Engineering APIs for Accelerator Controls Software

Bartek Urbaniec BE-CSS-CSA (*with special thanks to Anti Asko and Lukasz Burdzanowski*)

02/12/2021

https://indico.cern.ch/event/1054892

Agenda

- Introduction to APIs
- APIs in the Accelerator Control System
- APIs in more depth – what, why and how?
- Use-case study: CCDA, the Controls Configuration Data API
- Practical APIs:
  security, monitoring, alerting, tracing
- Operational experiences:
  performance, reliability, availability, testing, to cache or not to cache,…
- Outlook for Controls APIs

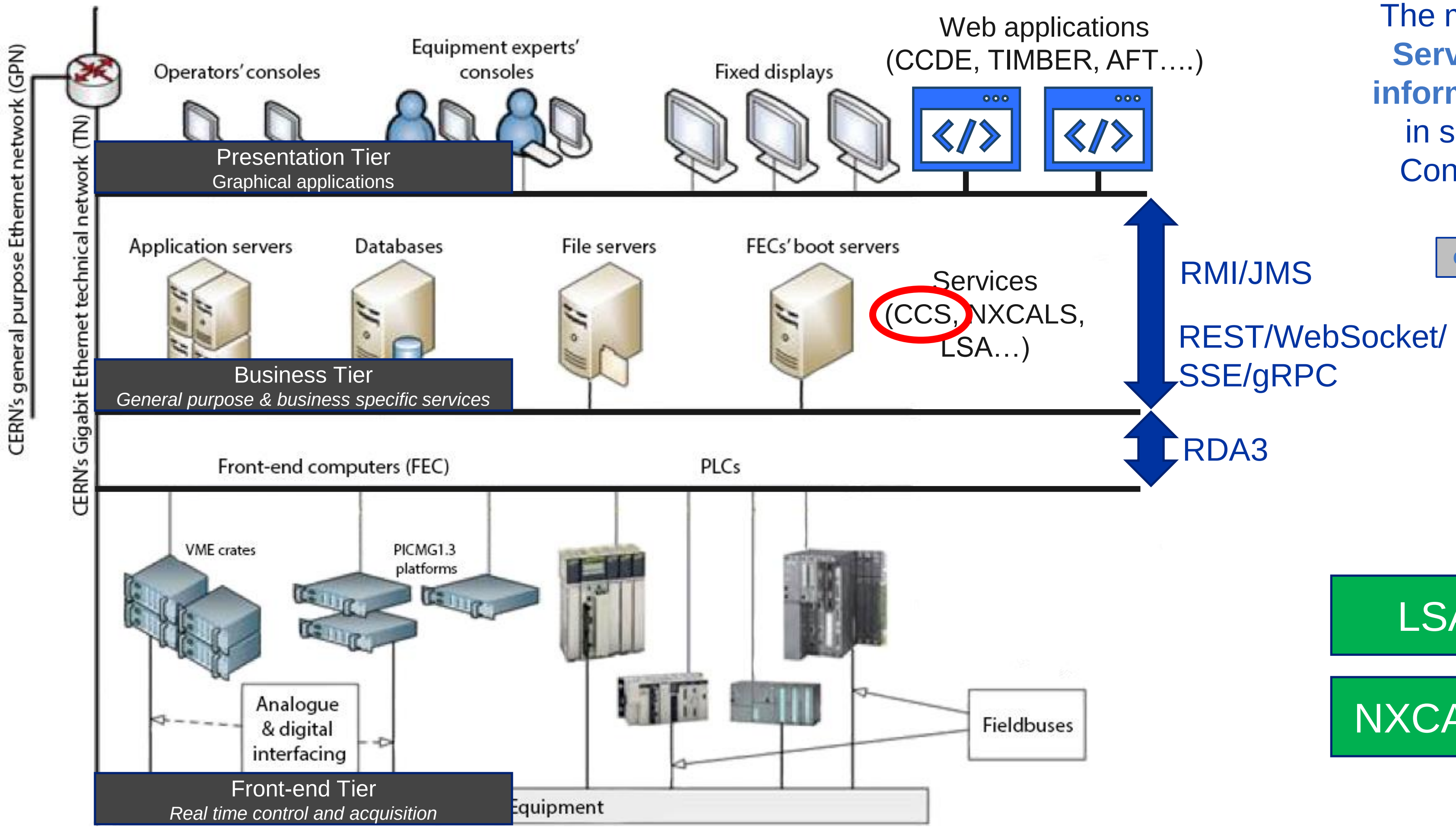
Introduction to APIs

Basic Introduction to APIs

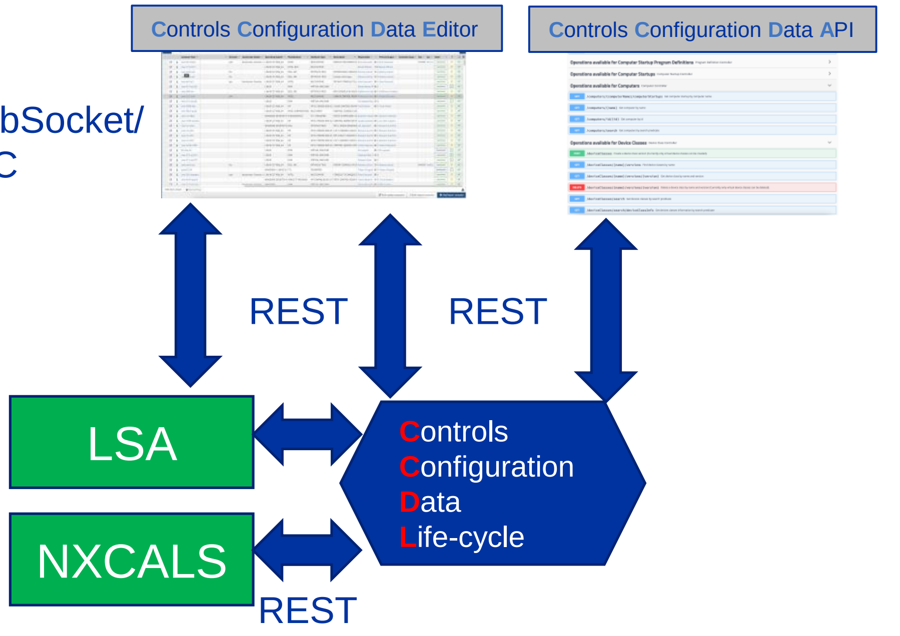
API stands for **A**pplication **P**rogramming **I**nterface

- An API is a connection between software components. It is a type of software interface, offering a service to other pieces of software.[wikipedia]
- APIs can be implemented using a variety of technologies e.g. RMI, WinCC OA, CORBA, OPC-UA, REST, SOAP, gRPC
- In the presentation I will mainly refer to REST APIs implemented in Java and used in the Accelerator Control System

APIs in the Accelerator Control System



The main purpose of the **Controls Configuration Service** (CCS) is to **unite and centralize** all the **information** relevant to the **Control systems** (CS) in such a way that integration between various Control sub-systems is consistent and efficient.

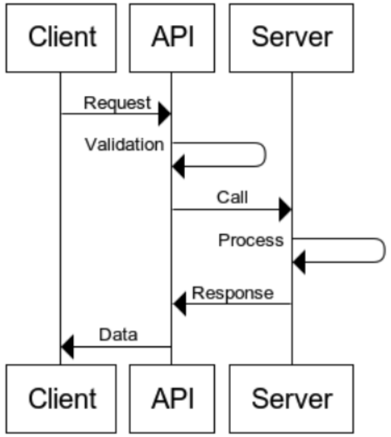


Controls REST-based APIs in practice

Controls **API**s are **sets of defined rules** that describe how software communicates with each another.
The API is an intermediary **layer,** between a Controls service (**Server**) and its **Clients** to exchange data & commands

Base client-server workflow:

1. A client initiates an API call - ***request***
2. If request is valid, the API makes a call to external program - ***server***
3. The server sends ***response*** to the API
4. The API transfers ***data*** to the client



The **API** is a **contract** between **server** and **client**

Why we need an API

Typing “**CERN**” in YouTube search will execute a query like below *YouTube.Search.list(‘**id,snippet**', {q: '**CERN**', maxResults: 25});*

OBD2 (On-board diagnostics) interface which allows to access car data. i.e. PID ***0D*** – car speed

- To **expose** system **data** and functionalites to clients
- To allow the clients to program specific **complex use cases** → “script” the service
- To enable and to improve **integration** between systems and services → “translate” data structure/etc.
- Internally, API can increase **quality of code** (and product) by breaking large **monoliths** into **smaller** functional services → low coupling, high-cohesion – OOD principles

Kinds of APIs

By Usage

- **Public API**
  open to *all* clients – any client (authenticated/authorised) may use it

- **Partner API**
  dedicated for *agreed* clients – available only to some clients, often via dedicated gateways

- **Internal API**
  not for *external* clients – inside internal network or between internal processes

- **Composite API (Proxy API)**
  used to combine several APIs into one

```
POST /order
{
  "order-request": [
    {
      "path": "/client”. "ref": "client", "body": {"name": "Bartek"}
    },
    {
      "path": "/order",
      "body": {"customer": "@{client.id}", order: {"name": "My new book"}}
    }
  ]
}
```

