dCache

a system for **storing and retrieving huge amounts of data**
supports many access protocols
popular with science grid use cases

Kafka

an **event streaming/message queuing platform**
producer/consumer model
fast, scalable

What is **polling**, the usual way we detect changes ?

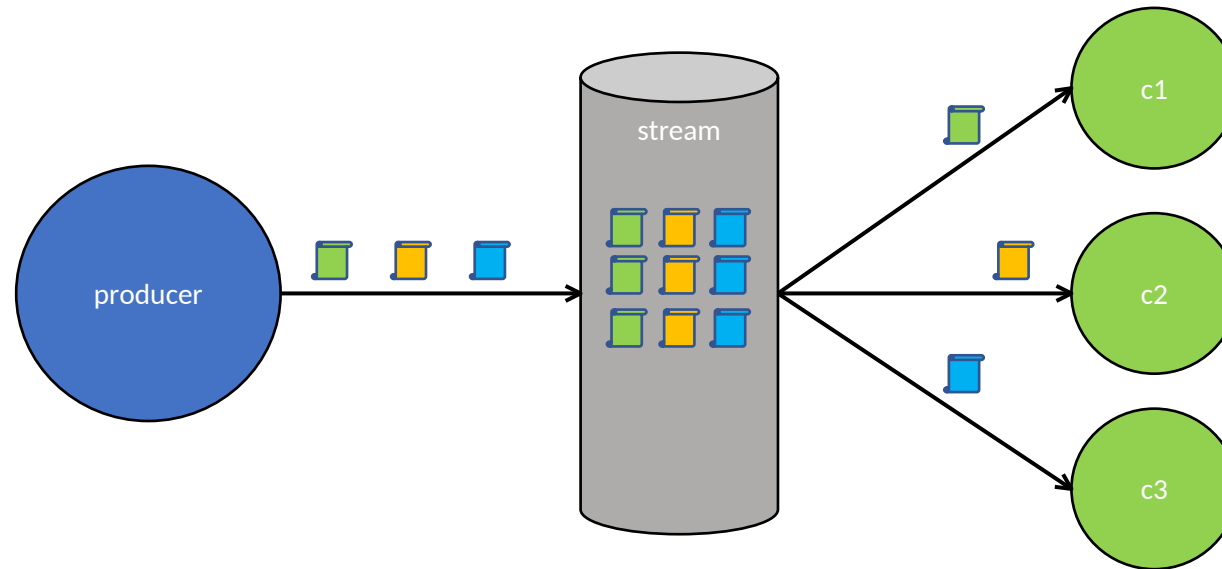


In a traditional **polling** approach, **the client questions the server** at regular intervals and reacts to its response.

