Automation & Robotics at Beamlines:

Lessons from the Field

Automated Sample Handling for Remote Access in Operando Catalysis Experiments at Scattering Beamline P65 @ PETRA III

By: Shrouk Ehab

Lund, 23rd Jan 2026

JOURNEY:

From Concept to Operation

My Contribution to ROCK-IT

Developing an automated sample changer as a key ROCK-IT use case, with end-to-end responsibility across the entire automation pipeline.

01

Hardware Integration

Mechanical systems and sensor deployment

02

Robot Control & Calibration

Precision motion planning and positioning

03

ROS 2 Architecture

Action-based software framework

04

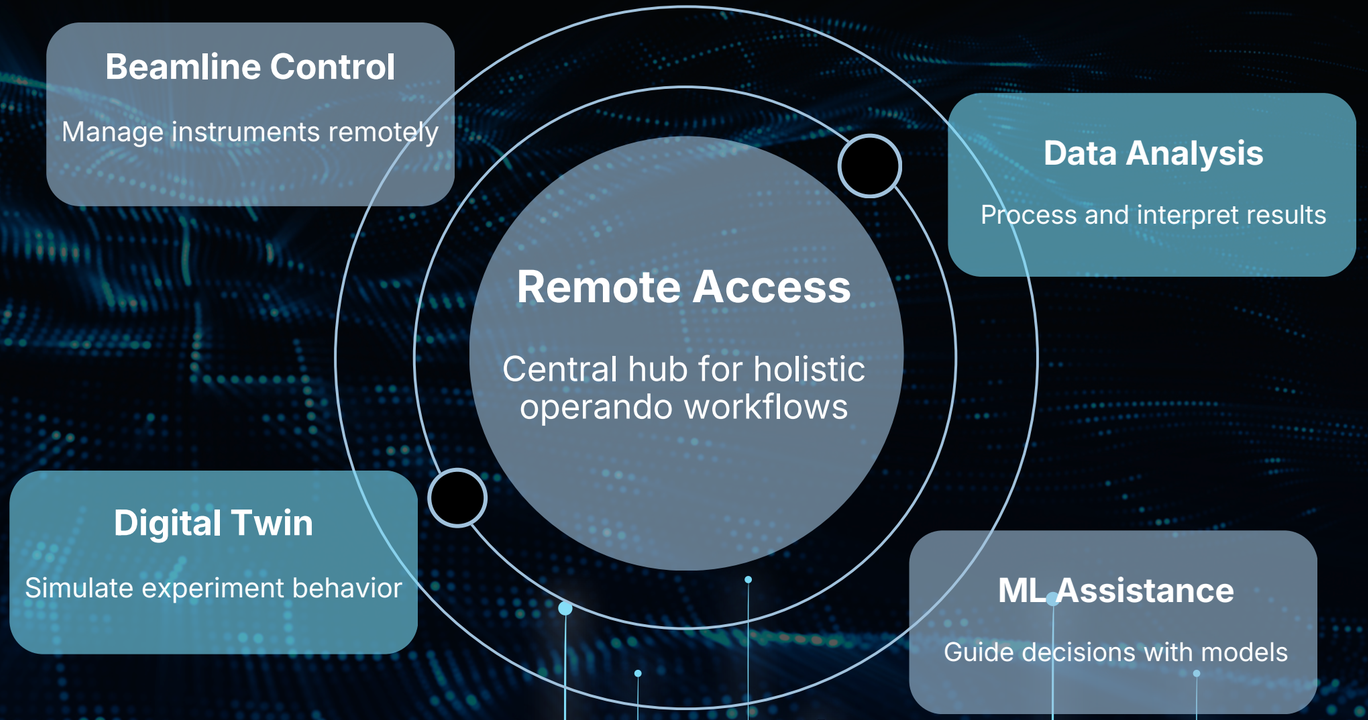
Beamline Commissioning

Real-world testing and deployment

ROCK-IT

WHAT'S ROCK-IT ?

ROCK-IT (Remote, Operando-Controlled, Knowledge-driven, and IT-based) is a Helmholtz-funded initiative that automates complex operando catalysis experiments, enabling scientists worldwide to conduct research with unprecedented accessibility and scalability.



