Contrôle-commande sur le projet LISA

Claude-Alban RANÉLY-VERGÉ-DÉPRÉ, Mathieu DUPONT, Éric KAJFASZ

Le projet LISA

Laser Interferometer Space Antenna (**LISA**) :
Antenne à ondes gravitationnelles

> En lien avec la thématique “la physique des astroparticules et la cosmologie” de l’IN2P3

Projet européen ESA

En collaboration avec la NASA (?)

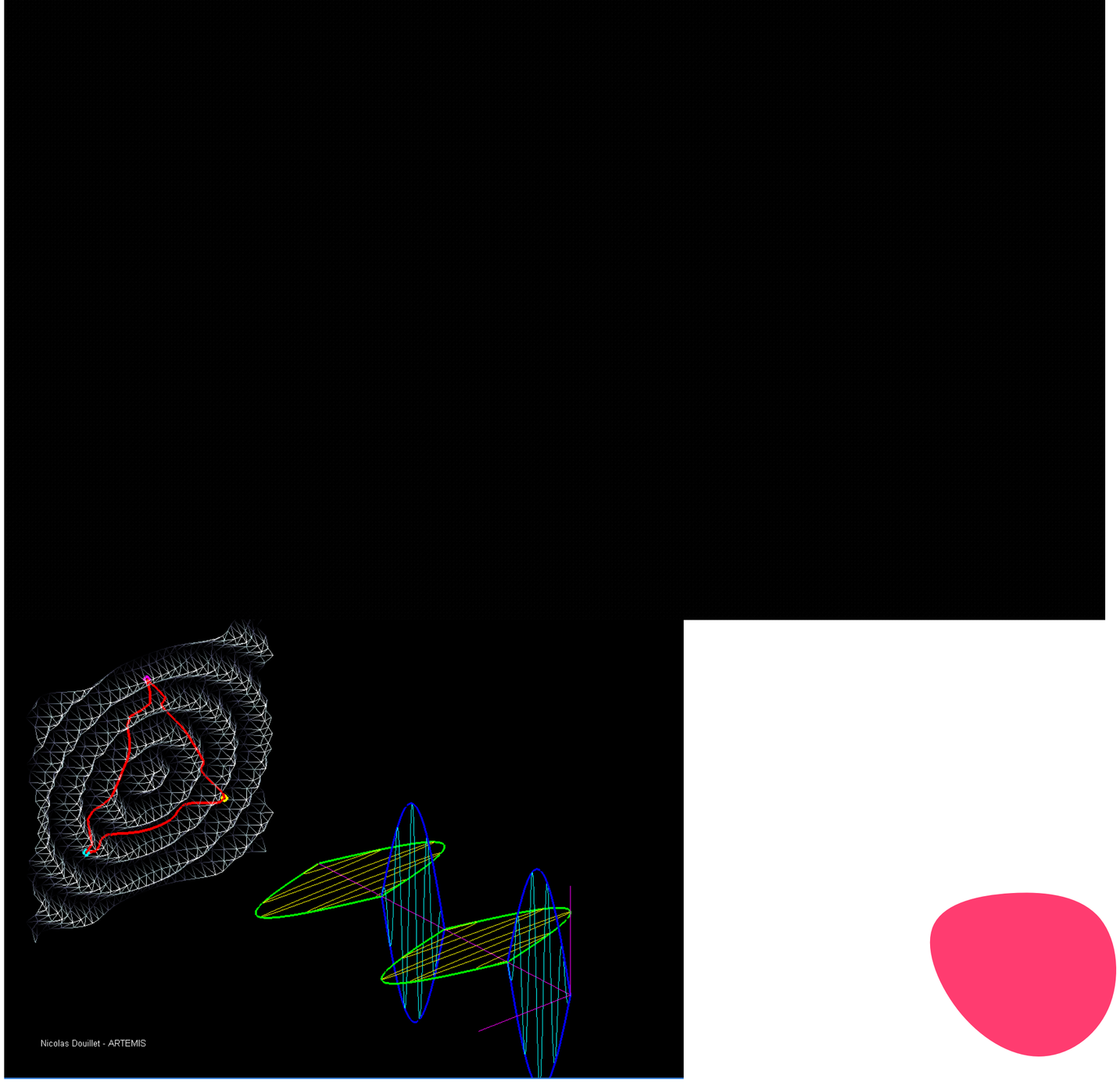
Le projet LISA

Laser Interferometer Space Antenna (**LISA**) :
Antenne à ondes gravitationnelles

