17th International Conference on Space Operations, Dubai, United Arab Emirates, 6 - 10 March 2023.

SpaceOps-2023, ID # 553

GSOC's Service-Oriented Ground System "HCC" - Status and First Experiences from Sounding Rocket Missions

Armin Hauke[a]*

[a] *Department Communication and Ground Station, German Aerospace Center, Münchner Straße 20, 82234 Weßling, Germany,* armin.hauke@dlr.de
* Corresponding Author

Abstract

At the German Space Operations Center (GSOC), a service-oriented ground system is developed under the working title "HCC". Beside the decomposition of the ground segment into services, these activities cover the definition and implementation of a common infrastructure framework consisting of basic services specifically suited for space operations: data transportation, security aspects, common configuration, centralized storage and further more. While primarily targeted to support satellite and human spaceflight operation, it was realized that this infrastructure also serves well for the ground data system of sounding rockets, launched and operated by GSOC's department "Mobile Rocket Base" (Moraba). In fact, using the HCC framework for the telemetry distribution of sounding rockets allows to bring the HCC basic services into operation with no need to have all the tools necessary for satellite operation already completed to work in the HCC context.

We describe the fundamental ideas of HCC along with its status, especially the implementation of the basic services. Some of them have been used for the first time in the context of the Mapheus 12 mission, executed by Moraba on October 21st 2022.

1. Introduction

The environment in space operations is changing without any doubt. Nowadays there are many and very flexible ways to fulfill a certain task in space. Especially the development of cube-sats has been proven to be a game changer. Along with much smaller and much cheaper satellites, the cost to launch them into space has decreased as well. Of course, these changes also have an impact to ground operations – in terms of cost as well as in terms of operation concepts. To cope with this, operations on ground has to be much more flexible than in the past and reduce the need (and special development) of mission specific tools. Also, it is clear, that ground operations will be much more automated than before.

But it is not alone the monetary pressure that drives this evolution. By using optical links directly from space craft to earth, the impact of the atmosphere to the link demands flexibility in using those downlinks. It will become necessary to react quickly to changing atmosphere conditions by either adapting the downlink or completely switch link between various reachable optical ground stations. Contrary to routine operations with typical radio frequencies, such reactions will not be planned well in advance but rather be applied ad hoc to the situation.

And last not least, the planned return of mankind to the moon and the outreach beyond will lead to operations concepts with much more automation on ground and autonomy on board. For example, a future space-station orbiting the moon will not be permanently manned contrary to how the ISS is operated right now. Such an unmanned station most probably shall not require intensive use of personnel on ground.

So, if flexibility and automation are the key features of space operation in the future, what are technologies and design patterns for the ground segments to enable this? The answer is a service-oriented design, a modular structure and well defined and standardized interfaces in between.

It is not by chance that these design features are implemented into ground segment software at various levels by many agencies and companies. Some examples are EGS-CC and derived from this the ground system EGOS-CC by ESA. Also, CNES' ground system ISIS shows these design features. On a larger scale, within CCSDS there are standards under way for service oriented cross support by ground stations, as well as a standard for a message abstraction layer in mission operations, MO/MAL.

At the German Space Operations Center (GSOC) a cross-departmental project "HCC" was launched to pave the path for GSOC to offer its competence and well-established tools in such a service-oriented way. Being focused on cube-sats and small sized ground segments at first, it was realized very quickly that the principles and ideas would fit very well also for satellite missions typically operated by GSOC before, and furthermore even in the context of

human space flight. At that point the project name "HCC" was re-defined to be "holistic control center" – holistic in a sense, that we apply the ideas of service orientation and modularity not only to the applications of some ground system but also to the ground segment as a whole.

2. HCC as a system

As pointed out in the previous section, HCC is not just a particular implementation of ground segment tools. Moreover, it is an underlying concept of how a ground segment is constructed and how the implementations act with each-other and together as a whole.

The vision is, that GSOC has a catalogue of services it offers. And these services are building blocks that can be selected as needed for a particular mission. Of course, there are constraints and dependencies in a way that some service may require other services to exist. But such constraints only emerge from the logic of the system (which functionality is needed) and not from technical aspects of the service implementation.

2.1 Use-cases and Benefits

Once this vision gets realized, we see advantages in many fields. Some of them are highlighted in the following as examples.

The modularity of the system allows to tailor the overall ground segment much better to the needs of a particular mission. In fact, it also allows to provide a set of requested services rather than a complete ground segment. That way GSOC can complement some customers own implementation and vice versa. This opens the door to much more flexible concepts how the workload of mission operation is distributed between mission owner and control centre or between different control centres.

As the building blocks of the whole system are defined by their functionality and their interfaces, single services can be replaced or updated without problems. Especially for a control centre like GSOC, operating many missions in parallel, this allows to keep the services for all missions on the same version. Improvements triggered by one mission become available for all other missions as well. In turn, implementing necessary updates to keep track with changes in the operating system or other elements, are done just service-wise and no longer per service and per mission.

Within the system, services are interacting with other services. And the organization of the communication should be just as easy as that. Know your communication-partner and do not bother about all the details of IP-settings, firewalls in between and all the other obstacles – let someone else (another service!) take care of this. In other words, within HCC an infrastructure is created that allows all entities connected to it to communicate, instead of administrating all needed peer-to-peer connections individually at both ends.

The essence of a service-oriented architecture is, that whoever does use or consume a service shall and must not care about how this service is realized internally. And the same is true vice versa, a service provider has to be agnostic with respect to implementation details of the service consumer. In particular, a service is provided the very same way, regardless if it triggered by some manual interaction of a human operator or by some automated technical device. This of course opens the door for highly automated operations.

2.2 Mission Specifics

It was well-considered in the previous section, to name this concept a vision. We are well aware that space missions differ and there won't be a one-fits-all solution to every mission. Nevertheless, this is not a show-stopper for the project HCC. Even if space segments differ significantly due to mission goals and payloads, there is a broad commonality among them, just because of the physics required to operate and keep a space-craft in orbit. Talking to experienced operations people at GSOC, there are estimations that around 60-80% of the operation tasks could be generalized. And with this number, one has to keep in mind that GSOC mainly operates cutting edge technology satellites.