Typical kinds of APIs for the Web

By Technology

- **RPC** (Remote Procedure Call)
  request-response protocol (XML-RPC, JSON-RPC)
- **SOAP** (Simple Object Access Protocol)
  messaging protocol with XML as exchange format
- **REST** (REpresentational State Transfer)
  set of architectural constraints, not a protocol or a standard.
  - Client/server architecture
  - Stateless
  - Cacheable
  - Uniform interface
  - Layered system
- **gRPC*** (Google)
  based on HTTP2.0, uses Protocol Buffer to serialise data to binary format
  ** yes, technically gRPC generates stubs for any client, abstracting HTTP2.0*
- **GraphQL** (Facebook)
  based on HTTP, allows clients to structure data

```
POST /getAccelerator HTTP/1.1
HOST: ccda
Content-Type: application/json
{"name": "LHC"}
```

JSON-RPC

```
<?xml version="1.0"?>
<soap:Envelope (..)>
<soap:Body>
 <m:GetAcceleratorDetails xmlns:m="https://ccda/accelerators">
  <m:Name>LHC</m:Name>
 </m:GetAcceleratorDetails>
</soap:Body>
</soap:Envelope>
```

SOAP

```
GET /api/core/accelerators/LHC HTTP/2.0
Host: ccda
Connection: keep-alive
Accept: text/html,application/json
Accept-Encoding: gzip, deflate, br
```

REST

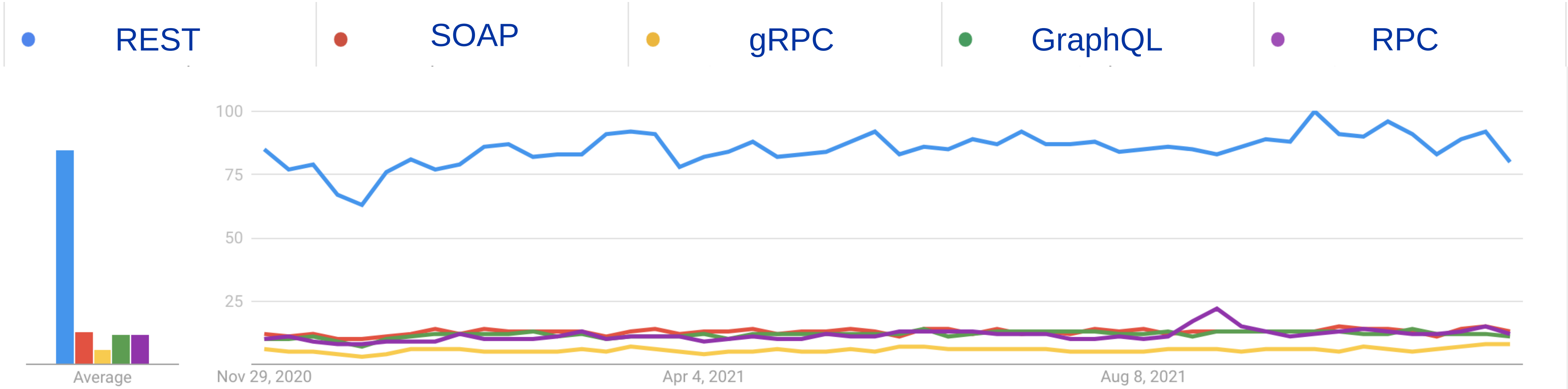
```
:method POST
:path /Accelerators/getAccelerator

(encoded message using protocol buffers)
```

gRPC

```
Request:
{
 accelerator {
  name
 }
}
Response:
{ "accelerator": {
   "name": "LHC"
  }
}
```

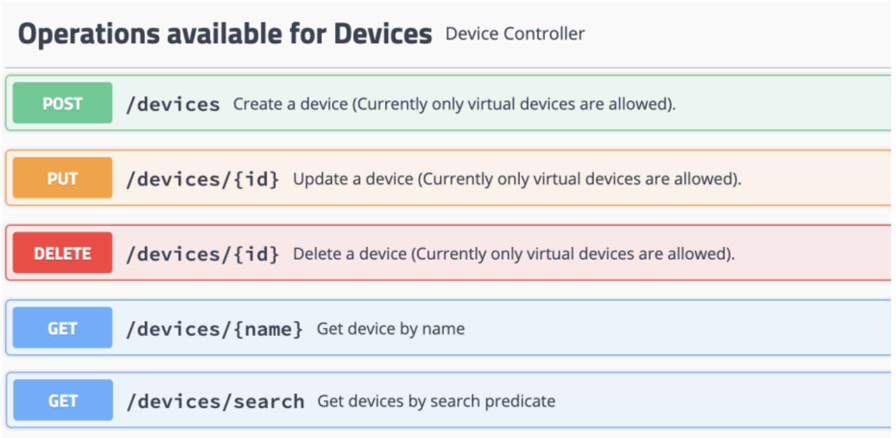
GraphQL

API Technology Trends



Worldwide interest over last 12months (Source: Google Trends)

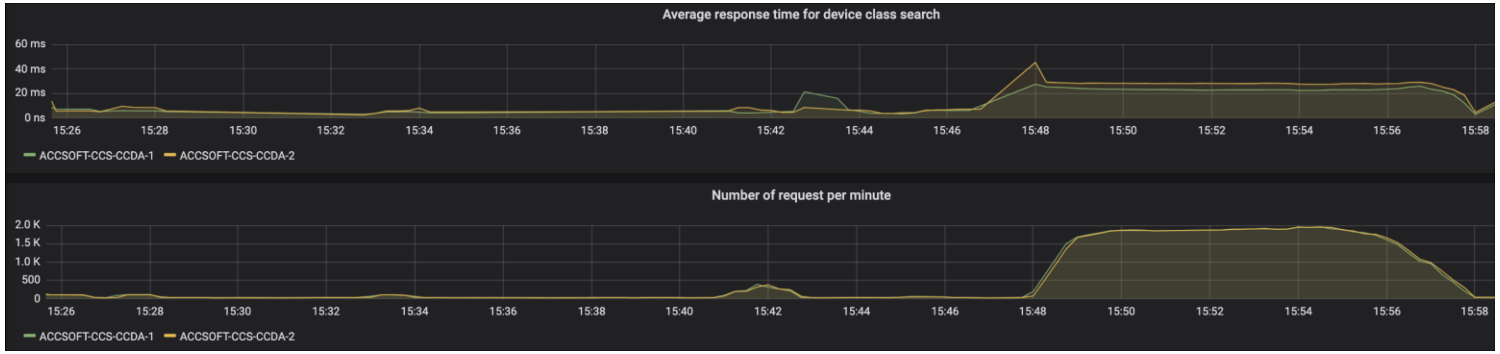
API – Goals & Challenges



- Make it **easy to work** with – should be **intutive** to the users

- Make it **stable** – limit **breaking changes** to bare minimum
  no changes/API versioning/EOLs

- Make it fast and performant
  number of requests should not significantly impact the API

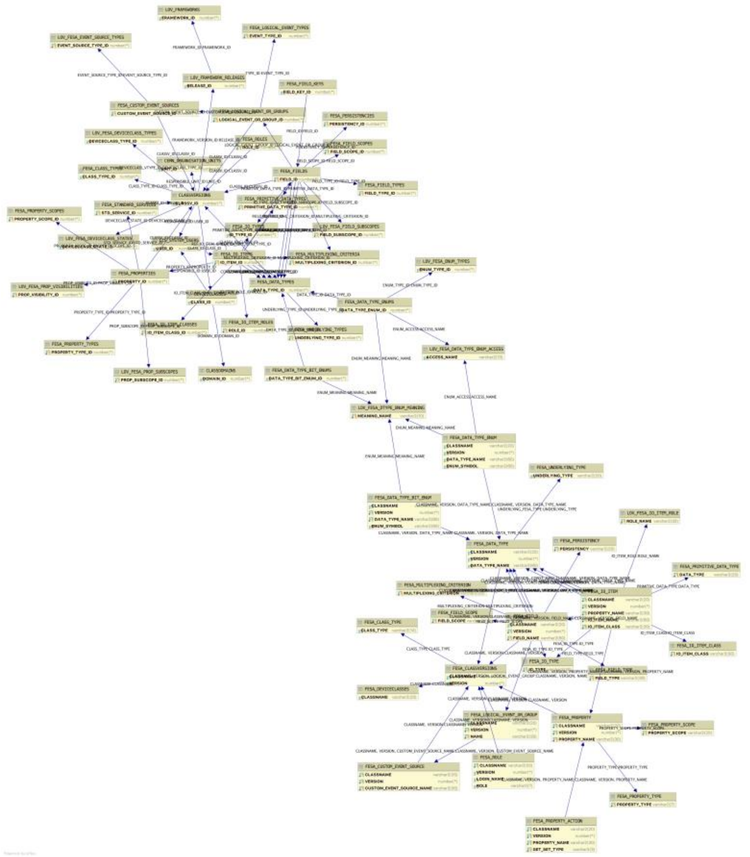


- Make it technology agnostic when justified – driven by users needs
  API should be accessible from as many technologies (languages) as it is possible

