Master Alarm Database – What format?

Introduction

A Master Alarm Database (MADb) is one of the fundamental building blocks required to underpin the success of any alarm improvement initiative. It is required for the life of your alarm system, and without one, you will struggle to manage your alarms appropriately.

What is a Master Alarm Database?

A Master Alarm Database is defined in the alarm management standards[1,2], as an "*authorized list of rationalized alarms and associated attributes*".

Whilst the starting point for this list is likely to be a dump of the configuration from your control system; there is so much more information that should be recorded for each alarm such as the cause of the alarm, consequence of not responding to the alarm, the required operator action to respond to the alarm, along with other ancillary information such as P&ID reference etc., which should also be captured and stored within your MADb.

What format is useful/acceptable?

Although in the latest edition of ISA 18.2, there is a note stating, '*The list can be in many forms and* not only in *the form of a database*'; as we need to manage huge amounts of information, some form of electronic 'database' would be sensible. This effectively means we have two options, a spreadsheet or a database.

How many times do we hear owner/operators or technical persons say things like, "We don't need a database application, we have a spreadsheet guru in our department/company" or "We have a master alarm database, look, here is our spreadsheet".

> **Q:** When is a database not a database?
>
> **A:** When it's a spreadsheet.

Spreadsheets were designed & have been developed over many years to aid with repetitive calculations and data analysis; great for accountants and producing pretty charts. But databases were designed and developed to manage data and textual information, and this is what a MADb needs to do, manage huge volumes of data and textual information.

In this paper, we will discuss why we believe a true database is the only sensible option for your MADb, and not a faux database, such as a spreadsheet.

Spreadsheets vs Databases

Your choice

So why would we choose to use a database for our MADb instead of a spreadsheet?

The comparison table below, (Table 1), highlights the main reasons why a database is a superior tool compared to a spreadsheet when implementing a MADb for your site.

Of course, many companies utilise a spreadsheet simply because it's easy, quick, and cheap. Most people without any training or special skills, can import a csv file into a spreadsheet and hey presto, a MADb has been created. But beware!

- Not all the data you want to manage comes as a single, easy to read csv file
- In many systems, each point type has different parameters that cannot easily be stored in one tab in a workbook. This means you will potentially have multiple tabs to manage in your spreadsheet MADB, and

[1] ISA 18.2 – Management of Alarm Systems for the Process Industries
[2] IEC 62682 (BS EN 61682) – Management of alarm systems for the process industries

- When the data is read in, does your preferred spreadsheet application automatically format it according to its own view of what the data should look like? This can be a real problem with date and time fields, and floating-point numbers or numbers stored as text
- Also, don't forget, once you have imported your control system configuration data, there are many extra fields which must be created and managed within your MADb

Feature / requirement	Spreadsheet	Database
Easy to create from scratch by untrained user	✓	×
Works well with large amounts of data	×	✓
Easy to test functionality	×	✓
Easily adapted to suit different systems	×	✓
Easy data integration from multiple sources	×	✓
Allows a digital signature to validate data integrity	×	✓
Easy, error free bulk copy and edit	×	✓
Audit trail for all changes	×	✓
Retains all historical values	×	✓
Can maintain multiple values for a single parameter	×	✓
Multi user concurrent access	×	✓
Multiple roles for reviewing/approving for Moc process	×	✓
Able to create ARM from data	×	✓
Displays information in a user friendly interface	×	✓
Allows comparison between master data & live data	×	✓

Table 1 - Spreadsheet vs database comparison

Futureproofing

Let us not forget that managing alarms is a requirement for the life of your plant; therefore, your MADb must be available and functional until your plant is decommissioned, and may need to go through many changes as control systems are upgraded or standards evolve.

Most people who create a MADb in a spreadsheet, do so to 'get the job out of the way quickly and at minimal cost', as they have been 'given' the task of managing alarms in addition to their existing workload. Whilst initially, a spreadsheet may seem to provide a solution; what happens when that person moves on and someone else takes on the alarm management role, who perhaps isn't as skilled with spreadsheets as the originator?

What happens when your control system is upgraded or changed, and the spreadsheet must be modified or completely redesigned? All the formulae and any user forms or code that have been generated will have to be re-written and re-tested, and how will your current alarm information be migrated from one spreadsheet to another. Who will do all this work, and will this be a cost effective use of their time?

