ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEPD033

THE TANGO AlarmHandler: ADVANCEMENTS IN CORE FUNCTIONALITY AND TOOLS

G. Scalamera*, L. Pivetta, G. Strangolino, Elettra-Sincrotrone Trieste S.C.p.A., Basovizza, Italy

Abstract

The AlarmHandler system is a key component for ensuring operational safety and efficiency in complex control systems. Key updates include improved support for array data types within the alarm evaluation logic, enabling more sophisticated and flexible condition definitions directly involving array or matrix data. Furthermore new tools, designed to extend the AlarmHandler's reach and usability, are now available. A dedicated notification service allows for configurable, multi-channel alerting, such as email and messaging platforms, facilitating timely operator awareness and response. Complementing this, new management utilities have been created to streamline the configuration, deployment, and maintenance of alarm definitions across distributed systems, significantly simplifying administrative tasks. This contribution details the architectural changes, implementation specifics, and the benefits these advancements bring in terms of system robustness, operator efficiency, and overall monitoring capability.

INTRODUCTION

Modern large-scale scientific facilities, from particle accelerators to astronomical observatories, depend on sophisticated control systems that monitor and manage thousands of devices, sensors, and subsystems operating continuously. The complexity and scale of these installations create critical challenges in maintaining operational efficiency, preventing equipment damage, and ensuring optimal performance. Within this context, alarm management systems are a fundamental tool, providing real-time monitoring capabilities that detect anomalous conditions and alert operators to potential issues before they escalate into serious problems. The TANGO Controls framework has established itself as a widely adopted solution for distributed control systems across many scientific facilities worldwide. As part of this ecosystem, the AlarmHandler system [1] plays a pivotal role in maintaining operational integrity by continuously evaluating alarm conditions based on device attributes and system states.

More and more often control systems rely on array-based and matrix data from advanced sensors, high performance detector systems, and multi-channel devices. Multi-dimensional data led to the development of array support in alarm formula evaluation. The enhanced system introduces native support for array data types within alarm evaluation logic, enabling direct processing of multi-dimensional data and sophisticated condition definitions that were previously impossible or required complex workarounds.

Furthermore, the operational demands of 24/7 scientific facilities require robust and flexible notification systems that extend beyond simple console alerts. Operators rely on multi-channel communication capabilities that can deliver contextual information through various platforms including email, messaging services, and mobile applications. At the same time flood protection mechanism is a key feature to prevent important notifications from being lost when too many are received.

Managing alarm configurations across distributed systems is a significant operational challenge. As structures become more complex, maintaining consistency, implementing updates, and ensuring adequate alarm coverage across hundreds or thousands of devices becomes increasingly difficult without dedicated management tools and automated implementation mechanisms. Significant advancements have been introduced to the TANGO AlarmHandler system that address this operational scenario: a comprehensive notification service architecture provides configurable, multi-channel alerting capabilities that improve operator awareness and response times.

Finally, graphical interfaces have been renewed, with updated widgets to improve overall usability, and new management utilities have been developed to simplify the entire alarm life-cycle, from initial configuration to deployment and ongoing maintenance.

THE AlarmHandler: SYSTEM OVERVIEW

The TANGO AlarmHandler represents a sophisticated, event-driven alarm management system that has evolved significantly since its initial implementation. Based on the EN IEC 62682 [2] standard, it provides a robust foundation for alarm management in distributed control systems. The AlarmHandler is implemented as a modular TANGO device server, written in C++, designed around an event-driven architecture that exploits TANGO publish/subscribe capabilities. The system operates asynchronously, triggering alarm evaluations only when relevant attribute values change, ensuring optimal performance and resource utilization.

A critical component is the formula parsing engine, implemented using the Boost Spirit Parser Framework. This choice provides an object-oriented recursive descent parser that builds an Abstract Syntax Tree (AST) for each alarm rule. Each formula is parsed once at device initialization, with subsequent evaluations performed using the optimized AST representation, resulting in superior performance compared to traditional parsing approaches.

