



Alarms and Notifications

PRE-DRAFT

Revision	0.2
Status	Pre-Draft. A candidate document submitted for consideration.
Date	2026-10-16
Identifier	alarms-and-notifications
Authors	Trent Scholl, Project Lead, Open Aquarium Project
Editors	Trent Scholl
License	Community Specification License 1.0

Pre-Draft. This is a proposed initial draft. The Working Group has not approved it, and it will change. Comment by opening an issue with the `alarms-and-notifications` label.

1 License

This Specification is subject to the Community Specification License 1.0 (the “Specification License”), available at <https://github.com/CommunitySpecification/1.0> and in `LICENSES/Community-Spec-1.0.md`. Contributions to this Specification are made under the Community Specification Contributor License Agreement 1.0.

The conditions file, JSON Schemas, and example files published with this Specification are licensed under the Apache License 2.0 (`LICENSES/Apache-2.0.txt`).

The Scope that bounds the patent commitments made under the Specification License is set out in `Scope.md`. Exclusion notices, license acceptances, and withdrawals are recorded in `Notices.md`.

Notes:

1. Appendices are informative (non-normative) unless an appendix is explicitly marked normative.

THESE MATERIALS ARE PROVIDED “AS IS.” The Contributors and Licensees expressly disclaim any warranties (express, implied, or otherwise), including implied warranties of merchantability, non-infringement, fitness for a particular purpose, or title, related to the materials. The entire risk as to implementing or otherwise using the materials is assumed by the implementer and user. IN NO EVENT WILL THE CONTRIBUTORS OR LICENSEES BE LIABLE TO ANY OTHER PARTY FOR LOST PROFITS OR ANY FORM OF INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS DELIVERABLE OR ITS GOVERNING AGREEMENT, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND WHETHER OR NOT THE OTHER MEMBER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

2 Compliance with Open Aquarium Tenets

2.1 Openness

The Specification is published under the Community Specification License 1.0, and its conditions file, schemas, and examples under the Apache License 2.0. Every channel it defines uses an open standard that anyone can implement without permission or payment: SMTP for email, HTTP for webhooks, MQTT, and Web Push. The default severity of every condition is published in a machine-readable file, `conditions.json`, so owners and reviewers can see, and argue with, how each fault is judged.

2.2 Interoperability

Every fault that the Device Classes, Fail-Safe Behavior, and Automation specifications define has one default severity here, whichever maker’s device reports it. Alarm messages have one JSON format for webhooks, MQTT, and push, so a home automation server, a dashboard, or an app from any maker can read alarms from any controller. Alarm settings, including the owner’s ranges for each parameter, move between controllers as a document (Section 19), and alarms are kept as Data Model events (Section 20).

2.3 Local Control

The controller’s own sounder, indicators, and alarm outputs need no network at all, and every controller has one of them for critical alarms, or warns its owners until it has one (Section 14.1). Every remote channel goes to a service the owner chooses, such as the owner’s own email provider, MQTT broker, or home automation server, and no alarm depends on a maker’s service (Section 16.2). Owners can export their alarm settings and alarm history at any time.

2.4 Safety

This Specification exists to make sure that a person learns of a problem in time to act. Every fault has a defined severity, and faults of vital or hazardous equipment are critical, which no owner can turn off (Section 7). Every controller has a local signal for critical alarms, or warns its owners until it has one (Section 14.1). Critical alarms sound until someone silences them, repeat and escalate to other people until someone acknowledges them, stay until someone acknowledges them even after they clear, and are never held by quiet hours, modes, or calibrations (Section 12). A water change holds only the conditions it makes expected, and sounds those still active as soon as it ends (Section 12.8). Failed deliveries raise alarms of their own (Section 16.7), and because a failed controller cannot raise an alarm, alarm outputs, heartbeats, and MQTT's last will let other equipment notice it (Section 18).

2.5 Sustainability

Alarms on abnormal power draw, consumables, and calibration catch failing equipment early, when it can still be repaired (Section 10). No alarm needs a paid subscription, a text message service, or a maker's cloud that may close, so equipment keeps alarming for as long as it runs.

Revision History

Revision	Date	Description	Authors
0.1	2026-10-14	Initial Pre-Draft.	Trent Scholl
0.2	2026-10-16	Alarms critical through their equipment group protected as critical codes are; a local signal for critical alarms on every controller, with a standing warning until there is one; expected conditions of a water change held, never dropped, and sounded when it ends; calibrations never hold critical alarms, and stop holding others when they end; notices when a Matter fabric is added or removed; alarm outputs with the equipment category <code>alarm-signal</code> , left to their alarm function on battery; one definition of sounding; silencing devices with <code>silence</code> ; standing warnings that cannot be turned off; <code>command-refused</code> critical only for the commands that matter; <code>notification-failed</code> rising with the alarms it could not deliver; power-outage notices after any power-loss mode, with a start from a recorded time; delays that never add up; a code for settings that refer to missing elements, and identifiers for power checks; a sound level for sounders; wiring and hardware faults critical on leak sensors and on overflow and low-level switches; a test procedure, mapped to its requirements, for every requirement; references cited by tag; one sentence per line.	Trent Scholl

Contents

1	License	2
2	Compliance with Open Aquarium Tenets	2
2.1	Openness	2
2.2	Interoperability	2
2.3	Local Control	2
2.4	Safety	3
2.5	Sustainability	3
	Revision History	3
3	Scope	7
4	References	7
4.1	Normative References	7
4.2	Informative References	8
5	Terms, Definitions, and Conventions	9
5.1	Requirements Language	9
5.2	Terms and Definitions	9
5.3	Notation	10
6	Introduction	10
6.1	Purpose	10
6.2	Design Principles	10
6.3	Overview	11
6.4	Example	11
7	Severities	12
7.1	Severity Levels	12
7.2	Equipment Groups	12
7.3	Assigning a Severity	13
8	Alarm Conditions	14
8.1	Codes	14
8.2	Subjects	14
8.3	One Alarm per Condition and Subject	14
8.4	Faults Reported by Devices	15
8.5	Conditions of the System and the Controller	16
8.6	Automation Faults	17
8.7	Leaks, Levels, Ranges, Power, and Custom Alarms	18
8.8	Consequences	19
9	Range Alarms	19
9.1	Ranges	19
9.2	Zones	20
9.3	Which Values Count	21
9.4	Raising and Clearing	22
9.5	Ranges and Control	22
9.6	Ranges in the Data Model	22
10	Equipment Alarms from Power Draw	22
10.1	Faults Reported by Devices	22
10.2	Power Checks	23
10.3	Default Checks and Learning	23
10.4	Equipment That Switches Itself	23
11	Custom Alarms	24
12	Alarm Behavior	24
12.1	States	24
12.2	Raising	25
12.3	Clearing and Latching	25
12.4	Acknowledgement	26
12.5	Silencing	26
12.6	Repeats and Reminders	26
12.7	Grouping and Floods	26
12.8	Modes, Water Changes, and Calibrations	27

12.9	Restarts and the Clock	27
12.10	Visibility and Permissions	27
13	Notices	28
13.1	Notices	28
13.2	Power Outages	28
13.3	Restarts	29
13.4	Other Notices	29
14	Local Signals	29
14.1	Sound	29
14.2	Indicators	30
14.3	Buttons	30
14.4	Signals on Devices	30
15	Alarm Outputs	31
15.1	The Alarm Function	31
15.2	Fail-Safe Configuration of Alarm Outputs	31
15.3	Example	32
16	Notifications	32
16.1	Recipients	32
16.2	Channels	32
16.3	Escalation Policies	33
16.4	Rules	33
16.5	Quiet Hours	34
16.6	Messages	34
16.7	Delivery and Failures	35
16.8	Tests	36
16.9	Content and Privacy	36
17	Channels	36
17.1	Apps	36
17.2	Push	36
17.3	Email	37
17.4	Webhooks	37
17.5	MQTT	38
17.6	Other Channels, Text Messages, and Calls	38
18	Supervising the Alarm Path	39
18.1	A Failed Controller Cannot Raise an Alarm	39
18.2	Heartbeats	39
19	Alarm Settings Documents	39
19.1	Encoding	39
19.2	Top-Level Members	39
19.3	Identifiers and References	40
19.4	Secrets	40
19.5	Export and Import	40
20	History	41
20.1	Alarm Records	41
20.2	Notices in History	41
20.3	Proposed Data Model Changes	41
20.4	Keeping History	41
21	Conformance	41
21.1	Controllers	41
21.2	Devices	42
21.3	Apps	42
21.4	Documents and Messages	42
21.5	Machine-Readable Definitions and Tools	42
21.6	Test Procedures	42
22	Security Considerations	46
Appendix A — Approval Checklist		46
Appendix B — Scenarios (Informative)		46
Appendix C — Commercial Alarm Features (Informative)		47

Appendix D — Open Issues (Informative) 48

3 Scope

This Specification defines:

- severity levels, and a default severity for every fault code of the Device Classes specification, every fault of the Automation specification, and every condition of the Fail-Safe Behavior specification;
- alarms from conditions of the system and the controller, from leak and level sensors, from the owner's desired, acceptable, and danger ranges for each parameter, from abnormal power draw, and from conditions the owner defines;
- how alarms are raised, acknowledged, silenced, cleared, repeated, and escalated;
- notices of power outages, restarts, and other events, kept until someone acknowledges them;
- local signals, and alarm outputs that switch equipment;
- notifications to people through apps, push, email, and webhooks, feeds to other systems through webhooks and MQTT, escalation policies, and quiet hours;
- supervision of the alarm path itself;
- a portable document for alarm settings, and a JSON format for alarm messages; and
- how alarms are kept in history.

This Specification does not define:

- how devices detect faults, which the Device Classes specification and the specifications of each kind of equipment define;
- what equipment does in response to a fault, a leak, or a low level, which the Automation, Fail-Safe Behavior, and Water Level and Water Change specifications define;
- how apps talk to a controller, which a controller API is to define [OAP-DP Appendix C, issue 1];
- who may do what, beyond naming the operations of the Security and Access specification that apply; or
- recommended values for water parameters, because the ranges are the owner's.

4 References

4.1 Normative References

- **[RFC 2119]** Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- **[RFC 8174]** Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017.
- **[RFC 8259]** Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, December 2017.
- **[RFC 3339]** Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.
- **[RFC 9562]** Davis, K., Peabody, B., and P. Leach, "Universally Unique IDentifiers (UUIDs)", RFC 9562, May 2024.
- **[RFC 4648]** Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.
- **[RFC 2104]** Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.
- **[FIPS 180-4]** National Institute of Standards and Technology, "Secure Hash Standard (SHS)", FIPS PUB 180-4, August 2015.
- **[RFC 9110]** Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, June 2022.
- **[RFC 8030]** Thomson, M., Damaggio, E., and B. Raymor, Ed., "Generic Event Delivery Using HTTP Push", RFC 8030, December 2016.
- **[RFC 8291]** Thomson, M., "Message Encryption for Web Push", RFC 8291, November 2017.
- **[RFC 8292]** Thomson, M. and P. Beverloo, "Voluntary Application Server Identification (VAPID) for Web Push", RFC 8292, November 2017.
- **[RFC 5322]** Resnick, P., Ed., "Internet Message Format", RFC 5322, October 2008.

- **[RFC 3834]** Moore, K., "Recommendations for Automatic Responses to Electronic Mail", RFC 3834, August 2004.
- **[RFC 6409]** Gellens, R. and J. Klensin, "Message Submission for Mail", STD 72, RFC 6409, November 2011.
- **[RFC 8314]** Moore, K. and C. Newman, "Cleartext Considered Obsolete: Use of Transport Layer Security (TLS) for Email Submission and Access", RFC 8314, January 2018.
- **[MQTT 5]** OASIS, "MQTT Version 5.0", OASIS Standard, 7 March 2019, <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html>.
- **[MQTT 3.1.1]** OASIS, "MQTT Version 3.1.1", OASIS Standard, 29 October 2014, with Errata 01, <https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html>.
- **[Standard Webhooks]** Standard Webhooks, "Standard Webhooks Specification", version 1.0.0, <https://github.com/standard-webhooks/standard-webhooks/blob/main/spec/standard-webhooks.md>.
- **[JSON Schema]** "JSON Schema: A Media Type for Describing JSON Documents" and "JSON Schema Validation", draft 2020-12, <https://json-schema.org/draft/2020-12>.
- **[TZDB]** Internet Assigned Numbers Authority, "Time Zone Database", <https://www.iana.org/time-zones>.
- **[OAP-DM]** Open Aquarium Project, "Open Aquarium Data Model", Revision 0.3, [specs/data-model/spec.md](https://open-aquarium.org/specs/data-model/spec.md).
- **[OAP-DC]** Open Aquarium Project, "Open Aquarium Device Classes", Revision 0.3, [specs/device-classes/spec.md](https://open-aquarium.org/specs/device-classes/spec.md).
- **[OAP-FS]** Open Aquarium Project, "Open Aquarium Fail-Safe Behavior", Revision 0.2, [specs/fail-safe-behavior/spec.md](https://open-aquarium.org/specs/fail-safe-behavior/spec.md).
- **[OAP-AU]** Open Aquarium Project, "Open Aquarium Automation", Revision 0.2, [specs/automation/spec.md](https://open-aquarium.org/specs/automation/spec.md).
- **[OAP-SA]** Open Aquarium Project, "Open Aquarium Security and Access", Revision 0.2, [specs/security-and-access/spec.md](https://open-aquarium.org/specs/security-and-access/spec.md).
- **[OAP-DP]** Open Aquarium Project, "Open Aquarium Device Protocol", Revision 0.2, [specs/device-protocol/spec.md](https://open-aquarium.org/specs/device-protocol/spec.md).

4.2 Informative References

- **[OAP-MB]** Open Aquarium Project, "Open Aquarium Module Bus", Revision 0.3, [specs/module-bus/spec.md](https://open-aquarium.org/specs/module-bus/spec.md).
- **[ISA-18.2]** ANSI/ISA-18.2-2016, "Management of Alarm Systems for the Process Industries". Its alarm states, shelving, chattering, and flood are the background of Section 12.
- **[IEC 62682]** IEC 62682:2022, "Management of alarm systems for the process industries". It is the international standard that corresponds to [ISA-18.2].
- **[IEC 60601-1-8]** IEC 60601-1-8:2006+AMD1:2012+AMD2:2020, "Medical electrical equipment — Part 1-8: General requirements, tests and guidance for alarm systems". Its colors and flashing rates for alarm indicators are the background of Section 14.2.
- **[CAP]** OASIS, "Common Alerting Protocol Version 1.2", OASIS Standard, 1 July 2010.
- **[UnifiedPush]** UnifiedPush, "UnifiedPush Specifications", <https://unifiedpush.org/>.
- **[OAP-WL]** Open Aquarium Project, "Open Aquarium Water Level and Water Change", Revision 0.2, [specs/water-level-and-water-change/spec.md](https://open-aquarium.org/specs/water-level-and-water-change/spec.md).
- **[OAP-DO]** Open Aquarium Project, "Open Aquarium Dosing", Revision 0.2, [specs/dosing/spec.md](https://open-aquarium.org/specs/dosing/spec.md).
- **[OAP-WT]** Open Aquarium Project, "Open Aquarium Water Testing", Revision 0.2, [specs/water-testing/spec.md](https://open-aquarium.org/specs/water-testing/spec.md).
- **[OAP-FP]** Open Aquarium Project, "Open Aquarium Flow Programs", Revision 0.2, [specs/flow-programs/spec.md](https://open-aquarium.org/specs/flow-programs/spec.md).
- **[OAP-LP]** Open Aquarium Project, "Open Aquarium Lighting Programs", Revision 0.2, [specs/lighting-programs/spec.md](https://open-aquarium.org/specs/lighting-programs/spec.md).
- **[OAP-PI]** Open Aquarium Project, "Open Aquarium Probe Interface", Revision 0.3, [specs/probe-interface/spec.md](https://open-aquarium.org/specs/probe-interface/spec.md).
- **[OAP-MM]** Open Aquarium Project, "Open Aquarium Matter Mapping", Revision 0.2, [specs/matter-mapping/spec.md](https://open-aquarium.org/specs/matter-mapping/spec.md).

5 Terms, Definitions, and Conventions

5.1 Requirements Language

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “NOT RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in BCP 14 [RFC 2119] [RFC 8174] when, and only when, they appear in all capitals, as shown here.

5.2 Terms and Definitions

Terms defined in the Device Classes specification [OAP-DC], such as device, endpoint, fault, and controller, in the Fail-Safe Behavior specification [OAP-FS], such as fail-safe behavior and unavailable input, in the Automation specification [OAP-AU], such as output, input, function, mode, and override, and in the Security and Access specification [OAP-SA], such as owner, user, role, operation, app, and service, have the same meaning here.

Condition: In this Specification, something that is true for a time, such as a fault, a leak, or a value outside a range, and that can raise an alarm. Where Automation conditions are meant, this Specification says so.

Code: The name of a kind of condition or notice, such as `stuck-on` or `power-outage` (Section 8.1).

Subject: What an alarm is about: a device, an endpoint, an element of the automation configuration, a range, a custom alarm, a channel, or the controller (Section 8.2).

Alarm: The controller’s record that a condition with a given code and subject has arisen and needs a person, from the time it is raised until it closes (Section 12).

Notice: An alarm about something that happened, such as a power outage, rather than about a condition that is still true (Section 13).

Severity: How urgently an alarm needs a person: `critical`, `warning`, or `info` (Section 7).

Protected alarm: An alarm that is critical by default or through its equipment group, which no owner can turn off or lower below `warning` (Section 7.3).

Standing warning: An alarm that stays shown to every owner while its condition lasts, and that no owner can turn off, such as a warning that a safety function has no path that uses no radio link (Section 7.3; Fail-Safe Behavior Section 13.4.6).

Raise: To create an alarm when its condition arises.

Clear: What an alarm does when its condition ends.

Acknowledge: To record that a person knows about an alarm (Section 12.4).

Silence: To stop the sound of alarms for a while, without acknowledging them (Section 12.5).

Sounding: The state of an alarm whose local sound and sounding alarm outputs are on: unacknowledged, not silenced, not held, and either active or a notice (Section 12.5).

