SpaceOps Conferences
28 May - 1 June 2018, Marseille, France
2018 SpaceOps Conference

10.2514/6.2018-2527

A Planning Tool for ISS payload operations and preparations

K. Struyven[1] and A. Diaz[2]
Space Applications Services, Leuvensesteenweg 325, 1932 Zaventem, Belgium

and

N. Brun[3]
B.USOC, Ringlaan 3, 1180 Uccle, Belgium

The Belgian User Support and Operations Centre (B.USOC) was supporting the operations of European Space Agency (ESA) payload Solar Monitoring Observatory (SOLAR) from February 2008 until February 2017. Mounted on an external platform of the Columbus module of the International Space Station (ISS), SOLAR tracked the Sun in order to perform quasi-continuous measurements of the solar irradiance. As the mission was extended beyond its initial planned duration of 1.5 years, a custom in-house tool was developed in order to support and automate the daily operational tasks and maximize the scientific return of operations within the existing external constraints impacting SOLAR operations. By 2012, B.USOC had its own, web-based knowledge management tool developed by the operator team. The tool gathers all necessary operational data from NASA/ESA and internal sources. The first function was to predict the upcoming Sun Visibility Windows defined by the attitude and orbital control of the station, indicating when sun observations could take place. The tool was therefore called Predictor and has been used intensively since its release to automate a lot of the daily routine tasks whilst reducing the operator inaccuracies and errors in these tasks. Additional features were added such as; displaying SOLAR telemetry, ensure configuration control of SOLAR command schedules, on-board file management, support for payload files uplink and transfer, operator shift planning, supporting daily timeline reviews, generating daily operations reports,.... all with a single interface to the operator. While the Predictor tool was all tailored to the SOLAR and ISS operations environment, the B.USOC team developed the idea of a more generic tool that could manage several future payloads simultaneously. With the SOLAR mission ending in February 2017 and other B.USOC assigned payloads Fluid Science Laboratory (FSL) and Atmosphere-Space Interactions Monitor (ASIM) coming up, B.USOC operators started to develop the generic successor of the Predictor tool, a more Generic Planning Tool. This paper will specify the features of the new Generic Planning Tool and the comparison with the Predictor tool, its integration in daily B.USOC operations, and how it applies modern web application development techniques to make a secure and extendable tool for operations support. Eventually, the benefit of such techniques for different control centres is discussed. In one of the first stages of development, the main question was whether the Predictor code would be altered to support other payloads, or a new tool would be made from scratch. Most of Predictor functionalities were specifically made for the SOLAR payload. Practically all code would have been changed. With that disadvantage, you could also start from scratch with the same effort. In combination with the fact that development dates back from 2012, a lot has changed in the meantime concerning how web applications are developed and maintained. Also the sources that Predictor accessed to gather all relevant data have changed over the years. Modern web technologies and communication protocols make it possible to retrieve Predictor relevant information more easily and faster. The retrieved

[1] ISS Operations Engineer, B.USOC, Space Applications Services, koen.struyven@spaceapplications.com
[2] B.USOC Ground Controller, Space Applications Services, alejandro.diaz@spaceapplications.com
[3] B.USOC Ground Controller, B.USOC, nicoloas.brun@busoc.be

information was coming from NASA sources (South Atlantic Anomaly data, comms data, thruster events, ISS attitude, ISS timeline activities ...), internal sources (staff availability ...) and other web based sources (ISS Two Line Element data). New development technologies are also able to support the rising mobile devices compatibility, scalability and increasing security measures. A lot of features from the Predictor can be transferred to the Generic Planning Tool and will be generalized to be able to support multiple payloads such as managing file uplinks and transfers from the ground to the payload, science planning scheduling based on defined constraints, daily timeline reviews, generating Daily Operations Reports (DOR) out of the data stored in the tool, or visualizing all the information an operator would need on console. All these features can be filtered based on what the user wants to see, e.g. the features related to the payload he/she is operating. Other and new features are: tracking of routine daily tasks by means of a checklist, displaying payload Events and TM, long term activity planning, and a fully interactive and configurable timeline where the user can add bands containing all kinds of data from the different sources. Since multiple payloads will be operated by B.USOC, the user can select which TM he would like to visualize on the timeline based on his needs at the moment. B.USOC operators are not only on console, they are also responsible for the payload operations preparations which is bound to schedules and deliverables. To incorporate such long-term planning, the planning tool is also able to support the planning and visualization on the timeline in a way that Microsoft Project can visualize long term planning. Due to all these different types of user usage, the user is therefore able to customize the tool based on his working position on the moment whether it is as a payload operator or as manager for; payloads preparations or increment coordination tasks for example. One of the main changes from a web development point of view that has now widely been adopted, is the strict separation between the client application (front-end) running in the user's web browser, and a service dealing with the database interactions (back-end) on the hosting server. The amount of web services that have such architecture are now almost endless (Google calendar, Google Maps, Alfresco, SharePoint, Wikipedia, LinkedIn, Skype, Redmine ...). This means that the user's own client application can access those services' data and use this data without being forced to use their own client application. Not only has it been decided to create the Generic Planning Tool application based on such architecture, other sources previously fetched by the Predictor have also made the change to this concept. In B.USOC's case, the back-end of the planning tool is developed by Ground Controller (GC) personnel. This service, named Hourglass, is responsible for the user's management, database interactions and operational data gathering. Whilst the client application, containing the B.USOC specific operations functionalities, is developed by the Operator team, being the end-user of the tool. The important aspect of creating modern web applications is to support scalability and movability. Most control centres require the planning of events, managing of tasks, version control of files, managing file uplinks and transfers, importing TM, etc. All sources of information that a control centre needs, are distributed among many origins and can be brought together in one tool to display all relevant data to the end user. How the control centre uses/visualizes these different types of data depends on the operations. It is up to the client application developer to define how they would like to use the data and tailor it for their own needs so routine tasks can be automated. And this model makes it easier for a control centre to develop their own web application, accessing already existing services, thus reducing the cost and skills accompanied with the development of such tool.