En lien avec la thématique “la physique des astroparticules et la cosmologie” de l’IN2P3

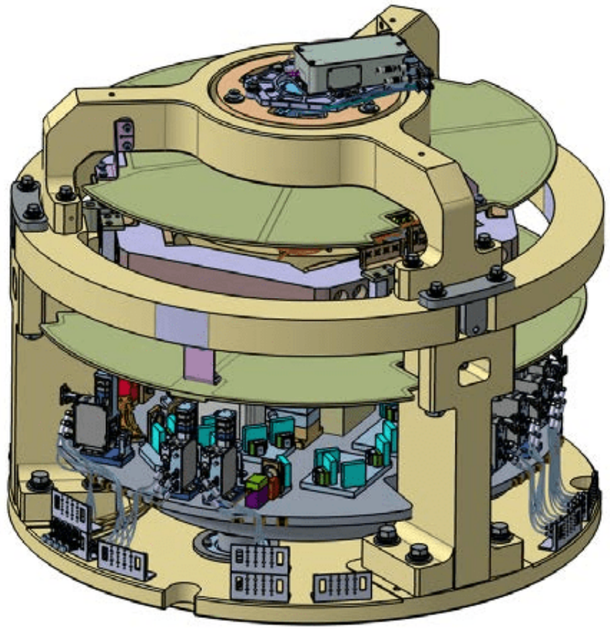
Projet européen ESA

En collaboration avec la NASA (?)