Latching: Staying until acknowledged, even after the condition has cleared.

Close: To end an alarm, once it has cleared and, where it latches, been acknowledged.

Equipment group: A set of Data Model equipment categories whose faults are critical: `vital` or `hazardous` (Section 7.2).

Range: The owner’s desired range, acceptable range, and danger limits for one parameter in one place (Section 9).

Local signal: A sound, light, or display on the controller itself (Section 14).

Local signal for critical alarms: A sounder built into the controller, or an alarm output whose alarm function signals every critical alarm (Section 14.1).

Alarm output: An output of the automation configuration that the controller switches to signal alarms, such as a siren or a contact to a house alarm panel (Section 15).

Recipient: A person, or a service acting for people, whom notifications reach (Section 16.1).

Channel: A way of delivering alarm messages: apps, email, a webhook, MQTT, or an extension (Section 16.2).

Notification: A message about an alarm sent to a recipient.

Feed: A channel that reports every change of every alarm to another system, rather than notifying a person (Section 16.2).

Escalation policy: Who is notified about an alarm, through which channels, and when, until someone acknowledges it (Section 16.3).

Quiet hours: Times when warnings and information are held, but critical alarms are not (Section 16.5).

Hold: To delay a sound or a notification until quiet hours, a mode, a water change, or a calibration end (Sections 12.8 and 16.5).

Expected condition: A condition that a water change makes expected in one of its phases, as the Water Level and Water Change specification names them, and whose alarm it holds until it ends (Section 12.8).

Alarm message: The JSON object that carries an alarm, or a change to one, through a channel (Section 16.6).

Maker's service: A service on the internet run by, or for, the manufacturer of the controller, a device, or an app.

Heartbeat: A message the controller sends at regular intervals, so that another system notices when the messages stop (Section 18.2).

5.3 Notation

JSON members, codes, severities, and the names of attributes, commands, and faults appear in `monospace`. Durations are in seconds unless stated. Timestamps are Data Model Timestamps: RFC 3339 dates and times with a UTC offset [RFC 3339] [OAP-DM Section 7.4]. References to other Open Aquarium specifications name them, as in "Fail-Safe Behavior Section 7.6", or use their tags from Section 4, as in "[OAP-FS Section 7.6]". A section number with a third part refers to a numbered item of that section: "Section 12.4.2" means item 2 of Section 12.4. Numbers marked provisional are to be confirmed with the breadboard controller and owners' experience (Appendix D).

6 Introduction

6.1 Purpose

Aquarium equipment fails at night, while the owner is at work, and while the owner is on holiday. Fail-safe behavior keeps the damage small, but a heater whose relay has welded shut, a leak, or a return pump running dry still needs a person, and soon.

Four widely used commercial controller ecosystems all alarm, but each in its own way. Each maker decides how serious a fault is, and none escalates an alarm to another person when the owner does not respond. Several depend on the maker's cloud to deliver alerts, and two of the four have dropped text messages.

This Specification defines alarms once, for every controller and every maker's equipment: how serious each condition is, how the owner is told, and how the controller keeps trying until someone responds.

6.2 Design Principles

1. **Every condition has a severity.** Every fault of every specification has a default severity, and faults of vital or hazardous equipment are critical.
2. **Someone is told, or the controller keeps trying.** Critical alarms repeat and escalate until someone acknowledges them, and a failed delivery is an alarm of its own.
3. **Critical means critical.** Quiet hours, modes, calibrations, and settings never hold a critical alarm, and nobody can turn one off. A water change holds only the conditions it makes expected, and only until it ends.
4. **Few, meaningful alarms.** Delays, hysteresis, grouping, and latching only where it matters keep alarms rare enough that owners do not learn to ignore them [ISA-18.2] [IEC 62682].
5. **Silencing is not acknowledging.** Stopping a buzzer does not tell anyone that a person has dealt with the problem.

- Local first.** The controller's own buzzer and alarm outputs work without a network, and remote channels use services the owner chooses, never only a maker's service.
- A failed controller cannot raise an alarm.** So other equipment watches it: alarm outputs sound when the controller stops, and heartbeats and MQTT's last will tell other systems when it goes quiet.

6.3 Overview

Conditions come from devices, from the controller, from the automation configuration, and from the owner's ranges, power checks, and custom alarms. Each raises an alarm with a severity. The alarm sounds locally, switches alarm outputs, and is sent to recipients by an escalation policy, and to feeds, until someone acknowledges it. When it closes, it is kept in history.

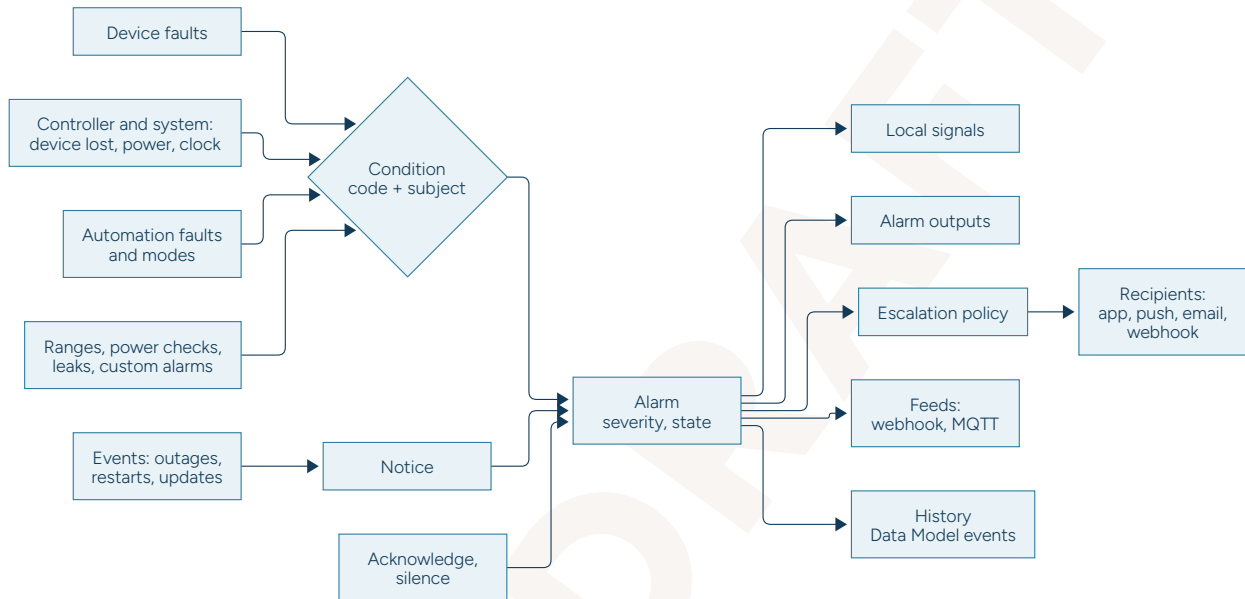


Figure 1 — From conditions to people

Table 1 — Where alarms come from

Source	Examples	Detected by	Section
Faults of devices	stuck-on, dry-run, empty, communication-lost	The device, in its faults	8.4
The system and the controller	device-lost, mains-power-lost, clock-invalid, stop-all	The controller	8.5
The automation configuration	input-unavailable, max-on-time, emergency-mode	The controller	8.6
Leak and level sensors	leak, overflow, reservoir-low	The controller, from endpoint states	8.7
The owner's ranges	outside-acceptable-range, in-danger-range	The controller, from readings and test results	9
Power draw	stuck-on, load-disconnected, power-abnormal	The device or the controller	10
The owner's own conditions	owner-defined	The controller	11
Events	power-outage, controller-restarted, firmware-refused	The controller	13

6.4 Example

A heater's relay welds shut at 21:40 on a weeknight.

- The power strip reports `stuck-on` on the heater's outlet. The outlet's `loadCategory` is `heater`, which is hazardous equipment, so the alarm is `critical` (Section 7.2).
- The controller's buzzer starts its critical pattern, its alarm light flashes red, and a siren on a module output sounds (Sections 14 and 15).

3. The critical escalation policy sends push notifications to Alex and Sam at once (Section 16.3). Nobody responds, so the notifications repeat every 10 minutes, and at 21:50 Alex also gets an email.
4. At 21:52 Alex acknowledges the alarm in the app. The repeats stop, and Sam gets a message that Alex has acknowledged it.
5. At 22:05 Alex unplugs the heater. The fault clears, the alarm closes, and it is kept in history as a Data Model event with the time Alex acknowledged it (Section 20).

Had nobody acknowledged it by 22:10, Jordan next door would have received an email and a text message.

7 Severities

7.1 Severity Levels

Every alarm has one of the severities in Table 2. Their number and names are provisional (Appendix D, issue 1).

Table 2 — Severity levels

Severity	Meaning	Respond within	Local sound	Notified	Repeated until acknowledged	Reminders	Stays until acknowledged	Held by quiet hours
critical	Livestock, equipment, or property is at risk now, or soon will be.	Minutes	Continuous	Yes	Every 10 min	Every 4 h	Yes	Never
warning	Something is wrong and needs a person, but nothing is at immediate risk.	Hours	Intermittent	Yes	Every 4 h	Every 24 h	No	Yes
info	Something needs attention when convenient, such as a calibration that is due.	Days	None	No; shown in apps	—	—	No	Yes

1. The columns after “Respond within” are defaults, which the owner’s settings may change within the limits of this Specification.
2. A **critical** alarm **MUST** always be signaled locally, by the controller’s local signal for critical alarms, which every controller has or warns its owners that it lacks (Section 14.1), **MUST** be notified to at least the system’s owners unless an owner has set otherwise (Section 16.4), and **MUST NOT** be held by quiet hours, modes, or calibrations (Sections 12.8 and 16.5). Only a water change holds one, and only while the condition is expected (Section 12.8).
3. Repeat and reminder intervals are provisional.

7.2 Equipment Groups

Many faults matter much more on some equipment than on others. A **stuck-on** light wastes power; a **stuck-on** heater can kill livestock within hours.

Table 3 — Equipment groups

Group	Meaning	Equipment categories
vital	Equipment whose stopping can harm livestock within hours: water movement, gas exchange, and temperature control.	return-pump, circulation-pump, air-pump, heater, chiller
hazardous	Equipment that can harm livestock, or flood a room, when it runs without control.	heater, chiller, doser, co2-system, ozone-generator, reactor, auto-top-off, ro-di, water-changer

- The equipment categories of an alarm are:
 - for an endpoint, its `loadCategory`, and the Data Model categories of the equipment of every Automation output bound to it;
 - for a device, those of all its endpoints;
 - for an Automation output, the category of its equipment;
 - for a function, interlock, or mode, those of the outputs it acts on; and
 - for an input, those of the outputs of every function, interlock, and mode that uses it.
- An alarm whose condition lists a group in its “Critical for” column, and whose categories include a category of that group, is `critical`, except as Section 8.6.4 says for `command-refused`.
- The groups are provisional, and need review by experienced aquarists, as the recommended fail-safe defaults do (Appendix D, issue 2).

7.3 Assigning a Severity

A controller gives each alarm, when it raises it:

- the severity that the owner has set for its code and subject (item 4), if any, but no lower than `warning` for a protected alarm (item 5);
- otherwise, `critical`, if item 2 of Section 7.2, or Section 8.4.3, applies;
- otherwise, the condition’s default severity in Tables 6 to 9 and 15 and `conditions.json`, or, for a code of another specification or an extension, the severity that defines.

The severity does not change while the alarm lasts, except that of `notification-failed` (Section 8.5.6). A condition that becomes more serious, such as a value that moves into the danger range, raises a further alarm with its own code (Section 9.2).

- An owner MAY change the severity of a code, for all subjects or for one, with a condition setting (Table 4). A condition setting MAY also turn a code off, and MAY change its raise and clear delays (Section 12.2).
- A protected alarm is one whose code’s default is `critical`, or that item 2 of Section 7.2 makes critical through its equipment group, or Section 8.4.3 makes critical on a sensor. No setting turns a protected alarm off or lowers it below `warning`: a setting that turns its code off does not apply to it, and one that sets `info` gives it `warning`. This holds whenever an alarm is raised, so a setting made before an output was bound to a heater does not turn off the heater’s `stuck-on`.
- A controller MUST NOT accept a setting that turns off a code whose default is `critical`, or that turns off, or lowers below `warning`, a code for a subject whose alarms are critical through their equipment group or by Section 8.4.3. It MUST ask an owner to confirm a setting that lowers a protected alarm, and MUST show such settings to every owner.
- A controller MUST NOT accept a setting that turns off, or changes the severity of, a standing warning: `safety-path-missing` [OAP-AU Section 13.7] or `local-signal-missing` (Section 14.1). A standing warning stays shown to every owner while it is active, even once someone has acknowledged it.
- A setting for one subject takes precedence over a setting for every subject.

Table 4 — Members of a condition setting

Member	Type	Presence	Description
<code>code</code>	string	REQUIRED	The code.
<code>subject</code>	object	OPTIONAL	Limits the setting to one subject: a <code>device</code> , an <code>endpoint</code> of it, or an automation <code>element</code> .
<code>severity</code>	string	OPTIONAL	The severity of the code’s alarms.

Member	Type	Presence	Description
off	true	OPTIONAL	The code raises no alarms, except protected alarms (Section 7.3.5). Not with <code>severity</code> , not for a code whose default is <code>critical</code> , and not for a standing warning.
delay	number (s)	OPTIONAL	How long the condition must last before the alarm is raised (Section 12.2).
clearDelay	number (s)	OPTIONAL	How long the condition must be absent before the alarm clears (Section 12.3).
notes	string	OPTIONAL	Free text.

8 Alarm Conditions

8.1 Codes

1. Each kind of condition and notice has a code: a registry name, made of lowercase letters, digits, and single hyphens, or an extension name, as defined in Data Model Section 7.6.
2. The codes of Device Classes faults [OAP-DC Section 8.4] and of Automation faults [OAP-AU Section 17.9] are used unchanged.
3. `conditions.json` lists every code this Specification knows, with its default severity, the groups for which it is critical, and whether a device or the controller detects it. It is normative, and Tables 2, 3, 6 to 9, and 15 are generated from it.
4. Other Open Aquarium specifications, such as Water Level and Water Change [OAP-WL], Dosing [OAP-DO], and Water Testing [OAP-WT], add their codes to `conditions.json`, and show them in their own tables, such as Water Level and Water Change Table 5.
5. A controller MUST raise alarms for every code of `conditions.json` whose condition it can detect. A controller MAY raise alarms for extension codes. An extension fault code that a controller does not understand raises no alarm, as Device Classes Section 7.9 requires, but the controller SHOULD show it with the device.

8.2 Subjects

Table 5 — Subjects

Kind	Members	Used for
controller	—	Conditions of the controller and its system as a whole, such as <code>mains-power-lost</code> .
device	device	Faults in a device's own <code>faults</code> , and <code>device-lost</code> .
endpoint	device, endpoint	Faults in an endpoint's <code>faults</code> ; leaks, levels, and power checks.
element	element	Automation faults, which name the element they concern [OAP-AU Section 17.9]; <code>emergency-mode</code> ; and the faults that functions of other specifications raise, such as a top-off's <code>timeout</code> (Water Level and Water Change Section 7.10).
range	range	Range alarms (Section 9).
power-check	powerCheck	Power checks' <code>reference-missing</code> (Section 19.3). Alarms that power checks raise are about the endpoint (Section 10.2).
custom	custom	Custom alarms (Section 11), and their <code>reference-missing</code> .
channel	channel	<code>notification-failed</code> (Section 16.7).

device is a device identifier as Automation Section 8.5 defines it, such as `module-bus:4f41503a000000010000a1b2c3d4e5f6`, `device-protocol`: followed by a key ID, or `controller:self` for the controller's own endpoints.

8.3 One Alarm per Condition and Subject

1. A controller MUST NOT have more than one alarm that is not closed for the same code and subject.
2. When a condition arises again while its earlier alarm has cleared but not closed, because it latches, the earlier alarm becomes active again.

3. When a device and the controller both detect a condition, such as `stuck-on` on an outlet whose power both measure (Section 10), the alarm's subject is the endpoint, so they raise one alarm.

8.4 Faults Reported by Devices

Every fault code that a device lists in its own `faults` or in an endpoint's `faults` [OAP-DC Section 7.6] raises an alarm, with the device or the endpoint as its subject. The alarm is active while the device reports the fault.

Table 6 — Default severities of Device Classes faults

Fault code	Meaning	Default	Critical for	Notes
<code>communication-lost</code>	No contact with the controller for longer than the fail-safe timeout.	warning	vital	The device's outputs are in their fail-safe behavior. Its controller usually raises <code>device-lost</code> first, and sees this fault only when contact returns; other sessions, such as an app's, see it at once.
<code>hardware-failure</code>	An internal hardware fault.	warning	vital, hazardous	The endpoint's output is likely in its unpowered state (Fail-Safe Behavior Section 11.3). Critical on a leak sensor, and on an overflow or low-level switch (Section 8.4.3).
<code>overcurrent</code>	Current above the rated or configured limit.	warning	vital, hazardous	Overload protection may have switched the output off. A short circuit in water needs a person.
<code>stuck-on</code>	An output draws current or keeps running while it is commanded off, for example because a relay contact has welded shut.	warning	hazardous	A heater whose relay has welded shut is one of the most dangerous failures in an aquarium. Controllers also detect it from power draw (Section 10).
<code>load-disconnected</code>	An output is on but its load draws no current, for example because equipment is unplugged or a solenoid coil has failed.	warning	vital	Equipment with its own thermostat draws nothing while it is satisfied (Section 10.4). Controllers also detect it from power draw.
<code>overtemperature</code>	The device or a part of it is too hot.	warning	vital, hazardous	A light dims itself; a heater out of the water, or a pump that is blocked, needs a person.
<code>undervoltage</code>	Supply voltage too low.	warning	vital	The device may restart or hold its outputs (Fail-Safe Behavior Section 9.3).
<code>overvoltage</code>	Supply voltage too high.	warning	vital	
<code>wiring-fault</code>	A supervised input detected an open or short circuit in its wiring.	warning	—	The input is unknown, so a leak or a high level would go unseen. Functions that use it raise <code>input-unavailable</code> . Critical on a leak sensor, and on an overflow or low-level switch (Section 8.4.3).
<code>sensor-disconnected</code>	A sensor or probe is not connected.	warning	vital	Critical for the built-in sensor of a heater or chiller. For a probe, functions that use it raise <code>input-unavailable</code> .
<code>out-of-range</code>	A reading is outside the range the sensor can measure.	warning	—	Usually a probe out of the water or broken. Range alarms judge the water itself (Section 9).
<code>calibration-due</code>	Calibration is due or overdue.	info	—	
<code>calibration-failed</code>	The last calibration attempt failed.	warning	—	
<code>expired</code>	A consumable or probe has passed its end of life or expiry date.	info	—	
<code>stall</code>	A motor is blocked or cannot turn.	warning	vital	A stalled return pump stops filtration and gas exchange.
<code>dry-run</code>	A pump is running without water.	warning	vital	A return pump running dry means the sump is low or empty. A top-off pump running dry means its reservoir is empty.