DESY.

Karlsruhe Institute of Technology

HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF

Helmholtz Zentrum Berlin

HELMHOLTZ

Challenges in Operando Catalysis

High-throughput & complex environment

Precise control of reaction conditions

Hazardous Gas Handling

Beam stability & alignment

Real-Time Data Processing

Our Aim

Motivation

Holistic Workflow for Complex Operando Experiments

Common look & feel across institutes for seamless operation

GUI-based experiment control and real-time analysis

Modular components with unified interfaces

Complete data lifecycle from acquisition to archiving

Robust cyber security protecting data and infrastructure

Why Beamline Automation Is Special ?

Not Industrial Automation

Core Distinctions

Mixed Environments

Precision mechanics, human interaction, and radiation constraints coexist.

Low-Volume, High-Variance

Samples are often unique, requiring adaptable handling and analysis protocols.

Frequent Reconfiguration

Experiments change often, demanding rapid system adjustments.

Tight Spatial Constraints

Limited space around the beamline necessitates compact and efficient designs.

Scientific Operators

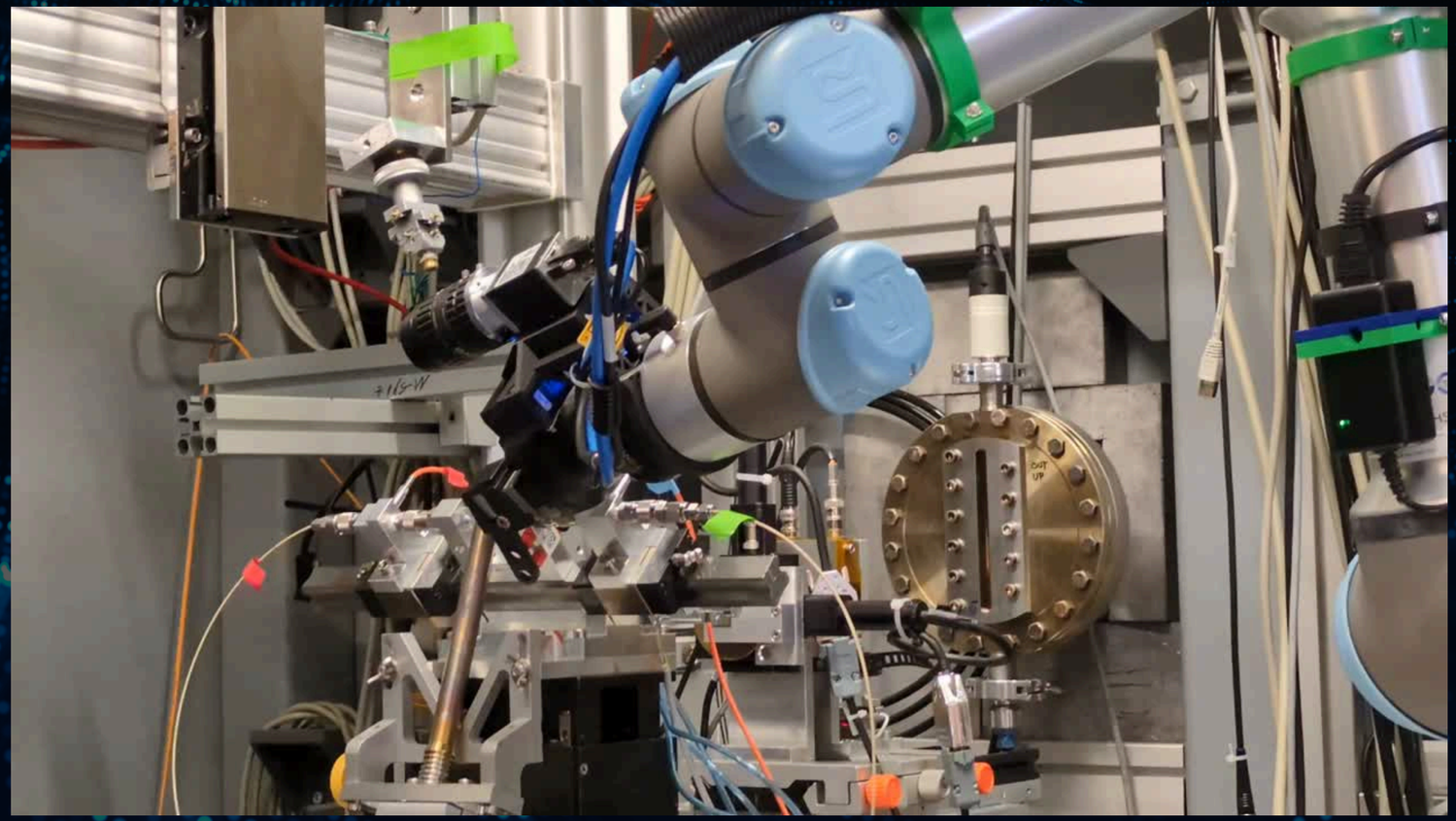
Users are scientists, not industrial operators, requiring intuitive interfaces and flexibility.