I. Nomenclature

API = *Application Programming Interface*
ASIM = *Atmosphere-Space Interactions Monitor*
B.USOC = *Belgian User Support and Operations Centre*
CORS = *Cross-Origin Resource Sharing*
DOR = *Daily Operations Report*
EC = *Experiment Container*
ESA = *European Space Agency*

FSL	=	*Fluid Science Laboratory*
GC	=	*Ground Controller*
ISS	=	*International Space Station*
LAN	=	*Local Area Network*
REST	=	*Representational state transfer*
SOLAR	=	*Solar Monitoring Observatory*
SDM	=	*Soft Matter Dynamics*
TM	=	*Telemetry*
USOC	=	*User Support and Operations Centre*
VMU	=	*Video Management Unit*

II. Introduction

The Belgian User Support and Operations Centre (B.USOC) is responsible for the planning and execution of multiple payloads on-board the International Space Station (ISS). The Solar Monitoring Observatory (SOLAR) has been operational since 2008, and although the mission would only last for 1.5 years, operations only ended by 2017. It is mounted on the external platform of the Columbus module, and contains 3 instruments. Due to its mechanical limitations, observations with the instruments were not possible for a continuous time. To optimize the scientific return, planning was key to gain the most out of the limiting time to perform observations. In order to support this planning, the web-based Predictor tool was developed to support planning and operations execution. It started out as a Sun Visibility predicting tool based on ISS data, it became quickly an all-round tool to provide support to most of the operational tasks of an ISS operator. Some of these tasks include the generation of the Daily Operations Report (DOR), resource planning, scheduling SOLAR observations/calibrations, configuration control of the files uplinked to SOLAR, support the uplink/transfer and downlink of files according to the ISS flight operations processes and timeline reviews from the ISS timeline, etc. [1, 2]

Next to SOLAR, B.USOC is also responsible for the Fluid Science Laboratory (FSL) and the Atmosphere-Space Interactions Monitor (ASIM) which require up to 24/7 operations. FSL is a science payload for conducting fluid physics research in microgravity. With a new Video Management Unit (VMU) MkII installed in FSL, it is ready for a new wave of experiments. The Soft Matter Dynamics (SMD) Experiment Container (EC) will research compacted granulars dynamics and foams generated in microgravity. ASIM is an Earth observation facility situated on the external platform of the Columbus module. It will study transient luminous events and terrestrial gamma-ray flashes situated above thunderstorms. These payload can be seen in Figure 1.

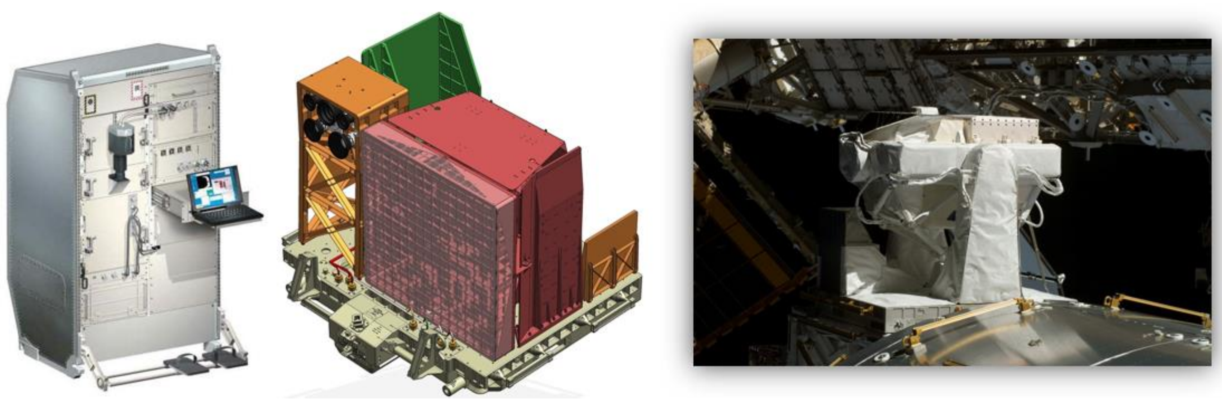

Figure 1. From left to right; FSL, ASIM, SOLAR

These operations are supported by the newly developed Generic Planning Tool which takes SOLAR's Predictor tool to the next level by making its functionalities more Generic to support the operations and the preparation of multiple science payloads in order to avoid a dedicated tool for every payload. This paper presents the operational and technical aspects of the Generic Planning Tool and will not go in any further detail about the specific science payloads since it should apply to any ISS operated payload. Section III will focus on the ISS payload preparations, the long-term planning. Section IV envisages the ISS payload operations, the daily activities of the payload's on console operator. Section V describes the technical aspect of the Generic Planning Tool. And eventually, section VI describes the summary and outlook of the tool.

