17th International Conference on Space Operations, Dubai, United Arab Emirates, 6 - 10 March 2023.

SpaceOps-2023, ID # 267

PyCon2 – Managing virtualized control rooms safely and efficiently

Ilja Verspohl[a]*, Nico Trebbin[a],

[a] *LSE Space GmbH, Friedrichshafener Straße 2, Gilching, Germany 82205, ilja.verspohl@lsespace.com*

Abstract

Monitoring and controlling the ISS Columbus module requires various operational resources and software tools. The main software component is the Monitoring and Control System (MCS) running on 48 clients. In addition to that, another 88 clients which display live telemetry or allow direct payload commanding are part of the ground segment infrastructure. Traditionally, each client runs on a physical computer placed in a dedicated position in one of the nine control and service rooms. With the migration from hardwired to fully virtualized control rooms at the Columbus Control Center (Col-CC) in 2015, every resource runs as a virtual machine on one of the two data centers. They can be accessed from any physical machine (thin client) within the operations network allowing flexible console positions. Having more virtual than physical clients, a connection broker was required to link and assign a virtual client to a thin client.

PyCon (Python Connect) was developed to easily establish a session to a virtual client depending on the user's privileges and the thin client's location. The ground control team can assign one or multiple virtual clients to a physical machine. The flight control team can then access the virtual machines thus having a fully functional console displayed. This takes the workload from the flight control team by providing only the resources that are required. This system has proven to be highly reliable and especially during the pandemic it was a strong asset allowing contact-free shift handovers. With the development of MCS-R, the successor of MCS, and an overall renewal of the current infrastructure at the Col-CC, PyCon2 was developed to match the new requirements. It features the core functionalities of PyCon with major enhancements. With future missions in the pipeline, PyCon2 is now highly configurable, so additional control rooms, new virtual and physical clients, and different screen setups can easily be implemented by simply changing the setup in the database. During the process of development, special attention was given to reducing software dependencies and enabling a minimum effort of maintenance. This flexibility makes PyCon2 feasible to manage virtualized control rooms way beyond space operations. It enables the virtualization of complex control and monitoring systems and makes space operations more efficient and fail-safe.

Keywords: Ground Operations, Columbus Control Center, virtualization

Acronyms/Abbreviations

AOS ..... Acquisition of Signal
ASP ..... Activity Specialist
BUGOR ..... Back-Up Ground Operations Control Room
Col-CC ..... Columbus Control Center
Col-Flight ..... Columbus Flight Director
COMET ..... Columbus Operations & Mission Execution Timeline Engineer
DRDB ..... Distributed Replicated Block Device
EGS-CC ..... European Ground Systems Common Core
ESA ..... European Space Agency
EUROCOM ..... Crew Communicator
FCT ..... Flight Control Team
GC ..... Ground Controller
GCT ..... Ground Control Team
GOCR ..... Ground Operations Control Room
GSOC ..... German Space Operations Center
GUI ..... Graphical User Interface
HTTPS ..... Hypertext Transfer Protocol Secure
ISS ..... Internation Space Station
KU-IPS ..... KU-Band Internet Protocol Service
KVM ..... Keyboard Video Mouse
LDAP ..... Lightweight Directory Access Protocol
LOS ..... Loss of Signal
LSR ..... Life Support Rack
MCC-H ..... Mission Control Center Houston
MCS ..... Monitoring and Control System
MCS-R ..... Monitoring and Control System - Replacement
MDT ..... MOD Display Tools
MPCC ..... Multi Purpose Control Computer
NFS ..... Network File System
OPS LAN ..... Operations Local Area Network
OS ..... Operating System
PyCon ..... Python Connect
RDP ..... Remote Desktop Protocol
SSH ..... Secure Shell
SysCon ..... System Controller
UID ..... Unique Identifier
USOC ..... User Operations Control Center

