Ground Assisted Onboard Planning Autonomy with VAMOS

Maria Theresia Wörle, Christoph Lenzen
DLR / GSOC Oberpfaffenhofen D – 82234
maria.woerle@dlr.de, christoph.lenzen@dlr.de

Abstract

The typical ground based mission planning system for a low earth satellite mission has one major drawback: The reaction time to onboard detected events includes at least the two upcoming ground station contacts.
To address this disadvantage, DLR/GSOC implements the software experiment VAMOS as part of the FireBIRD mission, in which mission planning autonomy will be transferred to the spacecraft up to some extent. This paper presents the outcome of the VAMOS design phase – a concept of minimized onboard complexity which allows onboard reaction to telemetry measurements and event detection. In order to minimize risks and the computational effort onboard a solution has been chosen that demands relatively simple tasks of the onboard autonomy but nevertheless will lead to maximizing the mission output and still takes care of all potentially to be considered resource constraints.

Introduction

VAMOS (Verification of Autonomous MissionPlanning Onboard a Spacecraft) is an experiment that is prepared at DLR/GSOC for performing scheduling and (re-) commanding tasks onboard the satellite BIROS which will be part of the FireBIRD mission.
VAMOS also consists of an on-ground component embedded in the FireBIRD mission planning system that will also be prepared and operated by/at GSOC.

This experiment extends previous research work of DLR/GSOC colleagues (see [Axmann, Wickler, 2006] or [Axmann, 2010]) who developed an approach for BIRD to react to the results of cloud detection and image compression algorithms by extending the timeline with additional acquisitions. But for some reasons these times no integration into the (already orbiting) satellite could take place and thus this functionality finally couldn't be tested in space at all.
So now, the aim is to really verify at last the applicability, usage and benefit of a mission planning component that is planning, scheduling and commanding in an automated way onboard a spacecraft. It will combine the computation power of on-ground hardware that enables complex calculation operations and resource propagations with the reaction times that are in general only available for a system directly embedded in the soft- and hardware onboard.

Motivation for onboard-autonomous mission planning activities

In general the mission planning process for a low earth orbiting (LEO) satellite is performed on-ground in a control center, where fix timelines are generated that contain the commands to be performed by the spacecraft in the timeframe between the next and one of the succeeding uplink sessions. The idea of onboard planning is to delegate a part of the complex mission planning process to the respective satellite. The algorithms of the on-ground scheduling engines can be very complex and sophisticated, but they cannot foresee what events will occur during the execution time of these timelines. This might require additional actions to be performed in near-real-time (NRT). For instance the detection of fire, volcano-eruptions, ships or the reduced usability of an image fully covered by clouds could be responded by triggering another acquisition over the same target. Cloud covered images and acquisitions of fire-monitoring campaigns in which no fire could be detected could be discarded immediately onboard which would result in free memory that could be used for additional acquisitions of lower priority that originally had remained unplanned therefore.
Sometimes the ground control system cannot even predict the exact state of all onboard resources after the execution of a scheduled task, such as the current heat conditions

onboard in case the cooling system isn't working deterministically, the exact gain of the solar panels that could be saved by an elderly battery or the fill-level of the onboard memory which might vary according to the content of acquired data and therefore its size after compression. Thus the on-ground scheduler will have to use worst case estimations for the consumption resp. resource availability values and so many datatakes will stay unplanned without actual needs. After finally performing its scheduled tasks, the spacecraft might still have resources left that could be exploited, but the ground system first gets to know about this when evaluating the telemetry dumps some time later after the next downlink contact.

In these situations it would be helpful to allow the satellite to autonomously introduce new commands. The challenge however is to assure that these additional commands fit into the existing timeline and that they do not violate any constraints.

However some severe obstacles have to be faced for an onboard mission planning system in comparison to a system running on-ground:

Obstacles to onboard autonomous scheduling in general

Onboard autonomy is often seen as an additional risk for the spacecraft health rather than a powerful feature to enhance the return of its mission. Mostly a fully deterministic and predictable spacecraft behavior is preferred. Furthermore the design of any onboard software component has to cope with limited computation resources, such as memory and processing power, and the fact that the reaction time in case of problems (including their detection at first) might be extended in comparison to on-ground systems that can be permanently monitored and perhaps directly fixed by human interaction on short notice.

Therefore some restrictions have to be complied with when designing an "autonomous onboard mission planning software" such as limitations in the calculation operations that can be performed, limitations of the commands that may be commanded and/or even generated spacecraft-internally and of course a thorough testing before and after integrating the software onboard.

Two example use cases

In the following two example use cases will be presented that should be feasible even with these restrictions.