With a database application such as MAC Solution's Guardian which is continually being developed and improved, it is very simple to import data from a new or different system whilst retaining the original system data for historical purposes, and migrating critical alarm parameters is a simple task.

Accessibility

How many times have you tried to open a spreadsheet only to be told it is locked for editing, but can be opened as a read only copy. Only one person can work on a spreadsheet at any one time. If you choose to open the locked spreadsheet and save as a copy, how do you update your changes in the 'official' MADb whilst guaranteeing you do not overwrite changes made the user who had the database open originally? In practice, this would be incredibly difficult.

A database, on the other hand, can be accessed and worked on by numerous people concurrently, (depending on the licensing of your application), with only individual records (tags) being locked as they are worked on. No problems for a collaborative team effort, where users may be situated in different offices.

Memory usage and performance

One of the problems with using a spreadsheet for your MADb, is memory. With a small control system of a few hundred tags with relatively few parameters, your spreadsheet may only amount to about 500MB. It will load quickly, and you should have no real problems.

However, once you start to add all the extra fields you need to document each alarm, perhaps several dropdown lists, (for every alarm), to aid data entry consistency; maybe some Visual Basic code or forms to make the spreadsheet more user friendly, your spreadsheet will soon balloon in size. If you have several thousand tags within your system, the problem is magnified. It won't be long before you get a "Not enough Memory" or "Not enough system resources to display completely" error message. Excel may also crash intermittently or stop working even with small to medium size files, (There is a Microsoft support article on how to deal with these issues).

The problem with a spreadsheet is; everything, every last byte of data for every tag, must be loaded into memory when the spreadsheet is opened. Unfortunately, Excel has its own memory management system which does not use all the available memory; so, when Excel tells you, you have run out of memory, you can't solve your problems by simply increasing system memory or the size of your swap file.

And let's not forget how long huge spreadsheets take to load, as they clog up your memory.

These are not issues with a database, especially where the application has a browser based interface; as only the data that are required within a specific record, are loaded into memory when needed.

User interface

Unless you spend huge amounts of time, effort and money in creating and testing user forms for your spreadsheet, you will be managing your MADb by scrolling left, right, up and down as you review alarms and attempt to copy data without errors. You may also have to manage, edit and copy data across several tabs, where data have been saved according to point type with each point type having a different set of parameters and parameter names.

In a database application, the data you wish to work on are presented to you in a predefined form, without the need to either hide columns, (dangerous when copying), or for excessive scrolling across tens or hundreds of columns to find the data you wish to manage.

Data validation

No matter the application, we can try to make life easier for the user by offering dropdown lists, carrying out calculations, or making selections based on the values of parameters within our dataset.

In a spreadsheet, every cell where a dropdown is required, needs a reference to the list which is usually stored on a separate 'control' sheet. We must ensure that every cell where the dropdown is required correctly references the list. Where a formula is used, every cell must be checked to ensure the correct formula is entered. Now, did we use absolute (using '$') or relative cell references? Did we use absolute for one cell in a formula and relative for the rest of the references? In our spreadsheet, if we use one calculation or dropdown, then we have 10,000 cells to check to ensure they are ALL correct.

If a cell is accidentally deleted or overwritten, yes it happens, or a new row is inserted/deleted, the validity of our data can no longer be guaranteed; and potentially, formulae may reference the wrong cells. In a worksheet where many thousands of cells can be impacted, this is yet another significant drawback of using a spreadsheet.

In a database, data and calculation validation presents no problem. The list, calculation or selection methodology, is defined in one place and used by each record as it is referenced or

edited by the database. The operation of the list, calculation or selection methodology needs to be tested and validated only once to prove its accuracy.

When we come to the important task of determining the priority of an alarm, either we must manually calculate and enter the priority into our spreadsheet many thousands of times, or we must make sure that every calculation within the spreadsheet is correct and hasn't been accidentally changed. Remember, there may be six or more priority states for an analogue tag and each state must have its individual, verified priority calculation.

In a spreadsheet, tens of thousands of checks may be required to ensure every reference and calculation is valid.

In a database, only one calculation is required which is used by every record. Once the code is written and tested, there is little possibility of it going wrong by accident and each record can be validated using a digital signature, which is not possible with a spreadsheet.

Ancillary information

It's one thing importing your control system's configuration, in whatever format you are presented with; but you also need cells to store all the extra fields you wish to document for each alarm within your MADb.

