



OPEN AQUARIUM PROJECT SPECIFICATION

Water Level and Water Change

PRE-DRAFT

Revision	0.3
Status	Pre-Draft. A candidate document submitted for consideration.
Date	2026-10-18
Identifier	water-level-and-water-change
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 `water-level-and-water-change` 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 checks file, JSON Schema, 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 checks file, schema, and examples under the Apache License 2.0. Every safety check it defines, and every requirement, has a published test procedure (Section 15.5), so owners, makers, and reviewers can verify a product’s claims for themselves. The configuration format is plain JSON with a published schema and validator.

2.2 Interoperability

The top-off logic is defined once, and runs the same way on a top-off unit and on a controller (Section 8.14), so a top-off unit from any maker works with any controller. Top-off, water change, and RO/DI functions are function types of the Automation specification, so they move between controllers from different makers with the rest of an owner’s configuration. They use Device Classes endpoints, Data Model equipment, quantities, and events, Fail-Safe Behavior terms, and Alarms and Notifications condition codes, and add no parallel vocabulary.

2.3 Local Control

Every function runs on the controller or on the top-off unit itself, with no network service, account, or vendor. Daily limits use elapsed time, so they need no network time (Section 7.7). Owners can export their configuration and the functions’ saved state, such as a water change in progress, with the rest of their automation.

2.4 Safety

This Specification exists to meet this tenet: failures of top-off, water change, and RO/DI equipment flood rooms and kill livestock. Every run that adds or removes water has a time limit and a backup sensor, and ends by itself if the controller fails (Sections 7.3 and 7.5). A level sensor that is lost, broken, or stale never lets a top-off, a water change, or an RO/DI system keep adding or removing water (Section 7.4). A leak locks out the equipment that adds water until a person has checked, whatever the configuration says, and stops the equipment that moves it (Section 11). A controller warns its owners for as long as a top-off or a system's leak response has no path that uses no radio link (Sections 8.17 and 11.1). Heaters stop when the water is too low to cover them (Section 12). A water change interrupted by a power cut never drains further when power returns (Section 9.10), keeps its equipment stopped until the system is refilled even when its mode ends early (Section 9.4), and the alarms it holds sound as soon as it ends (Section 9.13). A top-off that carries an additive, such as kalkwasser, is limited by volume, stopped while the pH is high, and recommended to stop when it loses its controller, which alone enforces the additive's limits (Section 8.16).

2.5 Sustainability

Daily limits, leak responses, and the RO/DI run limits stop water from being wasted on the floor or down the drain. Membrane flushing and TDS monitoring help an RO/DI membrane last, and show when deionization resin is used up, rather than replacing it on a schedule (Section 10). Dry-run detection protects pumps from damage (Sections 8.11 and 12.4). Because the logic is the same for every maker, owners can keep a top-off unit or an RO/DI system when they change controllers.

Revision History

Revision	Date	Description	Authors
0.1	2026-10-14	Initial Pre-Draft.	Trent Scholl
0.2	2026-10-16	A top-off unit with <code>failSafeAction</code> <code>continue</code> keeps its last commanded <code>enabled</code> , so a hold lasts until its <code>revertAfter</code> falls due or a person ends it; a standing warning while every leak sensor of a system is on a radio link; the <code>low-water</code> alarms that a water change makes expected are held, never dropped, and sound as soon as it ends or aborts; additives carried in top-off water, such as kalkwasser, with a daily volume limit and a pH interlock; water changes by dosing heads send each dose once, in parts no longer than <code>maxContinuousRun</code> , watch for heads that dose unexpectedly, need calibrated heads, and need a level sensor for their fill; a drain stops and the water change aborts when its fill supply runs out; an aborted water change that drained no more than 1 % of the system's water does not hold the top-off; the smaller response to fresh-water leaks off until an owner chooses it; <code>low-water</code> kept critical, which an owner can lower to a warning but not turn off; dry-run detection on a monitored pump checks that the pump runs; <code>dry-run</code> cleared only by a person; top-off units use the attributes, faults, and level-switch positions of Device Classes revision 0.2; records use the event types of Data Model revision 0.2; a test procedure, mapped to its requirements, for every requirement; references cited by tag; one sentence per line; every question for the Project Lead decided.	Trent Scholl
0.3	2026-10-18	The Project Lead's decisions of 17 October 2026 applied: the paths without radio of top-off shutoff and leak response judged as Fail-Safe Behavior revision 0.4 judges every path, with its <code>failSafeTimeout</code> of at most 300 s (provisional), for each output they stop, with <code>safety-path-missing</code> while one has none; a top-off's additive limit raises a code of its own, <code>additive-limit</code> , warning by default and critical for no equipment group, and <code>daily-limit</code> stays for limits that point to a leak; a top-off unit that carries an additive recommended <code>failSafeAction</code> state with <code>enabled</code> <code>false</code> , with a warning for one configured to <code>continue</code> ; reservoir refills recorded with <code>record-maintenance</code> ; the function types <code>top-off</code> , <code>water-change</code> , and <code>ro-di</code> , with their saved state as <code>state.functions["<function id>"]</code> , in place of the provisional names, which controllers still read; the schema's <code>\$id</code> https://openaquariumproject.org/schemas/water-level-and-water-change/0.3 ; the new water's salinity and temperature checked before every water change that can remove more than 5 % of the system's water, or an unknown volume; a home app's stop aborts a water change. Also: a water change whose mode ends early keeps the mode's stops until it returns to idle or aborts, and a mode's <code>duration</code> sized with a pause in each phase; a phase counts its time while a lost	Trent Scholl

Revision	Date	Description	Authors
		output's state is unknown; no condition is expected while an interruption holds a water change, so a power cut sounds a low sump at once; a top-off unit refuses a manual fill while it is held, a controller sends none, and one that adds an additive needs dose-now; with a top-off unit, an additive's limit watched from volumeToday while a fill runs; which holds of a top-off unit outlast a failed controller, among them the stop of its interlock on a return pump that a feed mode pauses; a water change's dosing heads given the local time zone, and the time limits of the dosing example corrected; the checks of Section 7.3 reach equipment under local control; "A pass criterion that checks what a SHOULD asks applies to a product that does it"; a test for each new requirement; references to the final revisions of the other specifications; issues 14 to 20 deferred beyond revision 1.0.	

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	7
5	Terms, Definitions, and Conventions	8
5.1	Requirements Language	8
5.2	Terms and Definitions	8
5.3	Notation	9
6	Introduction	9
6.1	Purpose	9
6.2	Design Principles	9
6.3	Overview	10
6.4	Example	11
7	Common Requirements	11
7.1	Where the Logic Runs	11
7.2	Function Types	11
7.3	Safety Checks Act on Every Run	12
7.4	Level Inputs and Unknown Values	12
7.5	Runs End by Themselves	13
7.6	Latched Faults	13
7.7	Rolling Limits	14
7.8	Volumes and Flow Rates	14
7.9	Saved State	14
7.10	Faults and Alarms	15
7.11	Visibility	16
8	Top-Off	17
8.1	Overview	17
8.2	Members	17
8.3	Safety Checks	18
8.4	Starting and Ending a Fill	19
8.5	Maximum Fill Time	20
8.6	Minimum Fill Interval	20
8.7	Backup High Sensor	20
8.8	Unavailable Sensors	20
8.9	Daily Limit	20
8.10	Learned Fill Time	21
8.11	Dry-Run Detection	21
8.12	Reservoir Tracking and Forecasts	21
8.13	Water Changes, Leaks, and Modes	22
8.14	Top-Off Units	22
8.15	Choosing Values	23
8.16	Additives	24
8.17	Paths Without Radio	25
9	Automatic Water Changes	25
9.1	Overview	25
9.2	Members	26
9.3	Safety Checks	27
9.4	Starting	28
9.5	Checks Before Draining	29
9.6	Draining	29

9.7	Filling	30
9.8	Settling and Ending	30
9.9	Pauses	30
9.10	Interruptions	30
9.11	Aborting, Resuming, and Abandoning	31
9.12	Water Changes with Dosing Pumps	31
9.13	Water-Change Modes and Expected Conditions	32
9.14	Choosing Values	33
9.15	Records	33
10	RO/DI Systems	33
10.1	Overview	33
10.2	Members	34
10.3	Safety Checks	35
10.4	Making Water	36
10.5	Limits	36
10.6	Flushing	36
10.7	Rinsing and TDS	36
10.8	Valves and Pumps	36
10.9	Choosing Values	37
11	Leaks and Overflows	37
11.1	Leak Sensors	37
11.2	Leak Response	37
11.3	Response by Equipment Category	38
11.4	Water Type	39
11.5	Unavailable Leak Sensors	39
11.6	Overflow Sensors	39
11.7	Siphons and Drain-Down	40
11.8	Safety Checks	40
12	Low Water	40
12.1	Levels That Matter	40
12.2	Heaters	41
12.3	Pumps	41
12.4	Dry-Run Detection	41
12.5	Low-Water Alarms	42
12.6	Safety Checks	42
13	Fail-Safe Behavior	43
13.1	Outputs	43
13.2	Lost Inputs	43
13.3	Restarts and the Clock	44
13.4	Firmware Updates	44
13.5	A Failed Controller	44
14	Recommended Values	44
15	Conformance	46
15.1	Controllers	46
15.2	Top-Off Units	46
15.3	Configuration Documents	46
15.4	Schema and Tools	46
15.5	Test Procedures	46
16	Security Considerations	54
Appendix A — Approval Checklist		55
Appendix B — Examples (Informative)		55
Appendix C — Commercial Features (Informative)		56
Appendix D — Open Issues (Informative)		56

3 Scope

This Specification defines:

- the safety logic of top-off, whether it runs on a top-off unit or on a controller, with reservoir tracking and forecasts;
- additives carried in top-off water, such as kalkwasser;
- automatic water changes, by pumps, valves, or dosing pumps, as a sequence of phases that resumes safely after a power cut and stops on a leak;
- the control of RO/DI systems: feed, flush, and rinse valves, reservoir sensors, and TDS monitoring;
- the response to leaks and overflows, by equipment category;
- low-water protection of heaters and pumps;
- what each function does when an input, a device, power, or the controller is lost;
- the conditions these functions raise as alarms; and
- a test procedure for every safety check.

This Specification does not define:

- how a controller evaluates a configuration, or the elements it hosts these functions in, which the Automation specification defines;
- what devices do when they lose contact with their controller, which the Fail-Safe Behavior specification defines, beyond recommending configurations for the equipment of this Specification;
- how alarms are delivered, which the Alarms and Notifications specification defines;
- how new saltwater is mixed or heated before a water change;
- recommended water parameters, such as a salinity to aim for; or
- the electrical design of pumps, valves, and sensors, and the electrical safety of aquarium appliances, which standards such as IEC 60335-2-55 cover [IEC 60335-2-55].

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.
- **[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>.
- **[OAP-DM]** Open Aquarium Project, "Open Aquarium Data Model", Revision 0.4, [specs/data-model/spec.md](#).
- **[OAP-DC]** Open Aquarium Project, "Open Aquarium Device Classes", Revision 0.5, [specs/device-classes/spec.md](#).
- **[OAP-FS]** Open Aquarium Project, "Open Aquarium Fail-Safe Behavior", Revision 0.4, [specs/fail-safe-behavior/spec.md](#).
- **[OAP-AU]** Open Aquarium Project, "Open Aquarium Automation", Revision 0.3, [specs/automation/spec.md](#).
- **[OAP-AN]** Open Aquarium Project, "Open Aquarium Alarms and Notifications", Revision 0.3, [specs/alarms-and-notifications/spec.md](#).
- **[OAP-SA]** Open Aquarium Project, "Open Aquarium Security and Access", Revision 0.3, [specs/security-and-access/spec.md](#).
- **[OAP-DO]** Open Aquarium Project, "Open Aquarium Dosing", Revision 0.3, [specs/dosing/spec.md](#).

4.2 Informative References

- **[OAP-MB]** Open Aquarium Project, "Open Aquarium Module Bus", Revision 0.4, [specs/module-bus/spec.md](#).

- **[OAP-MM]** Open Aquarium Project, “Open Aquarium Matter Mapping”, Revision 0.3, [specs/matter-mapping/spec.md](#).
- **[P0001]** Open Aquarium Project, “Proposal 0001: Reference Controller Architecture”, [proposals/0001-reference-controller-architecture.md](#).
- **[Firmware]** Open Aquarium Project, “Breadboard controller firmware”, [hardware/controller/firmware/controller/](#). Its top-off runs on two float switches, with a maximum run time, a rest between runs, and `revertAfter`.
- **[IEC 60335-2-55]** IEC 60335-2-55, “Household and similar electrical appliances — Safety — Part 2-55: Particular requirements for electrical appliances for use with aquariums and garden ponds”. It covers the electrical safety of aquarium heaters and similar appliances, but not pumps, which IEC 60335-2-41 covers.

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, and controller, in the Fail-Safe Behavior specification [OAP-FS], such as fail-safe behavior, unavailable input, safety function, and standing warning, in the Automation specification [OAP-AU], such as output, input, condition, function, interlock, mode, override, layer, and saved state, in the Alarms and Notifications specification [OAP-AN], such as condition code, subject, and latching, and in the Dosing specification [OAP-DO], such as head, product, and effect, have the same meaning here.

Top-off: Adding fresh water to a system to replace water lost to evaporation, which leaves the salt behind, so that the level and the salinity stay steady.

Top-off unit: A device that runs top-off by itself, with its own sensors, and presents an `auto-top-off` endpoint [OAP-DC Section 10.1].

Fill: One run of a top-off, from the moment it starts adding water until it stops.

Normal level: The level a top-off keeps, detected by its normal sensor.

Additive: A substance carried in top-off water, so that the top-off adds it whenever it fills (Section 8.16).

Kalkwasser: A saturated solution of calcium hydroxide in fresh water, also called limewater, added to replace the calcium and alkalinity that corals take up.

Backup sensor: A second sensor that stops a run if the sensor that should have ended it fails: a backup high sensor above the normal level for adding water, and a backup low sensor for removing it.

Reservoir: A Data Model vessel with the role `reservoir` [OAP-DM Section 9.7], which holds water for a function, such as top-off water, new saltwater, or the product of an RO/DI system.

New water: The water a water change adds.

Water change: Replacing some of a system’s water with new water.

Phase: One step of a water change: checking, draining, filling, or settling (Section 9.1).

Expected condition: A condition that a water change expects in some of its phases, and whose alarm the controller raises, shows, and records as usual, holding only its sound and its notifications until the water change leaves those phases (Section 9.13) [OAP-AN Section 12.8].

RO/DI system: Equipment that purifies mains water by reverse osmosis (RO) and deionization (DI) for top-off and new water.

