18^{th} International Conference on Space Operations, Montreal, Canada, 26 - 30 May 2025.

SpaceOps-2025, ID #145

Toucans – Next generation ground station scheduling at GSOC

Sebastian Wiesner[a*], **Thomas Fruth**[b], **Maria Wörle**[b], **Falk Mrowka**[b], **Jake Ashdown**[e]

[a]*Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR), German Space Operations Center, Weßling, Germany, sebastian.wiesner@dlr.de.*
[b]*Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR), German Space Operations Center, Weßling, Germany*
[b]*Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR), German Space Operations Center, Weßling, Germany*
[b]*Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR), German Space Operations Center, Weßling, Germany*
[e]*Havens-Above, Munich, Germany*
[*]Corresponding Author

Abstract

Historically control room and ground station scheduling consisted of many scattered, mission-specific approaches, ranging from old isolated desktop tools even to e-mails between mission and station personnel, resulting in a considerable amount of friction and an inferior scheduling experience to operators. To address these issues, GSOC develops Toucans (Tool for Unified Control Room, Antenna and Link Scheduling), a new system for integrated scheduling of control room operators, station passes, and optical links, across all missions operated at GSOC.

Despite its early stage, Toucans is already used in missions. We discuss how Toucans fits in the existing scheduling landscape at GSOC, what workflows it provides to operators, and the challenges we faced so far in this project. Eventually, we hope to consolidate various scheduling requirements and solutions, both for ground stations and optical links, in a single tool which delivers fully automated ground scheduling with minimal order deadlines and maximal routine time windows, a convenient and simple user interface for operators, and standard interfaces for automated use. In this paper, we discuss the core aspects of ground station scheduling Toucans eventually needs to address, and share our vision for a fully automated scheduling system. To achieve this goal, Toucans builds on our proven Reactive Planning framework together with the Plains planning library and our interactive PintaOnWeb user interface, as well as using the well-designed CCSDS service management standard as our core data model. Furthermore, a notable part of the paper discusses how we face the criticality of ground station scheduling by following a rigorous development process with strict quality assurance.

Keywords: Ground stations; Optical links; CCSDS; Scheduling; Planning

Acronyms/Abbreviations

Acronym/Abbreviation	Meaning
DLR	Deutsches Zentrum für Luft- und Raumfahrt e.V. (*German Aerospace Center*)
GSOC	German Space Operations Center
MPS	Mission Planning System
CCSDS	Consultative Committee for Space Data Systems
Toucans	Tool for unified control room, antenna, and link scheduling
SCOTA	SpaceCraft Orbit and groundTrack Analysis tool
MuMiCoRoS	Multi-Mission Control Room and pass Scheduler
GSSNG	Ground Station Scheduling Next Generation
OCI	Open Container Initiative
ADR	Architectural Decision Record
AoS	Acquisition of Signal
LoS	Loss of Signal
LEOP	Launch and Early Orbit Phase
QA	Quality Assurance

1. Introduction

After successfully employing our modern automated mission planning framework Reactive Planning (f.k.a. "Incremental Planning system" [1]) or TDP-1 [2] we expanded its scope beyond space missions, towards ground station scheduling at the *German Space Operations Center* (GSOC), which is part of the German Aerospace Center (DLR).

Historically, control room and station scheduling consisted of many scattered, ad-hoc, mission-specific approaches, causing a considerable amount of friction and headaches to everyone involved. Some progress towards consolidating our scheduling landscape has been made, initially with the routine scheduling tool MuMiCoRoS [3, chapter VI], which schedules conflict-free routine contacts for all missions operated at GSOC, and more recently with GSSNG [4], which consolidated station and configuration management, automates pass request handling, and provides basic scheduling to avoid conflicts between passes. However, both systems still require manual operations to some degree, and do not provide a holistic end-to-end scheduling workflow from missions all the way to various ground stations and back again.

We aim to fill this gap by applying our operationally proven and tested, fully automated mission planning framework, and our experience from operating multiple missions on top of this framework, to the problem of scheduling station contacts for missions GSOC operates. We also believe that our framework and our experience provides a good foundation to implement and evolve GSOC's future link planning concept [5], to address the growing need for optical communication from a scheduling perspective. By combining these aspects into a single tool we aim to offer a comprehensive scheduling system to GSOC and external customers, to schedule space to ground communication consistently across radio as well as optical links. The result of these considerations is Toucans, a new system and interactive tool for unified control room, antenna, and link scheduling.

In the following sections, we provide and overview over this new system, its current functionality, its architecture and our development processes. We outline the challenges we face and how our architecture and development processes help us to solve the challenges. Eventually, we conclude with our future vision for Toucans and lessons we learnt while development and deploying our new system.

2. Problem Statement

When designing and implementing a new system in an existing software landscape, it helps to take a step back and define the core problem the system is supposed to solve. Toucans, in its core, eventually aims to solve the following core problems of scheduling station contacts:

1. Scheduling of the GSOC-operated antennas at the Weilheim ground station [6], both, for missions GSOC operates, as well as for external customers who wish to use the station, incl. de-conflicting.

2. Ordering passes at stations operated by third-party providers, over a wide range of different, proprietary and provider-specific interfaces.

3. Scheduling long-term routine passes for all missions operated at GSOC, across antennas operated by GSOC itself (namely, those in Weilheim), antennas operated by other DLR institutes, and antennas operated by independent commercial vendors, incl. de-conflicting, with maximal predictability.

4. Scheduling special short-term passes for anomaly and contingency handling, with minimal order deadlines, across the same variable set of station providers.

5. Adapting the station schedule dynamically, to enable planning systems and missions operators to adjust the mission timeline dynamically in response to changes in demand for up- and downlinks, e. g. in case of increased downlink demand due to a high amount of orders.

Each of the above aspects comes with a few particular challenges:

1. To schedule antennas that GSOC operates for its own missions as well as for external customers, Toucans needs to meticulously manage antenna configuration, to precisely communicate the antenna configuration the customer needs to the automation and control system of the Weilheim ground station. Antenna configuration is a diverse field, not covered by any CCSDS standard yet, and not the immediate area of expertise for mission planning engineers.