III. ISS payload preparations

Each payload assigned to a USOC undergoes an integration process until it is launched and eventually operated.

The payload operations fall under the general organization of the ISS Operations which is done by ISS Increments. An increment defines a specific time on-board from crew vehicle undock to next crew vehicle undock. Each increment has its crew assigned and therefore, it should be known which crew should execute what experiments. This information is important in order to plan ahead the required crew time, crew training and ISS resources. When considering the timeline of an increment, it has a countdown to the start of the increment in months (I-x months) and as soon as the increment starts, the indication of the increment week number is kept (wk x).

As soon as a payload is assigned to a specific ISS increment, already a significant amount of deadlines for deliverable products and design/reviews is set.

Next to the actual work of reviewing/writing and creating the deliverable products, it is key for the operations team to keep a high level overview of the upcoming deadlines and assess the subsequent workload. A typical tool used to maintain such long term activities, constraints and deliverables is a Project management tool (such as Microsoft projects). Mainly used by project managers, they insert a work breakdown structure, plan ahead and define critical paths. Similarly, for each payload, a project managers keeps track of everything and has a complete overview of the status of the payload preparation from all the stakeholders involved. Since the USOC is only interested in the operational preparatory work and increment deliverable deadlines, a Project Management tool is usually not incorporated as a required functional tool in the operations environment. This is one of the aspects the Generic Planning Tool focuses onto, namely on the long-term activities and incorporates such events in a timeline view to get the same high-level overview as what is typically required by the project manager. There are plenty of tools to maintain events like Google calendar, Outlook, etc. The problem arises in how data can be displayed to the user (by means of a timeline) and not specifically what has been put in the tool. Part of the Generic Planning Tool is to view your calendar events and milestones in a timeline matter like you can visualize them in a typical project management tool. Figure 2 shows an example of an increment's activities schedule. The Generic Planning Tool Timeline is made in such a way that the user can define the view entirely. Each row is called a band and can contain events and specific time information. Bands can be added, deleted, moved up and down to create a user's Timeline depending on what and how they want to visualize the data. For the specific case as seen in Figure 2, 3 bands have been added to the Timeline;

- The UTC timescale.
- A 'increment timeline' band that requires input from the user; the increment start and end date which will then display the increment's timing expressed in months (before increment start) and week numbers (after increment start).
- A band with specific events related to the increment based on an activity's categories.

The major difference between activities you put in any other calendar or in the Generic Planning Tool is the categories field that contains predefined categories (tags) you can give to your activities. In such way, you can distinguish your activities and group them in a band.

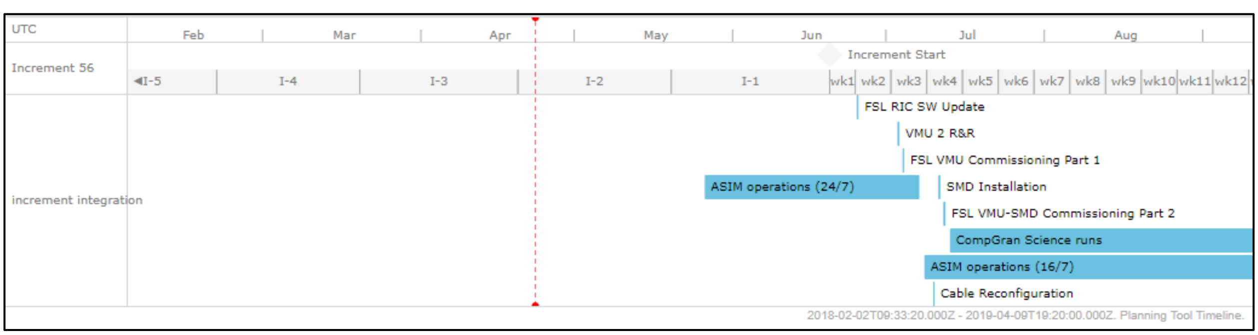

Figure 2. Timeline snippet of the Generic Planning Tool incorporating long term increment activities.

Next to events and deadlines, users have the ability to link specific tasks to activities. Minutes of meetings usually have corresponsding action items which can be translated immediately to a task assigned to a user, deadlines usually refer to documents that need to be written ,etc. There are a lot of task applications available, but the main advantage is to have them linked to an activity within the same tool as displayed in Figure 3 where the details of an event can provide an overview of the linked tasks.

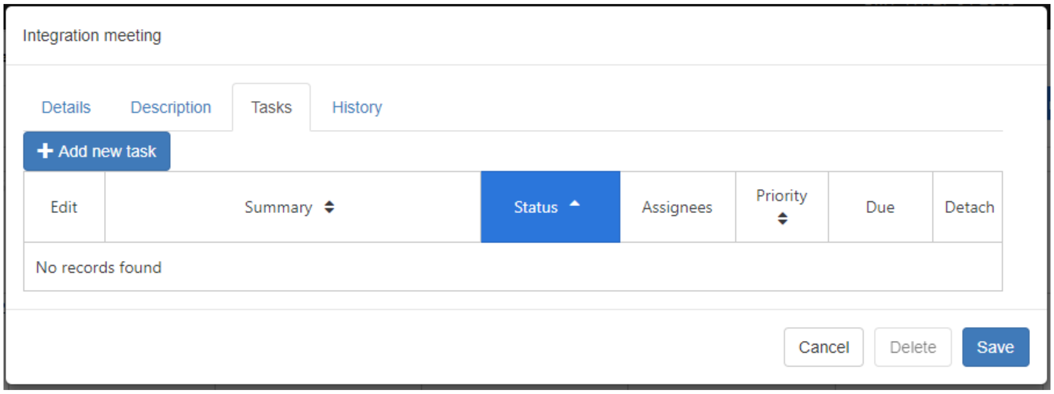

Figure 3. Details of an activity with attached tasks.

