On-board data analysis and real-time information system

K. Schwenk, D. Herschmann

German Aerospace Center (DLR), German Space Operations Center (GSOC), Münchner Str. 20, 82234 Wessling, Germany

Abstract

The DLR is currently working on an on-board data analysis and real-time information system, for satellites and other autonomous platforms. The system should offer query and alarming services with a query-response time of a few minutes. This is hard to archive today and would beneficial for many earth observation applications. A first prototype of the system has been designed and tested on a flight campaign. In this article we present the current status of the system, the next development steps for bringing the system closer to space, and the obstacles we have to tackle on the way.

1 INTRODUCTION

In classical satellite missions telemetry data, both operational and experimental, is received by assigned ground stations and later forwarded to addressed users via terrestrial networks. The amount of time till data is available for the users can take from several hours to days. This waiting time is even increased as most satellite missions don't have extra relay satellites to continuously send data to ground. The stored data can only be send during the brief contact times to the ground station and therefore is limited. In most cases it is also still unprocessed, so users have to extract relevant information afterwards. Pre-examination directly on-board the satellite is usually not possible.
At the German Aerospace Center (DLR) we are working on a software solution for these challenges. The so called On-board data analysis and real-time information system (ODARIS) is a general framework, where users can access their data directly on-board and receive relevant information in real-time. Besides the systems core functionalities, developers shall be able to easily implement their own application, depending on their specific requirements.

ODARIS has two major intentions:
The first intention is to decrease the latency until information, derived from satellite data, becomes available for the end-user. This is usually important for security and alarming services, but can also help to simplify the general operation of a satellite system.
The second intention is to support on-board data analysis. The capability to directly evaluate sensor data on-board offers tremendous potential to increase the efficiency of system operation and enables new kinds of use-case scenarios. For example, sending already processed product data to the ground, in place of considerably larger sensor data, saves valuable downlink bandwidth." Another key benefit is the option to react on the content of the sensor data directly on-board of the satellite. Especially alarming services, which continuously monitor sensor data, are benefiting from this capability by generatin their event-driven alarm messages more efficiently.
A first implementation of ODARIS was the "Autonomous real-time detection of moving maritime objects" (AMARO) system [1, 2]. It could detect ships via object detection algorithms directly onboard and send relevant information to end-users via the Iridium satellite network at any time. The system was demonstrated on an aircraft campaign in 2018 [3].
As a next consequent step, we are planning to demonstrate the ODARIS concept on one of the next DLR's technology demonstration satellites [4], as part of the Scalable On-board Computing

doi:10.25967/530037

for Space Avionics (ScOSA) flight experiment [5].

In this article we are presenting the current status of the software. This includes a short overview of the concept and its core features as well as the general system architecture. Afterwards we summarize the next necessary development steps for preparing the space experiment demonstration. This includes the planned demonstration scenario and application plus an overview of the obstacles we are currently facing.

2 CONCEPT

The general concept of ODARIS was developed within the AMARO project [2, 3] in the context of an on-board ship detection system.

The system concept is based on several use case scenarios. The most important one is shown in figure 1-4. It illustrates an alarm scenario, where a critical event (in this example a pirate ship) occurs, which can be spotted on the sensor data. The design goal is to notify the user within minutes and provide helpful information. The solution is detection and analysis of the event directly on-board the flight platform, and sending an alarm notification over a *real-time communication channel*, containing a first overview of the current situation. Additionally, as it is not always clear what information can be helpful prior, it should be possible to continuously react to user requests within minutes.

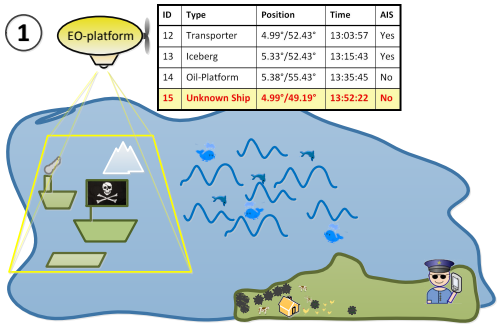

Figure 1: ODARIS user story (part 1) - images are taken by earth observation station and processed immidiatley on-board the EO-platform