2. To correctly order passes at stations operated by external providers, Toucans needs to support a wide range of heterogeneous and proprietary interfaces used by different providers, and account for differences in station behavior. For instance, some stations graciously handle changes in pass times as a result of changes in orbits, within acceptable limits, whereas other stations strictly adhere to the requested pass times.

3. To schedule routine passes for internal missions several weeks in advance, Toucans needs to adapt to different orbit characteristics across satellites. For instance, some missions have strict reference orbits which are maintained over a long time and make contact scheduling rather easy, whereas other missions have somewhat volatile orbits.

4. To schedule short-notice passes for internal missions, Toucans needs to provide an interactive user interface with which mission operators can quickly select available passes over supported stations and request passes with the required configuration. Furthermore, it needs to minimize the order deadline as much as possible, in order to help operators prepare necessary activities to handle anomalies and contingencies and contact their spacecraft as quickly as possible. After all, when mission operators are busy with difficult recovery activities, a station scheduling workflow should get out of their way, and not add additional overhead.

5. To adapt the station schedule dynamically in response to mission demand, Toucans needs to offer a feedback loops with guaranteed order deadlines, to enable missions to order or cancel station contacts dynamically while still maintaining a conflict-free timeline within the mission-specific commanding deadlines.

In this paper we will outline how Toucans plans to address these challenges.

3. Overview

In its current early state of development Toucans only addresses the last of the aspects we outlined in the previous section: It provides a graphical user interface (see figure 1) to view the current ground station schedule for all missions operated at GSOC, and lets operators select visibilities and submit short-notice schedule requests in case of anomalies and contingencies, as well as for occasional proficiency passes. For long-term rule- and priority-based scheduling of routine passes, GSOC still uses an operator-driven desktop application called MuMiCoRoS [3, chapter VI].

Toucans and MuMiCoRoS derive scheduling requests from the results of their scheduling algorithms and submit these to the GSSNG tool [4] which is responsible for scheduling and configuration management of the Weilheim ground station as well as for ordering passes at external station providers. For these purposes GSSNG assembles an overall schedule using input from MuMiCoRoS, Toucans, and various external systems, with some basic first-come first-serve conflict checking. Toucans as well as other mission-specific planning systems then receive this overall schedule from GSSNG.

Most systems still consume the schedule in GSOC's internal legacy format; Toucans and Weilheim however already use the simple schedule format standardized in CCSDS 902.1-B-1 [7]. Likewise, GSSNG accepts schedule requests in another GSOC-internal legacy format, but also consumes CCSDS 902.0-G-1 service package requests [8] over an experimental implementation of the CCSDS 902.9 Service Management Utilization Request Formats standard [9] which is currently in progress. Internally, Toucans and GSSNG represent all passes as CCSDS 902.0-G-1 service packages [8].

All scheduling systems also use input from GSOC's flight dynamics services to obtain orbit data and calculate ground station visibilities. Toucans uses GSOC's SCOTA service [10] for visibilities, which provides a modern GraphQL [11] interface to calculate station visibilities for various configurations. SCOTA in turn receives orbit information from GSOC-wide flight dynamics services over a JSON-based REST API. Figure 2 illustrates the interactions between these tools and systems.