Automated Sample Handling System

Overview

Hardware Components:

- UR10e Robotic Arm with RobotIQ 2F-85 gripper for versatile sample handling.
- Basler Camera for accurate QR-code scanning and precise sample localization.
- Pneumatic Clamping Station providing gas connections and controlled heating for operando conditions.
- 6-Slot Sample Magazine enabling parallelized and high-throughput experiments.

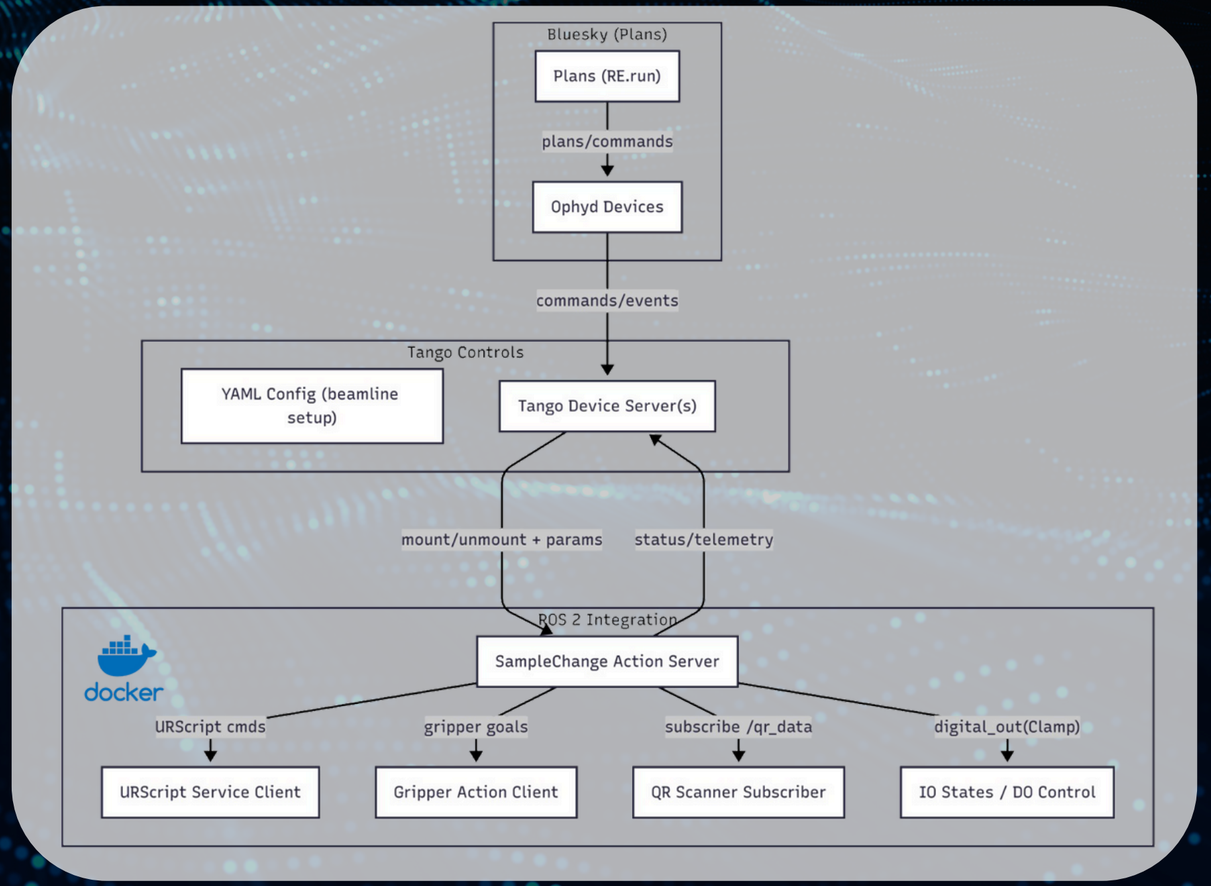


Automated Sample Handling System

Overview

Software & Control Architecture

- Bluesky: High-level experiment orchestration and comprehensive data management.
- Tango Control System: Coordinated control of instruments including the robot, environmental parameters, and detectors.
- ROS 2: Integrated pipeline for robotic motion planning, perception, and feedback control, ensuring reproducibility and user-friendly automation.



How ROS 2 Served Our System ?

