The ESO Astronomical Site Monitor upgrade

Gianluca Chiozzi[1a], Heiko Sommer[a], Marc Sarazin[a], Thomas Bierwirth[a], Dario Dorigo[a], Ignacio Vera Sequeiros[a], Julio Navarrete[b], Diego Del Valle[b]
[a]European Southern Observatory, Karl-Schwarzschild-Strasse 2, D-85748 Garching bei München.
[b]European Southern Observatory, Paranal Observatory, Chile.

ABSTRACT

Monitoring and prediction of astronomical observing conditions are essential for planning and optimizing observations. For this purpose, ESO, in the 90s, developed the concept of an Astronomical Site Monitor (ASM), as a facility fully integrated in the operations of the VLT observatory[1].
Identical systems were installed at Paranal and La Silla, providing comprehensive local weather information.
By now, we had very good reasons for a major upgrade:

- The need of introducing new features to satisfy the requirements of observing with the Adaptive Optics Facility and to benefit other Adaptive Optics systems.
- Managing hardware and software obsolescence.
- Making the system more maintainable and expandable by integrating off-the-shelf hardware solutions.

The new ASM integrates:

- A new Differential Image Motion Monitor (DIMM) paired with a Multi Aperture Scintillation Sensor (MASS) to measure the vertical distribution of turbulence in the high atmosphere and its characteristic velocity.
- A new SLOpe Detection And Ranging (SLODAR) telescope, for measuring the altitude and intensity of turbulent layers in the low atmosphere.
- A water vapour radiometer to monitor the water vapour content of the atmosphere.
- The old weather tower, which is being refurbished with new sensors.

The telescopes and the devices integrated are commercial products and we have used as much as possible the control system from the vendors.
The existing external interfaces, based on the VLT standards, have been maintained for full backward compatibility.
All data produced by the system are directly fed in real time into a relational database.
A completely new web-based display replaces the obsolete plots based on HP-UX RTAP.
We analyse here the architectural and technological choices and discuss the motivations and trade-offs.

Keywords: ESO, VLT, ASM, weather, seeing, site monitor, robotic system, telescope control, web-based user interface, relational database, Javascript, Angular

1. INTRODUCTION

As early as VLT first light, back in 1998, the Paranal ASM has proven to be a helpful tool for assessing from the control room the performance of telescopes and instrument (scientific image quality) against the variability of atmospheric observing conditions.

The weather and environmental parameters provided by the Astronomical Site Monitors at all ESO observatories are indispensable during the whole lifecycle of observations, from the scheduling to the final analysis of the data.

This data is used in many different ways, such as:

- by astronomers and telescope operators to decide the scheduling of observations and to optimize observation parameters; at the moment the system is not providing an official prediction for the seeing, but personnel at the

[1] gchiozzi@eso.org ; phone +49 89 32006-543.

observatory use the available data to make their own short term predictions regarding the seeing/wind/transparency to schedule the observations. Official predictions will be made available in the future;

- by the control systems of telescopes and instruments to automatically adjust internal parameters, for example for astronomical tracking corrections or for temperature control;
- by scientists, as integral part of the FITS files containing the scientific frames, to access the environmental conditions during each exposure for data analysis and quality control;
- by scientists interested in analyzing historical weather trends and in elaborating weather predictions, retrieving the historical data from the long-term ESO archive. Short and long term weather predictions are essential to make automatic scheduling of observations efficient.

The identical systems operational in Paranal and La Silla since the 90's were providing:

- atmospheric and environmental parameters, such as wind, humidity, temperature and dust particle content at different elevations, measured with sensors installed on a meteorological tower located at a convenient position near the telescope enclosures;
- seeing measurement estimated by a DIMM[2]: a 35-cm telescope with a technical CCD detector mounted at its Cassegrain focus, located on a 5 to 6 meter high tower with enclosure.

Since the expected improvement in the science image quality brought by the new VLT Adaptive Optics Facility[3] (now under commissioning) will be strongly dependent on the vertical distribution of the atmospheric turbulence, a priori knowledge of this parameter is a requirement to schedule AOF observations efficiently.

ESO has therefore decided to add two dedicated instruments to fulfill this task:

- a Multi Aperture Scintillation Sensor (MASS)[4] to measure the vertical distribution of turbulence in the high atmosphere and its characteristic velocity by analyzing the scintillation of bright stars;
- a SLOpe Detection And Ranging (SLODAR)[5] telescope, for measuring the altitude and intensity of turbulent layers in the low atmosphere by means of a triangulation method, in which the turbulence profile is recovered from observations of bright binary stars using a Shack-Hartmann wavefront sensor.

