SVT WP2 testing software

SVT WP2 SW developer group

- Ivan Amos Cali*, Yasser Corrales Morales*,
- Oleksandr Korchak**, Anhelina Kostina**,
- Liliana Teodorescu***, Lochana Ranatunge***

*** Brunel University of London

** Czech Technical University in Prague

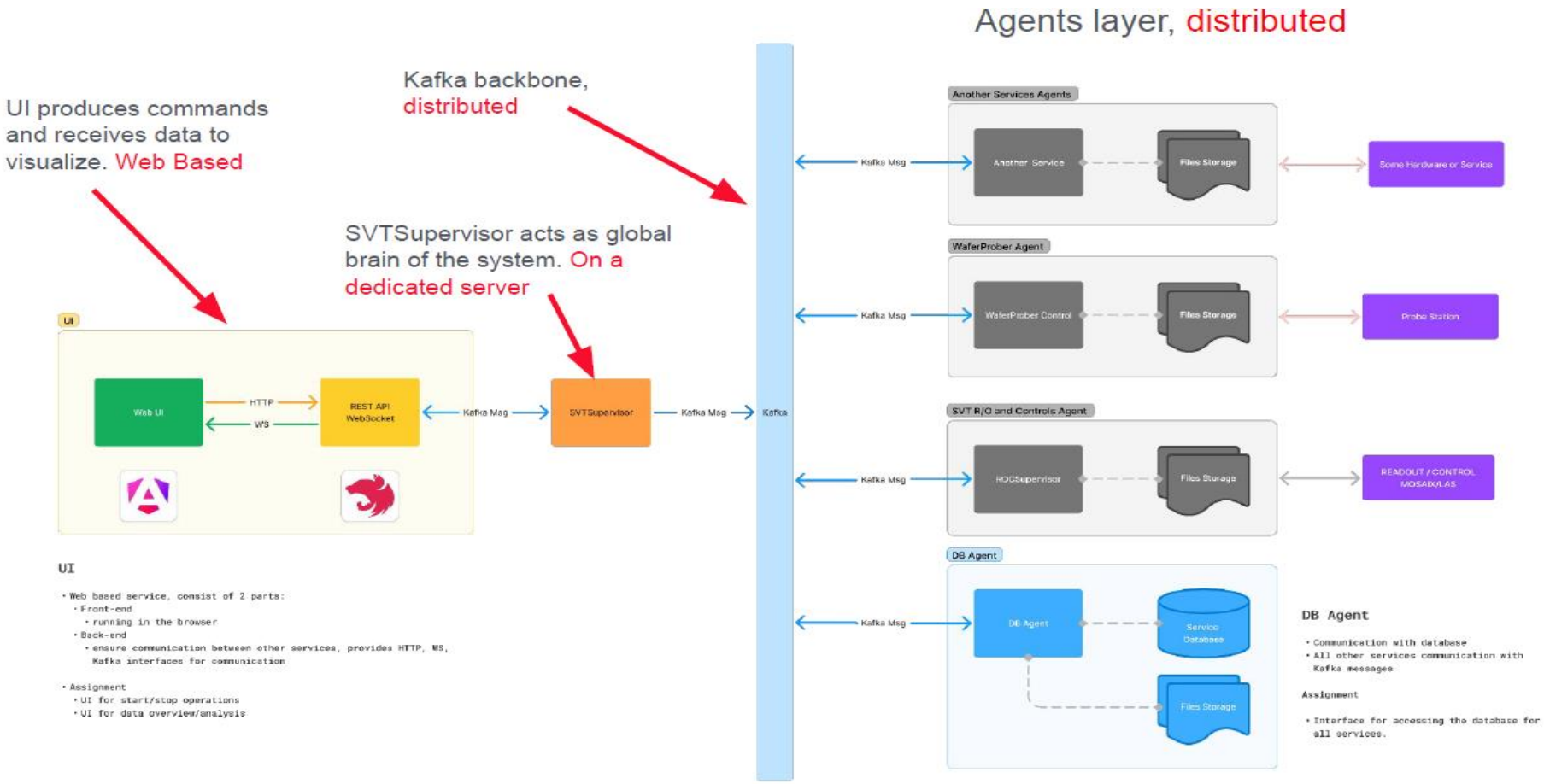
* Massachusetts Institute of Technology

WP2 SW group scope

