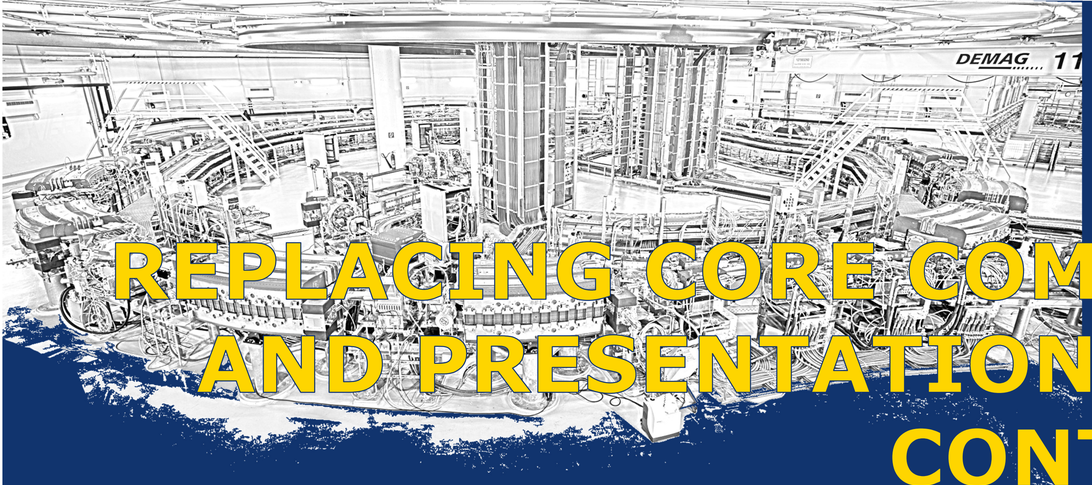

REPLACING CORE COMPONENTS OF THE PROCESSING AND PRESENTATION TIERS OF THE MEDAUSTRON CONTROL SYSTEM

A. Höller, **S. Vörös**, A. Kerschbaum-Gruber, C. Maderböck, D. Gostinski, L. Adler, M. Eichinger, M. Plöchl
EBG MedAustron GmbH, Marie Curie-Straße 5, 2700 Wiener Neustadt, Austria

Introduction

MedAustron is a synchrotron-based ion therapy and research facility in Austria, that has been successfully treating cancer patients since 2016. MedAustron acts as a manufacturer of its own accelerator. The presented project focuses on replacing the well-established WinCC OA SCADA system, enforcing separation of concerns mainly using .NET and web technologies, along with many upgrades of features and concepts where stakeholders had identified opportunities for improvement during our years of experience with the former control system setup for commissioning, operation and maintenance.

Commissioning Worker (COW)

The Commissioning Worker is an application that interprets Python and is smoothly integrated into the control system architecture and its user interface. COW provides the possibility to write Procedures with the capability of directly controlling and monitoring the accelerator, in combination with allowing the integration of additional Python libraries and applications (e.g., for measurement data analysis or calculation of machine parameters). It provides the possibility to either directly write and execute Procedures via the UI or integrate existing Procedures and Frameworks from a Git repository.

Table 1: Definition of COW Terms

Term	Definition
Framework	A Framework is a version-controlled collection of Python and support files. An example of a MedAustron Framework is PACMAN (Python Algorithms Coded for Measurement data ANalysis)[1].
Procedure	A Procedure is a script that can be executed in the COW. Procedures make use of the functionality provided by Frameworks or other Procedures.
Task	A Task can be executed/scheduled/triggered in the COW. It is the combination of a Procedure with input parameters and settings.
Task execution	A Task Execution is an instance of a Task.
Result	A Result is an artifact produced as an output of a Task Execution.

Figure 1: The MedAustron accelerator.

A common beam commissioning task is the alignment of the beam to pass through the optical centre of multiple quadrupoles for several extraction energies. As the absolute position of the in-line beam profile monitors (BPMs) cannot be assumed to be accurate, the offsets of the BPMs need to be determined as a prerequisite to iteratively optimize magnet setpoints for each energy. A single Procedure in COW would allow to autonomously perform the complete steering for all required energies, as depicted in Fig. 2.