IV. ISS payload operations

Once a payload is installed on-board the ISS, the operations phase starts and all the specific on console tasks need to be executed on day to day basis. B.USOC operators are on console 24/7 to monitor and control the payloads. Based on the input from the scientists, operators execute science measurements, observations and calibrations. All commanding is executed within a so-called Real-Time Commanding Window. An activity in the OPTIMIS Viewer lasting 24 hours and is planned whenever the payloads are ON. It provides the operator an immediate access to command the payload upon the go from the Columbus Flight Director (Col-FD).

This section will go through the on-console tasks of an operator that are supported by the Generic Planning Tool.

1. Short term planning

Regardless which payload the operator is commanding, the activities need to be coordinated in advance and defined when they will be executed. Just like the long-term planning of events, the short-term planning consists of a series of activities that need to be maintained and tracked. Whether these activities are observations, calibrations, an instrument activation, and power down, etc., they all have common fields to be filled out during the planning. A start/end time, summary, description ... The Generic Planning Tool adds a few fields to make them more descriptive and distinguishable during operations planning;

- Categories (tags) field has been added to categorize the events. This will make it easier to group and filter later on.
- Executed start/end time will make the operator able to enter when exactly the activity has started and ended.
- A status field can inform them whether an activity is scheduled, ongoing, completed or cancelled.
- The event history. Version control is key when planning data, operators can track who changed what and when. This is why the back-end service tracks every change made in the event. Users can easily see this and revert back to a certain condition from the past.
- User data fields. Sometimes, an activity is very specific for an instrument and requires more descriptive information for the execution. The User data field is actually a way to have the user define its own fields. Fields can be added or removed by the user depending on the activity's needs.

2. Timeline review

Another of the operator's task is to review the timeline that has been put into the OPTIMIS database by the ISS planners. This task is performed by checking the OPTIMIS Viewer timeline and compare it with our planned activities. For routine operations, these activities usually consist of the power resources, a real-time ground command window and file uplink/transfer or downlink activities. These activities will be planned as long as the payload is operational and the review can therefore be cumbersome and prone to human error when performed manual. The Generic Planning Tool will therefore allow the operator to define a list of planning inputs that should appear in the OPTIMIS timeline. OPTIMIS data is synchronized multiple times per day with the back-end service. Together with the inputs from the operator, the Generic Planning Tool will inform the operator in case there is a mismatch between planning inputs and scheduled activities or whenever the planning is correctly present in the OPTMIS database. The Timeline viewer supports the display of specific OPTIMIS bands with a content filter to only extract specific activities as demonstrated in Figure 4 .

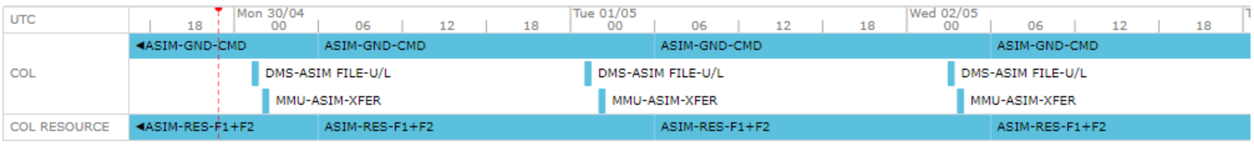

Figure 4. Recurring activities during nominal payload operations.

3. File maintenance, uplinks/transfers and downlinks

Some payloads make use of configuration files, or command schedules that will reconfigure the instruments or run command schedules to run measurements/observations automatically. These files have to follow the Columbus File Transfer process, consisting of the file to be uplinked to a specific slot on the Columbus Mass Memory Unit (MMU) and eventually be transferred to the payload. Only dedicated filenames can be used to identify the files and uplink them on-board. Moreover, a file has to be available to the Columbus Flight Control Team (FCT) at least 8 hours in advance before the STRATOS uplink activity from ground to the MMU is planned. This process is depicted in Figure 5. Additionally, the file's checksum is used to uniquely identify a file. This whole process can be reversed in case a file has to be downlinked from the payload to the USOC. [3]

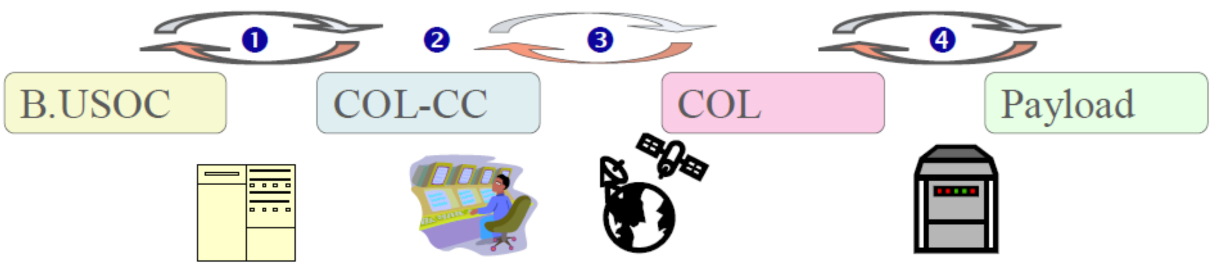

Figure 5. Columbus File Transfer process for file uplink and downlink.

