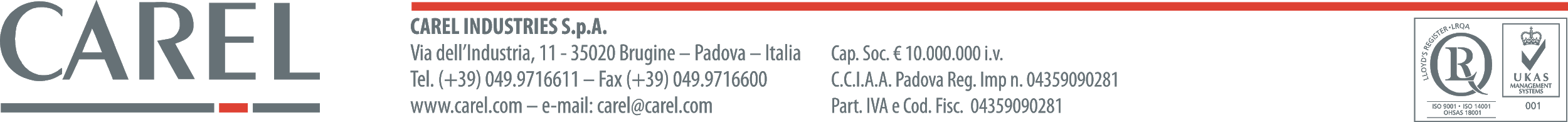
****

BOSS Family

Software Requirements Specification

**Alarms in events**

Version 1.0

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 12/06/2020 | 1.0 | first release | Schiavon J. |
| 15/06/2020 | 1.1 | added merged events + mockups | Ruvoletto T. |

**Table of Contents**

[**1. Introduction**](#_c2rhif18l43v) **3**

[1.1 Scope](#_d50sybacjom) 3

[1.2 Definitions, Acronyms and Abbreviations](#_n5evxp3yajon) 3

[1.3 References](#_yjnmrjdpvxyk) 3

[**2. Overall Description**](#_563mfco6j3m8) **3**

[**3. Specific Requirements**](#_in6v4zrfk64h) **3**

[3.1 Uses cases](#_qeqo8yvev67e) 4

[3.1.1 <Use case One>](#_8qo3ppit5cgu) 4

[3.2 Functionality](#_twenhj784ajy) 4

[3.2.1 <Functional Requirement One>](#_fh4fv69l98tk) 4

[3.3 Usability](#_qh8oh7vbt03) 4

[3.3.1 <Usability Requirement One>](#_7umcnjegabj) 4

[3.4 Reliability](#_1oi9mkjeyqg3) 4

[3.4.1 <Reliability Requirement One>](#_8prnb9ecf8h0) 4

[3.5 Performance](#_27kj7q47yjvy) 4

[3.5.1 <Performance Requirement One>](#_5urq7gj0ui5d) 5

[3.6 Security](#_nh5d315oku4m) 5

[3.7 Design Constraints](#_e0c9k3eputr4) 5

[3.7.1 <Design Constraint One>](#_v5tgm0xh50og) 5

[3.8 User Documentation](#_53r8ulwy1v7x) 5

[3.9 Interfaces](#_m2v6mtuobbfa) 5

[3.9.1 Hardware Interfaces](#_7u90ujlsyf9y) 5

[3.9.2 Software Interfaces](#_ckytmwc4dp9i) 5

[3.9.3 Communications Interfaces](#_mk3fz8w3g660) 5

[3.10 Legal, Copyright, and Other Notices](#_w7kuql2gzjs6) 5

**Software Requirements Specification**

# **1. Introduction**

*[The introduction of the* ***Software Requirements Specification (SRS)*** *should provide an overview of the entire* ***SRS****. It should include the purpose, scope, definitions, acronyms, abbreviations, references, and overview of the* ***SRS****.]*

*[Note: The Software Requirements Specification (****SRS****) captures the complete software requirements for the system, or a portion of the system. Following is a typical* ***SRS*** *outline for a project using only traditional natural-language style requirements - with* ***no use-case modeling.*** *It captures all requirements in a single document, with applicable sections inserted from the Supplementary Specifications (which would no longer be needed). For a template of an* ***SRS*** *using use-case modeling, which consists of a package containing Use-Cases of the use-case model and applicable Supplementary Specifications and other supporting information, see rup\_****SRS****-uc.dot.]*

*[Many different arrangements of an* ***SRS*** *are possible. Refer to [IEEE830-1998] for further elaboration of these explanations, as well as other options for* ***SRS*** *organization.]*

Connect an action to the alarms that have generated it is not simple, currently the events list shows when an action is managed from the dispatcher and the result of management, but to know what alarms have activated the action is not automatic.

## **1.1 Scope**

*[A brief description of the software application that the* ***SRS*** *applies to; the feature or other subsystem grouping; what Use-Case model(s) it is associated with; and anything else that is affected or influenced by this document.]*

Scope of this development is to permit to see and manage the alarms associated to a particular action and write in the events list.

