PINTA and TimOnWeb – (more than) generic user interfaces for various planning problems

Rainer Nibler, Falk Mrowka, Maria Theresia Wörle, Jens Hartung, Christoph Lenzen

Deutsches Zentrum für Luft- und Raumfahrt e.V., German Aerospace Center,
Münchener Straße 20, 82234 Weßling, Germany

Chris Peat

Heavens-Above GmbH, Pfingstrosenstraße 2, 81377 München, Germany

In the recent years, the "Program for INteractive Timeline Analysis" PINTA and the timeline visualization web-client "TimOnWeb", formerly also known as "Timeline Offline Navigator", developed at the German Space Operation Center (GSOC), were continuously improved and experienced several evolution steps. PINTA is a GUI application running on Windows-based computer systems. Its main purpose is to serve as the anchor tool for a mission planning operations engineer when generating or analyzing a mission timeline, including the possibility to call automatic planning and scheduling algorithms of the embedded generic planning library PLATO. TimOnWeb is a web application running on all common web browsers, allowing a remote user or even a remotely connected mission planning engineer to have insight into the current operational timeline as for example generated by PINTA and PLATO, relying on the same data representation. PINTA and TimOnWeb are the generic basis of many semi-automated mission planning systems for past, current and future spacecraft projects operated at GSOC, for example for the missions Grace, TET-OOV [12], FireBird [13], Grace-FollowOn and Eu:CROPIS. Furthermore, PINTA serves as the timeline analysis tool for validating the TerraSAR-X/TanDEM-X mission planning system [14]. The variety of use cases was further extended to support the planning of the on-call shifts at GSOC, and to support Launch and Early Orbit Phases (LEOPs) in its special configuration as the new generic editing tool for the so-called "Sequence of Events" [15]. The paper at hand will explain these use cases and the responsibilities of the tools, while giving an overview of PINTA, its components and their set-up, with laying an emphasis on the latest development achievements.

I. Introduction

The applications PINTA and TimOnWeb have a history of up to two decades serving for various spacecraft missions operated at GSOC. However, or rather, of course, they are undergoing a continuous evolution to be prepared for supporting multiple use cases with different degrees of manual interaction.

Figure 1 gives an overview of the set-up of PINTA with some of its plug-ins and its relation to our planning library PLATO. More details will be presented in chapter II, while showing the typical current use case of PINTA serving as the anchor component for semi-automated "standard" mission planning systems at GSOC with TimOnWeb being the visualization frontend for broadcasting the planning results via Web. Furthermore, it will be explained how PINTA relies on the GSOC modeling language, and how it works for setting up planning models.

Currently, so-called "Pinta/Plato for [mission XYZ]" core planning components are in use for Grace, TET and BIROS within the FireBird mission. Already implemented are the planning tools for the upcoming missions Grace-FO and Eu:CROPIS. A similar future system is planned e.g. for S2TEP. Especially challenging and currently under

development, it is foreseen to provide a mission planning solution for the Galileo Service Operator in the near future based on PINTA and TimOnWeb.

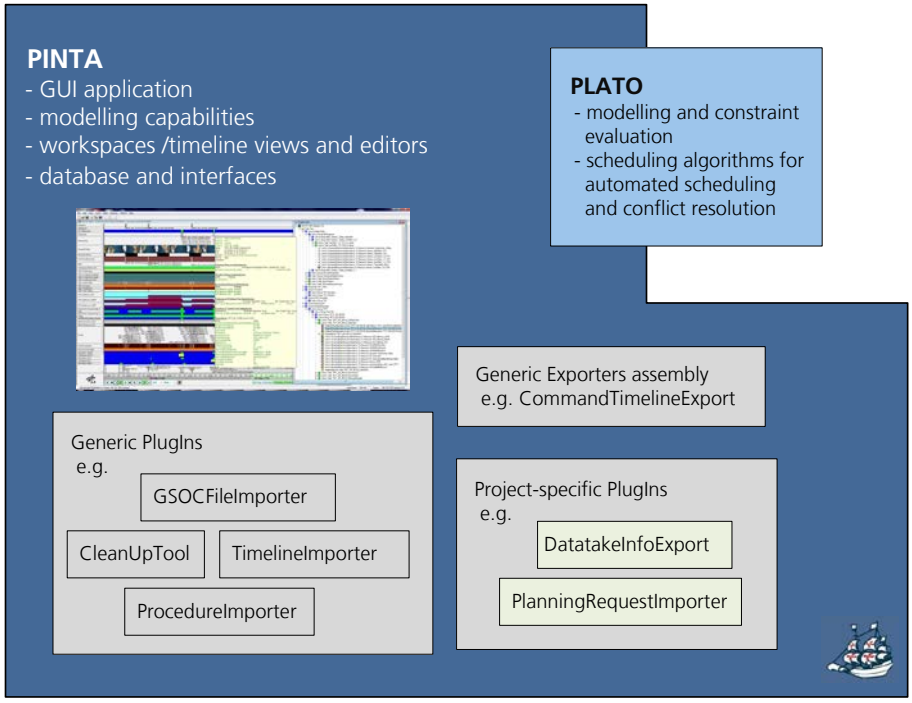


Figure 1: *The assembly of PINTA*

Furthermore, PINTA is also used for more complex missions with automated mission planning systems for analyzing the resulting command timelines including a model-based crosscheck for constraints, which is used for validating the TerraSAR-X/TanDEM-X Scheduler and CommandExport output, and is shortly described when addressing special import functionalities in chapter II.

The set-up and functionalities of TimOnWeb are described then as well in chapter II. On the one hand, it is used for visualization of the current command timelines and related information for each of the missions accessible via web, whenever applicable, depending on the mission requirements. On the other hand, TimOnWeb has to allow for displaying the so-called Sequence of past, current and upcoming events on the big screens in a control room when being part of the operational environment during LEOPs and LEOP preparation activities.

A relatively new feature of PINTA is its SoEEditor functionality for preparing and maintaining the Sequence of Events before and during the launch phases for low-earth as well as geostationary missions at GSOC is applied. It is described in more detail in chapter III including the changes that had to be implemented to face the new approaches and workflows related to this.

The high degree of configurability of PINTA and TimOnWeb is one of the main advantages needed for being able to support the various planning and scheduling demands with a variable degree of automation, and the further enhancement of the user-friendliness for external and/or non-expert users, is one of the main development drivers. For this reason, more emphasis will be laid on the recently completely reworked and extended exporter functionalities of PINTA. It will be explained how they are set-up and especially how it is achieved to allow for a very generic and extensive configuration of the different export formats in chapter IV.

Last but not least, in chapter V, a short survey of similar developments of GUI-based and/or GUI-supported planning and scheduling applications at other agencies and by commercial providers in the spacecraft operations environment can be found and an outlook to the future challenges and milestones for the development of PINTA and TimOnWeb is given.

II. The components of a typical, low-cost mission planning system at GSOC

Figure 22 gives an overview of a typical semi-automated mission planning system at GSOC (e.g. [13]) and shows how it is embedded within the mission operations ground segment and which internal and external interfaces are to be served in general.
In the following, these components and their main functionalities and set-up will be described in more detail.