It was also decided to fully integrate in the ASM system the Paranal Water Vapour Monitor[6], capable of providing a measure of the water vapour content of the atmosphere with high precision and time resolution, allowing the execution of infrared observations in periods of low precipitable water vapour. This instrument also provides infrared sky brightness measurements and other data, all integrated into a commercial LHATPRO instrument.

The ASM Upgrade project was started in 2013 with these main objectives.

At the same time, after more than 10 years of operation, it was also the right occasion to address the hardware and software obsolescence issues of the system. The ASM control workstation in particular was still a very old HP running the HP-UX operating system and using the proprietary RTAP SCADA system that had been at the core of the very early releases of the VLT Common Software. The main reason for not having upgraded earlier the ASM software was the extensive usage of the RTAP-specific plotting features to produce the ASM display panels in the control room.

The dependency on this legacy hardware and software was imposing operational limitations, allowing only for a small number of ASM displays in the control room, and not allowing convenient access to the current weather data from outside of Paranal. At the same time, it was not possible to integrate historical data (more than 24 hours) with the live ASM information in the control room.

As it will be described in detail in the next sections, the new Paranal ASM:

- has its foundations in the VLT Common Software platform. This provides a solid infrastructure and allows us to ensure full backward compatibility with the various systems accessing the ASM;
- integrates as much as possible commercial off-the-shelf instruments and devices, with the minimal possible changes in order to minimize development time and maintenance, and to be able to retrofit them easily when the vendors provide new features;
- uses recent mainstream technologies, programming languages and libraries, with a particular attention to what is being selected for the E-ELT. This aims at ensuring rapid development, longer lifetime and easier maintenance.

Commissioning of the new system was completed in March 2016 and since then it is providing official weather and environment data to the observatory.

2. SYSTEM OVERVIEW

The main requirement of an ASM is to be completely robotic and to require a minimum of maintenance so as to reduce the overhead on the observatory staff. Indeed the Paranal ASM DIMM had been operating over the years in wind conditions up to 18m/s with extremely low downtime, mostly used to exchange the DIMM CCD detector which had been purposely chosen identical to the technical CCDs installed at the twelve VLT foci.

The challenge in the ASM upgrade was thus to increase the system complexity, as needed to satisfy the new requirements, while maintaining robust robotic operational characteristics.

For this reason it was decided to install the MASS-DIMM and SLODAR instruments on identical commercial systems, for what concerns mount (Astelco NTM500) and enclosure (Halfmann). The SLODAR optical tube assembly is an optimized Dall-Kirkham design manufactured by Orion Optics, UK, while the MASS-DIMM telescope is a Celestron C11 (280mm diameter, f/10) with carbon fiber tube and modified primary mirror cell. The position of the assemblies on the Paranal platform was selected in order to minimize the disturbances produced by the other telescopes (in particular the VLT Unit Telescopes and VST). This resulted in data significantly more consistent with that observed by the astronomical instruments on the telescopes for the new DIMM with respect to the old one.

The MASS-DIMM instrument was developed jointly at Sternberg Institute (Moscow, Russia) and Cerro Tololo Observatory (La Serena, Chile) [4]. More than 30 devices are in use around the world. MASS-DIMM was the basis for the TMT and E-ELT site testing campaigns and is routinely operated at several observatories to support science operation.

The center for Advanced Instrumentation (CFAI, Durham University, UK) developed the SLODAR technique for characterization of the vertical profile of atmospheric optical turbulence [5]. SLODAR systems, based on small telescopes (typically 50cm aperture), have been employed for characterization of the optical turbulence at Paranal, ORM (La Palma), Mauna Kea and SAAO observatories. These studies have mainly concentrated on characterization of the 'ground layer' of turbulence in the first kilometer above the site, relevant to the development and application of ground-layer and multi-conjugate adaptive optics systems.