Fault code	Meaning	Default	Critical for	Notes
jam	A mechanism is jammed.	warning	—	
low	A container, reservoir, supply, or consumable is low.	info	—	Containers, food, and reagents usually last days after low.
empty	A container, reservoir, supply, or consumable is empty.	warning	—	Dosing, feeding, testing, or top-off has stopped.
waste-full	A waste container is full.	warning	—	
end-of-roll	A filter roll is used up.	warning	—	
timeout	An operation exceeded its maximum allowed time.	warning	hazardous	A fill that takes too long may be sending water somewhere else.
high-level	Water reached a high or overflow sensor.	critical	—	The water may soon overflow.
implausible-result	A measurement result failed a plausibility check.	warning	—	
sample-failure	A sample could not be drawn or analyzed.	warning	—	
battery-weak	The device's battery is nearly exhausted, and needs charging or replacing soon.	warning	—	A battery-powered sensor, such as a leak sensor, stops reporting when its battery runs out, and the functions that use it lose an input.

1. A controller MUST raise an alarm for a fault that a device reported before the controller lost it, and keep it active while the device is lost, because it cannot tell whether the fault has ended.
2. A fault that a class says stays until cleared by a command, such as a top-off unit's latched faults, is cleared with that command (`clear-latch`; Security and Access Table 6), not by acknowledging the alarm.
3. `wiring-fault` and `hardware-failure` are `critical` on a `leak-sensor` endpoint, and on a `level-switch` endpoint whose `position` is `overflow` or `low`, whatever their default. Either fault leaves the sensor's state unknown (Section 8.7.4), so a leak, an overflow, or a low level would go unseen, and the safety function that the sensor serves no longer works [OAP-AU Section 13.7]. Equipment groups cannot make them critical, because a sensor drives no output.

8.5 Conditions of the System and the Controller

Table 7 — Conditions of the system and the controller

Code	Meaning	Default	Critical for	Notes
device-lost	The controller has received nothing from a device for longer than its binding allows (Fail-Safe Behavior Section 7.6; Module Bus Section 12; Device Protocol Section 12.5).	warning	vital	Raised after 300 s. Its inputs are unavailable and its outputs are in their fail-safe behavior. The delay rides out short network outages, while range alarms watch the water.
clock-invalid	A clock is not valid (Fail-Safe Behavior Section 10.5): the controller's, or a device's, which the device reports as a Device Classes fault.	warning	—	Schedules are unknown, programs are held, and remote channels that need the time may fail.
mains-power-lost	The controller has detected the loss of mains power, and is in power-loss mode (Fail-Safe Behavior Section 9.5).	critical	—	Raised after 60 s. Only controllers that detect power loss raise it. The delay rides out brief interruptions.
battery-low	The controller is running on its backup battery, and the battery will soon be exhausted.	critical	—	
battery-fault	The controller's backup battery is missing, is not charging, or failed a test.	warning	—	The next power cut will stop everything.

Code	Meaning	Default	Critical for	Notes
<code>configuration-failed</code>	The controller could not configure a device, for example because its description is invalid or a register write failed, and does not command it (Module Bus Section 10).	warning	vital	
<code>node-conflict</code>	Two nodes on the module bus use one node-ID, or a node other than the controller acts as a controller (Security and Access Table 8).	critical	—	Modules may follow the wrong commands.
<code>capacity-exceeded</code>	The controller cannot take on another device or element because it has reached one of its limits, such as its number of modules.	warning	—	
<code>stop-all</code>	A person has stopped every output with <code>stop-all</code> (Security and Access Section 11.8).	critical	—	The return pump, heaters, and everything else are stopped until it ends.
<code>notification-failed</code>	The controller could not deliver notifications through a channel (Section 16.7).	warning	—	Takes the severity of the most severe alarm it could not deliver, and rises with it (Section 8.5).
<code>local-signal-missing</code>	The controller has no local signal for critical alarms: no sounder of its own, and no alarm output that signals every critical alarm (Section 14.1).	warning	—	A standing warning, which no owner can turn off. Critical alarms reach people only through notifications.
<code>reference-missing</code>	A power check or custom alarm refers to an element of the automation configuration that the controller cannot resolve (Section 19.3).	warning	—	The setting raises no other alarm until its reference is fixed.

1. **Devices.** A controller raises `device-lost` when Fail-Safe Behavior Section 7.6 requires it to raise a fault for a lost device, and `configuration-failed` when it does not command a device because it could not configure it [OAP-MB Section 10.2].
2. **Power.** A controller that can detect the loss of mains power MUST raise `mains-power-lost` while it is in power-loss mode [OAP-FS Section 9.5], once the loss has lasted the delay, and the `power-outage` notice when power returns (Section 13.2).
3. **Battery.** A controller with a backup battery MUST raise `battery-low` while it runs on the battery and expects to exhaust it within 30 minutes, or its charge is below 20 %, whichever comes first (provisional), and SHOULD raise `battery-fault` when the battery is missing, is not charging, or fails a test.
4. **Capacity.** A controller raises `capacity-exceeded` when it cannot take on a device, a module, or an element because of a documented limit, such as its number of modules or of node-IDs it can allocate.
5. **Stop-all.** `stop-all` is active while every output is held stopped by the `stop-all` operation [OAP-SA Section 11.8]. Its subject is the controller, and it names the user who started it.
6. **Notifications.** `notification-failed` takes the severity of the most severe alarm whose notification it could not deliver (Section 16.7). When a channel fails to deliver an alarm more severe than any it failed to deliver before, its `notification-failed` rises to that severity, and sounds and is notified again as if it had just been raised. It is the only alarm whose severity changes while it lasts.
7. **A local signal.** `local-signal-missing` is active while the controller has no local signal for critical alarms (Section 14.1). Its subject is the controller.
8. **References.** `reference-missing` is active while a setting refers to an element of the automation configuration that the controller cannot resolve (Section 19.3). Its subject is the power check or custom alarm that holds the reference.

8.6 Automation Faults

Every fault that the controller raises under Automation Section 17.9 raises an alarm, with the element it concerns as its subject.

Table 8 — Default severities of Automation faults

Code	Meaning	Default	Critical for	Notes
input-unavailable	An input used by a function, interlock, or mode is unavailable (Automation Section 9.8).	warning	vital	Raised after 60 s. Also critical when it keeps hazardous equipment running (Section 8.6.1).
inputs-disagree	The inputs of a derived input differ by more than its <code>maxDifference</code> (Automation Section 9.7).	warning	—	One of two redundant sensors is probably wrong.
max-on-time	A function stopped an output that reached its <code>maxOnTime</code> . Without <code>restTime</code> , the owner clears it (Automation Section 11.8).	warning	hazardous	Water that does not reach a top-off's sensor in time may be going somewhere else.
interlock-latched	A latching interlock holds outputs stopped until the owner clears it (Automation Section 13.2).	warning	vital	Equipment stays stopped until a person clears the interlock.
binding-missing	An output or input has no endpoint, or its endpoint does not match it (Automation Section 8.5).	warning	vital	
command-refused	An endpoint refused a command (Automation Section 17.4).	warning	vital, hazardous	Critical only when vital equipment cannot be started, or hazardous equipment cannot be stopped (Section 8.6.4).
program-held	A step program cannot advance because the clock is not valid (Automation Section 16.4).	warning	—	
unsupported	The controller does not support an element of the configuration (Automation Section 19.2).	warning	—	
emergency-mode	An emergency mode is active (Automation Section 14.7).	critical	—	Each active emergency mode has its own alarm (Section 8.6.3).
safety-path-missing	A safety function has no path that uses no radio link to an output it stops (Automation Section 13.7).	warning	—	A standing warning, which no owner can turn off. The configuration still runs (Fail-Safe Behavior Section 13.4).

1. `input-unavailable` is also `critical` when a function that uses the input keeps hazardous equipment running because of it, that is, when its `unavailable-input` behavior is `run`, a level, or `hold` [OAP-AU Table 18].
2. A controller raises `input-unavailable` also for an input used only by a custom alarm (Section 11).
3. `emergency-mode` is active while an emergency mode is active [OAP-AU Section 14.7], with the mode as its subject. Each active emergency mode has its own alarm.
4. `command-refused` is critical through its equipment groups only when the refused command would have started vital equipment or kept it running, or would have stopped hazardous equipment. Any other refusal has the default severity, such as a dose that a dosing head refuses at its daily limit, which the Dosing specification expects (its Section 10.2).
5. `safety-path-missing` is a standing warning (Section 7.3.7), with the element that carries the safety function as its subject [OAP-AU Section 13.7].

8.7 Leaks, Levels, Ranges, Power, and Custom Alarms

Table 9 — Default severities of other conditions

Code	Meaning	Default	Critical for	Notes
leak	A <code>leak-sensor</code> endpoint reports <code>leak</code> , or an endpoint reports the Device Classes fault <code>leak</code> because it has stopped for a leak.	critical	—	The Water Level and Water Change specification defines how equipment responds.
overflow	A <code>level-switch</code> endpoint whose <code>position</code> is <code>overflow</code> reports <code>wet</code> .	critical	—	

Code	Meaning	Default	Critical for	Notes
reservoir-low	A level-switch endpoint whose position is reservoir-low does not report wet.	warning	—	A single float is often the only warning before a top-off reservoir runs dry.
outside-desired-range	A value is outside the owner's desired range (Section 9).	info	—	
outside-acceptable-range	A value is outside the owner's acceptable range (Section 9).	warning	—	
in-danger-range	A value is beyond one of the owner's danger limits (Section 9).	critical	—	
power-abnormal	Equipment that is running draws more or less power than the owner expects (Section 10).	warning	vital	A pump that draws much less than usual may be running dry; a heater that draws less may have a failing element.
owner-defined	An alarm the owner defined on a condition of the automation configuration (Section 11).	warning	—	The owner sets its severity.

1. Every leak-sensor endpoint of a device the controller controls raises leak while it reports leak true.
2. Every level-switch endpoint whose position is overflow raises overflow while it reports wet true, and every one whose position is reservoir-low raises reservoir-low while it reports wet false.
3. These alarms do not depend on the automation configuration: a leak sensor raises leak whether or not an input of the configuration reads it.
4. While a supervised input reports wiring-fault, its state is unknown, so it neither raises nor clears these alarms; wiring-fault raises its own alarm (Table 6).

How equipment responds to leaks and levels is for the Water Level and Water Change specification.

8.8 Consequences

One failure often causes others. A lost probe module makes each of its inputs unavailable, and each input makes its functions apply their unavailable-input behavior.

1. An alarm is a consequence of another, its cause, when:
 - it is input-unavailable for an input whose endpoint's device is lost, or whose endpoint reports a fault that has an alarm; or
 - it is communication-lost for a device that is lost.
2. A consequence is raised, shown, and kept in history like any other alarm, but is not notified or sounded on its own while its cause's condition lasts, including while the cause waits for its delay, unless it is more severe than its cause. Instead, the cause's notifications list it (Section 12.7).

9 Range Alarms

9.1 Ranges

Owners set, for each parameter that matters to them, the values they aim for, the values they can live with, and the values that are dangerous. A range holds these for one parameter in one system, or in one vessel of it.

Table 10 — Members of a range

Member	Type	Presence	Description
id	identifier	REQUIRED	
name	string	OPTIONAL	Name shown to the owner.
system	identifier	REQUIRED	The Data Model system.
vessel	identifier	OPTIONAL	A vessel of system. Without it, the range applies to the system's circulating water (Section 9.3).
parameter	string	REQUIRED	A Data Model parameter identifier, such as temperature or nitrate.

Member	Type	Presence	Description
unit	string	REQUIRED	The unit of every number of the range. MUST be accepted for parameter [OAP-DM Section 9.2].
desired	object	OPTIONAL	min, max, or both: the values the owner aims for.
acceptable	object	OPTIONAL	min, max, or both: the values the owner can accept for a while.
danger	object	OPTIONAL	below, above, or both: the limits beyond which a value is dangerous.
methods	array	OPTIONAL	Data Model method types, such as titration or sensor. Only values measured by these methods count. Default: all.
delay	number (s)	OPTIONAL	How long a sensor's readings must stay outside before an alarm is raised. Default 60 s (provisional).
hysteresis	number	OPTIONAL	How far back inside a bound a sensor's reading must come before the alarm clears. Default 0.
notes	string	OPTIONAL	Free text.

1. A range MUST have at least one of `desired`, `acceptable`, and `danger`.
2. The bounds MUST be in order, for those that are given: `danger.below` ≤ `acceptable.min` ≤ `desired.min` < `desired.max` ≤ `acceptable.max` ≤ `danger.above`.
3. A system or vessel MUST NOT have two ranges for the same parameter.
4. Ranges are the owner's statement of what is right for their livestock. This Specification gives no default values, and a controller MUST NOT set a range without the owner's confirmation. A controller MAY propose values from published husbandry guidance, as starting points the owner confirms.

For example, a reef owner's temperature range in °C might be desired 25.0 to 26.0, acceptable 24.0 to 27.0, and danger below 22.0 or above 29.0.

9.2 Zones

The bounds divide the values of a parameter into zones, and each zone beyond the desired range has a condition of its own (Table 9).



Figure 2 — The zones of a temperature range, in °C

1. `outside-desired-range` is active while a value is below `desired.min` or above `desired.max`.
2. `outside-acceptable-range` is active while a value is below `acceptable.min` or above `acceptable.max`.
3. `in-danger-range` is active while a value is below `danger.below` or above `danger.above`.
4. The conditions are nested: a value in danger is also outside the acceptable and desired ranges, so all three alarms are active. They clear one by one as the value returns.
5. The alarm's `direction` is `high` or `low`, and its `limit` is the bound that was crossed.

Some controllers call everything outside the acceptable range “danger”. Danger limits that are set apart from the acceptable range leave room for a warning before a critical alarm. An owner who wants a critical alarm as soon as a value is not acceptable sets the danger limits equal to the acceptable range.

9.3 Which Values Count

A range applies to every value of its parameter that the controller records or receives for its place, whatever its source.

1. **Place.** A range with a `vessel` applies to values of that vessel. A range without one applies to values of the system that have no vessel or a circulating vessel [OAP-DM Section 8.2].
2. **Sources.** Values come from:
 - **readings:** sensors, and analyzers’ results [OAP-DC Sections 10.3 and 11.1], from every endpoint that measures the parameter in that place; and
 - **results:** other Data Model observations, such as the owner’s test kit results and laboratory results.
3. **Units.** The controller converts each value to the range’s `unit` with the Data Model registry [OAP-DM Section 9.5].

4. **Quality.** A reading counts only while its `quality` is `good` and it is no older than its maximum age [OAP-FS Section 10.2; OAP-AU Table 10]. A stale or faulty reading neither raises nor clears range alarms; its own faults raise their own alarms.
5. **Qualified values.** A value with the qualifier `<` or `>` counts as beyond a bound only where it must be, and as inside otherwise. For example, a phosphate result of `< 0.02 mg/L-PO4` is below a `desired.min` of 0.03, but a result of `> 0.5` is not known to be above a `danger.above` of 1.0. A qualified reading counts only while its `quality` is `good` (item 4), as when a sensor reports a reading beyond its measuring range while it can tell that its probe works: a conductivity reading of `> 80 mS/cm` is above a `danger.above` of 60.
6. **Severel sources.** The conditions of a range are active while the latest value of any source counts as beyond the bound. Each sensor and analyzer is a source of readings, and all results together are one source, whose latest value is the latest result. The alarm records the value that raised it, and its source.

9.4 Raising and Clearing

1. A reading raises an alarm once readings of its source have stayed beyond the bound for the range's `delay`.
2. A reading clears it once it is back inside the bound by at least `hysteresis`, and has stayed there for the alarm's clear delay (Section 12.3).
3. A result raises an alarm at once, and the alarm stays active until a later result is inside the bound.
4. A result says nothing about the values that follow it, so acknowledging an alarm that a result raised also clears it, and the result counts no longer.
5. A result with an older time than the latest result does not change an alarm.

For example, an owner's alkalinity test of 5.8 dKH raises `in-danger-range`, `outside-acceptable-range`, and `outside-desired-range` at once. They stay until a later test is within the bounds, or until someone acknowledges them, even though the owner tests only weekly.

9.5 Ranges and Control

1. Ranges raise alarms. They never control equipment; control is for the automation configuration. An owner who wants equipment to respond to a value uses an Automation interlock or function, which can read the same values.
2. A controller SHOULD warn the owner when the `limits` of a setpoint function [OAP-AU Section 12.4] for a parameter lie outside the acceptable range for the same parameter and place, because the controller could then drive the value into an alarm.

9.6 Ranges in the Data Model

Ranges are also useful outside alarms: apps draw them on charts, and an owner moving to a new controller wants to keep them.

Data Model revision 0.2 records them as Ranges, with the members of Table 10 other than `delay` and `hysteresis`, which belong to alarms [OAP-DM Section 8.9]. So a Data Model document can carry an owner's ranges as well as alarm settings can (Section 19); recording alarms with the other changes of Section 20.3 is Appendix D, issue 11.