1. Introduction

As a human spaceflight-rated spacecraft the Columbus module as part of the International Space Station (ISS) requires 24/7 monitoring and control coverage from ground to ensure a safe environment for the astronauts aboard, as well as to support experimental activities. At the Columbus Control Center (Col-CC) in Oberpfaffenhofen, three console positions are continuously manned: two positions from the Flight Control Team (FCT) and one position from the Ground Control Team (GCT): The Columbus Flight Director (Col-Flight) is the flight director for the Columbus module and coordinates the team in correspondence with the ISS flight director at Mission Control Center in Houston (MCC-H). STRATOS is responsible for the Columbus on-board systems and payloads. The Ground Controller (GC) is responsible for the overall ground segment and all data links between Col-CC, MCC-H, and the User Operation Control Centers (USOCs), responsible for payload operations across Europe. Additional console positions with less coverage are System Controllers (SysCon), as part of the GCT to configure and monitor the ground system, Columbus Operations & Mission Execution Timeline Engineers (COMET/EPIC) for mission planning activities and specialist support for on-board experiments, Crew Communicators (EUROCOM) for

communication with the European Space Agency (ESA) astronauts, and Activity Specialists (ASP) to support STRATOS during special on-board activities [1]. All those positions have their dedicated console in one of the control rooms. Each console has a different screen setup, as well as different software applications that need to be accessed by the respective operator. Figure 1 shows the control room K4 with the default console positions.

Figure 1: control rom K4 at Col-CC

Figure 2 shows the schematics of the consoles that are within the operations network (OPS LAN) and are used to monitor telemetry and send telecommands. It should be mentioned that only the consoles within the OPS LAN are considered and analyzed in this paper, since it's the separated network at Col-CC in which telemetry and telecommands are sent between servers and clients. Furthermore, there are computers available in the control rooms, that have access to other network areas for additional mission-relevant information and tools for mission planning.

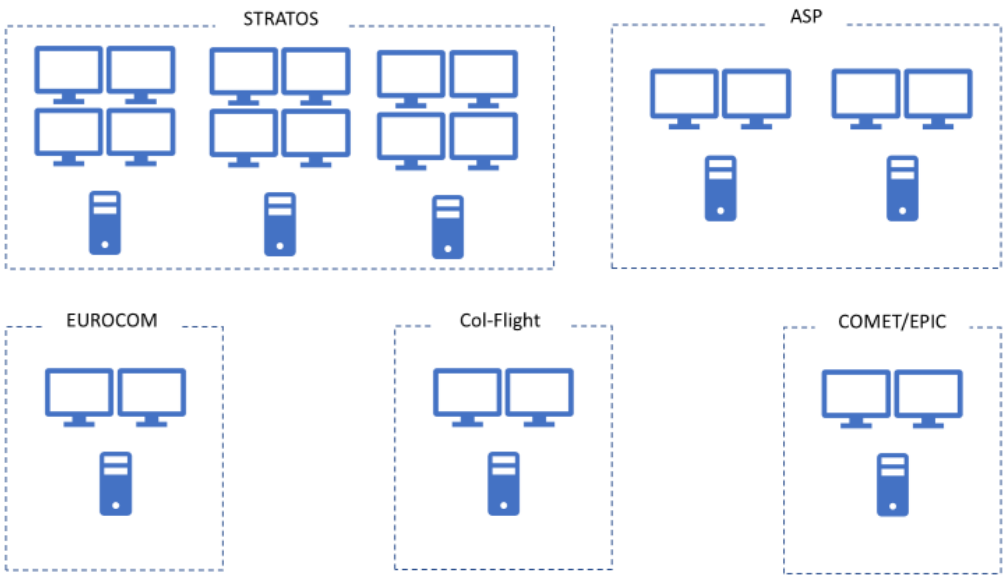

Figure 2: available consoles within the OPS LAN used for monitoring and commanding the Columbus module

1.1 Software Applications

Usually, each of the software applications runs on a dedicated machine which is located within the OPS LAN. The most important software system is the Monitoring and Control System (MCS), which enables the operator to monitor the Columbus systems and send commands to Columbus itself and to its payloads. It consists of a server-client architecture with several servers for dedicated tasks such as commanding or telemetry routing and clients that connect to the servers and process the telemetry. Those clients are operated by the flight control team on console. Satmon is an application that receives already processed telemetry data from the MCS and is used to display that telemetry in a highly configurable and convenient way. It is mainly used by STRATOS and Col-Flight. MDT provides and displays additional mission-relevant data to the operator e.g. the Acquisition/Loss Of Signal (AOS/LOS) status. Recently, new applications have been introduced to the Col-CC operations infrastructure: Dedicated clients allow direct monitoring and control of the onboard Multi Purpose Control Computer (MPCC) and the Life Support Rack (LSR) via the NASA KU-Band Internet Protocol Service (KU-IPS) by the flight controller [2]. Those clients have all different hardware requirements and run on different operating systems (OS). Table 1 gives an overview of the systems.