The Generic Planning Tool will support the operator maintaining the files at each location of the process. Files that are uploaded in the Generic Planning Tool are 'ground files', the status of the MMU (which files are on which MMU slot) and which files are on the payload. Files can be maintained in a flat manner, like in a table, or as we are used to it, in a directory structure. The Generic Planning Tool allows the user to define a location of the file to mimic a directory structure and the ground and on the payload. Figure 6 gives an example of the ASIM payload files structure. The Generic Planning Tool provides this information in a visual manner that is easy to grasp and provides the user with a good overview of the status. From left to right on Figure 6; there is a directory structure for the files in the system, indicated with the 'Ground' header, a list of the planned uplinks, the layout of the Columbus MMU, a list of scheduled transfers/downlinks and eventually the list of the files that are present on the payload.

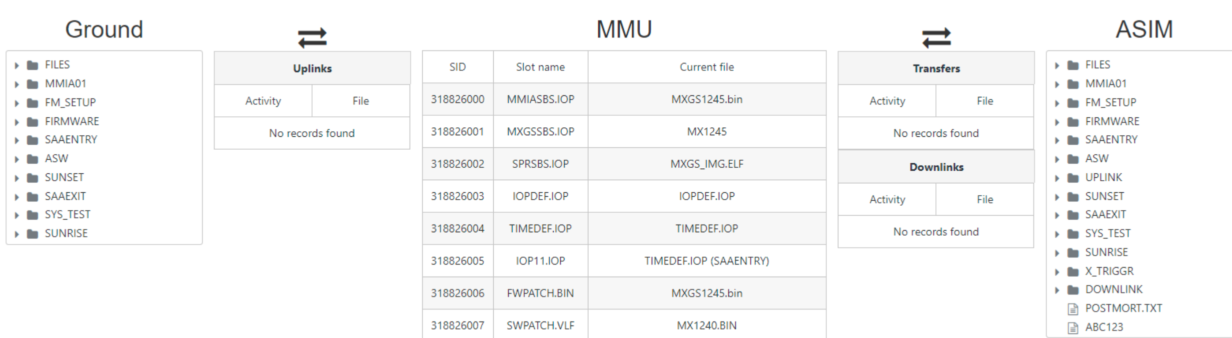

Figure 6. File maintenance in the Generic Planning Tool.

4. Timeline viewer

One of the main views an operator will keep an eye on, is the OPTIMIS Viewer. This timeline provides a good overview of what is important to ISS operations; ISS Day Night schedule, South Atlantic Anomalies (SAA) passes, communication data, crew activities, payload resources… An example of the OPTIMIS Viewer is shown in Figure 7.

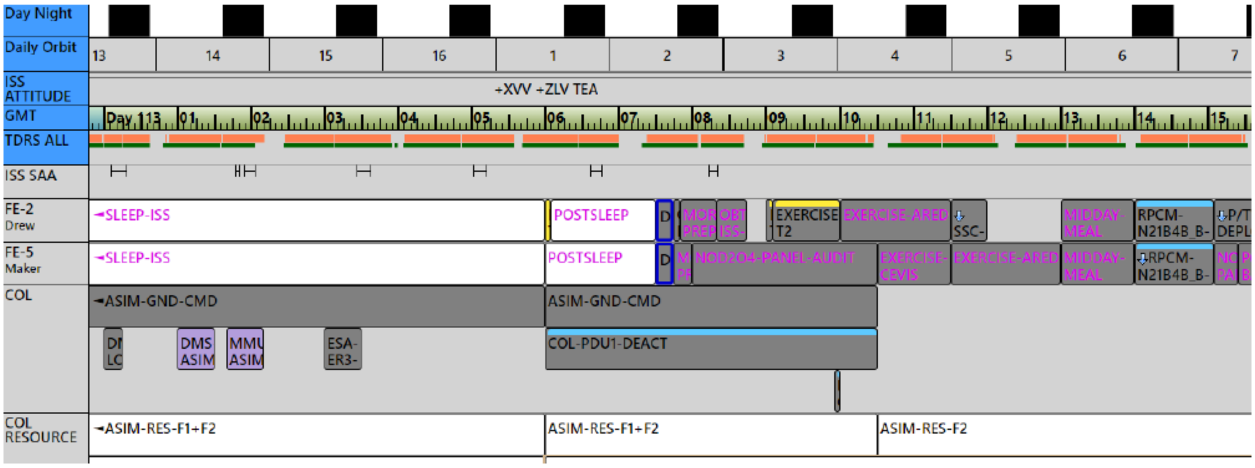

Figure 7. Snippet of ISS OPTIMIS Viewer data.