ROS 2 (Robot Operating System 2) proved to be a foundational choice for our beamline automation, offering critical capabilities that addressed the unique demands of our complex scientific environment. It's more than just a framework; it's an ecosystem designed for robust, distributed robotics.

Why ROS 2 Worked ?

Hardware Abstraction

Seamless integration of diverse hardware components like robot arms, grippers, and vision systems.

Action-based Workflows

Manages complex, long-running tasks such as automated sample changes with dedicated action servers.

Clear Separation of Concerns

Promotes modular code, making development, testing, and maintenance more efficient.

Modularity & Scalability

Native support for expanding the system and adapting to future autonomous beamline requirements.

Strong Ecosystem

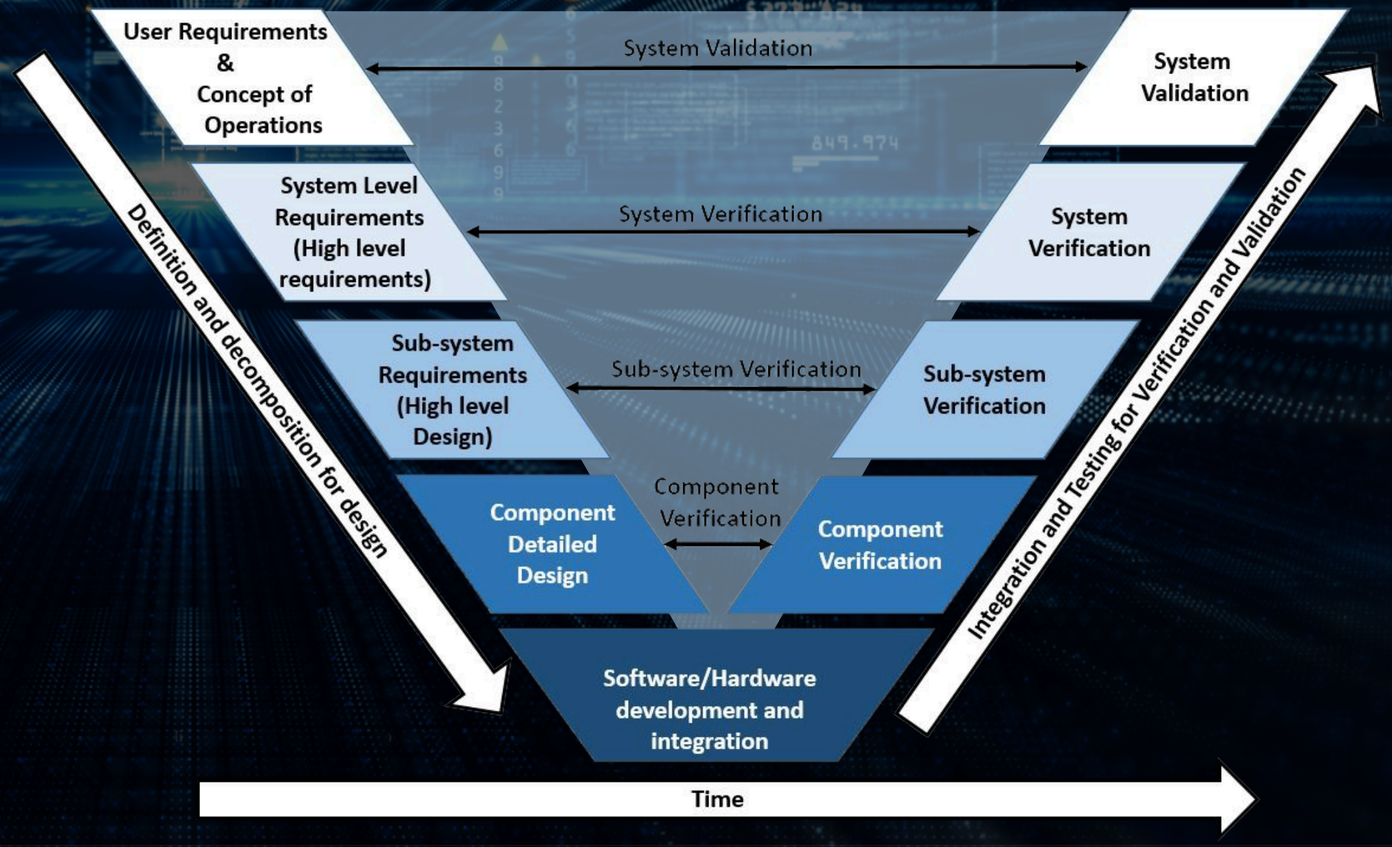
Access to a rich community and powerful tools for robotics development and perception.