* graziano.scalamera@elettra.eu

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEPD033

One AlarmHandler key architectural feature is based on dynamic attributes: each alarm rule is exposed as a TANGO attribute. This design enables granular event notification, allowing clients to subscribe selectively to specific alarms rather than monitoring the entire alarm space. Each alarm attribute conforms to the EN IEC 62682 standard, using a DevEnum data type with standardized values: NORM, UNACK, ACKED, RTNUN, SHLVD, DSUPR, and OOSRV. This approach facilitates the construction of hierarchical alarm systems, where higher-level AlarmHandler instances can monitor the states of lower-level alarms, enabling sophisticated alarm correlation and escalation strategies.

The AlarmHandler system utilizes a streamlined data management approach, storing alarm configurations directly in the TANGO database within attribute properties. This eliminates dependencies on external database systems while integrating seamlessly with standard TANGO administrative tools. Alarm history is provided through integration with the HDB++ archiving system, enabling comprehensive historical analysis and correlation with any other system data.

The enhanced parser for the alarm formula supports the definition of complex alarm conditions: provides mathematical functions (sin, cos, pow, min, max), ternary conditional operators, and comprehensive attribute quality handling. Formulas can reference TANGO attribute qualities directly and incorporate quality based logic, enabling alarm conditions based on data integrity as well as values.

The AlarmHander system demonstrated robust performance capabilities, successfully managing thousands of alarms with minimal resource overhead. High frequency event processing is supported without losing events; the CPU utilization remains low even under heavy alarm load conditions. However, evolving operational requirements have identified opportunities for enhancements in core functionality and supporting tools to better serve modern control system needs.

CORE ENGINE ENHANCEMENT: ADVANCED ARRAY PROCESSING

In large scale facilities with many instances of identical devices distributed across the installation,such as beam position monitors or detectors with multiple acquisition channels, TANGO device servers commonly implement DevSpectrum (one-dimensional array) or DevImage (two-dimensional array) attributes. Furthermore, N-dimensional array data types are planned for future TANGO implementations. AlarmHandler array support was limited to accessing individual elements by index using the syntax:

$$device/name/attribute[index]$$

This approach was restricted to one-dimensional arrays only. While acceptable for arrays with tens of elements, this becomes impractical for larger datasets containing thousands to millions of elements, resulting in thousands of individual alarm rules or formulas with thousands of conditions.

Multiple Indexing Support

The first enhancement extends indexing support to multiple dimensions. The syntax

$$device/name/attribute[index1][index2]...[indexN]$$

is now valid, and N-dimensional indexing is already supported in preparation for future TANGO N-dimensional array implementations.

Array Operations

The core enhancement enables condition evaluation across all elements of an array while maintaining syntactic compatibility. Array attributes can be used in formulas like scalars: the parser automatically extracts values and dimensions into C++ vectors, handling scalars as 1×1 arrays.

Unary operators and functions are applied to every element of the array, producing an output array of the same dimensions.

Binary operators and functions require that either both operands have identical dimensions or one operand must be a scalar. Otherwise, an exception is raised and the formula evaluates to the ERROR state. When both operands have the same dimensions, operators are applied element-by-element, producing an output array of the same size:

$$array1[i][j]\ operator\ array2[i][j] \rightarrow result[i][j]$$

When one operand is a scalar, the operation applies the scalar to every element of the array operand, producing an output array with the same dimensions as the non-scalar operand:

$$scalar\ operator\ array2[i][j] \rightarrow result[i][j]$$

The conditional operator (if X then Y else Z) with syntax $(X\ ?\ Y\ :\ Z)$ handles arrays by implicitly reducing X using logical OR to produce a boolean scalar value, then returning Y or Z based on this result.