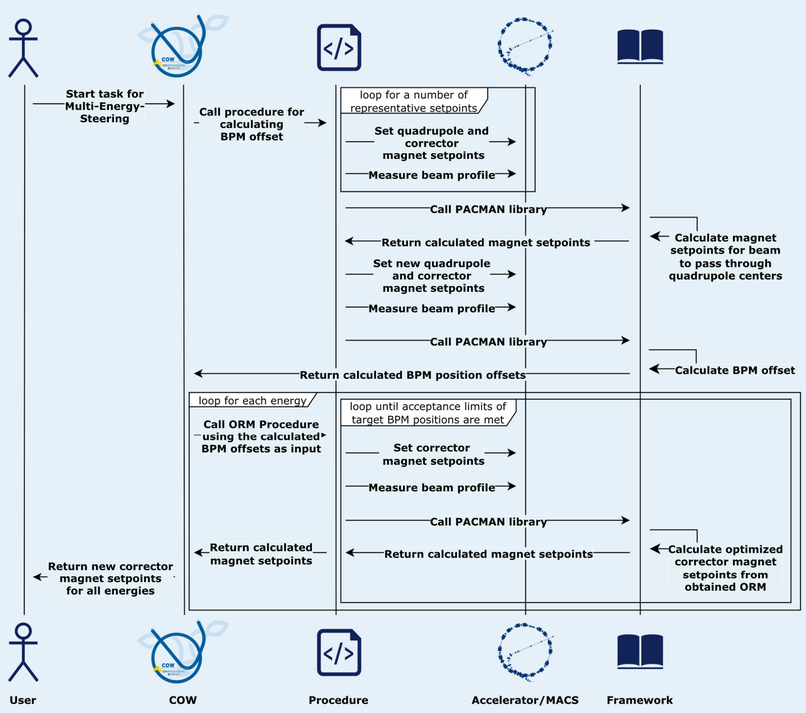

Figure 2: Sample Procedure „Multi-Energy-Steering".

Architectural Changes

Since the beginning of accelerator commissioning, the MedAustron Accelerator Control System (MACS)[2] has been exposed to countless feature requests, adaptations, extensions and environmental changes. Although the control system covers all current use cases, new needs have emerged, requiring the redesign of upper-tier core components.
The architecture needed to be redesigned to

- Support remote commissioning
- Address cyber security concerns
- Enable responsive design for PC as well as mobile device usage
- Enhance reliability and traceability with archiving
- Introduce a Python API for automated commissioning

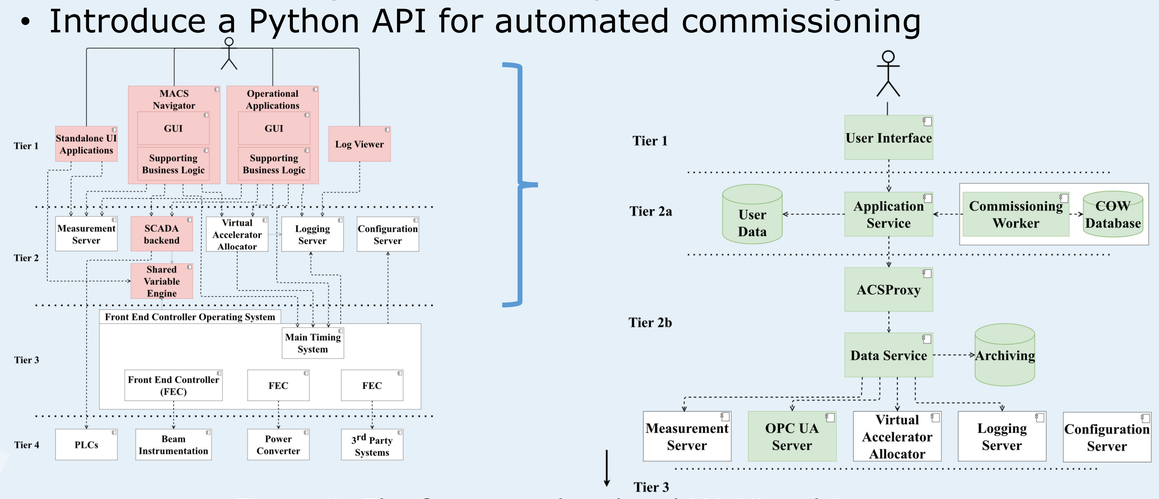

Figure 3: The former and updated MACS architecture.