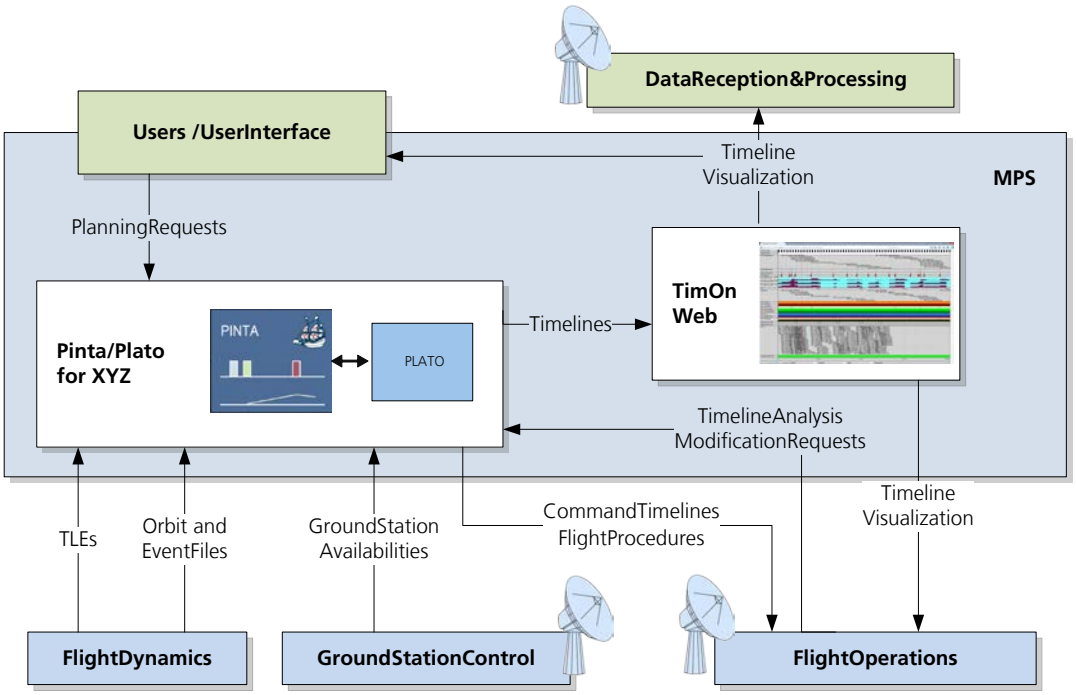


Figure 2: *Generic setup of a mission planning system with PINTA, PLATO and TimOnWeb.*

A. PINTA – Generate a timeline not only interactively

The core planning components of such a mission planning system are often called "Pinta/Plato for [mission XYZ]". All of these share a similar set-up and as their naming tells, they comprise GSOC's generic "**Program for INteractive Timeline Analysis**" PINTA and generic planning library PLATO ("**PLAnning TOol**").
Both operate on a representation of the respective mission's planning problem, the so-called planning model.

B. The Planning Model

A GSOC planning model is also referred to as the PINTA or PLATO or simply current "**Project**". In [17], a detailed overview of the modeling language can be found. Briefly described, the model consists of objects like "**Groups**", "**Tasks**", "**Parameters**", "**Resources**" and all the constraints that should be considered. Among others, the most important constrains are "**OrderedTimeDependencies**" between the Tasks, "**UpperBounds**" respectively "**LowerBounds**" of the Resources and "**Allocating-**", "**Accumulating**"- and "**ComparingResourceDependencies**" between the Tasks and Resources, with all but the inter-Task constraints being specified via "**Profiles**" over time. Scheduling and un-scheduling of the Tasks is done by adding/removing "**TimelineEntries**" to/from the Project's "**Timeline**". The Resource modifications/comparisons become active as soon as a Task is scheduled, while the constraints of the other Tasks already scheduled as well as the Resources themselves can indicate conflicts, i.e. whether or not the modification led to a conflict-free new Project state.

The so-called "**StartUpProject**" with the initial setup of the planning model typically comprises a Group and Task structure to allow the classification of input items (like Flight Dynamic events, Planning Requests, etc...) and planning process output. Furthermore, the StartUpProject already contains the Resource model that will have to be

used and considered throughout the planning process or for display purposes, including the initial fill level and potential Upper- and LowerBound Profiles of the Resources.

After the initial setup, so-called importers might add further Group-, Task-, Resource- and constraint –structures. Additional changes can be produced by the mission planning operations engineers manually and the scheduling algorithms automatically by creating, shifting, or removing TimelineEntries of the Tasks to/on/from the Timeline, which can then cause or remove conflicts on the Resources and in-between the Tasks, depending on the respective objective.

In general, the approach of a mission planning system and its planning runs is to transfer an old into a new Project state while maintaining a conflict-free Timeline to be further processed.

The typical planning model is composed and displayed in PINTA via the so-called "**Project Tree**". As an example, an excerpt of the planning model for the FireBird mission can be seen in Figure 3.

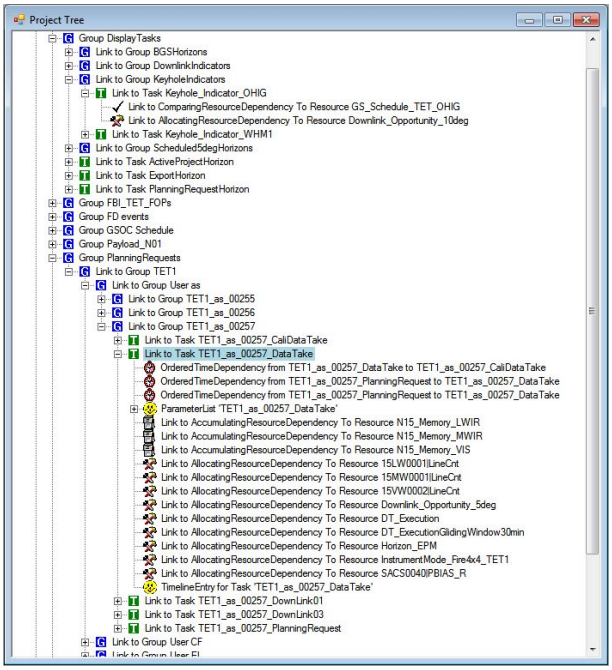


Figure 3: *The PINTA Project Tree displays an excerpt of the planning model for the FireBird mission.*

PINTA at GSOC is used for all projects and/or phases of projects whenever a (semi-)manual planning and/or (semi-) automatic inspection or crosscheck of an automatically generated planning result is required. It is a GUI application that can be used to interactively create and modify any planning model using the GSOC modeling language [17]. The planning model and the current timeline can be visualized and modified e.g. via drag-and-drop of model contents. Figure 4 shows an example snapshot of the PINTA GUI.

Figure 4: *Example snapshot of the PINTA GUI showing an FireBird mission timeline*

C. PINTA and its Plug-Ins

To extend the basic PINTA functionality with additional features, the plug-in interface is used. Generic plug-ins used within a typical mission planning system are:

- the "**CleanUpTool**": Before every planning run the current planning model state has to be cleaned in order to avoid runtime problems due to too much information stored in it and the underlying database. It is configurable for which model contents and up to which point in time the clean-up shall take place. This way it is ensured that only information is autonomously deleted that has no influence on the current planning state and future planning runs anymore. Previous states of the model are archived.

- the "**GSOCFileImporter**": This plug-in is invoked to ingest input files from other parts of the mission operations ground segment (see Figure 2). On a regular basis new Two-Line Elements and Event files from Flight Dynamics and so-called Schedule files from the ground-station scheduling office are received, and at the beginning of a planning run the latest of each type is processed. The content of the Event and Schedule files is filtered according to the current configuration of the Importer and is inserted into the planning model as scheduled tasks with constraints to modify the fill level profiles of the according resources.