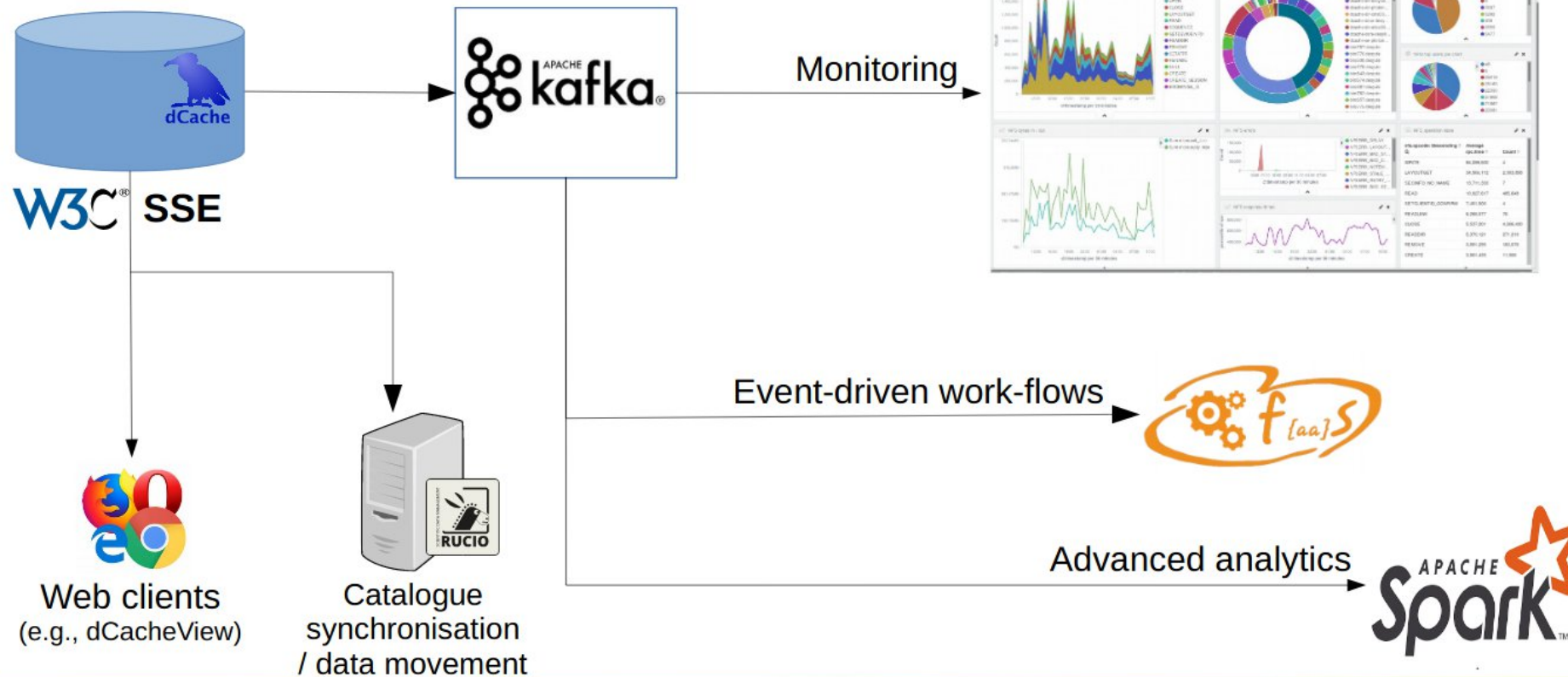
In **event-driven** architecture, the **producer** (= server) is the initiator.

Consumer: continuously listens for incoming messages in the stream

Producer: publishes an event in the stream when something happens



Using storage events



Based on slide from Marina Sahakyan

dCache storage events | Paul Millar | 2020-01-27 | Page 11

Context: december 2019, a campaign to simulate LSST images in the European grid. Hundreds of terabytes and hundreds of thousands of files will be generated.

Problem 1: that data must then be processed, but our grid storage (dCache) is not available on worker nodes via POSIX for batch processing

Problem 2: the size of the LSST pool in dCache is limited, and we cannot store the complete dataset