User Interface Changes

The changes on interface level have been introduced in close collaboration with the Operations team to enable a smooth transition between the two systems and implement missing functionalities. The client is either accessible via a standard web browser, or an installed Electron client.

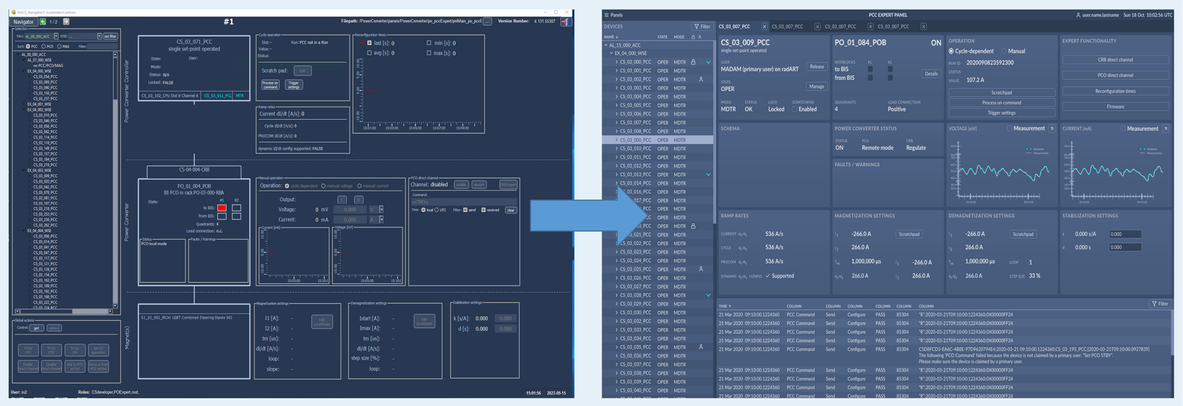

Figure 4: The former and updated MACS User Interface.

Technological Changes

Table 2: A Summarized View of the Technological Changes

Component	Former Technology	Replacement	Motivation for Replacement
User Interface	WinCC OA CTRL scripts	in-house developed Angular TypeScript	- Upgrade to state-of-the-art technology - Improve abstraction and encapsulation
SCADA business logic	WinCC OA CTRL scripts	in-house developed .NET C#	- Streamline client interfaces
Security gateway	Solved on IT level only	in-house developed .NET C#	- Advanced access control - Read only accessibility during operation
Lower tier device connectivity	WinCC OA OPC-DA & NI-PSP	Atvise OPC UA	- Reduce interface complexity - Increase security - Upgrade to state-of-the-art technology
Archiving of variables and measurements	not available	TimeScaleDB	- Archiving of all measurements, log messages, variable changes
Automated commissioning procedures	in-house developed .NET C# with UI	in-house developed .NET C# with Python API	- Provide an API to commissioners with Python knowledge

Conclusion

The updated control system (including COW) is designated to be used at the facility in Austria and other future facilities.
The expectancy towards the changes are easier maintainability of the control system (by reducing complexity and upgrading to state-of-the-art technology), increased usability (by incorporating user feedback throughout the project), better traceability (by directly integrating mechanisms for archiving measurement and log data into the architecture), increased robustness regarding cyber security and a great reduction in machine time required especially for commissioning but also for other maintenance and operations tasks. Additionally, these changes provide more flexibility for advanced users who prefer scripting Procedures to a classical user interface.

[1] A. Wastl et al., WEPOR045, IPAC'16
[2] J. Gutleber et al., MOBAUST03, ICALEPCS'11

Author Contacts:
angelika.hoeller@medaustron.at
+43 2622 26 100 254