Product water: The purified water an RO/DI system makes.

Flush: Running water quickly across an RO membrane to the drain, to clean it.

Rinse: Sending product water to the drain at the start of a run, until its quality settles.

TDS: Total dissolved solids, the Data Model parameter `total-dissolved-solids`, used to judge the quality of product water.

Run: A period during which an output of a function is running: a fill, a drain, or the making of RO/DI water.

Supply equipment: Equipment that adds water to a system or a reservoir: top-off, RO/DI systems, and the fill of a water change.

Leak: Water where it should not be, detected by a `leak-sensor` endpoint.

Zone: The leak sensors that start one emergency mode, and the systems whose equipment that mode acts on (Section 11.2).

Cover level: The lowest water level at which a heater is fully under water, as its maker marks it.

Intake level: The lowest water level at which a pump's intake stays under water.

Drain-down: The water that flows from a display into its sump when the return pump stops, until the display's level falls to its overflow.

Latched fault: A fault after which a function starts no run until a person clears it (Section 7.6).

Held: An output is held when a higher layer of the Automation precedence [OAP-AU Section 17.3], or one of this Specification's checks acting as an interlock, decides its state instead of the function that drives it.

Interruption: A power cut, a restart of the controller, or a power-loss mode, during a run.

5.3 Notation

JSON member names, function types, condition codes, and the names of attributes, commands, and faults appear in `monospace`. Durations are in seconds. Units are Data Model unit codes [OAP-DM]. References to other Open Aquarium specifications name them, as in "Automation Section 13.2", or use their tags from Section 4, as in "[OAP-AU Section 13.2]". A section number with a third part refers to a numbered item of that section: "Section 8.4.2" means item 2 of Section 8.4. Numbers marked provisional are to be confirmed with the breadboard controller and owners' experience (Appendix D).

6 Introduction

6.1 Purpose

Water level equipment fails in quiet, ordinary ways. A float switch sticks, an optical sensor grows a film of algae, a siphon starts, a solenoid valve sticks open, or the power goes out halfway through a water change. A top-off pump that keeps running can empty its reservoir into the sump and onto the floor, and in a marine system the fresh water lowers the salinity on the way. An RO/DI system fed from the mains has no reservoir to run out of, and can flood a room. A heater left running in a drained sump can crack, or start a fire.

Four widely used commercial controller ecosystems all offer top-off with time limits and a backup sensor, and three of them offer automatic water changes, but each defines its own checks, and only some treat a broken sensor as safe or stop on a leak. This Specification defines the checks once, for every controller and top-off unit, and gives each a test procedure, so that owners can rely on them whatever equipment they combine.

6.2 Design Principles

1. **Every run is limited twice.** Every run that adds or removes water ends at its normal sensor, and also at a backup sensor and a time limit, each of which works without the others. Water changes by dosing heads, whose heads limit what each dose and each day can move, are the one exception, and their fills still stop at a level sensor (Section 9.12).
2. **Unknown means stop.** A sensor that is lost, broken, or stale stops the run it ends or limits. Section 7.4 lists the few exceptions, which keep a return pump running or let a water change refill the system.
3. **Safety checks act on every run.** The backup sensors and time limits stop an output whatever asks for it, including an owner's override (Section 7.3).

4. **Runs end by themselves.** Every command that starts water flowing carries `revertAfter`, or, for a dosing head, is a dose that ends by itself, so the run ends even if the controller fails (Section 7.5).
5. **A person checks before water flows again.** After a fault that suggests water is going somewhere it should not, a function stays stopped until a person clears it (Section 7.6).
6. **Never drain further after an interruption.** A water change interrupted by a power cut or a restart goes on to fill, so the system returns to its normal level (Section 9.10).
7. **The same logic everywhere.** Top-off follows the same rules on a top-off unit and on a controller (Section 8.14).

6.3 Overview

Figure 1 shows a typical reef system with a sump, and the equipment this Specification controls.

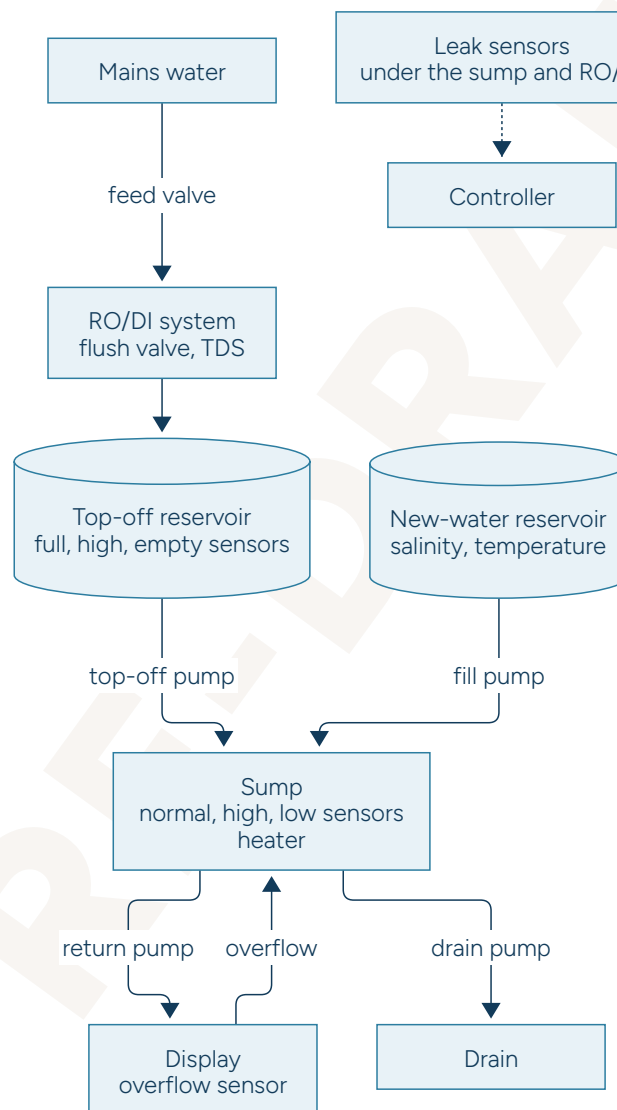


Figure 1 — Equipment this Specification controls in a typical reef system

Table 1 — Functions of this Specification

Function	What it does	Defined as	Section
Top-off	Replaces evaporated water from a reservoir, keeping the normal level, and may carry an additive.	A function type, or a top-off unit.	8
Water change	Replaces some of a system's water with new water, in phases.	A function type that follows a water-change mode.	9
RO/DI	Makes purified water into a reservoir.	A function type.	10

Function	What it does	Defined as	Section
Leak response	Stops equipment that moves water when a leak is detected.	Emergency modes.	11
Overflow response	Stops equipment that adds water to a vessel about to overflow.	Latching interlocks.	11.6
Low-water protection	Stops heaters and pumps when the water is too low to cover them.	Interlocks.	12

The top-off, water change, and RO/DI functions are function types that the Automation specification hosts [OAP-AU Section 11.10]. The leak, overflow, and low-water responses are built from Automation elements that already exist: emergency modes and interlocks. This Specification says how to configure them, and what a controller must check.

6.4 Example

A top-off's optical sensor grows a film of algae at 02:00, and from then on reports wet whether or not water covers it.

1. The sump level falls as water evaporates, but the sensor still reports wet. The top-off does not fill, and nothing else happens.
2. Over a day, the level falls 2 cm. The salinity rises a little, which the owner's range alarm on salinity may catch.
3. Had the sensor failed the other way, reporting dry while under water, the top-off would have filled. The fill stops when the backup high sensor becomes wet, 1 s later at most, and the top-off raises `high-level`, a critical alarm, and latches (Section 8.7).
4. Had the backup sensor failed too, the fill would have stopped at `maxFillTime`, 45 s after it started, with `timeout` (Section 8.5). The 45 s was chosen so that one fill cannot raise the level from the normal sensor to the sump's rim (Section 8.15).
5. Had the controller crashed during that fill, the pump would still have stopped at 47 s, because the controller sent the fill with `revertAfter` (Section 7.5).

In each case, the top-off stays stopped until the owner has cleaned the sensor and cleared the fault.

7 Common Requirements

7.1 Where the Logic Runs

1. A top-off runs either on a top-off unit, with the unit's own sensors, or on a controller, with level sensors and a pump or valve that the controller reads and commands. The same rules apply to both (Section 8.14).
2. Water changes and RO/DI systems run on a controller, as functions of its automation configuration.
3. A controller **MUST** run these functions in its control process, as Automation Section 17.1.3 requires, so that if the functions stop, the devices they command apply their fail-safe behavior.

7.2 Function Types

1. This Specification defines three function types that the Automation specification hosts [OAP-AU Section 11.10]: `top-off` (Section 8), `water-change` (Section 9), and `ro-di` (Section 10).
2. Each is a control function: each output it drives has no other function, and interlocks, modes, and overrides act on it as on any other output [OAP-AU Section 17.3], except where Section 7.3 says otherwise.
3. Every function of these types has the members in Table 2, and the members of its own type.

Table 2 — Members of every function of this Specification

Member	Type	Presence	Description
<code>id</code> , <code>type</code> , <code>name</code> , <code>notes</code> , <code>preset</code>			As for every function [OAP-AU Section 11.1].
<code>system</code>	identifier	REQUIRED	The Data Model system the function serves.

Member	Type	Presence	Description
vessel	identifier	OPTIONAL	The vessel whose level the function keeps or changes, or that an RO/DI system fills. It MUST belong to <code>system</code> .
outputs	array	REQUIRED	Every output the function drives.
extensions	object	OPTIONAL	Extensions, as in the Automation specification [OAP-AU Section 7.7].

4. `outputs` MUST list every output the function drives, including every output named by one of its roles, such as `drain.outputs`, and its roles together MUST name every output in `outputs`. A controller that does not support a function's type can then still find its outputs, and treat them as Automation Section 19.2 requires for an unsupported function.
5. These functions define their own behavior when an input is unavailable (Section 13.2), so they MUST NOT have `onUnavailable`.
6. A member described as a reference names an input or a condition of the configuration, which MUST be true or false. A reference MAY name an input that is true or false, such as a level switch's `wet`, as Automation Section 10.1 allows.
7. A configuration document that contains these functions is a JSON text [RFC 8259], as every Automation configuration document is [OAP-AU Section 7.1]. It MUST be valid against the Automation schema, and each function MUST be valid against its definition in `schema/water-level.schema.json` [JSON Schema].
8. Revision 0.2 named the function types `org.openaquariumproject.top-off`, `org.openaquariumproject.water-change`, and `org.openaquariumproject.ro-di`, and kept the saved state of every function of these types under `state.extensions["org.openaquariumproject.water-level"]`, keyed by function identifier. A controller reads them as the types and the saved state of this revision [OAP-AU Section 7.8], and writes those of this revision.

7.3 Safety Checks Act on Every Run

Some checks protect against water going where it should not, whatever asks for it to flow. A controller MUST apply the checks in this list at the precedence of interlocks [OAP-AU Section 17.3]: they stop the outputs they protect whatever any layer demands, including overrides and the emergency and power-loss modes, and they never start an output. Their stops reach an output under local control too, as every stop at that precedence does [OAP-AU Section 17.4.2].

1. A top-off's backup high sensor (Section 8.7), its `maxFillTime` (Section 8.5), its daily limit (Section 8.9), and its additive's daily limit (Section 8.16), on every run of its outputs.
2. A water change's backup low limit and `drain.maxTime` on every run of its drain outputs (Section 9.6), and its backup high limit and `fill.maxTime` on every run of its fill outputs (Section 9.7).
3. An RO/DI system's backup high sensor, its `maxRunTime`, and its daily limit, on every run of its feed outputs (Section 10.5).
4. The leak lockout, on every output of every top-off, water change, and RO/DI function (Section 11.2).

An owner who overrides a top-off pump to run, for example, gets a run that still stops at the backup high sensor and at `maxFillTime`. The controller shows these stops as it shows interlocks (Automation Section 17.10), naming the function and the check, such as "stopped by top-off 'Sump top-off': backup high sensor".

7.4 Level Inputs and Unknown Values

Every end condition and limit condition of this Specification, such as a top-off's `level`, a water change's `drain.limit`, or an RO/DI system's `full`, is true when its run must stop.

1. While such a condition is unknown, for example because its sensor reports `wiring-fault`, its device is lost, or its reading is older than its maximum age [OAP-AU Section 9.8], it counts as true: the run stops, and no run starts.
2. While a condition that starts or allows a run, such as an RO/DI system's `low` or a water change's `check.ready`, is unknown, no run starts.
3. A function raises `input-unavailable` for each input it uses that is unavailable [OAP-AU Section 17.9].

4. Numeric inputs of the optional quality checks, such as a top-off pump's power or a product water TDS, are exceptions: while one is unavailable, the function suspends that check, continues under its other checks, and raises `input-unavailable`.
5. Level switches SHOULD have the `supervised` feature [OAP-DC Section 11.2], so that broken wiring makes them unknown rather than wrong.
6. Each level switch's `position` SHOULD say what it detects [OAP-DC Section 11.2]: `normal` and `high` for a top-off's sensors, `drain` for a water change's drain sensor, `low` for a low-water sensor, `overflow` for an overflow sensor, and `reservoir-low`, `reservoir-high`, and `reservoir-empty` for a reservoir's sensors.

So a top-off whose normal sensor is lost treats it as "level reached", as a top-off unit does with a broken sensor.

These are the only exceptions to item 1, each because stopping would do more harm than the risk it avoids:

- A water change whose `leak` becomes unknown ends its drain and refills the system, and a fill continues, because the fill's own limits still apply (Sections 9.6 and 9.7).
- A top-off whose system's leak sensor is unknown continues, because its own limits bound what it can add (Section 11.5).
- The low-water and overflow interlocks on a return pump allow it to run while their sensors are unknown, because a return pump is vital equipment (Sections 11.6 and 12.3; Appendix D, issue 3).