10 Equipment Alarms from Power Draw

10.1 Faults Reported by Devices

Measuring the power each output draws reveals failures that nothing else reveals:

- **stuck on:** an output that is off but still draws power, such as a heater whose relay has welded shut;
- **disconnected:** an output that is on but draws nothing, such as an unplugged pump or a failed solenoid coil; and
- **abnormal:** an output that draws much more or less than usual, such as a pump running dry, a blocked impeller, or a heater with a failing element.

A device whose endpoint has the `power-monitoring` feature SHOULD report `stuck-on` and `load-disconnected` itself [OAP-DC Section 9.1], and those faults raise alarms like any other (Section 8.4).

10.2 Power Checks

A controller also checks the power drawn by any output whose endpoint reports `power` [OAP-DC Sections 9 and 10], against the owner's power check for that output.

Table 11 — Members of a power check

Member	Type	Presence	Description
<code>id</code>	identifier	REQUIRED	Identifies the power check, as the subject of its <code>reference-missing</code> (Section 19.3).
<code>output</code>	identifier	REQUIRED	The Automation output. It MUST be bound to an endpoint that reports <code>power</code> .
<code>running</code>	object	OPTIONAL	<code>min</code> , <code>max</code> , or both, in W: the power the equipment draws while it runs normally.
<code>stopped</code>	object	OPTIONAL	<code>max</code> , in W: the most it may draw while it is stopped.
<code>delay</code>	number (s)	OPTIONAL	How long a reading must stay beyond a limit before an alarm is raised. Default 30 s (provisional).
<code>idle</code>	number (s)	OPTIONAL	How long a running output may draw less than <code>running.min</code> before an alarm is raised, for equipment that switches itself off (Section 10.4). Default: <code>delay</code> .
<code>notes</code>	string	OPTIONAL	Free text.

A power check MUST have `running`, `stopped`, or both, and `stopped.max` MUST be less than `running.min`.

The controller judges the output's state from what its endpoint reports [OAP-AU Table 5], not from what it commanded, and:

1. raises `stuck-on` while the output has been reported stopped, and has drawn more than `stopped.max`, for `delay`;
2. raises `load-disconnected` while the output has been reported running, and has drawn no more than `stopped.max`, or no more than 2 W if `stopped` is not given (provisional), for `idle`;
3. raises `power-abnormal` while the output has been reported running, and has drawn less than `running.min`, but more than the limit of item 2, for `idle`, or more than `running.max` for `delay`; and
4. ignores the first `delay` after the output starts or stops, while the equipment's power settles.

Each alarm's subject is the endpoint, so that it coincides with a fault the device reports itself (Section 8.3), and the alarm names the output.

10.3 Default Checks and Learning

1. Without a power check, a controller SHOULD check every output that is carried by a `switch` endpoint with the `power-monitoring` feature as if `stopped.max` were 2 W (provisional), because an outlet that is off should draw nothing.
2. A controller MAY learn the power an output draws while it runs, for example its median power over a day, and propose a power check from it, such as `running` from 70 % to 130 % of the median (provisional). The owner confirms it.
3. A controller MUST NOT change a power check without the owner's confirmation.

10.4 Equipment That Switches Itself

Some equipment switches itself on and off while its output is on: a heater with its own thermostat, a chiller that delays its compressor, or a top-off unit on an outlet. Such equipment draws nothing for a while, which is not a fault.

1. For such equipment, the owner sets `idle` to the longest time it may draw nothing, or less than `running.min`, while it is on.
2. A device that reports `load-disconnected` itself cannot know this, so an owner MAY lengthen its delay with a condition setting (Table 4).

11 Custom Alarms

Owners alarm on things that no specification can foresee: a skimmer cup that is full, a door left open, or a reading from a sensor of their own. A custom alarm raises an alarm while an Automation condition, or an Automation input that is true or false, is true.

Table 12 — Members of a custom alarm

Member	Type	Presence	Description
id	identifier	REQUIRED	
name	string	REQUIRED	The alarm's title, such as "Skimmer cup full".
when	identifier	REQUIRED	An Automation condition, or an input that is true or false, of the controller's automation configuration [OAP-AU Sections 9 and 10].
severity	string	REQUIRED	The alarm's severity.
delay	number (s)	OPTIONAL	How long <code>when</code> must be true before the alarm is raised. Default 0.
clearDelay	number (s)	OPTIONAL	How long <code>when</code> must not be true before the alarm clears. Default as in Section 12.3.
system, vessel	identifier	OPTIONAL	Where the alarm applies, for its messages and history.
message	string	OPTIONAL	Text for the owner, such as what to do.
notes	string	OPTIONAL	Free text.

1. A custom alarm raises the code `owner-defined`, with the custom alarm as its subject.
2. While `when` is unknown, the alarm neither raises nor clears, and the controller raises `input-unavailable` for the input that makes it unknown (Section 8.6.2).
3. `when` MUST NOT be a condition that depends on the state of an alarm, so that alarms never raise one another in a loop.

12 Alarm Behavior

12.1 States

An alarm is in one of the states of Table 13.

Table 13 — Alarm states

State	Condition	Acknowledged	What happens
Active, unacknowledged	Active	No	Sounds, signals, notifies, repeats, and escalates.
Active, acknowledged	Active	Yes	Shown, with reminders. No sound, repeats, or escalation.
Cleared, unacknowledged	Cleared	No	Only for latching alarms and notices. Shown, with reminders, until acknowledged.
Closed	Cleared	Yes, or not needed	Kept in history.

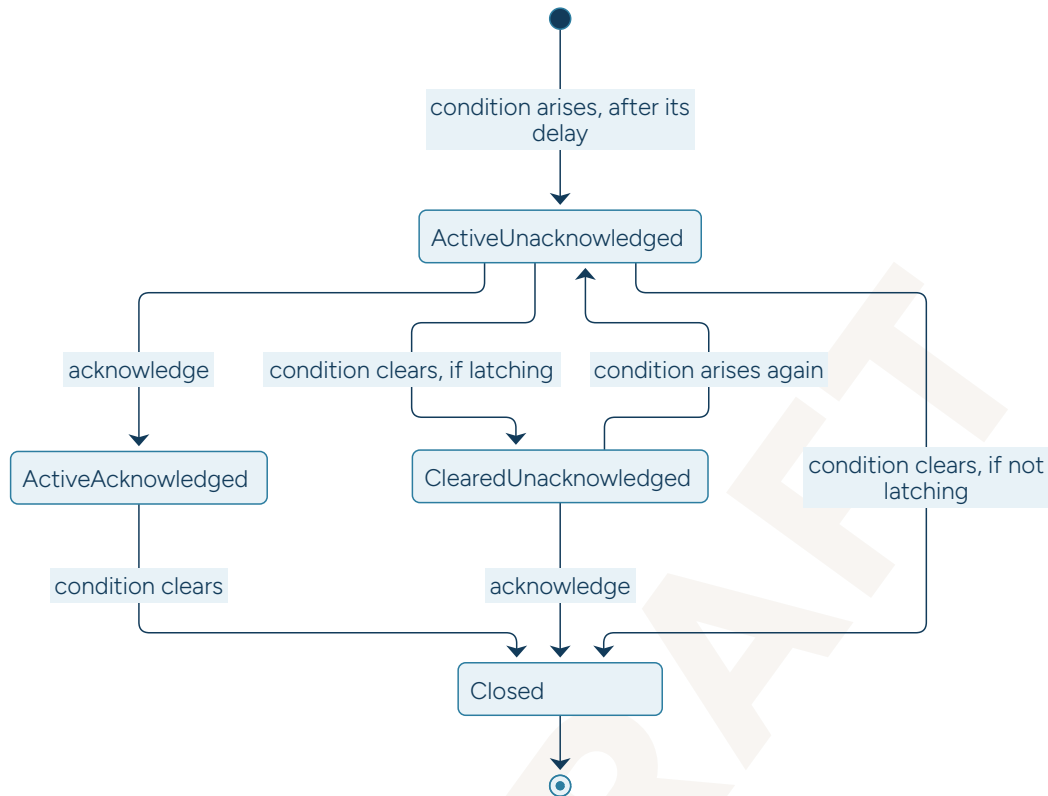


Figure 3 — Alarm states

1. An alarm latches if its severity's default says so (Table 2): `critical` alarms latch, and others do not (provisional).
2. Notices start cleared and unacknowledged, and close when acknowledged (Section 13.1).

12.2 Raising

1. A controller raises an alarm when its condition has lasted for the condition's delay: the delay in a condition setting, a range, a power check, or a custom alarm; otherwise the default in `conditions.json`; otherwise 0.
2. A controller MUST raise an alarm within 5 s of the end of its delay (provisional).
3. Each alarm has a UUID [RFC 9562] that stays the same until it closes, and a `raised` time.
4. Faults and conditions that are present when the controller starts raise their alarms once start-up is complete [OAP-FS Section 9.4], with the time they were first seen.
5. The delay is the only time a condition must last before its alarm is raised. Where another specification gives such a time for one of its codes, that time is the code's delay in `conditions.json`, and the condition itself has no time in it, so that the two never add up. For example, a pump's flow that has been below its minimum for 60 s raises `flow-low` [OAP-FP Section 13.5]: the condition is the low flow, and the 60 s is its delay.

12.3 Clearing and Latching

1. An alarm clears when its condition has been absent for its clear delay: the `clearDelay` of a condition setting or custom alarm, or 30 s (provisional). The clear delay keeps a condition that comes and goes, such as a float bobbing on a ripple, from raising alarm after alarm [ISA-18.2].
2. When an alarm clears:
 - if it does not latch, or has been acknowledged, it closes;
 - otherwise it waits, cleared and unacknowledged, until someone acknowledges it.
3. A cleared alarm no longer sounds, repeats, or escalates, but stays shown until it closes. The controller sends one `cleared` message to everyone who was notified of it (Section 16.6).

A critical alarm that clears by itself, such as a temperature that returned to normal before anyone saw it, still needs someone to know it happened.

12.4 Acknowledgement

Acknowledging an alarm records that a person knows about it. It does not fix anything, and it does not clear the alarm.

1. A user acknowledges an alarm with the `acknowledge-alarm` operation [OAP-SA Table 6], which needs the `override` role, through an app or a service. A person at the controller MAY also acknowledge alarms with its controls (Section 14.3), as physical presence [OAP-SA Section 9.10].
2. A controller MUST NOT accept an acknowledgement that does not come from an authenticated user with that role, or from physical presence. In particular, it MUST NOT accept a reply to an email or a text message as an acknowledgement.
3. An acknowledgement applies to one alarm. A user MAY acknowledge several, or all, at once.
4. When an alarm is acknowledged, its sound, its repeats, and its escalation stop, and the controller records who acknowledged it and when.
5. The controller SHOULD send an `acknowledged` message to everyone who was notified of the alarm, so that, for example, a neighbor knows not to come.
6. Every acknowledgement is recorded in the audit log [OAP-SA Section 12.1].

12.5 Silencing

Silencing stops sound, for a person who is dealing with a problem at the tank.

1. Silencing stops the local sound and the alarm outputs that sound (Section 15) for the alarms that are sounding, for the silence duration: 15 minutes by default, at most 60 minutes (provisional). It also silences the sounders of devices for the faults of those alarms (Section 14.4).
2. It does not acknowledge them, and their notifications, repeats, and escalation go on.
3. A new alarm, or the end of the silence while an alarm is still unacknowledged, sounds again.
4. A person at a controller that has a sounder MUST be able to silence alarms there without any credential, with a button or a display (Section 14.3). Users silence alarms remotely with the `acknowledge-alarm` operation.
5. A silence is recorded, with the alarms it silenced, but not in the audit log unless a user silenced them remotely.

An alarm is sounding while it is unacknowledged, not silenced, and not held (Sections 12.8 and 16.5), and is either active or a notice. The local sound (Section 14.1) and the alarm outputs whose function signals `sounding` (Section 15.1) follow exactly the alarms that are sounding, so an alarm that has cleared, or one that a mode holds, sounds on neither.

12.6 Repeats and Reminders

1. **Repeats.** While an alarm is active and unacknowledged, the controller notifies everyone its escalation policy has reached so far again, at the policy's `repeat`, or at the severity's default repeat (Table 2).
2. **Reminders.** The controller notifies them again, at the severity's reminder interval, about an alarm that needs attention but no longer repeats: one that is active and acknowledged, one that is cleared and waits for acknowledgement, and a notice that waits for acknowledgement.
3. Repeats and reminders are sent as `repeated` and `reminder` messages (Section 16.6).

12.7 Grouping and Floods

When many alarms are raised at once, owners need one clear message, not dozens.

1. A controller SHOULD combine the messages for one recipient and channel that fall due within 10 s of each other into one `summary` message (provisional).
2. A notification of a cause lists its consequences (Section 8.8).

3. When more than 10 alarms are raised within 10 minutes [ISA-18.2], a controller SHOULD send each recipient no more than one message per channel per minute, as summaries, except that the first notification of each critical alarm is never delayed.
4. A controller SHOULD show alarms of one range together, and alarms of one device together.

12.8 Modes, Water Changes, and Calibrations

While a maintenance or water-change mode is active [OAP-AU Section 14], the owner is usually at the tank, and some faults are expected, such as a probe out of the water.

1. While such a mode is active, the controller holds the sound and the notifications of `warning` and `info` alarms raised during the mode.
2. When the mode ends, it sounds and notifies those that are still active and unacknowledged, as if they had just been raised.
3. Modes never hold `critical` alarms.
4. A water change holds the alarms of the conditions that it makes expected, and only those: the codes, in the phases of the water change, that the Water Level and Water Change specification names [OAP-WL Section 9.13]. It holds them, critical ones included, but never drops them: the controller raises, shows, and records each such alarm as usual, and holds only its sound and its notifications.
5. As soon as the water change ends or aborts, or leaves the phases in which a condition is expected, the controller sounds and notifies each held alarm that is still active, as if it had just been raised, and notifies each that has cleared but waits to be acknowledged because it latched.
6. A probe's calibration holds the sound and the notifications of the alarms that the Probe Interface specification makes expected during it: the `input-unavailable` alarms of the probe's inputs [OAP-PI Section 14.7]. A calibration never holds a `critical` alarm.
7. A calibration's hold ends when the calibration ends: completed, failed, cancelled, or timed out. The controller then sounds and notifies the held alarms that are still active and unacknowledged, as if they had just been raised, although the probe's inputs may stay unavailable until a person confirms that the probe is back [OAP-PI Section 14.8].
8. Holds never apply to feeds, which report every change of every alarm (Section 16.2.4).

For example, while an automatic water change drains a sump, its `low-water` alarm, which is critical, is raised and shown but does not sound; if the sump is still low when the water change ends, the alarm sounds and is notified at once.

12.9 Restarts and the Clock

1. A controller MUST keep its alarms that are not closed across restarts and power loss, with their states, acknowledgements, silences, escalation steps, and held notifications.
2. After a restart, it checks each condition again. An alarm whose condition has ended clears, at the time of the restart.
3. Delays, repeats, reminders, and escalation are measured in elapsed time while the controller runs. After a restart, they go on from where they were, without counting the time the controller was off.
4. While the controller's clock is not valid [OAP-FS Section 10.5], quiet hours do not apply, so nothing is held by them, and remote channels that need the time, such as push and email over TLS, may fail (Section 16.7).

12.10 Visibility and Permissions

1. A controller MUST show, to every user with the `view` role, every alarm that is not closed: its severity, title, subject, state, when it was raised and cleared, and who acknowledged it.
2. Apps SHOULD show critical alarms above everything else.
3. A controller MUST show the history of closed alarms (Section 20).

Table 14 — Operations on alarms and the roles that permit them

Operation	Security and Access operation	Role
See alarms and notices	<code>view-state</code>	<code>view</code>
See closed alarms	<code>view-history</code>	<code>view</code>

Operation	Security and Access operation	Role
Acknowledge or silence alarms	acknowledge-alarm	override
Register one's own app installations for push, set one's own quiet hours, and test one's own contacts	manage-own-credentials (provisional)	view
Change ranges, power checks, custom alarms, condition settings, local settings, policies, and rules	change-configuration	full-control
Add or change recipients, channels, and heartbeats, and test them	manage-consent (provisional)	owner

Recipients and channels decide what leaves the owner's network, which is for owners [OAP-SA Section 16]; a user's own push subscriptions send alarms only to that user. Whether dedicated operations would be clearer is Appendix D, issue 8.

13 Notices

13.1 Notices

A notice reports something that happened and needs a person to know, such as a power outage. It has no condition that stays true.

1. A notice is raised cleared and unacknowledged, and closes when someone acknowledges it.
2. It sounds, signals, and is notified like an alarm of its severity, but does not repeat or escalate.
3. It is reminded at its severity's reminder interval until someone acknowledges it, so that it is reported until acknowledged.

Table 15 — Notices

Notice	Meaning	Default	Notes
power-outage	Mains power was lost and has returned, or the controller started after being without power (Section 13.2).	warning	Recorded as a Data Model power-outage event.
controller-restarted	The controller restarted without being asked to, for example after its watchdog (Section 13.3).	warning	
device-restarted	A device restarted without being asked to (Section 13.3).	warning	Its outputs went through their power-up behavior.
owner-recovery	An owner credential was added by recovery with physical presence (Security and Access Section 11.7).	warning	Sent to every owner.
firmware-reverted	New firmware failed to confirm, and a device returned to its previous firmware (Security and Access Table 8).	warning	
firmware-refused	A firmware image or update manifest failed verification (Security and Access Table 8).	warning	It may have been tampered with.
update-available	A firmware update is available for a device (Security and Access Section 15.9).	info	warning when the release fixes security problems.
fabric-added	A Matter fabric was added to the controller's bridge (Section 13.4).	warning	Sent to every owner. Another ecosystem can now use what the bridge offers.
fabric-removed	A Matter fabric was removed from the controller's bridge (Section 13.4).	warning	Sent to every owner.

13.2 Power Outages

1. A controller MUST raise a power-outage notice:
 - when mains power returns after the controller entered power-loss mode [OAP-FS Section 9.5], whether or not the loss lasted long enough to raise mains-power-lost (Section 8.5); and
 - when it starts after a loss of power that it can tell from its own reset reason, such as a power-on reset, if it was not running on a battery.