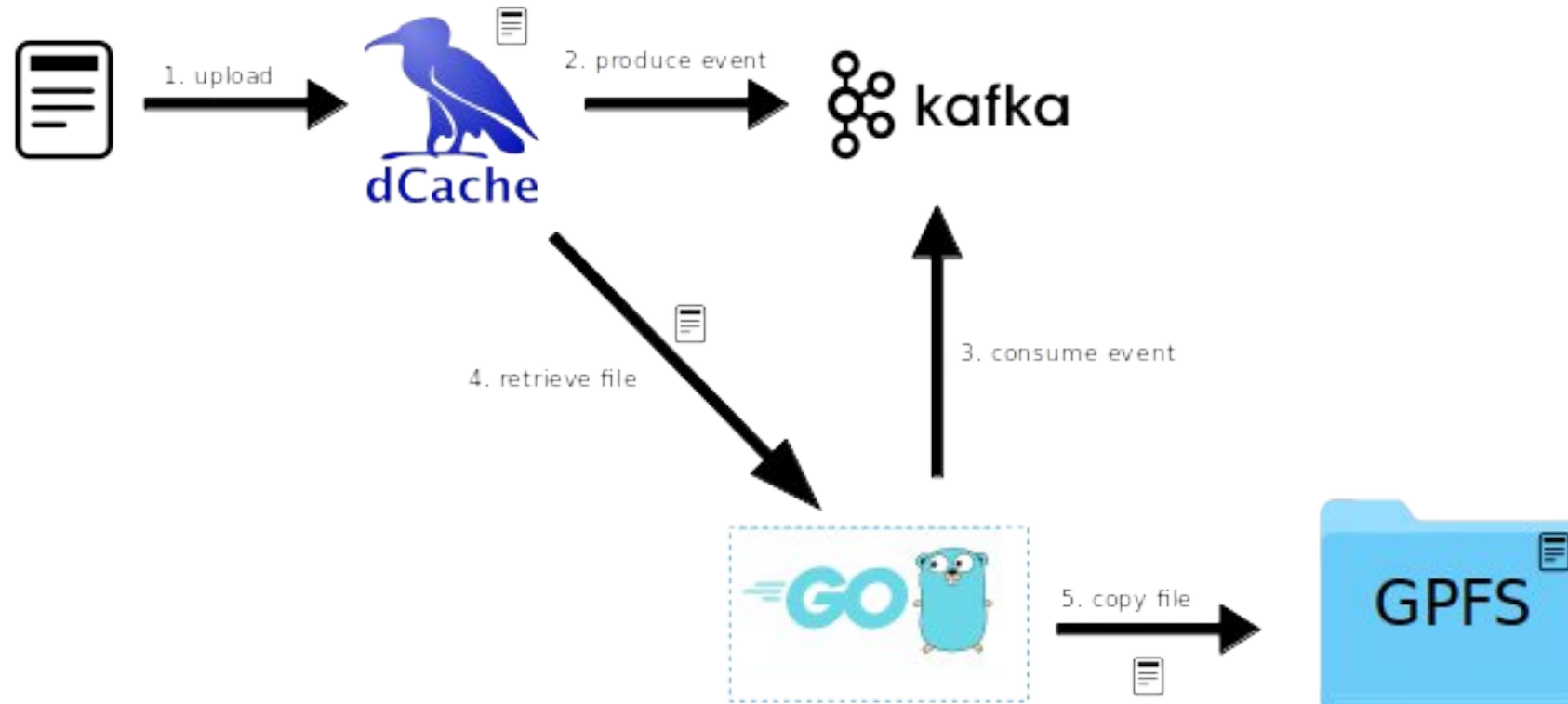
Solution: copy incoming data to our GPFS filesystem before processing it

Traditional approach:

- **manually** and **periodically** copy the data from dCache to GPFS
 - make sure not to **copy incomplete files** to GPFS (pause and wait for those running to finish ?)
 - empty the dCache pool to **free space for incoming data**
- => requires **human intervention** at each step
- => difficult to estimate the duration of the bulk copy operation for all files

Event-driven approach:

- automatically copy files to GPFS as soon as they are uploaded to dCache
 - **short, atomic copy operations**, happening **file by file**
- => does not require human attendance



Implementation details:

written in **Go**, source code available: <https://gitlab.in2p3.fr/bastien.gounon/dcalqr>

runs as a **daemon**, from a **single binary** executable

simple **toml configuration** file

x509 proxy authentication

automatic retry

checksum validation

metrics logging (throughput, file size)

configurable number of maximum parallel transfers

consumer groups (group **offset tracking** + built-in **scalability**)


```
[kafka]
topic = "billing"
servers = [ "10.0.0.10:9092", "10.0.0.11:9092", "10.0.0.12:9092", ]
consumergroup = "imsimprod001"

[dcache]
url = "https://ccdcaccli236.in2p3.fr:2880/lsst"
pool = "/pnfs/in2p3.fr/data/lsst"
folder = "/lsst/user/b/bastien.gounon"
retryseconds = 1
retrytimes = 10
maxdownloads = 10
timeout = 600

[local]
proxy = "/tmp/x509_proxy"
capath = "/etc/grid-security/certificates"
downloadfolder = "/sps/lssttest/datasets/desc/DC2/Run2.2i/sim/input"
permissions = '2750'
failedtasksfile = "failed.csv"
successfultasksfile = "success.csv"
```