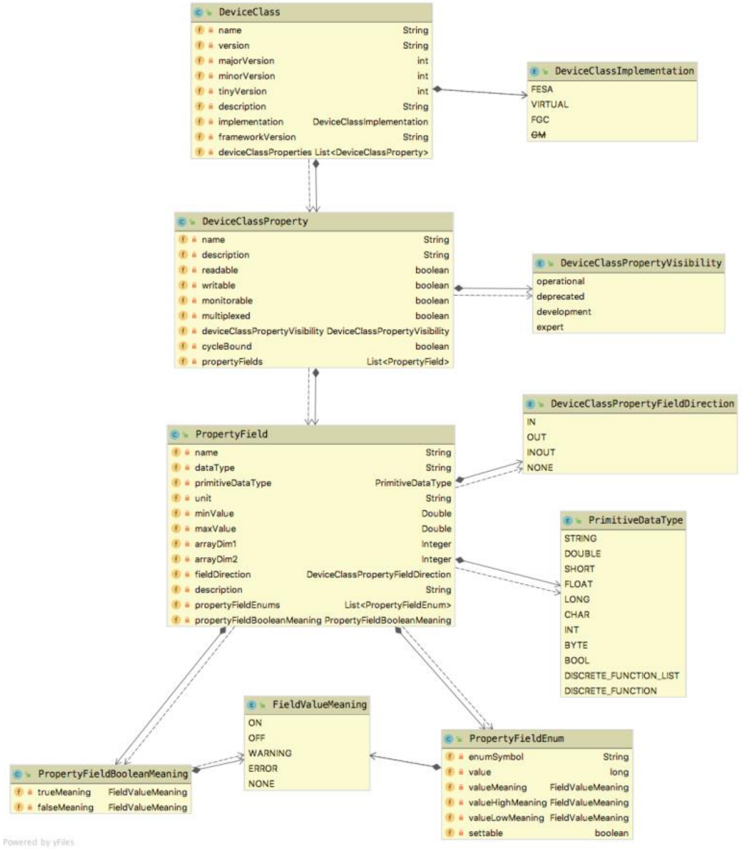
APIs in practice

CCS APIs – how we do it – real-life examples

Relational DB model of Controls Device Classes *



CCDA object-model (Java)



JSON representation


```
{
  "name":"XenericSampler",
  "version":"7.1.0",
  "majorVersion":7,
  "minorVersion":1,
  "tinyVersion":0,
  "description":"Everything can be subscribed and sampled",
  "implementation":"FESA",
  "frameworkVersion":"7.0.0",
  "deviceClassProperties":[
    {
      "name":"CycleDescription",
      "description":null,
      "readable":true,
      "writable":false,
      "monitorable":true,
      "multiplexed":false,
      "cycleBound":true,
      "propertyFields":[
        {
          "name":"cycleTimeStamp",
          "dataType":"scalar",
          "primitiveDataType":"LONG",
          "unit":null,
          "minValue":null,
          "maxValue":null,
          "arrayDim1":null,
          "arrayDim2":null,
          "fieldDirection":"OUT",
          "description":null,
          "propertyFieldEnums":[

          ],
          "propertyFieldBooleanMeaning":null
        },
        {
          "name":"cycleName",
          "dataType":"scalar",
          "primitiveDataType":"STRING",
          "unit":null,
          "minValue":null,
          "maxValue":null,
          "arrayDim1":32,
          "arrayDim2":null,
          "fieldDirection":"OUT",
          "description":null,
          "propertyFieldEnums":[

          ],
          "propertyFieldBooleanMeaning":null
        }
      ]
    }
  ]
}
```

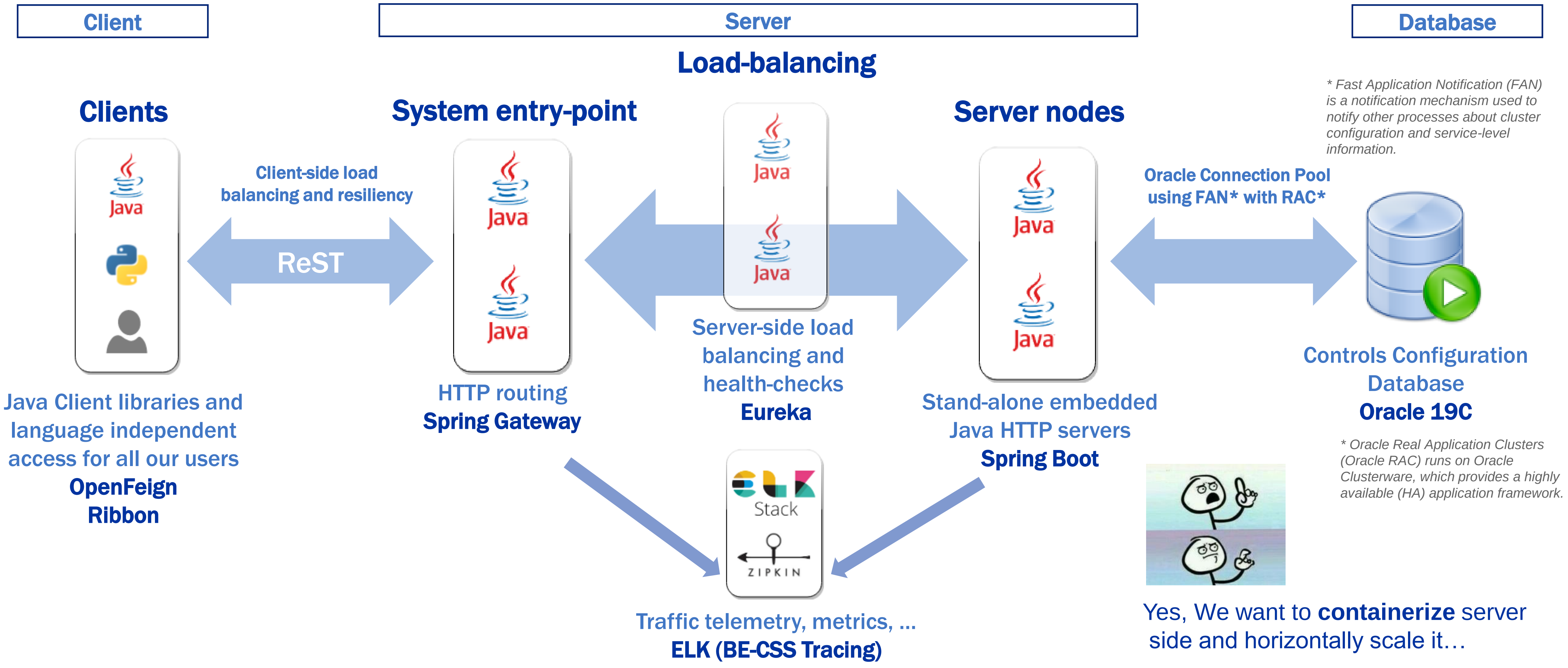

Complex DB model

* **Controls Device Class** – an abstraction of physical equipment or a process/service, used to interact with the machines (acquisition, settings, commands).

Simple and intuitive model

Language independent

The classic 3-tier architecture



API in action

Client

```java
Accelerator accelerator =
        CcdaClient.newInstance().getService(AcceleratorService.class).findByName("CPS");
```

```
bartek@macbe16640 ~ % curl -k https://[redacted].cern.ch:8900/api/core/accelerators/CPS
```


```json
{
  "name": "CPS",
  "description": "Cern Proton Synchrotron and beam transfer lines",
  "acceleratorZones": [
    {
      "name": "F63",
      "defaultZone": false
    },
    {
      "name": "T08",
      "defaultZone": false
    },
    {
      "name": "F16",
      "defaultZone": false
    },
    {
      "name": "F61",
      "defaultZone": false
    },
    {
      "name": "PS",
      "defaultZone": true
    },
    {
      "name": "F62_1",
      "defaultZone": false
    },
    {
      "name": "F62_2",
      "defaultZone": false
    }
  ],
  "timingDomain": {
    "name": "CPS"
  }
}
```


Jackson – high performance JSON processor for Java

Server

```java
@Api(tags = "Operations available for Accelerators")
@RestController
@RequestMapping("/core/accelerators")
@RequiredArgsConstructor
public class AcceleratorController {   Complexity is 3 Everything is cool!

    @NonNull
    private final AcceleratorService acceleratorService;

    @ApiOperation(value = "Get all accelerators")
    @GetMapping
    public List<Accelerator> findAll() { return acceleratorService.findAll(); }

    @ApiOperation(value = "Get accelerator by name")
    @GetMapping(value = "/{name}")
    public Accelerator findByName(@PathVariable String name) { return acceleratorService.findByName(name); }
}
```

```java
@Service
@Transactional
@RequiredArgsConstructor
public class AcceleratorService {   Complexity is 4 Everything is cool!

    @NonNull
    private final AcceleratorRepository acceleratorRepository;
    @NonNull
    private final AcceleratorMapper acceleratorMapper;

    @Cacheable(CacheName.Constants.CACHE_ACCELERATOR)
    public Accelerator findByName(String name) {   Complexity is 3 Everything is cool!
        return acceleratorMapper.toAccelerator(acceleratorRepository.findByNameIgnoreCase(name)
                .orElseThrow(() -> new EntityNotFoundException("Accelerator with name '" + name + "' was not found.")));
    }

    @Cacheable(CacheName.Constants.CACHE_ACCELERATOR)
    public List<Accelerator> findAll() { return acceleratorMapper.toAccelerators(acceleratorRepository.findAll()); }
}
```

```java
public interface AcceleratorRepository extends Repository<AcceleratorEntity, String> {

    Optional<AcceleratorEntity> findByNameIgnoreCase(String name);

    List<AcceleratorEntity> findAll();

}
```