2. Its details give the outage's `start` and `end`, where known, whether the controller ran on a battery (`onBattery`), and how it detected the outage (`detectedBy`: `power-fail-input` or `restart`).
3. A controller that can raise the notice after a restart MUST record the time, while its clock is valid, at least every 5 minutes (provisional), in storage that keeps it without power. After a restart, the start is the last time it recorded, and the end the time it started, if its clock is valid; otherwise they are unknown. So the start is at most 5 minutes before the power was lost.
4. A controller SHOULD say in the notice what the outage did, such as which outputs were stopped on battery and when they restart [OAP-FS Section 9.2], and SHOULD advise checking equipment that needs it, such as a canister filter after a long outage [OAP-FS Table 3].
5. The notice is recorded as a Data Model `power-outage` event (Section 20.2).

A short loss of mains power that ends before the `mains-power-lost` delay raises no alarm while it lasts, but does raise the notice, so the owner learns of it. A controller that keeps its time in a battery-backed clock can record it there, which needs no writes to flash memory.

13.3 Restarts

1. A controller MUST raise `controller-restarted` when it starts after a restart that no user asked for and that a power outage does not explain, such as after its watchdog [OAP-FS Section 11.1], a software fault, or a brownout.
2. A controller MUST raise `device-restarted` when a device restarts without being asked to, which it can tell from the device's `uptime` going down or from the binding.
3. A restart for a firmware update, or one that a user asked for with the `restart` operation [OAP-SA Table 6], raises no notice. Devices that restart because the controller's power was lost are covered by the `power-outage` notice.
4. Their details give the `reason`, where known (`watchdog`, `software`, `brownout`, or `unknown`), and how long the controller or device was off (`downtime`), where known.

13.4 Other Notices

1. A controller SHOULD notify every owner of `owner-recovery`, as Security and Access Section 11.7.3 requires, whatever the rules say.
2. A controller raises `firmware-reverted` and `firmware-refused` when it records them in its audit log [OAP-SA Table 8].
3. A controller that checks for updates raises `update-available` once for each new release of a device's firmware [OAP-SA Section 15.9], as `warning` when the release lists `securityFixes` [OAP-SA Section 15.6].
4. A controller with a Matter bridge MUST raise `fabric-added` when a Matter fabric is added to its bridge, and `fabric-removed` when one is removed, however it was added or removed [OAP-MM Section 7.3], and MUST notify every owner of each, whatever the rules say. Their details give the fabric's `label` and `vendor`, as the controller shows them to owners, and their subject is the controller. Any administrator of a fabric can let another ecosystem join the bridge, so these notices are how owners learn that one has.

14 Local Signals

14.1 Sound

A critical alarm needs a signal that someone at the tank, or in the next room, notices without a phone.

1. Every controller MUST have a local signal for critical alarms, or warn its owners until it has one (item 2). A local signal for critical alarms is a sounder built into the controller, or an alarm output (Section 15) driven by an alarm function whose `severities` include `critical`, whose `codes` and `systems`, if it has them, leave out no critical alarm, and at least one of whose outputs is bound to an endpoint. A controller without a sounder MUST support alarm functions, so that its owners can give it one.
2. While a controller has neither, it MUST raise `local-signal-missing`, a standing warning (Section 7.3.7), which tells its owners that critical alarms reach people only through notifications, and how to give it a signal.

3. A controller SHOULD have a sounder, which SHOULD give at least 75 dB(A) at 1 m (provisional), loud enough to hear in the next room.
4. A controller with a sounder MUST have a button or a display that silences it (Section 14.3).

Table 16 — Local signals by severity

Severity	Sound	Alarm indicator
critical	Continuous: three short tones (0.2 s on, 0.2 s off) every 2 s, until silenced or acknowledged.	Red, flashing at about 2 Hz while unacknowledged; steady red while acknowledged and active.
warning	Intermittent: two short tones when raised, and again every 30 minutes, until silenced or acknowledged.	Yellow, flashing at about 0.5 Hz while unacknowledged; steady yellow while acknowledged and active.
info	None.	None; shown in apps and on a display.

5. A controller with a sounder MUST sound every **critical** alarm that is sounding (Section 12.5), at any time, and MUST NOT let any setting turn this off.
6. The local quiet hours (Section 16.5) hold the sound of **warning** and **info** alarms, never of **critical** ones.
7. An owner MAY set a **warning** to sound only once when it is raised (**once**), or not at all, and an **info** to sound once.
8. A latched alarm that has cleared stops sounding, but its indicator keeps flashing until someone acknowledges it.
9. The patterns and rates are provisional. The colors and rates follow the practice of IEC 60601-1-8 [IEC 60601-1-8], so that a red flashing light means the same as on other equipment.

14.2 Indicators

1. A controller with an alarm indicator MUST show the most severe alarm that is not closed, as Table 16 says.
2. A controller SHOULD NOT use red, or a flashing light, on any other indicator, so that the alarm indicator is not confused with them.
3. A controller with a display SHOULD show the title and time of every alarm that is not closed, most severe first.

14.3 Buttons

1. A controller with a button MUST let a short press silence the alarms that are sounding (Section 12.5), before any other function the button has, whenever an alarm is sounding.
2. It MAY let a long press, of at least 3 s (provisional), acknowledge every alarm that is not closed, as physical presence. A controller with a display SHOULD instead let the person choose the alarms to acknowledge.
3. A controller MAY combine these with the stop-all of Security and Access Section 14.2.1, as long as a press meant to silence an alarm can never stop every output.

14.4 Signals on Devices

Some devices, such as top-off units, sound or show their own faults.

1. A device that detects a fault that endangers livestock or property, such as **high-level** or **dry-run**, SHOULD signal it locally with whatever indicator or sounder it has, even without a controller.
2. Such a device reports that it is sounding with its endpoint's **sounding** attribute, and stops with the **silence** command [OAP-DC Section 8.2].
3. When a person silences alarms (Section 12.5), the controller MUST send **silence**, for the silence duration, to each endpoint that implements it and reports **sounding** for a fault whose alarm was silenced. When such an alarm is acknowledged, it MUST send **silence** without a duration, which lasts until the endpoint's faults clear.
4. A controller MUST show the owner which devices report **sounding**, so that the owner can find a device that sounds but has no **silence**.

So an owner silences every alarm in one place, whichever device sounds it.

15 Alarm Outputs

15.1 The Alarm Function

An alarm output is an output of the automation configuration that signals alarms: a siren, a strobe, a light by the front door, or a contact to a house alarm panel or a building management system. The controller drives it with an alarm function, a function type that this Specification defines for the Automation specification to host [OAP-AU Section 11.10].

Until the Automation specification lists it among its function types, an alarm function's `type` is the extension name `org.openaquariumproject.alarm` (provisional; Appendix D, issue 9).

Table 17 — Members of an alarm function

Member	Type	Presence	Description
<code>id</code> , <code>type</code> , <code>name</code> , <code>notes</code>			As for every function [OAP-AU Section 11.1]. <code>type</code> is <code>org.openaquariumproject.alarm</code> .
<code>outputs</code>	array	REQUIRED	The outputs it drives.
<code>severities</code>	array	OPTIONAL	The severities of the alarms it signals. Default <code>["critical"]</code> .
<code>codes</code>	array	OPTIONAL	Only alarms with these codes. Default: all.
<code>systems</code>	array	OPTIONAL	Only alarms of these systems. Default: all.
<code>while</code>	string	OPTIONAL	When it signals an alarm: <code>sounding</code> (default), while the alarm is sounding (Section 12.5); <code>unacknowledged</code> , while it is unacknowledged; or <code>active</code> , while its condition is active, whether or not it is acknowledged.
<code>whenAlarm</code>	state	OPTIONAL	The outputs' state while any matching alarm is signaled: <code>run</code> (default), <code>stop</code> , or a level [OAP-AU Table 4].
<code>whenClear</code>	state	OPTIONAL	Their state otherwise. Default <code>stop</code> .

1. The function's demand is `whenAlarm` while at least one alarm matches `severities`, `codes`, and `systems`, and is signaled as `while` says, and `whenClear` otherwise.
2. It is a control function [OAP-AU Section 11.10]: each output has no other function, interlocks, modes, and overrides act on it as on any other output [OAP-AU Section 17.3], and `stop-all` stops it [OAP-SA Section 11.8].
3. It uses no inputs, so it has no unavailable-input behavior.
4. `sounding` suits sirens, `unacknowledged` suits a light that shows that something needs attention, and `active` suits contacts to other alarm systems, which keep their own records.
5. A controller SHOULD warn the owner when a mode, such as a power-loss or emergency mode, stops an alarm output.

15.2 Fail-Safe Configuration of Alarm Outputs

A controller that has failed cannot switch an alarm output on. The output's own fail-safe behavior can [OAP-FS Section 7.3]. Alarm outputs have the Data Model equipment category `alarm-signal`, which Data Model revision 0.3 adds, and Fail-Safe Behavior Table 3 gives its recommended configuration.

1. An alarm output that makes a sound or a light SHOULD be configured as Fail-Safe Behavior Table 3 recommends for `alarm-signal`: to run on loss of contact, that is `failSafeAction` state, with `failSafeState` its running state; `powerOnBehavior` `fail-safe`, with a `powerOnDelay` of 120 s, longer than the controller's start-up; and, on battery, `function`, which leaves it to its alarm function [OAP-AU Section 14.6]. Then the siren sounds when the controller stops supervising its device, but not after every power cut, and on battery it still signals alarms, among them `mains-power-lost`.
2. A contact to another alarm system SHOULD be closed while all is well, and open on alarm: `whenAlarm` `stop` and `whenClear` `run`, with a fail-safe state of `stop`, and a relay that is open when unpowered [OAP-FS Section 8.3]. Then the other system sees an alarm when the controller raises one, when the controller fails, and when the power fails.
3. An alarm output SHOULD be on a module or a device other than the controller, so that it can signal the controller's failure.

- When the owner gives an output the category `alarm-signal`, the controller SHOULD propose the configuration of item 1, as Fail-Safe Behavior Section 13.1.2 asks for every category, or that of item 2 for a contact to another alarm system.

15.3 Example

```
{ "id": "siren-critical", "type": "org.openaquariumproject.alarm", "outputs": ["siren"], "severities": ["critical"], "while": "sounding" }
```

```
{ "id": "panel-contact", "type": "org.openaquariumproject.alarm", "outputs": ["alarm-panel"], "severities": ["critical", "warning"], "while": "active", "whenAlarm": "stop", "whenClear": "run" }
```

The example `alarm-outputs.json` is a complete automation configuration with both.

16 Notifications

16.1 Recipients

Table 18 — Members of a recipient

Member	Type	Presence	Description
<code>id</code>	identifier	REQUIRED	
<code>name</code>	string	REQUIRED	Name shown in messages and to other recipients, such as "Jordan (next door)".
<code>user</code>	UUID	OPTIONAL	The Security and Access user the recipient is [OAP-SA Section 11.1]. REQUIRED to reach the recipient through apps.
<code>contacts</code>	array	REQUIRED	One or more contacts: each a <code>channel</code> , and, for email and extension channels that need it, an <code>address</code> .
<code>quietHours</code>	object	OPTIONAL	The recipient's quiet hours (Section 16.5).
<code>notes</code>	string	OPTIONAL	Free text.

- A recipient need not be a user: a neighbor with only an email address, or a text message service, can be a recipient.
- A recipient that is not a user learns nothing about the system except what alarm messages tell it, and cannot acknowledge alarms.
- A user's contacts on the `app` channel are their app installations that have registered for push (Section 17.2), and their sessions with the controller.

16.2 Channels

Table 19 — Channel types

Type	Delivers	Settings	Section
<code>app</code>	Notifications to the apps of recipients that are users: over their sessions with the controller, and by push.	None.	17.1, 17.2
<code>email</code>	Notifications by email, through an SMTP server the owner chooses.	<code>server</code> (<code>host</code> , <code>port</code> , <code>security</code> , <code>username</code> , <code>password</code>) and <code>from</code> .	17.3
<code>webhook</code>	Notifications to a recipient, or, with <code>feed true</code> , every change of every alarm, by HTTP POST.	<code>url</code> , <code>secret</code> , <code>feed</code> , <code>severities</code> .	17.4
<code>mqtt</code>	Every change of every alarm, to an MQTT broker the owner chooses. Always a feed.	<code>url</code> , <code>version</code> , <code>clientId</code> , <code>username</code> , <code>password</code> , <code>baseTopic</code> , <code>severities</code> .	17.5
An extension name	As its definer says.	As its definer says.	17.6

Every channel has an `id`, a `type`, and optionally a `name`, `notes`, and `detail` (Section 16.9).

- No maker's service.** A controller MUST NOT need a maker's service, or an account with one, to raise, show, acknowledge, or deliver any alarm. It MUST be able to deliver every alarm to its owners without one:

through its local signals and alarm outputs, and through app sessions, push, email, webhooks, and MQTT to services the owner chooses.

2. An owner MAY choose a maker's service, such as a push gateway for a native app (Section 17.2), or a relay service [OAP-SA Section 13.2]. It is then a notification service the owner chose [OAP-SA Table 12].
3. A controller MUST support the `app`, `email`, `webhook`, and `mqtt` channels (provisional; Appendix D, issue 6).
4. **Feeds.** A feed reports every change of every alarm whose severity is in its `severities` (default: all), whatever the rules, policies, and holds say: `raised`, `acknowledged`, `silenced`, `cleared`, and `closed` messages, and heartbeats if configured (Sections 16.6 and 18.2). A feed is for another system, such as a home automation server, which may show alarms, switch its own equipment, or notify people its own way. A feed is not a contact of a recipient.
5. Channels that leave the local network MUST use TLS (Section 17).

16.3 Escalation Policies

An escalation policy says who is notified about an alarm, through which channels, and when, until someone acknowledges it.

Table 20 — Members of an escalation policy

Member	Type	Presence	Description
<code>id</code>	identifier	REQUIRED	
<code>name</code>	string	OPTIONAL	Name shown to the owner.
<code>steps</code>	array	REQUIRED	One or more steps, in order of <code>after</code> . Each has <code>after</code> , the seconds from when the alarm was raised, and <code>notify</code> : a list of targets, each with <code>recipients</code> and optionally <code>channels</code> . Without <code>channels</code> , every contact of each recipient is used.
<code>repeat</code>	number (s)	OPTIONAL	How often to notify again, while the alarm is active and unacknowledged. At least 60. Default: the severity's repeat (Table 2).
<code>notes</code>	string	OPTIONAL	Free text.

1. When an alarm is raised, and at each step's `after` while the alarm stays unacknowledged, the controller notifies the step's targets: `raised` at the first step, `escalated` at later ones.
2. At every `repeat`, the controller notifies again everyone whom the steps have reached so far (Section 12.6).
3. Escalation stops when the alarm is acknowledged or clears.
4. A step's channels MUST be among the contacts of each of its recipients.

For example, push to the owners at once, email after 10 minutes, and a neighbor after 30 minutes:

```
{ "id": "critical", "steps": [
  { "after": 0, "notify": [{ "recipients": ["alex", "sam"], "channels": ["app"] }] },
  { "after": 600, "notify": [{ "recipients": ["alex"], "channels": ["email"] }] },
  { "after": 1800, "notify": [{ "recipients": ["jordan"] }] } ],
  "repeat": 600 }
```

16.4 Rules

Rules choose the escalation policy of each alarm.

Table 21 — Members of a rule

Member	Type	Presence	Description
<code>severities</code>	array	OPTIONAL	Applies only to alarms of these severities.
<code>codes</code>	array	OPTIONAL	Applies only to alarms with these codes.
<code>systems</code>	array	OPTIONAL	Applies only to alarms of these systems.
<code>policy</code>	identifier or <code>null</code>	REQUIRED	The escalation policy, or <code>null</code> to notify nobody.
<code>notes</code>	string	OPTIONAL	Free text.

1. The first rule, in document order, that applies to an alarm chooses its policy.

- When no rule applies, `critical` and `warning` alarms notify every user with the `owner` role through the `app` channel, and `info` alarms notify nobody (provisional; Appendix D, issue 7). So a new controller notifies its owners' apps before any setting is made.
- A controller SHOULD warn the owner when its settings send any critical alarm to nobody, and MUST show such settings to every owner.
- Alarms that notify nobody are still shown, sounded, and sent to feeds.

16.5 Quiet Hours

Quiet hours let owners sleep through a calibration reminder, but never through a leak.

Table 22 — Members of quiet hours

Member	Type	Presence	Description
<code>windows</code>	array	REQUIRED	Each with a <code>start</code> and an <code>end</code> , local times of day, and optionally <code>days</code> , the days of the week on which a window starts [OAP-AU Section 7.6]. A window whose <code>end</code> is earlier than its <code>start</code> ends the next day.
<code>hold</code>	array	OPTIONAL	The severities held: <code>warning</code> , <code>info</code> , or both. Default both. MUST NOT include <code>critical</code> .

- Quiet hours MUST NOT hold `critical` alarms, and a controller MUST NOT offer a way to make them do so.
- A recipient's quiet hours hold its notifications; the local quiet hours of the settings (Section 19.2) hold local sound and alarm outputs whose function signals `sounding`.
- During quiet hours, the controller holds the messages of held severities for that recipient. When quiet hours end, it sends one `summary` of the held alarms that still need attention: unacknowledged, or acknowledged and still active.
- Escalation goes on during one recipient's quiet hours: a later step may reach another recipient whose quiet hours have not begun.
- Times are in the settings' time zone, and quiet hours apply only while the clock is valid (Section 12.9).

16.6 Messages

Every channel carries the same alarm message: a JSON object [RFC 8259] valid against `schema/alarm-message.schema.json`.