Some OPTIMIS data is synchronized with the Generic Planning Tool's back-end service's database to provide the operator with access to OPTIMIS data in a different way. OPTIMIS Viewer is a read-only timeline with a limited set of data. As soon as the data is copied to an internal data source, it can be viewed/used in a completely different way, it becomes interactive. All of the USOCs have access to OPTIMIS Viewer but next to this data, payloads have their own planning which is not part of the public OPTIMIS data. It could be that the USOC is interested in other USOC-only events, like outages, maintenance activities, B.USOC Shiftplanner data, instrument status, … For these additional activities, USOCs tent to use different tools like spreadheets and calendar applications. In order to cope with those activities, the Generic Planning Tool has a calendar function of which the activities can be categorized. Next to these custom activities, it is the intention of the Timeline viewer to acquire as much data as possible from different sources and display them in a customizable timeline for the operator. Other data that can be displayed in the Timeline are; the B.USOC Shiftplanner data, Google Calendar data, instrument Telemetry/Events/Command history. The latter is coming from the Yamcs MCS deployed at the B.USOC. [4] Yamcs has an interface for applications to access the archive or provide a connection to display live Telemetry. An example of a customized timeline for the operator is provided in Figure 8. This timeline displays data from; OPTIMIS, own custom activities and Yamcs archived telemetry. When viewing all this information on top of each other, it can significantly help the operator to fine-tune the planning of a payload.

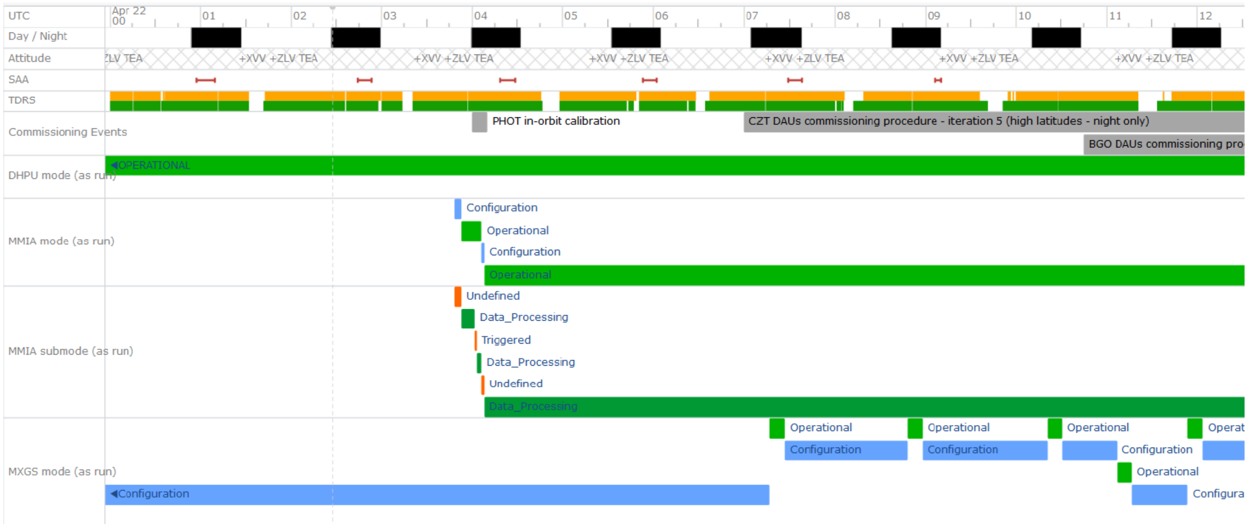

Figure 8. Customizable Timeline viewer.

5. Daily Operations Report

The payloads stakeholders and backroom support receive a Daily Operations Report (DOR) of the executed payload activities. It contains a summary with a high level written overview of the day activities, a detailed description of each performed activity and an overview of the open anomalies. The DOR is automatically generated by the Generic Planning Tool with information provided by the back-end service and the timeline is used in the DOR to visually report the performed activities. When the operator marked the activity complete and filled out the actual start and end time of an activity, the tool can assemble a list of activities. The open anomalies are kept as an entry in the task page having a specific predefined category. Based on the open anomalies, the latest status is fetched. In addition, the relevant status information of the payload, as defined by the operations team, can maintained in the tool. An example of this is the software version of the payload's Application Software (ASW). The only input remaining to be filled out by the user is the written summary.

V. Technical aspects

The main question at the beginning of the development of a new planning tool was whether the existing Predictor tool would be elaborated and extended for the use of multiple payloads, or a new application would be built from scratch. After a thorough assessment, it was quickly clear that the amount of modifications needed to update the Predictor would rise so high that it would be easier to build a new application. The Predictor tool was built on the Linux, Apache, MySQL, and PHP (LAMP) software stack model as illustrated in Figure 9. All scripting performed at one place and the back-end (database interactions) and front-end (GUI) was situated within the same code files. [2]

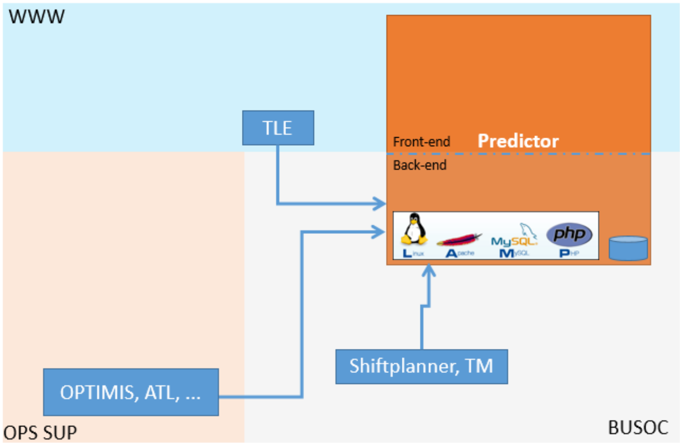

Figure 9. Schematic of the Predictor tool architecture using the LAMP model.

