The Columbus Desktop Trainer: An Alternative Solution to Operations Training

Jamie Denniston[1]
Deutsches Zentrum für Luft- und Raumfahrt e.V. / INSYEN, Oberpfaffenhofen, 82234 Wessling, Germany

Thomas Uhlig[2]
Deutsches Zentrum für Luft- und Raumfahrt (DLR) e.V., Oberpfaffenhofen82234 Wessling, , Germany

Simulators have long been established as one of, if not the, most important tools in the training of spacecraft operations staff. They have become vital in preparing new staff for work on console, and formal simulations are generally an essential step in becoming certified. On top of this, simulators are now being used for an increasingly large number of other tasks: new procedure validation, testing of new database deliveries, implementation and testing of derived telemetry parameters, testing of ground software updates, ie. new synoptic displays, and of course the simulations regularly carried out by certified staff both in preparation of onboard activities and to ensure that a high level of experience is preserved. This growing demand often results in constraints on the training process and scheduling difficulties for the use of the simulator. The traditional philosophy behind a simulator delivery, one or two instances accessible in a control room along with other ground software, is struggling to meet the needs of the various users. As well as scheduling the simulator itself, acquiring the necessary control room and the support staff can also be difficult, since checkpointing and configuration frequently require a specialist. Our group has envisaged and developed an alternative but complimentary solution known as the Desktop Trainer. Developed initially for the Columbus project, the Desktop Trainer uses virtualisation to combine all of the key software components used by operations staff on console with our new Columbus simulator, and condenses it into one powerful desktop computer. What makes our solution unusual is that it not only emulates the onboard systems of the spacecraft, but in fact provides a complete end-to-end simulation including the full ground segment. This implementation is portable, easily cloneable, and can be remotely accessed. By providing all trainees with their own Desktop Trainer it frees up considerable time on the existing simulator. No longer will they have to share the resources with numerous other activities, but instead can gain simulator access from the comfort of their own offices 24 hours a day. Consequently, the training process is both enriched and at the same time shortened, thus saving project money. This paper describes in much more detail the philosophy behind the Columbus Desktop Trainer, the system architecture and implementation, and the advantages it brings to the training process.

[1] Systems Engineer, German Space Operations Center (GSOC), Münchner Strasse 20 82234 Wessling, Jamie.Denniston@dlr.de
[2] Columbus Flight Director, German Space Operations Center (GSOC), Münchner Strasse 20 82234 Wessling, Thomas.Uhlig@dlr.de

Nomenclature

Col-CC = Columbus Control Centre
DaSS = Data Services Subsystem
DMS = Data Management System
FCT = Flight Control Team
GSOC = German Space Operations Centre
ISS = International Space Station
MCS = Monitoring and Control System
NTP = Network Time Protocol
TM = Telemetry
TQVS = Training Qualification and Validation Subsystem

I. Introduction

The Columbus Control Centre (Col-CC)[1], located inside the German Space Operations Center (GSOC) near Munich, is the main site responsible for the operation of the systems of the Columbus Module onboard the ISS. Furthermore the staff working at Col-CC has the responsibility of operating the European ground communications network and managing the operation of all European payloads aboard the ISS. Col-CC is staffed 24/7 by the Flight Control Team (FCT); a team of experts with various specialisations, who carry out all of the required monitoring, commanding, and communications.

Figure 1. Columbus Control Centre, Oberpfaffenhofen

The training and certification of the FCT members is undertaken by multiple different entities, all consisting of ESA certified instructors. The Col-CC Training Team is responsible for the preparation of the Columbus FCT for basic operations and consists of certified Flight Controllers with a combined expertise of all the necessary core disciplines. The scope of the Col-CC Training Team's responsibilities goes beyond simply training the staff to man the control room however, and their regular training courses are attended by a wide variety of people who are connected to the Columbus project in different ways. One of, if not the most important tool used by the training team in order to transfer and preserve knowledge during their training regimes is a simulator. Of course the value of simulators and simulation exercies is by no means unique to the Columbus project, but is in fact common to all modern space missions. The downside of this fact though is that this importance has resulted in an ever-growing demand on the use of the simulator, which is ultimately a limited resource.

In the case of the Columbus project, the use of simulators follows the conventional approach seen in most missions; a few instances of a simulator which can only be accessed from inside a control room, and always requiring the assistance of dedicated support staff to setup and configure the system. The sheer number of different people requiring access to the simulator combined with the multitude of different tasks that are performed, has resulted in a constraint on the training process in the form of scheduling. The main driver behind the philosophy of the Desktop Trainer is to avoid this constraint; by utilising the correct strategy this bottleneck can be removed from training. Consequently the Columbus Desktop Trainer has been designed and implemented with a view to, on one hand, enriching and streamlining the current operations training process, whilst on the other hand providing an extremely useful resource to qualified FCT members.