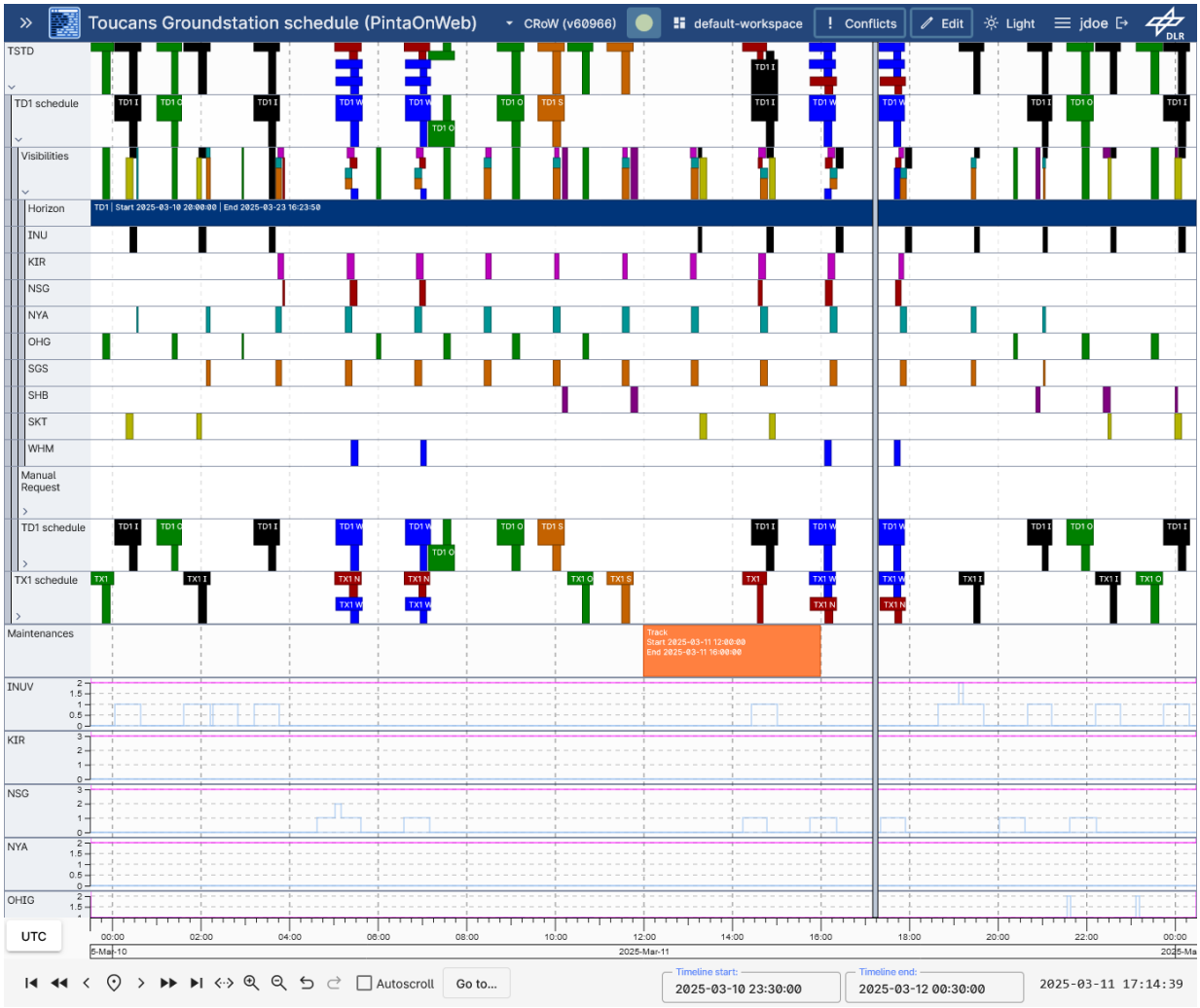
4. Workflows

After outlining how Toucans fits into the current landscape of ground station scheduling at GSOC, in this section we would like to briefly show how Toucans looks like from a user's perspective, and take a look at exemplary workflows Toucans offers to mission personnel.

4.1. Schedule Display

In its user interface Toucans first and foremost offers an interactive timeline view for the overall routine station schedule of all missions operated by GSOC, as well as a complete antenna schedule for the Weilheim ground station operated by GSOC. Figure 1 shows a screenshot of this view. Embedded in this schedule, Toucans also plots station visibility events of all missions operated by GSOC over all stations these missions may use, regardless whether these passes are already booked or not. In addition to the default layout Toucans offers a selection of alternative plot layouts to address different needs, such as a plot which renders sites and antennas instead of satellite missions.

Fig. 1. Toucans displaying schedule and visibilities for satellites and the allocation of antennas at ground stations



4.2. Graphical Schedule Requests

Mission personnel usually relies on the routine station schedule mentioned in the previous section during nominal operations. However, for activities not covered by this routine schedule, such as anomaly investigation, contingency handling, maintenance activities like software uploads, or even just occasional proficiency passes, Toucans provides mission personnel with a user interface to manually request additional passes from selected visibilities. Figure 3 shows this workflow in the UI:

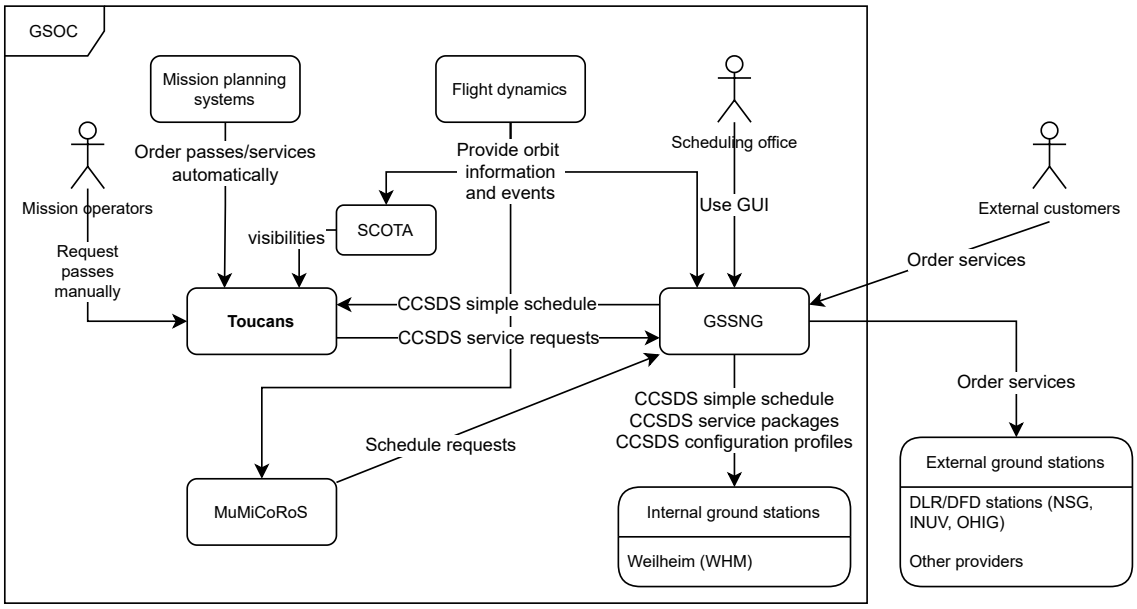
1. The user selects the desired visibility for their mission over the target station.

2. A dialog appears which lets the user configure the pass, in a mission-specific way: the user can select the kind of pass (e. g. S- or X-band), and the priority of the request.

3. The manual schedule request appears in the schedule timeline, and eventually gets confirmed by the updated schedule.

Toucans notifies users via email when their pass gets confirmed, if their pass is still not confirmed 24h after order and 24h prior to the desired AoS, or if a previously confirmed pass gets cancelled by a subsequent schedule update.

Fig. 2. Current status of Toucans in context



5. Architecture

Toucans builds upon GSOC's proven Reactive Planning framework (f.k.a. "Incremental Planning system" [1]) and its underlying Plains planning library [12] for automated and conflict-free scheduling, and uses PintaOnWeb [13] as interactive user interface to view and edit the schedule (see figure 4), and SCOTA [10] to calculate orbit events.

Plains provides a library to describe scheduling problems and implement scheduling algorithms, and defines the overall planning model format. Reactive Planning receives inputs from various sources, and applies scheduling algorithms to continuously update the planning model, in order to provide operators as well as other systems with an up-to-date schedule and with fast feedback to their schedule requests. On every update of the planning model Reactive Planning automatically stores the precise changes to the planning model in a database, to persist the current state of the planning model across restarts.

GSOC has successfully applied this architecture before in various missions and projects; refer to e. g. TDP-1 [2] or EnMAP [14, 15] for detailed discussions of this overall architecture. In this section, we would like to focus on two important aspects of our architecture which have not been discussed in the context of Reactive Planning previously.