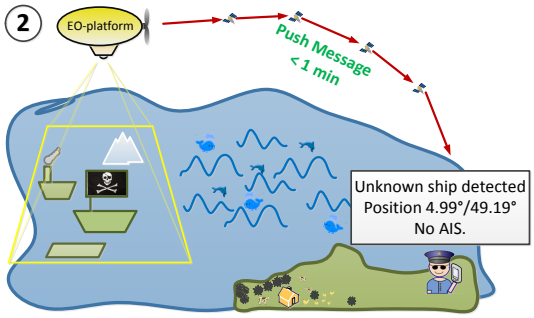

Figure 2: ODARIS user story (part 2) - the user gets informed that an unknown ship has been detected

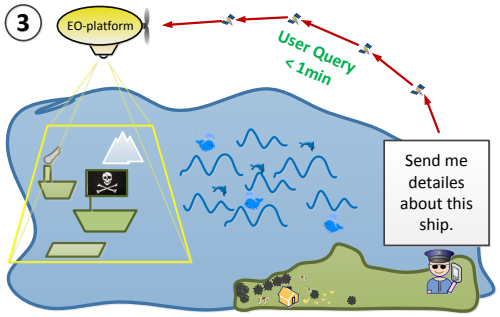

Figure 3: ODARIS user story (part 3) - the user sends a request for detailed information about the unknown ship

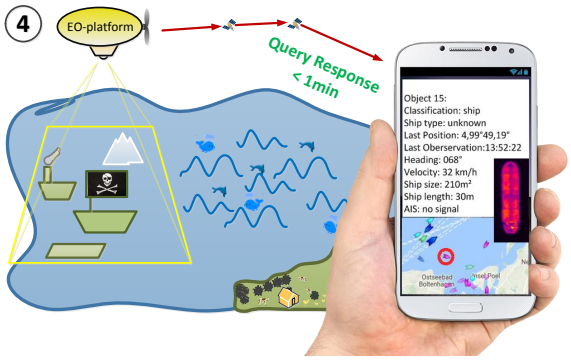

Figure 4: ODARIS user story (part 4) - the user receives detailed information about the unknown ship

Based on the use case scenarios the following system requirements were identified:

- Analyse sensor data and generate product data directly on-board the flight platform
- Provide on-board data access on demand
- Define alarm-events at any time for push notifications
- Limit waiting time for requested information or alarm message to a few minutes
- Implement own user applications without deeper knowledge of ODARIS

Based on these requirements the following core features for ODARIS were derived:

1. Space application service platform:
   User can implement their own application inside ODARIS.
2. Real-time communication capability:
   Information can be send/received worldwide within a few minutes
3. On board data processing capability:
   The on-board computing system provides reasonable computing capability for data analysis applications
4. Support for space- and aircraft-platforms:
   ODARIS can be integrated into several flight platforms

3 SYSTEM ARCHITECTURE

A complete layer model of an ODARIS system is illustrated on figure 5.

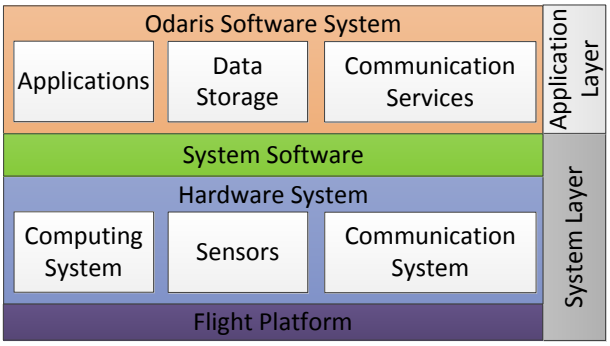

Figure 5: General system layer model of ODARIS

In general a ODARIS system can be divided in two major sections. The system layer, containing the hardware and the system software, usally containg the operating system and potentially additional system management software components. And the application layer, containing the acutal application and its dependencies, like auxiliary programs and software libraries. The ODARIS software system is part of the application layer and contains itself the use-case applications, data storage components and communication services. The specific implementation of the ODARIS system depends on the underlying system layer. Nevertheless one important development aspect of the ODARIS system is keeping most of its components system layer agnostic, so that it can be adapted with minimal modificiations on various on-board computing platforms.
For the AMARO flight-experiment a standard x86-64 Linux computing platform was used, along with an Iridium "Short Burst Data" communication device [6, 7], to send small messages, and an Automatic identification system (AIS) receiver attached [8], to verify detected ships. The specific layer architecture of the AMARO system is presented in figure 6. More details about AMARO can be found in [3] and in the next section.