- the "**ExecutionTimelineImport**": This plug-in is used to load a complete and already finalized timeline, consisting of so-called FOPs (Flight Operations Procedures) that have been generated in fully-automated planning runs, e.g. by the TerraSAR-X/TanDEM-X mission planning system [14]. Importing all this information into PINTA allows for a visual crosscheck for its consistency and validity supported by a stored set of constraints. Another possible use case was implemented for the FireBird mission, where this functionality is reused to ingest input XML's from the principle camera investigator that contain a FOP snippet to be used for so-called SystemOrders. For experimental acquisitions taken for these special requests the camera is then not commanded to switch to one of the standard acquisition modes, but instead it is configured with a FOP containing the ingested snippet.

- the "**Exporter**": See separate chapter IV.

Furthermore, for the various mission-specific planning systems, there exist numerous other corresponding plug-ins with very special functionalities. Only the ones for the FireBird mission shall be mentioned as examples, without going too much into detail: the "**PlanningRequestImporter**", the "**DatatakeInfoFileGenerator**" and the "**EnvelopeRequestImporter**" [13]. The plug-in functionality with a dedicated API allows for easily adding similar additional functionalities, depending on respective project's use cases.

For the further use cases of PINTA, when not serving as part of the core mission planning system of a spacecraft mission, the plug-in mechanism has also proven to be very appropriate to add problem-specific features. For the SoEEditor, for instance, these were the so-called "**ReferenceTasksHandler**", "**DragDropTaskFormsPlugin**", "**HinderDatabaseImporter**" [15] (see also chapter III).

D. The PLATO Algorithms

As already mentioned in the chapter before, PINTA allows for invoking PLATO algorithms. In general, all the fully automated as well as PINTA-based semi-automated mission planning systems at GSOC make use of these scheduling algorithms to automatically schedule and re-schedule indicator tasks, datatakes, downlinks, etc., depending on the mission-specific uses case, in order to prepare a conflict-free timeline for the chosen planning horizon. Furthermore, TimelineEntries of the FOP Tasks can be planned accordingly for the specified commanding timeframe.

These scheduling algorithms however are not implemented from-scratch but are composed from the generic algorithm assembly available in our PLATO library. The core idea of this set of algorithms, filters, etc. is to have a reusable, configurable suite of functionalities that can be variously combined and configured and operate on a planning project available in the GSOC modeling language.

A more detailed insight into this approach and an overview of most of the available algorithms and filters can be found in [16].

Regarding typical planning problems at GSOC, just some examples for basic algorithms shall be given here and outlined only very shortly to understand the principle:

- **"ChooseValuesToConsider"**: The time range to be forwarded to a sub-algorithm can be determined via various criteria, e.g. respective to the execution time of scheduled horizon Tasks.
- **"ObjectSelection"**: Various filters can be applied to determine to which sub-algorithms which of the currently considered Group(s) and/or Task(s) are to be forwarded in which order.
- **"ValueSelection"**: When having found the next Task to schedule, various filters can be applied to determine whether and with which execution time it is allowed to be scheduled. The invocation of sub-algorithms with time ranges derived from the new TimelineEntry's execution time is possible.
- **"ConstraintIgnorer"**: This allows temporarily deactivating constraints during the execution of a sub-algorithm, which can be necessary to handle circular dependencies. It enables scheduling and/or un-scheduling Tasks with a potential conflict first before trying to repair the solution by scheduling and/or un-scheduling other Tasks.

Examples for the application of such PLATO algorithms can be found for example in [13].

E. Manual Operations when Performing the Planning Runs

As already mentioned, typical mission planning systems with PINTA and PLATO are set up as semi-automated and thus interactive mission planning tools. The daily tasks left to a mission planning operations engineer can be among others the following:

- performing and monitoring the planning runs manually as specified in the according GOP (ground operations procedure)
- execution of recommendations for off-nominal operations and/or additional procedures for special planning tasks for additional payloads, if applicable
- user help-desk for inquiries from the "outside world", e.g. about the reason why a request could not be planned

In summary, there might be several manual steps to be performed in addition to the capabilities of the software components. In a more complex mission planning system, most of these are included into a fully automated software component or component assembly. However, depending on the mission type it can be decided that the described interactive approach fits best the given requirements for reliability combined with flexibility, and not to forget: cost efficiency.

F. TimOnWeb – Providing the Timeline Online via the Web

TimOnWeb is a graphical display tool to visualize the mission timeline and provide insight into an excerpt of the current state of the planning model to the users and other parts of the ground segment, e.g. the Flight Directors, via a website accessible from the WWW.

Within the use case of PINTA and TimOnWeb serving as components of a semi-automated mission planning system for instance, after every planning run, the latest resulting timeline can be broadcasted via this interface. Then, all requested and the actually planned and commanded Planning Requests with their parameters, the ground station contacts, sun- and shadow phases and the fill levels of various resources, like the onboard memory partitions can be displayed for the user community, amongst other information. Figure 1 provides a snapshot of the current TimOnWeb view for TET, one of the two spacecrafts of the FireBird mission.