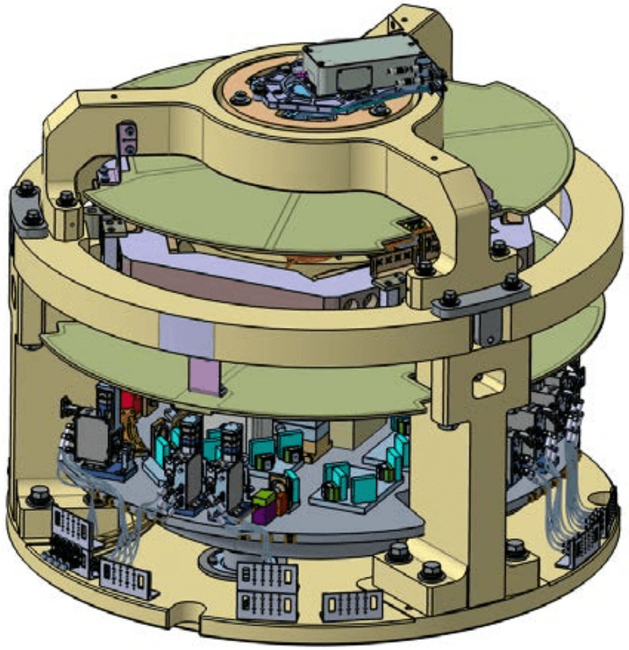
La part de la France

IDS : Interferometric Detection System

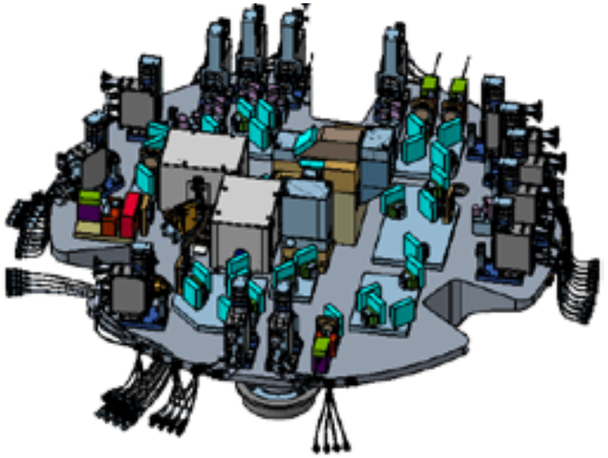


La part de la France

IDS : Interferometric Detection System

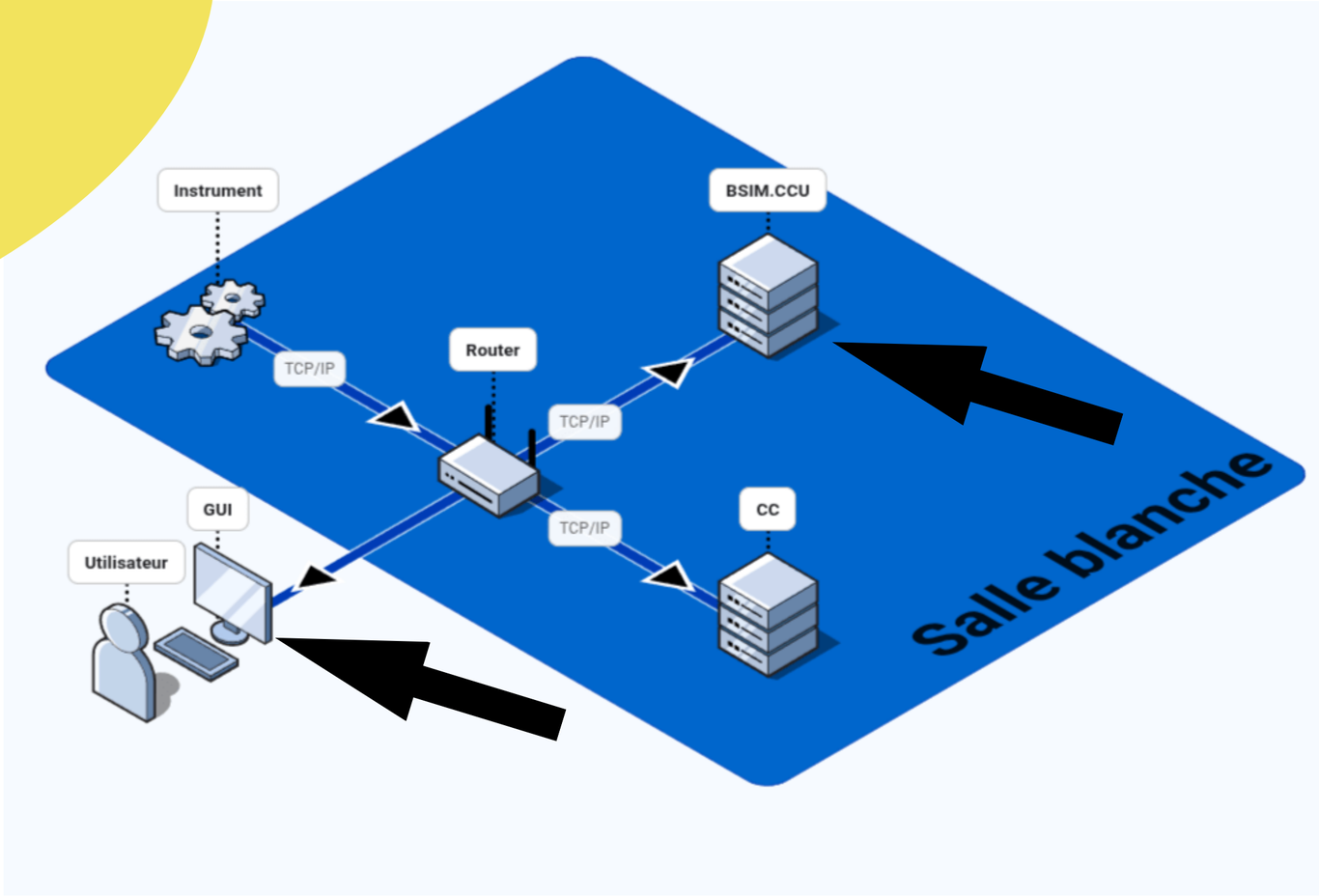


BSIM : Beam Simulator



Cahier des charges

- Plusieurs sous-systèmes à piloter
- Système distribué
- Système stable