Table 23 — Kinds of message

Kind	Sent when	To
<code>raised</code>	An alarm or a notice is raised	Recipients of the first step; feeds
<code>escalated</code>	A later step of the policy falls due	That step's recipients
<code>repeated</code>	The policy's repeat falls due	Recipients reached so far
<code>reminder</code>	The severity's reminder falls due (Section 12.6)	Recipients reached so far
<code>acknowledged</code>	Someone acknowledges the alarm	Recipients reached so far; feeds
<code>silenced</code>	Someone silences the alarm	Feeds
<code>cleared</code>	The alarm's condition clears	Recipients reached so far; feeds
<code>closed</code>	The alarm closes	Feeds
<code>summary</code>	Several messages for one recipient fall due together, or quiet hours end (Sections 12.7 and 16.5)	One recipient
<code>test</code>	Someone sends a test (Section 16.8)	The recipient or feed tested
<code>heartbeat</code>	A heartbeat falls due (Section 18.2)	The heartbeat's channel

Table 24 — Members of an alarm message

Member	Type	Presence	Description
<code>oapAlarmMessage</code>	string	REQUIRED	Revision of this format: " <code>0.2</code> ". A receiver also accepts " <code>0.1</code> ", which has no <code>power-check</code> subject.

Member	Type	Presence	Description
id	UUID	REQUIRED	Identifies this message. Each message has a new one; a retried delivery keeps it.
kind	string	REQUIRED	Table 23.
sent	timestamp	REQUIRED	When the controller created the message.
seq	integer	REQUIRED	A number that grows by one with each message the controller creates, so that a gap shows a lost message.
controller	object	REQUIRED	The controller's <code>keyId</code> [OAP-SA Section 8.3] and its <code>name</code> .
recipient	object	OPTIONAL	The recipient's <code>id</code> and <code>name</code> , for a message to one recipient.
alarm	object	see description	The alarm, as it is after the change the message reports (Table 25). REQUIRED for every kind except <code>summary</code> , <code>test</code> , and <code>heartbeat</code> , and absent for those.
alarms	array	see description	The alarms the summary reports. REQUIRED for <code>summary</code> , and absent otherwise.
step	integer	see description	The step reached, counting the first as 1. REQUIRED for <code>escalated</code> , and absent otherwise.
extensions	object	OPTIONAL	Extensions, as in the Data Model.

Table 25 — Members of an alarm

Member	Type	Presence	Description
id	UUID	REQUIRED	Identifies the alarm, from raised until closed.
code	string	REQUIRED	Its code.
kind	string	REQUIRED	<code>condition</code> or <code>notice</code> .
severity	string	REQUIRED	Its severity.
title	string	REQUIRED	A short title for people, of at most 200 characters, such as "Temperature above its danger limit".
text	string	OPTIONAL	More for people: what happened, and what to do.
raised	timestamp	REQUIRED	When it was raised.
active	boolean	REQUIRED	Whether its condition is active. Always <code>false</code> for a notice.
cleared	timestamp	OPTIONAL	When its condition cleared.
acknowledged	object	OPTIONAL	<code>time</code> , and <code>by</code> : the <code>kind</code> of actor (<code>person</code> , <code>service</code> , or <code>physical</code>), and the user's <code>user</code> UUID and <code>name</code> , where known.
silencedUntil	timestamp	OPTIONAL	When its silence ends.
closed	timestamp	OPTIONAL	When it closed.
subject	object	REQUIRED	Its subject (Table 5): <code>kind</code> and the members it needs.
system, vessel, equipment, output	identifier	OPTIONAL	Where it applies: the Data Model system, vessel, and equipment, and the Automation output.
parameter	string	OPTIONAL	The Data Model parameter, for range alarms.
value	object	OPTIONAL	The value that raised it: <code>value</code> , <code>unit</code> , and optionally <code>qualifier</code> , <code>time</code> , <code>method</code> , and the <code>device</code> and <code>endpoint</code> that measured it.
limit	object	OPTIONAL	The bound or limit crossed: <code>value</code> and <code>unit</code> .
direction	string	OPTIONAL	<code>high</code> or <code>low</code> .
details	object	OPTIONAL	Details of its code, such as a notice's (Section 13).
notified	object	OPTIONAL	In history: how many <code>messages</code> were sent, when the <code>first</code> was sent, and the escalation <code>step</code> reached.
extensions	object	OPTIONAL	Extensions.

The `examples` directory contains `message-critical.json` and `message-power-outage.json`.

16.7 Delivery and Failures

1. A controller SHOULD send each message within 5 s of it falling due (provisional).
2. When a channel fails to deliver a message, the controller SHOULD try again at growing intervals, from 10 s to at most 10 minutes, until it succeeds, the alarm closes, or 24 hours pass (provisional).

3. When a message about an alarm has not been delivered through a channel within 5 minutes, the controller **MUST** raise `notification-failed` for that channel (Section 8.5). Its notifications go through the other channels, and its local sound, if it is critical, makes sure someone at the tank hears it.
4. `notification-failed` clears when the channel next delivers a message.
5. A channel delivers a message when: an app session or MQTT broker accepts it, with QoS 1 for MQTT; a push service or webhook receiver responds with a 2xx status [RFC 9110]; or an SMTP server accepts it for delivery [RFC 6409]. Delivery to a service does not prove that a person has seen it; only acknowledgement does.

16.8 Tests

1. A user **MUST** be able to send a `test` message to any recipient or feed they may change (Table 14).
2. A controller **SHOULD** offer a test whenever a channel or recipient is added or changed, and **SHOULD** remind owners to test their channels every 90 days (provisional).
3. A test message says clearly that it is a test, and needs no acknowledgement.

16.9 Content and Privacy

Alarm messages leave the owner's network, as Security and Access Table 12 allows for notification services the owner chose.

1. A channel's `detail` is `full` (default) or `minimal`. A `minimal` message contains only the severity, the code, a title that names the system but no values, such as "Critical alarm: Living room reef", and the times; it leaves out `text`, `value`, `limit`, `details`, the subject's members other than `kind`, and the place.
2. A controller **SHOULD** propose `minimal` for channels whose messages a third party can read, such as a text message gateway. Push messages are encrypted from end to end (Section 17.2), so they need not be minimal.
3. Messages **MUST NOT** contain credentials, codes, keys, or the system's location.

17 Channels

17.1 Apps

1. The `app` channel delivers a message to each app installation of the recipient's user: over the installation's session with the controller, if it has one, and otherwise by push to the installation's push subscription (Section 17.2).
2. Apps receive messages, and acknowledge and silence alarms, through the controller API, which is yet to be specified (Appendix D, issue 10).
3. An app **SHOULD** use the most urgent class of notification its platform offers for critical alarms. For example, one phone operating system lets an app put notifications in a channel of high importance, which makes a sound and appears over other apps, and which the user may let override Do Not Disturb. Another offers critical alerts, which sound even when the phone is muted or in Do Not Disturb, to apps that the platform's maker has approved. It **SHOULD** let the user choose a different sound for each severity.

17.2 Push

A phone or browser that is not connected to the controller receives notifications through a push service: the service of its platform, or one the user chooses. The controller sends to it with the Web Push protocol [RFC 8030], which every current browser supports, and which UnifiedPush uses for push services that people can run themselves [UnifiedPush].

1. **Subscriptions.** An app installation registers a push subscription with the controller: the push service's endpoint URL, and the `p256dh` key and `auth` secret of Message Encryption for Web Push [RFC 8291]. A user registers subscriptions only for their own installations (Table 14), and the controller removes them when the installation's credential is revoked [OAP-SA Section 11.7].

2. **Identity.** The controller has its own VAPID key pair [RFC 8292], generated on the controller and never shared, and gives apps its public key so they can subscribe. Its VAPID tokens carry a `sub` contact URI that the owner chooses, because some push services require one.
3. **Encryption.** Every push message MUST be encrypted from end to end with Message Encryption for Web Push [RFC 8291], so that no push service, gateway, or relay can read it [OAP-SA Section 13.3.5].
4. **Headers.** The controller sends each message with a `TTL` of 86 400 s, an `Urgency` of `high` for critical alarms, `normal` for warnings, and `low` for information [RFC 8030 Section 5.3], and a `Topic` of the alarm's `id` without its hyphens [RFC 8030 Section 5.4], so that a newer message about an alarm replaces an older one that has not been delivered.
5. **Size.** The plaintext of a push message MUST NOT be longer than 3993 bytes [RFC 8291 Section 4]. A controller shortens a longer alarm message by leaving out, in order, `details`, `text`, and `value`.
6. **Responses.** A `201` response is delivery. On `404` or `410`, the subscription has ended: the controller removes it, and tells the user through their other contacts and in the app. On `413`, it shortens the message; on `429` or `5xx`, it tries again (Section 16.7).
7. **Gateways.** A native app on a platform whose push service accepts messages only with credentials belonging to the app's maker needs a push gateway run by that maker. Such a gateway is a maker's service (Section 16.2): the app's subscription names it as the endpoint, and it receives messages it cannot read. An app SHOULD also offer a way to receive notifications that needs no maker's service, such as Web Push in a web app, which current versions of one phone operating system support for web apps added to the home screen, without any developer account, or UnifiedPush on another [UnifiedPush].

17.3 Email

1. A controller submits email to the SMTP server of the channel [RFC 6409], with implicit TLS (`security tls`, port 465 by default) or STARTTLS (`starttls`, port 587 by default) [RFC 8314]. It MUST verify the server's certificate, and MUST NOT send a password before TLS is established.
2. Each email has the channel's `from` address, the contact's `address`, and a subject made of the severity, the system's name, and the title, such as "[Critical] Living room reef: Temperature above its danger limit".
3. Its body is plain text in UTF-8: the title, the text, the value and limit, when the alarm was raised, its state, and that the alarm is acknowledged in the app. It MAY carry the alarm message as an attachment of type `application/json`.
4. Each email has its own `Message-ID`, and every email about one alarm SHOULD refer to the first with `In-Reply-To` and `References` [RFC 5322 Section 3.6.4], so that mail programs show them together.
5. Each email SHOULD carry `Auto-Submitted: auto-generated` [RFC 3834], so that mail systems do not answer it automatically.
6. Some email providers accept SMTP submission only with an app password or with OAuth; this revision supports a username and password (Appendix D, issue 21).

17.4 Webhooks

1. A controller delivers a message to a webhook by an HTTP `POST` [RFC 9110] to the channel's `url`, with the alarm message as the body and `Content-Type: application/json`.
2. The URL MUST use `https`, except to a host on the local network, such as a home automation server; the controller MUST verify the server's certificate.
3. Each request is signed as the Standard Webhooks specification defines [Standard Webhooks]:
 - `webhook-id`: the message's `id`;
 - `webhook-timestamp`: the time of sending, in whole seconds since 1970-01-01T00:00:00Z; and
 - `webhook-signature`: `v1`, followed by the base64 encoding [RFC 4648] of the HMAC-SHA256 [RFC 2104] [FIPS 180-4] of the `webhook-id`, a full stop, the `webhook-timestamp`, a full stop, and the body, with the key decoded from the channel's `secret`.
4. A `secret` is `whsec_` followed by the base64 encoding of 24 to 64 random bytes. A controller generates it when an owner adds the channel, and shows it once, for the owner to give to the receiver; an owner MAY instead enter one the receiver gives.

5. A receiver SHOULD check the signature, comparing in constant time, SHOULD refuse a timestamp more than 5 minutes from its own clock, and SHOULD ignore a `webhook-id` it has seen before, so that it is not fooled by forged or replayed messages.
6. A `2xx` response is delivery. On `410 Gone`, the controller stops using the channel and raises `notification-failed`; on any other response, or none within 10 s, it tries again (Section 16.7).
7. A controller SHOULD keep bodies under 20 kB, splitting a summary if it must.

For example, the heartbeat

```
{"oapAlarmMessage": "0.2", "id": "5b0e8f9c-3d2a-4c71-9e46-2f8a1b7c0d93", "kind": "heartbeat", "sent": "2026-10-14T09:30:00Z", "keyId": "q8mB0x4yZp2Vt7LrKc3sHw9dNf1gJa6eUo5iXb2LTvE", "name": "Reef controller"}}, sent with the secret whsec_dvYkxV69fBt/Zt8zZlakek1TSSM3TdjIVPFVQS41NZ0= at webhook-timestamp 1791934200, has the webhook-signature v1,jfyDXKfoCsKwm4Q/72V0YbPmMmz0jvoQrZITgIK/BSw=.
```

17.5 MQTT

1. A controller connects to the channel's broker with MQTT version 5.0 [MQTT 5] or 3.1.1 [MQTT 3.1.1], as `version` says (default 5.0), and MUST support both.
2. The URL MUST use `mqtt`s (port 8883 by default), except to a broker on the local network, which MAY use `mqtt` (port 1883 by default); the controller MUST verify the broker's certificate.
3. It publishes under its base topic, `baseTopic`, by default `oap/` followed by its key ID, the topics of Table 26, with QoS 1, and payloads in UTF-8.

Table 26 — MQTT topics

Topic	Retained	Payload
<code><base>/status</code>	Yes	<code>online</code> while the controller is connected; <code>offline</code> when it disconnects, or, as its Will, when the broker loses it.
<code><base>/alarms/<id></code>	Yes	The latest message about each alarm that is not closed, where <code><id></code> is the alarm's <code>id</code> . A zero-length retained message removes it when the alarm closes [MQTT 5 Section 3.3.1.3].
<code><base>/messages</code>	No	Every message the feed receives (Section 16.2).
<code><base>/heartbeat</code>	No	Heartbeats (Section 18.2).

4. On connecting, the controller sets a Will on `<base>/status` with the payload `offline`, retained [MQTT 5 Section 3.1.2.5], and then publishes `online`, retained. It MAY set a Will Delay Interval of up to 60 s, so that a brief reconnection does not report it offline. Before it disconnects on purpose, it publishes `offline`.
5. After reconnecting, it publishes the retained topic of every alarm that is not closed again, and removes those of alarms that closed meanwhile.
6. In this revision, a controller MUST NOT act on anything it receives over MQTT, such as an acknowledgement, because MQTT messages do not identify a user (Appendix D, issue 10).

A home automation server can show the retained alarm topics as entities, and `<base>/status` tells it, without polling, when the controller has gone.

17.6 Other Channels, Text Messages, and Calls

1. A channel `type` MAY be an extension name, for channels this Specification does not define, such as a chat service. Such a channel meets every requirement of Section 16 that applies to it.
2. This revision defines no channel for text messages or phone calls (provisional; Appendix D, issue 5). They cost money for every message, need an account with a provider, and differ by country. In October 2026, a text message through one large provider cost about US\$0.01 in the United States and US\$0.05 to US\$0.11 in Australia, the United Kingdom, and Germany. US carriers have ended or are ending their free email-to-text gateways, and two makers of commercial controllers have stopped offering text messages.
3. An owner who wants text messages or calls can send webhooks to a provider of their choice, as a contact of a recipient, with `detail minimal` (Section 16.9).

18 Supervising the Alarm Path

18.1 A Failed Controller Cannot Raise an Alarm

A controller that has lost power, stopped running, or lost its network cannot tell anyone so. Other equipment must notice its silence:

1. **Devices** apply their fail-safe behavior, and SHOULD show lost contact on their indicators [OAP-FS Section 7.3].
2. **Alarm outputs** configured to run on loss of contact sound when the controller stops supervising them (Section 15.2).
3. **Heartbeats** let a monitoring service the owner chooses notice when they stop (Section 18.2).
4. **MQTT's Will** tells a home automation server when the broker loses the controller (Section 17.5).
5. **Apps** SHOULD show when they last heard from the controller, and SHOULD say so when they cannot reach it through a path that the owner set up for remote access.

18.2 Heartbeats

1. An owner MAY add heartbeats, each with a webhook or MQTT `channel`, and an `interval` of 60 s to 3600 s, by default 300 s.
2. Every `interval`, the controller sends a `heartbeat` message: by `POST` to a webhook's URL, signed as in Section 17.4, or to `<base>/heartbeat` over MQTT. A webhook for heartbeats can be the URL of a monitoring service that alerts the owner when the heartbeats stop.
3. A controller MUST create heartbeats in the process that runs its control functions and alarms, as it does supervision messages [OAP-FS Section 7.1.4], so that a controller whose control has stopped also stops its heartbeats.
4. A heartbeat carries no alarms, so it reveals only that the controller is running.

19 Alarm Settings Documents

19.1 Encoding

1. A controller's alarm settings are written as an alarm settings document: a JSON text [RFC 8259] in UTF-8, whose top-level value is an object.
2. It MUST be valid against `schema/alarm-settings.schema.json` [JSON Schema], and meet every requirement of this Specification.
3. Numbers are finite, durations are in seconds, and timestamps are Data Model Timestamps [OAP-DM Section 7.4].

19.2 Top-Level Members

Table 27 — Top-level members of an alarm settings document

Member	Type	Presence	Description
<code>oapAlarms</code>	string	REQUIRED	Revision of this Specification. For this revision, "0.2". A controller of this revision also reads "0.1" documents, whose power checks have no <code>id</code> , and gives each power check an <code>id</code> when it imports one.
<code>generator</code>	object	OPTIONAL	The product that wrote the document, as a Data Model Generator.
<code>created</code>	timestamp	OPTIONAL	When the document was written.
<code>name, notes</code>	string	OPTIONAL	A name and free text.
<code>timeZone</code>	string	REQUIRED	The time zone of quiet hours, as a name from the Time Zone Database [TZDB]. It MUST be the automation configuration's [OAP-AU Section 7.2].
<code>posixTimeZone</code>	string	OPTIONAL	The same time zone as a POSIX <code>TZ</code> string, as in Automation Section 7.2.
<code>systems</code>	array	REQUIRED	The Data Model systems the settings serve, as in the automation configuration. At least one.
<code>ranges</code>	array	OPTIONAL	Ranges (Section 9).