Array Slicing

When dealing with large arrays it is often necessary to operate on subsets rather than individual elements or the entire arrays. The enhanced syntax supports various slicing operations:

- single index for multi-dimensional arrays: $array[i_0]$ selects a specific slice
- comma-separated index lists: $array[i_0, i_1, i_2]$ selects specific elements
- range specification: $array[i_1 - i_2]$ selects elements from i_1 to i_2 (inclusive)
- all index indicator: $array[-1]$ selects all elements in that dimension
- combined syntax: any combination of the above

For example, to select row i_0 and all rows between i_1 and i_2 from a 2D array the following syntax can be used

$$device/name/array[i_0, i_1 - i_2]$$

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEPD033

which equals to

$$device/name/array[i_0, i_1 - i_2][-1]$$

and to

$$device/name/array[i_0, i_1 - i_2][0, last\_col\_index]$$

Reduction Operations

Two reduction functions convert arrays to scalar values for final alarm evaluation:

- OR(): applies logical OR to all array elements
- AND(): applies logical AND to all array elements

Since alarm rules must produce boolean results (true for alarm condition, false for normal condition), array results are implicitly converted to boolean arrays by testing each element as not equal to zero, then reduced using OR(). This ensures that an alarm condition is triggered if any array element satisfies the alarm criteria.

ALARM SYSTEM: THE NOTIFICATION SERVICE

A comprehensive notification service has been implemented as a new TANGO device: the AlarmNotify (Fig. 1). Following the same modular design philosophy as the AlarmHandler, multiple AlarmNotify instances can be deployed as needed, where each AlarmNotify connects to a configurable list of AlarmHandlers. The AlarmNotify service supports the configuration of recipients at multiple levels:

- individual alarm-specific recipients
- group-based recipients, interested in alarms belonging to specific groups
- global recipients, monitoring all alarms

For each alarm, the service builds and maintains a list of recipients organized by alarm state. The service subscribes to change events from the configured alarms and triggers notification commands when state transitions occur that have associated recipients. When an alarm state change event is received, the notification system sends a command with a DevString input argument containing alarm details formatted as "key=value" pairs, separated by semicolons. The standardized notification payload includes:

- *name*: the alarm identifier
- *handler*: the source AlarmHandler device name
- *values*: attribute values from the formula evaluation, formatted as JSON
- *url*: optional URL associated with the alarm for additional context
- *msg*: the configured alarm message
- *formula*: the alarm condition formula
- *groups*: alarm group associations
- *shlvd_time*: the configured shelved time duration
- *state*: the new alarm state that triggered the notification
- *time*: timestamp of the state change

The notification architecture supports multiple communication channels through dedicated TANGO devices. The AlarmMail device provides email notification capabilities, while additional channels can be easily implemented following the same command interface pattern. This modular approach allows integrating the preferred communication platform while maintaining consistent notification formatting and behavior.

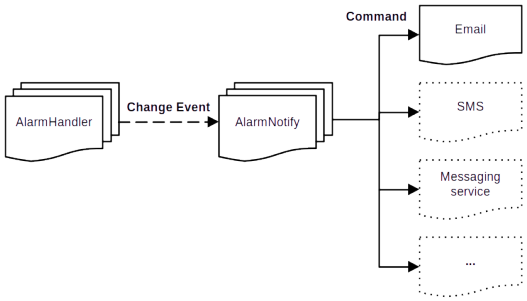

Figure 1: Notification block diagram.