I Architecture

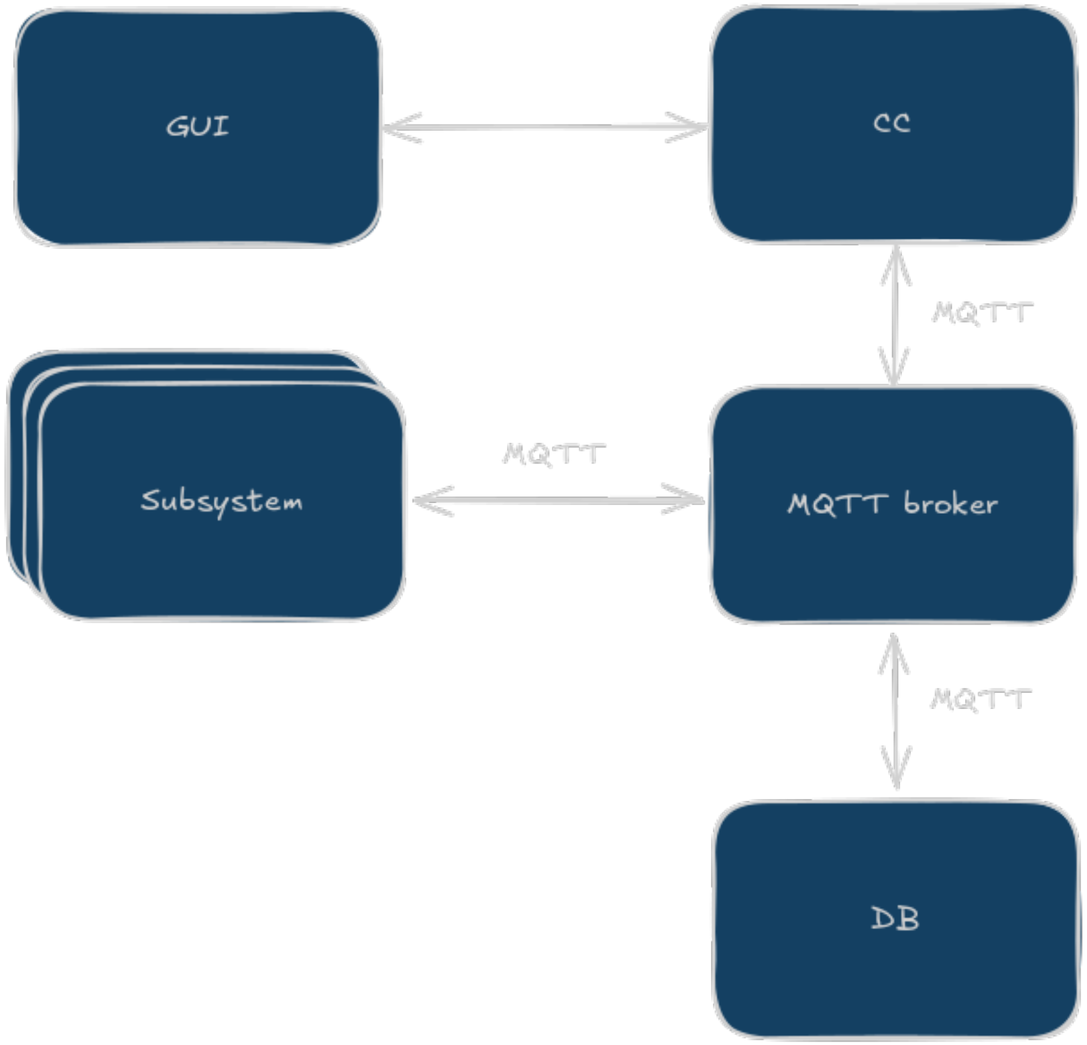
II Retour d’expériences

III Perspectives

Architecture de notre solution

Différents sous-systèmes indépendants

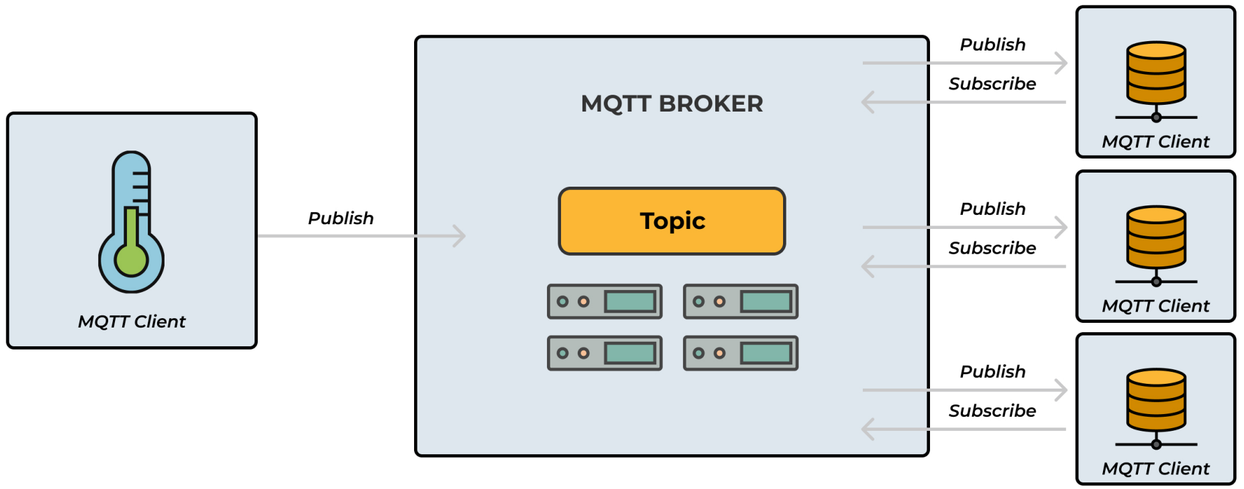
⇨ Binaires indépendants



Architecture distribuée

Protocole retenu : MQTT

- Architecture pub/sub
- Fonctionnalités avancées (QoS, Last Will)
- Ouvert
- Léger