However, even if a substantial part of a particular ground segment may be mission specific tools developed individually, the service-oriented design of the system helps to cleanly separate between generic and specific components. That makes the maintenance of the mission specifics much easier. And it also generates a much better visibility of the efforts arising from the mission specific developments.

3. HCC as a Middle Ware

In the previous chapter we have outlined, how a service-oriented architecture for space operations shall look like and why we expect great benefits from it. But already when touching the topic communication, it became clear that such an approach not only needs the space operation aspects to be organized in services. Just as essential is an infrastructure that ties all the services together. At this point, the term "HCC" no longer stands for principles and design only – one may call it a philosophy – but HCC also becomes a concrete software project.

After experimenting a bit with of-the-shelf solutions and finding, there are many tools for various aspects available, but none designed for the special environment needed to do space operations, we stepped back into a design phase. With this, we worked out eight topics that are commonly relevant for space operations: Security, Automation, Operations, Configuration, Logging, Storage/Archiving, Life Cycle Management, and Data-Transport. The order, they are given here, reflects roughly the dependencies they set to each other. And it is on purpose, that security and automation are listed first as we deliberately chose to have these aspects considered from the very beginning instead of building up a system and try to secure and automate it afterwards.

A second paradigm leading us from the very beginning is, the system and all its basic services shall be open to the various data types and formats used in space operations. Of course, the modularity of the system can only unfold its full power, if the interfaces of the modules are standardized, but we do not intend to create and enforce those standards ourselves.

The most obvious candidate affected by this paradigm is data transportation. If two services communicate with each-other, it shall remain up to those two services to agree upon the data formats and interaction patterns, while HCC provides the transport layer in between. If available, standards defined by CCSDS are welcomed. And most often, CCSDS specifications define data description and interaction but explicitly leave out the technical specifications of the transport layer.

We are aware, there are exceptions. The most prominent one may be the SLE-standard for TM/TC-exchange between ground stations and control centres. Again, one goal of HCC is, to create an environment in which such well-established and widely applied standards can be kept. Since SLE specifies a protocol (directly based on TCP/IP), not only a data format, it might be difficult to wrap the SLE protocol into the HCC data transport. But there is no need to do this anyway. The more interesting part is the so-called managed information, that communication partners need to share beforehand in order to establish an SLE-connection. Coming back to the picture of services as building blocks for a ground segment, two entities being part of an HCC-system can utilize HCC to organize and manage a direct peer-to-peer connection between them. This direct connection will lack the benefits provided by the HCC transfer-layer, but that may be balanced by the advantages provided by a different protocol.

A similar example would be storage and archiving. Neither will HCC set rules how to structure and format data before storing them, nor will HCC restrict the physical characteristics of the storage device in any way. But the specification of storage within HCC will ensure that data is always stored together with meta-information, stored data is accessible, access to the data is governed (interplay with security aspects), and so on.

3.1 Basic Services

Following the discussion above, the listed eight topics correspond to services that are utilized within any ground segment in one way or the other. The only exception may be the topic operations which is more related to user interfaces than representing a functionality that can be technically implemented as a service. Fig. 1 illustrates, how a service-oriented ground segment would be built out of HCC basic services and mission operation services.

In the following sections, we will specify the functionalities of these HCC basic services.

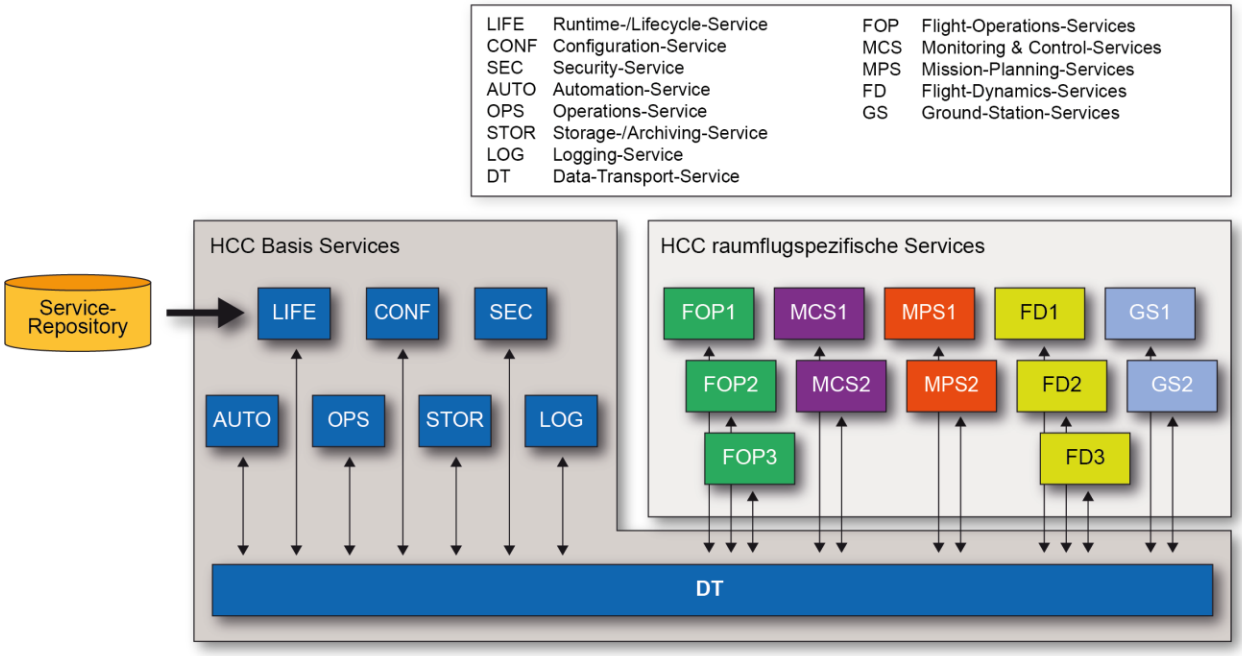

Fig. 1. A service-oriented ground segment built upon the HCC infrastructure. Shown in blue are the HCC basic services, accompanied by the space-flight specific services, as demanded by the particular mission.