Database

Lombok – reduce boilerplate code – e.g. auto generated getters/setters

```java
@Entity
@Immutable
@Table(name = "ACCELERATORS")
@Data
@Builder
@NoArgsConstructor
@AllArgsConstructor
public class AcceleratorEntity {

    @Id
    @Column(name = "ACCELERATOR")
    private String name;

    @Column(name = "ACCELDESCRIP")
    private String description;

    @OneToMany
    @JoinColumn(name = "ACCELERATOR")
    private List<AcceleratorZoneEntity> acceleratorZones;

    @ManyToOne
    @JoinColumn(name = "PLSMACHINE_ID")
    private TimingDomainEntity timingDomain;
}
```

Spring JPA in practice (with Hibernate) – an entity mapped to a table

	ACCELERATOR	ACCELERATOR_ZONE_NAME	IS_DEFAULT_ZONE
1	CPS	F63	N
2	CPS	T08	N
3	CPS	F16	N
4	CPS	F61	N
5	CPS	PS	Y
6	CPS	F62_1	N
7	CPS	F62_2	N

How to protect the API

Authentication

Authentication **verifies** client/user **identity**

Common Authentication Methods:

- password *(what you know)*
- token/digital id card *(what you posses)*
- finger print/face recognition *(what you are)*
- location (where you are, e.g.: all connections from CCC)

```java
public String login(String username, String password, ClientAddress clientAddress) {   Complexity is 4 Everything is cool!
    checkNotNull(username);
    checkNotNull(password);
    checkNotNull(clientAddress);
```

Authorization

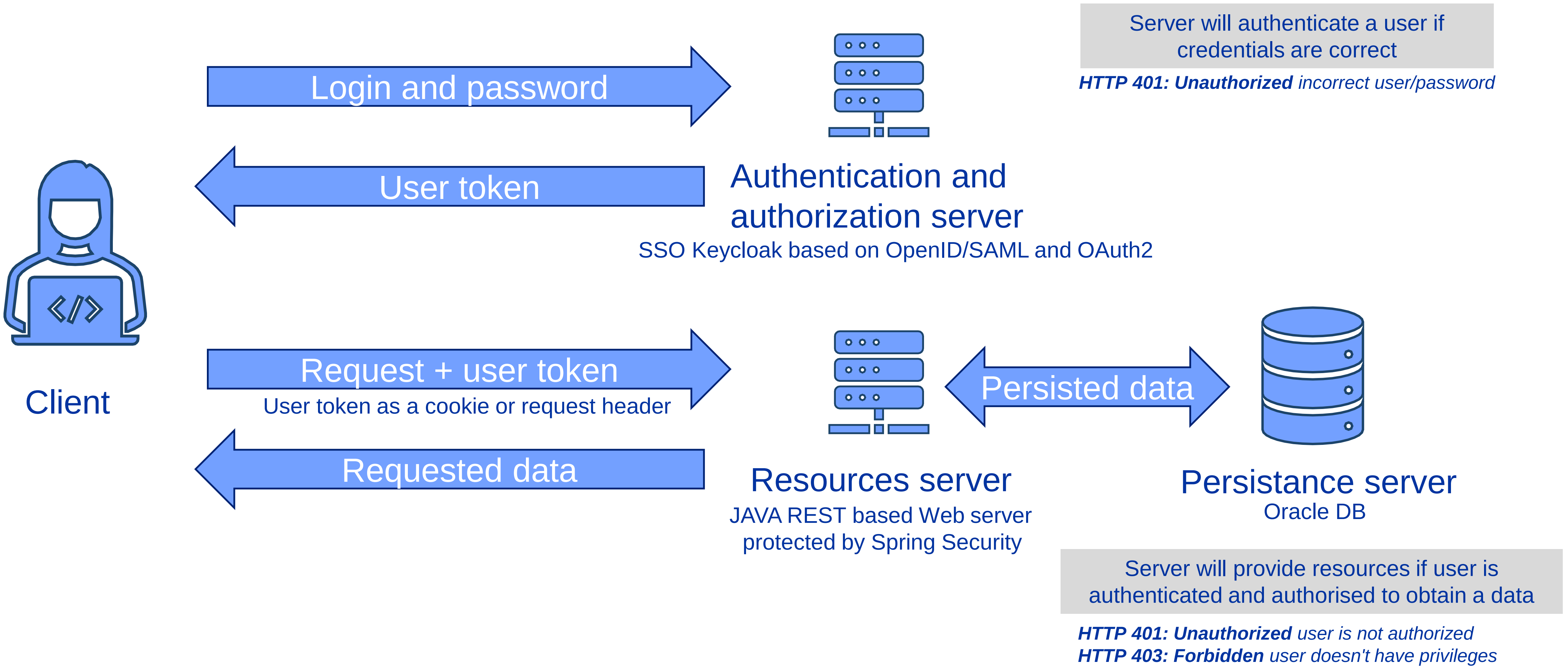
Authorization **determines** client/user **access to data**

Common Authorization Methods:

- RBAC - Role-based access control
  - *with a specific role (OP-Expert) you may control accelerator equipment*
- ABAC – Attribute-based access control
  - *with a specific attributes (time, location, role) you may control accelerator equipment only at a given time and from a given location*

```java
@PreAuthorize("hasRole('CCS-DEVICES-EDITOR') && @deviceClassAuthorization.validateDeviceClassForCreate(authentication, #deviceClass)")
public DeviceClass createDeviceClass(Authentication authentication, DeviceClass deviceClass) {   Complexity is 4 Everything is cool!
```

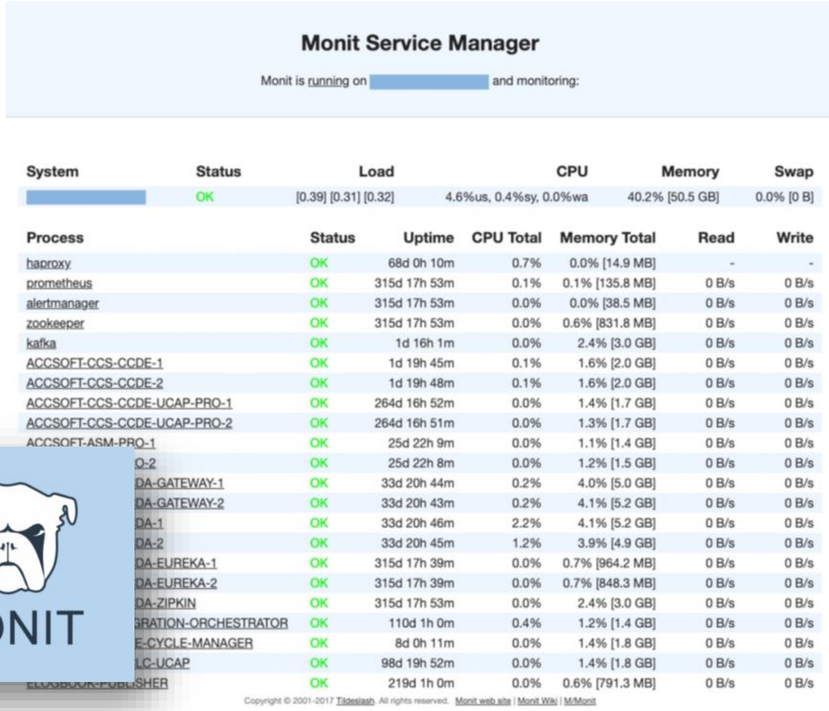
Authentication and authorisation basic workflow



Monitoring, alerting and tracing

Monitoring:

- shows that all process (servers) are up and running
- gives base information about processes condition

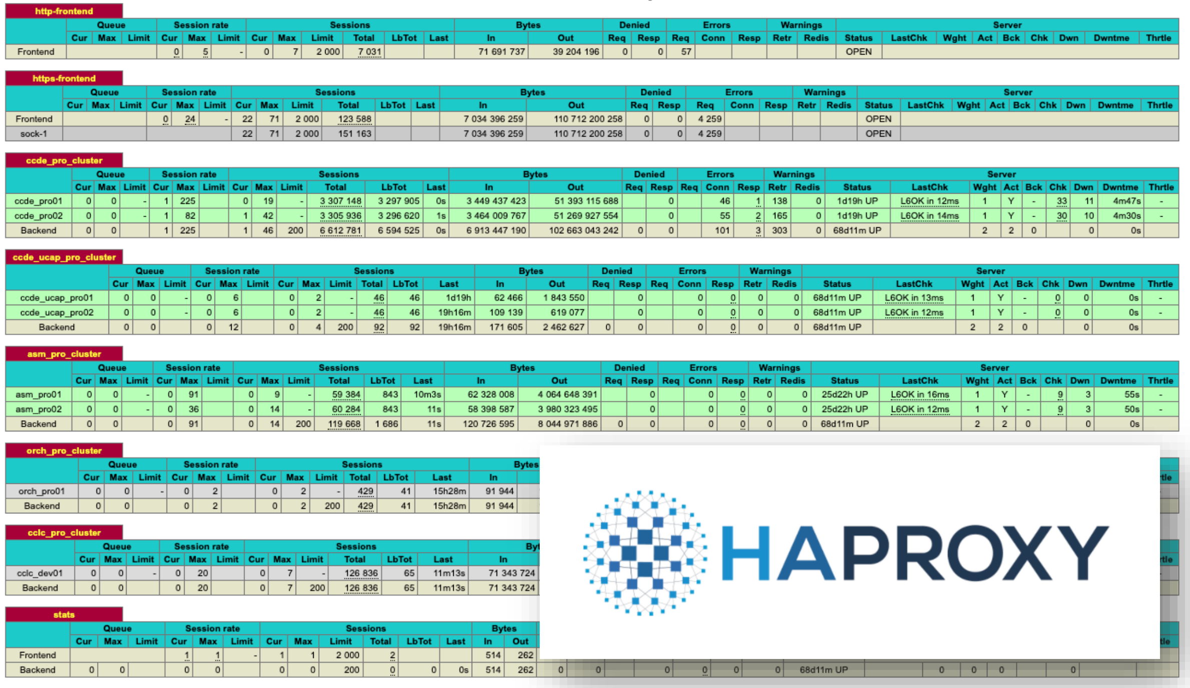


Monit Service Manager

Monit is running on and monitoring:

System	Status	Load	CPU	Memory	Swap
	OK	[0.39] [0.31] [0.32]	4.6%us, 0.4%sy, 0.0%wa	40.2% [50.5 GB]	0.0% [0 B]

Process	Status	Uptime	CPU Total	Memory Total	Read	Write
haproxy	OK	68d 0h 10m	0.7%	0.0% [14.9 MB]	-	-
prometheus	OK	315d 17h 53m	0.1%	0.1% [135.8 MB]	0 B/s	0 B/s
alertmanager	OK	315d 17h 53m	0.0%	0.0% [38.5 MB]	0 B/s	0 B/s
zookeeper	OK	315d 17h 53m	0.0%	0.6% [831.8 MB]	0 B/s	0 B/s
kafka	OK	1d 16h 1m	0.0%	2.4% [3.0 GB]	0 B/s	0 B/s
ACCSOFT-CCS-CCDE-1	OK	1d 19h 45m	0.1%	1.6% [2.0 GB]	0 B/s	0 B/s
ACCSOFT-CCS-CCDE-2	OK	1d 19h 48m	0.1%	1.6% [2.0 GB]	0 B/s	0 B/s
ACCSOFT-CCS-CCDE-UCAP-PRO-1	OK	264d 16h 52m	0.0%	1.4% [1.7 GB]	0 B/s	0 B/s
ACCSOFT-CCS-CCDE-UCAP-PRO-2	OK	264d 16h 51m	0.0%	1.3% [1.7 GB]	0 B/s	0 B/s
ACCSOFT-ASM-PRO-1	OK	25d 22h 9m	0.0%	1.1% [1.4 GB]	0 B/s	0 B/s
O-2	OK	25d 22h 8m	0.0%	1.2% [1.5 GB]	0 B/s	0 B/s
DA-GATEWAY-1	OK	33d 20h 44m	0.2%	4.0% [5.0 GB]	0 B/s	0 B/s
DA-GATEWAY-2	OK	33d 20h 43m	0.2%	4.1% [5.2 GB]	0 B/s	0 B/s
DA-1	OK	33d 20h 46m	2.2%	4.1% [5.2 GB]	0 B/s	0 B/s
DA-2	OK	33d 20h 45m	1.2%	3.9% [4.9 GB]	0 B/s	0 B/s
DA-EUREKA-1	OK	315d 17h 39m	0.0%	0.7% [964.2 MB]	0 B/s	0 B/s
DA-EUREKA-2	OK	315d 17h 39m	0.0%	0.7% [848.3 MB]	0 B/s	0 B/s
DA-ZIPKIN	OK	315d 17h 53m	0.0%	2.4% [3.0 GB]	0 B/s	0 B/s
GRATION-ORCHESTRATOR	OK	110d 1h 0m	0.4%	1.2% [1.4 GB]	0 B/s	0 B/s
E-CYCLE-MANAGER	OK	8d 0h 11m	0.0%	1.4% [1.8 GB]	0 B/s	0 B/s
LC-UCAP	OK	98d 19h 52m	0.0%	1.4% [1.8 GB]	0 B/s	0 B/s
ELOGBOOK-PUBLISHER	OK	219d 1h 0m	0.0%	0.6% [791.3 MB]	0 B/s	0 B/s



http-frontend

	Queue Cur	Queue Max	Queue Limit	Session rate Cur	Session rate Max	Session rate Limit	Sessions Cur	Sessions Max	Sessions Limit	Sessions Total	Sessions LbTot	Sessions Last	Bytes In	Bytes Out	Denied Req	Denied Resp	Errors Req	Errors Conn	Errors Resp	Warnings Retr	Warnings Redis	Server Status	Server LastChk	Server Wght	Server Act	Server Bck	Server Chk	Server Dwn	Server Dwntme	Server Thrtle
Frontend				0	5	-	0	7	2 000	7 031			71 691 737	39 204 196	0	0	57					OPEN								

https-frontend

	Queue Cur	Queue Max	Queue Limit	Session rate Cur	Session rate Max	Session rate Limit	Sessions Cur	Sessions Max	Sessions Limit	Sessions Total	Sessions LbTot	Sessions Last	Bytes In	Bytes Out	Denied Req	Denied Resp	Errors Req	Errors Conn	Errors Resp	Warnings Retr	Warnings Redis	Server Status	Server LastChk	Server Wght	Server Act	Server Bck	Server Chk	Server Dwn	Server Dwntme	Server Thrtle
Frontend				0	24	-	22	71	2 000	123 588			7 034 396 259	110 712 200 258	0	0	4 259					OPEN								
sock-1							22	71	2 000	151 163			7 034 396 259	110 712 200 258	0	0	4 259					OPEN								

ccde_pro_cluster

	Queue Cur	Queue Max	Queue Limit	Session rate Cur	Session rate Max	Session rate Limit	Sessions Cur	Sessions Max	Sessions Limit	Sessions Total	Sessions LbTot	Sessions Last	Bytes In	Bytes Out	Denied Req	Denied Resp	Errors Req	Errors Conn	Errors Resp	Warnings Retr	Warnings Redis	Server Status	Server LastChk	Server Wght	Server Act	Server Bck	Server Chk	Server Dwn	Server Dwntme	Server Thrtle
ccde_pro01	0	0	-	1	225		0	19	-	3 307 148	3 297 905	0s	3 449 437 423	51 393 115 688		0		46	1	138	0	1d19h UP	L6OK in 12ms	1	Y	-	33	11	4m47s	-
ccde_pro02	0	0	-	1	82		1	42	-	3 305 936	3 296 620	1s	3 464 009 767	51 269 927 554		0		55	2	165	0	1d19h UP	L6OK in 14ms	1	Y	-	30	10	4m30s	-
Backend	0	0		1	225		1	46	200	6 612 781	6 594 525	0s	6 913 447 190	102 663 043 242	0	0		101	3	303	0	68d11m UP		2	2	0		0	0s	

ccde_ucap_pro_cluster

	Queue Cur	Queue Max	Queue Limit	Session rate Cur	Session rate Max	Session rate Limit	Sessions Cur	Sessions Max	Sessions Limit	Sessions Total	Sessions LbTot	Sessions Last	Bytes In	Bytes Out	Denied Req	Denied Resp	Errors Req	Errors Conn	Errors Resp	Warnings Retr	Warnings Redis	Server Status	Server LastChk	Server Wght	Server Act	Server Bck	Server Chk	Server Dwn	Server Dwntme	Server Thrtle
ccde_ucap_pro01	0	0	-	0	6		0	2	-	46	46	1d19h	62 466	1 843 550		0		0	0	0	0	68d11m UP	L6OK in 13ms	1	Y	-	0	0	0s	-
ccde_ucap_pro02	0	0	-	0	6		0	2	-	46	46	19h16m	109 139	619 077		0		0	0	0	0	68d11m UP	L6OK in 12ms	1	Y	-	0	0	0s	-
Backend	0	0		0	12		0	4	200	92	92	19h16m	171 605	2 462 627	0	0		0	0	0	0	68d11m UP		2	2	0		0	0s	

asm_pro_cluster

	Queue Cur	Queue Max	Queue Limit	Session rate Cur	Session rate Max	Session rate Limit	Sessions Cur	Sessions Max	Sessions Limit	Sessions Total	Sessions LbTot	Sessions Last	Bytes In	Bytes Out	Denied Req	Denied Resp	Errors Req	Errors Conn	Errors Resp	Warnings Retr	Warnings Redis	Server Status	Server LastChk	Server Wght	Server Act	Server Bck	Server Chk	Server Dwn	Server Dwntme	Server Thrtle
asm_pro01	0	0	-	0	91		0	9	-	59 384	843	10m3s	62 328 008	4 064 648 391		0		0	0	0	0	25d22h UP	L6OK in 16ms	1	Y	-	9	3	55s	-
asm_pro02	0	0	-	0	36		0	14	-	60 284	843	11s	58 398 587	3 980 323 495		0		0	0	0	0	25d22h UP	L6OK in 12ms	1	Y	-	9	3	50s	-
Backend	0	0		0	91		0	14	200	119 668	1 686	11s	120 726 595	8 044 971 886	0	0		0	0	0	0	68d11m UP		2	2	0		0	0s	