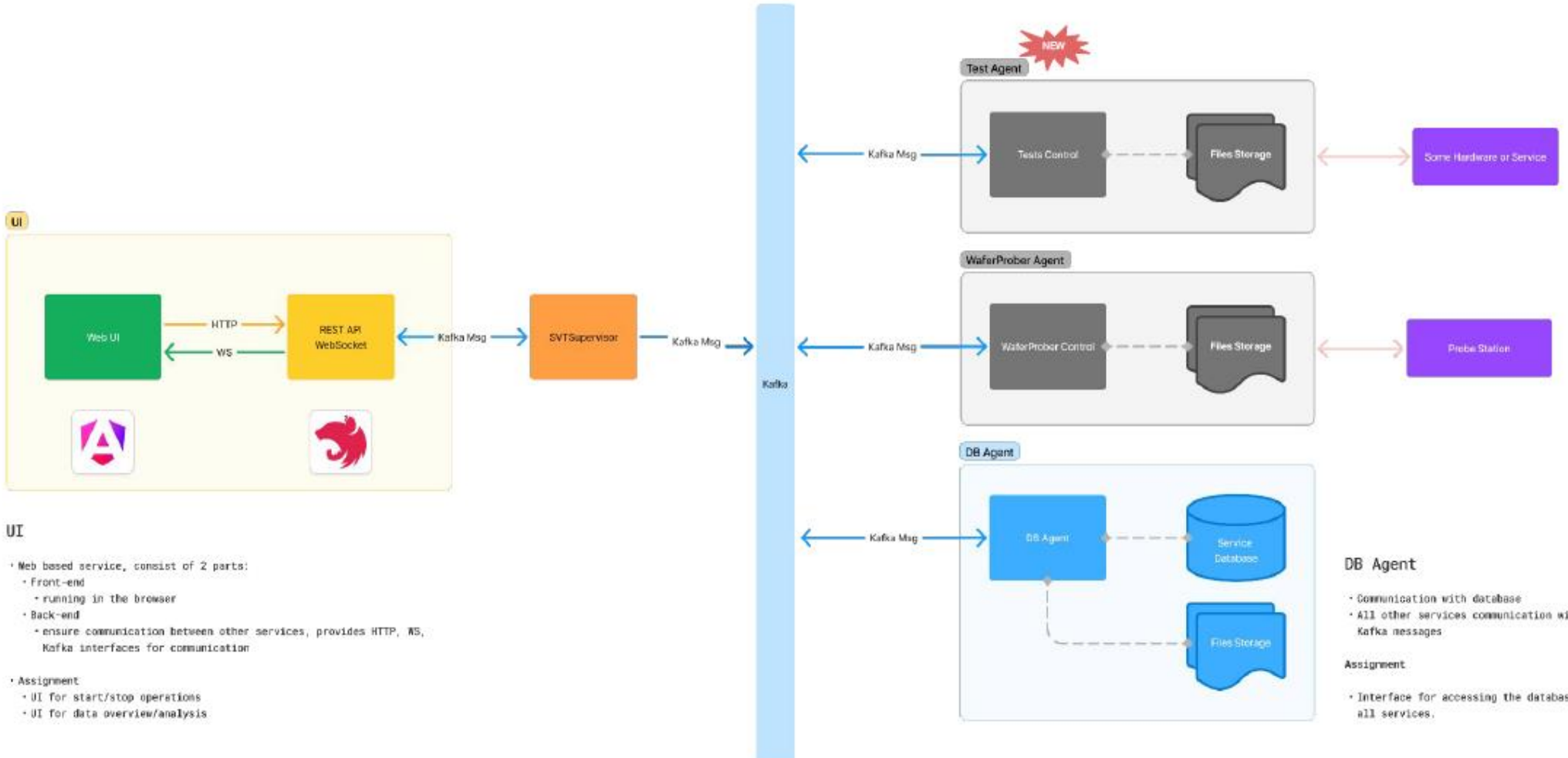
- Create a shared testing framework for all sub-components, expandable to production.
- Develop control and test software for each sub-component.
- Provide analysis tools for every testing stage.
- Support the Readout and Strategy team.
- Interact with the WP responsible for the final software integration.