Feedback

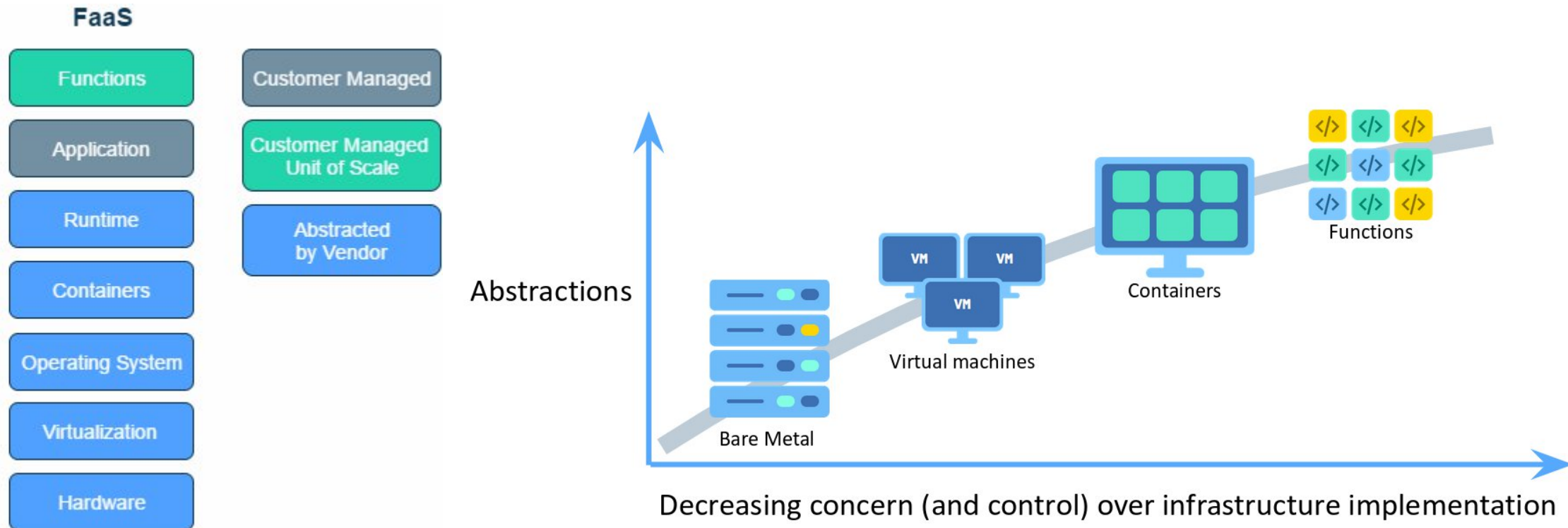
- hours saved in operations instead spent on development effort, but we also built valuable experience working with storage events
- successfully used to copy **~200TB of data and ~1M files** over the campaign duration
- dCache and Kafka both demonstrated **very good stability** and consistency
- failure rate < 1%

Limitations:

- small caveat when using SRM upload to dCache (issue [#5285](#))
- requests would sometimes hang for a few minutes, most likely due to local issues
- **consumers scaling not tested** since dCache does not support **multi-partitions topics**

Prospective: event-driven FaaS workflows ?

Function-as-a-Service is a computing model in which simple **functions** are run in response to **triggers** (usually events)



Prospective: event-driven FaaS workflows ?