7.5 Runs End by Themselves

1. When a controller commands an output of these functions into its running state, it MUST include `revertAfter` [OAP-DC Section 7.4], no longer than the time left until the run's time limit plus 2 s (provisional), so that the run ends even if the controller fails. The margin lets the controller, not the device, normally end the run. An `auto-top-off` output is the exception: the controller enables it without `revertAfter`, because the unit limits its own fills (Section 8.14).
2. A controller MUST send the command that starts a run only once, and MUST NOT send it again while the run continues. On a device of Device Classes revision 0.1, a revert returns the output to the state it had before the latest command, so a repeated command would make the revert return the output to running [OAP-DC Section 7.4.9]. On a device of revision 0.2 or later, a repeated command keeps the first revert's target, but moves the revert later, past the run's time limit [OAP-DC Section 7.4.6]. A run that pauses is stopped, and starts again with a new command.
3. When a run ends, for any reason, the controller MUST stop the output with a command without `revertAfter`, even when a mode or an override with an end time is what stops it, so that no later revert can start the run again. A top-off unit is held as Section 8.14.7 says instead. After a run, the function itself wants the output stopped, so the stop does not need to end when the mode or override does. This is an exception to Automation Section 17.5.1, which that specification allows [OAP-AU Section 11.10.8].
4. An output of a class whose running is an action rather than a state, such as a `doser`, is given commands that end by themselves: each `dose` delivers a volume no larger than the run has left, and runs no longer than the head's `maxContinuousRun` (Section 9.12).
5. Every output that a controller drives for these functions, other than a top-off unit, MUST stop on loss of contact (`failSafeAction` state, with its inactive state as `failSafeState`), and SHOULD follow the rest of Table 18. A controller MUST NOT run a function until its outputs are configured so, and MUST tell the owner what to change [OAP-FS Section 13.1]. Then an output whose controller is lost stops at its fail-safe timeout, even before its revert falls due.

The breadboard controller firmware already sends each top-off run this way [Firmware].

7.6 Latched Faults

1. Some faults latch, as each function's section says: the function stops at once, and starts no run until a person clears the fault.
2. Clearing a latched fault needs the `clear-latch` operation [OAP-SA Table 6]. Clearing does not by itself start a run: the function then runs as its inputs say.

3. A latched fault stays latched across restarts and power loss (Section 7.9). Several commercial top-offs clear their faults when power returns, so a fault can be forgotten after a power cut; this Specification does not allow that.
4. A controller **MUST** show, for each latched fault, the function, the fault, why it latched, and when. It **SHOULD** ask the person clearing it to confirm that the system has been checked.
5. While a fault of a function is latched, its condition code stays active in Alarms and Notifications terms, so a critical alarm about it does not close before someone clears it [OAP-AN Section 12.3].
6. The leak lockout (Section 11.2) is kept as a latched fault with the code `leak`. It raises no alarm of its own, because the leak sensor raises `leak` [OAP-AN Section 8.7].
7. Only a person clears a latched fault, as item 2 says. A restart, a recorded refill, or the end of the condition that latched it **MUST NOT** clear it. A controller **MAY** offer to clear a fault when a person does something that shows they have checked the system, such as recording a refill after `dry-run` (Section 8.11), but clearing it is still the person's own action.

7.7 Rolling Limits

1. Daily limits, such as a top-off's `maxDailyFillTime`, count the run time in the last 24 hours of elapsed time, not since midnight, so they work without a valid clock. A daily limit of volume, such as an additive's `maxDailyVolume` (Section 8.16), counts the volume of the runs of the last 24 hours in the same way.
2. A controller keeps the end time and duration of each run of the last 24 hours in its saved state (Section 7.9), and its volume, where it is known and a limit of volume needs it.
3. After a restart, runs before the restart still count. While the clock is not valid, the controller counts the time it was switched off as zero, which can only make a limit stricter.

7.8 Volumes and Flow Rates

1. Volumes and flow rates in a configuration are Data Model Quantities [OAP-DM Section 7.5]: a `value` and a `unit`. A volume's unit **MUST** be a volume unit, such as L, mL, gal, or gal-imp. A flow rate's unit **MUST** be a unit of the `flow-rate` parameter, such as L/h, L/min, mL/min, or gal/h.
2. A function knows a volume it moves in one of three ways:
 - measured, from each `doseCompleted` event of a `doser` endpoint [OAP-DC Section 9.7], or the `volume` of each `fillCompleted` event of a top-off unit that measures or estimates its fills [OAP-DC Section 10.1];
 - estimated, from the time an output ran and its `flowRate`; or
 - unknown, when it has neither.
3. Volumes written to devices are in the devices' units, such as mL, converted with the Data Model registry.
4. A controller **SHOULD** show which volumes are estimates.

7.9 Saved State

1. A controller **MUST** keep the state of each of these functions across restarts, in its saved state [OAP-AU Section 17.8], as `state.functions["<function id>"]` [OAP-AU Section 11.10].
2. Its form is defined by `$defs/savedState` in `schema/water-level.schema.json`, and Table 3 lists its members.
3. A controller **MUST** save it whenever it changes, and, while a water change drains or fills, at least every 10 s (provisional) and after each completed dose.
4. It is exported and imported with the configuration (Automation Section 19), so that, for example, a water change waiting for a person continues to wait on a new controller.

Table 3 — Saved state of each function

Member	Functions	Contents
<code>latched</code>	all	The codes of the function's latched faults, including <code>leak</code> for the leak lockout.
<code>runs</code>	top-off, RO/DI	Each run that ended in the last 24 hours, and, for a top-off with <code>learning</code> , its last <code>learning.samples</code> normal fills: <code>end</code> (timestamp), <code>duration</code> (s), <code>volume</code> (the volume of the run, where known), and, for a top-off, <code>normal</code> (whether the fill ended at the normal level).
<code>reservoir</code>	top-off	<code>remaining</code> , the estimated volume left, and <code>refilled</code> , when the reservoir was last refilled.
<code>phase</code>	water change	<code>checking</code> , <code>draining</code> , <code>filling</code> , <code>settling</code> , or <code>aborted</code> . Absent while idle.

Member	Functions	Contents
phaseTime	water change	How long, in seconds, the current phase has run, not counting pauses.
started	water change	When the water change started.
drained, filled	water change	The volumes drained and filled so far, where known.
dose	water change	The dose of a dosing head that the controller has sent and not yet counted (Section 9.12): output , the head; volume , the dose; and before , the head's dosedToday when the controller sent it.
calibrated	water change	When each dosing head of the water change was last calibrated, keyed by output identifier (Section 9.12).

7.10 Faults and Alarms

1. A controller raises the faults of these functions with the function as their subject, as it does for Automation faults [OAP-AN Section 8.2].
2. A top-off on a controller raises the same fault codes as a top-off unit [OAP-DC Section 10.1], so an owner gets the same alarm whichever runs it.
3. The equipment categories of a function are those of its outputs [OAP-AN Section 7.2], so a fault such as **timeout**, which is critical for hazardous equipment, is critical for a top-off, a water change, or an RO/DI system.
4. Table 4 lists the codes each function raises. Table 5 gives the codes this Specification adds, which are listed with their default severities in **conditions.json**.

Table 4 — Codes the functions raise

Code	Defined by	Raised by	When	Latches
timeout	Device Classes	Top-off, water change, RO/DI	A run reached its time limit (Sections 8.5, 8.10, 9.6, 9.7, 10.5).	Top-off and RO/DI: yes. Water change: through the abort, if it was filling.
high-level	Device Classes	Top-off, water change, RO/DI	A backup high sensor became wet (Sections 8.7, 9.7, 10.5).	Yes
dry-run	Device Classes	Top-off	The top-off pump runs dry (Section 8.11).	Yes
stuck-on	Device Classes	Water change	A dosing head of the water change doses when the controller sent it no dose (Section 9.12).	Yes
low, empty	Device Classes	Top-off, water change	The reservoir or the new-water supply is low or empty (Sections 8.12, 9.7).	No
waste-full	Device Classes	Water change	The waste container is full (Section 9.6).	No
input-unavailable	Automation	All	An input the function uses is unavailable (Section 7.4).	No
daily-limit	This Specification	Top-off, RO/DI	The run time of the last 24 hours reached its limit (Sections 8.9 and 10.5).	No
additive-limit	This Specification	Top-off	The volume of the last 24 hours of a top-off that carries an additive reached the additive's limit (Section 8.16).	No
drain-limit	This Specification	Water change	The drain reached its backup low limit (Section 9.6).	No
water-change-aborted	This Specification	Water change	A water change stopped after it began draining, before the system was refilled (Section 9.11).	Yes
water-change-skipped	This Specification	Water change	A water change did not start draining (Section 9.4).	No

Code	Defined by	Raised by	When	Latches
new-water-mismatch	This Specification	Water change	The new water differs from the system water (Section 9.5).	No
tds-high	This Specification	RO/DI	The product water's TDS is above the owner's limit (Section 10.7).	No
low-water	This Specification	Controller	A low-level sensor is dry (Section 12.5).	No

Table 5 — Conditions this Specification adds

Code	Meaning	Default	Critical for	Notes
low-water	A level-switch endpoint whose position is low reports wet false: the water is too low to cover heaters or pump intakes (Section 12.5).	critical	—	Raised after 30 s. While an automatic water change of its system drains, fills, or settles, it is raised and shown, but its sound and notifications are held until the water change ends or aborts, or an interruption, such as a power-loss mode, holds it. Low-water interlocks stop heaters, and usually the return pump.
drain-limit	An automatic water change drained to its backup low limit before its normal end (Section 9.6).	warning	—	The sensor, volume, or time that should have ended the drain did not. The water change goes on to fill.
water-change-aborted	An automatic water change stopped after it began draining, before the system was refilled (Section 9.11).	critical	—	The system may be short of water, and the top-off stays stopped unless the water change drained no more than 1 % of the system's water, until a person resumes or abandons the water change.
water-change-skipped	An automatic water change did not start draining, because a check failed or something prevented it (Section 9.4).	warning	—	No water was moved.
new-water-mismatch	The new water's salinity or temperature differs from the system water's by more than the owner allows (Section 9.5).	warning	—	The water change is skipped before it drains.
daily-limit	A function reached its limit of run time in 24 hours, the most a top-off or an RO/DI system may run, or a top-off unit reports the Device Classes fault daily-limit (Sections 8.9 and 10.5).	warning	hazardous	Water may be leaving the system somewhere, and the top-off replacing it. A top-off's limit on top-off water that carries an additive raises additive-limit instead.
additive-limit	A top-off that carries an additive, such as kalkwasser, added in the last 24 hours the most top-off water that the additive's limit allows, maxDailyVolume (Section 8.16).	warning	—	The top-off fills again as older fills leave the 24-hour window. In a dry season it usually means that the system evaporates more water than the limit allows, so its salinity rises meanwhile: dilute the additive, or check the limit. It is not critical for any equipment group, because the limit points to a strong additive, not to a leak.
tds-high	The product water of an RO/DI system has a total dissolved solids above the owner's limit (Section 10.7).	warning	—	The deionization resin, or the membrane, usually needs replacing.

7.11 Visibility

A controller MUST show the owner, for each function:

1. what it is doing, such as “filling, 12 s”, “waiting: minimum interval, 9 min left”, or “held: water change ‘Weekly’”;
2. its latched faults, and why they latched (Section 7.6);
3. for a top-off, its last fills, its fill time in the last 24 hours against its daily limit, its learned fill time, its reservoir estimate and forecast, and, with an additive, the volume of the last 24 hours against the additive’s daily limit (Section 8.16);
4. for a water change, its phase, the time the phase has left, the volumes drained and filled, and the alarms it holds (Section 9.13); and
5. for an RO/DI system, whether it is making water, flushing, or rinsing, and its latest TDS.

8 Top-Off

8.1 Overview

A top-off adds fresh water from a reservoir whenever evaporation lowers the level below its normal sensor, and stops when the level returns. Every fill is also limited by a backup high sensor, a maximum fill time, and a minimum interval between fills, and may be limited by a daily limit, a learned fill time, and dry-run detection.

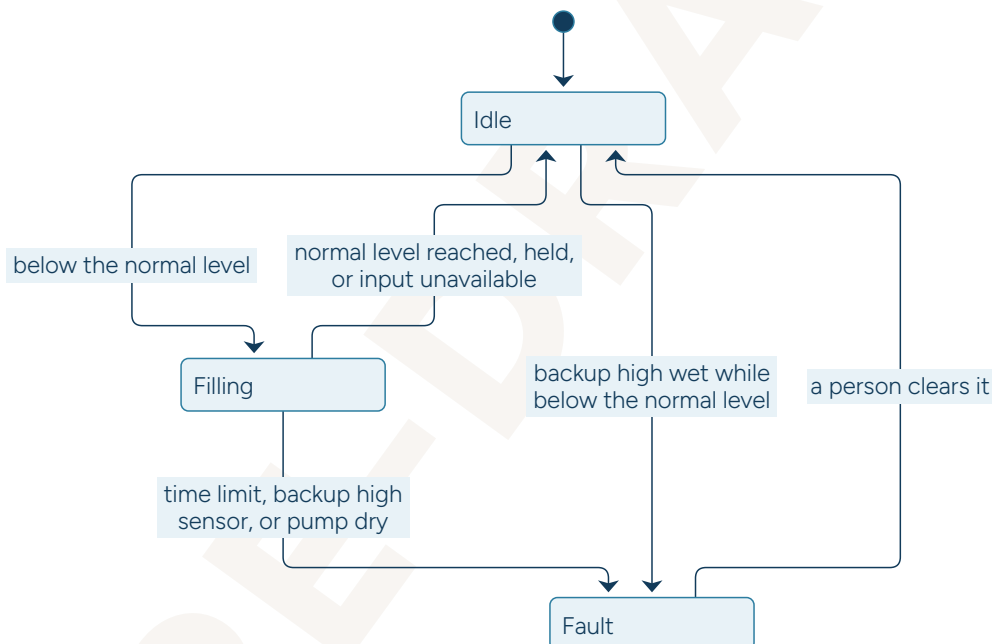


Figure 2 — Top-off states

The states are those of the `auto-top-off` class: `idle`, `filling`, `disabled` while it is held, and `fault` while a fault is latched [OAP-DC Section 10.1].

8.2 Members

A top-off function has the members of Table 2 and those of Table 6.

Table 6 — Members of a top-off function

Member	Type	Presence	Description
<code>outputs</code>	array	REQUIRED	The pump, valve, or switch that adds water, or one <code>auto-top-off</code> output (Section 8.14).
<code>level</code>	reference	REQUIRED, except with a top-off unit	True when the water is at the normal level.
<code>high</code>	reference	REQUIRED, except with a top-off unit	True when the water is at the backup high sensor.