SW Block diagram

Apache Kafka
- Distributed data streaming platform
- Transports data between systems decoupling the systems
- Responsible for receiving and sending the data through messages



Current status



UI

- Web based service, consist of 2 parts:
  - Front-end
    - running in the browser
  - Back-end
    - ensure communication between other services, provides HTTP, WS, Kafka interfaces for communication
- Assignment
  - UI for start/stop operations
  - UI for data overview/analysis

DB Agent

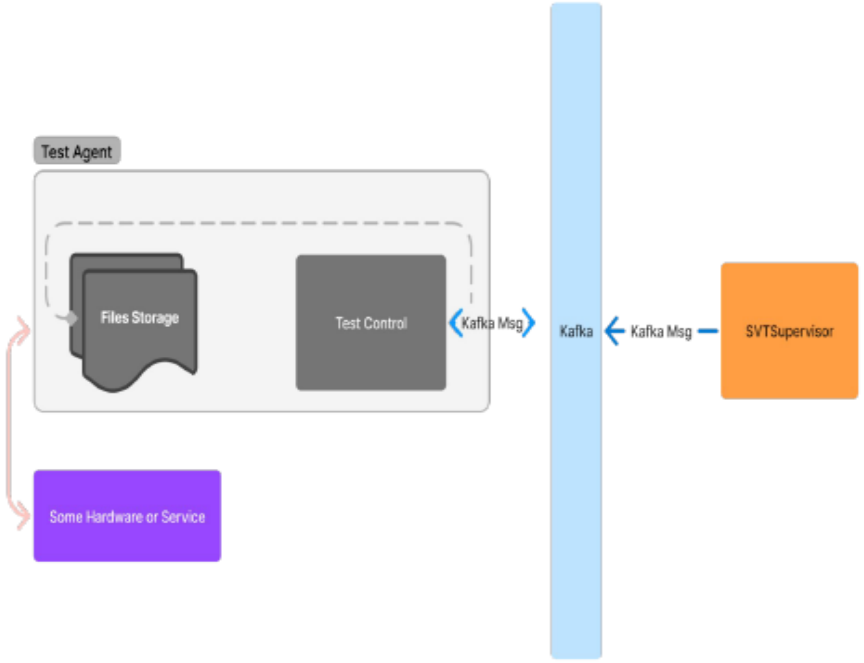
- Communication with database
- All other services communication with Kafka messages

Assignment

- Interface for accessing the database for all services.

Test Agent

Runs and controls dedicated test operations on its own, tracks the outcomes, and delivers consistent data for further analysis