For example, it makes it possible to **automatically**:

- uncompress and/or copy incoming data
- ingest new files into catalogs
- run **batch jobs** or data processing pipelines

...

Many FaaS platforms are already available, in and out of the cloud:



AWS - Lambda



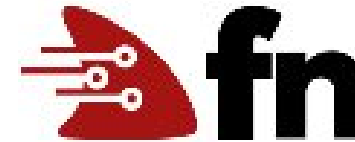
Microsoft Azure
Microsoft Function



Google Cloud
Google Function



Apache Openwhisk



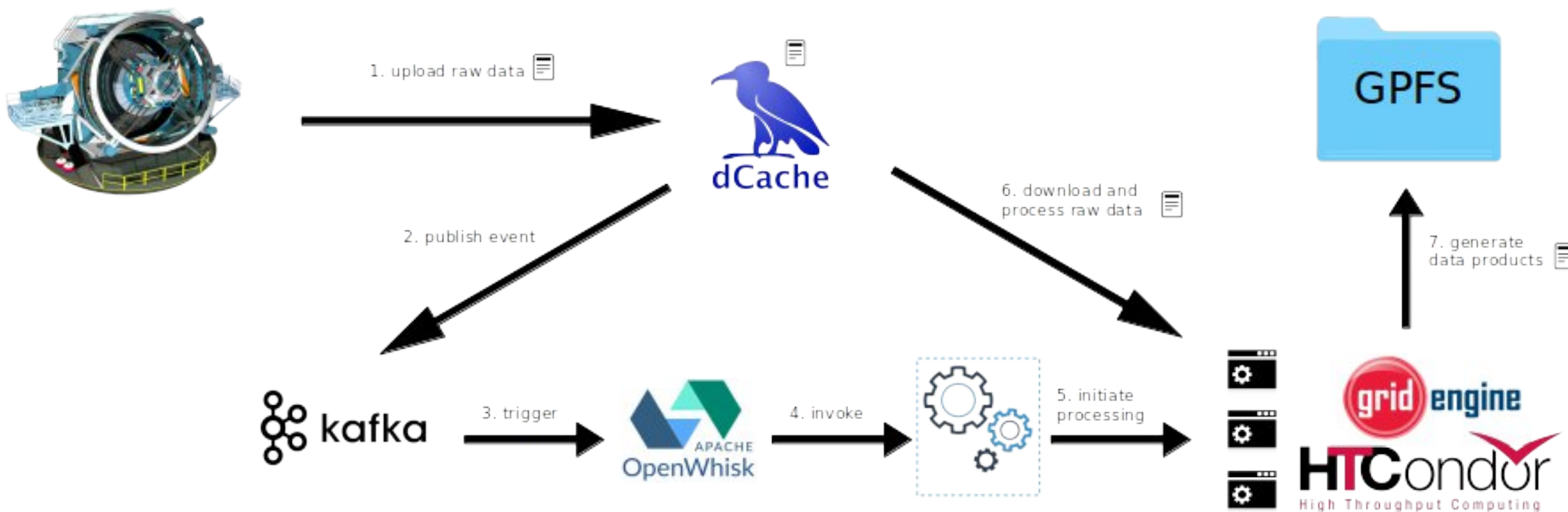
Fn Project



OpenFaaS

Prospective: event-driven FaaS workflows ?

Illustration of what a fully-automated data processing pipeline may look like:



Thank you!