For example, if an analogue tag has six alarm levels, (Underrange, LoLo, Low, High, HiHi & Overrange), each of these alarm conditions will require individual columns to store for example, cause, time to consequence, response, alarm class, comments and much more. In practice, this will require a minimum of 10 extra columns for each alarm state. So, we have now added 60 new columns to our worksheet, (perhaps many more depending on the extra information you choose to record). For a 10,000-tag database, we have just added 600,000 cells regardless of whether they contain data or not, and we must manage these cells somehow.

Management of Change

Modifications in any good Management of Change (MoC) system once proposed, should be subject to review and approval where review comments can be added, and a full history of the changes maintained.

Excel does have a limited track change facility, if enabled; where changes can be reviewed (previous and current cell content), and accepted or rejected. The history (of saved changes) can be logged to a new sheet; but by default, the history is lost after 30 days.

Changes are simply tracked by cell reference (e.g. Range AU1914). Figure 1 shows typical output from Excel when tracking changes. No reference to tag, alarm state or parameter, and is the user logged in (Ian.B) authorised to approve or reject changes? Quite useless for alarm management purposes.

Also, it is not possible to revert to earlier logged changes unless this is done by manually updating individual cells.

Action Number	Date	Time	Who	Change	Sheet	Range	New Value	Old Value	Action Type	Losing Action
27	12/12/2017	10:39	Ian.B	Cell Change	Alarms	AU1913	N		Result of rejected action	8
32	12/12/2017	10:42	Ian.B	Cell Change	Alarms	AU1927	Y	N		
33	12/12/2017	10:42	Ian.B	Cell Change	Alarms	AU1928	Y	N		
34	12/12/2017	10:43	Ian.B	Cell Change	Alarms	AU1928	N		Result of rejected action	33

Figure 1 - Excel audit trail

As a cell can only retain one value at any one time, if you wish to keep a track of the history of all the values in your MADb, then you will undoubtedly need to keep multiple versions of your MADb spreadsheet which will be horrendous to manage, and incredibly difficult to identify what has changed from one spreadsheet to another.

These limitations make the use of Excel and its tracking function an unworkable option for a MADb, which is intended to help you manage change throughout the life of your plant.

However, in a well-designed database application, an audit trail will be employed; which captures the changes, approver, date and time, etc., and retains this information so the history can be reviewed at any point in time. It is also possible to easily revert to an earlier point in the audit trail by selecting the required changes to be restored.

Bulk copy and edit

Bulk copying and editing of data are two important functionalities that a MADb should include; as this improves the accuracy, efficiency and speed of alarm rationalisation/review sessions. Of course, both can be accomplished in a spreadsheet or a database.

However, in a spreadsheet, copying is so much more difficult and error prone than in a database. Two problems are immediately evident:

- Has someone already pre-filtered and saved the workbook and you haven't noticed?
- When you change the filtering in a spreadsheet, the first row of the selected data is not automatically moved to. If you do not remember to scroll to the top of the worksheet, you may not select all the required cells

In both cases, this means you will not be copying the data to all the tags you wished to copy to.

In addition to these two issues, what if the data you wish to copy is not in contiguous columns?

Consider the following simple example. In the spreadsheet in Figure 2, we wish to copy only the parameters which are in columns G, R, S, AF and AH. To copy data from Tagname 1 to the following tags, we must perform 4 separate operations. Copy and paste column G, then columns R and S, column AF, and finally column AH. We cannot select the whole row to make it easy as there are parameters between the required columns that are specific to individual tags and MUST NOT be overwritten, e.g. column 'X' (Address).

B	C	G	R	S	W	X	AF	AG	AH
Tag Name	Tag Description	Alarmed	Off Label Digital	On Label Digital	Node Name	Address	type	Label	Severity
Tagname 1	Description 1	T	Normal	High	PLC_01	Area01_IN[1].0	ON		2
Tagname 2	Description 2	F	Off	On	PLC_01	Area01_IN[1].1	OFF		1
Tagname 3	Description 3	F	Off	On	PLC_01	Area01_IN[1].2	OFF		1
Tagname 4	Description 4	F	Off	On	PLC_01	Area01_IN[1].3	OFF		1
Tagname 5	Description 5	F	Off	On	PLC_01	Area01_IN[1].4	OFF		1

Figure 2 - Spreadsheet copy

In a database application such as MAC Solution's Guardian, given the same data as our spreadsheet example, we simply choose the tags we wish to copy to, select the individual parameters as shown below in Figure 3, and press finish.