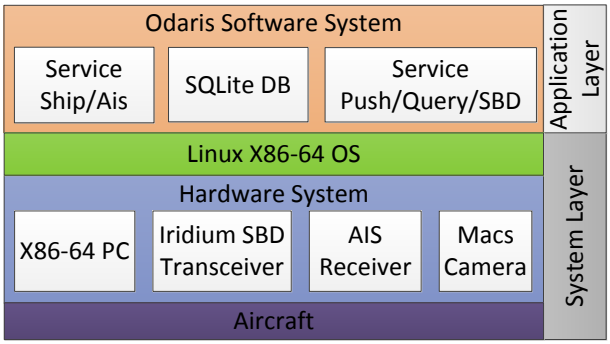

Figure 6: System layer model of ODARIS for the completed AMARO aircraft campaign

For the upcoming space flight experiment, ODARIS software system will be integrated in the ScOSA computing system [5], which has a significantly different system architecture, as the computing system used for AMARO. The layer model of the ScOSA system is shown in figure 7 and important parts of the system will be discussed within the the next chapters.

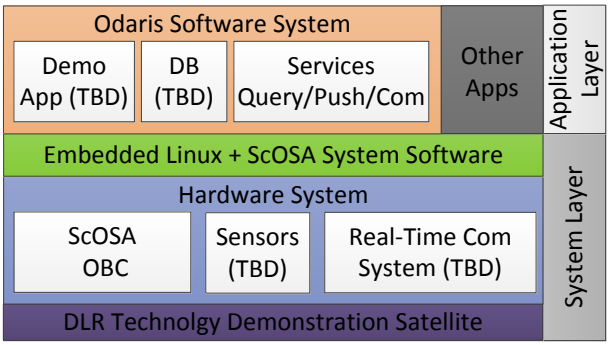

Figure 7: Expected system layer model of ODARIS for the future ScOSA flight experiment

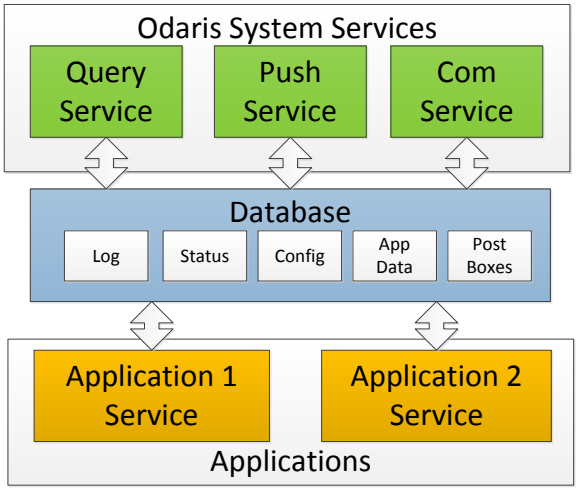

Figure 8: ODARIS service architecture

4 ODARIS SOFTWARE SYSTEM

The ODARIS software is located on the top of the system layer stack, as illustrated in figure 5 and handles data storage, application and communication management.

As described in [2, 3], the AMARO version of ODARIS uses a service-based approach which is illustrated in figure 8. The system contains of several independent service applications, each performing a specific task. To enable dynamic random data access, ODARIS uses a Structured Query Language (SQL)-database for data storage [9], including product data generated by the applications, logging data, service status information and configuration settings. The database is also used for internal communication by using messages via specific tables, that are tracked by the services. All services can independently access the database. A big advantage of using a database solution is the language SQL itself. It's an extremely versatile and domain-specific language for performing data base accesses and searching datasets. Additionally, it is also widely used and most of the application developers are already familiar with.

To enable data access as well as provide event-driven alarm-functionality, the dedicated services Query and Push are implemented.
Required data can be accessed via the Query service by sending requests as SQL-queries to ODARIS. Together with the extended search capabilities of SQL, the required product data can be specified on demand. For Example, the use case scenario presented on figure 3, requesting a list of all objects detected within a reasonable timespan in the region of interest. Furthermore, by modifying database entries, the service can also be used to reconfigure the system. This is important for system management, for example (de-)activating application and inserting/deleting (new) alarming events.
Via the Push service (alarming) events can be defined, which generate user-notifications automatically. For Example, in the use case scenario presented on figure 2, an alarming message should be generated, when a larger ship is detected, that usually has to provide AIS information but is not sending them. These events are defined as SQL-query expressions and will be periodically evaluated. If a critical condition is detected, a notification message will be created. Considering that these events are database entries, they can also be added, deleted and modified using the Query service on runtime.
The communication service is handling the external communication of ODARIS, by utilizing the available communication channels of the flight platform. The communication service is offering a high-level application communication inter-