## **1.2 Definitions, Acronyms and Abbreviations**

*RS:requirement specification*

Text marked in red describes functionality that is no longer used/required and should be removed.

Text marked in yellow describes functionality that is implemented without a clear specification, or that is not implemented yet because it is under discussion.

Text marked in green describes new functionality that should be added.

Text marked in orange describes functionality that is buggy/not implemented properly and it should be fixed.

## **1.3 References**

*[This subsection should provide a complete list of all documents referenced elsewhere in the* ***SRS****. Each document should be identified by title, report number (if applicable), date, and publishing organization. Specify the sources from which the references can be obtained. This information may be provided by reference to an appendix or to another document.]*

# **2. Overall Description**

*[This section of the* ***SRS*** *should describe the general factors that affect the product and its requirements. This section does not state specific requirements. Instead, it provides a background for those requirements, which are defined in detail in Section 3, and makes them easier to understand. Include such items as:*

*- product perspective*

*- product functions*

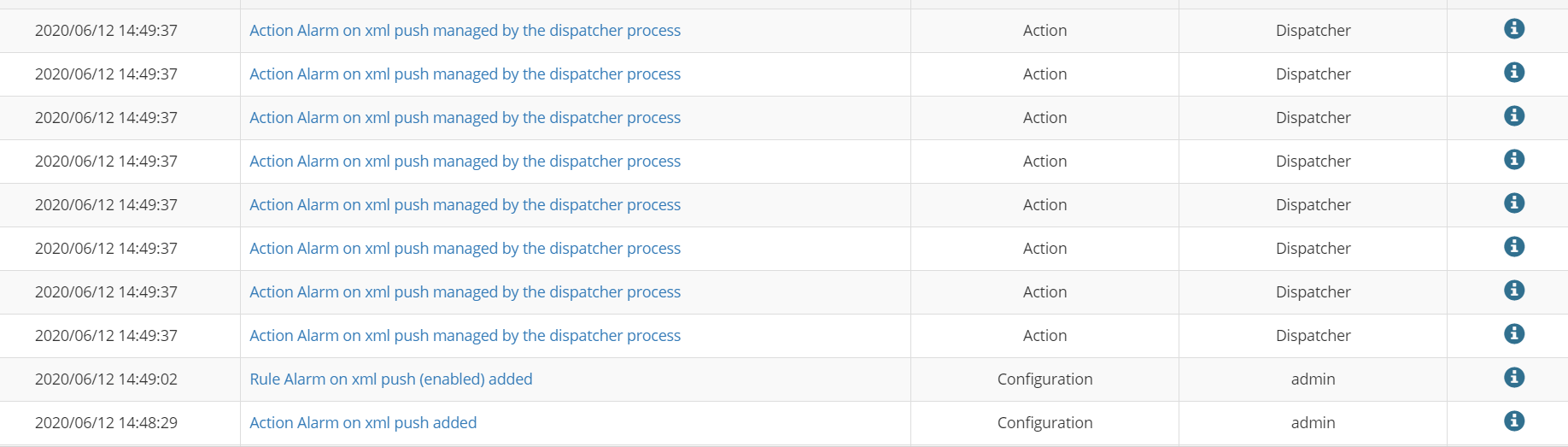
*- user characteristics*

*- constraints*

*- assumptions and dependencies*

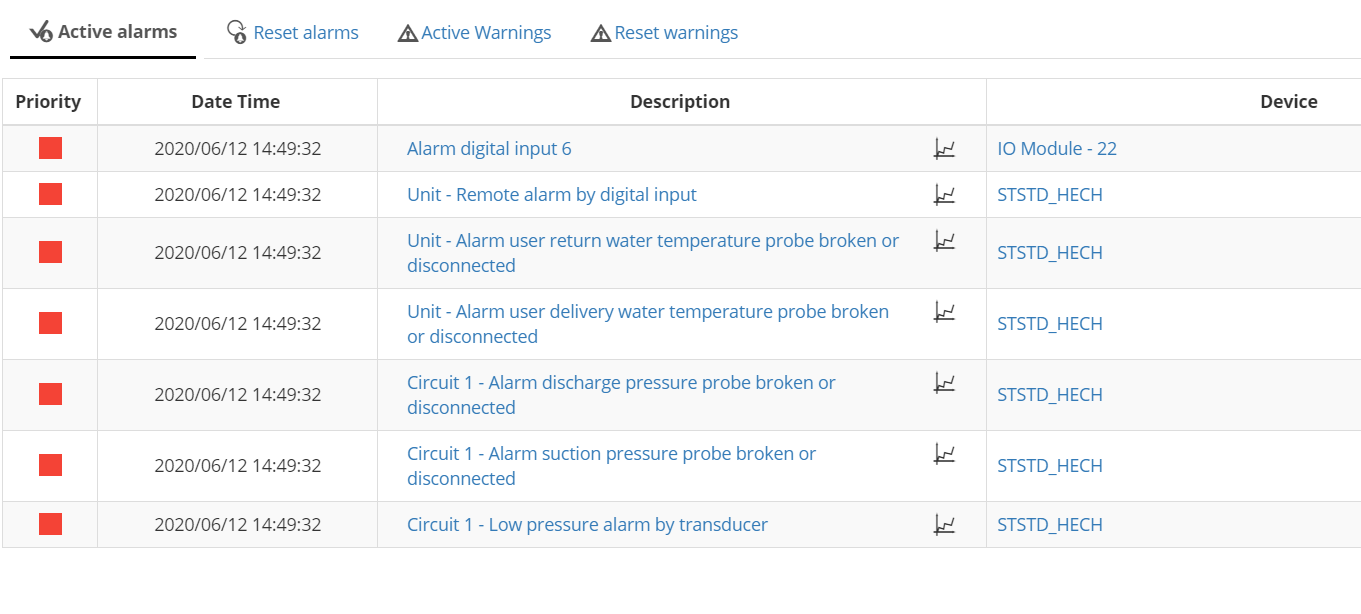
*- requirements subsets]*

Currently the events list have this aspect:

**

Any alarm produces a row when the activated action is written, but the alarm is not described.

To understand what alarm have triggered the action it’s necessary to go to the alarms page



and compare the date.

# **3. Specific Requirements**

*[This section of the* ***SRS*** *should contain all the software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. When using use-case modeling, these requirements are captured in the Use-Cases and the applicable supplementary specifications. If use-case modeling is not used, the outline for supplementary specifications may be inserted directly into this section, as shown below.]*