Member	Type	Presence	Description
<code>maxFillTime</code>	number (s)	REQUIRED	The longest a fill may run (Section 8.5).
<code>minFillInterval</code>	number (s)	REQUIRED	The shortest time from the end of one fill to the start of the next (Section 8.6).
<code>startDelay</code>	number (s)	OPTIONAL	How long the level must stay below the normal level before a fill starts. Default 10 (provisional). With a top-off unit, written to the unit (Section 8.14).
<code>overrun</code>	number (s)	OPTIONAL	How long a fill continues after the normal level is reached (Section 8.4). Default 0. With a top-off unit, written to the unit.
<code>maxDailyFillTime</code>	number (s)	OPTIONAL	The most fill time in any 24 hours (Section 8.9). RECOMMENDED.
<code>learning</code>	object	OPTIONAL	Learned fill time (Section 8.10): <code>factor</code> (default 3) and <code>samples</code> (default 10). With a top-off unit, written to the unit.
<code>dryRun</code>	object	OPTIONAL	Dry-run detection (Section 8.11): <code>input</code> , <code>below</code> , and <code>delay</code> (default 5 s). Not with a top-off unit.
<code>flowRate</code>	Quantity	OPTIONAL	The flow rate of the pump or valve, measured by the owner, for estimating volumes. Not with a top-off unit.
<code>reservoir</code>	object	OPTIONAL	Reservoir tracking (Section 8.12): <code>vessel</code> , <code>capacity</code> , <code>empty</code> , <code>lowVolume</code> , <code>lowDays</code> , and <code>stopWhenEmpty</code> .
<code>additive</code>	object	OPTIONAL	What the top-off water carries, such as kalkwasser (Section 8.16): <code>product</code> and <code>maxDailyVolume</code> .

For example:

```
{ "id": "ato", "type": "top-off", "system": "reef", "vessel": "sump",
  "outputs": ["ato-pump"], "level": "sump-normal-wet", "high": "sump-high-wet",
  "maxFillTime": 45, "minFillInterval": 900, "maxDailyFillTime": 600,
  "learning": {}, "flowRate": { "value": 300, "unit": "L/h" },
  "reservoir": { "vessel": "ato-reservoir", "capacity": { "value": 25, "unit": "L" }, "empty": "ato-reservoir-empty" } }
```

8.3 Safety Checks

Table 7 — Top-off safety checks

Check	Presence	Applies to	What it requires	Section
Maximum fill time	REQUIRED	Controllers, top-off units	Every fill stops when it has run for <code>maxFillTime</code> . The top-off raises <code>timeout</code> and starts no fill until a person clears it.	8.5
Minimum fill interval	REQUIRED	Controllers, top-off units	No fill starts sooner than <code>minFillInterval</code> after the previous fill ended, or after the top-off started.	8.6
Backup high sensor	REQUIRED	Controllers, top-off units	A second sensor above the normal level stops every run of the top-off at once and prevents fills while it is wet, at the precedence of interlocks. If it becomes wet during a fill or shortly after one, or while the normal sensor is dry, the top-off raises <code>high-level</code> and starts no fill until a person clears it.	8.7
Unavailable sensor means water	REQUIRED	Controllers, top-off units	A normal or high sensor that is unavailable, broken, or stale counts as wet: the top-off stops and starts no fill until the sensor is available again.	8.8
Fills end by themselves	REQUIRED	Controllers	Each fill is started by one command, carrying <code>revertAfter</code> no longer than the time left until <code>maxFillTime</code> plus 2 s, and is ended by a stop without <code>revertAfter</code> , so that a fill ends even if the controller fails, and no revert starts it again. The controller runs a top-off only on outputs that stop on loss of contact.	8.5
No fill during a water change	REQUIRED	Controllers	No fill starts during a water change of the same system, from its start to the end of its settle phase, or while an aborted water change of that system waits for a person, unless it drained no more than 1 % of the system's water.	8.13
Leak lockout	REQUIRED	Controllers, top-off units	A leak stops the top-off at once, and it stays stopped until a person has checked the system: on a controller, until a person clears the leak lockout, whatever the configuration;	8.13

Check	Presence	Applies to	What it requires	Section
			on a unit with its own leak sensor, until <code>resetFault</code> , even without a controller.	
Wait for the return pump	RECOMMENDED	Configurations	In a system with a sump, the top-off waits until the return pump has run for a while, through an Automation <code>requires</code> interlock with a <code>delay</code> , because the sump is not at its normal level while the return pump is stopped or restarting.	8.13
Daily limit	RECOMMENDED	Controllers, top-off units	The fill time in any 24 hours is limited to <code>maxDailyFillTime</code> . When the limit is reached, the top-off stops, raises <code>daily-limit</code> , and fills again only when the fills of the last 24 hours allow.	8.9
Learned fill time	OPTIONAL	Controllers, top-off units	Once enough normal fills have been seen, a fill that lasts longer than <code>factor</code> times the usual fill stops and raises <code>timeout</code> , which stays until a person clears it.	8.10
Dry-run detection	OPTIONAL	Controllers, top-off units	A top-off pump that draws less than its dry-run threshold for <code>delay</code> while it runs is running dry. It stops, raises <code>dry-run</code> , and starts no fill until a person clears it; recording a refill does not.	8.11
Reservoir tracking	OPTIONAL	Controllers, top-off units	The top-off estimates the water left in its reservoir, forecasts when it will run out, raises <code>low</code> and <code>empty</code> , and starts no fill while the reservoir's empty sensor is dry.	8.12
Holds of a top-off unit	REQUIRED	Controllers	A controller disables a top-off unit without <code>revertAfter</code> for a water change, an aborted water change, or the leak lockout, and with <code>revertAfter</code> for a hold whose end is known, such as a feed mode's or that of the top-off's interlock on a paused return pump, so that a unit configured to <code>continue</code> stays disabled when its controller fails during a water change, and fills again after a feed mode. It sends no manual fill to a unit it holds, and a unit refuses one while it is disabled.	8.14
Additive limits	RECOMMENDED	Controllers, configurations	A top-off that carries an additive, such as kalkwasser, stops when the volume of its fills in the last 24 hours reaches the additive's <code>maxDailyVolume</code> , raises <code>additive-limit</code> , and is held stopped while the system's pH is above its limit or unknown. A controller does not run a top-off whose fill volumes it cannot know against such a limit. A top-off unit that carries an additive stops filling when it loses its controller, and a controller warns about one configured to <code>continue</code> .	8.16
Paths without radio	REQUIRED	Controllers	A top-off on a controller has a path that uses no radio link while its output is wired to the controller, or stops by itself on loss of contact within a <code>failSafeTimeout</code> of at most 300 s. While it has none, the controller shows its owners a standing warning, raises <code>safety-path-missing</code> , and still runs the configuration. A top-off unit has a path through its own backup high sensor.	8.17

8.4 Starting and Ending a Fill

A top-off starts a fill when all of these hold:

1. `level` has been false, without interruption, for `startDelay`;
2. `high` is false;
3. at least `minFillInterval` has passed since the end of the previous fill (Section 8.6);
4. no fault of the top-off is latched;
5. its daily limits allow it (Sections 8.9 and 8.16);
6. its reservoir is not empty (Section 8.12); and
7. nothing holds its outputs: no mode, interlock, or override, and no water change of its system (Section 8.13).

A fill ends when the first of these happens:

1. `level` becomes true, or, with `overrun`, has been true for `overrun` seconds: a normal fill;
2. `high` becomes true (Section 8.7);
3. the fill reaches `maxFillTime` (Section 8.5) or its learned limit (Section 8.10);

4. a daily limit is reached (Sections 8.9 and 8.16);
5. the pump runs dry, or the reservoir becomes empty (Sections 8.11 and 8.12);
6. an input it needs becomes unavailable (Section 8.8); or
7. something holds its outputs, such as a feed mode, a leak's emergency mode, or a water change.

An overrun lets each fill add a little more water, so that a top-off fills less often; one common top-off kit runs on for 4 s. An overrun ends at once if `level` becomes unknown, and every other end still applies during it. Only a fill that ends at the normal level is normal. Every fill counts towards the minimum interval and the daily limit. A top-off SHOULD record each fill as described in Section 9.15.

8.5 Maximum Fill Time

1. Every fill MUST stop when it has run for `maxFillTime`. The top-off then raises `timeout`, which latches.
2. `maxFillTime` applies to every run of the top-off's outputs, whatever layer asks for it (Section 7.3), and each command that runs them carries `revertAfter` (Section 7.5).
3. `maxFillTime` MUST be greater than 0. Section 8.15 explains how to choose it.

A fill that takes too long is usually sending water somewhere other than the sump: down an overflow that has started to siphon, through a leak, or into a vessel the normal sensor is not in.

8.6 Minimum Fill Interval

1. A top-off MUST NOT start a fill sooner than `minFillInterval` after the previous fill ended, however it ended.
2. After the top-off starts, for example after a restart, the interval counts from the start, as if a fill had just ended.
3. `minFillInterval` MUST be greater than 0.

The interval limits how often a top-off can fill when a sensor flickers, and how fast it can add water when something has gone wrong: at most `maxFillTime` of flow in every `minFillInterval`.

8.7 Backup High Sensor

1. A top-off on a controller MUST have a backup high sensor, `high`, separate from its normal sensor, `level`. It SHOULD be mounted a little above the normal level, and below the point at which the vessel would overflow.
2. While `high` is true, the top-off MUST NOT start a fill, and MUST stop at once any run of its outputs, whatever layer asks for it (Section 7.3).
3. The top-off raises `high-level`, which latches, when `high` becomes true:
 - during a fill, or within 60 s (provisional) after a fill ended; or
 - while `level` is false, because the two sensors then disagree.
4. Otherwise, `high` becoming true raises nothing. In a sump, for example, drain-down wets both sensors when the return pump stops, and the top-off simply does not fill. One common top-off kit instead raises its alarm only once its high sensor has stayed wet for 10 minutes, for the same reason.

8.8 Unavailable Sensors

1. While `level` or `high` is unavailable, it counts as true (Section 7.4): the top-off stops any fill, starts no fill, and raises `input-unavailable`.
2. When the input is available again, the top-off resumes without a person, because no water was added while it was unknown.
3. A top-off unit does the same with its own sensors, raising `wiring-fault` or `sensor-disconnected` on the sensor's endpoint.

8.9 Daily Limit

1. A top-off SHOULD have `maxDailyFillTime`, which MUST NOT be less than `maxFillTime`.
2. When the fill time of the last 24 hours (Section 7.7) reaches `maxDailyFillTime`, the top-off stops the fill at once, raises `daily-limit`, and starts no fill until the fills of the last 24 hours leave room for one.

3. `daily-limit` does not latch: the top-off fills again as older fills leave the 24-hour window, so it keeps up with evaporation at no more than its daily limit, while the owner investigates.

A daily limit catches what the per-fill checks cannot: many fills that each end normally, because water is leaving the system somewhere and the top-off keeps replacing it.

8.10 Learned Fill Time

1. A top-off with `learning` keeps the durations of its last `samples` normal fills, and takes their median as its learned fill time.
2. Once it has at least half of `samples` normal fills, rounded up, it limits each fill to `factor` times the learned fill time, or `maxFillTime` if that is shorter. The default `factor`, 3 (provisional), is what one common top-off kit uses, after 5 fills. A fill that reaches the learned limit stops, and raises `timeout`, which latches.
3. `factor` MUST be greater than 1, and `samples` at least 2.
4. Fills that are not normal do not change the learned fill time.
5. A controller MUST show the learned fill time, and SHOULD let the owner restart learning, for example after changing the pump.

8.11 Dry-Run Detection

A top-off pump that runs dry draws less power than usual, and can be damaged.

1. `dryRun.input` names an Automation output input [OAP-AU Section 9.3] that reads the power or current of the top-off's output, such as a `switch` with the `power-monitoring` feature or a `pump` with the `telemetry` feature.
2. While a fill runs, and once the output reports running, a reading below `dryRun.below`, in the input's unit, for `dryRun.delay` without interruption means the pump is running dry. The top-off stops, and raises `dry-run`, which latches.
3. Only a person clears `dry-run`, with `clear-latch` on a controller or `resetFault` on a top-off unit, as every latched fault is cleared (Section 7.6). Recording a reservoir refill does not clear it, because a refill does not show that anyone has checked the pump, and a top-off unit cannot tell a person's refill from one an RO/DI function records. A controller SHOULD offer to clear `dry-run` when a person records a refill of the top-off's reservoir.
4. A controller SHOULD propose a `below` of 60 % (provisional) of the median reading during the top-off's last normal fills.

8.12 Reservoir Tracking and Forecasts

1. With `reservoir`, a top-off estimates the volume left in its reservoir: `capacity` less the volume of every fill since the last refill (Section 7.8). The estimate needs `flowRate`, or a top-off unit that measures its own volumes.
2. It forecasts when the reservoir will be empty, from the average volume used per day over the last 7 days, or since the last refill if that is shorter.
3. It raises `low` while the estimate is below `lowVolume`, or the forecast is less than `lowDays` days away (default 2, provisional).
4. While `reservoir.empty` is true, or unknown (Section 7.4), the top-off starts no fill, stops any fill, and raises `empty.reservoir.empty` is usually a level switch a little above the pump's intake, whose `position` is `reservoir-empty`.
5. With `stopWhenEmpty` true, the top-off also treats an estimate at or below zero as empty. The default is `false`, because an estimate drifts, and a top-off that stops wrongly lets the salinity rise.
6. A person records a refill, to `capacity` or to a volume they give. A refill is also recorded when an RO/DI function that fills the same vessel ends a run because the reservoir is full (Section 10.4).
7. A reservoir level switch whose `position` is `reservoir-low` raises `reservoir-low` by itself [OAP-AN Section 8.7]. A top-off does not raise `low` for it as well.

8.13 Water Changes, Leaks, and Modes

1. **Water changes.** A top-off MUST NOT fill during a water change of its system, from the moment the water change starts until it ends, nor while an aborted water change of its system waits for a person (Section 9.11), unless the volume that water change drained is known and is no more than 1 % (provisional) of the system's water volume [OAP-DM Section 8.2] (Appendix D, issue 6). Otherwise it would replace drained saltwater with fresh water. Replacing 1 % of a marine system's water with fresh water lowers its salinity by 1 %, about 0.35 psu, less than the difference that the new-water check allows (Section 9.5.5), while a top-off stopped for hours lets the salinity rise as the water evaporates.
2. **Leaks.** A leak locks the top-off out until a person clears the lockout (Section 11.2), whatever the configuration says. A top-off unit with its own leak sensor also stops by itself (Section 8.14).
3. **The return pump.** In a system with a sump, the sump is not at its normal level while the return pump is stopped, or for some minutes after it restarts. A top-off SHOULD therefore require the return pump with an Automation `requires` interlock and a `delay` of 300 s (provisional), so that it neither fills during drain-down nor fills as the return pump empties the sump and then overfills when the display drains back.
4. **Other modes.** Feed, maintenance, and power-loss modes SHOULD stop the top-off, as the Automation presets recommend [OAP-AU Table 29].

8.14 Top-Off Units

A top-off unit runs Sections 8.4 to 8.12 by itself, with its own sensors and settings, whether or not a controller is connected.