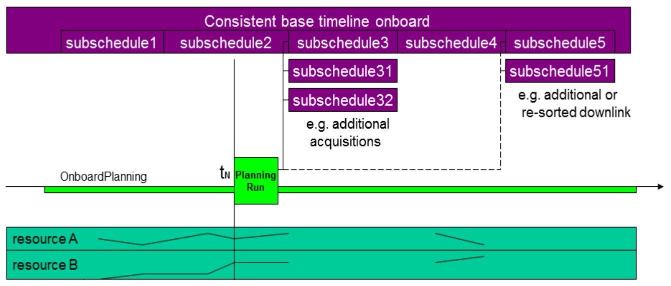

Figure 1

Figure 1 shows a use case in which a consistent base timeline is commanded to the spacecraft and currently executed there. For several subschedules of it there exist additional alternatives that have also been commanded but are not activated yet. The alternative subschedules 31/32 might for instance contain the content of the according original subschedule 3 and the commands for additional acquisitions. At the bottom, snippets of resource fill level profiles are sketched. E.g. resource A could show the power availability onboard, which is constantly filled by the gain of the solar panels and reduced whenever the instrument for data acquisition is used. Resource B might show the development of the memory usage, which increases during storage of new data and stays constant in the time in-between as long as no downlink takes place.

The onboard mission planning software then would check the fill level of the selected resources at a pre-defined point in time near the end of the execution of subschedule 2, and compare it with given bounds resp. decision values that lead to the choice of one of the alternatives.

Note that there are two profile snippets drawn during the execution time of subschedule 4. These shall remind of the impact on the two resources that are to be expected in the future, i.e. after the execution of subschedule 3/31/32, and have to be held available when the execution of the fix-commanded subschedule 4 is ongoing. This means that whatever the consumptions of the actions performed by the additional/alternative subschedules might be, they are not allowed to exceed time-specific limits. Therefore, when making the decision whether not only the current resource availabilities have to be considered but also the effects of the to be executed subschedules and the resource modifications already scheduled for the "future" in the base timeline, as just described.

Different considerations might be needed and evaluated in case this future may be changed, i.e. the base timeline may be modified, too, e.g. by removing some resource consuming task from it. Also the existence of different priorities for each acquisition or other to be performed task can be resp. might have to be applied therefore.

Another application of the subschedule selection is indicated with subschedules 5/51 that could contain different downlink schedules in terms of commanding of additional transmission times or another sequence of the to be downlinked data. This can be useful either in combination to the selection of other/additional acquisitions or might be useful in case the resource check at some time shows that an overflow of one memory part or the overwriting of not yet transmitted data is impending. Then this could be preferred in the next downlink in case it is likely to be transmitted in another, later ground station contact otherwise.

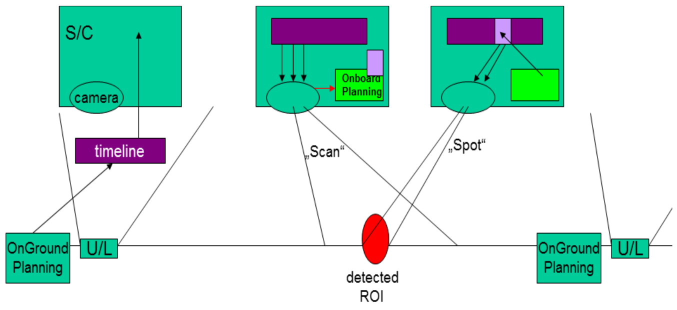

Figure 2

Figure 2 shows another possibility how an autonomous onboard mission planning software could be applied. The nominal ground-generated base timeline sent to the spacecraft during an uplink session commands the instrument to acquire several wide-view pictures in a scan-mode for example. In the gap between the ground station contacts (and thus on-ground mission planning runs that could change the commanded timeline customarily) some classification software that examines the acquired data directly onboard might detect a fire hot-spot or another interesting event and would trigger the onboard planning software that could, in a near-real-time decision, create an additional command sequence and insert it into the performed timeline in order to for instance make the instrument immediately looking at the detected target area again and acquire a kind of spot-mode picture with higher resolution. If not having the possibility to react so fast by the onboard actuators e.g. for changing the view direction by changing the attitude, also the use case of taking another picture during the next upcoming visibility of this region of interest can be imagined instead (depending on the target visibility in (one of) the next orbit(s) of course).
Transferred to the idea of handling the downlink, here the trigger due to the detection of some event in an acquisition could lead to the re-sorting of the downlink sequence in order to transmit the data of the fire event as fast as possible to the ground for example, provided that the onboard data-storage and –downlink management would be capable of such a re-sorting or the explicit announcing of data packages in the downlink commands.