## **3.1 Uses cases**

### 3.1.1 New event detail

The new event details should contain the list of the alarms that have generated the row.

## **3.2 Functionality**

*[This section describes the functional requirements of the system for those requirements which are expressed in the natural language style. For many applications, this may constitute the bulk of the* ***SRS****Package and thought should be given to the organization of this section. This section is typically organized by feature, but alternative organization methods may also be appropriate, for example, organization by user or organization by subsystem. Functional requirements may include feature sets, capabilities, and security.*

*Where application development tools, such as requirements tools, modeling tools, etc., are employed to capture the functionality, this section document will refer to the availability of that data, indicating the location and name of the tool that is used to capture the data.]*

### 3.2.1 Collapse events with multiple actions

The first change to be done, that would increase this feature usefulness, is to collapse multiple events into the same event. As of today, when an alarm event is triggered, the event list looks as following:



In the example above, the condition for the action is a list of 6 alarms, and the action is **XML push**. The action was triggered by 4 alarms that happened on the same timestamp: the rule dispatcher in this case “merges the alarms” and executes the action only ONCE (the XML push is sent once with all 4 alarms together). This process is the same for the merging of alarms on the action **EMAIL**.

However, since events are added at action dispatch time, the event list will contain 4 events, that represent each “trigger” (each alarm that triggered the action), but the action is actually executed once.

The development consists in merging the event too, so that we only have **one** event entry for merged actions (scheduled action on alarms condition for email and XML push, if merge is enabled)