orch_pro_cluster

	Queue Cur	Queue Max	Queue Limit	Session rate Cur	Session rate Max	Session rate Limit	Sessions Cur	Sessions Max	Sessions Limit	Sessions Total	Sessions LbTot	Sessions Last	Bytes In
orch_pro01	0	0	-	0	2		0	2	-	429	41	15h28m	91 944
Backend	0	0		0	2		0	2	200	429	41	15h28m	91 944

cclc_pro_cluster

	Queue Cur	Queue Max	Queue Limit	Session rate Cur	Session rate Max	Session rate Limit	Sessions Cur	Sessions Max	Sessions Limit	Sessions Total	Sessions LbTot	Sessions Last	Bytes In
cclc_dev01	0	0	-	0	20		0	7	-	126 836	65	11m13s	71 343 724
Backend	0	0		0	20		0	7	200	126 836	65	11m13s	71 343 724

stats

	Queue Cur	Queue Max	Queue Limit	Session rate Cur	Session rate Max	Session rate Limit	Sessions Cur	Sessions Max	Sessions Limit	Sessions Total	Sessions LbTot	Sessions Last	Bytes In	Bytes Out	Denied Req	Denied Resp	Errors Req	Errors Conn	Errors Resp	Warnings Retr	Warnings Redis	Server Status	Server LastChk	Server Wght	Server Act	Server Bck	Server Chk	Server Dwn	Server Dwntme	Server Thrtle
Frontend				1	1	-	1	1	2 000	2			514	262																
Backend	0	0		0	0		0	0	200	0	0	0s	514	262	0	0		0	0	0	0	68d11m UP		0	0	0		0		

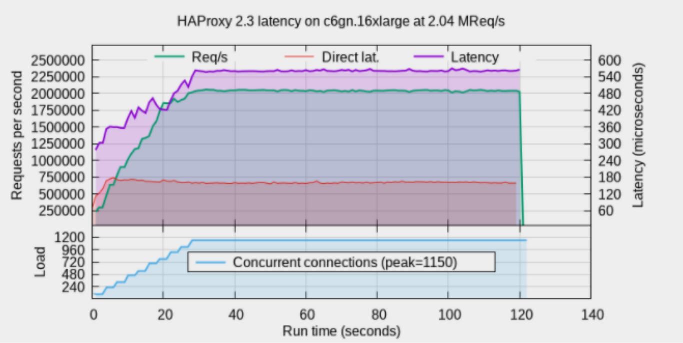
Telemetry with HTTP Load balancer based on HAProxy

HAPROXY

Free and **open source high availability load balancer** for TCP and HTTP based applications

On 64-core ARM server HAProxy is able to handle more than 2 million requests per second.

after HAProxy Team analysis



Number of waiting sessions | Number of new session per second | Number of current and total sessions | Traffic that was generated by the application | Number of non OK responses | Statistics about the application uptime

ccde_pro_cluster

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
ccde_pro01	0	0	-	0	225		0	19	-	3 436 179	3 426 906	10m27s	3 586 846 432	52 010 207 337		0		46	1	138	0	6d18h UP	L6OK in 12ms	1	Y	-	33	11	4m47s	-
ccde_pro02	0	0	-	0	82		0	42	-	3 434 962	3 425 621	10m27s	3 603 999 197	51 912 183 135		0		55	2	165	0	6d18h UP	L6OK in 14ms	1	Y	-	30	10	4m30s	-
Backend	0	0		0	225		0	46	200	6 870 838	6 852 527	10m27s	7 190 845 629	103 922 390 472	0	0		101	3	303	0	72d22h UP		2	2	0		0	0s	

Cum. sessions: 3 436 309
Cum. HTTP responses: 3 436 076
- HTTP 1xx responses: 19 (0%)
- HTTP 2xx responses: 3 403 898 (99%)
- HTTP 3xx responses: 16 458 (0%)
- HTTP 4xx responses: 15 591 (0%)
- HTTP 5xx responses: 110 (0%)
- other responses: 0 (0%)

Avg over last 1024 success. conn.
- Queue time: 0 ms
- Connect time: 12 ms
- Response time: 289 ms
- Total time: 9 912 ms

ccde_ucap_pro_cluster

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
ccde_ucap_pro01	0	0	-	0	6		[illegible]	[illegible]	[illegible]	[illegible]	73	1d17h	237 936	13 751 965		0		0	0	0	0	72d22h UP	L6OK in 13ms	1	Y	-	0	0	0s	-
ccde_ucap_pro02	0	0	-	0	6		[illegible]	[illegible]	[illegible]	[illegible]	73	17h42m	181 831	6 139 823		0		0	0	0	0	72d22h UP	L6OK in 13ms	1	Y	-	0	0	0s	-
Backend	0	0		0	12		[illegible]	[illegible]	[illegible]	[illegible]	146	17h42m	419 767	19 891 788	0	0		0	0	0	0	72d22h UP		2	2	0		0	0s	

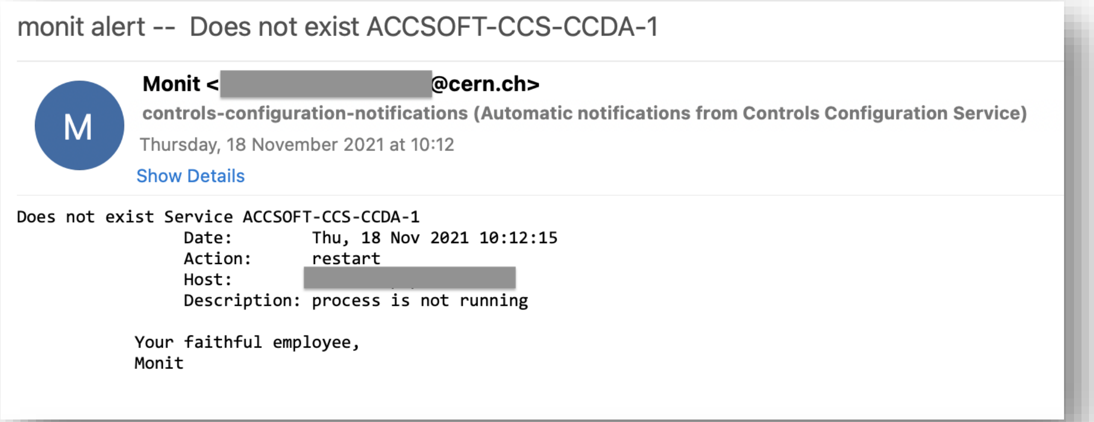
asm_pro_cluster

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	[illegible]	[illegible]	[illegible]	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
asm_pro01	0	0	-	0	91		0	[illegible]	[illegible]	[illegible]	[illegible]	1h55m	63 801 983	4 207 416 609		0		0	0	0	0	30d20h UP	L6OK in 12ms	1	Y	-	9	3	55s	-
asm_pro02	0	0	-	0	36		0	14	-	61 391	872	7m7s	59 433 904	4 097 136 979		0		0	0	0	0	30d20h UP	L6OK in 13ms	1	Y	-	9	3	50s	-
Backend	0	0		0	91		0	14	200	122 040	1 745	7m7s	123 235 887	8 304 553 588	0	0		0	0	0	0	72d22h UP		2	2	0		0	0s	

Monitoring, alerting and tracing

Alerting

- notifies us about all anomalies – helps to prevent system downtime
- allows to improve quality of service – problems are fixed before users spot them



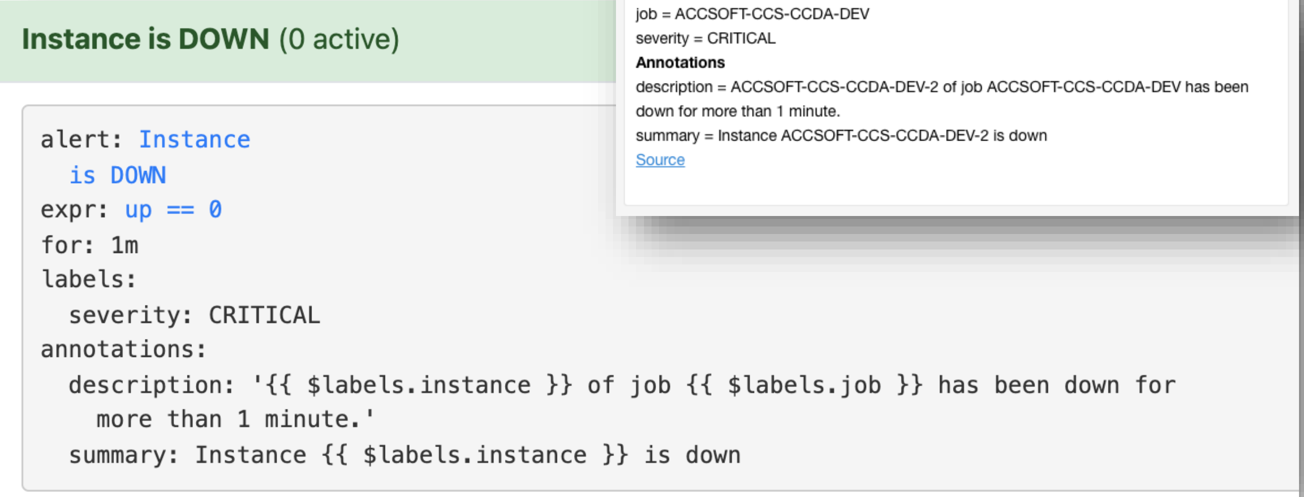
monit alert -- Does not exist ACCSOFT-CCS-CCDA-1