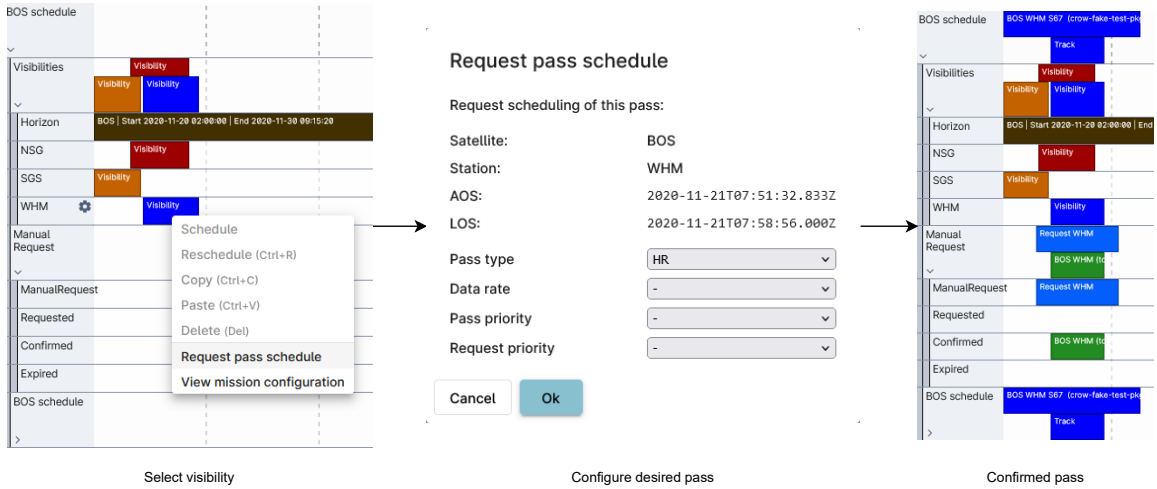
5.1. GraphQL for transparency and long-term compatibility

Toucans makes heavy use of GraphQL [11] for interfaces between internal services (such as SCOTA [10]) as well as for the public-facing API in PintaOnWeb. Compared to traditional REST APIs based on JSON or XML, GraphQL offers two distinct advantages:

- GraphQL APIs provide full introspection with first-class documentation in a machine-readable format. This enables full code generation for remote APIs, with type information and embedded documentation, and allows operators and developers to use standard off-the-shelf API clients to interact with our GraphQL APIs for prototyping integrations, as well as for ad-hoc data extraction, analysis, or automation. This makes our APIs fully transparent to other departments and systems within GSOC, and enables seamless integration and automation of Toucans.

- GraphQL schemas are fully typed and include deprecation annotations as first-class and machine readable feature. Together, this provides well-defined and testable semantics of API compatibility as well as automated tracking of deprecations. Toucans uses this for automated compatibility tests to catch incompatible changes in other services before these changes affect operational deployments. This in turn enables us to constantly evolve APIs used within GSOC, without breaking API clients, and with a clear path for deprecation of legacy APIs.

Fig. 3. Schedule request workflow

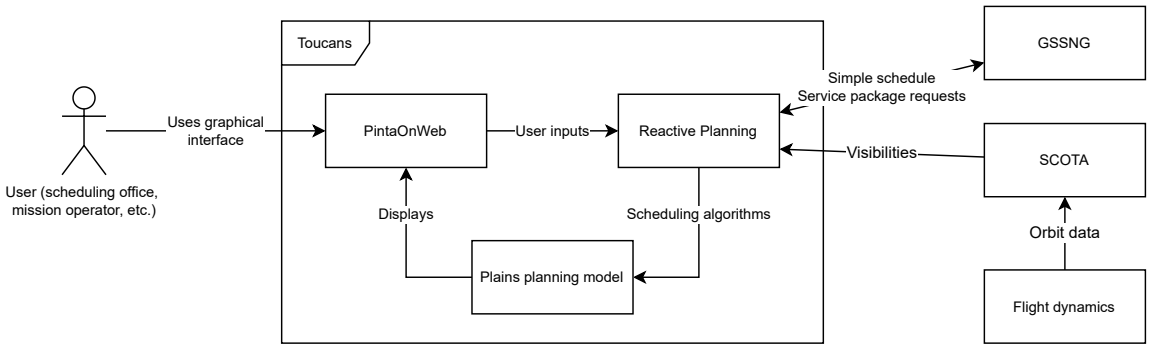


5.2. *Code over Configuration*

Software systems at GSOC often aimed at moving as many facts as possible into configuration, and strived to provide operators with maximum configurability and flexibility. This focus resulted from obsolete software release and deployment processes and regulations: historically, testing and deployment was often done manually, in time-consuming procedures, accompanied by lengthy bureaucratic processes to ensure that no essential manual step in the entire testing and deployment process was missed. As a result, deploying a new software release was often a rather large effort, spanning multiple days if not weeks, which often delayed roll out of important changes. On the other hand, operators are usually permitted to routinely change configuration as part of their procedures, or under supervision from development or system engineering personnel. In this environment, an extreme amount of configurability often helped to bypass lengthy deployment processes and quickly roll-out essential changes to operations personnel.

However, in this environment the actual operational configuration of any software system tends to slowly but inevitably drift away from the configuration used for development and testing: while new configuration features naturally get rolled out to operators in new software releases, configuration changes made by operators do not automatically propagate back to the development and testing teams. This can lead to the unfortunate situation that

Fig. 4. Toucans design



software is developed and tested with configuration and test data which is quite distinct and sometimes even entirely disconnected from what is used in operational deployments of the software. Software tests and quality assurance processes frequently only cover theoretical workflows and scenarios, whereas scenarios actually used in day-to-day operations are entirely untested and hence tend to break in new software versions. At the same time it becomes increasingly hard for development and quality assurance personnel to reconstruct operational configurations for development and testing purposes, which complicates root cause analysis of software errors and thus delays bug fixes. It also complicates integration across different software systems, because end-to-end integration tests only cover artificial test configurations and never actually validate the intended operational setup, leading to integration errors even after formally passing all required integration tests.

Recent mission planning software solutions at GSOC try to reverse the trend, and Toucans is no exception. It attempts to avoid configuration by operations wherever possible: Toucans does not let operators create sites or satellites in the system, or change the configuration for e. g. visibility events or pass requests. Instead, every change requires a code change. This code change is subject to our standard quality assurance processes for all code changes, which include code review by another developer, full coverage by automated unit and integration tests, mandatory documentation, and automated test and quality assurance pipelines. The next section describes these development processes in detail.