The key concept of the Desktop Trainer is to condense all of the necessary ground monitoring and commanding software into one machine, which when combined with our simulator will effectively produce a 'Col-CC console in a box'. Unlike the arrangements currently employed in the GSOC control rooms, where the various software components are spread over several computers representing servers and clients, the Desktop Trainer uses virtualisation to allow everything to run on one powerful desktop computer. In the case of Columbus the important elements which are included in the system are the Data Services Subsystem (DaSS), Monitoring and Control System (MCS), Satmon[2], and of course our Columbus simulator. The familiarisation with these software components, especially MCS and Satmon, form a fundamental stage in the early training of new FCT members. This paper will detail specifically the way in which the Columbus Desktop Trainer could positively influence the Columbus training process, but it will also make the more general case for a Desktop Trainer in all space missions.

II. Multi-layered Training Concept

The primary goal of the Desktop Trainer hasn't been to replace the existing simulator, Training Qualification and Validation Subsystem (TQVS), which is currently used for the majority of training in the Columbus Project. TQVS's use of exactly the same onboard software as is found onboard Columbus is a training advantage that cannot be stressed too lightly. Additionally, in the case of the Data Management System (DMS), a lot of the hardware used onboard Columbus is also included in the TQVS system. The downside however of this approach is that consequently TQVS can often be slow to start, restart, or load checkpoints. Furthermore the use of real hardware in TQVS results in it being much more difficult to readily clone the simulator to large scales if the demand requires it. Our philosophy has always been that a second more streamlined simulation environment could reduce some of the considerable load on TQVS and very effectively augment the training process. Using different simulators for different tasks and indeed different levels of training is not a new idea, but in fact can be seen across various disciplines. Pilot training is a useful illustration of this point, as can be seen in Figure 2 below. Actual flying time and even full motion simulators are known to be expensive and consequently limited resources, so therefore it doesn't make sense to perform all stages of the training within the one environment. For the early stages of training and familiarisation, and in fact for any task where the fidelity requirements are acceptable, a fixed place simulator can be used. Furthermore desktop flight simulators are becoming more and more common for teaching pilots the basics of topics such as flight controls and cockpit layout[3].

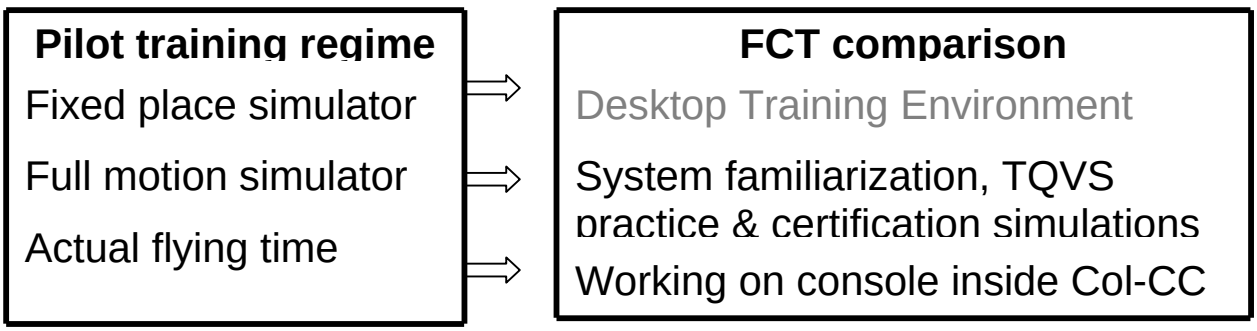

Figure 2. Comparison of pilot training with FCT training

If compared with the present FCT situation the difference is easily apparent; currently all FCT training, and additionally all routine FCT tasks which require a simulator are carried out on TQVS, even though TQVS is an expensive and very limited resource. It is therefore not surprising to discover that there are regularly problems in scheduling simulator time. Whilst tasks such as official simulation certifications would still need to be carried out on the official spacecraft manufacturer simulator, a large amount of lower priority tasks could be carried out on the Desktop Trainer and ultimately result in a much more efficiently managed training process.