Monit <[redacted]@cern.ch>
controls-configuration-notifications (Automatic notifications from Controls Configuration Service)
Thursday, 18 November 2021 at 10:12
Show Details

```
Does not exist Service ACCSOFT-CCS-CCDA-1
                Date:        Thu, 18 Nov 2021 10:12:15
                Action:      restart
                Host:        [redacted]
                Description: process is not running

           Your faithful employee,
           Monit
```

```
Request URL: https://[redacted]/api/edge/configuration/hardware-types/VFC_HD_BSRA/logical-hw-interfaces/vfc_hd_bsra/versions/0.0.1-dev
Request method: PUT
Referer header: null
Application name: ACCSOFT-CCS-CCDA-PRO
Username: belohrad

Root cause: NullPointerException:

Request body:
null

Stack trace:
java.lang.NullPointerException
        at cern.accsoft.ccs.ccda.domain.edge.mapper.BlockInstanceMapper.toBlockInstanceEntities(BlockInstanceMapper.java:18)
        at cern.accsoft.ccs.ccda.domain.edge.mapper.EdgeConfigurationMapper.lambda$buildBlockInstanceEntities$9(EdgeConfigurationMapper.java:143)
        at java.base/java.util.ArrayList.forEach(ArrayList.java:1541)
        at cern.accsoft.ccs.ccda.domain.edge.mapper.EdgeConfigurationMapper.buildBlockInstanceEntities(EdgeConfigurationMapper.java:141)
        at cern.accsoft.ccs.ccda.domain.edge.mapper.EdgeConfigurationMapper.toRegisterMapDefinitionEntity(EdgeConfigurationMapper.java:38)
        at cern.accsoft.ccs.ccda.domain.edge.service.HardwareModuleRegisterMapDefinitionService.save(HardwareModuleRegisterMapDefinitionService.java:43)
        at cern.accsoft.ccs.ccda.domain.edge.service.HardwareModuleRegisterMapDefinitionService.update(HardwareModuleRegisterMapDefinitionService.java:60)
        at cern.accsoft.ccs.ccda.domain.edge.service.HardwareModuleRegisterMapDefinitionService$$FastClassBySpringCGLIB$$c0fd85a0.invoke(<generated>)
        at org.springframework.cglib.proxy.MethodProxy.invoke(MethodProxy.java:218)
```

Prometheus



Instance is DOWN (0 active)

```
alert: Instance
  is DOWN
expr: up == 0
for: 1m
labels:
  severity: CRITICAL
annotations:
  description: '{{ $labels.instance }} of job {{ $labels.job }} has been down for
    more than 1 minute.'
  summary: Instance {{ $labels.instance }} is down
```

1 alert for job=ACCSOFT-CCS-CCDA-DEV

View In AlertManager

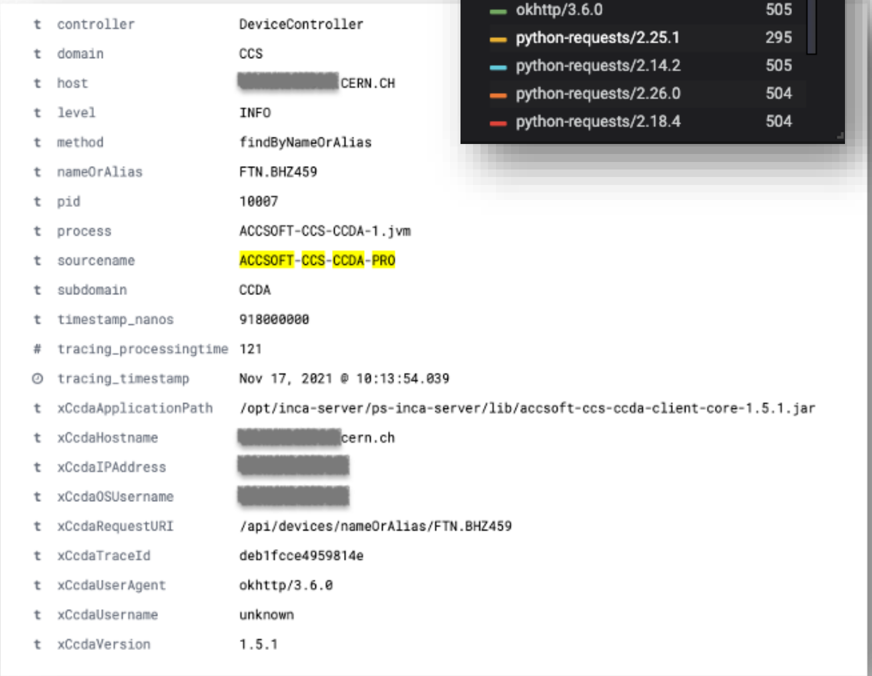
[1] Firing

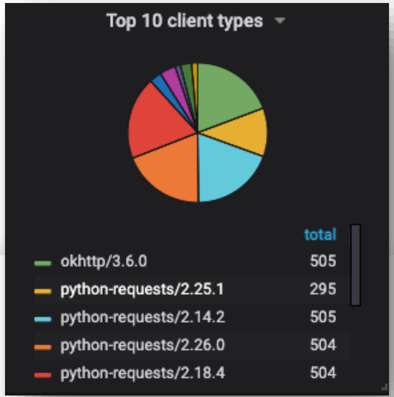
Labels
alertname = Instance is DOWN
instance = ACCSOFT-CCS-CCDA-DEV-2
job = ACCSOFT-CCS-CCDA-DEV
severity = CRITICAL
Annotations
description = ACCSOFT-CCS-CCDA-DEV-2 of job ACCSOFT-CCS-CCDA-DEV has been down for more than 1 minute.
summary = Instance ACCSOFT-CCS-CCDA-DEV-2 is down
Source

Monitoring, alerting and tracing

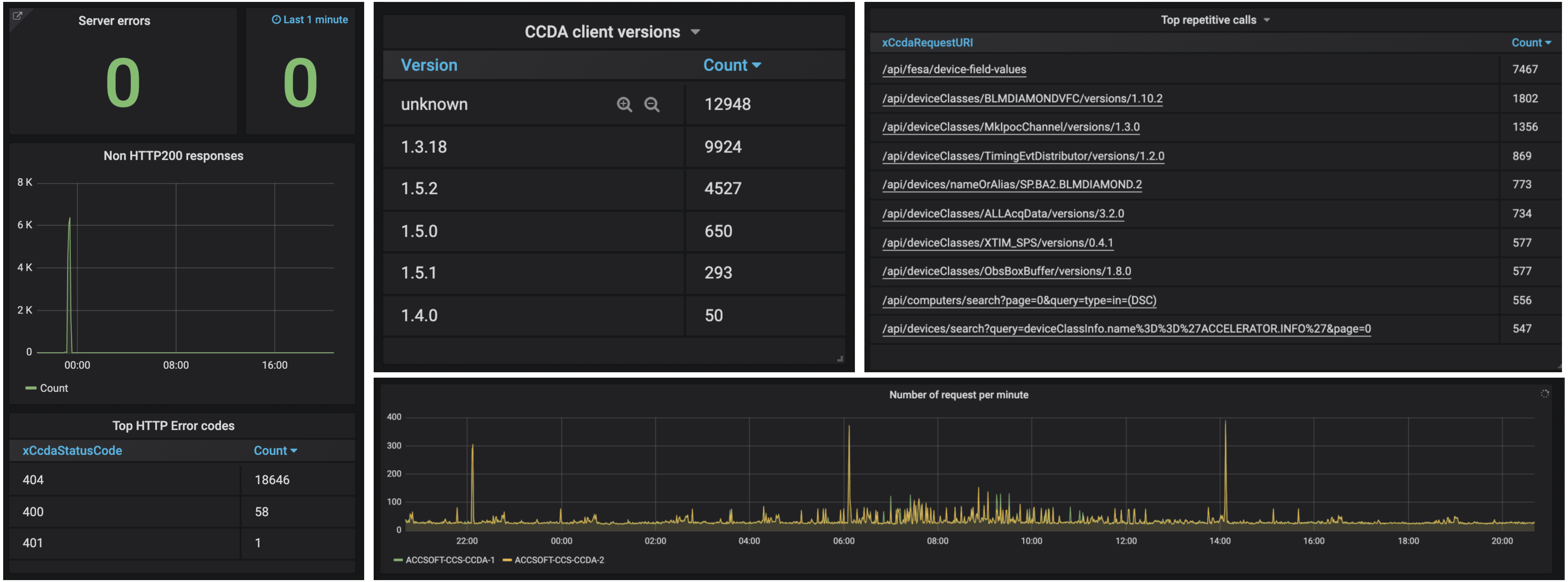
Tracing:

- informs us about usage of our API – who, how, when
- allows us to analyse and discover anomalies of a running system





Tracing - Grafana



Availability vs Reliability

Availability

The percentage of time when system or service is operational from point of view of its clients

Highly-available system: *"zero-downtime" operations, including rolling-upgrades*