As a result, the software we develop, and which passes many automated tests with every code change, is quite literally the same software we provide to operators, which gives us a high degree of confidence in our software releases, and allows us to fearlessly deploy new versions from our code repository at any point in time.

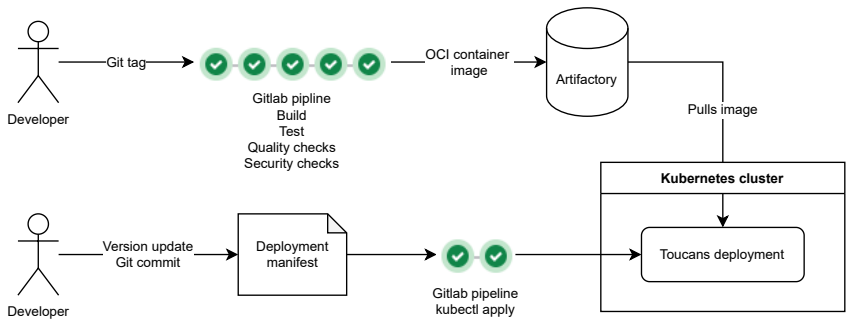
6. Development and Deployment

In the previous section we explained that Toucans requires code changes for every configuration change to minimize divergence between development, testing, and operational deployments. Practically, this can only work if Toucans' release and deployment processes are drastically shorter and simpler than conventional processes at GSOC. We achieve exactly this in two ways:

1. We make strict and rigid quality assurance part of every-day development, so as to maintain a main branch of development in a state which can always be released at any time without requiring any extra QA process.
2. We automate as much as possible: we not only require automated integration- and unit-tests for every change, we also maintain a fully automated release process.

To automate deployment we build on Gitlab[1], a wide-spread code hosting and automation platform, and Kubernetes[2], the leading and de-facto standard container orchestration system. Figure 5 illustrates the deployment workflow for Toucans:

Fig. 5. Toucans deployment



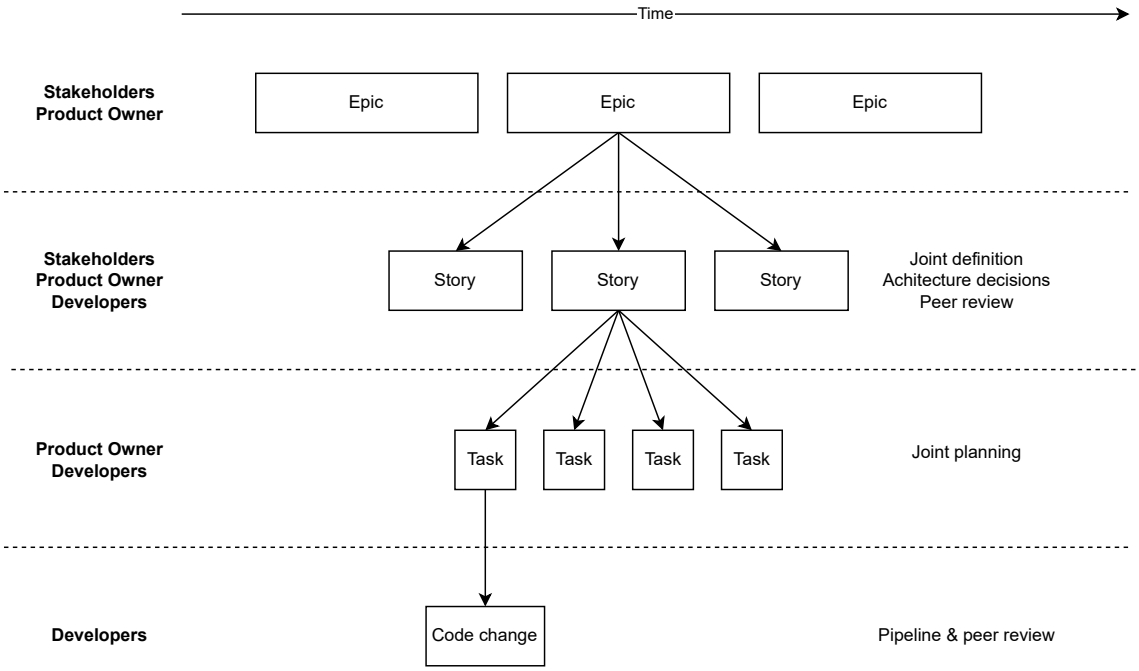
1. A developer pushes a Git tag; this triggers a Gitlab pipeline which runs automated unit and integration tests, security and code quality checks, and eventually pushes an OCI[3] container image to the GSOC artifactory[4].

[1] https://about.gitlab.com/
[2] https://kubernetes.io/
[3] Open Container Initiative, https://opencontainers.org/
[4] JFrog Artifactory, https://jfrog.com/artifactory/

2. The developer then pushes a Git commit to update the Kubernetes deployment manifest, which triggers another Gitlab pipeline to check the manifest for syntax errors and apply it to the Kubernetes cluster Toucans runs on. Kubernetes then automatically pulls the image, cleanly shuts down Toucans to persist the full state of the application, and then restarts Toucans in the new version. At startup Toucans recovers the complete state of the planning model as well as all ongoing processing from the database, and then seamlessly continues where it The Gitlab pipeline waits for this step to complete, and fails if Toucans fails to start in the new version; the developer gets notified and can immediately roll back to the previous version, using the same deployment process.

To maintain the main branch of the code repository in a state from which we can deploy at any time we make strict quality assurance an essential part of our development process. We use a simplified Kanban-like development process: a mid- to long-term roadmap gets broken up into dedicated epics which are then split into multiple user stories, which are then prepared for implementation by planning technical implementation tasks. Figure 6 illustrates this hierarchy. On each of these levels, peer review is an essential aspect.

Fig. 6. Planning hierarchy



1. On the lowest level, as part of our *definition of done*, it is required that every code change which implements a technical task is accompanied by automated tests covering the changes, and by corresponding updates to our code and architecture documentation. Gitlab pipelines then run all automated tests for every change, and must pass before a code change is eligible for being merged into the main branch. Any failing test will prevent the code change from making its way into the main branch, and by implication, into the operational deployment of Toucans. Additionally, every code change must undergo peer review: at least one other team member needs to review and approve the code change before it may be merged to the main branch. This maintains code quality and distributes knowledge about implementation aspects, technical design, and best-practices among all team members.