face, hiding details of the platform specific communication solution from the applications. This allows developers to implement portable applications, independent of the communication solution of the flight platform. Having a high-level communication user interface, also simplifies the development process as the application developer has not to worry about the implementation details of the communication system.
The communication service application interface is comparable to an email client, offering output-/input- postboxes for sending and receiving messages. These postboxes are implemented as database tables, which can be accessed by the application services and other system services. The communication service checks the output-postbox regularly and tries to send these messages over the available communication device. If a message is received from a communication channel, the communication service stores it inside the input-postbox. At the moment, the input-postbox is regularly checked by every service, which can receive messages. The incoming messages will be fetched and further processed by the addressed service.
The communication service decouples the process of sending and receiving messages over the communication channels, form the acquisition and generation of messages by application and system services. The message service also controls the order of message transmission, based on priority levels assigned to the messages. These are the two key enabling features, allowing different services concurrently utilizing the communication system, without disturbing other services and blocking the transmission of important messages.

5 ON-BOARD COMPUTING PLATFORM

In this section a brief overview of the ScOSA system, as the next targeted on-board computing platform for ODARIS, will be presented. A far more detailed description can be found in [5, 10]. During the former AMARO mission, ODARIS was running on a standard x86-64 Linux operation system [3]. The resources to implement a full-featured Linux Operating System (OS) will not be available on the upcoming DLR technology demonstration satellite. For the space experiment, ODARIS will be integrated into the ScOSA system as shown in figure 7 and will be used to demonstrate the systems real-time performance capability.
The ScOSA system is a next generation computing on-board platform developed as an DLR wide endeavour. Its main focus relies on reliability, performance, usability and cost-efficiency. It is a distributed computing system, consisting of several independent computing nodes, connected by a space-wire network [5, 10]. There are different kinds of computing nodes available, each serving a different task:

- High Performance Nodes for executing computing tasks
- Reliable nodes meant for supervision of the system and controlling Fault detection, isolation, and recovery (FDIR)
- Interface Nodes meant for attaching to external system components as the satellite bus and sensors

A scheme of the ScOSA distributed platform is shown in figure 9 as well as an image of the development board we are currently using for prototyping in figure 10.

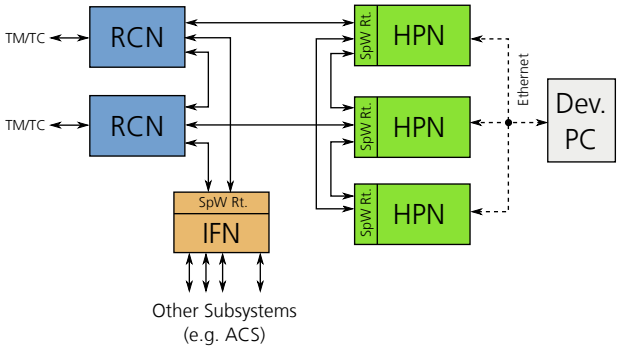

Figure 9: Block diagram of the ScOSA system (source [5])

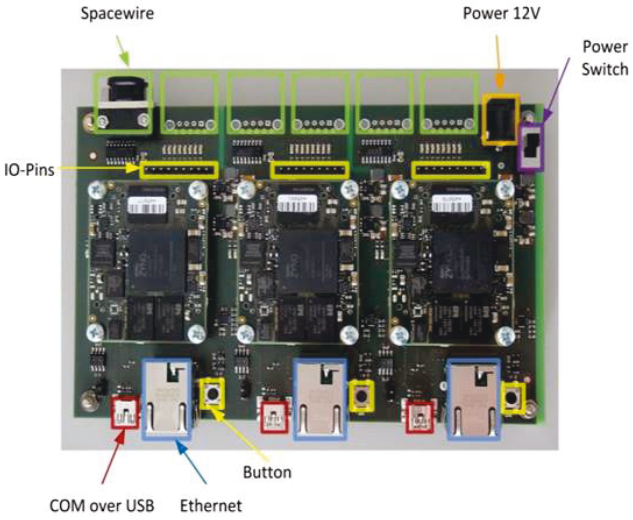

Figure 10: ScOSA development board containing three High Performance Node (HPN)

The HPNs are currently based on the programmable System on a chip (SOC) Xilinx-Zynq Z7020 device, which has a dual core ARMCortex-A9@1 GHz combined with a Field Programmable Gate Array (FPGA) [11]. For more detailed specification see [5].