Table 1: Overview of the used software applications within the operations network

	Function	Operating System	Amount of clients	Available mission modes
MCS	Main Monitoring & Control System	SLES 10 SP2 / 12 SP5	48	OPS, SIM, TEST
Satmon	Displaying Telemetry	Windows Server 2019	6	OPS, SIM, TEST
MDT	Displaying Telemetry	CentOS 6.8	6	OPS
MPCC	Monitoring and Control System dedicated to MPCC	Windows 10	18	OPS, SIM
LSR	Monitoring and Control System dedicated to LSR	Windows 10	2	OPS

In addition to the existing clients, a next-generation MCS, using the emerging European Ground Systems Common Core (EGS-CC) as a framework, is already in development under the project name Monitoring and Control System – Replacement (MCS-R) and will be added as a dedicated software application [3].

1.2 Control Room Configuration

Several control rooms are available at Col-CC for dedicated Columbus operations, simulations and testing activities. K3 and K4 are the main control rooms within Col-CC. One being the active control room and one being an active standby control room as a backup. Additionally, K11 is a fully equipped control room located in another building and acting as a geo-redundant backup, but is also used for simulations and testing activities. During nominal operations Col-Flight, STRATOS, and EPIC/COMET have their consoles set up in K4. The Ground Operations Control Room (GOCR) and the Back-Up Ground Operations Control Room (BUGOR) are smaller control rooms dedicated to the Ground Controllers and System Controllers with each room being located in another building for geo-redundancy. MCS has been deployed in three instances, each one consisting of 16 clients and 10 servers. Each instance can be configured to one of three different mission modes: OPS, SIM, and TEST. Traditionally, OPS mode is configured in K4, TEST mode in K11, and SIM mode in K3. This allows parallel testing, maintenance, and simulations while the nominal operations are not affected. Also, the other software systems are available in different mission modes.

To access a console, the operator must unlock the console with a personalized smart card. An operator is usually assigned one of several functional roles on the smart card, which gives him access to specific consoles in a configuration that depends on the functional role.

1.3 Virtualization at Col-CC

To simplify the infrastructure of the ground segment and to ensure the availability of all software applications, the decision was made in 2014 to virtualize the operating infrastructure [4]. All of the mentioned software

applications, usually running on a dedicated computer were transferred onto virtual machines running on one of the two data centers at Col-CC running VMWare® ESXi as hypervisor. The workstations in the control rooms were replaced by thin clients, which are now only used for smart card authentication and establishing a remote session to the desired virtual machine.

Virtualization drastically reduces the number of dedicated workstations, by centralizing the computational resources into the data centers and leaving only a few thin clients from which the software applications can be accessed. This reduced the number of physical machines to one computer per console, allowing better hardware maintenance and more efficient energy consumption and storage usage [5, 6].

In addition, virtualization allows multiple applications that require different operating systems or different versions of the same operating system, to be used on one platform. Environments for old operating systems for which hardware can no longer be procured can be preserved. The key advantage is however, that all software applications are now available from any console within the OPS LAN. This allows a dynamic usage of the software systems and Keyboard Video Mouse (KVM) switches are not required anymore.