3.1.1 Security (HCC-SEC)

The security service provides a centralized user management to the overall system. This includes two aspects: authentication and authorization. Any component connecting to an HCC-system (technically that is, connecting to the data transport service) must pass a verification at HCC-SEC. This is true for the pure connection of an application, as well as for the login of a human user.

The verification of the login by HCC-SEC is done with respect to a centralized user database such as LDAP or Active Directory. In fact, it is foreseen that HCC-SEC can access several databases depending on the login request. For example, all DLR employees have user credentials in a DLR-wide system, while at GSOC an additional LDAP system is hosted to enable mission operation in collaboration with personnel not affiliated at DLR, and hold functional accounts for mission operations. Also, the requirement newly coming with HCC, that applications itself have to be authenticated demand a database to hold the information, which applications are registered for which mission.

The login of services at HCC-SEC is closely related to their connection to the data transport service (HCC-DT). If the TCP/IP connection between a component and HCC-DT is terminated, HCC-DT informs HCC-SEC on the connection loss and consequently this component is removed from the list of logged-in services and all corresponding session tokens are revoked.

Together with the authentication, HCC-SEC reports for each user role-based information for authorization. This enables any service connected to the HCC-core to check and decide, if an incoming request is authorized and hence will be processed. Again, it is on purpose that the HCC basic services provide the functionality to establish such mechanisms, but it is left to the actual implementation of each service, to which extend those mechanisms are used. On one hand, it protects the HCC project from the work to create access rules suitable for every mission. On the other hand, this freedom makes it more attractive for existing applications to be integrated in an HCC environment.

Technically, each message sent any service within HCC, contains some kind of session token of the sender. With this token, the receiver can consult the security service and check the access rights of the sender. Internally this information may or may not be stored by the receiving side for a while, effectively creating a session validation between sender and receiver. For operations with much higher security demands, the check can also be done for every single incoming message – taking into account all the drawbacks concerning increased traffic and reaction times.

Part of the communication between HCC components and HCC-SEC is also the ability that HCC-SEC informs all relevant services about changes in some user's role assignment. That way HCC-SEC can actively revoke rights

for users during run-time without the need for all components, to check the actual authorization for some client for each incoming request.

3.1.2 Data-Transport (HCC-DT)

A necessary key component of HCC is the ability to exchange any data between its services. This functionality is provided by the data transport service HCC-DT. It is built as a tree-structure of several nodes. The nodes (except the top-level root-node) establish a TCP/IP-connection to their parent node and act as server for other nodes or services. The decentralized implementation of HCC-DT allows to distribute the whole HCC-system areal. It also allows to organize the distributed system in a way to connect components with large amounts of data exchange between them to the same DT-node.

The tree-structure allows furthermore to give a project a vertical structure corresponding to a logical structure, e.g. centralized mission-wide services and components for single space-segments in missions operating constellations or fleets of satellites. Horizontally the tree-structure can be utilized to incorporate instantiation of services, e.g. for redundancies or load-balancing.

Finally, single nodes can be set up as proxies in order to have dedicated bridging-points between areas, especially network areas. Together with the incorporated security concept, this allows to extend the data transport service from within a single control centre to other partners or customers.

Services connecting to the data transport service can communicate to other services using service names only. The actual physical position of the communication partners is irrelevant for their exchange and handled by HCC-DT. In turn, any component providing some functionality can be reached by any other component.

The DT-nodes can completely be set up in a hot redundant way with two (or more) physical applications for each logical node.

Data exchange between services over HCC-DT works on a packet base, each packet consisting of an HCC-header and one or more bodies, that can be freely determined by the services. In that way, using HCC-DT has no constrains on formats and structures of application data.

All payload bodies can undergo an end-to-end encryption by the sender/receiver. Only the mandatory first body of a packet, containing the addressing of the packet, is readable and evaluated by HCC-DT. Encrypted communication is the default setting of HCC, but can be disabled by configuration, e.g. for performance reasons. If desired, an additional encryption on transport layer operating on the TCP/IP-level between the DT-nodes can be applied.

3.1.3 Configuration (HCC-CONF)

For a distributed system such as HCC a centralized configuration is essential. Therefore, the configuration service HCC-CONF is part of the HCC-core system. HCC-CONF by default manages all configuration items to define and run the HCC-system itself. This is in particular the structure and localization of the core-system such as for example the tree-structure of HCC-DT.

Any application connecting to the HCC-system contacts HCC-CONF after registration with HCC-SEC and receives its configuration. The global master configuration held at HCC-CONF is a structured one and the structure maps to the roles as defined at HCC-SEC. With this mechanism each application receives exactly those configuration items only, that are accessible for the particular application. Due to the group structure on the other hand, several applications may have access to the same configuration items, keeping their configurations coherent by design.

Also controlled by the access rights through HCC-SEC, the configuration service restricts read-only access to the configuration (items) or allows for write operations to alter the master configuration. Changes in the master configuration are communicated to the relevant components. The enforcement of the changes is done in a step wise process. This allows all application to receive (and acknowledge) the altered configuration first, to evaluate the changes and may prepare their own services to apply them, and only finally after that the changed configuration is activated. This mechanism not only assures that all components switch configuration in parallel, it also allows to prepare a change in the background and activate it in some suitable time slot depending on mission execution.

Likewise, to the other basic services, the configuration service offers functionality. The core components of HCC obviously rely fully on HCC-CONF and it is recommended that all services make use of this centralized tool. However, there may be configurable items that are important for a single service but plays no role at all for any other service. In such cases, these configuration items might as well be just kept locally to their service. Similarly, in a transition phase when already existing functionality is transferred into the HCC system, it can be helpful to connect such a system to the data transport layer only at first, and add further functionality from the HCC basic services such as configuration later on.