Service is available for 99%

```
{
"status": 404,
"errorType": "PATH_NOT_FOUND",
"message": "Current path
'/api/core/accelerators/LHC' was not found",
"timestamp": "2021-11-19T10:52:00.578311Z"
}
```

Reliability

The probability that system will meet designed performance standards and produce correct output for a specific time

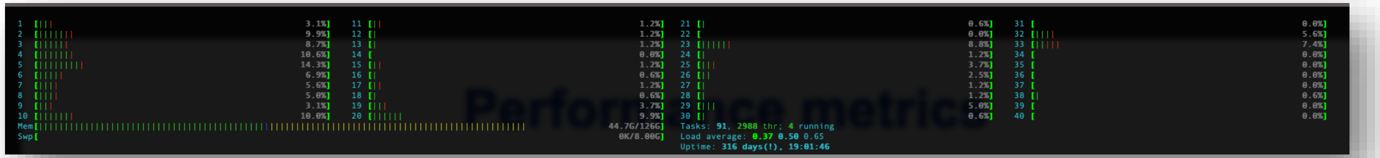
Mean Time Between Failure (MTBF)*: total time in service / number of failures*
Failure Rate (λ)*: number of failures / total time in service.*

No more than 5 failures per day

```
GET '/api/core/accelerators/LHC'
{
"name": "CPS",
"description": "Cern Proton Synchrotron and beam transfer lines",
"acceleratorZones": [
{"name": "F63","defaultZone": false},
{"name": "T08","defaultZone": false},
```

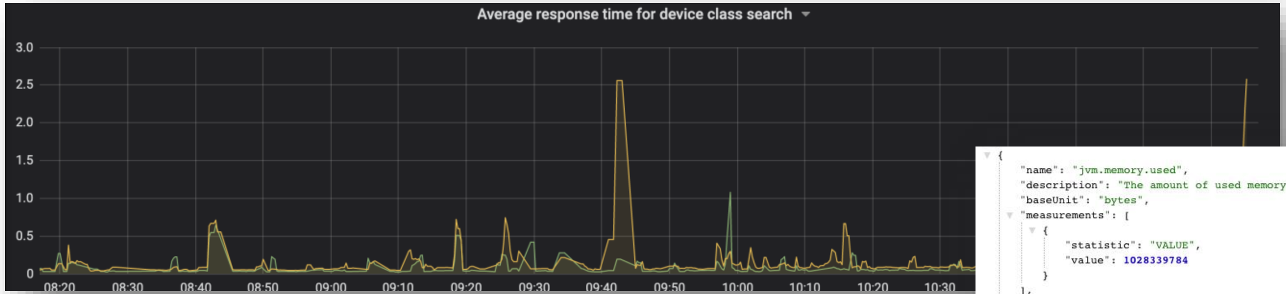
Performance metrics

Computer **performance** is the amount of useful work accomplished by a computer **system.**
Performance can be estimated in terms of accuracy, efficiency and speed of execution.



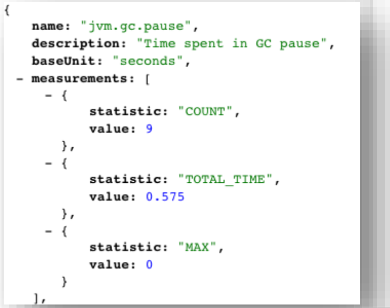
We can measure this with various metrics:

- **Technical** (hardware) – health of the system
  Memory, CPU, disk space, number of running processes, networking etc



- **Performance** – Indicator of the problem
  Mean/Max response time, latency, throughput, errors rate

- **Implementation specific** – JVM metrics, Spring Actuator
  GC configurations (type of GC), number of threads (running, waiting, blocked), process memory



* containers bring another layer of HW abstraction, and own metrics

How to verify our system - testing

- **Unit, integration, regression tests – availability and reliability**

	Declarative: Checkout SCM	1 - pre-Build	2 - Test: JUnit	3 - Test: Integration	4 - Build: Publish	5 - Test: Integration Backward compatibility	6 - Deploy	Declarative: Post Actions
Average stage times: (Average full run time: ~23min 45s)	2s	1min 23s	1min 26s	8min 20s	1min 26s	7min 43s	3min 20s	68ms
#699 Nov 18 11:58 1 commit	2s	1min 21s	1min 27s	8min 26s	1min 27s	7min 47s	3min 19s	76ms

- **Stress testing – performance**

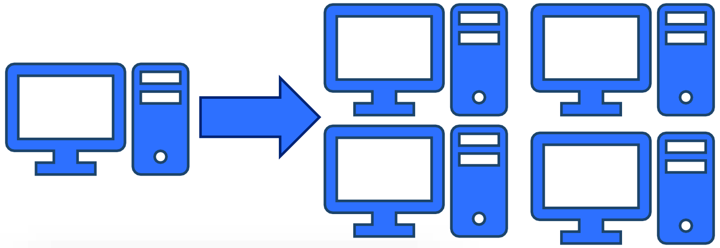
Gatling

7 - Gatling Stress testing
2min 59s
2min 59s

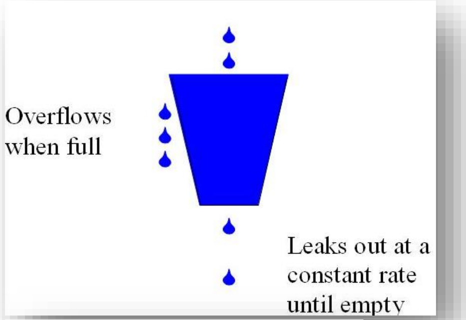
```
================================================================================
2021-10-14 10:43:10                                           161s elapsed
---- Requests ------------------------------------------------------------------
> Global                                              (OK=12577  KO=0     )
> findSelectors                                       (OK=611    KO=0     )
> findFesaFieldValuesByName                           (OK=566    KO=0     )
> findDevicesByQuery                                  (OK=1196   KO=0     )
> findAcceleratorByName                               (OK=1210   KO=0     )
> findDeviceClassesByQuery                            (OK=1220   KO=0     )
> findAllAccelerators                                 (OK=1210   KO=0     )
> findDeviceByName                                    (OK=1802   KO=0     )
> findDeviceClassVersionsByName                       (OK=1781   KO=0     )
> findDeviceClassVersionsByNameAndVersion             (OK=1781   KO=0     )
> findComputerByName                                  (OK=1200   KO=0     )
```

```
---- Global Informa
> request count                                      12577 (OK=12577  KO=0     )
> min response time                                      4 (OK=4      KO=-     )
> max response time                                   1215 (OK=1215   KO=-     )
> mean response time                                    86 (OK=86     KO=-     )
> std deviation                                        168 (OK=168    KO=-     )
> response time 50th percentile                         36 (OK=36     KO=-     )
> response time 75th percentile                         75 (OK=75     KO=-     )
> response time 95th percentile                        278 (OK=278    KO=-     )
> response time 99th percentile                        973 (OK=973    KO=-     )
> mean requests/sec                                 78.118 (OK=78.118 KO=-     )
---- Response Time Distribution ------------------------------------------------
> t < 500 ms                                         12197 ( 97%)
> 500 ms < t < 2000 ms                                 380 (  3%)
> t > 2000 ms                                            0 (  0%)
> failed                                                 0 (  0%)
```

Improving API throughput



- ***Horizontal scaling***
  adds resources to handle more client requests



- **Client throttling** (e.g. Leaky bucket algorithm)
  limits misbehaving clients by rejecting requests
- **Circuit breaker**
  stops requests in case of consecutive errors

- **Caching mechanism**
  reduces time needed to get data

To cache or not to cache

The premise of a cache - to store and provide already processed data

Benefits:

- limits unnecessary IOs → in many systems, physical IOs are the slowest operations
- load on the server and related services is reduced to minimum → output/data is processed only once

Drawbacks:

- increased complexity of the system (embedded cache vs standalone)
- eviction strategy* – especially difficult for complex system with mutable data
- challenge of consistency in distributed systems – every client should see the same state of cached data
  → there are solutions: e.g.: distributed cache like Apache Ignite, Redis

Cache eviction strategies:
- by time, e.g.: after n-seconds (TTL)
- by access frequency, e.g.: after 10k cache reads
- on-demand, e.g.: explicit cache purge
- by size, e.g.: max no. of elements in the cache

```
@Cacheable(CacheName.Constants.CACHE_ACCELERATOR)
public Accelerator findByName(String name) {   Complexity is 3 Everything is cool!
    return acceleratorMapper.toAccelerator(acceleratorRepository.findByNameIgnoreCase(name)
            .orElseThrow(() -> new EntityNotFoundException("Accelerator with name '" + name + "' was not found.")));
}
```

Caching is not a silver bullet for performance issues

Summary and outlook for Controls APIs

Designing any API is a real challenge:

- Intuitive, easy to use, without unnecessary complexity, consistent with existing API.
- Requires understanding of current needs and be open to follow the future needs

Development work of hundreds of people is based on Controls APIs

- Services, applications, scripts – all are based on our APIs

As software technology evolves, our APIs must follow, whilst remaining as stable as possible

- Technology obsolesce of Java RMI is reason to renovate to more modern solutions like ReST
- The renovation must limit any negative impact on our existing users, yet it is an opportunity to:
  - Facilitate programming technology-agnostic access for clients using other languages (e.g. Python)
  - Increase consistency across our multitude of client APIs