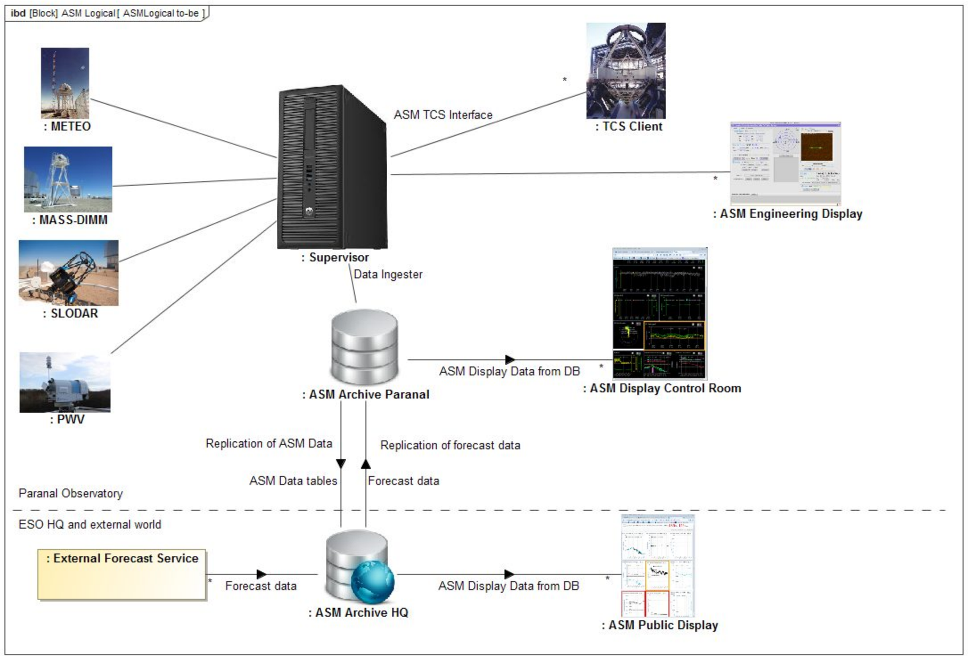

Figure 1. ASM system architecture

While the MASS-DIMM and SLODAR subsystems are operational only during night time, as they rely on the observation of stars, the meteo tower and water vapour radiometer are in operation 24/7.

In the old ASM, the control system was responsible both for collecting the data and providing the display of the weather conditions in the control room. Once a day, during day-time, part of the data was archived in a relational database for offline usage. The data available in the control room was limited to the last 12 or 24 hours.

For the new system (see the system architecture in Figure 1) we have decided to completely decouple data display, retrieval and analysis from data collection. The control system (see section 3) is responsible for collecting the data and sending it directly to a relational database. Any application needing the data, including the site monitor display in the control room, gets the data directly from the relational database. Data replication allows (as described in section 5 below) access both in Paranal and in near real time also in Garching with appropriate quality of service characteristics.

We also integrate into the database forecast data provided by external services. The European Center for Medium Range Forecasts' (**ECMWF) data cubes** that currently arrive at ESO consist of 2 data streams of up to 9 meteorological variables in total, at 0.125deg horizontal resolution for 25 vertical levels

1. *ESS* (**Boundary conditions** data; hourly forecasts from main-6-hourly base times, out to T+90hrs)
2. *ESD* (**Operational model** data; 3-hourly forecasts from main-12-hourly base times, from T+90 to T+192hrs)

The main application accessing data from the relational database is the weather display (see section 6) implemented using web technology. This allows to offload the CPU needed for generating the graphs to the web browsers running on the clients, while the data is retrieved from the database and not any more from the control workstation. In this way we do not have any more the constraints of the old system on the number of consoles that are allowed to display weather data. Moreover, if very complex graphs, with a lot of historical data, are requested by users, only the client web browser will struggle and the rest of the system will not be affected.

In parallel to the stream of data going to the relational database, the ASM control system provides also real time data access to telescopes and instruments, using the same interfaces that were provided by the old system, for which we have maintained full backward compatibility.

3. CONTROL SYSTEM

The control system of the old ASM was heavily based on the classical VLT architecture:

- devices were connected to IO boards on VME-based Local Control Units (LCUs), with the code primarily developed in ANSI C;
- the DIMM instrument was using a VLT Technical CCD camera, developed in-house by ESO;
- the mount of the telescope was controlled by the same tracking software as used on the VLT Unit Telescopes, modified to support the specific hardware;
- an HP Unix workstation (obsolete since years) was running the supervisory software, written primarily in C++ using the VLTSW framework.