Another advantage of using more than one simulator for a multi-layered training process was discovered during the development of our Columbus simulator. The problem with using one simulator is that there is effectively only access to one opinion on any scenario that hasn't been observed onboard the spacecraft and is therefore not available in real telemetry archives. No matter how accurate a simulator is, it will never be able to perfectly mimic the real situation, and therefore by introducing a second simulator it offers a second opinion on the more unusual spacecraft scenarios. If the two simulators differ in how they represent the behaviour of a parameter then this is in fact a great way of increasing the depth of knowledge in the operations team; it doesn't matter which simulator is correct because the debate and discussion which is initiated by this conflict will always cause the FCT member to delve much deeper into the problem than they normally would. This very situation occurred during the development of the simulator where the investigation into which answer was correct led so deep that the software architects of the spacecraft manufacturer were contacted to help clear it up.

III. System Details

As has been mentioned already, the key driver behind the design of the Columbus Desktop Trainer has been that everything a user requires to recreate a realistic Col-CC console position should be condensed and made available on one desktop machine. This allows our system to be portable and, if a laptop was chosen of a high enough specification, theoretically could permit the whole environment moving outside of GSOC. Of course the other huge requirement was to keep the final solution as low cost as possible, and as such allow it to be readily cloned to a scale where, if required, every user could have access to their own personal system. It has been important to ensure that the monitoring and commanding ground software that is included in the Desktop Trainer is not modified enough that it would appear any different to a trainee using it. Any deviations in our training environment from the real on-console environment would result in negative training, which is the complete antithesis of our philosophy. Correspondingly it has been crucial to provide the exact same versions of the software that is currently used on console, with all the same functionality and configurations.

A. Columbus Simulator

In addition to all of the monitoring and commanding software that is necessary in the Desktop Trainer, a simulator is also needed to complete the loop. In the case of the Columbus Desktop Trainer, our group has implemented our own Columbus Simulator which is built around an existing simulation framework known as SimGen. SimGen was developed by our group some years ago and has since been used as the core of various simulators, both inside and outside of the space domain. One of the key drivers in the architecture of the Columbus Simulator has been that of a "breadth-first" as opposed to "depth-first" approach. In practice what this means is that during the implementation of the simulator we start by creating all of the necessary models but initially with a very low level of detail; blank TM packets or arbitrary fixed parameter values. Whilst this level of fidelity would not be useful for any sort of in-depth simulations, it can be used as a source to feed early versions of telemetry systems and can test the end to end data flow in a mission's ground systems. From this position the simulation models are then progressively and iteratively refined and consequently the level of fidelity increases as the project life-cycle moves forward, until the accuracy is high enough to be used for training on the behaviour of the onboard systems. Our Columbus Simulator has been extensively tested and validated against the highest standard available; real-time and archived telemetry.

The Columbus Simulator consists of 2 GUIs: the Monitoring Window and the SimControl, snapshots of which can be viewed in Figure 3 and Figure 4 below. In a training environment these two GUIs would nominally be accessed by the trainer and not the trainees. The purpose of the Monitoring Window is mainly to allow the user to view in real-time the configuration and key outputs of the simulation. A user can view the number of processed parameters being simulated, the number of TM packets being sent and at what rates, the AOS / LOS state, output messages from the simulator, and various other connection and configuration statuses. The SimControl GUI however is the main interface for a user to interact with the simulation. A user is presented with the ability to load & save checkpoints, and to pause, stop, start, speed up, and slow down the simulator. Additionally the flight database is presented as a parameter tree whereby a user can locate any command or parameter that they wish to send or influence respectively. Simply right clicking a command provides the user the option to send it, whilst right clicking a parameter provides the options to either fix a static value, apply a ramp function to the value, or to view a real-time graph of the parameter behaviour.

Furthermore The SimControl GUI includes the ability to save a list of multiple interactions (sending of commands or altering parameter values) in a file which is very useful for creating and storing large complicated failure scenarios. The simplicity of directly altering the behaviour of parameters in this way has been very attractive for the Col-CC Training Team, because it allows them a very convenient tool to quickly alter a trainee's system in a way that is completely hidden from that trainee's monitoring and commanding displays. Whilst the cause is hidden, the effect is of course visible to the trainee and how quickly and effectively a trainee responds to this input is a crucial indication of what level of proficiency the trainee has reached. The fact that the failure inputs can be stored as files means that complex scenarios can be loaded into multiple trainees' systems at the same time and consequently they can be compared against each other.