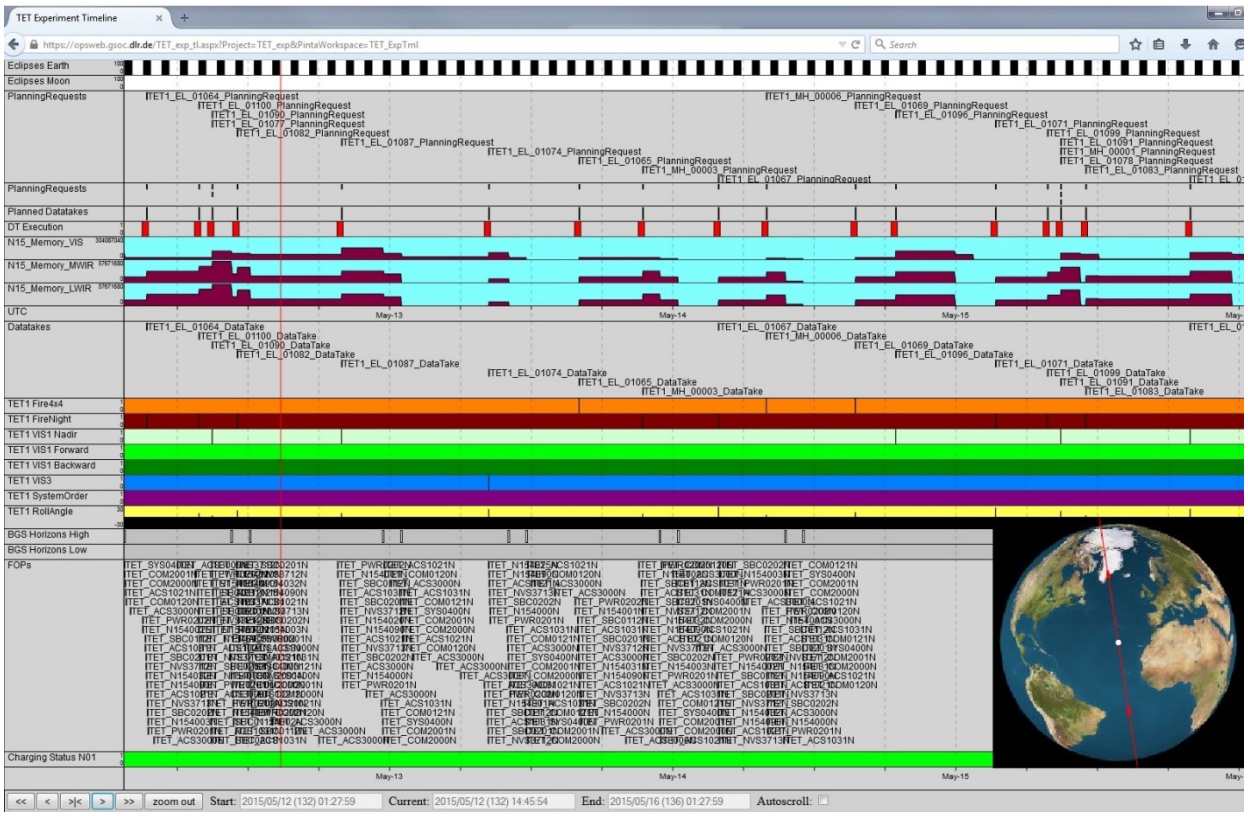


Figure 5: *Snapshot of TimOnWeb, showing an excerpt of the TET Timeline.*

Compared to PINTA, TimOnWeb does not implement its own planning model representation; the generic server part is based on the PLATO library, enabling TimOnWeb the usage of the extensive functionality of PLATO, such as conflict tracing, filter algorithms etc. As TimOnWeb is using JSON files to transfer the data to be displayed from the server to the client, the tool is not restricted to only share the planning model representation, but can also display information mapped from other sources.

Furthermore, the TimOnWeb client, implements state-of-the-art Web technologies like HTML5 and WebGL and relies on open-source JavaScript libraries, enabling the application to make use of wide-spread, verified generic functionalities, such as jQuery, jQueryUI, Moment and satellite-js. The latter one for example is used to display ground station elevation plots and ground-track plots, as shown in the lower right corner of Figure 5.

More detailed information about the technical implementation of TimOnWeb can be found in [15].

III. SoEEditor - PINTA as Sequence of Events Tool

Several desired usability improvements were determined in past LEOPs and new requirements were defined accordingly. This led to the decision to investigate the possibility to generate new SoEs with the PINTA and TimOnWeb tool suite. To support the special operations requirements of LEOPs, Commissioning Phases and pre-launch simulations of new spacecraft missions, the following top-level tasks to be served have been identified and can be provided:

- Possibility to (re-)import Flight/Ground Operation Procedures (FOPs/GOPs)
- Possibility to (re-)import orbit related events for spacecraft(s), e.g. ground station contacts or shadow transitions
- Possibility to (re-)import mission specific events without having to adapt the core application
- Easy generation of the SoE within the graphical timeline display

- Adding FOPs/GOPs and custom defined events via Drag&Drop onto the Timeline
- Moving/Deleting of non-orbit related events directly within timeline view
- Definition/Modification/Deletion of TimeDependencies between events directly within the timeline display

- The workload for updating the SoE after launch delays or other similar changes shall be as low as possible
- Possibility to display finished SoE on big screen environments in control rooms and on local screens

A. Editing Sequence of Events with PINTA

The first three requirements are covered by PINTA plug-ins, which convert the information from the source files to the PINTA planning model: The so-called "**ProcedureImporter**" plug-in reads the procedure files and creates Tasks needed for scheduling the procedures in the SoE. The orbit events can be imported by the already mentioned generic "**GSOCFileImporter**" plugin which is used in various other missions' operations for routine timeline generation. Furthermore, additional plug-ins have been developed with mission-specific functionalities.

One major challenge for the SoE generation with PINTA was the necessity to define time dependencies between procedures, orbit events and non-orbit related events. As the PINTA data structure had a main disadvantage modeling OrderedTimeDependencies. These are modeled between two Tasks, which in turn can have multiple TimelineEntries. This means that the TimeDependencies defined for the Tasks are effective for each TimelineEntry of the depending Task (see Figure 6). Thus, when e.g. one procedure is needed to be scheduled multiple times with TimeDependencies to different events each time, the existing modelling features did not support this with reasonable effort. To solve this issue the PINTA data structure was extended by a template/instance Task feature ensuring that the source (template) Task will be copied each time a new instance of it will be created, with the rule enforced that every instance Tasks must not have more than only one TimelineEntry at a time. The imported procedures are an example how this mechanism is used, see also Figure 6.

Additionally, the graphical timeline view had to be extended by the possibility to create these dependencies directly within the graphical display. To improve the usability during solving TimeDependency violations (e.g. after a reimport of orbit-related events or a launch delay), an algorithm was implemented to allow the user to reschedule all Tasks assigned to the conflicting dependency by only one click or shortcut.

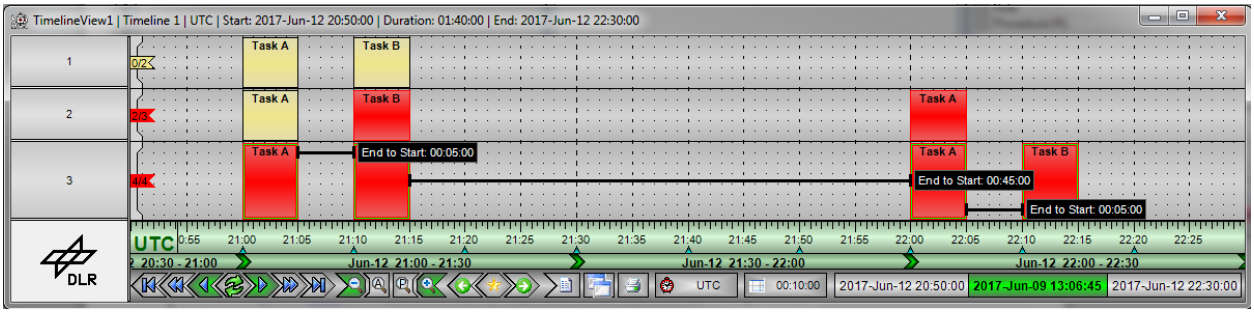


Figure 6: *The TimeDependency issue when a Task has multiple TimelineEntries is shown. For this example, two Tasks (Task A and Task B) were defined which have a TimeDependency, specifying that the start of Task B has to be exactly five minutes after the end of Task A. The first plot shows how the TimeDependencies are designed to work. In the next plot an additional TimelineEntry for Task A was added to the Timeline exactly one hour after the first TimelineEntry of Task A. This one immediately shows a conflict as the TimeDependency to Task B is violated (it starts before the start of the second TimelineEntry of Task A). The TimelineEntry of Task B also displays a conflict because the TimeDependency between both Tasks has to be valid for all TtimelineEntries of each Task. Trying to solve the conflict by adding a second TimelineEntry for Task B five minutes after the end of the second TimelineEntry of Task A leads to all TimelineEntries of both Tasks having a conflict as shown in the last plot.*

For the time being, the launches of BIROS and HAG-1 were the first ones at GSOC that were supported by the PINTA-based SoEEditor. These LEOPs have proven that the new SoE editing tool is suitable for low-earth orbiting as well as geostationary spacecraft missions. The next projects in the queue to make use of it will be Grace-FO with its specialty to launch two spacecraft at the same time, Eu:CROPIS, PAZ, EDRS-C.

B. Displaying Sequences of Events via TimOnWeb

Many missions require the possibility to display the generated mission timelines or SoEs in the control and support rooms within a control center but also on screens and user PCs outside of it. Therefore, a Web based application was developed which can read the project file generated by Pinta and display the content within a Web browser. On the one hand, the browser Web page can contain an alphanumeric table in which the events of the timeline are displayed in a HTML table, on the other hand, a graphical view can be configured in which the timeline are displayed as in the timeline view of Pinta, and also the combination of both display types is possible (see example in Figure 2).