The new control system does completely without LCUs, as there is no need any more of connecting devices to IO boards, and there are no strict real time control requirements. Instead, MASS-DIMM and SLODAR have their own Linux workstations for the control, and there is a central supervisory Linux workstation to coordinate the operation of all instruments:

- the same Vaisala meteorological station delivers local weather data, but its serial line has been connected to a Serial-2-Ethernet converter and the data gets collected directly by the supervisory workstation using a simple socket-based application written in Java. This frees us also from the constraints of serial cable length. Given the available Ethernet bandwidth, relaying serial connections over Ethernet is a very practical solution and our new configuration is working since almost 2 years without any problem;
- the two telescope mounts are driven by the standard controller developed by the vendor. This Linux-based system implements tracking in sky coordinates, with pointing model corrections. The tracking performance is more than sufficient for our purposes, coupled with basic auto-guiding implemented by both the SLODAR and

MASS-DIMM supervisors. In this way we removed the VLT LCU tracking software. The communication with the mount controller is as well via Ethernet, using a simple socket communication protocol;
- instead of VLT Technical CCDs, the DIMM uses a commercial GigE Prosilica camera, connected via Ethernet and implementing standard GigE interfaces. This camera, operating at a frame rate of 100Hz and 5ms exposure time, fulfills our requirements and we do not need the extra features of the VLT TCCD;
- the MASS instrument is connected to its control workstation through an RS485-2-USB cable. This imposes a proximity between the instrument and the workstation (of the order of 10 meters), but the vendor is planning to implement in the future an Ethernet interface that would allow us to remove also this constraint.

Having removed the LCUs and the TCCD improves a lot our maintainability and resilience to obsolescence issues and reduces significantly the cost of the system.

We also phased out almost all C and C++ code, leaving only, for backward compatibility, the access library provided to the VLT telescopes for accessing the ASM data.

Our new control supervisory code has been implemented mostly in Java and Python, leveraging the large amount of high quality libraries available. We think that in this way the code will be much more maintainable and extensible.

For what concerns the low level control for the SLODAR and the MASS-DIMM instruments, we are using the original code delivered by the respective developers. In this way we profit from the synergies with the other users of those systems. Still, we have requested to have full source code and we have analyzed and reviewed the code base, to ensure that we can eventually maintain it and that it satisfies our basic coding requirements. We have actually implemented ourselves or requested features/changes that have been added to the common code base. The only drawback is that in order to reuse as much as possible of the MASS-DIMM vendor operational procedures, implemented in Tcl, we have extended their original code. Even though Tcl is a fading language, this is not a major problem for us, since we have a lot of experience in ESO, where it is still widely used in the VLT. Nevertheless, it might have been better to re-implement completely this part in Python and we will consider this option for a future upgrade.

4. DATA PROCESSING

We largely separated into dedicated processes the part of the control system that distributes and manipulates the data coming from the various ASM subsystems. Also here it was necessary to maintain full interface compatibility and to integrate the applications into the existing lifecycle and communication infrastructure.

The project had decided early to feed the new ASM display from a relational database instead of accessing buffered data in the control system. We no longer could use the VLT's general batch data ingestion framework with its unpredictable delays. We still use the existing PAF (Parameter File) FITS file format[11], though, for intermediate data representation. Such data files are now produced by each subsystem's data supervisor. In the absence of modern data flow middleware in the VLT control system, the data files get copied locally or remotely (rsync) to central data ingestion folders, where they get immediately picked up by dedicated database ingestion processes.

As a novelty in the VLT, the database defines the mappings of data file keywords to database tables and columns in a generic way, to allow adding keywords without changing code. On startup, the ingestion processes read out database meta tables to learn about the current mappings, which they then apply as data comes in.

The total time from creation of a data file through copying and ingestion to the relational database is typically less than one second.

The ASM subsystems deliver their data at different levels of complexity:

- SLODAR produces ready-to-use PAF data files in the black-box vendor's software;
- METEO publishes custom ASCII datagrams that we read from a socket connection;
- LHATPRO publishes custom data files for us to poll from an FTP server;
- MASS-DIMM uses for each detector a two-staged pipeline, for which the data supervisors follow data as it gets appended to files, mix output of different pipelines, and use stateful parsers to identify new data blocks in the continuous streams of new data lines.

The data supervisors not only reformat and aggregate device data, but also perform data scaling to different units, and compute derived data items (for example the METEO wind speed is used for the calculation of atmospheric turbulence parameters).

In addition to processing the data for ingestion in the relational database, the supervisors also publish all data in the hierarchical VLT online database. There, data is tagged with a timestamp, from which quality flags can be automatically derived. The VLT Telescope Control Software picks up ASM data from the online database, both for its own purposes and to make the ASM data available to the instruments.

Automated operation of the ASM requires a robust data flow. Any of its stages can be restarted independently of the others, or will wait if started too early. No data can ever be lost, thanks to buffering in the file system or the online database. Re-processing of old data after restart is recognized and suppressed. Data files get deleted only after the ingestion of their content in the relational database has been verified independently.