Contenu des paquets

Technologie retenue : JSON

- Flexibilité des messages
- Pas de contraintes de taille pour les messages mais BSON est une option
- Facilement intégrable dans n'importe quel langage de programmation

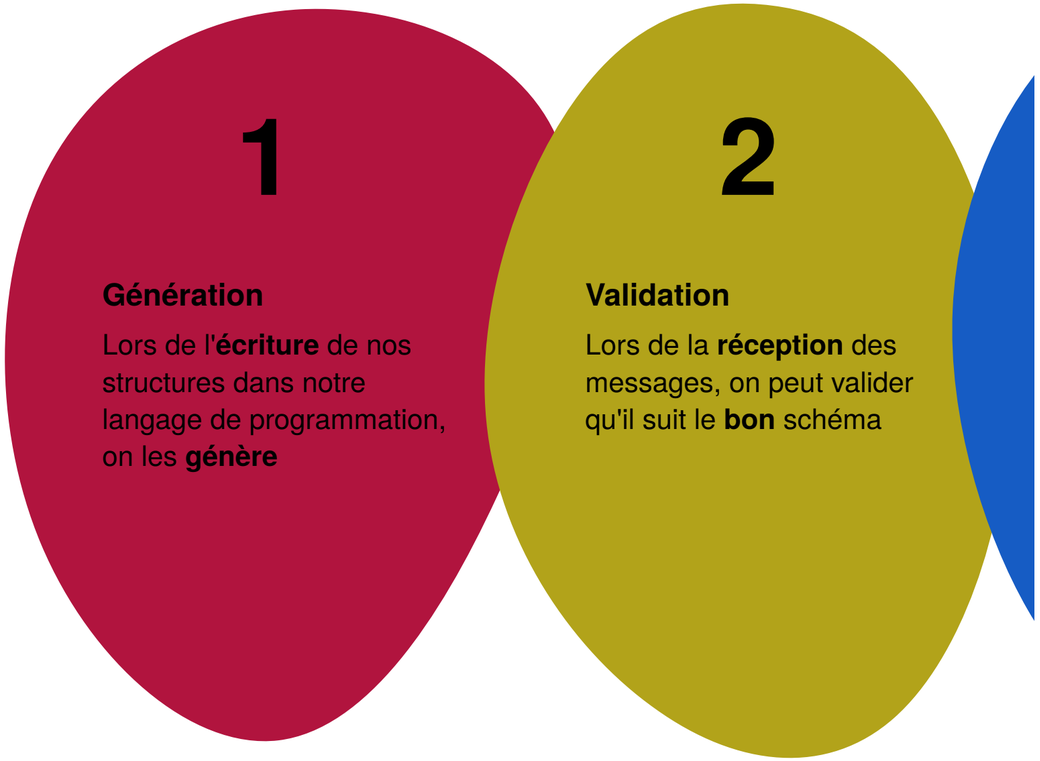
```
{
    "menu": {
        "id": 1,
        "value": true,
        "popup": {
            "menuitem": [
                { "value": "New", "onclick": "CreateNewDoc()" },
                { "value": "Open", "onclick": "OpenDoc()" },
                { "value": "Close", "onclick": "CloseDoc()" }
            ]
        }
    }
}
```

Un format flexible avec *aucune* spécification c'est un mal de tête pour le prochain à s'en occuper