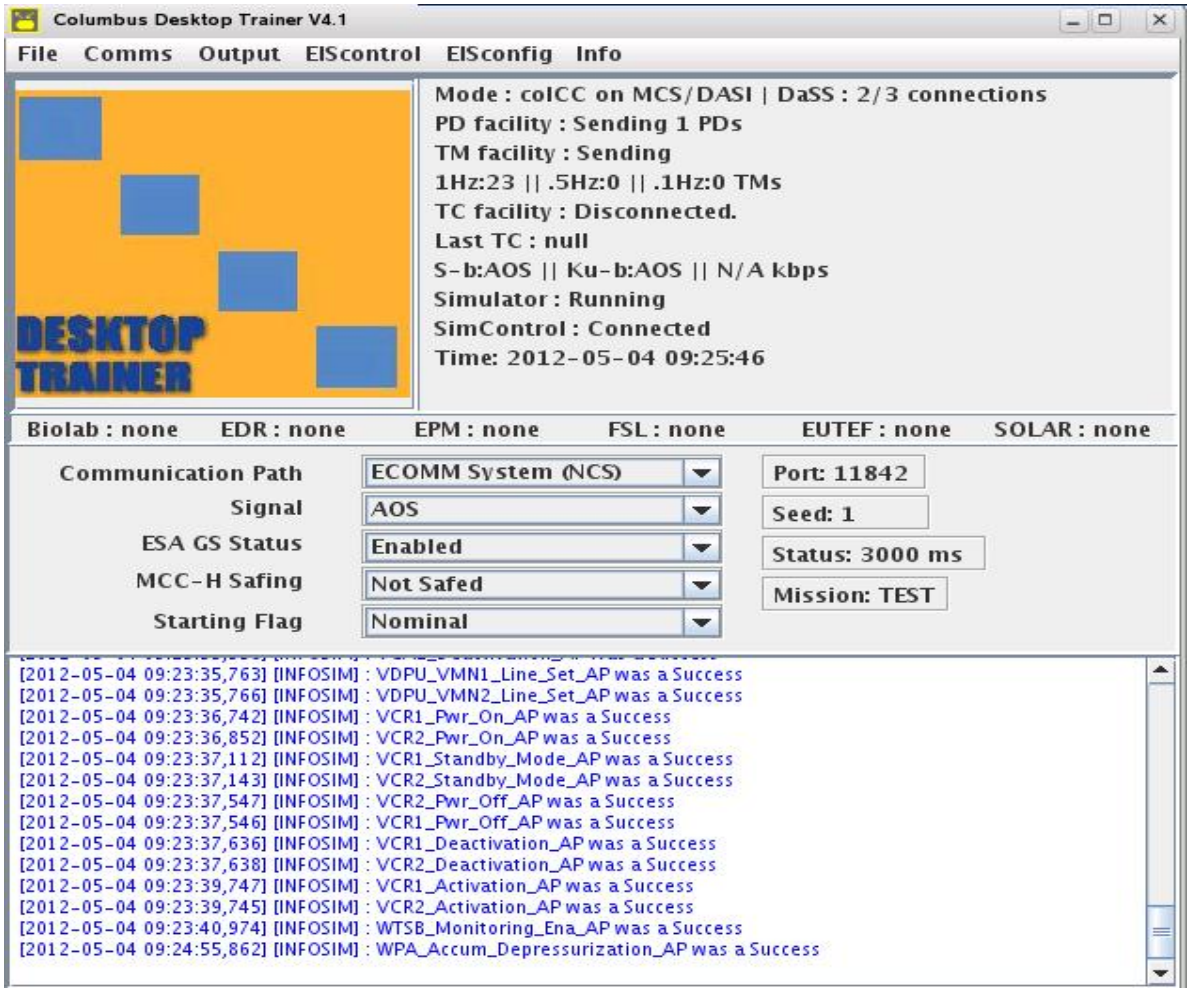

Figure 3. Monitoring Window GUI of our Columbus Simulator

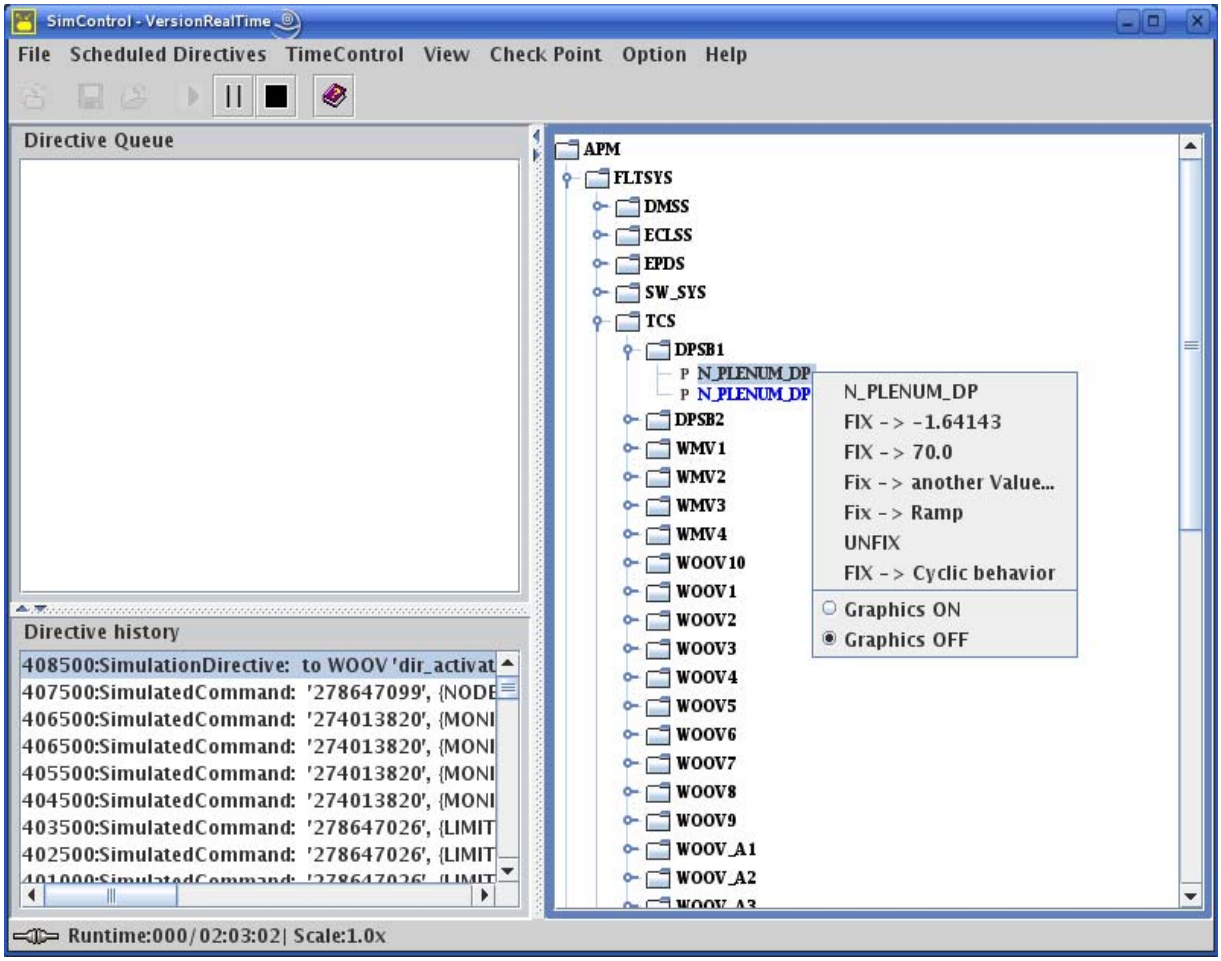

Figure 4. SimControl GUI of our Columbus Simulator

B. Deployment Options

Several different ways of configuring and presenting our system to the users have been demonstrated during the development of the project and it would seem that the best configuration often depends upon how and where it will be used: by trainees or by certified FCT members, in classrooms or in offices, in stand-alone simulations, or in group sessions. As such, we will not provide a 'best solution' here as to how the Desktop Trainer should be delivered; actually one of our key advantages is how flexible the solution can be configured to meet the needs of the users. For training purposes, remote access is an important driver because it means that a trainee and trainer can access the same system without having to physically sit side by side. In the case of the Col-CC training, a dedicated classroom is contained within GSOC, where the only hardware the trainees have access to are standard windows-based laptops. Consequently the solution we provided was one powerful server which contained multiple training environments that could be remotely connected to using the laptops in the classroom. Alternatively, and indeed this solution has been provided at earlier stages of development, would be to host one complete training environment on one physical machine and then clone these physical machines up to whatever scale was required.

Regardless of how many training environments that are virtualised within the desktop computer or server, the Desktop Trainer provides for two different training 'modes' which are both equally important. Each mode is illustrated in figures 5 and 6 below and can be summarised as either a stand-alone simulation, or a group session simulation. In a stand-alone simulation each trainee has access to their own personal training environment, which is completely separated from any other trainee's environment. Of course even in this configuration the trainer would be able to access every display that the trainee was using so that they can monitor their progress.