All data flow applications are implemented in Java, because of better support for multithreading, database connection pooling (c3p0), parsing, and many other tasks of data flow. To fully integrate into the VLT software environment, we wrapped the C API of the VLT common software using generated Java binding code, which is much more versatile and easier to maintain than the hand-written JNI code that already existed. We ran the tool jnaerator on the VLTSW header files, using the BridJ Java-C binding target; the alternative binding technology JNA is more widely used, but BridJ we found clearer in the mapping of complex pointer constructs. As we use multithreading in the Java layer above a single-threaded VLTSW, all Java-to-C calls run asynchronously through a thread pool of size one, which could easily be extended to more threads for native calls in the future. The binding at runtime has worked flawlessly, even in specialized cases such as subscribing to weather alarms, or when filling large tables in the online database. We also use BridJ to wrap the slalib astronomical routines for Java.

Other 3rd party tools that worked well for the dataflow software were ftp4j for in-memory FTP downloads, and JNotify for file system event notification with JDKs < 1.8.

5. DATABASE

The ASM database contains the atmospheric information for ESO astronomical sites. The database has been created in the existing ESO operational data flow system, which consists of a local database server in the observatories and a central database server at ESO Headquarters in Garching. Synchronization of the databases is achieved by means of database replication technology (see Figure 1).

The commercial technology used in the operational architecture is based on SAP ASE for the relational database system and SAP replication server for data replication. We have been using this technology at ESO for more than 20 years and it has been proved to be stable and reliable throughout the lifetime of the observatory.
The distributed architecture of the ESO dataflow system allows fast and reliable access to the data from the observatory, protecting them from connectivity outages from headquarters. The database in headquarters contains the full history of the observatories, while the database in the observatories can be kept small, therefore improving the recovery time of the local database if needed.
The average data rate measured when all instruments are in operation is around 3kB per minute and the total amount of data is of the order of 2.5MB per day. The data rate is supported by the current infrastructure and therefore no extra hardware was required for the implementation of the project.
As mentioned in the data processing section, the process that ingests data in the database uses some configuration tables in the database itself to identify the data destination table and column. In this way, any new parameter can be easily added by introducing a new database table or database column and adding the proper configuration in the database. These configuration tables are used as well by the web display to decide which parameters can be plotted in the tool. Therefore, making a new instrument or parameter available for plotting using the web interface is a straight forward operation and does not require any code change.
Forecast data retrieved from external services are inserted into the database at ESO headquarters using dedicated scripts and are added to the configuration tables. In this way they can be used in the ASM web display. Such data are replicated from headquarters to Paranal to be available at the observatory as well.
In order to allow the display of large historical time ranges, database tables containing atmospheric information are down-sampled into hourly, daily or weekly averages, populated periodically by an external process.

The data stored in the ASM database can also be queried and retrieved directly using standardized query forms, common to all ESO observatories. The Ambient Condition Database query forms are publicly available at the following URL:

http://archive.eso.org/cms/eso-data/ambient-conditions/paranal-ambient-query-forms.html

The corresponding http requests can be used also to programmatically retrieve data.

6. WEB DISPLAY

Starting from the assumption that the measured data points can be queried in near real-time from a relational database, the software architecture and technology choices for the ASM web display were determined by the following drivers:

Easy access from anywhere – As opposed to the legacy solution, live updating plots should be easily accessible not only from within the Paranal control room, but also from stakeholders worldwide. Consequently we decided for a web-based solution using modern browsers (Chrome, Firefox, Safari) as visualization platform driven by JavaScript.

Off-the-shelf charting/plotting support – Some of the ASM visualization requirements were fairly specific and demanding. In particular we had the requirement of reproducing exactly the same display layout in the control room, with the same features, as the existing application implemented with the RTAP SCADA system. We initially considered implementing the graphical visualization using the fascinatingly flexible JavaScript library d3.js[7]. However, after a period of prototyping charting and plotting with d3.js, we realized that it was too low-level for our needs and we were spending too much time reinventing what others had already gone through successfully, such as drawing multiple x or y axes, positioning linear or logarithmic tick marks or supporting interactivity such as zooming. We eventually decided for the dedicated charting solution highcharts.js[8] and its companion library highstock.js, providing a very nice solution for exploring and progressively zooming into years of historic data. We found these libraries to be extremely feature-rich, well-documented and supported, highly customizable and to have a large user base, saving us major development efforts.