Each node is an unique computing device, running either an embedded Linux or an Real-Time Executive for Multiprocessor Systems (RTEMS) OS, depending on the use case. The ScOSA computing platform at a whole is composed of the ScOSA system software running on top of the OS. From the Operating System's perspective, the scosa software is a system process, containing management services and the different user applications (e.g. ODARIS).
The management services are responsible for initialization and (re-)configuration the complete system, handling (inter-)node communication, providing FDIR layer and task management.
User applications have to be integrated via a provided data-flow programming API. A scheme for illustrating this programming pattern is shown in figure 11. The data-flow programming scheme was chosen, as it naturally reflects the distributed hardware architecture of the system. An application can be divided into sub tasks, that can be executed on different HPNs. In case of a node failure, they can be migrated to other functional nodes. More details about application development for the ScOSA system can be found in [12].

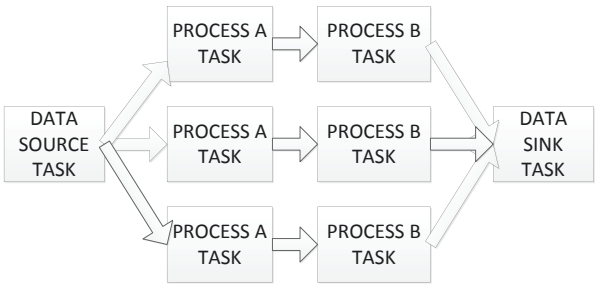

Figure 11: Illustration of data-flow driven application, using parallel blocks of execution

6 REAL-TIME COMMUNICATION SYSTEM

The primary sub component for achieving data access from ODARIS within a few minutes is the communication system. For the AMARO flight experiment presented in [3] we have chosen the Iridium SBD service [7]. This allowed us to send and receive short text message with a size of around 300 bytes, independently of the aircraft's location. The data latency study which can be viewed on figure 12 revealed that for 84% of the messages the average query-response latency was around 1.87 minutes. Therefore basic real-time communication capability can be achieved for aircraft flight platforms.

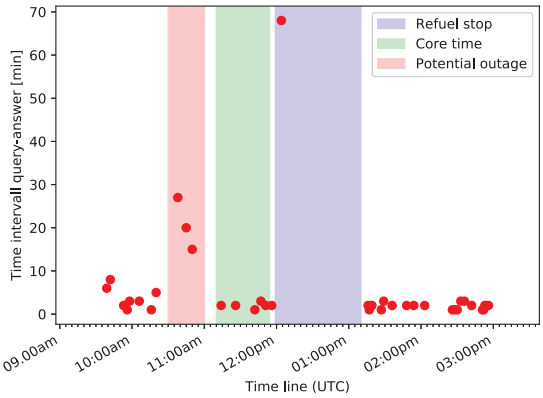

Figure 12: Query-answer delay statistic, of the AMARO flight campaign (source [3])

For space flight platforms this is a different story. We still want to use one of the classical telecommunication-satellite constellations, to preserve real-time communication, that can be addressed worldwide at any time. However a

theoretical study [13] revealed, that there are severe limitation factors.
The most important one is, that the satellite must be in line of sight with one of the telecommunication satellites.
The other limitation is the Doppler frequency shift. For satellite-to-satellite communication the $f_{doppler} = \frac{V_{rel}}{c} * f_{Carrier}$ frequency shift can be significantly higher than for the specified ground-to-satellite communication use case, due to much higher relative velocities between the sender and receiver devices.
In figure 13, example use-cases analysing the theoretical communication capability between the Iridium [7] and Orbcomm [14] constellations and some client satellites are presented. The results indicate, that the communication using these telecommunication providers, at best, would be very limited. But another study, that mapped the performance of the Globalstar network [14] utilizing three CubeSats, showed far better results [15]. At the moment we are still planning to use satellite communication services, and consider solutions in the Cubesat market like [16, 17] as reasonable options to demonstrate the real-time capability concept of ODARIS.