The FireBIRD mission

FireBIRD is a scientific mission that is dedicated especially to the detection and monitoring of high temperature events (HTE) all over the world. It will be a constellation consisting of the two satellites TET-1 and BIROS operated by DLR.
TET-1 (abbreviation for the German expression “Technologieerprobungsträger 1”, i.e. a carrier for proving new technologies) was successfully launched on July 22nd 2012 and currently serves the testing of industrial and scientific experimental payloads and spacecraft technologies in the On-Orbit Verification (OOV) program of DLR. Beginning with its second year of operations, a camera system as one of these payloads will become the main payload on TET-1, which then will belong to the FireBIRD mission.
BIROS (Berlin InfraRed Optical System) which is planned to be launched in 2014 in contrast is mainly dedicated to this constellation from the beginning, even though it will also carry a number of additional experiments.
Both spacecraft are similarly constructed based on the bus of BIRD and carry a camera system consisting of a bi-spectral infrared hot spot recognition sensor system together with a three-channel optical sensor as multi-functional camera. The camera system is developed by the DLR institute for optical information systems located in Berlin and will provide a highly improved resolution in comparison to other currently orbiting fire monitoring systems. In addition to their HTE detection and monitoring function the two spacecraft will as well be used for other scientific earth observation applications.
For details see [Ruecker et al.].

Mission-specific challenges for the onboard planning experiment: Environment for the onboard planning

On BIROS, the mission planning software will be integrated into the payload processing unit (PPU), which means that it cannot be updated but through a PPU software upload. The PPU will be providing a RODOS operating system (see [RODOS links]), which assures real-time execution of its processing cycle, provided the software does not exceed its calculation budget.
This means that both – code complexity and calculation complexity – should be restricted to a minimum. Furthermore all awaited stages of extension, as far as possible, should be prepared, compiled and integrated into the spacecraft best before launch (though deactivated then at the beginning of the mission and switched on step by step).

VAMOS

In order to achieve minimum complexity for the onboard software, VAMOS is split up into three components, the on-ground add-on to the mission planning system, the onboard component OBoTiS, which restricts to activating pre-calculated timeline alternatives and the onboard component OBETTE, which adds new timeline alternatives and their activation criteria.

OBoTiS (OnBoard Timeline Selection)

When OBoTiS is activated, the onboard planner will check at predefined times, whether certain telemetry parameters stay within pre-calculated values. If all these telemetry checks are passed, the respective timeline extension will be activated.

This part of the onboard software remains extremely simple, still allowing the exploitation of the satellite resources to their maximum. However on ground multiple scenarios together with their conditions that determine when activation may be performed have to be prepared, taking into account not only the current resource states given by telemetry values but also the needs of later datatakes of higher priority, i.e. those that will still have to be acquirable later on. The automated creation of these timeline extension scenarios and their conditions, will be described later on in section “On-Ground Add-On”.

The following pictures shall illustrate the OBoTiS workflow:

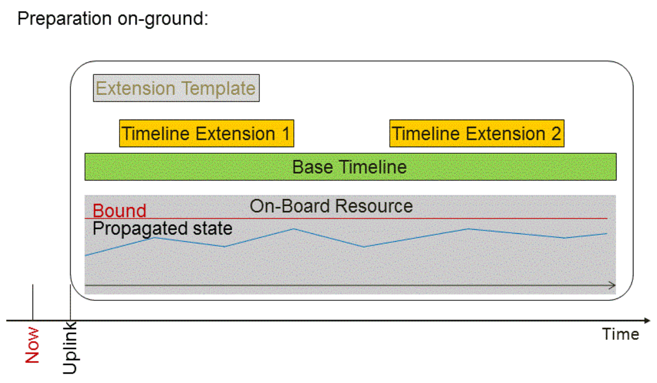

Figure 3

Figure 3 shows the uplink of the on-ground pre-calculated base timeline (green) and two timeline extensions (yellow), which may be activated in case certain telemetry conditions are met.

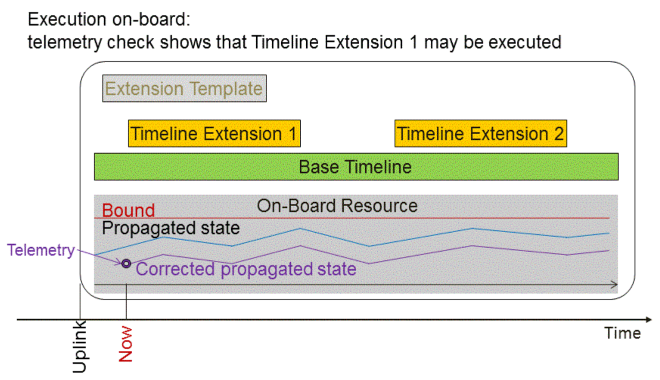

Figure 4

Figure 4 shows the time of decision whether to activate timeline extension 1 or not to activate it. Here, the telemetry value is far below the propagated value (maybe an unusable image file has been deleted), therefore timeline extension 1 may be activated:

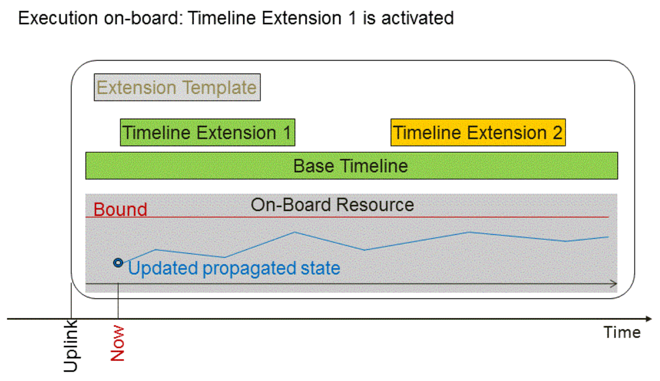

Figure 5

OBETTE (On Board Event Triggered Timeline Extension)

When OBETTE is activated, the onboard planner will listen to event information generated by other spacecraft components which indicate that an additional image should be taken.