Separation of concerns: data access and visualization – Some browser-based user interface libraries use a server-centric approach that takes the data to be displayed and renders a server-side visualization which is subsequently sent to the browser in a proprietary format, thus creating a tight coupling between model and view. A complete separation between data access and visualization was very important for us since various stakeholders at ESO have increasing demand to access live or historic weather data programmatically rather than looking at a plot. As a general strategic trend in the evolution of the VLT/ELT dataflow at ESO, we increasingly strive for web-based solutions, exposing services through RESTful APIs to be consumed by entirely independent graphical user interfaces as well as by programmatic clients. For the ASM, we designed a simple, yet powerful REST API that consumes an HTTP GET call with *from ..to* parameters to specify the time interval for which data points are requested and a *fields* parameter that lists the required physical measurement values. Results are subsequently returned in the lightweight JSON format that is easily consumable by JavaScript clients. The GET call:

```
http://www.eso.org/asm/api/?from=2016-05-24T09:53:11.000Z&to=2016-05-24T09:54:30.467Z&fields=meteo_paranal-rhum1,meteo_paranal-rhum2
```

returns the relative humidity on Paranal at 30m and 2m above the ground in terms of the following JSON structure:

```
{"meteo_paranal-rhum1" : [[1464083651000, 8]], "meteo_paranal-rhum2": [[1464083651000, 8]]}
```

In order to prevent excessive DB queries, different restrictions apply depending on whether the query is made anonymously or by an authenticated user, for example anonymous users are only allowed to retrieve up to 24 hours of data. In addition to this basic interface, our solution provides various other APIs to support the implementation of a user interface, such as an API to retrieve the meta-information describing all available plots and APIs to save and retrieve JSON configuration files.

We implemented the back-end of the ASM web display using the Grails framework[9], which has become the standard for web applications in the VLT/ELT dataflow. It has tremendously increased our productivity by providing comprehensive support for the development of web applications, including sophisticated object-relational mapping, authentication and authorization, easy consumption of HTTP requests, JSON serialization and a rich ecosystem of plugins.

Configurability - It was obvious from the start that a bigger challenge in the web display was to provide sophisticated configurability to the various users of the web display on two levels:

- chart-level: for each chart, the user should be able to configure the size of the displayed time interval and its refresh frequency; the y-axes including their title, color, left/right positioning, linear or logarithmic scale, minimum/maximum or auto scaling, number of ticks, grid lines if any; the plots to be displayed including applicable y-axis, color, line width, marker size, warning and alarm thresholds, acoustic alarm sounds. Given that our charting solution highcharts.js is configured by passing a hierarchical JavaScript object to the chart constructor, we simply extended this structure to store additional configuration information specific to our solution.
- page-level: for each page, the user should be able to configure the number of charts per row, the relative width of each chart in a row, the height of each row, the actual charts to be displayed and their order, inspired by a "portal"-like website. As an extension to the chart-level configuration, we defined an additional object structure to represent page-level configuration (Figure 2).