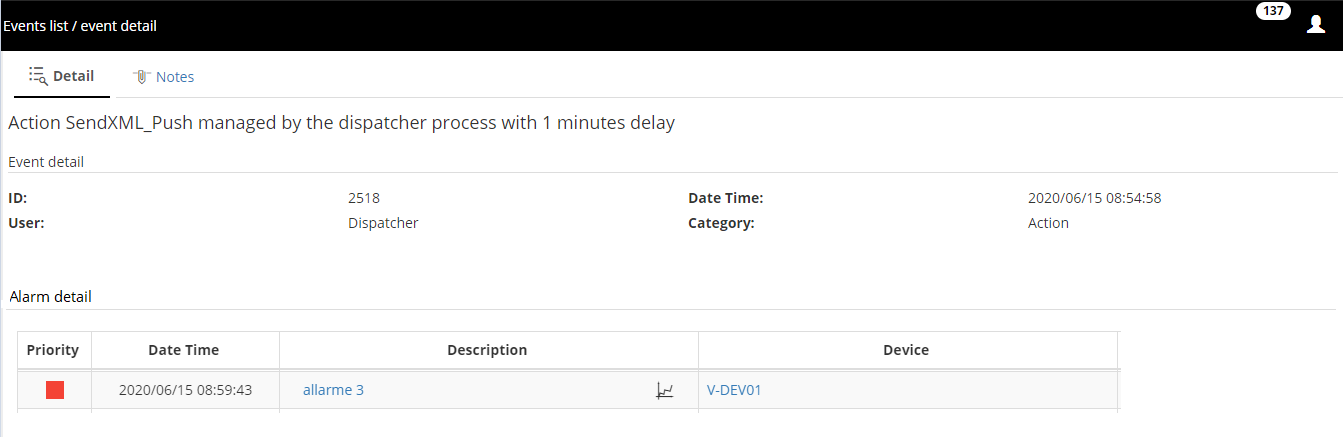


For this, it’s necessary to evaluate at rule dispatch time if the actions are merged and add an event entry in that portion of the code instead of when evaluating the action.

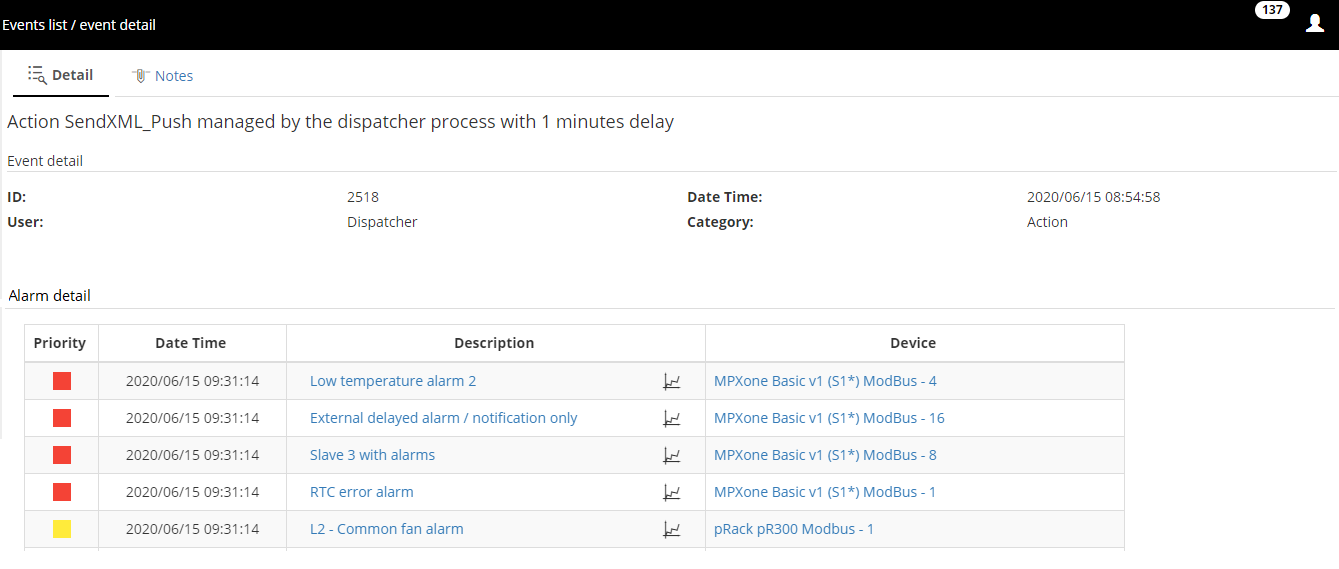
### 3.2.2 Increase level of detail of event

The second change to be done, which is the main topic of this RS, is to add to event detail page the information about the alarm(s) that triggered the rule. This means adding a table that resembles the alarm table, and this table must contain the same level of detail of the alarm list, for the inquired alarm.