Member	Type	Presence	Description
<code>powerChecks</code>	array	OPTIONAL	Power checks (Section 10).
<code>customAlarms</code>	array	OPTIONAL	Custom alarms (Section 11).
<code>conditions</code>	array	OPTIONAL	Condition settings (Section 7.3).
<code>local</code>	object	OPTIONAL	Local signals: <code>quietHours</code> (Section 16.5), <code>silenceDuration</code> (60 s to 3600 s, default 900 s; Section 12.5), and <code>sound</code> , the sound of warning (<code>intermittent</code> , <code>once</code> , or <code>none</code>) and of <code>info</code> (<code>once</code> or <code>none</code>) alarms (Section 14.1).
<code>recipients</code>	array	OPTIONAL	Recipients (Section 16.1).
<code>channels</code>	array	OPTIONAL	Channels (Section 16.2).
<code>policies</code>	array	OPTIONAL	Escalation policies (Section 16.3).
<code>rules</code>	array	OPTIONAL	Rules (Section 16.4).
<code>heartbeats</code>	array	OPTIONAL	Heartbeats (Section 18.2).
<code>extensions</code>	object	OPTIONAL	Extensions, as in Data Model Section 7.6.

19.3 Identifiers and References

- Identifiers are Data Model Identifiers [OAP-DM Section 7.3]. Every `id` of a range, power check, custom alarm, recipient, channel, and policy **MUST** be unique in the document, and different from every system and vessel.
- References within the document, such as a policy's recipients and channels, **MUST** name objects of the right kind in the same document.
- References to the automation configuration, the `output` of a power check and the `when` of a custom alarm, name its elements [OAP-AU Section 7.3]. A controller **MUST** check them whenever either document changes, and, while it cannot resolve one, **MUST** raise `reference-missing` for the power check or custom alarm that holds it, which then raises no other alarm. The automation configuration's own `binding-missing` stays for outputs and inputs without a usable binding [OAP-AU Section 8.5].
- Members that this Specification does not define are not allowed, except in `extensions`. Consumers **MUST** ignore extensions they do not understand, and **SHOULD** preserve them.

19.4 Secrets

- Passwords and webhook secrets are secrets. A controller **MUST** keep them as Security and Access Section 8.4 requires, and **MUST NOT** export them [OAP-SA Section 16.6].
- A document without them is valid; after importing one, the owner enters them again.
- Push subscriptions and the controller's VAPID key pair are not part of alarm settings.

19.5 Export and Import

- A controller **MUST** be able to export its alarm settings as a document at any time, without a network service.
- A controller **MUST** validate a document before using it, and **MUST** reject it, without changing its settings, if it is not valid.
- Before using an imported document, a controller **MUST** show an owner what it changes, and in particular every recipient and channel that alarms would leave the owner's network through, and **MUST** have the owner confirm it.
- Ranges, power checks, and custom alarms can be shared with other owners. Before sharing, a controller **SHOULD** remove recipients, channels, heartbeats, and `notes`, which identify people.

20 History

20.1 Alarm Records

1. When an alarm closes, the controller **MUST** keep it in its history as a Data Model event [OAP-DM Section 8.6]. An export of history **SHOULD** also include the alarms that are not closed, without `end` if they are active.
2. In Data Model revision 0.1, an alarm event is encoded as follows (provisional, until the changes of Section 20.3):
 - `type` is `org.openaquariumproject.alarm`;
 - `time` is when the alarm was raised, and `end`, when its condition cleared;
 - `system`, `vessel`, and `equipment` are the alarm's, where known; and
 - `extensions` holds, under `org.openaquariumproject.alarm`, the alarm as Table 25 defines it, with `notified`.
3. An alarm about the controller as a whole is recorded once for each system the controller serves.

20.2 Notices in History

1. A `power-outage` notice is recorded as a Data Model `power-outage` event, whose `time` and `end` are the outage's start and end, with the notice under `extensions` as in Section 20.1.
2. `controller-restarted` and `device-restarted` are recorded as events of type `org.openaquariumproject.restart` (provisional), whose `time` is when the controller or device restarted, whose `equipment` is its Data Model equipment, where known, and with the notice under `extensions`.
3. Other notices are recorded as in Section 20.1, with `time` the time they were raised.

The example `history.json` is a Data Model document with each kind of record.

20.3 Proposed Data Model Changes

Table 28 — Changes proposed for Data Model revision 0.2

Change	Purpose	Until then
An event type <code>alarm</code> , and an Event member <code>alarm</code> that holds an alarm (Table 25).	Alarms in history, readable by any product.	The event type and extension <code>org.openaquariumproject.alarm</code> (Section 20.1).
An event type <code>restart</code> , with the <code>reason</code> and <code>downtime</code> of Section 13.3.	Restarts of controllers and devices in history.	The event type <code>org.openaquariumproject.restart</code> .
A top-level member <code>ranges</code> , of Range objects with the members of Table 10 other than <code>delay</code> and <code>hysteresis</code> .	Owners' target ranges, for charts, reports, and moving to a new product (Data Model Appendix C, issue 12).	Ranges in alarm settings documents (Section 19).

20.4 Keeping History

1. A controller **SHOULD** keep alarm records for at least one year (provisional).
2. Users with the `export-data` permission can export them with the rest of the history [OAP-SA Section 16.6], and erasing history erases them.
3. Acknowledgements and remote silences are also kept in the audit log [OAP-SA Section 12], which history erasure does not erase.

21 Conformance

21.1 Controllers

A conforming controller:

1. raises alarms for every code of `conditions.json` whose condition it can detect, with the severities of Section 7 (Section 8);
2. raises range alarms, power check alarms, and custom alarms (Sections 9 to 11);

3. raises, acknowledges, silences, clears, latches, repeats, reminds, groups, and holds alarms as Section 12 defines, and keeps them across restarts;
4. raises notices of power outages and restarts, and keeps them until acknowledged (Section 13);
5. has a local signal for critical alarms, or raises `local-signal-missing` (Section 14.1), and sounds and shows alarms as Section 14 defines, with the sounder, indicators, buttons, and display it has;
6. drives alarm functions (Section 15);
7. supports the `app`, `email`, `webhook`, and `mqtt` channels, escalation policies, rules, and quiet hours, and needs no maker's service (Sections 16 and 17);
8. sends heartbeats from its control process (Section 18.2);
9. imports and exports alarm settings documents (Section 19), and keeps alarms in history (Section 20); and
10. passes the test procedures of Section 21.6.

21.2 Devices

A conforming device reports its faults as the Device Classes specification defines [OAP-DC Section 7.6], and SHOULD signal dangerous faults locally (Section 14.4). This Specification asks nothing else of devices.

21.3 Apps

A conforming app:

1. shows every alarm that is not closed, critical alarms first, with the members of Table 25 that a person needs;
2. lets users acknowledge and silence alarms with their credentials, as their role permits (Section 12.10);
3. registers Web Push subscriptions, if it receives push (Section 17.2); and
4. lets users send tests (Section 16.8).

21.4 Documents and Messages

A conforming alarm settings document or alarm message is valid against its JSON Schema and meets every requirement of this Specification that applies to it, including those a schema cannot express, such as the order of a range's bounds, and references.

21.5 Machine-Readable Definitions and Tools

`conditions.json` is normative, and Tables 2, 3, 6, 7, 8, 9, and 15 are generated from it by `tools/render_spec_tables.py`, which also checks that every Device Classes fault code and every Automation fault code has a default severity. The JSON Schemas, `schema/alarm-settings.schema.json` and `schema/alarm-message.schema.json`, are normative for structure; where they and this document differ, this document governs.

The validator, `tools/validate_alarms.py`, checks alarm settings documents, alarm messages, the alarm functions of automation configurations, and the alarm records of Data Model documents. It also signs and verifies webhooks as Section 17.4 defines. It is informative.

The `examples` directory contains:

- `reef-system.json`: the alarm settings of the reef system of the Automation examples, with ranges, power checks, a custom alarm, condition settings, local quiet hours, three recipients, push, email, a text message gateway, a home automation feed, MQTT, a critical policy that escalates from push to email to a neighbor, and heartbeats;
- `minimal.json`: one range, with every other setting at its default;
- `alarm-outputs.json`: an automation configuration with a siren and an alarm panel contact (Section 15);
- `message-critical.json` and `message-power-outage.json`: alarm messages; and
- `history.json`: a Data Model document with alarm records (Section 20).

21.6 Test Procedures

Tests use a controller with modules or simulated devices whose faults and readings the tester can set, a clock the tester can set, and a mail server, webhook receiver, MQTT broker, and push service under the tester's

control. Each test names the requirements it checks. Every MUST and MUST NOT of Sections 7 to 20 and 22 is checked by at least one test.

Table 29 — Test procedures

Test	Requirements	Method	Pass criteria
Fault severities	7.2, 7.3.1 to 7.3.3, 8.1.5, 8.4	Make devices report each Device Classes fault, on endpoints that drive a light and endpoints that drive a heater.	Each raises an alarm with the default severity of Table 6 for the light, and is critical on the heater where Table 6 lists hazardous or vital .
Protected alarms	7.3.5, 7.3.6	Try to turn off leak , and stuck-on for an outlet that drives a heater, and to lower each to info ; then lower stuck-on for that outlet to warning . Turn off stuck-on for an outlet that drives a light, bind the outlet to a heater, and make it report stuck-on .	The controller refuses the first four settings. It asks the owner to confirm the fifth, and shows it to every owner. The last alarm is raised, and is critical .
Standing warnings	7.3.7	Try to turn off local-signal-missing , and to change the severity of safety-path-missing ; then acknowledge an active standing warning.	The controller refuses both settings. The acknowledged warning stays shown to every owner while it is active.
Local signal	14.1.1, 14.1.2, 21.1	Start a controller that has no sounder, and no alarm output; then configure an alarm function that signals critical alarms on a bound siren.	local-signal-missing is raised, and shown to every owner, until the alarm function is configured; then it clears, and a critical alarm sounds the siren.
Critical sound	7.1.2, 14.1.5	Raise a critical alarm during local quiet hours, with every setting at its most permissive.	The sounder plays the critical pattern until it is silenced or the alarm acknowledged.
Indicators	14.2.1	On a controller with an alarm indicator, raise a warning, then a critical alarm; acknowledge the critical alarm; clear it.	The indicator follows Table 16 for the most severe alarm that is not closed.
Silence	12.5, 14.1.4, 14.3.1	Raise a critical alarm, press the button, then wait 15 minutes.	The sound stops at once without any credential; notifications go on; the sound resumes after 15 minutes.
Device sounders	12.5.1, 14.4.3, 14.4.4	Make a top-off unit that implements silence report high-level with sounding true; silence alarms at the controller; then acknowledge the alarm.	The controller shows that the unit is sounding. It sends the unit silence with the silence duration, and, after the acknowledgement, silence without a duration.
Acknowledgement	12.4	Raise a critical alarm notified to two recipients; reply to its email with "ACK"; try to acknowledge it as a user with the view role; then acknowledge it from one app as a user with the override role.	The reply and the view user change nothing. After the acknowledgement, repeats and escalation stop, and the audit log records who acknowledged it and when.
Escalation	12.6, 16.3	Use the policy of Section 16.3 and leave a critical alarm unacknowledged for 35 minutes.	Push at 0 minutes and every 10 minutes after; email from 10 minutes, and every 10 minutes after; the third recipient from 30 minutes.
Default rules	7.1.2, 16.4	With no rules, raise a critical alarm, a warning, and an info alarm; then add a rule that sends critical alarms to nobody.	Every owner's apps are notified of the critical alarm and the warning, and nobody of the info alarm. The rule is shown to every owner.
Latching	12.1, 12.3	Raise a critical and a warning alarm, then end both conditions before anyone acknowledges them.	The warning closes after its clear delay. The critical alarm stops sounding and escalating, sends cleared , and stays until acknowledged.
Chattering	12.3.1	Make a leak sensor change every 5 s for a minute.	One alarm, which clears 30 s after the last change.
Sensor faults	7.3.5, 7.3.6, 8.4.3	Make a leak sensor, an overflow switch, and a level switch whose position is high each report wiring-fault , then hardware-failure ; then try to turn off wiring-fault for the leak sensor.	Each alarm of the leak sensor and the overflow switch is critical ; those of the high-level switch are warnings. The setting is refused.

Test	Requirements	Method	Pass criteria
One alarm per condition	8.3	Make a power strip report <code>stuck-on</code> on an outlet whose power the controller's power check also sees; end the condition before the critical alarm is acknowledged, and bring it about again.	There is one alarm, with the endpoint as its subject, and it becomes active again.
Lost devices	8.4.1	Make a device report <code>overtemperature</code> , then disconnect it.	The alarm stays active while the device is lost.
Delays	12.2	Make a pump's flow fall below its <code>minFlow</code> , and keep it there.	<code>flow-low</code> is raised 60 s after the flow fell, within 5 s, and not later.
Modes	12.8.1 to 12.8.3	Start a maintenance mode, then raise a warning and a critical alarm; end the mode.	The critical alarm sounds and is notified at once. The warning is held, and sounds and is notified when the mode ends, while it is still active.
Expected conditions	12.8.4, 12.8.5, 12.8.8	Run an automatic water change whose drain takes a sump below its low-level switch, with a feed configured, and end it with the sump still low; run another, and abort it during the fill.	<code>low-water</code> is raised, shown, and sent to the feed during the drain, but neither sounds nor is notified. It sounds and is notified as soon as the water change ends or aborts.
Calibration holds	12.8.6, 12.8.7	Calibrate the pH probe that a CO2 function uses, finish the calibration, and confirm the probe a minute later; then calibrate the temperature probe that a heater function uses.	The pH input's <code>input-unavailable</code> , a warning, is held until the calibration ends, then sounds and is notified while it is still active. The heater's <code>input-unavailable</code> , critical, is never held.
Quiet hours	16.5	During a recipient's quiet hours, raise a warning and a critical alarm.	The critical alarm is notified at once; the warning is sent in a <code>summary</code> when quiet hours end, if still unacknowledged.
Ranges	9	Sweep a temperature reading through every zone of a range and back, then record a manual test beyond a danger limit, then report a conductivity reading of <code>> 80 mS/cm</code> with <code>quality good</code> against a <code>danger.above</code> of 60.	Each condition is raised after <code>delay</code> and cleared with <code>hysteresis</code> , as Section 9.4 says; the manual test raises its alarms at once, which stay until a later test is inside or someone acknowledges them; the conductivity reading raises <code>in-danger-range</code> .
Power checks	10.2, 10.3.1	Draw power from an outlet that is off; unplug a running load; halve its power. Repeat on an outlet with <code>power-monitoring</code> and no power check.	<code>stuck-on</code> , <code>load-disconnected</code> , and <code>power-abnormal</code> are raised after their delays, with the endpoint as subject; without a power check, <code>stuck-on</code> is raised above 2 W.
Owner's confirmation	9.1.4, 10.3.3, 19.5.3	Let the controller propose a range and a power check that it learned; import an alarm settings document that adds a webhook.	Nothing takes effect until an owner confirms it. Before the import, the controller shows every recipient and channel that alarms would leave the network through.
Consequences	8.8	Disconnect a probe module that the heater's function uses.	<code>device-lost</code> is notified after its delay, listing the inputs that became unavailable; they are not notified on their own unless more severe.
Battery	8.5.3	Run the controller on its battery until its charge is below 20 %.	<code>battery-low</code> is raised.
Power outage	8.5.2, 13.2	With a battery, remove mains power for 30 s, then for 5 minutes. Without a battery, remove it for 5 minutes, 4 minutes after the controller last recorded the time.	With a battery: the 30 s outage raises a <code>power-outage</code> notice only; the 5-minute outage raises <code>mains-power-lost</code> after 60 s, then a notice when power returns. Without: a notice after the restart, whose start is the time last recorded. Each notice stays until acknowledged.
Restart	13.3	Force the controller's watchdog, then restart it with the <code>restart</code> operation; force a device's watchdog.	<code>controller-restarted</code> after the first, nothing after the second, and <code>device-restarted</code> after the third.

Test	Requirements	Method	Pass criteria
Matter fabrics	13.4.4	With a rule that sends warnings to nobody, add a Matter fabric to the controller's bridge from a home app, then remove it.	<code>fabric-added</code> and <code>fabric-removed</code> are raised, each with the fabric's label and vendor, and every owner is notified of each.
Alarm outputs	15	Raise a critical alarm with the outputs of <code>alarm-outputs.json</code> , silence it, then stop the controller's control process.	The siren follows <code>sounding</code> ; the panel contact opens while the alarm is active; when the controller stops, the siren sounds and the contact opens at their fail-safe timeouts.
Channels	16.2.3, 16.2.5, 16.9.3, 17, 22.5	Deliver a critical alarm through each of <code>app</code> (session and push), <code>email</code> , <code>webhook</code> , and <code>mqtt</code> , the last with MQTT 5.0 and with 3.1.1; then repeat with servers whose certificates are not valid, and with a push endpoint over <code>http</code> .	Each delivers a valid alarm message with no credentials, codes, keys, or location; push is encrypted [RFC 8291], no longer than 3993 bytes, with <code>Urgency high</code> ; the webhook's signature verifies; the MQTT retained topic appears and is removed when the alarm closes. With invalid certificates and <code>http</code> , nothing is delivered, and no password is sent.
Delivery failure	8.5.6, 16.7	Make the webhook receiver answer 500 for 6 minutes for a warning, then for a critical alarm.	Retries at growing intervals; <code>notification-failed</code> as a warning after 5 minutes, rising to critical when the critical alarm fails too; it clears on the next delivery.
No maker's service	16.2.1	Block every internet host except those of the test's own services.	Every test above that does not use push still passes.
Tests	16.8.1	As a user, send a test to a recipient and a feed that the user may change.	Each receives a <code>test</code> message that says it is a test.
Heartbeats	18.2.3	Stop the controller's control process while its network keeps running.	Heartbeats stop.
MQTT	17.5.4, 17.5.6	Publish an acknowledgement of an active alarm to the broker; then cut the controller's network.	The alarm stays unacknowledged. The broker publishes <code>offline</code> on <code><base>/status</code> .
Restarts keep alarms	12.9	Restart the controller with an acknowledged active alarm, a latched alarm, and an unacknowledged notice.	Each keeps its state; escalation goes on from where it was.
Visibility	12.10	As a user with the <code>view</code> role, look at the alarms that are not closed, and at the history.	Every alarm that is not closed is shown with its severity, title, subject, state, times, and who acknowledged it; closed alarms are in the history.
Settings	19.1, 19.3.4, 19.4, 19.5.1, 19.5.2	Import <code>reef-system.json</code> and <code>minimal.json</code> , the first with an extension the controller does not understand, with the automation configuration of the reef system, then export; enter a password and a webhook secret, and export again.	Each is accepted after the owner confirms its channels; the export is valid and equal apart from <code>created</code> , <code>generator</code> , and secrets, and holds no password or secret.
Invalid settings	9.1, 10.2, 11, 16.3.4, 16.5, 19.1.2, 19.2, 19.3.1, 19.3.2, 19.5.2	Import alarm settings documents that each break one rule, such as: ranges whose bounds are out of order; two ranges for one parameter; a power check with <code>stopped.max</code> above <code>running.min</code> ; a custom alarm whose <code>when</code> depends on an alarm; a policy step whose channel is not a contact of its recipient; quiet hours that hold <code>critical</code> ; an <code>id</code> used twice; and a time zone other than the automation configuration's.	Each is rejected, and the settings are unchanged.
Missing references	19.3.3	Remove from the automation configuration the output that a power check names.	<code>reference-missing</code> is raised for the power check, which raises no other alarm until the output is back.
History	20.1	Close an alarm, then export the history.	The alarm is recorded as a Data Model event, as Section 20.1 says.