OBETTE derives certain parameters (e.g. the required execution time and looking angle) from this event. A predefined command template will be copied, filled with these parameters and added to OBoTiS as new timeline extension, together with the corresponding set of telemetry conditions that are also derived from the event parameters.

Of course also more than a list of templates with different pre-defined settings or parameters derived from the configuration might be used from which the algorithm then first chooses one according to the event type or event parameters.

Details of this mechanism will be described in section “OBETTE on-ground add-on”.

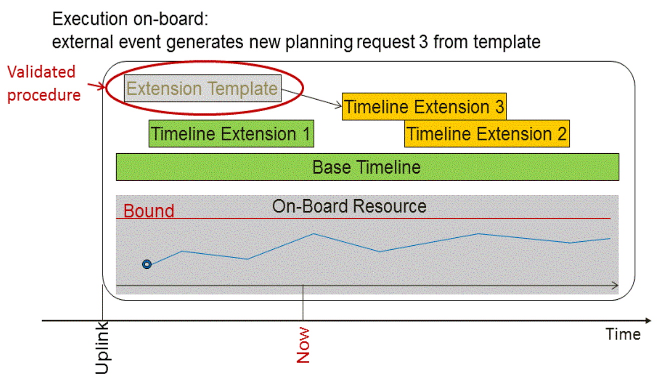

Figure 6

Figure 6 shows the evaluation of an event: timeline extension 3 is generated from the template and the event's parameters.
Thereafter OBoTiS is in charge to activate or discard this timeline extension:

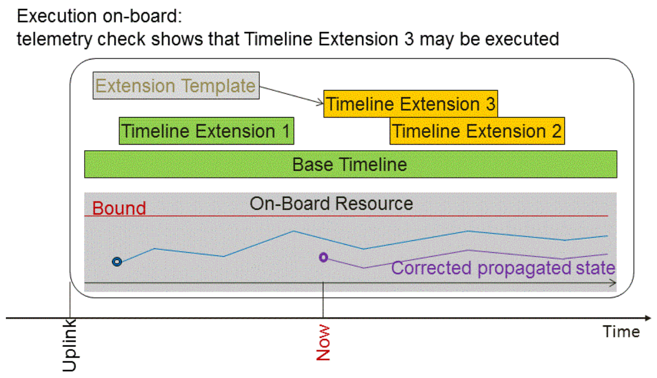

Figure 7

Figure 7 shows the decision time when OBoTiS has to decide whether to activate or discard timeline extension 3. In this case the telemetry values show that activation is possible:

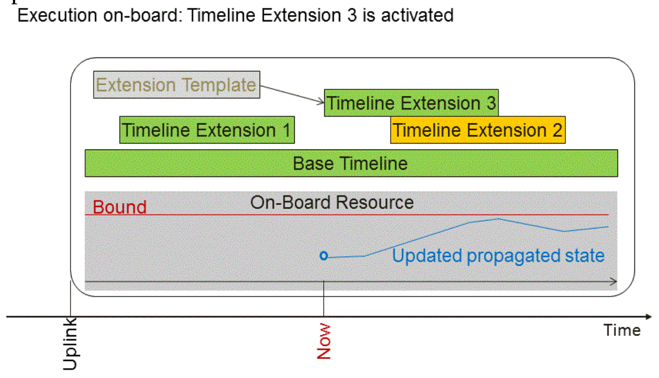

Figure 8

But of course it will also happen that a telemetry check fails and thus a timeline extension is rejected. And in addition, overlapping timeline extensions must not be activated in parallel, so in the depicted example, timeline extension 2 must be rejected:

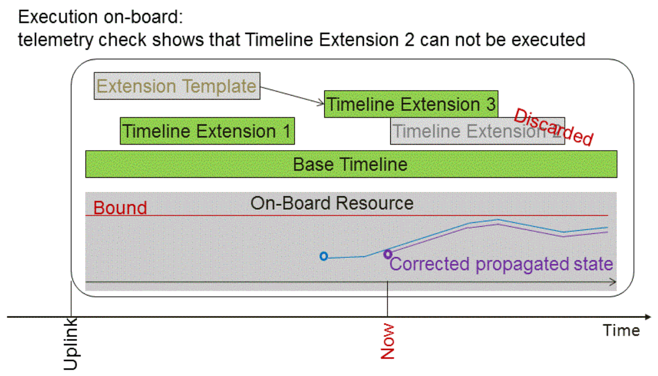

Figure 9

Priorities

Within the OBoTiS functionality, the onboard telemetry check compares on-ground calculated thresholds against the real-time telemetry as measured onboard the satellite. Since such a threshold is specified individually for each timeline extension, it will reflect the priority of the contained datatake: The base timeline used to determine the telemetry check for timeline extension 1 will already contain all timeline extensions of higher priority, which especially also includes all future timeline extensions of higher priority. This way, the higher the priority of a timeline extension, the less timeline extensions are part of the base timeline when calculating the thresholds for this timeline extension. This means that the thresholds will be more relaxed for timeline extensions containing high-priority datatakes.
For the on-ground scheduler this means to schedule all timeline extensions in the order of their priority, ignoring resource bounds, and before adding a timeline extension to the timeline, the maximum telemetry value at decision time (i.e. just before the timeline extension starts) is determined, which assures that the bound will not be exceeded in the future of this decision time.
For OBETTE, we define a constant priority. This way we only need to uplink one propagated state, which corresponds to a base timeline including all timeline extensions of higher priority than the OBETTE-generated timeline extensions. In case we had multiple event triggers of different priority, this approach might be extended by uploading multiple propagated states, which reflect base timelines including timeline extensions of different priority levels. However, then OBETTE-generated timeline extensions of lower priority with an earlier decision time might block later OBETTE-generated timeline extensions of higher priority.
Of course, the overall priority concept might be somehow disturbed by OBETTE, too: In case a new high-priority

event occurs after the decision of OBoTiS in favor of a low priority timeline extension has been made, a later medium-priority ground-generated timeline extension may be blocked, even though it would have been executed if only the low- or only the additional high-priority timeline extension would have taken place before. However this is an inevitable drawback of allowing timeline extensions of different priorities to become activated when new events may generate timeline extensions of even higher priority.

Commanding Interface

The above described mechanisms depict the ideas behind the onboard scheduling features. However the BIROS spacecraft does not support the ingestion of timeline extensions into an existing base timeline. Therefore each timeline extension must form a separate, consistent timeline block, which may be activated individually depending on the corresponding telemetry and envelope checks. The whole timeline must be represented by such timeline blocks that are commanded to the spacecraft.

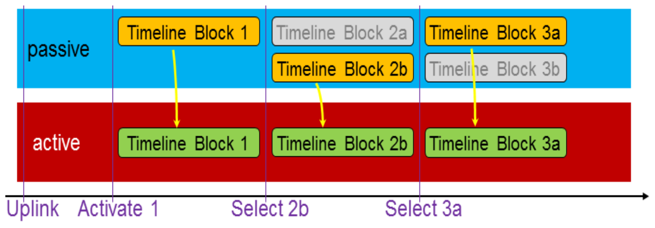

Figure 10

Figure 10 shows the ground-prepared, commanded Timeline Blocks 1, 2a, 2b,3a and 3b, of which Timeline Block 1, 2b and 3a are activated by OBoTiS.

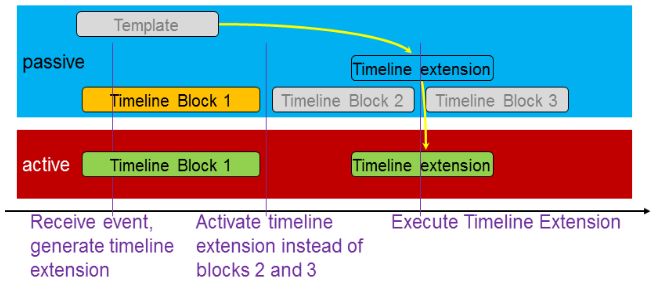
Template
passive
Timeline extension
Timeline Block 1
Timeline Block 2
Timeline Block 3
active
Timeline Block 1
Timeline extension
Receive event, generate timeline extension
Activate timeline extension instead of blocks 2 and 3
Execute Timeline Extension

Figure 11

Figure 11 depicts OBETTE: an event is received from which an additional timeline extension is derived. This timeline extension doesn't have to fit together with the existing timeline blocks. Thus its activation and the decision about this must be performed before the earliest overlapping timeline block starts in order to be able to decide to discard all overlapping timeline blocks.

On-Ground Add-On

So far, we explained the onboard mechanisms, which were designed to have minimum complexity. In this section we describe which support the on-ground mission planning system has to provide in order to achieve the goals of onboard autonomy.