As a special feature, with TimOnWeb it is possible to support simulation runs without the need to modify the execution times in the source files e.g. to train the execution of a LEOP SoE prior to a launch. Therefore, one or multiple time offsets can be specified on a separate Web page, which will be considered for displaying the live content without additional effort.

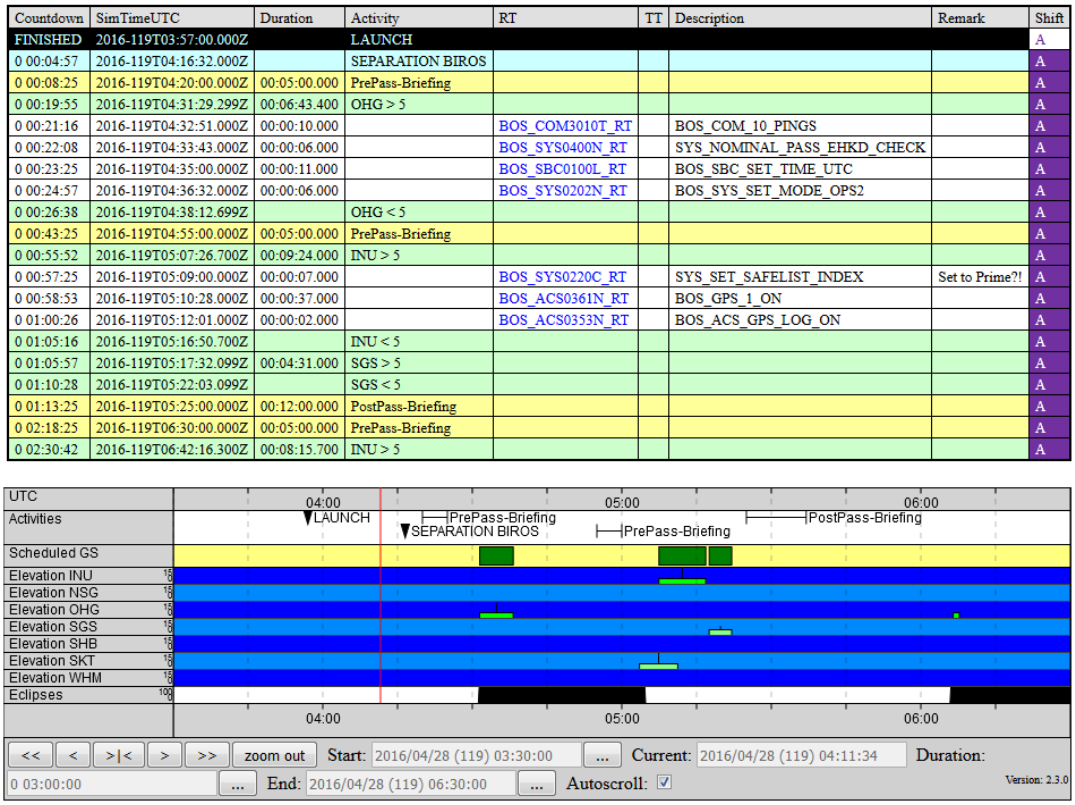


Figure 7: *View of TimOnWeb displaying the SoE for the LEOP of the Earth observation spacecraft BIROS. At the top, the alphanumeric table can be seen showing past, currently active and upcoming events. In this example, the table shows the activities of the first ground station passes after the separation of the spacecraft. The graphical part at the bottom presents other parts of the events on the mission timeline. In comparison to the table, it contains additional events like all possible ground station passes (not only the scheduled ones) and the solar eclipses in the last plot line.*

IV. Exporter & Scripting Language

A. Exporter

The continuously growing number of projects PINTA has to handle and the hereby involved requirements to support more and more different export formats, led to the decision to completely redesign its information export functionalities. The goal was to create a combined exporter functionality, which can handle all currently necessary file formats with the possibility to easily add more formats for future use cases. Furthermore, it should be possible

that every single output value can easily be configured within the GUI and can be filled with every possible value from the PINTA data model structure and even every combination of these values (see next sub-chapter “Scripting Language” for further explanations).

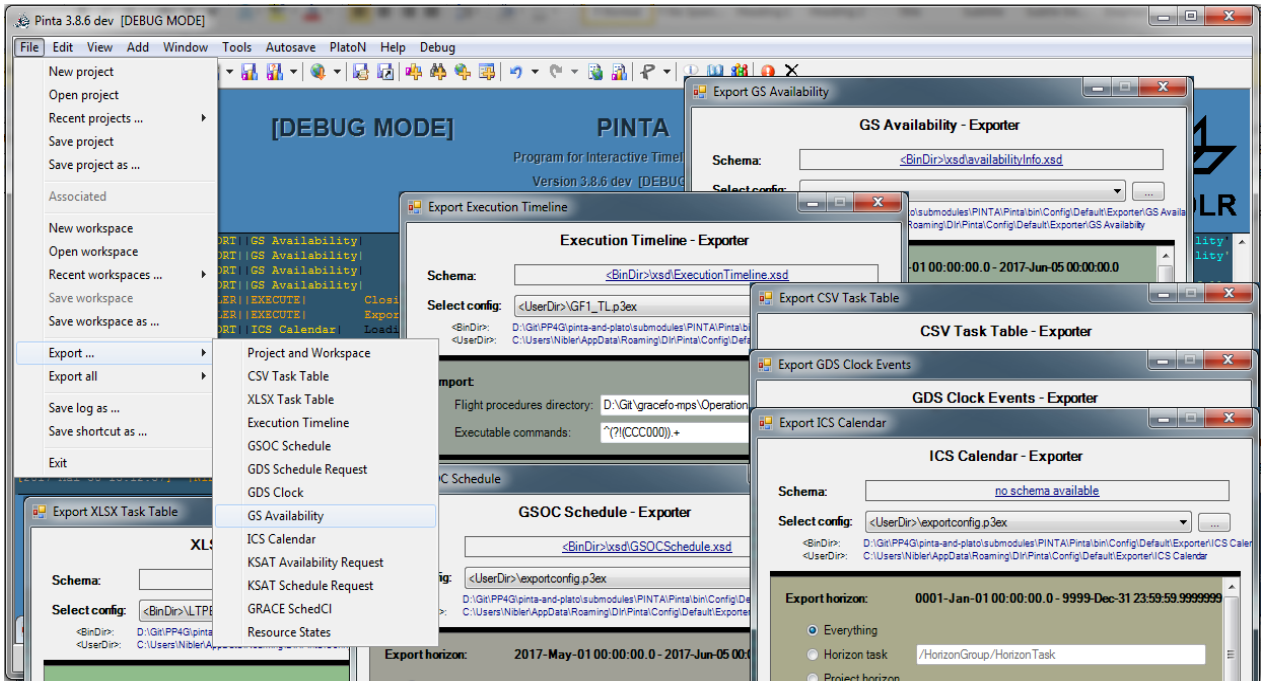


Figure 8: *An example of the main window of the current PINTA 3.8.6 version with several exporter configuration windows and the extended context menu “File -> Export …”. Every exporter has its own color encoding to help avoiding user operating errors.*

Before their configuration capabilities will be described in more detail, the most important exporters should be briefly explained:

- **“Project and Workspace”:** Contains the core functionality that is also available for all other exporters. It is able to export all basic PINTA files, the project file (*.p3pj), the workspace file (*.p3ws), the export configuration file (*.p3ex), the log files (*.log/*.rtf), the export summary file (*.txt) and even shortcut links (*.lnk).
- **“CSV Task Table”:** Creates a comma-separated values file (*.csv) in which all the exported columns can be freely configured.
- **“XLSX Task Table”:** Generates an Excel-file (*.xlsx) in which all the cells can be individually filled, and highlighted with different color and text styles.
- **“Execution Timeline”:** This is the main exporter for many missions. It generates a MOIS compatible file (*.xml) with the FOP sequence for every satellite.
- **“GSOC Schedule”:** The GSOC schedule file (*.xml) contains all ground station availabilities that were approved by the respective missions’ ground stations for an upcoming timeframe. This exporter is especially needed for simulating input for other sub-systems during preparation phases before actually negotiating with ground stations.
- **“GDS Schedule Request”:** Creates a request file (*.xml) for requesting ground station contacts from the responsible GDS (ground data system) sub-system at GSOC.
- **“GDS Clock”:** Generates the input file (*.xml) for the “GDS Blue Clock” which displays the scheduled passes in the control rooms at GSOC.