Broadening the scope from the HCC system to a full featured mission ground segment, it is also worth to spend some time on the question, what precisely is treated as "configuration". For example, the description of the mission itself can be seen as configuration. But that does not necessarily mean, it must be stored and managed by the configuration service. Usually there is some kind of "mission database" containing the information how to interpret telemetry, and how to parameterize telecommands. All this information is traditionally held within the missions M&C system. And that can stay this way, there is no reason to rip this out and move it to HCC-CONF. One can achieve the envisioned service-oriented structure might as well by implementing an interface to the M&C system that provides the contents of the mission database to other services.

3.1.4 Logging (HCC-LOG)

Similar to the configuration, HCC mandatorily comes with a centralized logging service. Any component connected to the HCC data transport layer can utilize this logging service to store information on events. Usage of the HCC data transfer service assures that all logged events are automatically annotated with a set of basic information such as identification of the source (application as well as additional information like master/slave, instance number for redundant or load balanced systems) and time.

The interface of HCC-LOG defines commonly, which information has to be provided for every log-event. This assures all components do their logging in comparable manner, e.g. definition of log-levels.

Besides the centralized storage of logging information for all components, HCC-LOG provides various methods to access this information. Human users can filter the logging for time-spans, certain processes etc. In the same way, automated processes can access the logging information to perform periodic checks or create reports. While generating logging information is granted to all components, access to this information is governed by HCC-SEC and its role-based authorization functionality.

Alike the configuration discussed earlier, for logging data one may distinguish between information that is relevant for the functionality of the entire system as a whole, and private information that may be relevant for a single service but with no interest to correlate it to other services. As it is true for other HCC basic services, there is no mechanism to prevent any process to locally store logging information. In fact, it is recommended that HCC-LOG is utilized for events affecting other services or resulting from problems encountered by other services. But HCC-LOG is not meant to replace any local process logging entirely.

3.1.5 Runtime and Lifecycle (HCC-LIFE)

The run-time- & lifecycle service controls the run-state and version of all services building up the HCC system. Part of this service is an HCC-component called little-daemon (LD) that is deployed to each server (bare metal as well as virtual machines) supposed to host HCC components. LD is connected to the operating system of its host in a way that LD starts automatically with every boot.

The main task of this application is to provide an interface between HCC and the OS of the host to allow HCC to start processes/HCC-services. Besides, each LD is an HCC-component itself, providing monitoring and logging information to HCC, receiving its configuration from HCC-CONF, and being reachable for any other HCC component with adequate permissions.

As part of HCC-LIFE, every LD checks the version of the HCC-services installed on the particular host. For version updates, LD can receive the updated versions of executables or libraries through HCC-DT out of a service-repository, put them in place on the local file-system or connect them to the operating system. This can be done in parallel to ongoing operation of the recent versions of the services. If all relevant components are ready for the upgrade (may including preparation of configuration changes), the update can be executed commonly.

Beside the version control of all components, the second task of HCC-LIFE is to provide a real-time system monitoring (HCC-MON). While the logging information in HCC-LOG is mainly used to analyse issues posteriori, the monitoring service is suited to indicate developing incidents beforehand. Similar to the logging, each HCC component can freely define parameters and update them with values. Those values are centrally received by the monitoring service as raw data. Specialized processors can subscribe to these values and perform correlations, trend-analyses, detect limit violations and further more.

For the LD-components discussed above, the provided monitoring data contains basic information on the OS, such as CPU usage, memory consumption, number of processes, available disc space etc.

3.1.6 Storage and Archiving (HCC-STOR)

The storage and archiving service provides to all services within an HCC-system the opportunity to store data. As already mentioned in the introduction to this chapter, the storage service is the one that has to be most generic. Literally all kinds of data shall be handled by it. What the interface of this service requires is, all stored data gets

annotated. A basic set of meta data is created by the HCC system automatically, e.g. information like data source (service that invokes the storage) or time. Other information such like validity time or meta-data describing the contend, are assured by the structure of the service interface.

Establishing the storage and archiving service is not a task emerging only from the HCC project. Independent from HCC, a working group has been established at GSOC, to specify what is needed to create a centralized GSOC storage. This includes a survey, which data-types are stored, which access-times are needed, which meta-data shall be added, together with technical questions, which storage media to use, how to prevent data loss and many more aspects. All of this fit very well with the HCC concept, and using HCC and its basic services as path to store and retrieve data, already provides solutions to some problems worked out, for example in the context of security and access control.

Another aspect highlighted within this working group is the transition from all the individual storage areas populated right now to a centralized common one. Having HCC-STOR as access point for all stored data, this transition can be short-cut at once on a logical level. Requests for storage and retrieval are always communicated to HCC-STOR, but the physical access to the data within this service takes place at one out of the variety of existing storage areas.

3.2 Implementation Status

In the previous section, we have worked out descriptions of services, needed to build up an infrastructure suited to support a service-oriented ground segment for space operations. In the following, we will describe what has been realized already, to which extend such an infrastructure is ready to use, and if there are already existing service implementations for space operations utilizing the HCC technology.

3.2.1 Basic Services

Although from a logical point of view, we have set the security service and operations/automation to the highest priority, for implementation purposes the root of all is an existing transportation layer. Therefore, the data transport service HCC-DT was the starting point for the actual coding, quickly followed by the security and the configuration services.

In fact, HCC-DT is very mature by now. The individual DT-nodes have been optimized for data throughput. In an optimized setup, the DT-nodes can handle traffic loads close to the limitations of the underlying network. Optimization here means, that the packaging of the data is done in a decent way. Both, too large bulks of data and too few data per package will obviously limit the performance. If the packaging of the data cannot be altered for any reasons, the modular structure of HCC also allows to apply some optimization. If necessary, one can think of a dedicated hardware to host a single DT-node only and being optimized to do I/O. With respect to the data, we have tested HCC-DT under various scenarios, ranging from individual messages (as within a classical request/response communication) to large data chunks (file-transfer) on one extreme and streaming data (for example audio-data) on the other. As mentioned above, the complete data transport service can be set up in a fully hot redundant way. And finally, we have successfully tested to span the entire service over a wide area network, installing several DT-nodes on computers in individual home offices and connect them through public internet with local firewalls and all the network components in between.