– ~~Albert Einstein~~ Un ingénieur fatigué

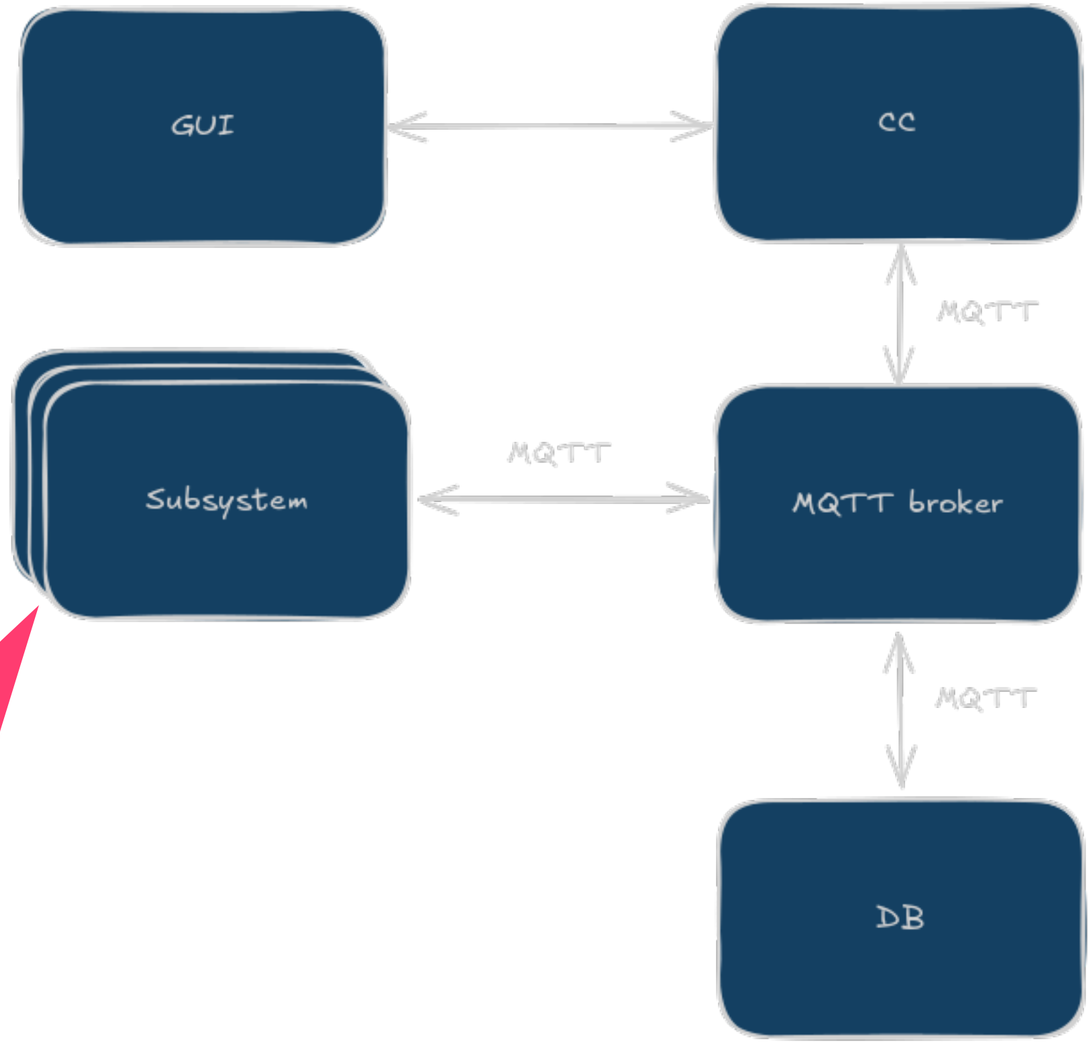
Spécifications des messages JSON

Spécifications des messages JSON




```json
{
    "$schema": "http://json-schema.org/draft-07/schema#",
    "type": "object",
    "properties": {
        "menu": {
            "type": "object",
            "properties": {
                "id": {
                    "type": "integer"
                },
                "value": {
                    "type": "boolean"
                },
                "popup": {
                    "type": "object",
                    "properties": {
                        "menuitem": {
                            "type": "array",
                            "items": {
                                "type": "object",
                                "properties": {
                                    "value": {
                                        "type": "string"
                                    },
                                    "onclick": {
                                        "type": "string"
                                    }
                                },
                                "required": ["value", "onclick"]
                            }
                        }
                    },
                    "required": ["menuitem"]
                }
            },
            "required": ["id", "value", "popup"]
        }
    },
    "required": ["menu"]
}
```

Architecture de notre solution



Le langage de programmation

Langage développé initialement à Mozilla dans le cadre de Firefox pour remplacer C++

- Langage qui est mature (+10 ans depuis 1.0)
- Intégration avec C facile
- Multithreading *sans bugs*
- Performances quasi-égale au C avec “une syntaxe plus proche de Python”

Google (2025)

"For medium and large changes, the rollback rate of Rust changes in Android is **~4x lower than C++"**

Microsoft (2019)

"we think that Rust represents **the best alternative to C and C++** currently available."