- **“ICS Calendar”:** Generates iCalendar files (*.ics) that can be directly imported in mail/calendar applications (e.g. MS Outlook).

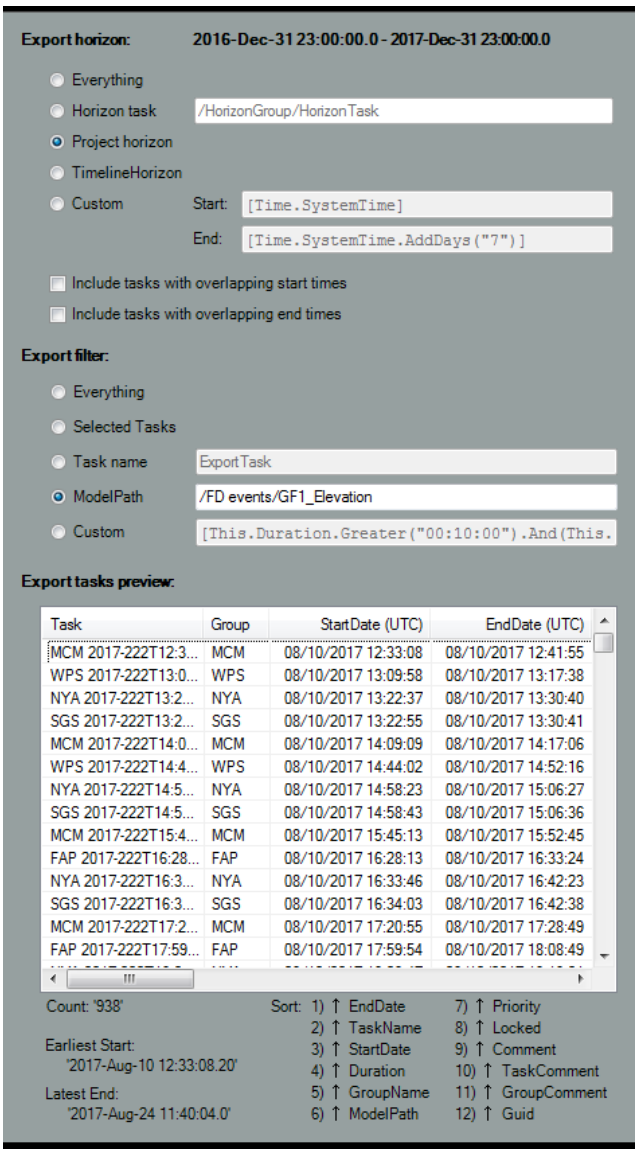


Figure 9: *Configuration of the export horizon, the export filter and the export sort order*

The majority of exporters within PINTA are used to export a subset of all the scheduled tasks, see Figure 9, which shows an example.

On the one hand, this selection can be temporally restricted. Therefore, it is possible to select “**Everything**”, i.e. the whole content of the current timeline, a “H**orizon Task**” whose timeline entry defines the start and end time, the “**Project Horizon**” and accordingly the “**Timeline Horizon**”, or the PINTA Scripting Language can be used to freely define a “**Custom**” export horizon.
Furthermore, it is possible to define the behavior whether to include or exclude tasks with overlapping timeframes.

On the other hand, filters can restrict the subset of tasks. Here it is possible to select “**Everything**” without restriction, the “**Selected tasks**” that were manually chosen via the multi-select capabilities in the timeline view, a RegEx filter applied to the “**Task name**” and accordingly the “**ModelPath**” (a unique link to the representation of a Group or Task within the planning model tree structure), or the PINTA Scripting Language can also be used to freely define a “**Custom**” export filter.

When defining the desired subset of all scheduled tasks with temporal and filter restrictions, the result is immediately displayed to the user in a preview list. Furthermore the sort order can be easily configured (ascending or descending) by selecting the corresponding task properties.

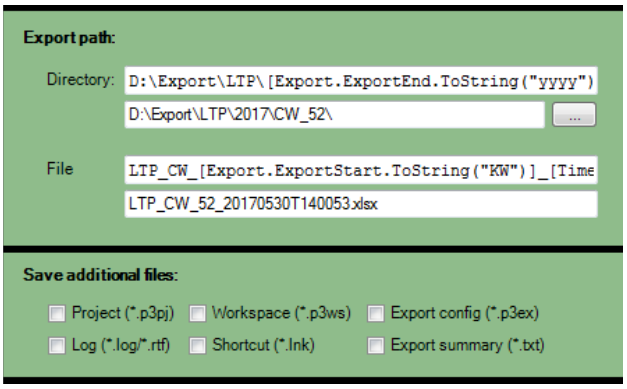


Figure 10: *Configuration of the dynamically generated export directory, the export filename and the export settings for additional basic files.*

To allow the creation of the exported files to be as automated as possible, the export directory and file name can also be pre-configured. When specifying these configurations it is again possible to apply the PINTA Scripting Language and freely combine project information as variables of the paths instead of solely having to rely on fixed strings.

Figure 10 demonstrates one of the common use cases. Thus, it is possible to dynamically create an export directory whose path contains selected values of the export timeframe and a file name that contains the current time in any configurable format.

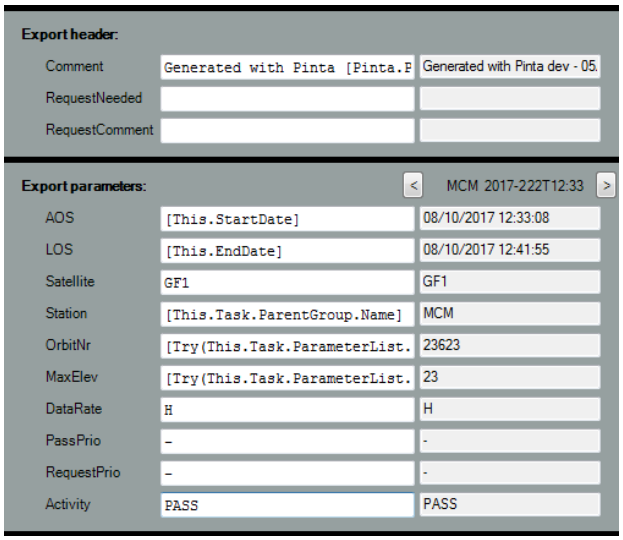


Figure 11: *Configuration of ground station request information, which is only available in the "***GDS Schedule Request***" exporter.*

In order to extend every exporter by format specific functionalities, additional panels and controls can be embedded in the GUI, if necessary.