2. On the level of stories, the technical implementation plan for a user story is discussed and iterated upon by at least two developers, and frequently by the whole development team, to make sure that all team members know the story, and to increase the likelihood that all technical aspects of the implementation are duly considered.

3. On the level of epics, we again prepare stories in teams, and we require that all substantial additions or changes to the architecture of our software are proposed in structured form, iterated and refined in the development team, and then voted upon, to ensure that all architecture changes are documented and known to all development team members, and again, to increase the likelihood that all aspects of the architecture change are duly considered.

To propose architecture changes in structured form, we make use of Architectural Decision Records (ADR)[5] to document and discuss every significant change to our software architecture in a structured form. An ADR consists of a comprehensive description of the problem to solve and its overall context, a list of constraints or intentions driving the decision, one or more proposals to solve the problem, and finally, an explicitly documented and substantiated decision together with an analysis of potential outcomes of this decision. Like our process of writing code changes, the process of writing ADRs again heavily relies on peer review: we require that every ADR undergoes peer review by at least one other developer, and frequently even require a full review by all team members, including a consensus for the final decision. To convey architecture decisions we rely on diagrams a lot: we found that UML component, deployment, state, and sequence diagrams to be very effective in communicating architecture intentions and consequences.

All ADRs become part of our full architecture documentation, based on the helpful Arc42[6] template. As such, the process of writing ADRs not only provides us with high-quality architecture decisions, it also automatically provides us with documentation for all such decisions which gives us a chance to actually learn from past mistakes and change our architecture for the better, one ADR at a time.

7. Challenges

While developing Toucans we routinely face three major challenges inherent to ground station scheduling.

7.1. Interfaces and CCSDS

As described in sections 1 and 2, ground station scheduling at GSOC consists of many scattered and mission-specific approaches, which naturally implies the existence of many different interfaces. These interfaces often evolved historically along the requirements of specific missions. As a consequence, these interfaces tend to lack a consistent meaning and implementation across multiple projects, and are frequently augmented with mission-specific extensions or manual workflows. Some interfaces also evolved from interfaces originally intended for human operators, and are thus not a good fit for consumption by automated systems. Section 3.1 of [4] lists some examples of these legacy interfaces, such as the GSOC legacy XML schedule format. This XML consists of a sequence of *schedule_entry* elements like this (example from [4]):

```
<schedule_entry priority="-" station="INUV/INU"
  stop_time="2023-01-09T00:37:44.00" start_time="2023-01-09T00:02:20.00"
  support_id="TD1" activity="SPT" data_rate="-" is_pass="false">
</schedule_entry>
<schedule_entry max_elevation="34.7" priority="P" station="INU"
  stop_time="2023-01-09T00:32:44.00" start_time="2023-01-09T00:22:20.00"
  support_id="TD1" activity="PASS" data_rate="-" is_pass="true">
</schedule_entry>
```

On closer inspection, these two entries already illustrate quite a few issues with this interface and its data model:

1. An attribute can have vastly different meanings depending on the value of other attributes:

   (a) The *station* attribute sometimes denotes a site and sometimes a specific antenna on a site, depending on the value of the *is_pass* attribute.

   (b) The timestamps sometimes point to AoS/LoS events, and sometimes just internal station support times, depending on the value of the *is_pass* attribute.

2. The first entry denotes the internal station support time for the actual pass described by the second entry, yet while obviously related these two entries do not have any actual relation in the XML document. To consistently display this schedule pass and support entries we have to correlate both by their overlapping timestamps.

[5] https://adr.github.io/
[6] https://arc42.org/

Toucans already moved away from the GSOC legacy XML schedule format to the CCSDS 902.1-B-1 Simple Schedule [7] format which solves these issues with a clearly defined, consistent, and hierarchical data model: in a simple schedule every attribute has an unambiguous meaning, and all related entities have a hierarchical relationship. Likewise, Toucans internally uses CCSDS 902.0-G-1 service packages and service package requests [8] as its core data model, and also profits from the standardized definition of these entities: a "pass" has different meaning to different people or service providers, whereas service packages, service package requests, etc. are clearly defined and unambiguous. This greatly simplifies conversations about the meaning of our core data model.

However, CCSDS is not only a blessing, but poses a challenge on its own, as it does not yet cover some crucial areas of service management. In particular, Toucans would need a finalized publication of CCSDS 902.9 Service Management Utilization Request Formats [9] for a future-proof way to exchange information about service packages with other systems at GSOC and beyond. Other important areas like the APIs and workflows governing the specified interfaces, for instance to request service packages, are not even covered by draft standards yet.

As a consequence, we are effectively creating our own proprietary copy or variant of CCSDS 902.9 Service Management Utilization Request Formats which may not entirely fit a future final version of this standard, and thus may require adjustments at some point in time, across multiple systems at GSOC, to move our current implementation of this standard to a final version. In other places we effectively create simple substitute APIs to communicate service packages and service package requests, in want of any standard we could use. A speedier finalization of CCSDS 902.9 together with standardized workflows and APIs around it would help a lot.

7.2. Hidden Knowledge

While Toucans and GSSNG use service packages as their core data model, with clearly defined meanings, legacy systems have historically grown, proprietary, and often not well documented data models. As a result, we frequently find "hidden knowledge" in these systems which subsequently propagates into workflows. Uncovering and then retrospectively documenting this hidden knowledge to understand the behavior of legacy systems takes considerable resources, and carries a significant quality risk when interacting with legacy systems.

7.3. Criticality of Ground Station Scheduling

Finally, the criticality of ground station scheduling presents a significant challenge on its own. As ground stations stand are crucial for space operations, proper ground station scheduling is immensely critical to successful nominal space operations. Any changes at this point can have fatal effects to all missions operated at a control center: breaking the scheduling system of a single mission inhibits operations of a single mission, but breaking the ground station scheduling system would impact every single mission operated at GSOC and even missions operated by external customers using GSOC's ground station. However, we are confident to reduce this risk with our rigid development and quality assurance processes outlined in section 6.