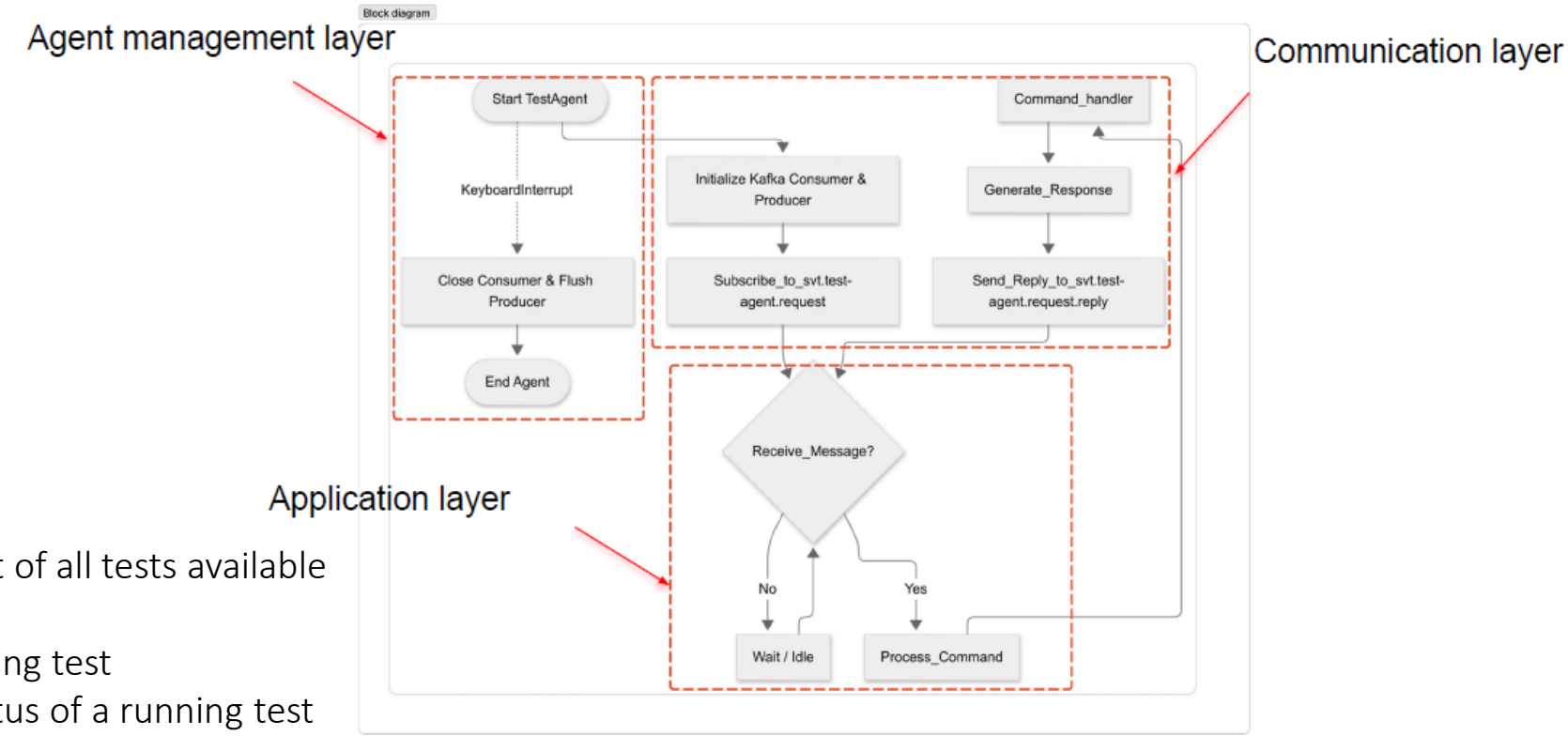
Test Agent – Why?

- Dedicated service to **run tests independently**
- Avoids overloading different agents
- Provides **scalability & reliability**
- Standardized results & telemetry to DB

Innovation

- **Test Agent = execution of any electric tests**
- **Plugin-based**: easy to add new tests
- **JSON with trace IDs** → full traceability
- Flexible: single commands or automated workflows

Test Agent architecture



COMMANDS:

- GetAllTests: Get the list of all tests available
- RunTest: Initiate a test
- AbortTest: Stop a running test
- TestStatus: Get the status of a running test
- RunLoopTest: Run a particular test iteratively
- RunTestPlan: Run a sequence of tests

Results of commands directed through Kafka messages to UI and DB

Command registry

```python
# ------------------------------------------------------------------------------
# Default command handlers
# ------------------------------------------------------------------------------
DEFAULT_COMMAND_HANDLERS: Dict[str, Callable] = {
    "GetAllTests": ALL_HANDLERS.get("GetAllTests"),
    "RunTest": ALL_HANDLERS.get("RunTest"),
    "AbortTest": ALL_HANDLERS.get("AbortTest"),
    "TestStatus": ALL_HANDLERS.get("TestStatus"),
    "RunLoopTest": ALL_HANDLERS.get("RunLoopTest"),
    "RunTestPlan": ALL_HANDLERS.get("RunTestPlan"),
}


# ------------------------------------------------------------------------------
# Chip-specific command overrides
# ------------------------------------------------------------------------------
CHIP_COMMAND_OVERRIDES: Dict[str, Dict[str, Callable]] = {
    "SLDO": {
        # Example of adding a chip-specific handler:
        # **newTest("CustomSLDOTest")
    },
    "NVG": {
        # Add NVG-specific overrides here
    },
}
```

SLDO tests

SLDO Table

1. SLDO ID (unique)
2. Wafer ID (unique)
3. Testing institute (assuming all tests are done at one location)
4. Arrival date
5. Equipment

Test conditions Table

1. SLDO ID (unique)
2. Test ID (unique)
3. Test name (Eg: Power ramp up/PSRR/Power Ramp rate/DAC Scan/Overcurrent/Irradiation)
4. Mode (Eg: Mode 0 and Mode 1)
5. Voltage ramp rates (Default value 3.1kV/s (from chip specification))
6. Load capacitance (4 enums)
7. Load current (3 enums)
8. Temperature (4 enums)

Power Ramp up table

Initial power supply test.