1. A top-off unit MUST have a normal sensor and a backup high sensor, each presented as a `level-switch` endpoint whose `position` is `normal` and `high`. They SHOULD have the `supervised` feature.
2. It MUST implement `minFillInterval`, which Device Classes makes REQUIRED from its revision 0.2 [OAP-DC Section 10.1]; a unit of revision 0.1 MUST implement it too.
3. It MUST keep its latched faults across power loss, and clear them only on `resetFault`.
4. A top-off unit with a `leak-sensor` endpoint MUST stop filling at once when it detects a leak, report the fault `leak`, or, for a unit of Device Classes revision 0.1, `state fault`, and start no fill until `resetFault`, even without a controller.
5. It enforces its limits on manual fills (the `fill` command) as on any other, and refuses a manual fill while it is not enabled, so that a manual fill never gets around a hold [OAP-DC Section 10.1].

A controller uses a top-off unit through a top-off function whose only output is the unit's `auto-top-off` endpoint.

6. The function MUST NOT have `level`, `high`, `dryRun`, or `flowRate`, which belong to the unit, and its `reservoir` has only `vessel` and `capacity`. Its `startDelay`, `overrun`, `maxDailyFillTime`, and `learning` are written to the unit, where the unit implements them (item 8).
7. The function's demand is run, which enables the unit, except while something holds it (Section 8.13), when the controller disables the unit with `set` and `enabled false`. The controller MUST disable the unit for a water change, an aborted water change that holds the top-off (Section 8.13.1), or the leak lockout without `revertAfter`. A top-off unit SHOULD have `failSafeAction continue` (Table 18), unless its water carries an additive (Section 8.16.7), and such a unit keeps its last commanded `enabled` when it loses contact [OAP-FS Section 7.3; OAP-DC Section 8.3.2], so a failed controller leaves it disabled, rather than letting it refill a drained system with fresh water, until the controller returns and a person resumes or abandons the water change or clears the lockout, or a person enables the unit with its own controls, for as long as that local control lasts [OAP-FS Section 12.4]. This too is an exception to Automation Section 17.5.1 [OAP-AU Section 11.10.8]. A hold whose end is known is sent with `revertAfter` [OAP-AU Section 17.5], so the unit fills again when the revert falls due, whether or not the controller is there: a feed mode's, and the stop of the top-off's `requires` interlock on a return pump that a feed mode pauses, which ends once the interlock's `delay` has passed after the pause [OAP-AU Section 17.5.5]. A unit whose `powerOnBehavior` is `last` restores the target of such a revert after a power cut [OAP-FS Section 9.1], so it fills again after the power cut rather than staying held.
8. The controller MUST write `maxFillTime`, `minFillInterval`, and `reservoir.capacity` to the unit's attributes, and `startDelay`, `overrun`, `maxDailyFillTime`, `learning.factor`, and `learning.samples` where the function

has them and the unit implements them, as Table 8 maps them, within a minute of their differing. Where the unit does not implement one of them, the controller MUST tell the owner that the unit uses its own setting instead, except for `maxDailyFillTime`, which the controller then enforces itself (item 9).

9. For a unit that does not implement `maxDailyFillTime`, such as one of Device Classes revision 0.1, the controller enforces `maxDailyFillTime` itself, from the durations of the unit's `fillCompleted` events, by disabling the unit until the window allows a fill. The controller always enforces an additive's `maxDailyVolume` itself, from the volumes of the unit's `fillCompleted` events, in the same way (Section 8.16).
10. A controller MUST NOT send `fill` to a top-off unit while it disables the unit (item 7), or while the limits it enforces itself leave no room for a fill (item 9), and MUST tell the person who asked for it why. The unit would refuse it anyway (item 5), but a person should learn what holds the top-off, rather than see a manual fill fail.

Table 8 — Top-off settings on a controller and on a top-off unit

On a controller	On a top-off unit [OAP-DC Section 10.1]
<code>level, high</code>	The unit's <code>level-switch</code> endpoints with <code>position normal</code> and <code>high</code> .
<code>maxFillTime</code>	<code>maxFillTime</code> .
<code>minFillInterval</code>	<code>minFillInterval</code> .
<code>startDelay</code>	<code>startDelay</code> , from Device Classes revision 0.2; otherwise the unit's own.
<code>overrun</code>	<code>overrun</code> , from Device Classes revision 0.2; otherwise the unit's own.
<code>maxDailyFillTime</code>	<code>maxDailyFillTime</code> , from Device Classes revision 0.2, which the unit enforces, reporting <code>daily-limit</code> ; otherwise the controller enforces it (item 9).
<code>learning</code>	The <code>learning</code> feature, which reports <code>learnedFillTime</code> , with <code>learningFactor</code> and <code>learningSamples</code> from Device Classes revision 0.2.
<code>dryRun</code>	The <code>dry-run-detection</code> feature.
<code>reservoir.capacity</code>	<code>reservoirCapacity</code> , with the <code>reservoir</code> feature.
<code>additive.maxDailyVolume</code>	No attribute; the controller enforces it (item 9).
Reservoir estimate	<code>reservoirRemaining</code> .
The leak lockout of a unit's own leak sensor	The fault <code>leak</code> , from Device Classes revision 0.2.
A person clears a latched fault	<code>resetFault</code> .
A person records a refill	<code>refill</code> .
Fill records	<code>fillCompleted</code> events.
The function's state	<code>state</code> .

8.15 Choosing Values

`maxFillTime`, `minFillInterval`, and `maxDailyFillTime` depend on the system, so this Specification gives no defaults for them (Appendix D, issue 1). A controller MUST NOT run a top-off until the owner has set `maxFillTime` and `minFillInterval`, and SHOULD help the owner work out all three as follows, from the vessel's `dimensions` [OAP-DM Section 8.2] and the pump's measured flow rate, for the owner to confirm.

1. **`maxFillTime`.** One fill at full flow must not be able to raise the level from the normal sensor to the point where the vessel overflows: $\text{maxFillTime} \leq V / Q$, where V is the volume between the normal level and the overflow point, and Q is the pump's flow rate. It must also be longer than a normal fill, with room to spare.
2. **`minFillInterval`.** The top-off must be able to keep up with evaporation on the hottest, driest day: $\text{maxFillTime} \times Q \times (86\,400 / \text{minFillInterval})$ should be several times the most water the system loses in a day.
3. **`maxDailyFillTime`.** About twice the fill time of the system's heaviest day of evaporation.

For example, a sump's return section is 40 cm by 30 cm, its normal sensor is 5 cm below the rim, and the top-off pump moves 300 L/h, or 5 L/min. V is $40 \times 30 \times 5 \text{ cm}^3$, or 6 L, so a fill must stop within 72 s; a `maxFillTime` of 45 s leaves a margin. The system loses up to 6 L a day, which takes 72 s of pumping, so a `maxDailyFillTime` of 150 s is enough, and a `minFillInterval` of 900 s allows up to 96 fills a day. Owners report evaporation of about 0.5 to 2.5 % of a system's volume a day, and top-off pumps move from about 13 to 380 L/h.

4. **The reservoir.** Where it is practical, a top-off reservoir SHOULD hold no more than V, the volume between the normal level and the overflow point, so that even if every check fails, emptying the reservoir cannot overflow the vessel, as two top-off makers advise.

8.16 Additives

Some owners add a substance to their top-off water, most often kalkwasser, to replace the calcium and alkalinity that corals take up. The top-off then doses the additive whenever it fills, so a fault that adds too much top-off water also adds too much of the additive. Kalkwasser raises the pH of the water it enters, and too much of it can raise the pH of a whole system to harmful levels within hours.

1. A top-off whose water carries an additive SHOULD declare it with `additive` (Table 6), whose members are:
 - `product`, REQUIRED: the additive, as a Dosing product [OAP-DO Table 8]: its `name`, and optionally its `manufacturer`, `gtin`, and `effect` [OAP-DO Section 12.8]; and
 - `maxDailyVolume`, RECOMMENDED: the most top-off water, a Data Model Quantity in a volume unit, that the top-off may add in any 24 hours.
2. While the volume of the fills of the last 24 hours (Section 7.7) is at or above `maxDailyVolume`, the top-off MUST stop any fill, start no fill, and raise `additive-limit`, as Section 8.9 says for `maxDailyFillTime` and `daily-limit`; a fill MUST stop as soon as its volume, as far as the controller can know it while the fill runs, reaches what the limit leaves. `additive-limit` does not latch, so the top-off fills again as older fills leave the window. It is a code of its own, `warning` by default and critical for no equipment group (Table 5), because a top-off that reaches the additive's limit is usually keeping up with evaporation in a dry season, while `daily-limit`, which is critical for hazardous equipment, says that water may be leaving the system.
3. `maxDailyVolume` needs the volume of each fill (Section 7.8): `flowRate` on a controller, or a top-off unit whose `fillCompleted` events give their `volume`. A controller MUST NOT run a top-off that has `maxDailyVolume` but no way to know the volume of its fills, and MUST tell the owner what to set. With a top-off unit, the controller learns a fill's volume from its `fillCompleted` event when the fill ends, and, while it runs, from the unit's `volumeToday`, where the unit reports it [OAP-DC Section 10.1]: it disables the unit as soon as `volumeToday` shows that the fill has reached what the limit leaves. A unit that does not report `volumeToday` can add up to one fill more than the limit, which its `maxFillTime` bounds, so a controller SHOULD tell the owner so, and SHOULD propose a `maxDailyVolume` smaller by the volume of one fill.
4. A top-off with an additive SHOULD be held stopped by a `block` interlock [OAP-AU Section 13.2] while the system's pH is above a limit, with `onUnavailable block`, so that it adds no additive while the pH is high or unknown. For kalkwasser, a limit of pH 8.45 (provisional) is RECOMMENDED, the pH at which makers' examples stop dosing kalkwasser [OAP-DO Section 13.1].
5. A controller SHOULD propose `maxDailyVolume` from the additive's `effect` and the system's volume [OAP-DO Section 12.8]: the volume that changes the effect's parameter by no more than Dosing Table 13 allows in a day [OAP-DO Table 13], or, where that is smaller, twice the water the system loses on its heaviest day of evaporation.
6. A controller SHOULD warn the owner about a top-off with an additive that has no `maxDailyVolume`, whose `maxDailyVolume` is larger than item 5 proposes, or that no pH interlock holds.
7. Only the controller enforces an additive's limits, its `maxDailyVolume` and its pH interlock: a top-off unit knows nothing of them. A top-off unit whose water carries an additive SHOULD therefore have `failSafeAction` state, with `failSafeState` `{"enabled": false}`, so that it stops filling when it loses its controller, rather than `continue` (Table 18) [OAP-FS Table 3]. A controller MUST warn its owners about a top-off unit whose water carries an additive and whose `failSafeAction` is `continue`, for as long as it is so, saying that the unit would go on adding the additive without its limits if the controller failed.

For example, saturated kalkwasser at room temperature holds about 1.6 g/L of calcium hydroxide, which adds about 43 meq/L of alkalinity, about 120 dKH, and about 870 mg/L of calcium (provisional). In a system of 300 L, the 1.4 dKH a day of Dosing Table 13 is reached by about 3.5 L of it ($1.4 \text{ dKH} \times 300 \text{ L} \div 120 \text{ dKH}$), about 1.2 % of the system's volume, so a `maxDailyVolume` of 3.5 L is proposed. If the system loses more water than that in a day, the top-off stops at the limit and the salinity rises a little, while `additive-limit` tells the owner, who can dilute the kalkwasser.

```
{ "id": "ato", "type": "top-off", "system": "reef", "vessel": "sump",
  "outputs": [ "ato-pump" ], "level": "sump-normal-wet", "high": "sump-high-wet",
  "maxFillTime": 45, "minFillInterval": 900, "maxDailyFillTime": 150, "flowRate": { "value": 300, "unit": "L/h" },
  "additive": { "product": { "name": "Kalkwasser, saturated",
    "effect": { "parameter": "alkalinity", "value": 0.12, "unit": "dKH", "volume": { "value": 1, "unit":
    "L" } } },
    "maxDailyVolume": { "value": 3.5, "unit": "L" } } }
```

with a `block` interlock on the top-off pump whose `when` is a `compare` of the system's pH above 8.45, and whose `onUnavailable` is `block`.

8.17 Paths Without Radio

Top-off shutoff is a safety function: a top-off that keeps filling floods a room [OAP-AU Section 13.7.2]. Like every safety function, it needs a path that uses no radio link, a way to stop its output that still works when every radio link fails [OAP-FS Section 13.4].

1. A top-off on a top-off unit has such a path, through protection inside the unit: its own backup high sensor stops its fills by itself (Section 8.14.1) [OAP-FS Section 13.4.2].
2. A top-off on a controller stops while one of its level sensors is unavailable (Section 8.8), so it has such a path, whatever links its level sensors use, while its output's device is connected to the controller by wire, or stops the output by itself when it loses contact, with a `failSafeTimeout` of at most 300 s (provisional), as Fail-Safe Behavior Section 13.4.3 judges a path [OAP-FS Section 13.4.4]. It has none while its output is reached over a radio link and its `failSafeTimeout` is longer.
3. A controller MUST check each top-off for such a path after it starts, and whenever its configuration, a binding, or an endpoint's fail-safe configuration changes [OAP-FS Section 13.4.5].
4. While a top-off has no such path, a controller MUST show its owners the standing warning of Fail-Safe Behavior Section 13.4.6, naming the top-off and its output, and saying what would give it one, such as a `failSafeTimeout` of 300 s or less, and MUST raise `safety-path-missing` [OAP-AU Section 13.7.6]. It MUST NOT refuse the configuration because of it [OAP-FS Section 13.4.7].

For example, a top-off pump on an outlet reached over Wi-Fi, with a `failSafeTimeout` of 120 s, has a path, because the outlet stops the pump by itself if its link fails, and the top-off stops if a level sensor's link fails. The same outlet with a `failSafeTimeout` of 3600 s has none, and the controller warns until the timeout is 300 s or less.

9 Automatic Water Changes

9.1 Overview

A water change function replaces some of a system's water with new water. It checks that the new water is ready, drains, fills, and lets the system settle.