The second mode is the group session simulation, where multiple trainees are all connected to the one simulator. Each trainee will still have access to their own personal commanding and monitoring clients so they can each work independently, however since they are all connected to the one simulator every trainee's action will influence every other trainee's simulation. In the solution we provided to the Col-CC Training Team, both stand-alone simulation environments and a group session environment were contained within the one server, and therefore it was easy to switch between the two options at will.

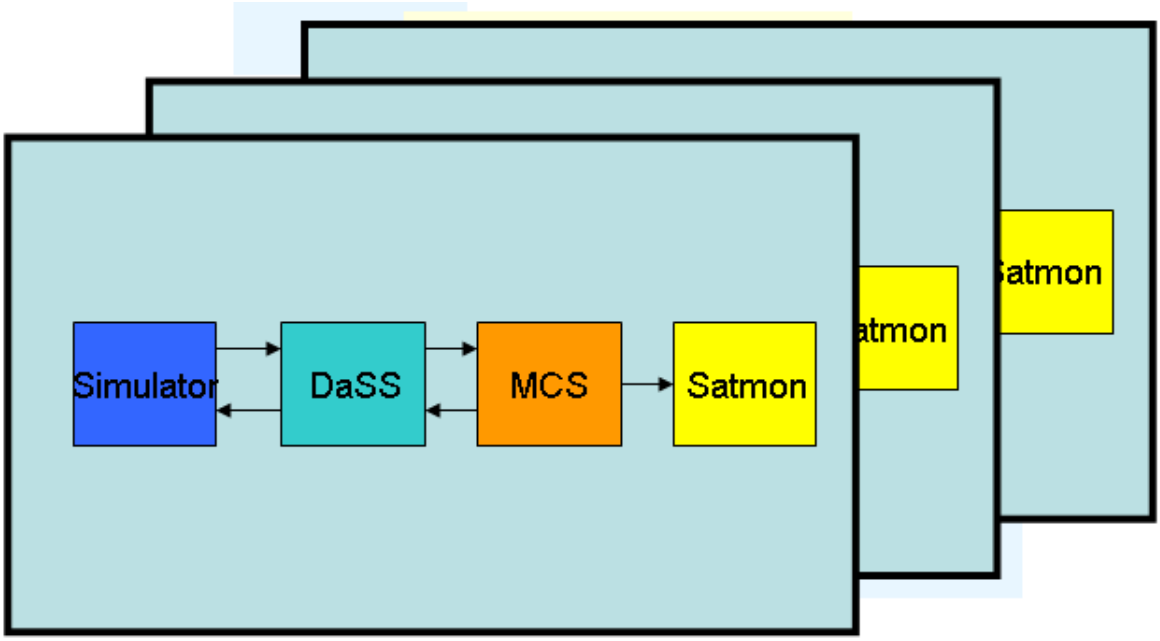

Figure 5. Multiple stand-alone simulations

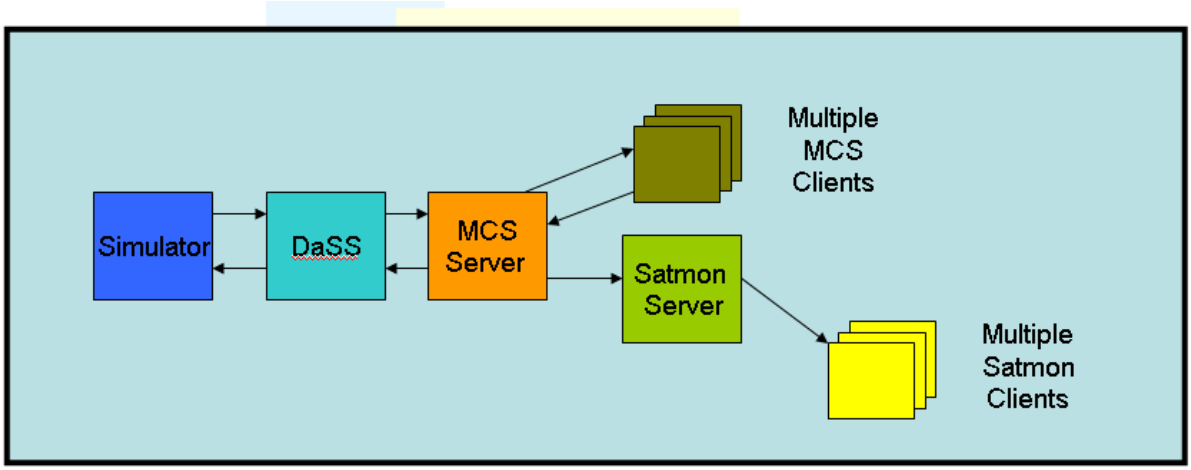

Figure 6. Group session simulation: Single simulator & multiple clients

In the course of developing the Desktop Trainer one problem we encountered was the time-keeping within the various virtual machines contained within our environment. It was essential that each piece of software within the Desktop Trainer was running with the same time to keep the system stable and ensure correct data flow. Each virtual machine, according to its requirements, was running a different operating system; some Linux and some Windows. Initially the virtual machines were running slightly out of sync, and even small differences could build up over long simulations. The solution to the problem, which is in fact now essential to the Desktop Trainer, was to include an NTP server. The inclusion of the NTP server, along with some alterations in the kernel settings in the Linux machines, resulted in very stable time-keeping across the whole system.

IV. Application in Training

The tasks which the Desktop Trainer can cater for can be split into two categories; those which are part of the training process and those which are carried out by qualified FCT members either in support to their daily duties or in the area of preserving and maintaining high operational quality. This section will deal with the former of these two applications.

One of the shortcomings in the current training setup is that trainees need to perform all of their basic monitoring and commanding familiarisation inside a control room. In the case of Columbus, MCS is a fairly complicated piece of software which incorporates a large number of different features which need to be mastered. Consequently the initial training phase can involve a steep learning curve, which when combined with the scheduling issues already mentioned can slow the training process down. The Desktop Trainer effectives provides a sandbox environment for all trainees to spend as much time they require to familiarise themselves with the ground systems, without having to worry about booking a control room in advance. All of the basic yet vital functions such as creating command stacks, applying ground monitoring limits, or looking up parameters in the database, can be mastered; and by taking the process out of the control room and into a much simpler setup, it can be carried out with minimal support. This sandbox environment goes beyond simply familiarising a