22 Security Considerations

1. **Blocked notifications.** Anyone who can block the network can stop remote notifications. Local signals, alarm outputs, heartbeats, and MQTT's Will are the defenses (Section 18).
2. **False alarms.** Anyone who can forge device reports can raise false alarms, or many of them. Authenticated bindings [OAP-SA Section 10] prevent forgery on the network, and grouping limits floods (Section 12.7).
3. **Forged acknowledgements.** Acknowledgement needs an authenticated user with the `override` role or physical presence (Section 12.4), so nobody can stop escalation by replying to an email or a text message, or by publishing to an MQTT broker.
4. **Webhooks.** Signatures, timestamps, and message IDs let receivers reject forged and replayed messages (Section 17.4). Secrets are kept as secrets and never exported (Section 19.4).
5. **Push endpoints.** Apps give the controller the URLs it sends push to, so a user could make the controller send requests to another host. A controller **MUST** send push only to `https` URLs, and **SHOULD** refuse push endpoints on the local network unless an owner approves them.
6. **What leaves the network.** Alarm messages say when something went wrong, and acknowledgements say when people are at home. Only owners choose where messages go (Table 14); push is encrypted from end to end; and `minimal` detail keeps values from services that can read them (Section 16.9).
7. **MQTT brokers.** Anyone who can subscribe to the broker's topics can read alarms. Owners **SHOULD** protect their broker with authentication and access control.
8. **Turning alarms off.** Nobody can turn off a protected alarm or a standing warning, and every owner can see settings that lower a protected alarm or send critical alarms to nobody (Sections 7.3 and 16.4). Changes to alarm settings are recorded in the audit log [OAP-SA Section 12].
9. **New ecosystems.** Any administrator of a Matter fabric can add another ecosystem to the controller's bridge, and so give it what the bridge allows. Every owner is told of each fabric added or removed (Section 13.4), so that a fabric nobody expected is noticed.

Appendix A — Approval Checklist

Complete this checklist before requesting Working Group approval. Link each item to the evidence.

Item	Status	Details
Section 2 describes compliance with all five tenets.	Done	
A public review period of at least 30 days has been announced and completed.	Not started	
At least two independent implementations exist, at least one from an organization not represented among the Authors.	Not started	
The implementations have exchanged alarm settings, and delivered alarm messages to each other's receivers.	Not started	
The implementations have passed the test procedures in Section 21.6.	Not started	
Every normative reference is publicly available.	Done	
The conditions file, schemas, and examples are published, and the examples pass the validator.	Done	
The Open Issues appendix is empty, or each open issue is explicitly deferred.	Not started	
No unresolved Exclusion Notices for this Specification are recorded in <code>Notices.md</code> .	Done	
The Maintainer has recorded consensus to approve.	Not started	

Appendix B — Scenarios (Informative)

How a system configured like `reef-system.json` responds.

Table 30 — Example scenarios

Scenario	What happens
The heater's relay welds shut at night.	<code>stuck-on</code> is critical, because heaters are hazardous. The buzzer, the alarm light, and the siren start; Alex and Sam get push notifications, repeated every 10 minutes; Alex gets an email after 10 minutes; Jordan, after 30. The range alarms follow as the temperature rises.
The return pump runs dry.	The pump reports <code>dry-run</code> , critical because return pumps are vital; its power check also sees it draw much less than usual.
A leak sensor gets wet while the owners are away.	<code>leak</code> is critical. The leak emergency mode stops the pumps [OAP-AU Section 14.7], which raises <code>emergency-mode</code> too; both are notified in one message. Nobody acknowledges, so Jordan gets an email and a text message after 30 minutes.
The power fails for two hours, with a battery.	After 60 s, <code>mains-power-lost</code> is critical. After about 90 minutes, <code>battery-low</code> too. When power returns, both clear, and the <code>power-outage</code> notice tells everyone what happened, until someone acknowledges it.
The power fails for ten seconds, with no battery.	The controller restarts and raises a <code>power-outage</code> notice, a warning, which it sends to Alex in the morning, because of quiet hours.
The Wi-Fi router restarts at 04:00 and is down for three minutes.	The power strip is lost after 30 s, but <code>device-lost</code> waits 5 minutes, so no alarm is raised. Its outlets' fail-safe behavior applies at 120 s, and ends when the controller resumes control.
A pH probe is unplugged while the owner cleans it, in a maintenance mode.	<code>sensor-disconnected</code> and <code>input-unavailable</code> are warnings, held while the mode lasts; the probe is back before the mode ends, so nobody is notified.
An automatic water change drains the sump below its low-level switch at 10:00.	<code>low-water</code> is critical, but expected while the water change drains, fills, and settles: it is raised, shown in the apps, and sent to the home automation feed, but neither sounds nor is notified. The sump refills, so it clears; because it latched, Alex and Sam are told of it when the water change ends, and it stays until one of them acknowledges it.
The owner's weekly alkalinity test is 6.8 dKH.	<code>outside-acceptable-range</code> (warning) and <code>outside-desired-range</code> (info) are raised at once, and stay until a later test is back inside.
The controller's firmware stops running.	Heartbeats stop, and the uptime monitor alerts Alex. The broker publishes <code>offline</code> . After 30 s, the siren's module applies its fail-safe behavior, and the siren sounds.

Appendix C — Commercial Alarm Features (Informative)

How the alarm features of commercial controllers correspond to this Specification, as a guide for importers. Each row describes the approach of one of four widely used commercial controller ecosystems.

Table 31 — Commercial alarm features

Approach	Their model	In this Specification
Programmed alarm outputs	Alarm outputs, one that sends email and two audible ones for alarms and for warnings, programmed with if-then statements; push and email; text messages ended in November 2025.	Custom alarms on Automation conditions (Section 11); the audible alarm and warning outputs become <code>critical</code> and <code>warning</code> , and alarm functions drive any outputs that followed them.
Partial severities with repeats	Partial severities; push, email, and a legacy text message service; repeats while active.	Severities, channels, and repeats of Sections 7, 12.6, and 17.
Color levels tied to channels	Three levels per input or output: yellow (email), orange (email and push), and red (email, push, and sound); an alert output; a notification interval of 1 hour by default. Text messages were removed in 2021.	Red is <code>critical</code> ; orange is <code>warning</code> ; yellow is <code>warning</code> with an email-only policy, or <code>info</code> . The alert output becomes an alarm function.
Desired and acceptable ranges	Owner-set desired and acceptable ranges per parameter; everything outside the acceptable range is "danger", with an optional notification and audible alarm.	Desired and acceptable ranges map directly (Section 9). The danger zone of this model is <code>outside-acceptable-range</code> ; an owner who wants it critical sets danger limits equal to the acceptable range.

Appendix D — Open Issues (Informative)

Issues 1 to 27 were open in revision 0.1. The Project Lead decided issues 3, 27, and 28 to 30 on 15 October 2026, issue 4 through Fail-Safe Behavior Appendix C, issue 10, and issue 31 on 16 October 2026. Each issue that revision 0.2 resolves says how. Issue numbers do not change, because other specifications refer to them.

Questions for the Project Lead

1. **Severity levels.** This revision has three: `critical`, `warning`, and `info` (provisional). Alarm management practice recommends three or four priorities [ISA-18.2], and some alerting formats use five [CAP]. A fourth level above `critical`, for leaks and other threats to property, is the main alternative, at the cost of making `critical` mean less. The names avoid `advisory`, `caution`, and `alarm`, which Cyphal's heartbeat health on the module bus already uses, or which every severity is. The recommendation is three levels with these names.
2. **Equipment groups and default severities.** The groups of Table 3 and the defaults of Tables 6 to 9 and 15 need review by experienced aquarists, like the recommended fail-safe defaults.
3. **Lowering critical alarms.** *Resolved in revision 0.2.* The Project Lead decided that alarms that are critical through their equipment group (Section 7.2) are protected as codes whose default is critical are. Section 7.3 now calls both protected alarms: owners cannot turn them off, and can lower them no further than `warning`, after confirming, with the setting shown to every owner. A setting that a protected alarm would break never applies to it, even when the setting was made before the alarm's subject joined the group.
4. **Alarm outputs.** *Resolved in revision 0.2.* The Project Lead decided Fail-Safe Behavior Appendix C, issue 10: alarm outputs have the Data Model equipment category `alarm-signal`, which Data Model revision 0.3 adds, recommended in Fail-Safe Behavior Table 3 to run on loss of contact, to apply `fail-safe` after power-up, with a power-on delay of 120 s, and, on battery, to be left to their alarm function. Section 15.2 now uses the category and the recommendation, and is no longer provisional; the Automation specification's power-loss modes leave such outputs to their function (its Section 14.6).
5. **Text messages and calls.** Should they be in scope, given their cost (Section 17.6)? The recommendation is to keep them out of the required channels, reach them through webhooks to providers that owners choose, and define a common webhook format for text message providers if makers want one.
6. **Required channels.** Every controller must support push, email, webhooks, and MQTT (Section 16.2). A small controller might support fewer under a declared conformance level, as Automation Appendix D, issue 13, asks for automation.
7. **Defaults without settings.** A controller with no rules notifies its owners' apps of critical alarms and warnings (Section 16.4). Should warnings wait until an owner has chosen?

For other specifications

8. **Security and Access: operations.** Registering push subscriptions and one's own quiet hours use `manage-own-credentials`, and changing recipients and channels uses `manage-consent` (Table 14), provisionally. Dedicated operations, such as `manage-notifications` and `manage-own-notifications`, may be clearer. Section 17.2.3 answers Security and Access Appendix D, issue 25.
9. **Automation: the alarm function.** The Automation specification should list `alarm` among its function types, in place of the provisional extension name `org.openaquariumproject.alarm` (Section 15.1). Table 8 and Section 15 answer Automation Appendix D, issue 18.
10. **Device Protocol: a controller API.** Apps need methods to list and subscribe to alarms, acknowledge, silence, send tests, and register push subscriptions, and the controller's VAPID public key [OAP-DP Appendix C, issue 1]. An MQTT gateway (Device Protocol Appendix C, issue 2) could share the topics of Section 17.5, and would need a way to identify users before it accepts acknowledgements.
11. **Data Model.** The changes of Section 20.3: event types `alarm` and `restart`, and target ranges (Data Model Appendix C, issue 12). Data Model revision 0.2 adds them: the event types `alarm` and `restart`, an Event member `alarm`, and Ranges without `delay` and `hysteresis` (its Sections 8.9 and 8.13). This Specification should record its history with them (Section 20) in its next revision.
12. **Device Classes: device sounders.** *Resolved in revision 0.2.* Device Classes revision 0.2 adds `sounding` and `silence` (its Section 8.2), and Section 14.4 now uses them, so that an owner silences every alarm, a device's own included, in one place. It also adds the faults `clock-invalid`, `battery-weak`, `leak`, and `daily-limit`, which `conditions.json` gives default severities.

13. **Fail-Safe Behavior.** *Resolved by Fail-Safe Behavior revision 0.2.* Its Appendix C, issue 10, gives signaling equipment a recommendation (issue 4), and its Section 7.6 requires the fault for a lost device on which Section 8.5 relies.
14. **Water Level and Water Change, Dosing, and Water Testing.** These specifications should add their codes to `conditions.json`, such as a missed dose, a daily dosing limit reached, or a water change that could not finish, define the response to `leak` and `overflow`, and say which conditions a water change makes expected (Section 12.8). The `Water Level and Water Change` specification has done so for its part: it adds `low-water`, `daily-limit`, and five other codes (its Table 5), defines the responses to leaks and overflows (its Section 11), and makes `low-water`, a critical condition, expected while an automatic water change drains, fills, or settles, because the water change guards the level then (its Section 9.13). Section 12.8.4 now holds such a condition rather than leaving it unraised, and Water Level and Water Change Appendix D, issue 23, applies the same decision there. The `Dosing` specification adds `dose-missed`, `dose-adjust-limit`, and `calibration-drift` (its Table 5); a dose that a head refuses at its daily limit raises `command-refused`, and is then missed (its Section 10.2), at its default severity (Section 8.6.4). The `Water Testing` specification adds `results-stale`, `testing-suspended`, `correction-limit`, `waste-nearly-full`, and `test-due` (its Table 7), and raises `implausible-result` when a scheduled test and its retests give no accepted result (its Section 10.5). It proposes that `input-unavailable` for an input whose analyzer's results are stale, or whose testing is suspended, be a consequence of that condition (Section 8.8; its Appendix D, issue 11). The `Lighting Programs` specification [OAP-LP] adds `program-not-loaded` (its Table 5), and has lights report `overtemperature` while they dim themselves for heat (its Section 17.1). The `Flow Programs` specification adds `flow-low` and `flow-control-limit` (its Table 5), and raises `program-not-loaded` for pumps that do not run the programs their controllers write (its Section 15.2). Each of them gives the time a condition must last as its delay in `conditions.json`, not as part of the condition (Section 12.2.5).
15. **Module Bus.** `configuration-failed` and `node-conflict` replace the provisional codes of the breadboard controller firmware, and `capacity-exceeded` its codes for too many modules and a full allocation table.

Details of this Specification

16. **Numbers.** The delays, the clear delay of 30 s, the repeat and reminder intervals, the 5 minutes before `notification-failed`, the grouping window, the sound patterns, the sound level of 75 dB(A) at 1 m, the battery thresholds, the 5 minutes between recorded times, and the power check defaults need confirming on the breadboard controller and with owners.
17. **Shelving.** Alarm management practice lets operators shelve a nuisance alarm for a limited time [ISA-18.2]. This revision offers acknowledgement and condition settings instead. Should an owner be able to shelve an alarm, with an expiry, like an override?
18. **Trends.** Alarms on rates of change, such as a temperature falling by 1 °C an hour, or an alkalinity that has fallen for three tests in a row, are not defined.
19. **Languages.** Titles and text are in the controller's language. Apps can translate codes, but not owners' custom alarm names. Should messages carry a language tag, and should controllers offer several languages?
20. **Several controllers.** How alarms are shared between cooperating controllers, and how a standby controller takes over alarm duty, is not defined.
21. **Email authentication.** Some email providers require OAuth 2.0 for SMTP submission. Should controllers support it, given its need for a browser-based sign-in?
22. **Acknowledgement by people without apps.** A neighbor reached by email cannot acknowledge. A signed, single-use acknowledgement link would need a way to reach the controller from the internet, such as the relay of Security and Access Section 13.
23. **Ranges for derived values.** Ratios, such as calcium to magnesium, and values derived from several parameters, have no ranges.
24. **Ranges on devices.** Should ranges be given to devices, so that a device can signal a dangerous value by itself while its controller is lost?
25. **Test vectors.** A reference implementation of the alarm states, with recorded conditions and expected messages, would help implementations agree.
26. **Certification.** Should passing the test procedures be part of a future compatibility mark?

27. **Calibration holds.** *Resolved in revision 0.2.* The [Probe Interface](#) specification makes a probe's inputs unavailable from the start of its calibration until a person confirms the probe is back and its readings have settled (its Sections 14.7 and 14.8), and asks that the sound and notifications of their [input-unavailable](#) alarms be held, as during a maintenance mode. The Project Lead decided that a calibration hold never holds critical alarms, and that its hold of other alarms ends when the calibration ends: completed, failed, cancelled, or timed out. Sections 12.8.6 and 12.8.7 now say so. Raised by the Probe Interface specification (Appendix D, issue 9).

Decided by the Project Lead

28. **A local signal.** *Resolved in revision 0.2.* Every controller has a local signal for critical alarms: a built-in sounder, or a configured alarm output (Section 14.1). Until it has one, it raises [local-signal-missing](#), a standing warning that no owner can turn off (Section 7.3.7).
29. **Expected conditions.** *Resolved in revision 0.2.* A condition that the Water Level and Water Change specification makes expected is held, never dropped, and only for the codes and phases that specification names (Section 12.8.4). A held alarm, critical ones included, sounds and is notified as soon as the water change ends or aborts (Section 12.8.5).
30. **Matter fabrics.** *Resolved in revision 0.2.* The notices [fabric-added](#) and [fabric-removed](#) tell every owner when a Matter fabric is added to, or removed from, the controller's bridge, as the Matter Mapping specification requires (its Section 7.3.3), whatever the rules say (Section 13.4.4). The Project Lead's decision covers fabrics removed as well as those added.

The following issues arose in revision 0.2:

31. **Faults that hide a leak or an overflow.** *Decided by the Project Lead on 16 October 2026; resolved in revision 0.2.* [wiring-fault](#) and [hardware-failure](#) on a [leak-sensor](#) endpoint, or on a [level-switch](#) endpoint whose [position](#) is [overflow](#) or [low](#), leave its state unknown, so a leak, an overflow, or a low level would go unseen (Section 8.7.4). They were warnings, because equipment groups describe outputs, and a sensor has no output, so they disabled a safety function [OAP-AU Section 13.7] while nobody was told urgently. The Project Lead accepted the recommendation to make them critical on such endpoints: Section 8.4.3 does so by a rule of its own rather than an equipment group, and they are protected alarms (Section 7.3.5).