OBoTiS on-ground add-on

As a first step, we restrict to OBoTiS, in which the onboard planner decides which timeline blocks to activate and which to discard. Therefore OBoTiS needs the following information for each timeline block:

- the timelineID of this timeline block
- the time interval of this timeline block
- a set of telemetry checks, each consisting of
  1. the time when to check this real-time telemetry
  2. the memory-address where to read the telemetry value
  3. the threshold which the telemetry value must not exceed

When deriving the timeline extensions from the planning requests, the on-ground scheduling process will consist of the following steps in order to supply the according conditions:

- Define the base timeline as an empty timeline.
- For each timeline block's planning request, in descending order of priority (highest priority will have the downlink sessions):
  1. In case there exists an overlapping not-discarded planning request of higher priority with start time later or equal, discard this planning request, because the onboard decision whether to activate the higher-priority datatake mustn't be blocked by the lower-priority planning request's execution.
  2. Define the decision time of the timeline extension as starttime – 1sec.
  3. For each resource, propagate the resource profile that results when adding this planning request to the timeline and derive the minimum remaining availability (including the consideration of the whole future and beginning with the planning request's timeline entry). For the critical resources, this value will become negative some time.
  4. The resource availability as calculated in the preceding step is added to the propagated value of the resource at the time when this check shall be performed. This is the maximum value the telemetry may reach when checking this condition and is stored as the threshold condition to the timeline extension.
  5. If there exists no planning request of higher priority overlapping with the current planning request, keep the modifications on the propagated resource profiles (in order to ensure that lower-priority

planning requests will preserve sufficient resources). Otherwise the resource modifications resulting of this planning request must be discarded before going on with the next planning request.

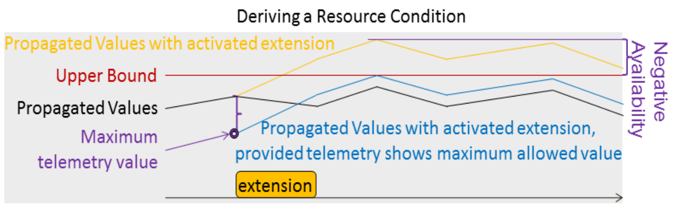
Deriving a Resource Condition
Propagated Values with activated extension
Upper Bound
Propagated Values
Maximum telemetry value
Propagated Values with activated extension, provided telemetry shows maximum allowed value
extension
Negative Availability

Figure 12

Figure 12 illustrates the calculation of a resource condition, which needs to be checked by OBoTiS before activating a timeline extension: first the propagated value including the timeline extension is calculated. The remaining availability is added to the propagated value at the decision time. In case the telemetry will show that the value is less or equal to this threshold, it can safely activate the timeline extension, since the corrected propagated values with activated extension will remain below the upper bound.
Note that this rule reflects all future resource modifications of timeline extensions of higher priority, so unless OBETTE adds a new timeline extension of even higher priority, the priority rules are strictly obeyed.
Furthermore, note that we do not have any problem at the beginning of a new timeline horizon, where we do not know the actual state of the resources - the absolute value of the propagated state at decision time is completely irrelevant: adding a value x to the state at decision time leads to an availability reduced by x, which means that the threshold, which is calculated as *propagated value at decision time* + *availability* remains constant. So whatever state the resource will actually have when the new scheduling horizon begins, the thresholds calculated by this rule are correctThe past is completely reflected in the telemetry check: It does not matter how we have reached the observed telemetry values, all that matters is that we activate the timeline extension only when there is enough margin for the future.

OBETTE on-ground add-on

In order to support event-triggered timeline extension generation onboard the satellite, the ground planner cannot perform the above illustrated calculation itself. Instead it must supply remaining availability profiles to the onboard planner, which indicate for each point in time, how much availability is left at this point in time, taking into account the whole future. This remaining availability must be calculated on basis of a timeline, which includes all ground-prepared timeline extensions of higher priority than the one the onboard-generated timeline extensions would get. This way, sufficient margins are preserved only for those timeline extensions which have higher priority than the OBETTE-generated timeline extensions:

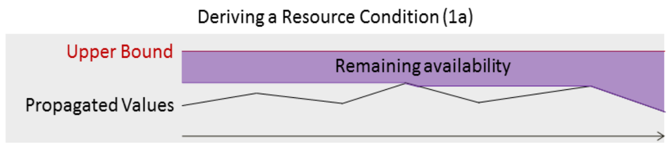

Figure 13

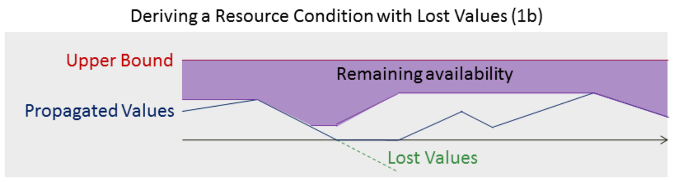

Figure 14

In addition to the availability profile, the onboard planner must be given the on-ground propagated values profile. When generating a new timeline extension derived from an onboard event, the onboard planner will derive the resource consumption of this timeline extension from the event parameters or use a fix, configured value. The resource condition for this timeline extension now can be calculated as follows:

1. *Decision time*
   = 1 sec before the decision time of the first overlapping timeline extension of lower priority or the new timeline extension's execution time, whatever comes first. This allows the new timeline extension to block overlapping timeline extensions.
2. In case an overlapping timeline extension of higher priority with decision time later than this decision time exists, the whole timeline extension is to be discarded. (Otherwise the high-priority timeline extension might be blocked.)
3. *Remaining availability including this extension*
   = remaining availability profile at decision time - resource consumption of this timeline extension. This represents the remaining availability as it would have been propagated on-ground. Note that this value may be negative.
4. *Telemetry threshold*
   = Propagated value at *Decision time*
   + *Remaining availability including this extension*

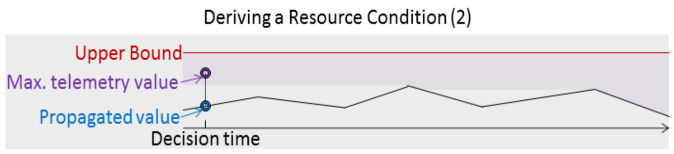

Figure 15

The *telemetry threshold* indicates the maximum telemetry value which will be allowed to be observed at *decision time*.

Please note that although we have to transfer profiles to the satellite, we do not have to perform any profile operations other than evaluating the profile at a given time. The onboard software therefore remains simple and fast. However storing a resource profile may consume some memory. In case this turns out to become a bottleneck, the on-ground planning system may be adapted to simplify the propagated values profile and the remaining availability profile. If this is necessary, it has to be ensured that the simplified profiles have lower values than the original profiles.

Note that, similar to the OBoTiS case, we do not have a problem in case the executed timeline before decision time differs from the timeline used for propagation, because the calculated threshold only depends on the expected future resource consumptions. The past is completely reflected in the telemetry check.

Testing and activation of extensions

In general it is planned to test the features of each stage of extension of the onboard software in simulation mode before arming it. During these times it will only log what it had measured and what it would have done resp. how it would have decided. The log will be checked on ground, compared with the other spacecraft telemetry and the result will be tested against the ground mission planning system, i.e. it will be checked whether this would have reacted the same way.

Afterwards, if proven reliable and ready for use, the respective code will be configured for "real" execution during dedicated campaign time frames. All in all a similar approach as performed for the TAFF (autonomous formation flight keeping system of the TanDEM-X mission), see [Ardaens et al.]. If then the functionalities are thoroughly tested and presented to run smoothly, and when the main satellite user groups agree on the profit it will have for the whole mission it can be transferred into a nominal operational use.

Currently Planned VAMOS Use Cases on BIROS

In the first increment the onboard mission planning shall be able to activate the execution of additional or alternative datatakes. Perhaps it might also modify the downlink sequence of acquired data by (re-)configuring the sequence of memory partitions to be transmitted. Furthermore it shall react to triggers coming from the main classificator as well as of the experimental image analyzer. If one of them classifies an image as useless, e.g. due to cloud coverage, VAMOS can be used to delete the according data package, and then use the freed memory for a new acquisition which it will pick from the ground-calculated timeline extensions (see OBoTiS). On the other hand, if they have detected a fire event and provide the onboard planner with the next visibility of this region, a completely new acquisition will be generated and commanded "spacecraft-internally" (see OBETTE). This is part of the second increment.

Outlook to the future

We are convinced that the future will bring more and more the need for (partially) autonomous systems like the one described above. Not only for missions that operate far away from the earth so that the long distance imposes a reaction delay and which therefore have to react autonomously to detected events, but also for low-earth-orbiting spacecraft that don't have a continuous ground station contact, since a tight net of ground stations as well as communicating via one or more relay satellites are assumed to be too expensive, especially for scientific missions. Especially the reaction delay is not negligible and solvable by a purely ground-based system, since for the delay at least the time spans from detection to downlink, processing on-ground and commanding the actions back to the spacecraft have to be counted. So real NRT-reactions need the autonomy of the spacecraft up to some extent, not only in terms of navigation and motion (e.g. for formation keeping, as well as for approximation operations and docking maneuvers), but also in terms of scheduling and tasking its payload instruments and other onboard actuators. The same applies for optimized resource exploitation, see the use cases described in the introduction (optical missions, clouds, not predictable memory usage after compression, …).

Next steps will be an operationalization of such features, adapted to the respective mission's goals. With the experience gathered by the experiment VAMOS, for upcoming missions also non-experimental planning and scheduling solutions that contain onboard components can be offered and developed.

And once the onboard support is required by a mission, a big advantage will be that the surrounding onboard software might be adapted or provided accordingly. As stated in the beginning, VAMOS restricts to the rare given available memory and computation power for its onboard calculations. However further research is ongoing in order to move the more complex profile propagation to the spacecraft, e.g. by using a dedicated FPGA. This would allow more sophisticated features such as to support the strict priority concept with multiple priorities for OBETTE or even a continuous optimization of the onboard timeline based on the increasing accuracy of a recalculated resource profile propagation.