trainee with the ground software systems, but could also be valuable to immerse a trainee amongst the telemetry of the satellite's onboard systems. Our simulator has been extensively validated against real-time telemetry, and as such a trainer can be confident that by allowing a trainee the maximum amount of time to experience high fidelity telemetry, the training process will be enriched and trainees will be better prepared for actual console work. In general this approach will allow a lot more time to put the theory which is taught by the training team into practice. Currently it is quite normal for a lecture or demonstration to be carried out, followed by a period of several days or even weeks before the trainees have the opportunity to practice again what they learned. If the TQVS schedule is busy, then the training is held up. Our strategy allows for seamless and smooth progress and of course a shorter training period, where quality isn't sacrificed, means savings in project costs.

Sections of the training blocks taught by the Col-CC Training Team are in the form of lectures which are carried out in dedicated classrooms at GSOC. Since connections are not possible to TQVS from this area, there is no possibility to use live simulations as part of these lectures. Screenshots of telemetry displays are the only way to visually illustrate an aspect of a lesson. The Desktop Trainer, either situated physically in the classroom or connected remotely elsewhere, could offer a much more detailed and comprehensive illustration; sensor noise, telemetry fluctuations, deltas and rates of change simply cannot be demonstrated with a screenshot. The use of our system in live training lectures would not simply be more intuitive, but would also be interactive. Onboard scenarios such as corrective procedures, maneuvers, hardware reconfigurations etc. could first be demonstrated by the trainer and then followed up by the trainee being provided with the opportunity to the run the scenario themselves.

For Columbus, as with many space projects, the key indicator of whether or not a trainee has reached the level of proficiency to allow them to be officially certified is their performance in official simulation tests. This is the cornerstone of the training process and currently all of the trainee's preparation has to be performed, and scheduled, on TQVS. Whilst everyone would still have to sit the same certification simulations, and these would ultimately be performed on TQVS, the use of a Desktop Simulator would vastly increase the amount of time the trainees have to prepare and practice the simulations. Consequently the period of time required to certify a trainee would be shortened, and project money could be saved.

One of the most important features of the Desktop Trainer design has been the simulator GUI which allows the trainer to interact very easily with the simulation. The ability to insert failures was one of the essential requirements of the Col-CC Training, and with the Desktop Trainer this couldn't be easier. No complex scripting or coding is required, and no support from specialist staff is required.