What may be needed additionally in the future are specialized DT-nodes to serve as proxy-applications, for example at the boundaries of control centre premises. Those special applications may require dedicated ways to establish the connections to their parent node and/or to their connected clients. Technically it is clear how to implement this, it just has not been done as we do not have the use case yet. From the perspective of the entire data transport service within HCC, we do not see major problems arising.

The next service to be implemented was HCC-SEC. Here we do have a service that basically provides all the features discussed – authentication and authorization – and all the necessary communication to other services within an HCC-system. However, the implementation at the moment is done in a straight forward approach to have something to work with. That means, internally the information is not held in a way, it should be done for an operational mission. For instance, there is not yet a connection from the service to a professional user database. Users and their permissions are just stored within some database locally. Of course, this is not acceptable for real operations, but moving this information from that database into a dedicated management system such as LDAP and retrieve the data through an interface to this external system is transparent for the rest of the HCC-system. Here the service-oriented architecture proves its advantages. Changing the user database does only require to change the interface between this database and HCC-SEC. All other services access user data through the security service only and hence do not need to change anything at all.

A similar situation we have with the configuration service. There is a working implementation that provides the configuration to all other services, but this configuration so far is held internally in a single xml-file locally. Likewise, the security service, this will be replaced by a database in the future. However, the essential features are implemented, such as access control (which service gets which configuration items), versioning of changes, and most importantly the process to distribute changes during run-time. Having started with the HCC basic services for obvious reasons, the content of the configuration currently is defining the structure of the HCC as a system. For this, a structured file such an xml is perfectly suited. However, as soon as more and more real application services get connected to HCC and use HCC-CONF to manage their configuration, the situation will change. But as for the security service, improving the implementation of the configuration service is possible and decisions on how to internally represent the complete configuration can be skipped for now and also altered later. Even hybrid solutions are possible, such as keeping the system related data within an xml-structure and having other service configuration in a separated database.

With working implementations of security, data-transport, and configuration, we had a minimalist set of services to create an HCC infrastructure. In fact, those three services are mandatory to build up a system, as those three rely on the existence of each-other. Security and data-transport require to retrieve their configurations from HCC-CONF, HCC-CONF in turn relies on the security service to manage access to the data. Without the data-transport service there is no communication between services at all and HCC-DT also cannot work without the security service, because other services trying to connect to some DT-node are blocked as long as they do not succeed to perform a valid login with HCC-SEC.

Having a distributed system at this stage already, the next basic service to be added was the logging service. Implementing HCC-LOG followed the same strategy as the other basic services before: have a complete implementation of the interface and start with a straight forward version of the service itself. For the logging service that means, messages sent to HCC-LOG are stored in flat files locally for now. Therefore, retrieval of stored log messages is not optimized at all, but possible with all needed features, e.g. access control.

Last not least, we have started to provide a working implementation of the run-time and life-cycle service. As discussed above, here we have several tasks combined: installation and version updates of services, as well as monitoring of the run-time behaviour of the HCC services and the underlying hardware. In terms of applications, therefore HCC-LIFE is split into two independent parts: the little daemon LD for installation and versioning and a separated monitoring service. The little daemon is currently under development, the monitoring service as a first working implementation available. What is mission with HCC-MON are features like storage and history retrieval, but as a real-time system, it is capable to collect monitoring information from other services and also provide them to subscribed clients. An important feature of the monitoring service is, it works without a predefined configuration of the set of monitoring data. Services can create and remove monitoring parameters during run-time and update them in between. This also reflects the service-oriented structure of HCC. The monitoring service offers functionality to distribute monitoring information, it does not require from other services to provide certain information.

3.2.2 Mission Operations Services

As soon as the mandatory three services HCC-SEC, HCC-DT and HCC-CONF were available, a test-bed was created with a permanently running HCC infrastructure, open for other services to connect to it. That way, anyone at GSOC responsible for space operation software is able to adapt the interfaces of his software in a way, HCC-DT is used as transport layer and the functionality provided becomes an HCC-service.

To make this adaption for software developers as easy and attractive as possible, an API has been established that provides methods to access the HCC basic services. This API is currently available in JAVA and C/C++ and there is also a python-variant based on the C-implementation. With these three programming languages, most of the software projects at GSOC are covered, or at least can rather easy be adapted. The most important part of the API of course is its interface to the data transport service. Most of the subtle details of the protocol defined by HCC are completely covered within the API, providing high level functionality to the user, such as “send this data to this destination”. Coordination of outgoing requests and incoming responses is done as well as error handling, e.g. through timeouts if answers are missing, or – in the other direction – inform a waiting recipient if a response gets delayed and processing still takes some more time. Also, the communication with the security service is handled directly by the API, checking by default the authorization of any requester at its first appearance and cashing this information as session for a given time-interval. The structure of an application utilizing the HCC-API is shown in Fig. 2.