2. PyCon

To simplify the login process and the establishment of the remote sessions from a thin client to a virtual client, Python Connect (PyCon) was introduced as a connection broker [4]. PyCon Login is installed on the thin clients and can automatically establish a remote session to a desired virtual machine. When a user logs on to a thin client using a smart card, PyCon retrieves the user's functional role that is mapped to the smart card. In addition to that, the location of the thin client and the allowed mission mode (OPS, SIM, TEST) are considered. PyCon takes all this information into account and returns a list of all virtual machines that are available to the user. The operator can then, if desired, select from the available virtual machines and launch individual remote sessions to connect to an MCS and/or Satmon client. Dependent on the console setup and the number of available screens, PyCon automatically places the remote sessions on to the correct screen. For example, a STRATOS console with four screens is configured to have the upper two screens used for Satmon and the lower two screens used for MCS. Since a combination of Windows® and Linux™ platforms are in use at Col-CC, two different protocols need to be supported. The Remote Desktop Protocol (RDP) is used to establish a remote session with Windows® clients and the NoMachine® NX protocol is used for the Linux™ remote sessions.

In addition to the user facing PyCon Login, PyCon also consists of an administration tool. This administration tool allows dynamic assignment of virtual machines to thin clients. Ground Controller and System Controller can select via drag and drop, which virtual machine can be accessed from which thin client and by which user group. Additionally, the allowed mission mode can be set for each thin client or directly for all thin clients within one control room.

All the information is stored in a database which has been deployed in a redundant manner. A MySQL™ database comprises tables for thin clients (physical machines) and virtual clients of all types. The tables of the thin clients and virtual clients have specific information such as the domain name and IP address.

A unique identifier (UID) is defined for each thin client. This thin client-specific UID defines which virtual client and therefore which software application is available to which console. When assigning a virtual client to a thin client, the UID of the thin client is linked to the UID of the virtual client. Since the software applications are all multi-user systems, multiple UIDs can be assigned to a single virtual client, however, the number of active connections to a virtual client can be limited. In addition to that, it is possible that multiple thin clients have the same UID. This allows having the same configuration for multiple consoles. This is used to have the same operator setup for a certain console in multiple control rooms and allowing a relocation of an operator into another control room without any reconfiguration. Therefore, PyCon gives the operator the same working environment regardless of the control room and synchronization of the prime and backup control rooms is ensured.

3. PyCon 2

After more than 8 years in operation, several factors led to the decision to fully redevelop PyCon in order to fulfill new requirements and prepare the ground infrastructure for operations until the planned end-of-life of the ISS program in 2030.

3.1 Requirements

In general, the following requirements have been defined for the designing process of PyCon 2:

- *All client-specific data shall be stored in a database.*
  To easily add and remove physical clients as well as virtual clients, all the data shall be stored in the database. This is intended, so no patches for PyCon are required if new applications are introduced to Col-CC.

- *All room-specific data shall be stored in a database.*
  As the thin and virtual clients, the room setup may change in the future. New rooms may be built up as a control or support room. Since no rooms are hard-coded in PyCon any longer, the room configuration should behave dynamically as defined in the database.

- *Thin client configuration shall be decoupled from the underlying hardware.*
  With new applications in the testing phase, the FCT and GCT may prefer a different screen setup. With the ability to change the screen settings for a console, a new setting should be easily implemented into the screen configuration in PyCon. Therefore, the PyCon administration tool shall give the option to reconfigure the screen setup on demand.

- *The PyCon administration tool shall be implemented as a server application with a web Graphical User Interface (GUI).*
  With this setup, access to the administration tool shall be more convenient. Moving the administration tool to the back-end ensures having only one administration tool running and sending queries to the database. In addition to that, a Lightweight Directory Access Protocol (LDAP) authentication can be implemented to define who is allowed to use the administration tool.

- *All network communication shall be encrypted.*
  With connections between the front-end and back-end including the exchange of sensitive login data, all network connections shall be encrypted.

- *Software dependencies shall be as few as possible.*
  To have a long-lasting application, which requires a minimum amount of maintenance, the number of dependencies on external libraries shall be kept as low as possible.

- *The operator shall not be bothered with selecting the right applications*.
  Although PyCon 2 shall be highly configurable. All the configuring shall be performed by GCT in compliance with the FCT. This shall ensure, that an operator will always have the same screen and application setup.

- *The installation and roll-out process shall not affect ongoing operations.*
  To ensure that no ongoing operations are impacted, special attention needs to be given during the roll-out process.