An Honest Note

Debugging distributed systems is hard and requires stringent discipline in defining and maintaining interfaces between components.

The V-Model for Mechatronics Systems

Let's delve into the V-Model, a structured development process that integrates specification, design, integration, and validation phases. This framework is particularly powerful for complex mechatronic systems, ensuring quality and traceability throughout the project lifecycle.



Phase 1: Requirements

The "Why" Trap

The initial phase of any project, especially in scientific automation, is fraught with the challenge of translating broad scientific desires into actionable, technical specifications.

The Lesson: A signed requirement document is your best friend. It formalizes needs and prevents scope creep.

Phase 2: Design

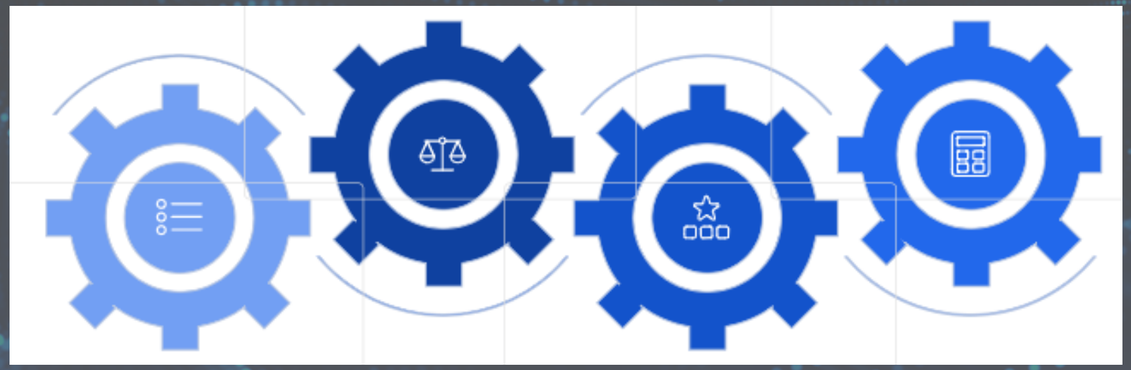
Navigating the Component Maze with Structured Decision-Making

From Subjective Choice to Objective System Design

To move beyond arbitrary selections, we adopt a systematic 4-step framework based on Multi-Attribute Decision Making (MADM) [2]. This ensures that every component choice is thoroughly evaluated against our defined requirements, reducing risk and optimizing performance.