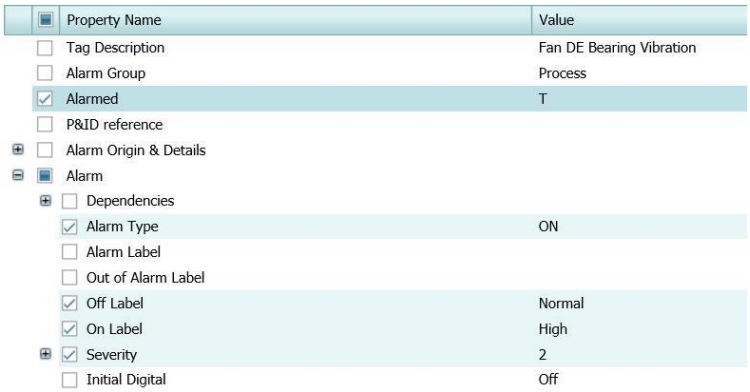

Figure 3 - Database copy

Only the parameters we wish to copy, are copied to all the selected tags. As part of the process, we also create an audit trail containing for each parameter, the original value, the updated value and which tag the value was copied from, not forgetting all our other MoC fields, such as who was

involved in the review, who approved the copy, date etc. This cannot be accomplished in a spreadsheet.

Audit and enforcement

When specifying alarm systems, the EEMUA[3] guidelines recommend that, "*Tools should be provided to audit the current alarm database and compare against a recorded approved database to produce an exception report and optionally, to automatically reset 'unauthorised' changes*".

With a spreadsheet, audit and compare is not so easy. Consider how you will compare the control system configuration to your MADb spreadsheet.

How will you compare, perhaps tens of thousands of values and identify the ones which may have changed, in a spreadsheet which could consist of significantly more than 2 million cells, and where data may be spread over multiple tabs?

With a database tool, audit and compare is an incredibly easy task to perform. Export your control system configuration and import it into your MADb tool. Click a button and within seconds, tens of thousands of values can be checked against those held in your MADb, and a report of exceptions created, whether there are many differences, or only one. If your MADb is linked directly to your control system, this comparison could even be carried out automatically.

Other considerations

In addition to the above noted points, things become ever more complex with spreadsheets if you implement, for example, state based alarming; where multiple values must be stored for a number of parameters.

As a cell in a spreadsheet can only ever contain one value at any one time, how would you maintain multiple parameter values for differing plant states without either creating numerous duplicate tags or implementing a lookup function and table(s) of separate values which must be replicated for all tags. (See concerns above regarding performance and data validation).

In a database application which has been designed to manage advanced alarm handling functionality, maintaining multiple parameter values is not a problem.

Conclusions

If you have a small number of tags in your system and simply want to count the number of alarms you must manage, then a spreadsheet may be a quick and relatively easy way to meet your needs; although this is not always the case, depending on the complexity of your system and the format of the data you are presented with.

Once you begin to consider management of change and alarm rationalisation, which may be a requirement from a regulator; then for the reasons discussed, the use of a spreadsheet as a MADb becomes an unworkable option.

If you are serious about managing and maintaining your alarms with robust management of change control throughout the lifetime of your plant, which you should be; then the only real option for a Master Alarm Database is a database application, designed with alarm management as its core functionality.

[3] EEMUA 191 - Alarm Systems - A Guide to Design, Management and Procurement

Ian M Brown
MAC Solutions (UK) Ltd
Alarm Rationalisation and Services Manager

Educated in Electrical and Electronic Engineering and with over thirty-seven years of experience in the process industries, Ian has accumulated knowledge across a range of sectors, including industrial, chemical, speciality chemical, pharmaceutical, petrochemical, power generation, nuclear and oil & gas; and has held a variety of technical (hardware & software configuration), maintenance management and consultancy roles.

Having successfully led a number of alarm rationalisation projects for clients over the past ten years, which resulted in significant reductions in annunciated alarm rates and improvements to their management of alarms; Ian's expertise covers IEC 62682, ISA 18.2 and EEMUA 191. In addition to being a member of the ISA, he is also a TÜV certified Functional Safety Engineer (6424/13).

Contact Details:
Tel: +44 (0)1246 733120
Mob: +44 (0)7808 039250
Email: ian.b@mac-solutions.co.uk
Web: http://www.processvue.com