1. Test ID (Unique)
2. V_in target (Default value 1.55V (from chip specification))
3. I_in limit (Default value 1.5A(from chip specification))
4. V_out

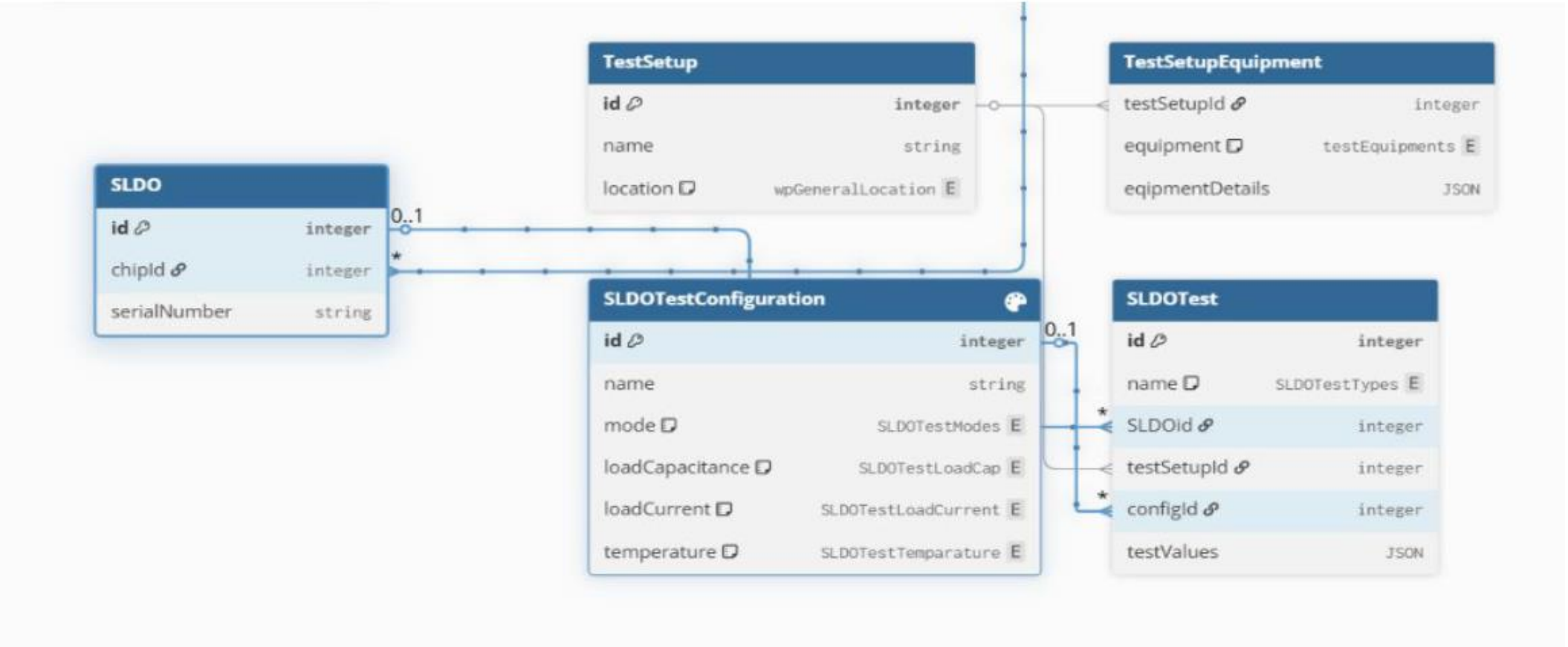
PSRR Table

PSRR- Power supply rejection ratio(dB).