Iridium	ISS	BIRD	CAMP	CAMP Mod.
Contact	9%	1%	0%	35%
Duration	1:56	2:24	0:00	10:17
Gap	0:20:00	3:20:00	-	0:19:00

Orbcomm	ISS	BIRD	CAMP	CAMP Mod.
Contact	10%	1%	3%	no analysis
Duration	5:22	0:45	2:07	no analysis
Gap	0:22:20	1:10:00	0:50:00	no analysis

Figure 13: 24h study, showing average contact time, average contact duration and average contact gaps between two satellite contacts (source [13]), CAMP mod is CHAMP orbit with inverted flight direction (Right Ascension of Ascending Node (RAAN) 180 degree shift)

7 REFERENCE APPLICATION

To give users an overview of the systems capabilities, a reference application will be implemented. This will also serve as technology demonstrator for ODARIS within a space environment during the ScOSA flight experiment [5].
The starting point will be the former ship detection application from AMARO [1]. To increase the outcome of the flight experiment we don't only want to reuse the former algorithm, but also enhance it by including most recent research on satellite autonomy at DLR. This approach also demonstrates the usage of the ODARIS Framework from a user's perspective. As our focus is only on the higher-level application software, we plan to implement one of today's well established machine learning algorithms. This modern approach shall improve the results of object detection within ODARIS along with reducing unnecessary data processing by pre-filtering unusable pictures (e.g. cloud coverage).
A new and promising approach for image processing is the implementation of a deep neural network. Therefore, we currently focus our research on identifying suitable algorithms in the field of convolutional neural networks.

8 ODARIS STATUS SUMMARY

In this section a brief summary of the current status of ODARIS itself as well as a demonstrator scenario for testing the software is presented.
A first version of ODARIS is executable and provides most of the targeted features. The single services run in parallel as dedicated processes within a Linux environment. Their status (running, frozen, stopped) can be tracked on Operating System level.
At the moment, we are finishing the migration of ODARIS to an armv7-architecture as the official test and future mission environment. Yet we still a x86-architecture as our development environment.
The on-board data processing within ODARIS currently includes the reception of user queries, extraction of requested informations from a SQL-database and responding to the user. Furthermore, a push notification service is running in the background to directly send specific information as answer messages on predefined events.
To find a suitable concept for the real time communication system a detailed background study for possible satellite-telecommunication networks was performed.
As demonstration application within ODARIS the ship detection service from AMARO is currently

still running.
For further prototyping we defined a complete demonstration scenario to test our system as a whole. The demonstrator scenario is split into 2 parts, Application and User query. For the application part we still use ship detection as use case scenario. A camera will be simulated from outside the ODARIS environment and a fixed set of labelled test images will be transmitted to the ODARIS system via Ethernet. Afterwards the ship detection service stores all results inside a SQL-database. The result, e.g. number of detected ships on an image, can be compared with the corresponding test images. Therefore, we can comprehend the complete data processing chain within ODARIS from receiving data to storing information to the corresponding tables in the database.
In the second part we confirm the correct processing of user queries. As we don't have access to a satellite telecommunication-network on ground at the moment, we bypass the Communication Service of ODARIS and place direct user queries into the Query service. Afterwards we can check if the correct answer message would have been sent via the Communication Service. During this demonstration part we test different kinds of user queries to capture the most important situations and recognize undefined behavior.

9 CONCLUSION AND WAY FORWARD

We are facing three major challenges while preparing the on-board experiment version of ODARIS.
The first challenge is providing a highly available on-board real-time communication system with low latency in a space environment. For the former AMARO mission on an aircraft, we used the SBD-service of the Iridium satellite-telecommunication network [7]. This service is capable to send and receive short text messages at any time, and with a turn-around delay averaging three minutes. For the space experiment, we want to use the same or a similar satellite-telecommunication system. But as these services are designed for terrestrial usage it is yet not clear if they can be used in space with the same performance. Furthermore, the orbit of the flight experiment of the next DLR satellite, where ScOSA with the ODARIS experiment should be demonstrated, is not defined at this moment, which determines the possible contact coverage to the network. Therefore, the communication performance could also be limited.
The second challenge is the integration of ODARIS on a space-compatible on-board computing platform. For the aircraft campaign version we used a small-sized native x86-64 Linux OS, but such a powerful computing platform will not be available for the space mission. Also the ODARIS software will be adapted for being compliant with space quality and operational standards.
At last the planned reference application within ODARIS will potentially face similar problems. Machine learning algorithms, and on-board image processing in general, needs a lot more resources than classic satellite system software usually provides. Additionally, ODARIS will not be the only experiment running during the upcoming ScOSA flight experiment. Therefore, the realisation of the planned reference application will depend on the available resources.
Besides the major challenges we focus on the current general basic version of ODARIS and enhance it within the next 2 years to realize our vision. The integration into the flight experiment will be performed in parallel after the necessary mission requirements are fixed.