Figure 11 for example shows configuration fields that are only available for the **"GDS Schedule Request"** exporter.
The XML schema file (*.xsd) for this specific export format (*.xml) requires a header with optional fields for informational purposes, like a comment.
Furthermore, for every request additional information is needed. Besides the start time ("AOS: Acquisition of signal") and the end time ("LOS: Loss of signal") further fields are configurable, like "Satellite", "Station", "Max Elevation", etc…
Here again, the PINTA Scripting Language can be used to specify the export values.

Other examples of exporter specific configurations can be seen in Figure 12 & 13.

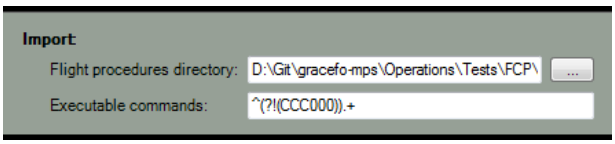


Figure 12: *Within the "***Execution Timeline***" exporter, a directory needs to be configured to import MOIS compatible templates for every used flight procedures. Furthermore, a Regex filter is used to distinguish executable from non-executable commands.*

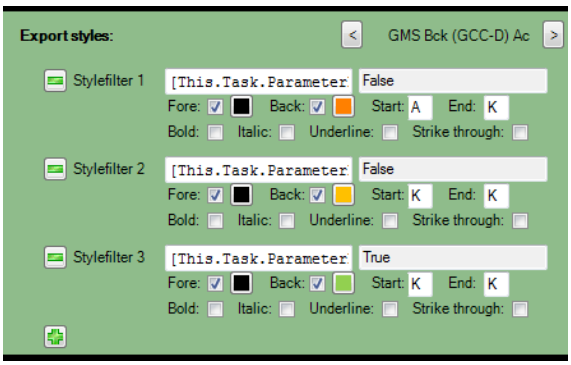
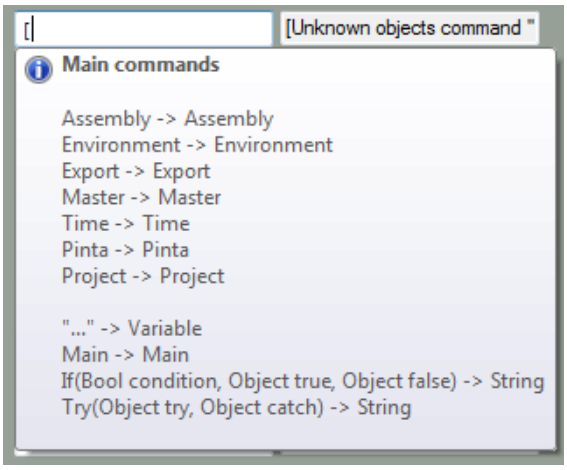


Figure 13: *Configuration of the color and text styles within the "***XLSX Task Table***" exporter.*

B. Scripting Language

During the redesign of the PINTA exporter, the idea to implement a scripting language came up to fulfill all the requirements for the different export formats. It is operating on the PINTA data model and has full read access to all the contained information. To make the usage as convenient as possible the syntax and the available commands are close to the .NET programming languages Visual Basic and C#.

To avoid learning and remembering commands the user is assisted by suggestions of intelligent code completion functionality during the configuration process (see examples in Figures 14 & 15).



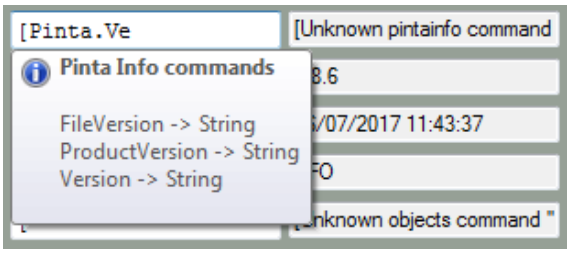


Figure 14 (left): *The "***Main commands***" tool tip window.*
Figure 15 (above): *The "***PintaInfo commands***" tool tip window. After every keystroke, the user gets a list of all possible commands that fit to the already entered input. The evaluated value of the command is displayed in the output textbox, as long as the command is valid; otherwise, an exception string is shown.*

The PINTA Scripting Language not only supports access via property access, it also allows conditional statements:

"`If(<condition>, <true-part>, <false-part>)`" and "`Try(<try-part>, <catch-part>)`"

These can also be used in nested and therefore more complex expressions. Figure 16 shows some easy examples.

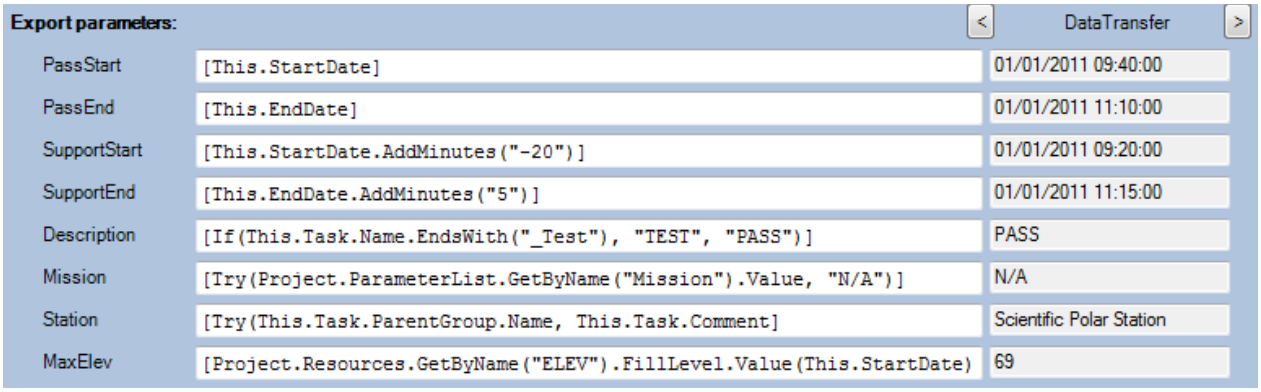


Figure 16: *Configuration example within the "***GDS Clock***" exporter. All the export parameters are defined with the PINTA Scripting Language and are evaluated for every TimelineEntry during the export process. For debugging purposes, the resulting values can already be previewed within the GUI. The arrow buttons then allow for scrolling through all the objects that were specified for the export (see Figure 9).*

V. Similar Applications, Conclusion and Outlook

Of course, the use cases and Mission Planning problems faced by PINTA and TimOnWeb do not uniquely exist at GSOC, but exist for spacecraft missions in general. Thus, a huge variety of similar applications has been developed at other agencies as well as by commercial providers, either as standalone display and/or planning and scheduling

tools or as Web-based solutions to support the cooperation and/or coordination of multiple users of certain spacecraft missions or space-related on-ground resources.

Among these are, for instance, the planning system for sharing the ground station capacities within the Deep Space Network Scheduling System with all its components (see e.g [1][2]), ASPEN-RSSC as applied for planning Rosetta experiment operations (presented in [3]) that again includes SPIKE and ASPEN, all developed at JPL, which can be seen a bit like pedants to PINTA and PLATO with their widespread application in automated and semi-automated planning systems for various missions, or SPIFe, an integrated planning and scheduling toolkit for exploration missions [4], which again, together with the so-called Ensemble architecture, serves as basis for the LADEE Activity Scheduling System (LASS) [5], all developed by NASA. For the ISS operations, including e.g. planning of the Columbus payload operations, first the Consolidated Planning System (CPS) software with the Onboard Shortterm Plan Viewer (OSTPV) as one of its user front-ends was in use ([6]), and nowadays as their modern substitution SCORE together with the OPTIMIS Viewer (see e.g. [11]) can be compared to PINTA and TimOnWeb regarding their functionalities.