1. Test ID (Unique)
2. V_in target (Default value 1.55V (from chip specification))
3. I_in limit (Default value 1.5A(from chip specification))
4. Signal amplitude
5. Signal frequency
6. V_out amplitude.
7. PSRR

Rest of measurements in the document uploaded on indico with these slides

DB tables for SLDO



Test registry

```
CHIP_TEST_DEFINITIONS = {
    "SLDO": {
        "default": {
            "testConfiguration": {
                "mode": {"unit": None, "enum": [0, 1]},
                "loadCapacitance": {"unit": "nF", "enum": [10, 100, 1000, 10000]},
                "loadCurrent": {"unit": "mA", "enum": [40, 500, 900]},
                "temperature": {"unit": "C", "min": -20, "max": 125},
            },
            "testValues": {
                "inputs": {
                    "vInTarget": {"unit": "V", "min": 0, "max": 1.55},
                    "iInLimit": {"unit": "A", "min": 0, "max": 1.5},
                    "rampRate": {"unit": "kV/s", "default": 3.1},
                },
                "outputs": {
                    "vOut": {"unit": "V"},
                }
            }
        },
```

```
"tests": {
    "PowerRampUp": {
        "testConfiguration": {},
        "testValues": {}
    },
    "PSRR": {
        "testConfiguration": {},
        "testValues": {
            "inputs": {
                "signalAmplitude": {"unit": "mV"},
                "signalFrequency": {"unit": "Hz"},
            },
            "outputs": {
                "psrr": {"unit": "dB"}
            }
        }
    },
```

Example commands – Run Test

Examples:

Option 3 - RunTest suffix New

Example Value | Schema

```
{
  "command": "RunTest:New",
  "testId": 5,
  "data": {
    "chipId": 301,
    "testSetupId": 88,
    "configId": 66,
    "params": {
      "chipName": "SLDO",
      "testName": "PowerRampUp",
      "TestConfiguration": {},
      "inputs": {}
    }
  }
}
```

Examples:

Option 3 - RunTest suffix PreDef

Example Value | Schema

```
{
  "command": "RunTest:PreDef",
  "testId": 6,
  "data": {
    "chipId": 302,
    "testSetupId": 89,
    "configId": 67,
    "params": {
      "inputs": {}
    }
  }
}
```

- Commands are sent in **JSON format** to request tests.
- Two options are supported:
  - **New test** – defines chip, setup, configuration, and parameters.
  - **Predefined test** – runs a stored configuration with minimal input.
- Structure ensures **standardized communication** between agents and database.

Results of the command (reply message)

Test Agent :: Replies / Error handing

Media type: application/json

Examples: Test succeeded reply

Controls Accept header.

Example Value | Schema

```
{
  "test_id": 1,
  "type": "RunTestReply",
  "testStatus": "TestSuccess",
  "testValues": {
    "inputs": {
      "vInTarget(V)": 1.55,
      "iInLimit(A)": 1.5,
      "rampRate(kV/s)": 3.1
    },
    "outputs": {
      "v_out(V)": 1.3
    }
  }
}
```

Success

Test failed

Media type: application/json

Examples: Test failed reply

Controls Accept header.

Example Value | Schema

```
{
  "test_id": 1,
  "type": "string",
  "testStatus": "TestFail",
  "testError": "string"
}
```

Replies are returned in **JSON format**.

Each reply contains the **test ID, status, and details**.

Two main outcomes:
Success → includes measured input/output values.

Failure → includes error information for debugging.

Ensures **clear and standardized communication** of results.

Further work

Development

Test Agent – refine the agent, the commands, the test definitions

UI – add functionality for tests

connect Test Agent with DB and UI

Usage

Use it for SLDO tests (at least in Brunel) to test the software and the workflow

- upload test results “manually”
- interface the test system with this software and perform the tests with it (if possible)

Plans for ER2 testing
(presented by SVT WP2 convenors)

ER2 Wafers distributions

Following the requests of the various institutions we propose the following wafers distribution:

- November 2025:
  - 12 wafers received at CERN and 1 will be diced immediately to produce 5 MOSAIX and several babyMOSAIX cards
  - 2 wafers are sent to ONRL for wafer probing testing
  - 9 wafers remain at CERN/MIT for wafer probing testing

- April 2026:
  - 1 wafers to LBL for dicing and thinning tests
  - 8 wafers to WP4 for producing a qualification model (diced and thinned)
  - Test System preparation for ER3/LAS
    - 1 wafer at CERN/MIT shared with ONRL
    - 1 wafer at BNL (if site ready) shared with BRUNEL (if site ready)
    - PAD wafers can be used to exercise the software

babyMOSAIX and MOSAIX produced with the components for WP4 will be mounted on carrier cards