In the meantime, a lot has changed in the landscape of web applications and because of the rise of mobile platforms, the number of frameworks to build single page applications has boomed. In order to take advantage of this increasing availability of frameworks, the single page application type had been chosen. The biggest change compared to the Predictor tool's architecture is the way data is communicated and how code is separated. There is no single point that runs the entire application. Data services and GUI has been split up, each responsible for one part of the tool. Figure 10 illustrates the architecture and shows the different building blocks of the whole system. The single page application itself will be fetched from an http server by the client's browser. This application will run entirely in the user's browser without having the data used by the tool. This part is called the front-end or GUI and contains the interactions with the user and fetches data from as indicated with the double blue arrows in Figure 10. This part is developed by the operator team at B.USOC because it is developed based on what the operators want to see and how they will use the Generic Planning Tool. The back-end side consists of the planning tool's custom database named Hourglass, developed by the BUSOC GCs and contains the user authentication, storing of events, tasks, OPTIMIS data, etc. This method of communication between the front-end and back-end is part of the client-server architecture. The main principle of this architecture is separation of concerns. If the user interface is separate from the database, it improves portability of the interface over multiple platforms. It is more scalable, making it possible to have one grow without the other being affected. It is expected that the user interface of the Generic Planning Tool will continue to change based on the requirements of the payloads and on console tasks without altering the back-end service. The client/server have to communicate according to web standards. Representational State Transfer (REST) is an architecture style that defines these constraints of communication. It provides a way of computer systems to communicate with one another on the internet. A RESTful web service can manipulate web resources by using a set of stateless operations. This method aims for fast performance, reliability and the ability to have different applications grow independently without affecting the whole system. It is a now widely adopted concept that is still increasing in popularity throughout the web. The Generic Planning Tool front-end is thus a standalone application that alters data from different resources. From the Generic Planning tool front-end point of view, the Hourglass back-end service dealing with internal B.USOC data is thus just another web service that it makes use of. Next to the Hourglass service, the front-end also access' Google Calendar data from the regular internet, payload archived Telemetry from the B.USOC Yamcs server, internal Shiftplanner data, etc. Each of these components work individually and can be used stand alone or by other/new applications if needed.

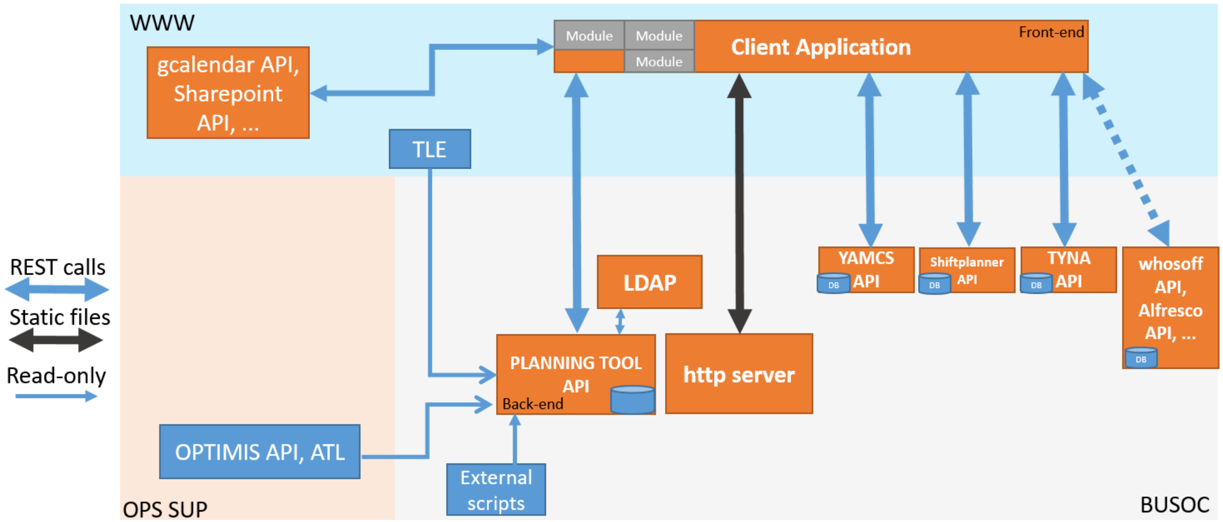

Figure 10. Schematic of the Generic Planning Tool, providing a strict separation between the front and back-end.

A. Back-end

The back-end named Hourglass is, specifically developed to deal with the Generic Planning Tool's needs. It provides a REST Application Programming Interface (API) to the front-end client application. Authentication is provided by means of a JSON Web Token (JWT), according to RFC7519. The database scheme and objects are designed according to RFC5545.

Hourglass has been written in the Go language because it is strongly typed, compiled and it is one single binary which makes it easy to deploy and distribute. The back-end is only fetching data from the database, there are no hard computations which makes it easy to maintain and speeded up the development.

The Hourglass service is managed via systemd. Systemd manages the process and restarts it if the application exits with an error. It is running behind an apache server via proxy directives. All the services accessed by the front-end application are proxied behind the Apache server meaning there is only one single gateway to get the information needed for operations from all the different sources.