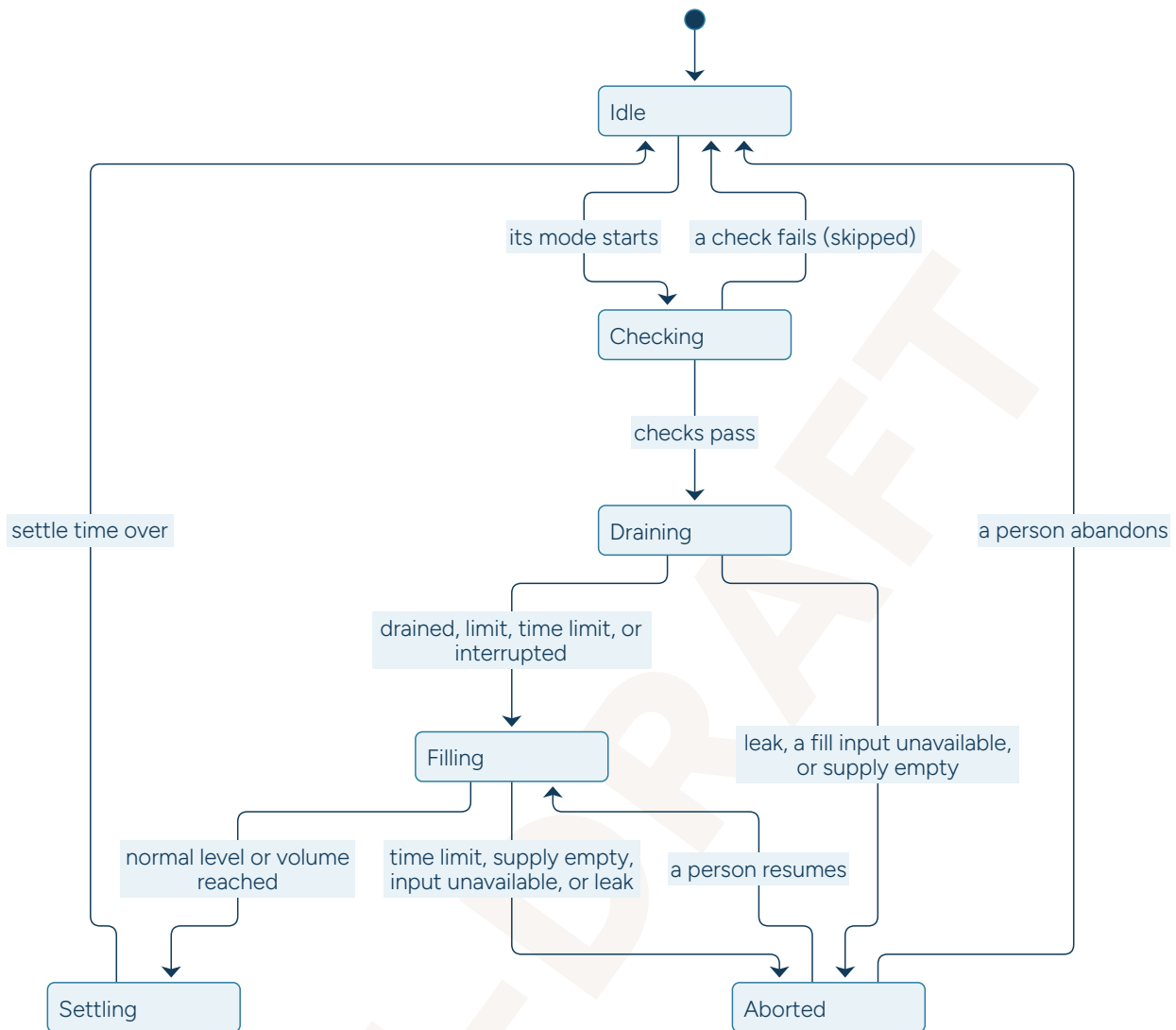


Figure 3 — Water change phases

A water change follows a mode of the kind `water-change` [OAP-AU Section 14]: the mode starting starts the water change, and the water change ends the mode when it finishes. While the mode is active, it holds the equipment the owner lists in it, such as the skimmer, at the mode's precedence.

9.2 Members

A water change function has the members of Table 2 and those of Table 9, and its `drain` and `fill` have the members of Table 10.

Table 9 — Members of a water change function

Member	Type	Presence	Description
<code>mode</code>	identifier	REQUIRED	The <code>water-change</code> mode the function follows (Section 9.4).
<code>drain</code>	object	REQUIRED	The drain phase (Section 9.6).
<code>fill</code>	object	REQUIRED	The fill phase (Section 9.7).
<code>settle</code>	number (s)	OPTIONAL	How long the system settles after the fill (Section 9.8). Default 600 (provisional).
<code>check</code>	object	OPTIONAL	Checks before draining (Section 9.5): <code>salinity</code> , <code>temperature</code> , <code>ready</code> , and <code>maxWait</code> .
<code>leak</code>	reference	OPTIONAL	True while a leak is detected in the water change's zone (Section 9.11).
<code>maxPause</code>	number (s)	OPTIONAL	The longest pause before a phase gives up (Section 9.9). Default 900 (provisional).

Table 10 — Members of drain and fill

Member	Type	Presence	Description
outputs	array	REQUIRED	The outputs that run during the phase.
until	reference	see below	True when the phase has done its work: for <code>drain</code> , when enough water has been drained; for <code>fill</code> , when the water is at the normal level.
volume	Quantity, or "drained" for <code>fill</code>	see below	The volume to move. "drained" means the volume the drain removed.
time	number (s)	OPTIONAL, <code>drain</code> only	Drain for this long.
limit	reference	REQUIRED, except for a phase of <code>doser</code> outputs	The backup sensor: for <code>drain</code> , true when the water is at the lowest level allowed; for <code>fill</code> , true when the water is at the backup high sensor.
maxTime	number (s)	REQUIRED	The longest the phase may run, not counting pauses.
flowRate	Quantity	OPTIONAL	The flow rate of the phase's outputs, measured by the owner, for estimating volumes.
direction	forward or reverse	OPTIONAL	The direction of <code>doser</code> outputs. Default <code>forward</code> .
wasteFull	reference	OPTIONAL, <code>drain</code> only	True when the waste container has no room.
supplyEmpty	reference	OPTIONAL, <code>fill</code> only	True when the new-water supply is empty.

1. `drain` MUST have at least one of `until`, `volume`, and `time`, and `fill` at least one of `until` and `volume`. The first to be reached ends the phase.
2. A `volume` needs a way to know the volume (Section 7.8): `doser` outputs, or `flowRate`. A `fill.volume` of "drained" needs the drain's volume to be known in the same way.
3. No output is in both `drain.outputs` and `fill.outputs`.
4. `drain.outputs` and `fill.outputs` are each of the classes `pump`, `switch`, `valve`, `level-output`, or `doser`.
5. A water-change mode followed by a water change MUST NOT list the function's own outputs.

For example, a water change that drains the sump to a level switch and refills it to the top-off's normal sensor:

```
{ "id": "wc", "type": "water-change", "system": "reef", "vessel": "sump", "mode": "wc-mode",
  "outputs": ["drain-pump", "fill-pump"],
  "drain": { "outputs": ["drain-pump"], "until": "sump-drained", "limit": "sump-low-dry", "maxTime": 300 },
  "fill": { "outputs": ["fill-pump"], "until": "sump-normal-wet", "limit": "sump-high-wet", "maxTime": 360,
    "supplyEmpty": "new-water-empty" },
  "check": { "salinity": { "system": "salinity", "newWater": "new-water-salinity", "maxDifference": 1.0 },
    "temperature": { "system": "temp", "newWater": "new-water-temp", "maxDifference": 2.0 } },
  "leak": "leak-any" }
```

9.3 Safety Checks

Table 11 — Water change safety checks

Check	Presence	Applies to	What it requires	Section
Phase time limits	REQUIRED	Controllers	Draining and filling each have a time limit that counts only the time the phase runs, and that also limits every run of the phase's outputs, whatever asks for it. A drain that reaches its limit stops, raises <code>timeout</code> , and goes on to fill. A fill that reaches its limit stops, raises <code>timeout</code> , and aborts.	9.6
Backup low limit	REQUIRED	Controllers	A backup low condition stops every run of the drain outputs at once, whatever the drain's normal end, at the precedence of interlocks. For a drain by dosing heads, it is recommended rather than required (Section 9.12).	9.6
Backup high limit	REQUIRED	Controllers	A backup high condition stops every run of the fill outputs at once, whatever the fill's normal end, at the precedence of interlocks, and raises <code>high-level</code> , which stays until a person clears it. For a fill by dosing heads, it is recommended rather than required (Section 9.12).	9.7
Stop on a leak	REQUIRED	Controllers	A leak stops every output of the water change at once and aborts it. It does not refill until a person resumes it.	9.11

Check	Presence	Applies to	What it requires	Section
Resume without draining more	REQUIRED	Controllers	After a power cut, a restart, or a power-loss mode during a water change, it never drains further: an interrupted drain goes on to fill, and an interrupted fill continues for the time it had left. A power-loss mode stops its outputs whether or not it lists them.	9.10
Unavailable inputs	REQUIRED	Controllers	An unavailable drain input ends the drain, and the water change goes on to fill. An unavailable fill input stops the fill, and the water change aborts.	9.6
Conditions to start	REQUIRED	Controllers	A water change does not start while an input it needs is unavailable, <code>leak</code> is true or unknown, its supply is empty, an emergency or power-loss mode is active, one of its faults is latched, or one of its dosing heads lacks its limits or a calibration. It is skipped instead.	9.4
Runs end by themselves	REQUIRED	Controllers	Each command that runs a drain or fill output carries <code>revertAfter</code> , no longer than the time the phase has left plus 2 s.	9.6
Pauses	REQUIRED	Controllers	While a higher layer holds an output the water change needs, or the output's device is lost, the water change pauses, counting the time while a lost output's state is unknown. A pause longer than <code>maxPause</code> ends a drain and aborts a fill. A stop from a home app aborts the water change.	9.9
Volume balance	REQUIRED	Controllers	A fill by volume adds the volume the drain removed, counted from each completed dose or metered run. When the fill cannot add it, the water change aborts.	9.12
Dosing heads	REQUIRED	Controllers	A water change by dosing heads sends each dose once, no larger than the head's <code>maxDoseVolume</code> and no longer than its <code>maxContinuousRun</code> , with a rest after it, and runs only heads that have their limits set and have been calibrated. A head that doses when the controller sent no dose is stopped, and <code>stuck-on</code> is raised, which aborts the water change and stays until a person clears it.	9.12
New water matches	REQUIRED	Controllers, configurations	Before draining, the new water's salinity and temperature are compared with the system water's. If either differs by more than allowed, or cannot be measured, the water change is skipped. Every water change that can remove more than 5 % of the system's water, or an unknown volume, has both checks.	9.5
Supply and waste ready	RECOMMENDED	Controllers	Before draining, every condition in <code>check.ready</code> is true, such as a new-water reservoir that holds enough water and a waste container with room. During the drain or the fill, an empty supply stops the water change, and it aborts.	9.5

9.4 Starting

- `mode` MUST name a mode of the kind `water-change`, and two water change functions MUST NOT name the same mode.
- When its mode starts, by a person, by its `start.when`, such as a schedule [OAP-AU Section 14.3], the water change starts in the checking phase, unless item 3 prevents it.
- The water change does not start, and is skipped, while:
 - an input it needs to drain and fill is unavailable;
 - `leak` is true or unknown;
 - `fill.supplyEmpty` is true;
 - an emergency or power-loss mode is active;
 - one of its faults is latched, or it waits for a person (Section 9.11);
 - one of its dosing heads lacks `maxDoseVolume` or `maxDailyVolume`, or has not been calibrated (Section 9.12); or
 - another water change of the same system runs.
- A skipped water change raises `water-change-skipped`, with the reason, and ends its mode.

5. When its mode ends before the water change finishes, by a person, by `end.when`, or at the end of its `duration`, the water change finishes now: from checking, it returns to idle; from draining, it goes on to fill; from filling, it continues to fill; and settling ends at once.
6. The mode's `duration` SHOULD be at least the sum of `check.maxWait`, `drain.maxTime`, `fill.maxTime`, and `settle`, and `maxPause` once for the drain and once for the fill, because pauses do not count towards a phase's time (Section 9.9.1).
7. Water-change modes that water changes of different systems follow may be active together, and starting a water-change mode ends any other that may not be active with it [OAP-AU Section 14.3.3]. So when the mode of one water change starts while another of the same system runs, the running one finishes now (item 5), and the new one is skipped (item 3).
8. When the mode ends early, while the water change goes on to fill or fills (item 5), the controller MUST keep each stop that the mode's `outputs` gave, at the mode's precedence, until the water change returns to idle or aborts, as if each had an `extend` that lasted until then [OAP-AU Section 14.4.3]. It sends them with `revertAfter` for the longest the water change may still take: the fill time it has left, and `maxPause`, and again whenever that changes [OAP-AU Section 17.5]. So a heater or a return pump that the mode stopped does not start again while the water is still below it.

9.5 Checks Before Draining

In the checking phase, the water change checks that it can finish before it removes any water.

1. Every input it needs to drain and fill MUST be available, every condition in `check.ready` MUST be true, and `fill.supplyEmpty` and `drain.wasteFull`, where given, MUST be false. `check.ready` lists conditions such as "the new-water reservoir holds enough water", "the new water is mixed", or "the waste container has room".
2. With `check.salinity`, the new water's salinity, `newWater`, and the system's, `system`, MUST differ by no more than `maxDifference`. Both inputs MUST measure the same parameter, such as `salinity`, `specific-gravity`, or `conductivity`, and `maxDifference` is in the unit of `system`.
3. With `check.temperature`, the same applies to the new water's temperature and the system's.
4. The water change waits up to `check.maxWait` (default 0) for the checks to pass, for example while the new water warms up, and is then skipped, with `new-water-mismatch` if a salinity or temperature check failed.
5. A controller SHOULD offer the salinity and temperature checks for every water change whose new water is measured, and SHOULD propose a `maxDifference` of 1.0 psu, or its equivalent, for salinity and 2.0 °C for temperature (provisional).
6. A water change MUST have both `check.salinity` and `check.temperature`, unless the most its drain can remove is known to be no more than 5 % (provisional) of the system's water volume [OAP-DM Section 8.2] (Appendix D, issue 2). The most a drain can remove is the smallest of its `volume`, and its `time` and its `maxTime` each times its `flowRate`, where it has them; a drain that ends only at `until` and has no `flowRate` can remove an unknown volume. A water change of `doser` outputs whose `drain.volume` is small, such as the many small water changes of Section 9.12.9, needs neither check.

New water mixed to the wrong salinity, or much colder than the system, is one of the few ways an automatic water change can harm livestock while every sensor works.