3.2 PyCon 2 Software Architecture

To fulfill the requirements, defined in the previous section, PyCon 2 consists of a front-end and a back-end. The front-end gives the operator the graphical interfaces to edit the PyCon database using a browser application (Web GUI) and to launch a remote session to a virtual machine (PyCon Login). The back-end consists of the database (PyCon DB) and a web application (PyCon Config) using the Flask™ framework. Figure 3 gives an overview of the different software components on the back-end and front-end which build up PyCon 2. Additionally, the communication protocols are shown.

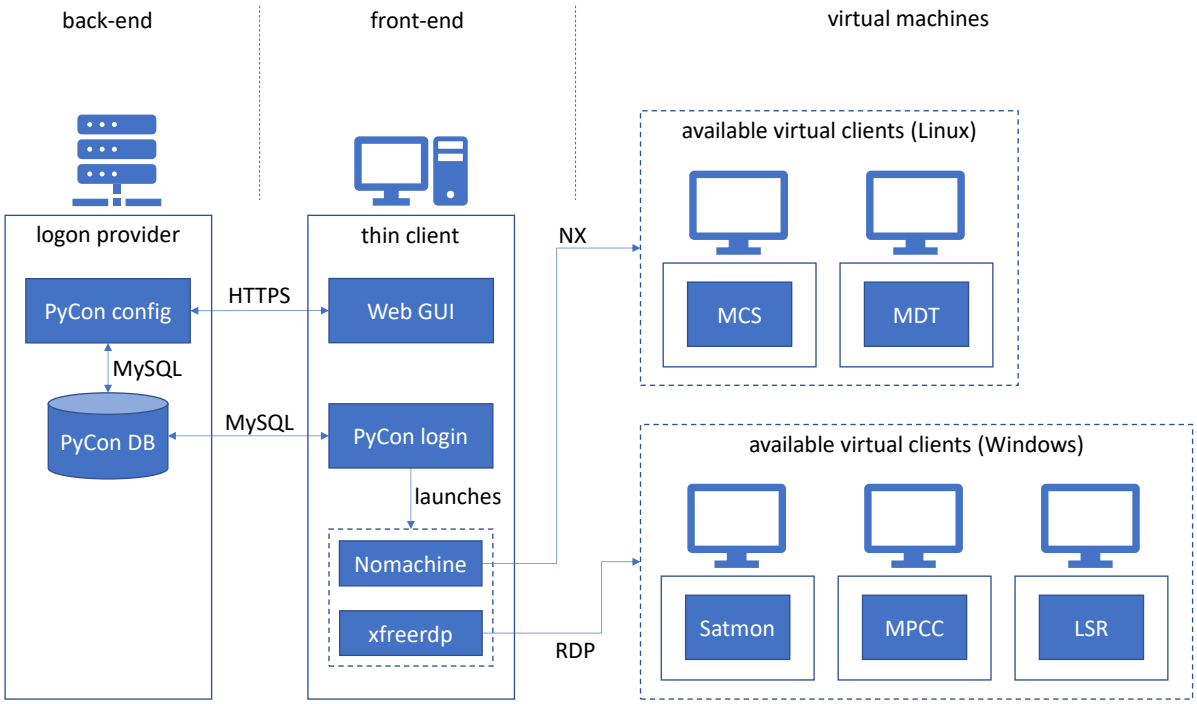

Figure 3: Overview of the different software components

3.3. PyCon 2 Front-End

Since the hardware of the current thin clients has reached the end-of-life, a replacement and migration to new hardware will take place soon. To fit modern OS standards, the new hardware will come with a new operating system OpenLeap™ 15 SP3. Awesome™ WM 4 is used as a window manager because it is highly configurable and allows the implementation of custom-made widgets. In addition to that, Awesome has shown good performance and reliability on the old thin clients. With that platform as a front-end, PyCon Login runs with Python 3, and its GUI runs with PyQt 5. Many functions have been transferred from PyCon 1 with minor changes.

With PyCon 1 having MPCC and LSR added in the aftermath, those virtual machines were not included as standalone virtual clients. PyCon 2 treats all virtual clients, which are stored in the database, the same. With this feature, virtual clients and complete instances of a software system consisting of multiple clients, can be easily added to the database and directly accessed via the front-end. In addition to that, clients of a completely new type can be added as needed and with little effort.