Études scientifiques de comparaison de langages

Pourquoi ce langage ?

- Pas de réalisation existante
- Stabilité
- Possibilité d'utiliser le même langage pour toutes les utilisations (Binaire, GUI déportée, Application Web)
- Interaction *simple* avec librairies constructeurs (écrites en C)

Exemple

```rust
#[derive(Debug, Deserialize, Serialize, JsonSchema)]
pub struct Message {
    /// Date at which the message was sent
    pub run_id: u64,
    /// The number of non-leap nanoseconds since January
    /// 1970 0:00:00 UTC (aka "UNIX timestamp")
    /// in the current date, time and offset from UTC.
    pub timestamp: i64,
    /// The JSON encoded data retrieved from the tas
    pub data: Measurements,
}

/// The named aggregation of the measurements made by the
#[derive(Debug, Deserialize, Serialize, JsonSchema)]
pub struct Measurements {
    /// The payload of the measurements made by the syste
    /// Using a BTreeMap here to be synchronized with `Da
    #[serde(flatten)]
    pub data: BTreeMap<String, Vec<Measurement>>,
}
```

```json
{
    "run_id": 0,
    "timestamp": 1727790724,
    "data": {
        "temperature_one": {
            "timestamp_fine": 1727790724,
            "raw_value": 50,
            "value": 20.4,
            "unit": "C"
        },
        "temperature_two": {
            "timestamp_fine": 1727790724,
            "raw_value": 60,
            "value": 20.6,
            "unit": "C"
        }
    }
}
```

Exemple

```rust
#[derive(Debug, Deserialize, Serialize, JsonSchema)]
pub struct Message {
    /// Date at which the message was sent
    pub run_id: u64,
    /// The number of non-leap nanoseconds since January
    /// 1970 0:00:00 UTC (aka "UNIX timestamp")
    /// in the current date, time and offset from UTC.
    pub timestamp: i64,
    /// The JSON encoded data retrieved from the tas
    pub data: Measurements,
}

/// The named aggregation of the measurements made by the
#[derive(Debug, Deserialize, Serialize, JsonSchema)]
pub struct Measurements {
    /// The payload of the measurements made by the syste
    /// Using a BTreeMap here to be synchronized with `Da
    #[serde(flatten)]
    pub data: BTreeMap<String, Vec<Measurement>>,
}
```

JSON Schema

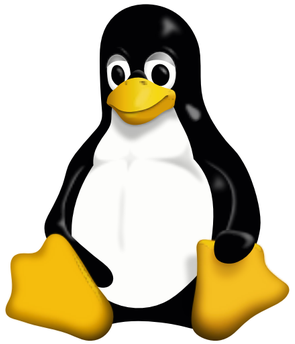
```json
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "title": "Message",
  "type": "object",
  "properties": {
    "data": {
      "description": "The JSON encoded data retrieved from the tas",
      "$ref": "#/$defs/Measurements"
    },
    "run_id": {
      "description": "Date at which the message was sent",
      "type": "integer",
      "format": "uint64",
      "minimum": 0
    },
    "timestamp": {
      "description": "The number of non-leap nanoseconds since January 1, 1970 0:00:0
      "type": "integer",
      "format": "int64"
    }
  },
  "required": [
    "run_id",
    "timestamp",
    "data"
  ],
  "$defs": {
    "Measurement": {
      "description": "Physical measurement made by the system",
      "type": "object",
      "properties": {
        "raw_value": {
          "description": "Raw value sent by the test equipment",
          "type": "number",
          "format": "double"
```

Un choix souvent peu disponible

le système d'exploitation

Notre choix s'est porté sur un système :

1. léger
2. peu intrusif
3. adaptable à nos besoins (open-source)



Un composant en particulier : systemd

Un outil (système d'initialisation) qui permet de définir *un ordre de démarrage* pour les applications, tout en offrant de nombreux autres *déclencheurs* :

- service : pour un service/démon
- socket : pour une socket réseau (de tous types : UNIX, fichier …)
- mount : pour un système de fichiers (exemple : home.mount)
- swap : comme mount mais pour les partitions d'échanges
- automount : comme mount mais pour un système de fichiers monté à la demande
- device : pour un périphérique
- timer : pour l'activation basée sur une date
- path : pour l'activation basée sur des fichiers ou des répertoires
- target : macro-unité qui permet de grouper plusieurs unités (exemple : multi-user.target pour définir une cible)

(liste non exhaustive)

Bonus # 1 : Fonctionnalités inattendues — systemd-mkosi

Contruire une ISO *personalisée* d'une machine

- Fichiers de configuration
- Binaires
- Mots de passe
- ...