8. Vision

In this paper we discussed how Toucans takes its place in scheduling of traditional ground station contacts. Now we would like to provide a quick look at the road ahead.

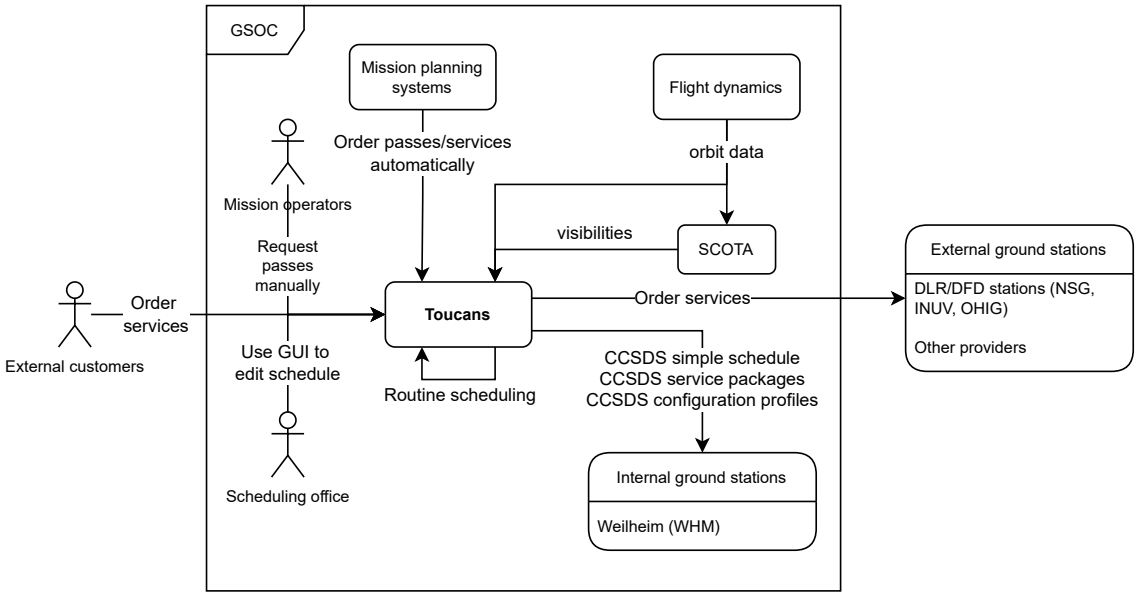
8.1. Fully Automated Scheduling

Our long-term plan is to have a truly integrated and unified tool for all ground station scheduling needs, superseding all existing systems at GSOC (see figure 7, in comparison to figure 2).

This will reduce the number of involved components and interfaces, and thus decrease maintenance effort for the scheduling system. It will furthermore enable us to automate all ground station scheduling workflows at GSOC in Toucans which will allow us to reduce personnel required to cover shifts or non-routine activities like LEOPs. Together this reduces costs and improves our services to low-cost scientific missions. It will also provide a highly reactive scheduling system with a much larger ground station scheduling timeline, extended in both directions:

- With fully automated routine scheduling we could schedule routine contacts much further in advance, and then automatically update the schedule when the contact time comes closer and orbit prediction becomes more precise.
- On the other end of the timeline we could decrease the order deadline for short-notice passes without any manual intervention; mission operators could request last minute passes outside of regular office hours and have them confirmed almost in real time. An on-call night shift dealing with a spacecraft contingency could then already schedule extra passes for the next day shift for further recovery activities, shortening recovery times and saving precious mission time.

Fig. 7. Toucans in our vision



And, not least of all, we also hope to finally provide our own operators with one seamless scheduling software for space to ground communication which knows all the scheduling rules and has a consistent and transparent view to the overall schedule, to help them schedule station contacts and links efficiently and effortlessly, and enable them to fully focus on their core mission responsibilities.

8.2. Integration of Optical Links

A shorter order deadline and increased scheduling reactivity will also enable Toucans expand into the growing field of optical communication, which provides a number of additional challenges [16]. In particular, optical space-to-ground links may be hindered by local cloud coverage on short notice, thus requiring a highly reactive scheduling system. Within the frame of the European Optical Nucleus Network [17], Toucans will build upon GSOC's link planning concept [5] to gradually evolve from a testbed that explores suitable interfaces into a fully operational scheduling system for optical ground stations. On our way towards this goal, lessons learned from upcoming laser link missions such as CubeISL [18] or Compasso [19] will enable us to shape our tool suite further.

8.3. Real-time Schedule Distribution

Having a highly reactive scheduling system also requires real-time distribution of scheduling results. To this end Toucans will prototype fully-automated real-time scheduling distribution on top of GSOC's future service-oriented ground system and infrastructure framework HCC [20]. HCC participants such as mission planning or automation systems can use HCC to subscribe to the current schedule; whenever the schedule changes Toucans distributes the updated schedule to all subscribers, across network boundaries. This will enable mission planning systems as well as ground station sites in various network zones to react to the schedule update almost as soon as it is released, which contributes to shortening order deadlines for ground station passes and optical links.

9. Conclusion

Modernizing ground station scheduling in an existing control center is a huge undertaking, and we have a long way to go still before our vision of fully integrated and fully automated ground station scheduling becomes reality at GSOC. However, even though we still have a long way to go, we can outline a few conclusions and lessons learnt already now.

9.1. Interfaces and CCSDS

We have learned that CCSDS provides a very well designed and very helpful data model for representing passes and schedules. However, CCSDS unfortunately does not yet provide a good base for automated APIs around ground station scheduling, simply because the relevant CCSDS 902.9 Service standard [9] is far from being finished. As an alternative, we had good experiences with GraphQL-based APIs, both in PintaOnWeb [13] and in SCOTA [10]. In particular, the infrastructure for automated compatibility testing and first-class support for deprecations help us to design APIs which are very stable, but can still evolve to account for new use cases or improvements. Hence, we currently find custom APIs on top of GraphQL a good choice for APIs around CCSDS service packages, until the applicable CCSDS standards are finished.