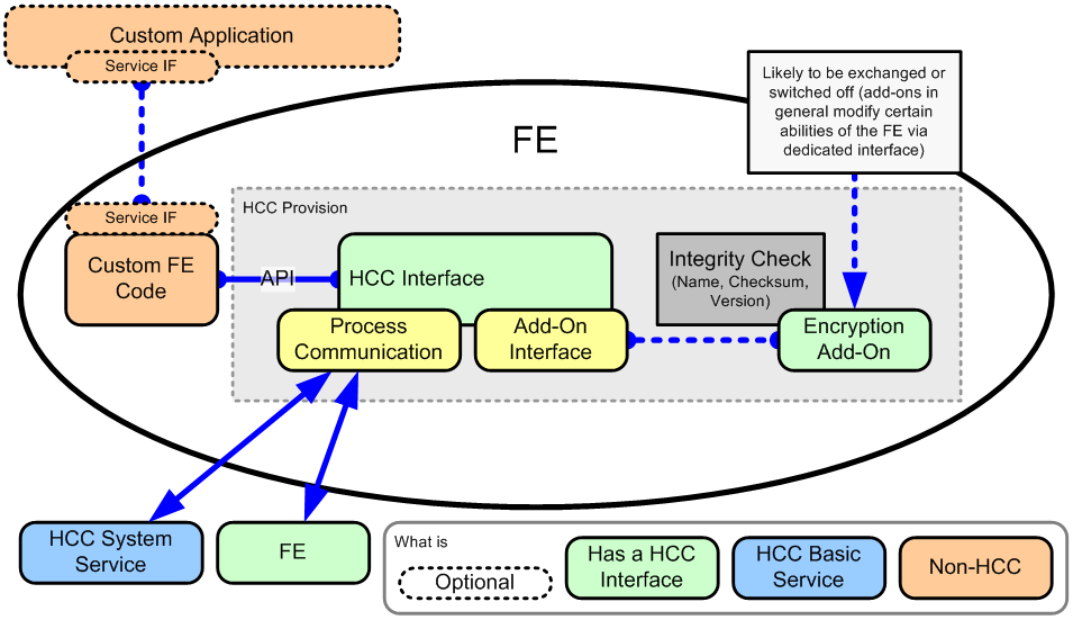

Fig. 2. Integration of a custom application into HCC. Within the application (FE for functional entity), the HCC-API provides the binding to all other services of the HCC-system, basic services (shown in blue) as well as other mission operation services (other FE indicated in green).

The detailed behaviour of the API can be configured to some extend – for example the timing intervals can be set – but where the configuration does not provide the needed flexibility, the underlying lower level methods are accessible as well and the application can be fine-tuned completely as needed. For instance, if one has a high security application or a service that behaves quite differently depending on the permissions granted, the process chain of receiving a message, check the sender's permissions with HCC-SEC and decide on how to proceed can be taken over by the main application and executed step by step using low-level functions of the API.

A first user of the test-environment provided by HCC is the flight dynamics department at GSOC. These colleagues had a head start in the sense, that their applications already were organized in a service-oriented way, independently from HCC. Examples here are requests for orbit data for a given space-craft or conversions of those orbit descriptions into pointing files for ground stations. Thus, in that case the change from the recent implementation to HCC was merely a change of the transport layer from placing files to an FTP-server to send the file content via HCC-DT. But also, in this rather easy use-case, HCC already offers some interesting new features. With the file-based system operated so far, flight dynamics assured that the most recent data was available at all times, but it was opaque to users, when there were updates to the data. In fact, in most cases re-calculation of orbits took place based on fixed intervals in time, e.g. once a day. Changing from file-based technology to the HCC system, new possibilities arise. Now a signalling mechanism is easy to implement – called "FD notification service" – that informs interested partners if changes to the provided data have occurred.

Test-partner for these flight dynamic implementations of HCC services was the DLR ground station at Weilheim. For technical reasons, it was only a provisional implementation not connected to the operational antenna control system. But it was shown that the desired functionality is established and the HCC technology is foreseen to connect the ground station operations to the departments at GSOC located in Oberpfaffenhofen in the future. Beside flight dynamics data, the planning and booking process for the antennas at Weilheim ground station, executed by the scheduling office at GSOC, is another candidate to be implemented in a service-oriented architecture based on HCC technology. In this case, it is more than an exchange of the transport layer, as the scheduling office currently implements a new developed system following the CCSDS service management standard.

Finally, there are the core processes for space operations, the monitoring and control system for the space-craft, the mission planning system and offline-tools to process and analyse received telemetry. For those systems, first studies have been carried out to investigate how the existing interfaces can be adapted to the HCC transport service, how further functionality provided by the other HCC basic services can be utilized, and last not least how the existing systems and tools fit into the envisioned service-oriented architecture of the whole ground segment.

As for the evolution of the ground station scheduling system, the project HCC is not the only driving force for mission operation systems. For space-craft M&C, there is the undergoing integration of EGS-CC into GSOC's multi-

mission environment, planning tools evolve from well-in-advance planning cycles to much more reactive systems, and also TM analysis changes drastically by using artificial intelligence. From our projects point of few, these evolution processes should be coupled to HCC, not only technically but also in the way that such systems in future follow the ideas of a service-oriented ground segment by nature.

4. The Moraba Ground Segment

From the very beginning, HCC was focused on satellite operations. Spanning the range from typical earth observation missions such as Terra-SAR/TanDEM to the emerging field of cube-sats on one side and also having in mind human spaceflight with the Columbus module at the ISS on the other extreme, this already seemed to be very challenging. Indeed, within the project it was always agreed that we would try to cover as much application cases as possible, but keep a realistic view on the problem in case the various requirements would spread to broadly. With this latent scepticism in mind, project members were surprised when they were contacted by colleagues from Moraba (MRB), GSOC's department "Mobile Rocket Base", following up an internal presentation on HCC.

At Moraba, a design phase was under way to specify an upgrade or new implementation of their ground segment. Functionally, the ground segment consists of several antenna stations all receiving the complete telemetry transmitted by the rocket. This telemetry is structured into several TM-streams – 16 different streams in this case – and the TM-streams are distributed to a variety of users, operators from Moraba controlling the rocket itself, as well as various experiment operators responsible for the different payloads on board. The main task of the ground segment is, to provide to each of those data sinks the relevant parts of the telemetry and allow for each destination a selection, which ground receiver shall be the source (see Fig. 3).