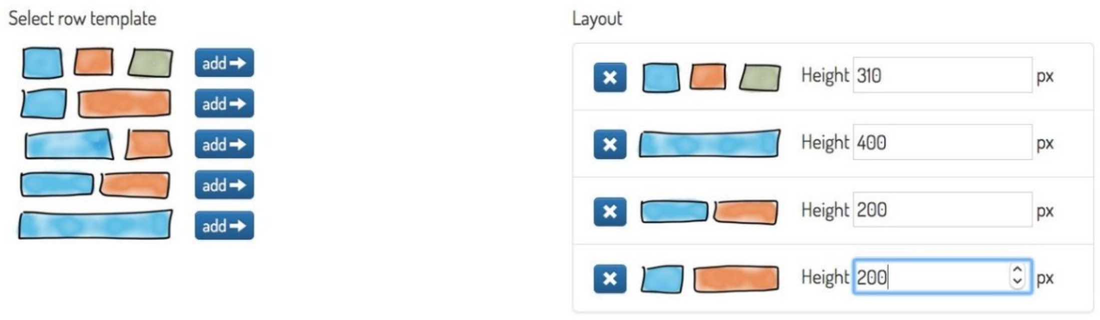

Figure 2. Configuring page layout inspired by portal websites

Of course, simply configuring the page and its charts in memory is not a viable solution. The user has to be able to persistently store entire configurations and reload them anytime. We allow the user to permanently save configuration "profiles" identified by a *namespace* and a *name*, which are therefore easily sharable with other users. Since JavaScript objects can be trivially serialized into JSON, we chose to merge both chart- and page-level configurations into a single object and save each configuration profile as a single, self-contained JSON file in the database. Offering a rich and dynamic configuration user interface in a web browser - similar to a desktop GUI application - is not trivial. Over the last couple of years, browser programming in JavaScript has undergone a major paradigm change: for a long period of time, dynamic browser behavior was primarily achieved using asynchronous server calls (AJAX) triggering callback functions that would directly modify the browser's DOM tree. This often led to hard to understand chaining of callbacks (callback hell), a codebase that was difficult to understand and maintain and a tight coupling of data and view. More recently, browser programming has clearly moved away from this *imperative* programming paradigm to a *declarative* one using a client-side model-view-controller pattern. While the controller fetches and maintains the model data, the view *declares* its dynamic dependencies on model changes. The DOM is never directly modified. For the web display, we chose one of the most widespread frameworks supporting this declarative paradigm, i.e. Google's angular.js version 1.x[10]. While the underlying concepts of angular.js do have a significant learning curve, we were able to implement the ASM user interface with comparably little JavaScript code, which is well structured, maintainable and scalable. Implementing a desktop-like application in a browser also comes with a specific new challenge: the behavior of the browser when pressing the back button and the requirement to be able to bookmark different states of the application, such as a particular configuration profile or a view of a specific time interval. When loading such a bookmarked page, the angular application must reinitialize itself into that particular state. All of this is enabled by the angular framework and its so-called client-side routing support.

Reusability - The primary purpose of the ASM is to display live weather data. However, our software implementation is not specific to the type of data displayed. The meta-information about the available plots (name, description, axis label, precision, scientific or engineering) is all stored in suitable database tables. Therefore, our solution is highly generic and re-usable and can be adapted to other domains displaying live data simply by configuring this meta-information in the database. The application was already reused for displaying live telemetry data of ALMA antennas and for an ELT GUI prototype.

Reliability and Performance - At the beginning of the project, we were concerned about the reliability of polling data from a server using JavaScript in a browser. We made sure that the users are immediately alerted with a strong visual indication when live data can no longer be loaded from the server. In that case, specifically users in the Paranal control room would have to refresh the browser and, if the problem persists, initiate appropriate troubleshooting. After half a year of operational experience we can conclude that the JavaScript polling in modern browsers is actually very reliable.
We were also concerned about the database query load when many users are viewing live data at the same time. Comprehensive load tests with Apache Benchmark showed that the load is actually surprisingly low.