Questions?

contents of a dCache storage event in Kafka:

```
{
  "date": "2019-07-09T11:11:57.024+02:00",
  "owner": "/O=GRID-FR/C=FR/O=CNRS/OU=CC-IN2P3/CN=Adrien
Georget",
  "msgType": "request",
  "clientChain": "134.158.240.106",
  "mappedGID": 239,
  "cellName": "WebDAV-ccdcalitest12",
  "session": "door:WebDAV-ccdcalitest12@webdav-
ccdcalitest12Domain:AAWNO/PLJUg:1562663516332000",
  "subject": [
    "UsernamePrincipal[ageorget]",
    "UidPrincipal[3915]",
    "LoAPrincipal[IGTF-AP:Classic]",
    "EntityDefinitionPrincipal[Person]",
    "FQANPrincipal[/dteam/NGI_FRANCE/sites/IN2P3-CC]",
    "GidPrincipal[239,primary]",
    "/O=GRID-FR/C=FR/O=CNRS/OU=CC-IN2P3/CN=Adrien Georget",
    "FQANPrincipal[/dteam/NGI_FRANCE]",
    "FQANPrincipal[/dteam/NGI_FRANCE/sites]",
    "GroupNamePrincipal[logdteam,primary]",
    "Origin[134.158.240.106]",
    "EmailAddressPrincipal[adrien.georget@cc.in2p3.fr]",
    "FQANPrincipal[/dteam,primary]"
  ],
  "transferPath": "/pnfs/in2p3.fr/data/doma/testWebdav",
  "sessionDuration": 692,
  "storageInfo": "disk:doma@osm",
  "cellType": "door",
  "fileSize": 119936222,
  "mappedUID": 3915,
  "VERSION": "1.0",
  "queuingTime": 0,
  "cellDomain": "webdav-ccdcalitest12Domain",
  "client": "134.158.240.106",
  "pnfsid": "00004F0AF7BEAE5A4CBEBF0FBF036D01F4C4",
  "billingPath": "/pnfs/in2p3.fr/data/doma/testWebdav",
  "status": {
    "msg": "",
    "code": 0
  }
}
```