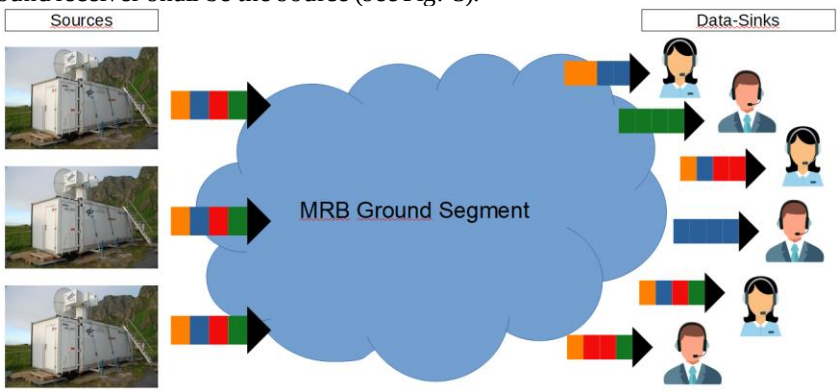

Fig. 3. Sketch of the Moraba TM distribution. TM-streams are indicated by different colour. The selection of the data sources has to be done dynamically for the various streams and data sinks.

The data routing provided by the HCC data-transport service looked very appealing to our colleagues. And although it was never our goal to replace the well-established real-time TM/TC-connections, we started prototyping a demonstrator, if HCC-DT would be feasible to distribute rocket telemetry for Moraba.

There is a second reason, HCC seemed to fit perfectly for Moraba's needs: In order to stay mission-ready during the full process of upgrading their ground segment and minimize the risk, Moraba planned to do a step-wise approach, replacing one component after another, instead of building a new system completely in parallel and switch everything at once. Here the concept of the HCC data-transport service of being agnostic with respect to payload data, was a key feature. This allowed Moraba to use HCC-DT purely as transport layer without touching the definition of their telemetry formats. That way, telemetry processing and display could be kept as they were, while replacing the data collection and selection upstream.

In the same step-wise approach, the utilization of HCC was completely narrowed to data-transport for now, but other HCC inherent features, especially the ability to control data access using the security service, were promising to be added later on.

4.1 Telemetry Distribution at Moraba with HCC

As mentioned above, the first prototype for Moraba was a sequence of just two data connections. The first one wrapping telemetry frames, received with a rate of 4Mbit/s, into the HCC protocol and send them to a component supposed to decompose the frames into stream-segments, and the second one forwarding the stream-segments to the third and last component, the receiver. Besides the successful proof of principle, there were two important outcomes from this first prototype: Firstly, the data-transport service proofed to be as flexible as expected in terms of data

formats, and data-rates of 4 Mbit/s were handled without notable load to the system. Secondly, once being familiar with the structure of the provided API, it turned out to be quite handy to make proprietary user code an HCC-service by connecting it to an HCC-system utilizing the HCC-API.

After this promising start, the next step was done by replicating the data source component in various peculiarities reflecting the different technical implementation of the different TM stations, how the TM-frames are delivered. Together with this, the second component had to be extended not only to decompose the TM-frames into stream-segments, but also allow the selection of TM-sources for the stream-segment receivers. At his point, the advantages of HCC and its holistic approach started to unfold.

It was realized, there is no need to have a dedicated component to do the distribution of the various stream-segments to the receivers. This functionality can be covered by the configuration of the entire system. That way, data traffic can be reduced drastically. Instead of bringing all data to a central hub that does the distribution, the data transport service itself can be seen as a virtual or de-localized implementation of such a hub, providing all necessary data (and only those) from the various sources connected to it.

Moving the functionality of data distribution into the configuration of all the data sources raised the question, how to change the data paths during operation without having to change the entire system configuration each time. The solution was to apply the actual change directly at the data source at any time, bypassing the configuration service. To make such a change persistent and bring the system configuration into a consistent state, the change is played back into the configuration by the component where the change was applied.

This process is definitely something, that will never be applied in the context of satellite operations, as configuration changes have to undergo a controlled process of testing and deployment in such an ecosystem. However, in the service-oriented architecture of HCC, there is not that much difference between both concepts. In both cases, the configuration service receives a message via its service interface to change a particular value. This request is always checked at first, if the sender has the permissions to do so, and only if the permissions are granted, the configuration service acts. Here we benefit from the concept, that within HCC not only humans in front of an MMI are users, but every component connected to the system is authenticated and linked to particular permissions. Regardless of who the human being is, that commands a change – this identity of course is controlled by some login-process as well – the configuration service identifies the component used to issue any change request. Usually, such requests have to stem from an administration tool such as an config editor, but it is just a question of permissions to allow other processes to create requests and have them executed.

With this setup, bench tests at Oberpfaffenhofen were carried out and rather quickly the trust in this new system grew in a way that HCC was considered to become the operational system for upcoming missions.

4.2 Mapheus 12

The first mission where the new HCC-based telemetry distribution was used operationally finally was the Mapheus 12 mission, launched from Esrange at Kiruna on October, 21st 2022. Although no problems were encountered during the test campaigns, there was an open point that raised some concerns. As HCC replaced the data distribution between legacy systems for TM ground reception and TM processing, the packetizing of the data was bound to follow what had been used in the past, the stream segments with a size of 15 Byte per packet. It is clear that an elaborated protocol, such as the HCC-protocol designed to provide an inherent security layer, is not optimized to handle packets as small as 15 Bytes. The obvious solution would be to accumulate a larger chunk of several stream segments, but this may cause timing problems at the receiving end, since the data no longer arrives continuously but in bursts.

As said before, one of the selling points of HCC for Moraba was the ability to replace parts of the ground segment one by one, keeping the existing legacy system for all others tasks. So, instead of taking the risk that the existing TM processor may need adaptations to the modified timing of the incoming data, it was decided to transport the telemetry out of band directly on the network layer using UDP (see Fig. 4), as done for previous missions.