Interessé(e) ?

Bonus #2 : Fonctionnalités inattendues — systemd-sysupdate

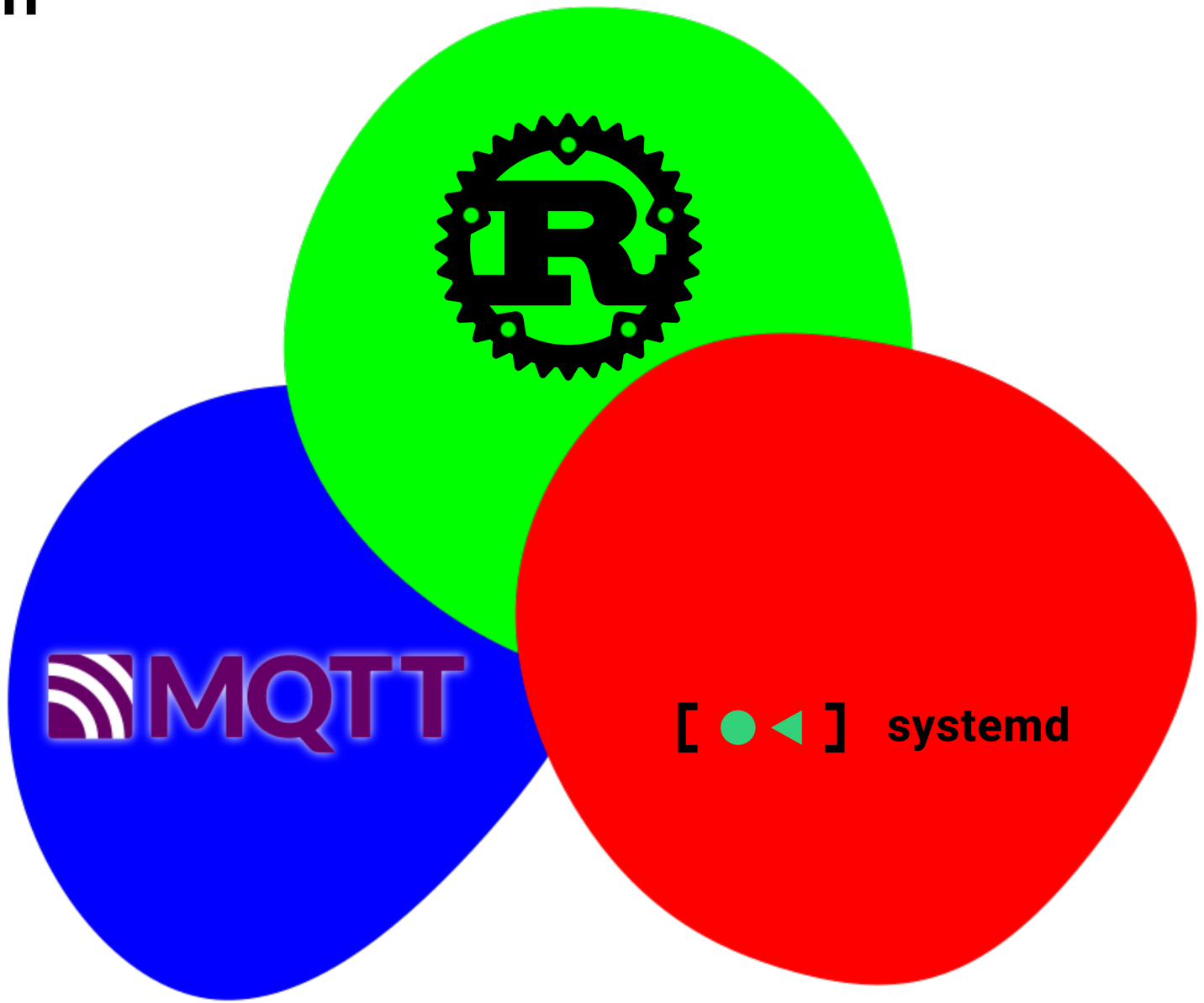
Un gestionnaire de mise à jour

avec possibilité de retour en arrière atomique (partition A/B)

Une version ratée ? Redémarrez votre appareil et
c'est comme si rien ne s'était passé!

Interessé(e) ?

Conclusion



Merci!

Questions?

carvd@cppm.in2p3.fr