Commercial tools covering a similar scope and relying on similar ideas for the presented use cases, and being provided especially for Earth Observation missions, but also for interplanetary missions, for instance are the Orbit Logic STKScheduler [7] or the GMV flexplan [8] applications, just to name a few, that help solving space-related planning problems with enabling user interaction with different extent of automated planning and scheduling support in the background.

No evaluation or critical comparison and discussion of these should be made here, as we would not want to and are not able to judge these in detail anyway, but overall, we see us as well positioned with our GSOC tool suite. During the long mission planning experience at GSOC PINTA and TimOnWeb proved to be flexible, reliable and generically applicable for solving various mission planning problems with a different degree of automation. The range of applications here is not restricted to serving single spacecraft missions, but also comprises solving multi-mission planning problems like the allocation of on-call shift personnel and supporting the sharing of ground resources such as control room and ground station availabilities. Furthermore, they can be easily combined with the other components and libraries of the GSOC mission planning tool suite, such as PLATO (see above), the event-calculation library SCOTA, and the so-called “Swath Preview and Ordering Tool” SPOT based on it [13]. Equally important is the well-established integration with the other GSOC sub-systems, such as flight dynamics, satellite and ground station monitoring and control via generic interfaces. In addition, the approach of having a comfortable in-house development is much appreciated and has proven to be beneficial as it easily allows for including direct feedback from operations personnel as well as other mission responsible persons and expertise for estimating probable future requirements.

While in the last years the focus of the ongoing evolution of PINTA and TimOnWeb lay on modernization, enhancing the robustness and extending the features, generality and use cases to be servable, currently enhancing the user friendliness is very important. This comprises simplifying workflows, improving the GUI functionalities, performance enhancements, provision of simplified and extended information export capabilities, and, last but not least, documentation. For instance, e.g. a convenient handbook and the provision of trainings and instruction material have to be promoted.

Nevertheless, in parallel, of course, there are always modifications for specific project and user needs wherever requested and viable. Having the control about the development of the software tool-suite it is ensured that further modifications, feature enhancements and functionality extensions will not be missed to stay well-prepared for and in the future as well. Above all, one big future goal is the integration of PINTA and TimOnWeb to one application becoming PintaOnWeb, i.e. enabling the ship [18] to sail into the open waters of the WWW with all its consequences and boundary conditions.

Among the big milestones for the next years, one is establishing the connectivity to the upcoming generic GSOC Reactive Planning framework (formerly known and presented as the “Incremental Planning System” [9]), making PINTA, or probably PintaOnWeb then, one of the optional front-ends of this, to allow for having interactive mission planning systems also basing on the same integrated, widely configurable and scalable tool suite as already foreseen for the fully automated use cases. Further, not less important, development tasks will be accompanying common developments of the GSOC modelling language as well as the international standardization of interfaces and mission planning and scheduling services in the CCSDS environment [10].

Besides, a steady renewal comprising ongoing “smaller” and “bigger” modifications and extensions and avoiding obsolescence will always be part of the lifecycle of PINTA and TimOnWeb, or, sometime in the future,

PintaOnWeb, trying to not only having to react to new requirements but trying to already foresee them beforehand as far as possible, or, as a popular saying tells, “stagnation is like a step backwards”.

References

[1] M.D. Johnston, M. Levesque, S. Malhorta, D. Tran, R. Verma and S. Zendejas, “NASA Deep Space Network: Automation Improvements in the Follow-the-Sun Era,” *24th International Joint Conference on Artificial Intelligence (IJAI)*, 2015.

[2] M.D. Johnston, D. Tran, B. Arroyo and C. Page, “Request-Driven Scheduling for NASA's Deep Space Network,“ *24th International Joint Conference on Artificial Intelligence (IJAI)*, 2015.

[3] S. Chien, G. Rabideau, D. Tran, M. Troesch, J. Doubleday, N. Federico, M. Costa Sitja, M. Perez Ayucar, C. Vallat, B. Geiger, N. Altobelli, M. Fernandez, F. Vallejo, R. Andreas and M. Kueppers, “Activity-Based Scheduling of Science Campaigns for the Rosetta Orbiter,“ *24th International Joint Conference on Artificial Intelligence (IJAI)*, 2015.

[4] A. Aghevli, A. Bencomo and M. McCurdy, “Scheduling and Planning Interface for Exploration (SPIFe),“ *21st International Conference on Automated Planning and Scheduling*, 2011.

[5] J.L. Breslina, "Activity Planning for a Lunar Orbital Mission,” *Twenty-Seventh Annual Conference on Innovative Applications of Artificial Intelligence (IAAI)*, 2015.

[6] T. Uhlig, F. Sellmaier and M. Schmidhuber, “Spacecraft Operations,” chap. 5.3.7 “Planning Tools”, Springer, 2014, ISBN: 97837091180302014.

[7] W.A. Fisher and E. Herz, “A Flexible Architecture for Creating Scheduling Algorithms as used in STK Scheduler,” *8th International Workshop for Planning and Scheduling in Space (IWPSS)*, 2013.

[8] A.T. Kavelaars, A. Barnoy, F. Colmenero, M. Pereda, J. Tejo, G. Garcia and T.W. Beech, “Evolution of a Flexible Mission Planning and Scheduling System for Complex Missions: flexplan,” *6th International Workshop on Planning and Scheduling for Space (IWPSS)*, 2009.

[9] M.T. Wörle, C. Lenzen, T. Göttfert, A. Spörl, B. Grishechkin, F. Mrowka and M. Wickler, “The Incremental Planning System - GSOC's Next Generation Mission Planning Framework,” *13th International Conference on Space Operations*, 2014.

[10] “Mission Planning and Scheduling Concept, Draft Green Book,” *Report Concerning Space Data System Standards*, 2016.

[11] A.C. Rich, “Flight Planning and Procedures,” *https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20160007117.pdf*, cited on 2017-06-07.

[12] A. Spörl, C. Lenzen, M.T. Wörle, J.H. Hartung, A. Braun and M. Wickler, “Mission Planning System for the TET-1 OnOrbitVerification Mission,” *13th International Conference on Space Operations*, 2014.

[13] M.T. Wörle, A. Spörl, J. H. Hartung, C. Lenzen and F. Mrowka, “The Mission Planning System for the Firebird Spacecraft Constellation,” *14th International Conference on Space Operations*, 2016.

[14] C. Lenzen, M.T. Wörle, F. Mrowka, M.P. Geyer and R. Klaehn, “Automated Scheduling for TerraSAR-X/TanDEM-X,” *7th International Workshop for Planning and Scheduling in Space*, 2011.

[15] J.H. Hartung, R. Nibler, C. Peat, A. Spörl, M.T. Wörle and C. Lenzen, “GSOC SoE-Editor 2.0 - A Generic Sequence of Events Tool,” *14th International Conference on Space Operations*, 2016.

[16] C. Lenzen, M.T. Wörle, F. Mrowka, A. Spörl and R. Klaehn, “The Algorithm Assembly Set of Plato,” *12th International Conference on Space Operations*, 2012.

[17] “GSOC Planning Modelling Language,” *http://www.dlr.de/rb/Portaldata/38/Resources/dokumente/GSOC_dokumente/RB-MIB/GSOC_Modelling_Language.pdf*, cited on 2017-06-07.

[18] Wikipedia article “Pinta (ship),” *https://en.wikipedia.org/wiki/Pinta_(ship).*