9.6 Draining

1. The drain runs its outputs: pumps, switches, valves, and level outputs in their running state, with `revertAfter` (Section 7.5), and `doser` outputs with `dose` commands in `direction` (Section 9.12).
2. The drain ends, and the water change goes on to fill, when the first of these happens:
 - `until` becomes true;
 - the volume drained reaches `volume`;
 - the drain has run for `time`;
 - `limit` becomes true, which raises `drain-limit`;
 - the drain has run for `maxTime`, which raises `timeout`;
 - `until` or `limit` becomes unavailable, which raises `input-unavailable` (Section 7.4);
 - `wasteFull` becomes true, which raises `waste-full`;
 - `leak` becomes unknown;

- its mode ends early (Section 9.4.5); or
 - it is interrupted (Section 9.10).
3. `limit` stops every run of the drain outputs at once, whatever asks for it (Section 7.3). It SHOULD be a sensor other than the one `until` uses, at the lowest level that keeps heaters and pump intakes under water (Section 12.1). A level switch that `until` reads SHOULD have the `position drain` (Section 7.4.6).
 4. The water change aborts instead (Section 9.11) when `leak` becomes true, an emergency mode acts on one of its outputs, a home app stops one of its outputs (Section 9.9.3), a fill input, `fill.until` or `fill.limit`, becomes unavailable, or `fill.supplyEmpty` becomes true or unknown, because it can then not refill safely. Aborting stops the drain at once, so no more water is drained than can be replaced.

9.7 Filling

1. The fill runs its outputs as the drain does, with `revertAfter`.
2. The fill ends, and the water change settles, when `until` becomes true or the volume filled reaches `volume`.
3. When `limit` becomes true, the fill stops at once and raises `high-level`, which latches, and the water change settles. `limit` stops every run of the fill outputs at once, whatever asks for it (Section 7.3). It is usually the top-off's backup high sensor.
4. The water change aborts (Section 9.11) when:
 - the fill has run for `maxTime`, which raises `timeout`;
 - `supplyEmpty` becomes true or unknown, which raises `empty`;
 - `until` or `limit` becomes unavailable;
 - `leak` becomes true, an emergency mode acts on one of its outputs, or a home app stops one of its outputs (Section 9.9.3); or
 - a pause lasts longer than `maxPause` (Section 9.9).
5. While `leak` is unknown, the fill continues, because it restores the system, and its own limits still apply.

9.8 Settling and Ending

1. After the fill, the water change settles for `settle` seconds, with its mode still active, so that the level steadies and the new water mixes before the top-off and the other equipment resume. One commercial controller waits 10 minutes.
2. It then returns to idle, ends its mode, and records the water change (Section 9.15).

9.9 Pauses

1. While something holds an output that the current phase runs, such as a feed mode, an override, or an interlock, the phase pauses. Its time limit does not count the pause.
2. A drain that has paused for longer than `maxPause` ends, and the water change goes on to fill; a fill that has paused for longer than `maxPause` aborts.
3. An emergency mode is not a pause: it aborts the water change (Section 9.11). A power-loss mode is an interruption (Section 9.10). Nor is a stop that a home app makes through a Matter bridge: from the start of the drain to the end of settling, it aborts the water change, so that only a person in an Open Aquarium app resumes or abandons it, and a home app's later `on` does not let the water change go on [OAP-MM Section 12.11].
4. An output that is under local control also pauses the phase. So does an output whose device is lost, once the controller takes the output to be in its fail-safe behavior, which has stopped it (Section 7.5.5) [OAP-AU Section 9.3.3]; until then, while the output's state is unknown, the phase's time limit counts that time as if the output ran, and the volumes estimated from its flow rate count it as Section 9.12.2 counts a dose whose volume cannot be known: a drain as having drained nothing, and a fill as having filled at its flow rate. So an output that ran on unseen is not run again for the whole time the phase had left.

9.10 Interruptions

A water change MUST NOT drain further after an interruption: a power cut, a restart of the controller, or a power-loss mode.

1. When a power-loss mode starts, the water change stops all its outputs at once, whether or not the mode lists them, and waits until the mode ends.
2. When the controller starts again, or the power-loss mode ends, a water change that was:
 - checking starts its checks again;
 - draining goes on to fill;
 - filling continues to fill, for the fill time it had left;
 - settling settles for the time it had left; and
 - aborted stays aborted.
3. It does so whatever the state of its mode, and whether or not the clock is valid, because `phaseTime` and the volumes drained and filled are in its saved state (Section 7.9).
4. The fill target of "drained" is the volume drained as last saved. Because progress is saved at least every 10 s and after each dose, the error is at most 10 s of drain flow; the fill's `until` and `limit` still apply.

A controller restarted halfway through a drain cannot tell how long it was off, or whether its new water is still ready. Filling at once returns the system to its normal level, a smaller water change, which is the safe outcome. One commercial controller documents the same behavior; a water change written as ordinary timed rules can instead leave the fill unrun, so that the top-off refills a drained sump with fresh water.

9.11 Aborting, Resuming, and Abandoning

1. When a water change aborts, it stops all its outputs at once, raises `water-change-aborted`, which latches, and ends its mode. The low-water interlocks of Section 12 then protect heaters and pumps, and the top-off stays stopped, unless the water change drained no more than 1 % of the system's water (Section 8.13.1).
2. A person who has checked the system may then either:
 - resume it: the water change goes on to fill, with the full `fill.maxTime`; or
 - abandon it: the water change returns to idle, and the top-off may fill again. Both need the `clear-latch` operation [OAP-SA Table 6], and resuming needs the fill's inputs to be available and no leak.
3. Before a person abandons a water change, a controller SHOULD warn that the top-off will replace the missing water with fresh water, which lowers the salinity of a marine system, and, for a top-off with an additive, adds as much of the additive as its daily limit allows (Section 8.16).
4. A water change aborted by a leak MUST NOT refill until a person resumes it.

9.12 Water Changes with Dosing Pumps

Dosing pumps can carry out small, frequent water changes: one head removes old water, and another adds new water.

1. A phase whose outputs are `doser` endpoints sends `dose` commands, each no larger than the head's `maxDoseVolume`, the volume the phase has left, and the volume the head delivers at the dose's rate in its `maxContinuousRun`, which Device Classes makes REQUIRED from its revision 0.5 [OAP-DC Section 9.7]; a head of an earlier revision may lack it, and then gives no such bound. After each dose, the controller lets the head rest for at least as long as the dose ran (provisional), as Dosing Section 14.7 does for a long dose [OAP-DO Section 14.7], because a head refuses a run longer than its `maxContinuousRun`, and may refuse one that starts before it has rested [OAP-DC Section 7.14.3]. The water change counts the volume of each `doseCompleted` event.
2. A controller MUST send each dose only once, as Dosing Section 10.5 requires for a program's doses. Before it sends a dose, it saves it as the water change's `dose`, with the head's last reported `dosedToday` as `before` (Section 7.9), and it MUST NOT send that dose again after a restart, a power cut, a lost `doseCompleted` event, or a lost response. It counts what the dose delivered as Dosing Section 10.5.3 says. When that cannot be known, it counts a drain dose as having drained nothing, and a fill dose as having filled its whole volume, so that the fill adds less water rather than more.
3. A fill of "drained" then adds the volume the drain removed, within the heads' calibration. When the new-water head cannot deliver it, for example because it reports `empty` or refuses a dose, the water change aborts, so no further water is removed.
4. For a phase of `doser` outputs only, `limit` is RECOMMENDED rather than REQUIRED: the heads' `maxDoseVolume` and `maxDailyVolume` [OAP-DC Section 9.7] bound what the phase can move. A controller

MUST check that both are set on each such head before it runs the water change, and skips the water change otherwise (Section 9.4.3).

5. A fill of `doser` outputs MUST have `until` or `limit`, so that a level sensor stops a fill that would raise the level too far. The heads' calibrations always differ a little, and over many water changes the differences add up, so that a fill of `"drained"` alone would slowly raise or lower the system's level.
6. Each head of a water change MUST be calibrated with the water it moves, as Dosing Section 8.2 describes, before the water change runs it, and SHOULD be calibrated again at the interval that Dosing Section 8.4 recommends. A controller records when it last calibrated each head as `calibrated` in the saved state, and MUST NOT run a water change on a head that has not been calibrated since it was bound to the water change.
7. A controller MUST watch each head of a water change for unexpected dosing, as Dosing Section 14.5 defines it, with the doses the water change sent in place of a program's. When a head doses unexpectedly, the controller sends it `stop`, and the water change raises `stuck-on`, which latches: a water change in progress aborts (Section 9.11), and no water change starts until a person clears it.
8. Each head's `maxDailyVolume` SHOULD be about 1.5 times (provisional) the volume the water change moves in a day, and its `maxDoseVolume` no larger than the volume of one water change. A controller SHOULD propose these values, and SHOULD warn the owner when a head's limits are more than twice what this item proposes, because a head that runs away can move all of it.
9. Continuous water changes are many small ones, each started by its mode's `start.when`, such as a schedule every two hours. The mode of such a water change usually lists only the top-off.
10. A head driven by a water change is not also driven by the Dosing specification's programs, because an output has only one function. Its doses still count towards its `dosedToday` and `maxDailyVolume`. A controller primes it, or runs its calibration, only when a person asks [OAP-DC Section 7.14.5].
11. A controller MUST write to the `program` of each head of a water change that has the `local-program` feature what Dosing Section 15.3.2 has it write to a head that a controller runs, within a minute of their differing: a program that runs no doses of its own and carries the local time zone [OAP-DO Section 15.3.2]. So the head neither keeps running a program that an earlier configuration gave it, nor loses track of local midnight, at which its `dosedToday` starts again [OAP-DO Section 14.2.4].

For example, a water change that moves 250 mL every two hours, 3 L a day, gives each head a `maxDoseVolume` of 250 mL and a `maxDailyVolume` of 4.5 L. A head whose `maxContinuousRun` is 120 s, dosing at 100 mL/min, moves each 250 mL as a dose of 200 mL, a rest of at least 120 s, and a dose of 50 mL.

9.13 Water-Change Modes and Expected Conditions

1. The Automation presets give the water-change mode actions for a water change done by a person [OAP-AU Table 29]. For a water change done by this function, Table 12 gives the RECOMMENDED actions where they differ.

Table 12 — Recommended mode actions for an automatic water change

Equipment	Action	Why
auto-top-off	stop	REQUIRED in effect by Section 8.13.
return-pump	follow while <code>drain.limit</code> is above the pump's intake level; otherwise stop	Keeps the system's water moving, so the water drained is the system's water.
heater	follow while <code>drain.limit</code> is above the heater's cover level; otherwise stop	The low-water interlock still protects it (Section 12.2).
circulation-pump	follow	
protein-skimmer	stop, then 600 s more	Its level depends on the sump's.

2. A water change of a system makes one condition expected: `low-water`, raised by each `level-switch` endpoint of that system whose `position` is `low` (Section 12.5), in the draining, filling, and settling phases. It makes no other condition expected, and none in the checking phase, while it is aborted, or while an interruption, such as a power-loss mode, holds it (Section 9.10).
3. While an expected condition is active, the controller raises, shows, and records its alarm as usual, and holds only its sound and its notifications [OAP-AN Section 12.8.4]. It never drops the alarm: as soon as the water

change ends or aborts, or an interruption holds it, the controller sounds and notifies the alarm if it is still active, as if it had just been raised, and notifies it if it has cleared but waits to be acknowledged [OAP-AN Section 12.8.5]. So a sump left low when a power cut stops a water change calls a person at once, rather than staying silent for the whole outage. The water change's backup low limit and the low-water interlocks of Section 12 act throughout, whatever the alarm's sound does.

4. During a water change done by a person, `low-water` is raised and sounds as usual: it is critical, and modes never hold critical alarms [OAP-AN Section 12.8.3], so the person silences or acknowledges it. Warnings, such as `reservoir-low` from the new-water reservoir and values outside the owner's acceptable range, are held during either kind, as Alarms and Notifications Section 12.8.1 already requires.

For example, while an automatic water change drains a sump below its low sensor, `low-water` is raised and shown in the apps, but does not sound; if the fill does not bring the level back above the sensor, the alarm sounds and is notified as soon as the water change ends.

9.14 Choosing Values

As for top-off, the phase time limits depend on the system, and the owner sets them.

1. `drain.maxTime` is about 1.5 times the time the drain takes to remove the planned volume at its flow rate, including, for dosing heads, the rests between their doses (Section 9.12).
2. `fill.maxTime` is about 1.5 times the time the fill takes to add it.
3. `drain.limit` sits above the higher of the cover level of the heaters and the intake level of the pumps in the vessel drained (Section 12.1).

For example, a 10 % water change on a 300 L system replaces 30 L. With a drain pump and a fill pump each moving 10 L/min, each phase takes 3 minutes, so each `maxTime` is 270 s, and the mode's `duration` is at least 270 + 270 + 600 s of settling, plus 900 s of `maxPause` for each phase, or about 50 minutes (Section 9.4.6). The mode ends as soon as the water change does, so a long `duration` holds nothing for longer. Such a water change can remove 10 % of the system's water, so it checks its new water's salinity and temperature (Section 9.5.6).

9.15 Records

1. A controller SHOULD record each water change that drained water as a Data Model event of the type `water-change` [OAP-DM Section 8.6]: `time` when the drain started, `end` when the fill ended, `amount` the volume filled, or drained if that is all it knows, `equipment` the Data Model equipment of its outputs, if they share one, and `notes` saying why, if it did not finish.
2. It SHOULD record the new water's salinity and temperature, measured in the checking phase, as observations of the new-water reservoir's vessel.
3. A controller SHOULD record top-off as events of the type `top-off`, either one per fill, or one per local day, with `occurrences` the number of fills, `time` the start of the first, `end` the end of the last, and `amount` their total [OAP-DM Section 8.6.2]. For a top-off with an additive, `product` is the additive's product, as a Data Model Product: its `name`, `manufacturer`, and `gtin`, without its `effect`.
4. A controller SHOULD record each refill of a reservoir, by a person or by an RO/DI function, as an event of the type `refill`, with `vessel` the reservoir and `amount` the volume added, where it is known.
5. A controller SHOULD record each run of an RO/DI system as an event of the type `ro-di-run`, with `time` and `end` the run, `vessel` the reservoir it filled, `amount` the product water made, and `wasted` the water sent to the drain by flushes, rinses, and the membrane's concentrate, where they are known.

10 RO/DI Systems

10.1 Overview

An RO/DI system makes product water from mains water through sediment and carbon filters, a reverse osmosis membrane, and deionization resin, and stores it in a reservoir for top-off or new water. A controller opens its feed valve to make water, and closes it when the reservoir is full.