This is an example for an action with 1 alarm (every action as per version 1.5.1)



This is an example for an action with multiple alarms (as for development step 3.2.1 of this RS):



The elements of each alarm entry in the table must be:

* Priority
* Date/Time
* Description (with link to trend page)
* Device

## **3.3 Usability**

*[This section should include all of those requirements that affect usability. For example,*

*- specify the required training time for a normal users and a power user to become productive at particular operations*

*- specify measurable task times for typical tasks or base the new system's usability requirements on other systems that the users know and like*

*- specify requirement to conform to common usability standards, such as IBM's CUA standards Microsoft's GUI standards]*

### 3.3.1 <Usability Requirement One>

*[The requirement description goes here.]*

## **3.4 Reliability**

*[Requirements for reliability of the system should be specified here. Some suggestions follow:*

*- Availability-specify the percentage of time available ( xx.xx%), hours of use, maintenance access, degraded mode operations, etc.*

*- Mean Time Between Failures (MTBF) - this is usually specified in hours, but it could also be specified in terms of days, months or years.*

*- Mean Time To Repair (MTTR)-how long is the system allowed to be out of operation after it has failed?*

*- Accuracy-specify precision (resolution) and accuracy (by some known standard) that is required in the system's output.*

*- Maximum Bugs or Defect Rate-usually expressed in terms of bugs per thousand of lines of code (bugs/KLOC) or bugs per function-point( bugs/function-point).*

*- Bugs or Defect Rate-categorized in terms of minor, significant, and critical bugs: the requirement(s) must define what is meant by a "critical" bug; for example, complete loss of data or a complete inability to use certain parts of the system's functionality.]*

### 3.4.1 <Reliability Requirement One>

*[The requirement description.]*

## **3.5 Performance**

*[The system's performance characteristics should be outlined in this section. Include specific response times. Where applicable, reference related Use Cases by name.*

*- response time for a transaction (average, maximum)*

*- throughput, for example, transactions per second*

*- capacity, for example, the number of customers or transactions the system can accommodate*

*- degradation modes (what is the acceptable mode of operation when the system has been degraded in some manner)*

*- resource utilization, such as memory, disk, communications, etc.*

### 3.5.1 <Performance Requirement One>

*[The requirement description goes here.]*

## **3.6 Security**

*[Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.]*

## **3.7 Design Constraints**

*[This section should indicate any design constraints on the system being built. Design constraints represent design decisions that have been mandated and must be adhered to. Examples include software languages, software process requirements, prescribed use of developmental tools, architectural and design constraints, purchased components, class libraries, etc.]*

*Applicable standards*

*[This section describes by reference any applicable standard and the specific sections of any such standards which apply to the system being described. For example, this could include legal, quality and regulatory standards, industry standards for usability, interoperability, internationalization, operating system compliance, etc.]*

*Supportability*

*[This section indicates any requirements that will enhance the supportability or maintainability of the system being built, including coding standards, naming conventions, class libraries, maintenance access, maintenance utilities.]*

### 3.7.1 <Design Constraint One>

*[The requirement description goes here.]*

## **3.8 User Documentation**

*[Describes the requirements, if any, for on-line user documentation, help systems, help about notices, etc.]*

## **3.9 Interfaces**

*[This section defines the interfaces that must be supported by the application. It should contain adequate specificity, protocols, ports and logical addresses, etc. so that the software can be developed and verified against the interface requirements.]*

### 3.9.1 Hardware Interfaces

*[This section defines any hardware interfaces that are to be supported by the software, including logical structure, physical addresses, expected behavior, etc. ]*

### 3.9.2 Software Interfaces

*[This section describes software interfaces to other components of the software system. These may be purchased components, components reused from another application or components being developed for subsystems outside of the scope of this* ***SRS*** *but with which this software application must interact.]*

### 3.9.3 Communications Interfaces

*[Describe any communications interfaces to other systems or devices such as local area networks, remote serial devices, etc.]*

## **3.10 Legal, Copyright, and Other Notices**

*[This section describes any necessary legal disclaimers, warranties, copyright notices, patent notice, wordmark, trademark, or logo compliance issues for the software.]*

*Licensing Requirements*

*[Defines any licensing enforcement requirements or other usage restriction requirements that are to be exhibited by the software.]*