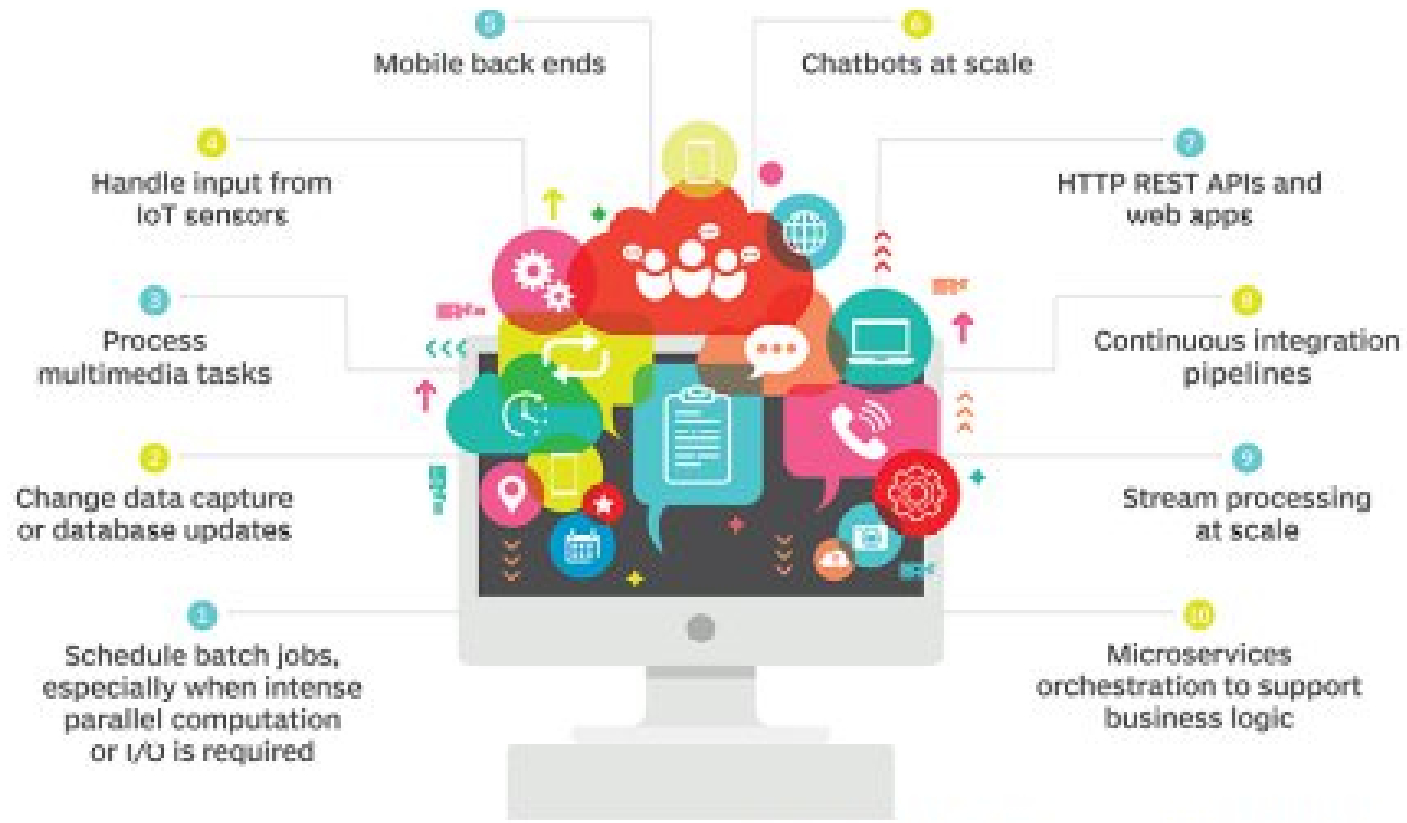
dcalqr logs excerpt:

```
2020/01/23 17:47:14 [UPLOAD] New file on partition 0 : /pnfs/in2p3.fr/data/lsst/lsst/user/j/james.perry/23003/23003691/fits_01188251_10.tar (f9c82897-0976-ed88-61d8-0db305308527)
2020/01/23 17:47:14 [CURRENTLY_RUNNING] 1 running tasks: [f9c82897-0976-ed88-61d8-0db305308527] (SYSTEM)
2020/01/23 17:47:15 [COPY_INIT] Creating destination file /sps/lssttest/datasets/desc/DC2/Run2.2i/sim/input/23003/23003691/fits_01188251_10.tar.part (f9c82897-0976-ed88-61d8-0db305308527)
2020/01/23 17:47:15 [COPY_START] Getting data from https://ccdcaccli236.in2p3.fr:2880/lsst/lsst/user/j/james.perry/23003/23003691/fits_01188251_10.tar (f9c82897-0976-ed88-61d8-0db305308527)
2020/01/23 17:47:15 [COPY_RUN] Copying data to /sps/lssttest/datasets/desc/DC2/Run2.2i/sim/input/23003/23003691/fits_01188251_10.tar.part (f9c82897-0976-ed88-61d8-0db305308527)
2020/01/23 17:47:16 [COPY_OVER] https://ccdcaccli236.in2p3.fr:2880/lsst/lsst/user/j/james.perry/23003/23003691/fits_01188251_10.tar to /sps/lssttest/datasets/desc/DC2/Run2.2i/sim/input/23003/23003691/fits_01188251_10.tar (checksum validation: OK) (f9c82897-0976-ed88-61d8-0db305308527)
2020/01/23 17:47:16 [COPY_REPORT] 217 MB copied in 379.387386ms, 4355.91 Mbps (f9c82897-0976-ed88-61d8-0db305308527)
```

Pros for function-as-a-service architecture:

- **Developer-friendly**
- **Focus on features**
- **Isolation**
- **Scalability**
- **Pay-per-use / No idle time: “If the events don’t happen, the workflow is not run and you do not have to pay for servers to host it.”**
- **Zero infrastructure management (in the cloud)**
- **Instantaneous deployment and update**

10 common uses for serverless platforms



source: <https://searchitoperations.techtarget.com/definition/serverless-computing>