9.2. Configuration, Deployment, and Quality

As outlined in section 7 the sheer criticality of ground station scheduling in a control center provides quite a challenge, which we address with rigorous quality assurance. In our experience, automated testing, deployment and strict peer reviews are substantial for code quality. Likewise, on a higher lever, structured decision making and, again, strict peer review are essential to maintain a high-quality architecture.

A high quality in code and architecture in turn enables us to deploy fast and frequently, to quickly deliver reliable changes to users. This drastically reduces the time between a change request and its final rollout, which enables us to adapt to requests for adaptations simply by changing the code, instead of having to provide operators with means of configuration. A small configuration surface in turn makes testing easier, simply because there are much less combinations of possible behavior to test, and easier testing contributes back to code quality.

We believe that a well-balanced mix of standardized interfaces and custom APIs together with our focus on quality will help us overcome the challenges we described in this paper. Gradually extending our existing workflows with more functionalities, interfaces and automation will pave the road towards a fully automated system for both radio frequency passes and optical links, thus enabling Toucans to shape the landscape of ground station scheduling at GSOC for the next decade and beyond.

References

[1] Wörle, M. T., Lenzen, C., Göttfert, T., Spörl, A., Grishechkin, B., Mrowka, F., and Wickler, M., "The Incremental Planning System – GSOC's Next Generation Mission Planning Framework." *13th International Conference on Space Operations*. Pasadena, California, USA, May 2014.

[2] Wörle, M. T., Lenzen, C., Wiesner, S., Prüfer, S., Petrak, A., and Müller, K., "Replacing the TDP-1 Mission Planning System – more than just another Technical Demonstration Project." *72th International Astronautical Congress*. Dubai, United Arab Emirates, Oct. 2021.

[3] Nibler, R., Hartung, J., Krenss, J., Fürbacher, A., Mrowka, F., and Brogl, S., "PINTA – one Tool to plan them all." *16th International Conference on Space Operations*. May 2021.

[4] Gnat, M., Frase, W., Barbieri, E., and Lagadrilliere, P.-A., "Implementing the CCSDS Service Management Interface at GSOC – Challenges, Obstacles and Considerations." *17th International Conference on Space Operations (SpaceOps 2023)*. 2023.

[5] Fürbacher, A., Fruth, T., Wiebigke, A., Wörle, M. T., Mrowka, F., Saucke, K., Martín Pimentel, P., and Knopp, M., "Concept for generic agile, reactive optical link planning." *CEAS Space Journal* (Jan. 2025). ISSN: 1868-2510. DOI: 10.1007/s12567-025-00592-0.

[6] German Aerospace Center, ed., *Ground Station Weilheim*. June 2022. URL: https://www.dlr.de/de/rb/forschung-betrieb/portfolio/dlr_rb_portfolio_groundstationweilheim.pdf.

[7] Consultative Commitee for Space Data Systems, *Cross Support Service Management—Simple Schedule Format Specification*. Tech. rep. CCSDS 902.1-B-1. Version 1. Consultative Commitee for Space Data Systems, Dec. 2021.

[8] Consultative Commitee for Space Data Systems, *Extensible Space Communication Cross Support—Service Management–Concept*. Tech. rep. CCSDS 902.0-G-1. Version 1. Consultative Commitee for Space Data Systems, Sept. 2014, p. 128.

[9] Consultative Commitee for Space Data Systems, *Service Management Utilization Request Formats*. Tech. rep. CCSDS 902.9-R-1. Consultative Commitee for Space Data Systems, Sept. 2022.

[10] Gross, E., Fruth, T., Dauth, M., Petrak, A., and Mrowka, F., "SCOTA – The Mission Planning Orbit Analysis Tool at GSOC." *72th International Astronautical Congress*. Dubai, United Arab Emirates, Oct. 2021.

[11] GraphQL Foundation, *GraphQL*. 2021. URL: https://spec.graphql.org/October2021/.

[12] Lenzen, C., Wörle, M. T., Prüfer, S., Wickler, M., and Fürbacher, A., "GSOCs Planning Library: History, Generic Features and Lessons Learnt." *Proceedings of the 13th International Workshop on Planning and Scheduling for Space (IWPSS)*. Ed. by S. Chien, J. M. Delfa, and T. S. Vaquero. July 2023, pp. 54–62.

[13] Wiebigke, A., Krenss, J., Hartung, J., Wiesner, S., Wörle, M. T., and Nibler, R., "PintaOnWeb - The Front End of GSOC's Next Generation Mission Planning Systems." *17th International Conference on Space Operations*. Dubai, United Arab Emirates, Mar. 6–10, 2023.

[14] Fruth, T., Lenzen, C., Gross, E., and Mrowka, F., "The EnMAP Mission Planning System." *15th International Conference on Space Operations*. Marseille, France: American Institute of Aeronautics and Astronautics, May 2018. DOI: 10.2514/6.2018-2525.

[15] Prüfer, S., Lenzen, C., Wiesner, S., Krenss, S., Wörle, M. T., and Mrowka, F., "Use Cases and Algorithms of the EnMAP Mission Planning System." *16th International Conference on Space Operations*. May 2021.

[16] Kaushal, H. and Kaddoum, G., "Optical Communication in Space: Challenges and Mitigation Techniques." *IEEE Communications Surveys & Tutorials* 19.1 (2017), pp. 57–96. ISSN: 1553-877X. DOI: 10.1109/COMST.2016.2603518.

[17] Krynitz, M., Heese, C., Knopp, M. T., Schulz, K.-J., and Henniger, H., "The European Optical Nucleus Network." *16th International Conference on Space Operations*. 2021.

[18] Rödiger, B., Rüddenklau, R., Schmidt, C., and Lehmann, M., "Acquisition Concept for Inter-Satellite Communication Terminals on CubeSats." *Small Satellites Systems and Services - The 4S Symposium 2022*. May 2022.

[19] Dauth, M., Scharringhausen, J., Rajapakse, A., and Kling, U., "The Compasso Mission: Operational Strategies for ValidatingOptical Technologies On-Board the ISS." *17th International Conference on Space Operations*. Mar. 2023.

[20] Hauke, A., "GSOC's Service-Oriented Ground System "HCC" - Status and First Experiences from Sounding Rocket Missions." *17th International Conference on Space Operations (SpaceOps 2023)*. Mar. 2023.