Furthermore the cooperation possibilities resp. onboard communication/commanding possibilities to various actuators as well as triggers that inform about events could be extended as soon as autonomy is no longer seen as an additional risk for the satellite health rather than a big benefit. With such features of autonomy, the value return of scientific as well as commercial missions can be further enhanced in comparison to fully ground-controlled systems.

Conclusion and assessment

With the paper on hand we wanted to present a new approach to distribute mission planning tasks to the autonomy of a spacecraft without a loss of complexity of the found decisions and finally performed plans for acquisitions and other payload operations. It combines the NRT-capabilities and accessibility to up-to-date, exact current telemetry values with the processing power, calculation and propagation capabilities of the on-ground mission planning system to a relatively simple, low-risk application that matches the pre-conditions given on the hosting spacecraft.

Previous research works, developments and ideas for onboard mission planning from outside DLR suggest, for instance, an onboard greedy search with a limited number of constraints ([Khatib et al.]), probability estimations as basis for the onboard decisions ([Gough et al.]), or using the local search algorithms, similarly onboard as for the on-ground scheduling when (re-)planning lists of acquisitions together with a (partly downstream) checking of the resource developments for the future (as with CASPER as part of ASE on EO-1, see e.g. [Rabideau et al., 2006 & 2009]) are suggested.

In comparison to these, for VAMOS a different approach was invented to cope with the given practical specifications and restrictions: A distributed system, in which as well the base timeline with potential timeline extensions is commanded via timeline blocks as the embedded onboard planner itself newly create such command blocks from templates and pre-calculated profiles, and then these consistent timeline parts are activated or discarded for execution after “only” checking a list of several pre-calculated criteria, but nevertheless with the goal to achieve a similar optimized exploitation of all to be considered onboard resources without exceeding any limits.

In general, it must be clear that such a system requires (depending on the complexity) even more development and pre-launch testing efforts than purely ground-based systems. But, apart of relay-satellite-solutions or a really dense ground station network that would also require a complex mission planning system for rapidly (re-)tasking the spacecraft, (and that have to be considered separately in terms of costs), real NRT-reactions can only be achieved with (at least partly) autonomous onboard automated mission planning features, not only for “far-distance” but also for LEO missions.

For VAMOS, we are looking forward to integrate and run this software into the BIROS spacecraft’s onboard processing units and the according on-ground mission planning system. It will verify that with such a solution the overall generated output of a mission that combines the challenges of Earth observation with those of “Earth watching” (as introduced by [Damiani et al.]) can be enhanced and a NRT-reaction to detected events with a complex computation background is possible. In the future then even more sophisticated onboard solutions might follow.

References

Ardaens, J.S., D’Amico, S., Fischer, D.: *Early Flight Results from the TanDEM-X Autonomous Formation Flying System*; 4th Spacecraft Formation Flying Missions & Technologies Conference, St. Hubert, Canada, 2011.

Axmann, R. (2010): *Interactive acquisition scheduling for low earth orbiting satellites*; München, Verlag Dr. Hut, ISBN: 978-3-86853-571-6.

Axmann, R., Wickler, M.: *Development and Verification of an Autonomous Onboard Mission Planning System - an Example from the BIRD Satellite*; SpaceOps 2006, AIAA 2006-5851.

Damiani, S., Verfaillie, G., Charmeau, M.-C.: *An Earth Watching satellite Constellation: How to Manage a Team of Watching Agents with Limited Communications*; Autonomous Agents & Multiagent Systems/International Conference on Autonomous Agents - AAMAS 2005, ACM 1-59593-094-9, pp. 455-462.

Gough, J., Fox, M., Long, D.: *Plan Execution under Resource Consumption Uncertainty;* In: *Proceedings of the Workshop on Connecting Planning Theory with Practice at 13th International Conference on Automated Planning and Scheduling (ICAPS'04).*

Khatib, L., Frank, J., Smith, D., Morris, R., Dungan, J.: *Interleaved Observation Execution and Rescheduling on Earth Observing Systems;* In: *Proceedings of the ICAPS-03 Workshop on Plan Execution*. Trento, Italy. June 2003.

Rabideau, G., Chien, S., McLaren, D.: *Tractable Goal Selection with Oversubscribed Resources;* International Workshop on Planning and Scheduling for Space (IWPSS 2009). Pasadena, CA, 2009.

Rabideau, G. et al.: *Mission Operations of Earth Observing-1 with Onboard Autonomy;* Second IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT 2006). Pasadena, CA, 2006.

RODOS links, as on March 11, 2013:
http://www.dlr.de/irs/en/desktopdefault.aspx/tabid-5976/9736_read-19576/,
http://www.dlr.de/sc/en/desktopdefault.aspx/tabid-5276/8847_read-15871/.

Ruecker, G., Lorenz, E., Hoffmann, A., Oertel, D., Tiemann, J., Halle, W.: *Upcoming and prospective fire monitoring missions based on the heritage of the BIRD (bi-spectral infrared detection) satellite;* IGARSS 2012, in: *Geoscience and Remote Sensing Symposium (IGARSS), 2012 IEEE International.*