|  |  |
| --- | --- |
| RAS  Introduction | Abstract  This document provides an overview of the different functional modules in the RAS application and motivates the need for such a solution. |

# RAS

**Why RAS?**

* O&M: Monitor our customer’s assets to meet or exceed committed performance
* Maximize the performance where possible, and capture part of that value
* Manage our risks of premature failure, reducing warranty costs
* Intelligent reporting about <0.1% of data through smart filtering
* Notification system that can detect anomalies and alert before a failure occurs
* Remotely diagnose assets for effective (one shot) maintenance
* Remote parameter configuration/tweaking
* Fleet-wide remote firmware update
* Remotely control assets (on/off/derate/geo-boundaries)
* Aggregate assets from BTM to Utility scale (fleet control)
* Enable demand side management services and revenues
* Provide responsive GUIs & dashboards to end users (resi, mobile, stationary ...)
* Build data analytics, underpinning our value and technology
* Improve performance predictability knowledge
* Offer compelling and user-friendly dashboards and reports
* Work with a generic solution (not battery specific) to offer fully integrated data services to inverters, motors, transformers, feeders, meters, boilers, PV panels, windmills, flow batteries, CHP, etc…
* Highly configurable, no need for programming
* Ability to work both as a localized solution behind customer firewall (fog) or cloud based
* No monthly cost (as opposed to third party solutions)
* Flexibility to connect to third-party sources and tools
* Go downstream !

**RAS** is short for Remote Asset Services.

An **asset** can be any kind of equipment. For example, a city bus, a stationary storage system, a street lamp, a vending machine, etc. The asset usually has a significant economic value for its user, the customer.

To protect that value and ensure reliable service, features such as data logging and data analysis are important, flagging any misuse or exceptions. Servicing a high number of such assets can be very time consuming requiring (remote) configuration and update tools.  
Some assets need to be grouped and virtualized as one. RAS addresses all those challenges and more.

For RAS, it is required that the asset contains a **device** (or the functions thereof) that on one hand measures and logs data (about the asset), and on the other hand exchanges data with the RAS server.  
The MFC is an example of such a device.

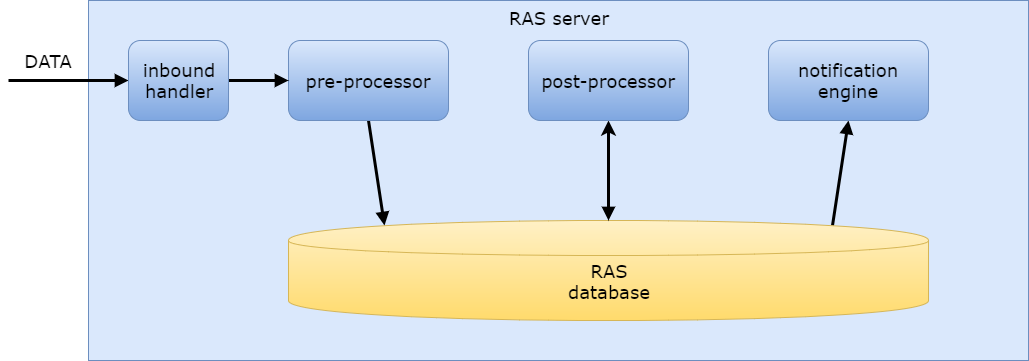


The RAS design has taken into consideration a multitude of **communication protocols**, ranging from FTP for simple file transfer to MQTT for advanced IOT devices.

Even though the device is the contact point between the asset and RAS, RAS assigns data to the asset, not the device. The distinction between an asset and a device is crucial to RAS.

Assets can be assigned to **asset types** allowing easier management of groups of assets. Asset types can be related to each other to form hierarchies of asset types.

The data that is sent to RAS passes several stages before it is stored in the RAS database:



The data is first handled by an **inbound handler** that is usually specific for the communication protocol. For example, to support the FTP protocol RAS has an FTP service to receive data.

The **pre-processor** will pick up the sent data and will start processing the data: it identifies the specific data format and then performs specific validation routines on the data. Finally, the pre-processor stores that data in the RAS database. The validation rules can be edited per asset type in the RAS administration console.

After the data is inserted into the database by the pre-processor, the **post-processor** will kick into action. Using the available data, it can be configured to perform calculations and aggregations to facilitate a higher insight into the data, and thus the operation of the asset. The results of these calculations are inserted back into the RAS database. This functionality makes analysis and dashboarding/reporting significantly easier. It allows to separate the few complex and many simpler tasks over different organizational roles. The post-processor rules can be edited per asset type in the RAS administration console.

All RAS features can be configured from within any supported web browser.

Using **dashboards**, the user can check the data of the asset online. Many different forms of data visualization are available to select from, each offering a lot of customization options. Custom filters allow the user to see only the relevant data. Filters can also be stored and edited.