B. Front-end

The Front-end/GUI is responsible for the interaction with the users and is a stand-alone single page application. To adapt to the fast changing requirements and 'easy to update' modular methodology, the Angular 2+ JavaScript framework was chosen in order to speed up development and comply with present security and performance requirements like the mobile market. It works cross platform and with modules making it easy to add independent blocks of code. Like any other modern front-end framework, Angular adapts the Model View Controller (MVC) pattern in its architecture.

- **Model**: The data received by the REST API from a server. Usually JSON data, which already consist of objects that can be directly used by the application.
- **View**: HTML, and the directives when the Document Object Model (DOM) needs to be manipulated.
- **Controller**: Queries the REST API, provides callbacks to respond to events.

Next to the Angular core framework, a lot of other JavaScript libraries have been used to ease development and to display the data according to the user needs; fullcalendar, bootstrap, primeng, jwt-decode, moment, timeline…

When dealing with front-end web applications that need to fetch data from different web services, a mechanism called Cross-Origin Resource Sharing (CORS) comes into the picture. CORS uses HTTP headers to let a user application gain permission to access resources from a server that is on a different domain (origin) than the application's host server. For security reasons, browsers restrict cross-origin requests initiated from the user application scripts. The only way to solve this issue for the user's application is to enable CORS on the data server. This issue has been popped up and fixed for the Hourglass web service and the B.USOC Shiftplanner.

C. External sources of information

As shown in Figure 10, the front-end fetches a lot of information from different data sources. Some of these sources are located on the secure Ops Support LAN at B.USOC; OPTIMIS data, TOPO data. Others are situated within the local B.USOC network; Hourglass, Shiftplanner and Yamcs data while others are present on the internet; Google Calendar. Hourglass will only collect the OPTIMIS data and store it in its own database. Because OPTIMIS data is located in a secured environment, it cannot be accessed directly from the front-end application without the necessary special access. All other data is fetched directly from the front-end to the available REST API service. In order to comply with REST protocols, the B.USOC internal Shiftplanner has been updated to cope with REST calls so the data can easily be manipulated by other tools like the Generic Planning Tool front-end application. Yamcs server has a REST API available making it already easy to integrate it in web applications. [http://www.yamcs.org] [4]

D. Security

The planning tool has been placed inside the Demilitarized Zone (DMZ) network of B.USOC. Operators can access it from computers connected to a secured VPN connection over the Internet. Having the need to interface several external sources of information, some of them available on the Internet, and due to the criticality of some of the other sources of information accessed and contained in this tool, the host machine need to implement the proper preventive countermeasures. A firewall performing stateful packet inspection was put in place to monitor and secure the system from potential threats.

E. Scalable environment

The planning tool runs on a scalable Linux Ubuntu Long Term Support hosted on a VMware ESXi virtual machine. This environment provides redundant high availability ensuring that the application is hardware fault tolerant and available 24/7, to cope with B.USOC continuous operations.

By using VMware's virtualization technology, if an extraordinary increase in processing power is needed by the planning tool, it is possible to quickly allocate more resources to the guest virtual machine hosting the planning tool.

VI. Summary and outlook

The Generic Planning Tool will still evolve in the future due to the changing requirements and required improvements on the user/operator side. By separating the front-end from the back-end and by making use of REST calls between client and server, only one part needs to be updated without altering others. It's all about the interface between the different parts that should remain the same. And it is important to keep in mind that the Generic Planning Tool's functionalities, look and feel are purely defined by the front-end application that makes use of all these different web services for data collecting and storing. The Generic Planning Tool is just the tip of the iceberg of a great ecosystem that can be used by other control centres to develop their own custom tool relatively fast and easy. If more data sources should be taken into account by a specific payload, it is a great asset if this data is accessible by means of a REST web service because only the front-end application will need to be changed to fetch the data. When developing future data services, it is important to keep in mind that the user side will most probably be a web application that makes use of these principles.

Acknowledgments

The authors thank the ESA ISS Operations Team within the Directorate of Human and Robotic Exploration and the Belgian Science Policy Office (ESA Prodex and other programmes) for their funding. B.USOC is part of the Royal Belgian Institute for Space Aeronomy (BIRA-IASB). Further we thank the ESA Payload Operations Management, the former ESA Mission Science Office, the Columbus Control Centre Flight Control Team, the SOLAR, FSL and ASIM payload developers and Science Teams, and the B.USOC for the good collaboration and support.

References

[1] A. Sela, A. Michel, "The SOLAR Mission Tool - an integrated ops tool," *Proceeding of SpaceOps 2012*, AIAA, Sweden, Stockholm, 11-15 June 2012.

[2] A. Diaz, J.-M. Wislez, S. Klaï, C. Jacobs, D. Van Hoof, A. Sela and A. Karl, "SOLAR Predictor: A Knowledge Management Tool Supporting Long-Term Console Operations," *Proceeding of SpaceOps 2014*, Pasadena, CA, USA, 2014, 5-9 May.
[3] Columbus Control Centre, "Joint Operations Interface Procedures", OPS-PRO-1IF-120-GSOC
[4] A. Sela, M. Mihalache, D. Moreau, "YAMCS - A Mission Control System," *Proceeding of SpaceOps 2012*, ID-1280790, Sweden, Stockholm, 2012, June.