Flood Protection Mechanism

Flood protection plays a critical role in maintaining notification effectiveness during high-frequency alarm scenarios. Traditional shelving at the AlarmHandler level, while effective, requires manual intervention and disables notifications across all channels simultaneously. The AlarmNotify implements an intelligent flood protection algorithm using two configurable parameters:

- time period: the monitoring window duration
- maximum notifications: the threshold count within the time period

For each individual alarm, the system monitors notification frequency within the configured time period. When the maximum notification count is reached:

- immediate notifications cease for that specific alarm
- subsequent state changes are collected and queued
- aggregated notification is sent when the time period expires, containing all collected state changes

This approach provides several advantages:

- no information loss: all alarm state changes are preserved and notified
- automatic recovery: no manual intervention required
- per-channel control: different notification channels can maintain independent flood protection settings
- reduced noise: operators receive consolidated information rather than overwhelming individual notifications

The flood protection mechanism ensures that critical alarms remain visible while preventing notification overload during periods of high system activity or device instability.

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEPD033

STREAMLINING OPERATIONS: NEW MANAGEMENT TOOLS

While the AlarmHandler was designed with configuration ease in mind, using the TANGO database for configuration storage and providing access through *Load*, *Modify*, and *Search* commands or direct attribute property editing, the need for a dedicated graphical user interface becomes apparent as systems complexity grows.

The existing configuration approach, while flexible, presented several operational challenges:

- distributed knowledge requirements: users needed to know which AlarmHandler handles a specific alarm
- technical interface complexity: direct attribute property manipulation required familiarity with TANGO database structures
- lack of unified view: no consolidated interface existed for managing alarms across multiple AlarmHandler instances
- limited filtering: no mechanism to present users with only relevant alarm subsets.

To address these limitations, a new TANGO device, the AlarmManager, has been developed. Designed with modularity in mind as all the other alarm system components, the AlarmManager accepts a configurable list of AlarmHandler instances and automatically builds a comprehensive alarm inventory. Users do not need to keep track of alarm distribution across different handlers.

Moreover, when an alarm name is specified in the tag attribute, the AlarmManager validates the alarm's existence and dynamically exports its properties as accessible attributes. This abstraction layer provides users with a simple and immediate access to underlying database structure, which are kept secure, while maintaining full configuration access.

The AlarmManager device provides a standardized interface for alarm management operations, regardless of the underlying AlarmHandler(s), creating a single point of access for distributed alarm configurations.

Graphical User Interface

Built upon the AlarmManager foundation, a comprehensive GUI application has been developed that significantly improves the user experience (Fig. 2). The new GUI interface displays the complete alarm inventory as a hierarchical, searchable, tree structure, enabling users to browse alarms intuitively, without knowledge of underlying system architecture, filter alarms by searching a text in their fully qualified names and navigate large alarm datasets efficiently.

When an alarm is selected, the GUI dynamically loads and shows the configuration properties as editable fields, allowing to view the configuration just in time and access to actual alarm settings. Immediate modification of alarm parameters is available, as well as duplicating any existing configuration, which simplifies cloning existing alarms as templates for new ones. In addition, the GUI allows renaming existing alarms, even when this involves moving them to a different AlarmHandler.

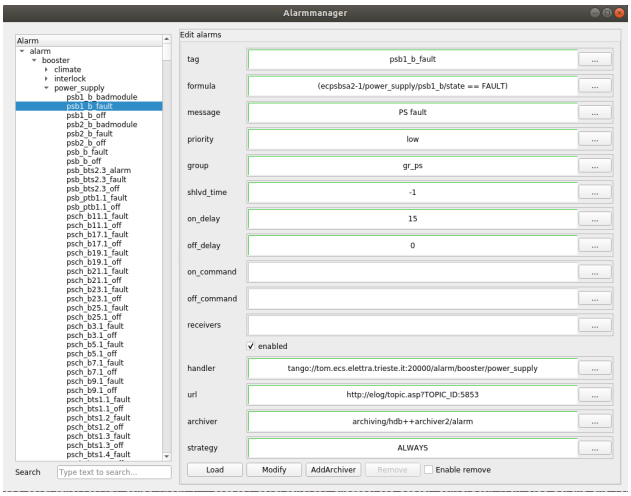

Figure 2: AlarmManager GUI.