Dashboard data visualizations (from DevExpress website, which is the vendor of the dashboarding tool):

|  |  |
| --- | --- |
| Chart |  |
| Scatter chart |  |
| Grid |  |
| Pies |  |
| Cards |  |
| Gauges |  |
| Pivot |  |
| Choropleth map |  |
| Geo Point map |  |
| Bubble map |  |
| Pie map |  |
| Range filter |  |
| Images |  |
| Text box |  |
| Treemap |  |

Dashboards offer interactivity, for example to compare data of similar assets from different groups (such as geographical areas). An unlimited number of dashboards can be created, updated and deleted from the console. Dashboard also offer several export functions (picture or raw data).

RAS also has a **reports** feature. As with dashboards, reports can be customized by the user but reports do not have the interactivity as they target print/pdf. These can also be created, updated and deleted from the online console. Also the reports engine uses DevExpress technology.

Example screenshots (from DevExpress website):

* General content

|  |  |
| --- | --- |
| bar code |  |
| check box |  |
| gauge |  |
| label |  |
| character comb |  |
| picture box |  |
| rich text |  |
| table |  |
| zip code |  |

* Extended data

|  |  |
| --- | --- |
| chart |  |
| pivot grid |  |
| sparkline |  |

Both dashboards and reports can be for one asset or groups of assets. We recommend a simple training to get started with dashboard and reports.

Key to allowing focus on only the data the user is interested in, are **filters**. A filter is a list of conditions that limit the scope of the data. A filter is created using a user-friendly wizard and allows filtering on those assets, properties, values and time ranges the user is concerned about. A filter can be applied to data used in dashboards and reports, and accelerate the user experience.

RAS can be configured to notify a (group of) user(s) in case an asset/device reports specific data that meets a certain preconfigured filter (E.g. an event). For example: if the SOC of a battery drops below 15% an e-mail message is sent to a service engineer. This is handled by the **event engine**. Again, event rules can be set in the RAS administration console.

The following filters are supported:

|  |  |
| --- | --- |
| Filter type | Short description |
| Simple value | simple evaluation of a value using relational operators |
| Value change | check if value has changed compared to previous value |
| Threshold | checks if value rises above/drops below a specific value |
| Watchdog | check if an asset has sent data to RAS for a specific time |

The **messaging service** of RAS is used to configure the outbound communication to the assets’ stakeholders. RAS can be configured to send **e-mail messages** to a set of users and/or to show **website notifications** on the RAS website. A message can be configured to be sent at regular intervals (e.g. the first day of every month) or when a specific event occurs (defined in the event engine). Such an event can be asset-related (e.g. temperature exceeds a certain value) or system-related (e.g. a user has performed too many login attempts). RAS offers a powerful WYSIWYG editor to compose these messages that can optionally contain a link to a dashboard or report.

Assets can be configured in the **asset management** user interface. Both permanent and dynamic properties of the Asset can be consulted and added (permanent: information that does not originate from the asset itself and doesn't change).

An asset can be managed by multiple companies. This is useful is an asset was passed through a distribution channel to the end customer, and multiple parties requires certain sets of data. Each user belongs to one company.

The RAS platform login function requires a username and password (**authentication**). **Authorization** is implemented around item-level security and feature-level security.

Users will be assigned one or more **permissions**. Whether a user is allowed to access a particular feature (e.g. create a dashboard) depends on the permission(s) the user has (**feature-level security**).

Using **item-level security**, it can be configured to allow specific users access to specific items (e.g. user U1 can edit dashboard D1 but cannot edit dashboard D2 while user U2 can only edit dashboard D2).

RAS is not only about getting data from the asset. Using RAS it is also possible to **Control** the asset's behavior remotely, i.e. send data (configuration, commands, ...) to the asset, e.g. update an asset's configuration parameter.

The **Firmware management** module of RAS is responsible for the remote management of the firmware of the devices (note: device, not asset): firmware repository, update paths, hardware/firmware compatibility management, asset group firmware update scheduling, recovery etc.

The **RAS monitor's** job is an administrative module to keep an eye on the RAS application as a whole. It tries to answer questions like for example:

* how are the indices of the database performing?
* how fast are data queries being executed?
* do we need extra processing power?
* how many data files are uploaded on a daily basis?
* how many assets are configured in RAS?
* how many unused reports are there in the system?
* are we getting many login attempts with false passwords?

The **RAS logging** module records all system activity. Every important system event is written to a dedicated log allowing RAS service personnel to reconstruct what happened when analyzing specific events.

The purpose of **RAS auditing** is to register all important actions that a user performs in RAS. This functionality is essential for registering activities that may have affected a specific process or event.

**RAS technical terminology**

Some clarification about technical terminology that is used:

* Asset: the asset
* AssetType: the type of the asset
* AssetTypeProperty: property of the asset (e.g. 'CURRENT')
* AssetPropertyValue: value of the property (e.g. 150)
* ...