References

[1] Schwenk, K., Willburger, K., and Pless, S., "Amaro-autonomous real-time detection of moving maritime objects: introducing a flight experiment for an on-board ship detection system," in [*Earth Resources and Environmental Remote Sensing/GIS Applications VIII*], Michel, U. and Schulz, K., eds., **10428**, 71 – 81, International Society for Optics and Photonics, SPIE (2017).

[2] Schwenk, K., Willburger, K., and Pless, S., "A prototype on-board ship-detection system," in [*Deutscher Luft- und Raumfahrtkongress (DLRK) 2017*], *DGLR*

BERICHTE , Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., Bonn (2017).

[3] Willburger, K., Schwenk, K., and Brauchle, J., "Amaro—an on-board ship detection and real-time information system," *Sensors* **20**(5) (2020).

[4] Kottmeier, S., Hobbie, C. F., Orlowski-Feldhusen, F., Nohka, F., Delovski, T., Morfill, G., Grillmayer, L., Philpot, C., and Müller, H., "The eu:cropis assembly, integration and verification campaigns: Building the first dlr compact satellite," in [*International Astronautical Congress IAC 2018*], *69th International Astronautical Congress* (2018).

[5] Treudler, C. J., Benninghoff, H., Borchers, K., Brunner, B., Cremer, J., Dumke, M., Gärtner, T., Höflinger, K. J., Lüdtke, D., Peng, T., Risse, E.-A., Schwenk, K., Stelzer, M., Ulmer, M., Vellas, S., and Westerdorff, K., "Scosa - scalable on-board computing for space avionics," in [*International Astronautical Congress IAC 2018*], *Proceedings of the International Astronautical Congress* (2018).

[6] Wireless Innovation Ltd., "Michroburst - out of the box ready sbd modem - https://www.wireless-innovation.co.uk/product/wiltd-michroburst/," (2020).

[7] Iridium Communications Inc., "Iridium short burst data (sbd) - https://www.iridium.com/services/details/iridium-sbd," (2020).

[8] U.S. Department of Homeland Security, "Navigation center of excellence - ais messages - https://www.navcen.uscg.gov/?pageName=AISMessages," (2020).

[9] SQLite Consortium, "Sqlite - https://www.sqlite.org/," (2020).

[10] Lüdtke, D., Westerdorff, K., Stohlmann, K., Börner, A., Maibaum, O., Peng, T., Weps, B., Fey, G., and Gerndt, A., "Obc-ng: Towards a reconfigurable on-board computing architecture for spacecraft," in [*2014 IEEE Aerospace Conference*], (2014).

[11] Xilinx Inc., "Zynq-7000 soc product description - https://www.xilinx.com/products/silicon-devices/soc/zynq-7000.html," (2020).

[12] Schwenk, K., Ulmer, M., and Peng, T., "Scosa: application development for a high-performance space qualified onboard computing platform," in [*Proc. SPIE 10792, High-Performance Computing in Geoscience and Remote Sensing VIII*], *Proceedings of SPIE* **VIII**, SPIE Remote Sensing (2018).

[13] Schönegger, S., "Investigation of leo communication networks for space telemetry and commanding," *diploma thesis - Fachhochschule Technikum Wien and DLR* (2004).

[14] Orbcomm Inc., "Orbcomm website - https://www.orbcomm.com," (2020).

[15] Voss, H. D., F.Dailey, J., Orvis, M. B., White, A. J., and Brandle, S., "Globalstar link: From reentry altitude and beyond," in [*2016 Small Satellite conference*], (2016).

[16] Pumpkinspace Inc., "Nsl_eyestar communication modem - https://www.pumpkinspace.com/store/p200/NSL_EyeStar-D2_Duplex_Globalstar_Communication_System.html," (2020).

[17] Santangelo, A. D. and Skentzos, P., "Utilizing the globalstar network for satellite communications in low earth orbit," in [*54th AIAA Aerospace Sciences Meeting, San Diego, California, USA*], (2016).