Automated Archiving Management

The AlarmManager goes beyond configuration management, also supporting the configuration of historical archiving for alarms. The system automatically handles HDB++ [3] archiving configuration for managed alarms, providing a default behavior, which exploits a dedicated system-wide instance of the HDB++ EventSubscriber, and a custom setup where the EventSubscriber instance to be used can be specified per-alarm.

This integrated approach ensures that alarm historical data collection begins immediately upon alarm activation, eliminating the manual coordination previously required between alarm configuration and data archival setup.

Impact on Operations

These management tools collectively address the administrative overhead that previously hindered efficient alarm system maintenance. By providing intuitive interfaces, automated processes, and centralized, yet modular, management capabilities, operators can focus on alarm logic and monitoring rather than navigating complex configuration procedures. The tools maintain full compatibility with existing command-line and programmatic interfaces, while significantly lowering the barrier to effective alarm system administration.

THE AlarmHandler GUI

A dedicated GUI has been developed for the TANGO AlarmHandler. The *alarm-ng* application, shown in Fig. 3, implemented with Qt [4], presents a main view listing alarms together with their timestamp, status (NORMAL or ALARM), acknowledgment state (ACK or NACK), and severity level (high, medium, low, lowest). Selecting an item provides access to a detailed view that includes a descriptive

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEPD033

message, a link to extended documentation, the evaluation formula, and the input values used in the computation. An additional view, structurally equivalent, is dedicated to disabled alarms. Furthermore, the application integrates the HDB++ historical database, a high-performance time-series archiving system for TANGO Controls, which allows users to navigate and visualize the history of recorded alarms.

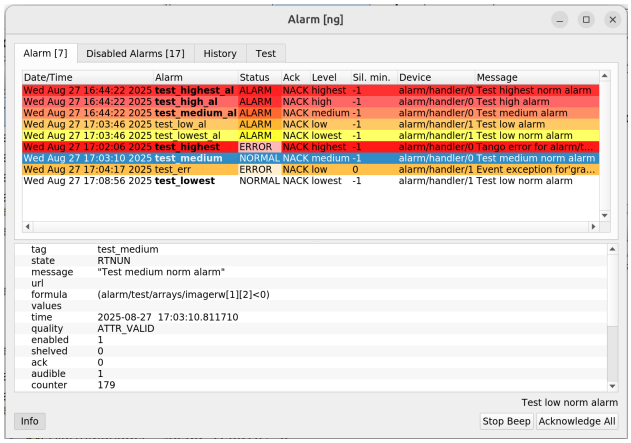

Figure 3: AlarmHandler GUI.

CONCLUSION

The AlarmHandler, togheter with the additional devices and tools, provides a solid foundation for alarm management, compliant to the current standards. The improvements presented in this document represent a significant evolution of the TANGO AlarmHandler system, responding to critical operational requirements that have emerged with the increase in the size and complexity of control systems. The modular design ensures that facilities can adopt single components, based on their specific operational needs and implementation constraints, while maintaining compatibility with the AlarmHandler ecosystem. All the improvements make the TANGO AlarmHandler a robust and intuitive solution for alarm management in large scale scientific facilities. Future developments will ensure that AlarmHandler continues to meet the evolving needs of modern control systems.

REFERENCES

[1] G. Scalamera, L. Pivetta, and S. Rubio-Manrique, "New developments for the TANGO Alarm System", in *Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, pp. 797–800. doi:10.18429/JACoW-ICALEPCS2017-TUPHA165

[2] *Management of alarms systems for the process industries*, International Electrotechnical Commission, Geneva, Switzerland, Rep. 62682, 2014.

[3] L. Pivetta *et al.*, "HDB++: A New Archiving System for TANGO", in *Proc. ICALEPCS'15*, Melbourne, Australia, Oct. 2015, pp. 652–655. doi:10.18429/JACoW-ICALEPCS2015-WED3O04

[4] Qt Framework, https://www.qt.io/product/framework