Assign Weights
Rank criteria on a 1–5 importance scale

Calculate & Decide
Compute weighted scores and select best option



Define Criteria
List requirements and component alternatives

Score Options
Rate each component against every criterion

ROCK-IT

Phase 2: Design

Simulate or Stumble

Roadblock: TCP & Gripper Reality vs CAD

Key Challenges:

- Custom fingers ≠ simple offset
- Tilted fingers introduce orientation-dependent error
- Teaching precise grasp poses manually is exhausting and unreliable

How to Pass It:

Aggressive use of CAD & kinematics simulation

Fig: Sample changer @ MySpot beamline

ROCK-IT

ROADBLOCK 2

Teaching Poses in Constrained Spaces

Precision in robotic manipulation isn't just about the robot's capabilities; it's heavily influenced by the environment and the methods used for teaching. Our second major hurdle emerged when attempting to program precise poses within the physically restrictive beamline environment.

The "Feel" of Free-Drive

Relying on constant human force during free-drive mode is fatiguing and inherently imprecise, making it difficult to maintain stability for accurate pose teaching.

Human Precision Limits

Even highly skilled operators struggle with the sub-millimeter accuracy required. Small angular errors at the robot's base translate into significant deviations at the gripper's fingertip.

Amplified Angular Errors

Due to the robot's kinematics, a tiny angular misalignment (e.g., 0.1 degree) can cause the gripper to miss its target by several millimeters or even centimeters over the length of the arm.

Solution Strategies for Robust Pose Teaching

01

Teach Approach & Retreat

02

Separate Precision Needs

03

Utilize Mechanical Truth

The Lesson: If a pose is consistently hard to teach, it's often a design problem in the setup or tooling, not a user problem. Rethink the interface between robot and environment.

ROCK-IT

ROADBLOCK 3

Software–Hardware Contract Violations

Robotics systems fail at interfaces, not in components.

Stale Poses
Poses recorded once were reused after hardware components were subtly changed, leading to misalignments and collision risks.

QR Data Assumptions
Assumptions that QR code data would always arrive perfectly formed and in time led to errors when environmental factors interfered.

Small Gripper Tweaks Break Workflows
Even minor changes in gripper finger geometry or material can alter grasp mechanics, requiring entire pick-and-place routines to be re-taught and re-verified, disrupting established automated workflows.

The Lesson: Robust system design requires explicit interface contracts and thorough validation at every integration point. Anticipate and handle discrepancies between the expected and actual states of hardware and software.

Integration Is Not Linear

The V-model often depicts a smooth, sequential flow from requirements to validation. However, in complex robotic systems, integration is rarely a linear process. Changes in one phase can cascade, invalidating work done in others, forcing continuous adaptation.

The Ripple Effect of Change

Mechanical Changes Invalidate Software

A slight modification to a physical component, like a sample holder, can render existing software trajectories and CAD models obsolete, demanding costly re-programming and re-validation.

Calibration Affects Motion Planning

Recalibrating a robot's TCP or adjusting sensor offsets can subtly shift its kinematic model, leading to motion planning errors and potential collisions if not carefully re-integrated.

The Lesson: Integration must be a continuous, ongoing process, not a final phase. Embrace iterative development and build in mechanisms for rapid adaptation to changes across all system layers.

Outlook & Future Improvements

Parametric Positioning

Transition from manual "teaching" of robot poses to defining them parametrically in code or through precisely engineered physical fixtures. This drastically reduces human error and boosts repeatability.

Enhanced Calibration

Develop automated and adaptive calibration workflows for TCPs, robot base frames, and sensor offsets. Implement continuous monitoring to detect and correct any deviations in real-time, maintaining high precision.

Increased Autonomy & Resilience

Integrate advanced decision-making capabilities and sophisticated fault detection/recovery mechanisms. This allows the system to self-correct and handle unexpected events, minimizing downtime and human intervention.

Acknowldgements

rock-it-project.de

The ROCK-IT team at the 4th ROCK-IT project meeting at HZDR, October 2024.


ROCK-IT
Q&A
Session
DESY.


Thank You for
Attention