In PyCon 1, all consoles had a fixed layout. Having a four-screen STRATOS console, PyCon 1 always launched Satmon onto the upper two screens and MCS onto the lower two screens. After adding MPCC to PyCon 1 (as an additional item in the top menu), MPCC had always been launched onto the lower screen in a new tab. Therefore, the operator had to choose whether MCS or MPCC shall be displayed. Having the screens configurable, completely new layouts and arrangements of the various virtual clients can be achieved. Figure 4 shows some possible configurations in comparison to the old, fixed layout in PyCon 1.

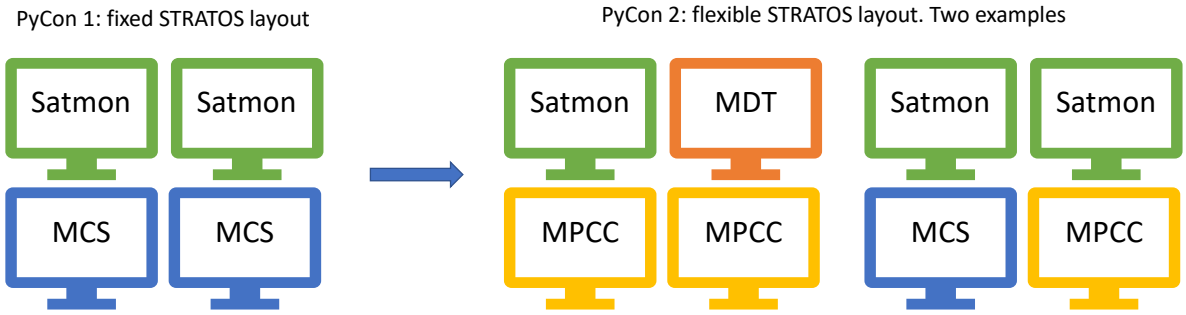

Figure 4: Possible screen layouts for a four-screen STRATOS console

Since GCT is in charge of assigning the virtual clients to the thin clients, it will also be in charge of setting up the screen configuration. Additionally, it can be defined which clients will be launched in the background, allowing multiple virtual clients to operate on one screen. A virtual client can then be brought to the foreground by selecting the correct tab.

As PyCon 1, the NX and RDP protocols are used to establish the remote session with the virtual clients. PyCon 2 is configured to limit the possible connections to a virtual client to allow only one active connection at a time. This ensures that the resources of the virtual machines are not overused and no additional processes are disturbed by unknown or leftover connections.

3.3 PyCon 2 Back End

The back-end consists of two servers running with SLES™ 15 SP4. As virtual machines they are located within the OPS LAN and can be reached from any thin client. Those two servers form a cluster sharing a virtual cluster IP address and are synchronized via Distributed Replicated Block Device (DRDB).

A MySQL™ 8 database is running on the cluster allowing mirroring and consistency and therefore high availability. The database consists of three databases:

- *pycondb*
  This database holds all the client specific data for each thin client and virtual client in a table for each client type. It holds the information, how to reach the client in the network. For thin clients, the room where it is located, the UID, the available mission modes, and the default screen configuration is stored. For the virtual clients access information, the current UID, and the mission mode of that client is stored.

- *pyconsshkeys*
  This database stores the Secure Shell (SSH) keys for the virtual clients. The SSH keys are required to authenticate on a virtual client using the NX protocol. This table has restricted reading and writing rights.

- *pyconlog*
  This database acts as an audit log and records all sessions set up and who has accessed the thin client. With this log, it is always possible to trace who has used which console, as the virtual clients are only accessed with functional, but not with personal accounts. This database also has limited read and write permissions.