A trainer can easily browse the parameter tree contained in the GUI, as described earlier, to locate parameters they want to alter or commands they want to send. Furthermore the trainer's setup allows for them to remotely view the trainee's displays so that they can remotely monitor how and when a trainee responds to the inputs. This simple action and reaction approach is an important method for preparing a trainee to respond correctly to anomalies and unusual conditions that they could encounter on console. These failure inputs can be saved as files using the GUI which not only means that the same failure can be sent to the same trainee if he/she doesn't respond correctly, but also means that the same failures can be sent to various trainees, using their own personal Desktop Trainer, to compare their reactions. Another important advantage of our system, and is related to the previous point, is that loading and saving checkpoints takes only a few seconds. Consequently failure simulations can be repeated over and over without any concern for the time it takes to reset the system. This is also very useful during long procedures; checkpoints can be created at each stage such that any errors far down the line don't result in the need to restart the whole procedure.

One last point to make regarding the training advantages available with the Desktop Trainer would be that with our environment it is possible to have several trainees performing the exact same actions at the same time. In the current training setup, group exercises have to be performed on one simulator and although each trainee will have access to their own personal monitoring and commanding client, the fact that they are all connected to the one simulator rules out the possibility of them carrying out exactly the same action in a lot of scenarios, due to the cross-influence each trainee would have on everyone else's simulation. By deploying the Desktop Trainer in a classroom scenario we could allow multiple trainees to all sit in front of their own personal simulations; if the trainers wanted to teach, for instance, commanding then every trainee could execute exactly the same commanding procedure without having to worry about interference from each other.

V. Support to Daily Operations

Whilst so far the primary focus has been on how the Desktop Trainer could compliment the training process, it also has several convenient uses to the FCT members in support to their daily duties. The key point is the same as already mentioned in that for the FCT to gain access to things like a simulator, monitoring and commanding software, or even a live database they must first book a control room in advance; if the room is already booked and if the schedule is busy then their work must go on hold.

The affordability and scalability of the Desktop Trainer means that we could easily provide every FCT member with access to their own system either physically in their office or, since remote connections are key to our design, remotely connected from their office to a server somewhere else in the building. Sometimes a member of the FCT may need to take screenshots of a monitoring or command display such that it can be used for a presentation or to illustrate in a point in a meeting. Instead of having to prepare for this in advance, confirm that they can schedule a control room, and then needing someone to setup the simulator for them, they could very easily use the Desktop Trainer within a matter of minutes to obtain what they need. This similar ease of access is also applicable to a variety of other tasks which are carried out regularly. Whilst database changes are understandably very strictly controlled, a lot of the preparation work could be done on the Desktop Trainer such that minimum time is actually spent in the control room to make the changes. Occasionally synoptic displays both in Satmon and in MCS need to be updated or created, and again we can take the testing of these displays out of the control room and into the FCT member's office. Similarly the creation and testing of new derived parameters, monitoring scripts, and even the early stages of new procedure development could be carried out from the convenience of an office.

VI. Conclusion

Scheduling time on a simulator and its corresponding control room are often constraints on the training process of satellite operations staff, but in the future it shouldn't have to be. The introduction of the Desktop Trainer into the process could completely remove this limitation, with the result that the period of time required to train new members of the team would be considerably reduced. Whilst the training period would decrease, the actual time that a trainee could spend in a console environment would be much increased; consequently trainees would be much better prepared for real operations and the cost in achieving this would be reduced. Whilst simulators and training tools in other domains have increasingly moved towards a desktop format, the space industry has been slow in this direction.

This low cost and streamlined solution has various important advantages to the training process which are simply not possible with conventional setups. Whilst the Desktop Trainer has initially been developed for the Columbus project we believe that its advantages are universal and should appeal to satellite operators of all types. Large projects, where many staff and groups have to share a limited amount of simulation time, would of course benefit from the easy round the clock access to a simulation environment. However, the Desktop Trainer has been designed to enrich the training process of any mission and offers several appealing characteristics; a simple interface for the insertion of failures by a trainer, very fast loading and saving of checkpoints, the ability to remotely access the environment from either an office or a classroom, and the ability to switch between one group simulation and several individual simulations in parallel, combine to form a training tool which is both convenient and extremely effective.

References

[1]Kuch, T., Sabath, D., "*The Columbus-CC - Operating the European laboratory at ISS.*" *Acta Astronautica*, Vol. 63, 2008 pp. 204-212

[2]Peat, C., Hofmann, H., "SATMON – A Generic User Interface for Satellite Control" URL: http://www.docstoc.com/docs/28410328/SATMON---A-Generic-User-Interface-For-Satellite-Control

[3]Mitchell, C. M., *Horizons in Pilot Training: Desktop Tutoring Systems*, edited by N. B. Sarter and R. Amalberti, Progress in Engineering In The Aviation Domain, Lawrence Erlbaum Associates, New Jersey, 2000, pp. 211-252.