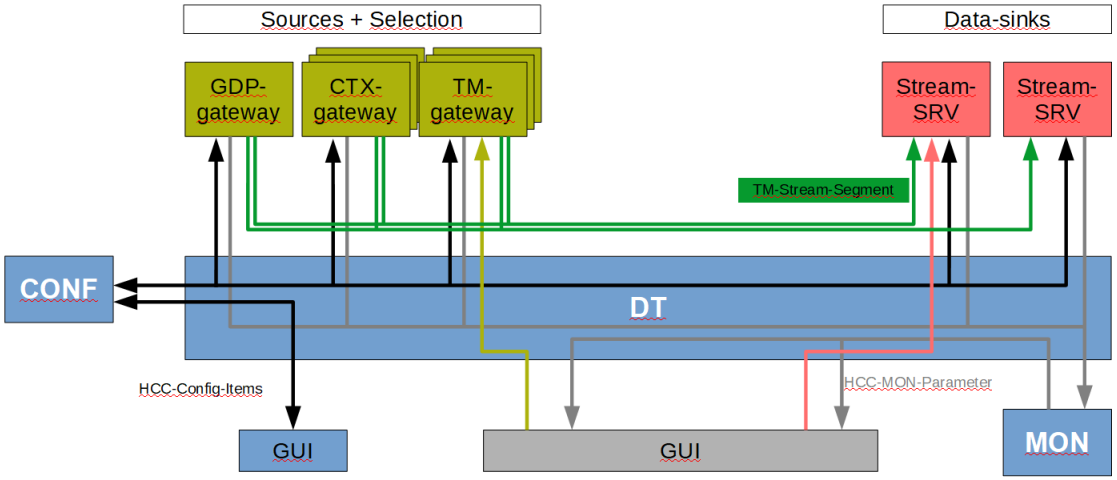

Fig. 4. Architecture of the HCC-based TM distribution as operationally used for Mapheus 12. The assignment of data-sources and -sinks is done on system level and stored as configuration for persistence. Information about the complete ground system is collected by the HCC monitoring service and from there distributed for display. Changes in the setup are commanded directly to the components and provided to the system configuration from there, if applicable.

It is worth to note that this setup finally corresponds to what has been planned by HCC in the very beginning: using HCC to coordinate the real-time telemetry connection between ground station and mission control centre, while having the actual TM transport out of band using well established and dedicated protocols – SLE in case of satellite telemetry, native UDP in case of Mapheus 12.

Furthermore, the decision to bypass HCC-DT for the telemetry was done to eliminate a small remaining risk. No problems have been encountered during the test campaigns, but there was no way to test the entire HCC based system with all TM receiving stations in situ at Kiruna. From all the testing it is quite likely that HCC-DT could have handled even the small stream segments from all sources without any problems, but it was decided not to take any chances.

Thus, Mapheus 12 finally was launched at 09:25 local time for a flight of 15 minutes reaching an altitude of about 260 km. During the entire campaign, countdown and pre-flight tests as well as the actual flight, the HCC based ground segment worked without any problems. [1]

Already with this first use of the HCC based system, some advantages of the service-oriented design of HCC became obvious. Beside the possibility to slim down the system by replacing a dedicated component by pure configuration, the holistic approach to have one system was literally visible. All engineers with access to HCC had the opportunity to observe all monitoring data of the system. Being distributed all over Esrange site, this was a completely new experience to the personnel, being used so far to have access only to ground monitoring of their local devices.

4.3 Next steps at Moraba

With the success of Mapheus 12, more and more parts of the Moraba ground segment will be included in the HCC based system. The next milestone will be, to add the telecommand part, namely the collection of the commands from all TC sources and the forwarding to the uplink station. With the much lower data rates on the uplink and the TC not coming continuously as a stream, TC transportation will definitely be handled by HCC-DT. Implementations of the necessary components are currently in work and under testing. They are expected to be used operationally for missions in mid of 2023.

In parallel, the issue with the tiny stream segments is further investigated. This includes further testing of HCC-DT under extreme loads with tiny data packets, as well as testing the behaviour of the legacy telemetry processors with data arriving in chunks of many stream segments at once. Based on the results it will be decided, in which phase of the ground segment evolution the TM transportation will be handled by the HCC-DT service. If not possible earlier, it will be realized once the telemetry processor will come in a new version, directly connected to the HCC system. At this point, the Moraba ground segment will benefit from all the advantages HCC offers, similar to the envisioned use-cases for satellite operations.

6. Conclusions

With the first operational usage of an HCC-based system, the project HCC has made a major step forward. Technically, it was proven that the implementations of the HCC basic services work very well, even in a context they have not been explicitly designed for. Conceptually, the power of the service-oriented design and the advantage of the availability of a common infrastructure has been revealed. Thus, the status of HCC has changed from a developing project with prototyping to applications that actually can be used.

Nevertheless, HCC as a complete ground system is far from being finalized. Besides the data transport service, the basic services need to be upgraded to be able to support a large-scale satellite mission with all their needs. And – as importantly – the actual services for space operations as in use at GSOC have to be made HCC-compatible. However, with the success of Mapheus 12, we have a solid foundation now to start and speed up this work.

For Moraba, following the insights of the Mapheus 12 mission, there came up completely new ideas for the further evolution of their ground segment. The further usage of the HCC technology is set and the included principles are widely appreciated.

Acknowledgements

The HCC-team has benefitted a lot from the cooperation with Moraba. The timing of short missions at higher frequency is much better suited for an incremental implementation, compared to much longer-lived satellite projects. That gave us the opportunity to bring some services into operation rather quickly and try them against mission requirements. In the name of the entire team, the author would like to thank all colleagues from Moraba for their great cooperation, and in particular all members of the Mapheus 12 campaign.

In addition, special thanks are dedicated to Udo Häring and Sven Kuhlmann, who are the real creators of HCC. Without them, the project would still be a collection of good ideas – they have made them real by very hard work.

References

[1] DLR, Experimente und ein „Spion“ in Schwerelosigkeit, 21. October 2022
https://www.dlr.de/content/de/artikel/news/2022/04/20221021_experimente-und-ein-spion-in-schwerelosigkeit.html