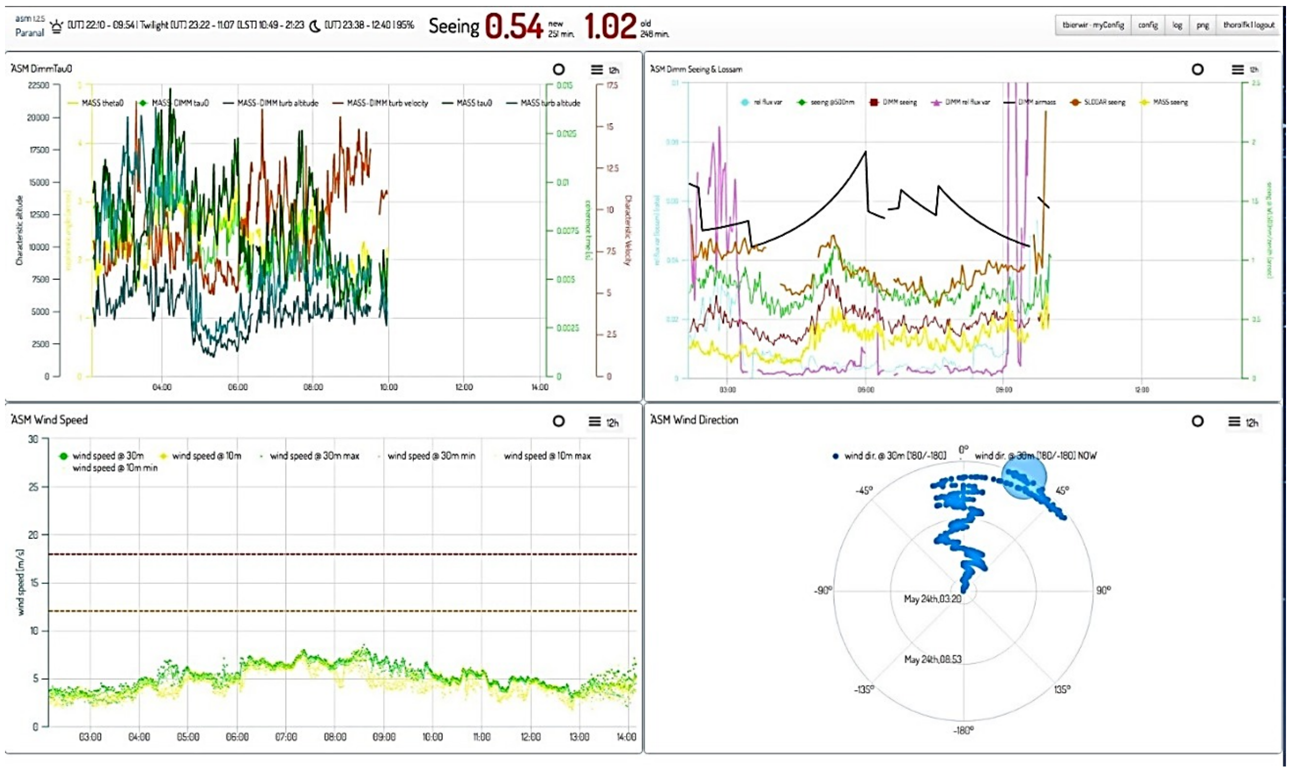

Figure 3. Live display of several charts

The choice of productive technologies allowed us to focus our developments efforts and to deliver value to the stakeholders. Some notable features are: display of upper and lower warning/alarm thresholds and selectable acoustic alarms when they are crossed (Figure 3), polar plots of the wind direction where radius is time and the angle is the direction, easy zooming of a chart to full screen (lower-right plot in Figure 3), shading of twilight and night times, interactive and highly performing exploration of the complete history with progressive loading of data in the required resolution.

7. CONCLUSIONS

Despite the fact that the new system integrates additional devices and provides a much wider set of parameters, the switch to a more modern hardware architecture and programming languages has allowed us to implement what we believe is a more robust, maintainable and extensible system.

On the control system side we have already received requests for improvements and for adding further data. The next big testbed will be when the new Adaptive Optics Facility, currently being integrated, will start using the atmospheric profile data that the ASM is providing. At this stage new requests from actual usage will for sure come up.

As far as the web display is concerned, we conclude that using highcharts.js for visualization is highly recommended and saved us a lot of development effort. The usage of the "declarative" client-side web framework angular.js was essential for building a desktop-like dynamic user experience in the web browser and is also highly recommended. Currently, the ASM client is still bundled and deployed along with the REST server. A further improvement could be to completely decouple the JavaScript client and the Grails server so that both can be deployed independently. In the VLT/ELT dataflow, several major projects are underway that use angular.js version 2 developed and deployed independently from the server lifecycle.

We found extremely valuable the possibility of running the new system in parallel to the old one for more than one year. This has allowed us to compare the operational behavior of the two systems and tune our new software. During this time we could also continuously improve the stability without impacting operations, collecting at the same time feedback from the users and directly implementing what was requested.

The upgrade of the La Silla ASM and the installation of a new system at the E-ELT site (Cerro Armazones) using the same technology are now being planned. Already now La Silla data, coming from the old ASM, has been integrated in the new ASM Database and Display applications. This makes it already possible to compare the weather conditions at the two sites.

Also the ALMA observatory has shown interest in the display technology, and prototypes are being written.

Public access to the ASM display is available at: http://www.eso.org/asm

ACKNOWLEDGMENTS

We would like to thank here the very many colleagues who provided requirements and feedback for the upgrade of the ASM in Paranal and Garching, and in particular S.Brillant, C.Martayan and R.Arsenault.

REFERENCES

[1] Sandrock, S., Amestica, R., Duhoux, P., Navarrete, J. and Sarazin, M., "VLT Astronomical Site Monitor: control, automation and dataflow" Proc. SPIE 4009 (2000).
[2] Sarazin, M., Roddier, F., "The ESO Differential Image Motion Monitor", Astron. Astrophys. 227, 294-300 (1990)
[3] Kuntschner, H., et al., "Operational Concept of the VLT's Adaptive Optics Facility and its instruments," Proc. SPIE, 8448, 07 (2012)
[4] Tokovinin, A., Kornilov, V., "Accurate seeing measurements with MASS and DIMM", Mon. Not. R. Astron. Soc 381, 1179-1189 (2007)
[5] Osborn, J., Wilson, R., Butterley, T., Shepherd, H., Sarazin, M., "Profiling the surface layer of optical turbulence with SLODAR", Mon. Not. R. Astron. Soc. 406, 1405-1408 (2010)
[6] Kerber, F., et al., "A Water Vapour Monitor at Paranal Observatory", Proc. SPIE, 8446, 135 (2012)
[7] Data Driven Documents, https://d3js.org
[8] Highcharts, http://www.highcharts.com
[9] Grails Web Framework, http://www.grails.org
[10] angular.js, https://angularjs.org
[11] ESO Data Interface Control Document, GEN-SPE-ESO-19400-0794, 201