The PyCon configuration tool allows the user (usually GCT) to view and edit the database. It is built as a web application that receives and processes configuration parameters from a browser application. The browser application is written in JavaScript™ and transfers data via Hypertext Transfer Protocol Secure (HTTPS) GET and POST requests. The accessibility of the browser application is limited to GCs and SysCons. Therefore, a user logs in using the personal LDAP credentials. This allows the PyCon configuration to be managed from any console within the OPS LAN, provided the user is authorized to do so. The browser application provides a graphical user interface where a configuration overview is displayed and virtual clients can be assigned to a thin client. In addition to that, the mission mode of the MCS instance can be monitored and set as well as the allowed mission modes of single

consoles or entire rooms. With this function, consoles and rooms can be restricted from a set of virtual clients, ensuring no interference with real-time operations when running a simulation or testing in a different control room. To allow convenient troubleshooting, the generated PyCon log files of each thin client, as well as the server access logs can be viewed and filtered within the web GUI.

4. Roll-out and Outlook

The introduction of PyCon 2 should be done in such a way that it disrupts ongoing operations as little as possible. As for PyCon 1, the user accessible files will be located on a shared Network File System (NFS) which is accessible from all thin clients. However, PyCon 2 is launched by running a script on the local file system of the thin client, which in fact is a symlink to the NFS. This approach ensures, that all thin clients run with the same version of PyCon. Since the roll-out of PyCon 2 takes place together with the hardware migration to new thin clients at Col-CC, the symlink needs to be set once at every physical machine. This allows a step-by-step migration of the hardware and PyCon 2. In the first step of the migration, a STRATOS setup consisting of three four-screen consoles and other computers that are not in the OPS LAN, will be setup in a support room. This gives the FCT the chance to test the console setup and get familiar with the new hardware and new operating system. If the FCT approves the setup, migration can continue with the remaining control rooms. Having the setup in the control room K11, used for simulations, the setup can also be validated in one of the regular simulations performed in the Col-CC. During this process, all the consoles within the OPS LAN, that still run with old thin clients will not be affected at all.

5. Conclusion

PyCon 2 ensures safe and efficient operations of the Columbus module for the next upcoming years. Together with the ongoing hardware migration, Col-CC is prepared for operations until the planned end-of-life of ISS in 2030. With new software tools in the pipeline and major modernizations on board of Columbus, PyCon 2 enables the easy integration of upcoming monitoring and control systems without interfering with ongoing operations. As the virtualization of complex computer systems and networks is taking place in many control centers and not only within the space domain, a connection broker like PyCon 2 is suitable to access the required virtualized resources in a secure and convenient way. In addition, the configuration can be easily monitored and access can be tracked. Thanks to the great work in the development of PyCon 1 the general principles and a lot of knowledge could be adopted. The successful operation of PyCon 1 over many years has once again proven the potential of virtualisation, but also highlighted the challenges ahead. With PyCon 2 as an adequate successor to PyCon 1, the success story of virtualisation at the German Space Operations Center (GSOC) and for the Columbus project should continue for many exciting years to come.

References

[1] D. Sabath, G. Söllner, T. Kuch, T. Müller. The Future of Columbus Operations. SpaceOps 2014 Conference, Pasadena, USA, 2014, 5 – 9 May.
[2] T. Müller, D. Burdulis, C. Corsten, MPCC and Ku-IPS. New Ways to Control the Next Generation of Columbus Payloads - Ground Segment Aspects. 68th International Aeronautical Congress, Adelaide, Australia, 2017, 25 – 29 September.
[3] N. Trebbin, M. P. Geyer, A.-K. Schroeder-Lanz, C. Stangl. Never change a running system? A renewal of the Columbus Monitoring and Control System. SpaceOps 2018 Conference, Marceille, France, 2018, 28 May – 1 June.
[4] N. Trebbin, Virtualization of the Columbus Control Room Infrastructure. SpaceOps 2016 Conference, Daejeon, Korea, 2016, 16 – 20 May.
[5] M. Schmidhuber, U. Kretschel, T. Singer, A. Uschold. Virtualizing Monitoring and Control Systems: First Operational Experience and Future Applications. Progress in Astronautics and Aeronautics, 236, AIAA, 2011, pp. 381–393.
[6] J. Prieto, J. Feiteirinha, P. Bizarro, E. Gomez, M. Pecchioli. Use of Virtualisation Techniques for Ground Data Systems. SpaceOps 2008 Conference, Heidelberg, Germany, 2008, 12 – 16 May.