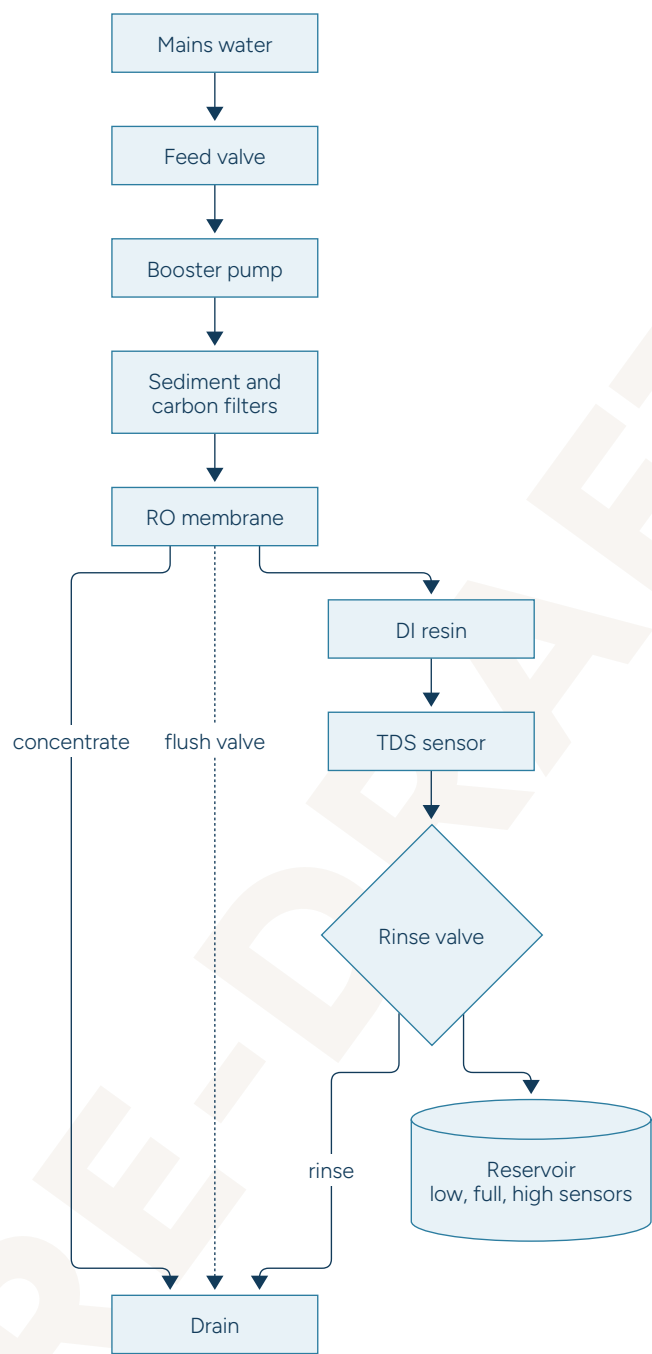


Figure 4 — An RO/DI system and its valves

Because the feed comes from the mains, an RO/DI system can supply water without limit. A feed valve stuck open, or a full sensor that fails, floods the room, so every run is limited by a backup high sensor and a time limit, and a leak closes the feed.

10.2 Members

An RO/DI function has the members of Table 2 and those of Table 13.

Table 13 — Members of an RO/DI function

Member	Type	Presence	Description
vessel	identifier	RECOMMENDED	The reservoir it fills.
feed	array	REQUIRED	The outputs that make water: the feed valve, and a booster pump if there is one.

Member	Type	Presence	Description
<code>full</code>	reference	REQUIRED	True when the reservoir is full.
<code>high</code>	reference	REQUIRED	True when the water is at the reservoir's backup high sensor.
<code>low</code>	reference	OPTIONAL	True when the reservoir is low enough to start making water. Without it, a run starts whenever <code>full</code> is false.
<code>when</code>	reference	OPTIONAL	Water is made only while this is true, for example during the night.
<code>maxRunTime</code>	number (s)	REQUIRED	The longest a run may last (Section 10.5).
<code>maxDailyRunTime</code>	number (s)	OPTIONAL	The most run time in any 24 hours. RECOMMENDED.
<code>flush</code>	object	OPTIONAL	Membrane flushing (Section 10.6): <code>outputs</code> , <code>before</code> , <code>after</code> , <code>interval</code> , and <code>duration</code> .
<code>rinse</code>	object	OPTIONAL	Rinsing (Section 10.7): <code>outputs</code> and <code>time</code> .
<code>tds</code>	object	OPTIONAL	TDS monitoring (Section 10.7): <code>input</code> , <code>max</code> , <code>delay</code> (default 600 s), and <code>stop</code> (default false).
<code>leak</code>	reference	OPTIONAL	True while a leak is detected in the RO/DI system's zone. RECOMMENDED.
<code>flowRate</code>	Quantity	OPTIONAL	The rate at which it makes product water, for estimating volumes.

10.3 Safety Checks

Table 14 — RO/DI safety checks

Check	Presence	Applies to	What it requires	Section
Maximum run time	REQUIRED	Controllers	Every run that makes water stops when it has lasted <code>maxRunTime</code> . The function raises <code>timeout</code> and starts no run until a person clears it.	10.5
Backup high sensor	REQUIRED	Controllers	A second sensor above the reservoir's full level stops every run of the feed outputs at once, at the precedence of interlocks, and raises <code>high-level</code> , which stays until a person clears it.	10.5
Unavailable sensor means full	REQUIRED	Controllers	A full or high condition that is unavailable counts as true: the feed outputs stop and no run starts until it is available again.	10.4
Stop on a leak	REQUIRED	Controllers	While <code>leak</code> is true or unknown, the feed outputs stop and no run starts. A leak in the system also locks the function out until a person clears the lockout.	10.4
Runs end by themselves	REQUIRED	Controllers	Each command that runs a feed, flush, or rinse output carries <code>revertAfter</code> , no longer than the run's time left plus 2 s.	10.5
Feed closed without control	REQUIRED	Controllers	Feed outputs stop on loss of contact, and the controller runs the function only once they do. A controller warns the owner when a feed valve's endpoint reports <code>normallyOpen</code> true. A booster pump runs only while the feed valve reports open.	10.8
Daily limit	RECOMMENDED	Controllers	The run time in any 24 hours is limited to <code>maxDailyRunTime</code> . When the limit is reached, the function stops, raises <code>daily-limit</code> , and runs again only when the runs of the last 24 hours allow.	10.5
Membrane flushing	OPTIONAL	Controllers	The flush outputs open, with the feed outputs running, for <code>before</code> seconds at the start of each run, for <code>after</code> seconds at its end, and for <code>duration</code> seconds after every <code>interval</code> seconds of making water.	10.6
TDS monitoring	OPTIONAL	Controllers	After <code>delay</code> seconds of a run, a product water TDS above <code>max</code> raises <code>tds-high</code> , and stops the run if <code>stop</code> is true. A rinse sends product water to drain at the start of a run.	10.7

10.4 Making Water

1. An RO/DI function starts a run when `low` is true, or, without `low`, when `full` is false, and all of these hold: `when`, if given, is true; `full` and `high` are false; `leak` is false; no fault is latched; the daily limit allows it; and nothing holds its feed outputs.
2. A run flushes first, if `flush.before` is set, then makes water: it opens the feed valves, then starts any pump in `feed` once the valves report open, and rinses for `rinse.time`, if set.
3. A run ends normally when `full` becomes true. It then flushes, if `flush.after` is set, and records a refill of every top-off reservoir in its `vessel` (Section 8.12).
4. A run also ends when `high` becomes true, at `maxRunTime`, at the daily limit (Section 10.5), when `leak` becomes true or unknown, when `full` or `high` becomes unavailable, or when something holds its feed outputs.
5. A run stops its pumps first, then closes its valves.
6. While `full` or `high` is unavailable, it counts as true (Section 7.4). While `low`, `when`, or `leak` is unknown, no run starts.
7. A leak in its system also locks the function out until a person clears the lockout (Section 11.2).

10.5 Limits

1. Every run MUST stop at `maxRunTime`, which raises `timeout`, which latches.
2. While `high` is true, every run of the feed outputs MUST stop at once, whatever asks for it, and the function raises `high-level`, which latches (Section 7.3).
3. An RO/DI function SHOULD have `maxDailyRunTime`, not less than `maxRunTime`. When the run time of the last 24 hours reaches it, the run stops and the function raises `daily-limit`, which does not latch.
4. `maxRunTime` and `maxDailyRunTime` apply to every run of the feed outputs, including flushes, and every command that runs a feed, flush, or rinse output carries `revertAfter` (Section 7.5).

10.6 Flushing

Flushing runs water quickly across the membrane to the drain, which washes off deposits and helps it last.

1. With `flush`, the function opens `flush.outputs`, with its feed outputs running, for `before` seconds at the start of each run, for `after` seconds at its end, and for `duration` seconds after every `interval` seconds of making water.
2. During a flush, `rinse.outputs`, if any, are open, so that product water goes to the drain.
3. Flushes are part of the run, so the run's limits apply to them (Section 10.5).
4. One RO/DI maker advises flushing for 1 to 2 minutes before and after making water; one commercial controller flushes for 60 s at the start and after every hour of running.

10.7 Rinsing and TDS

The first water a membrane makes after standing has a higher TDS than the rest, until its rejection recovers.

1. With `rinse`, the function opens `rinse.outputs` for the first `time` seconds of each run, so that this water goes to the drain. One RO/DI maker advises running for 10 minutes before testing the TDS of new product water.
2. With `tds`, after `delay` seconds of a run, or after the rinse if that is longer, a product water TDS above `max`, in the unit of `tds.input`, raises `tds-high`. With `stop true`, the run also stops.
3. `tds-high` clears when a later run's TDS, after the delay, is at or below `max`.
4. `tds.input` SHOULD measure the water after the DI resin, so that `tds-high` shows that the resin needs replacing. One RO/DI maker says this water should read zero, so a `max` of 1 mg/L is a usual choice.

10.8 Valves and Pumps

1. A feed valve SHOULD be closed when unpowered (`normallyOpen false` [OAP-DC Section 9.8]), and every feed output MUST stop on loss of contact (Section 7.5.5). A controller MUST warn the owner when a feed valve's endpoint reports `normallyOpen true`.
2. A pump in `feed` MUST run only while every valve in `feed` reports open.

3. A reservoir filled by an RO/DI system SHOULD also have a mechanical float valve, or an RO automatic shut-off valve, that stops the product water when the reservoir is full, as a layer of protection that works without electricity.

10.9 Choosing Values

1. `maxRunTime` is about 1.5 times the time the system takes to fill the reservoir from its low sensor to its full sensor, at its slowest production rate. Production falls in cold water and at low mains pressure.
2. `maxDailyRunTime` is about twice the run time of a day of the system's heaviest use.

For example, a membrane rated at 75 US gallons a day makes about 12 L/h at its rated pressure and temperature, and perhaps 8 L/h in winter. Refilling 20 L between the low and full sensors then takes up to 2.5 h, so `maxRunTime` is about 3.75 h.

11 Leaks and Overflows

11.1 Leak Sensors

1. Leak sensors are `leak-sensor` endpoints [OAP-DC Section 11.3], read as Automation endpoint inputs. They SHOULD have the `supervised` feature.
2. Owners SHOULD place them where water escapes first: under the sump, under external filters, around the RO/DI system and reservoirs, and in the cabinet.
3. Each leak sensor's input SHOULD give the `system` it protects, or be the input of a Data Model equipment item that does.
4. A system SHOULD have at least one leak sensor under its sump that is connected to its controller by wire [OAP-FS Section 13.4.4], because a radio link is never the only path for a safety function; sensors on radio add coverage elsewhere.
5. Leak response is a safety function [OAP-FS Section 13.4]: for each system, its leak lockout (Section 11.2.3), and the interlocks and emergency modes whose conditions depend on a leak sensor of that system, are one safety function [OAP-AU Section 13.7.3]. It has a path that uses no radio link for an output that it stops when one of them stops the output with one, as Fail-Safe Behavior Section 13.4 judges a path:
 - the output's device is connected to the controller by wire, or stops the output by itself when it loses contact, with a `failSafeTimeout` of at most 300 s (provisional), and the leak sensors whose devices are connected to the controller by wire are enough to stop it, whatever the others report [OAP-FS Section 13.4.3], as a wired leak sensor is for the leak lockout, and for an emergency mode that starts on `any` of the system's leak sensors; or
 - the output's own device stops it on a leak, with a leak sensor of its own, as a top-off unit does (Section 8.14.4) [OAP-FS Section 13.4.2]. So while every leak sensor of a system is on a radio link, its leak response has no such path for any output that it stops with the controller's leak sensors alone, because a lost link hides a leak rather than stopping anything.
6. A controller MUST check the leak response of each system for such a path, for each output that it stops, after the controller starts, and whenever its configuration, a binding, or an endpoint's fail-safe configuration changes [OAP-FS Section 13.4.5]. While the leak response has no such path for an output, the controller MUST show its owners the standing warning of Fail-Safe Behavior Section 13.4.6, naming the system, the outputs, and its leak sensors, and saying what would give it one, such as a leak sensor wired to the controller, or a `failSafeTimeout` of 300 s or less, and MUST raise `safety-path-missing` [OAP-AU Section 13.7.6]. It MUST NOT refuse the configuration because of it [OAP-FS Section 13.4.7]. The system of a leak sensor is defined in Section 11.2.3.
7. A leak sensor raises the `Leak` alarm by itself [OAP-AN Section 8.7].

11.2 Leak Response

The leak response is an Automation emergency mode [OAP-AU Section 14.7].

1. Each leak sensor SHOULD start an emergency mode, through the mode's `start.when`, usually an `any` condition of the leak inputs of one zone. A controller SHOULD warn the owner about a leak sensor that starts no emergency mode.
2. The mode's actions SHOULD follow Table 15 for each output's equipment category, for every system of its zone. A controller SHOULD propose them when the owner creates the mode, and SHOULD warn about actions that differ.
3. Whatever the configuration, a controller MUST hold stopped every output of every top-off, RO/DI, and water change function of a system, at the precedence of interlocks (Section 7.3), from the moment a leak sensor of that system reports a leak until a person clears the hold after the sensor is dry: the leak lockout. The system of a leak sensor is the `system` of the inputs that read it, or of its Data Model equipment; a leak sensor of a device the controller controls whose system is unknown locks out every system. An RO/DI system whose reservoir serves several systems SHOULD also list the leak sensors of the others in its `leak`. An emergency mode SHOULD still list these outputs, so that the owner sees them stopped there. The lockout does not hold dosing heads that a dosing function drives, which add little water; a leak emergency mode stops them, as Table 15 recommends [OAP-DO Section 13.2] (Appendix D, issue 25).
4. A leak sensor whose input, or whose Data Model equipment, names a vessel that is not circulating, such as a top-off or new-water reservoir, protects that reservoir and the equipment that fills and empties it. An emergency mode started only by such sensors need stop only supply equipment, as the "On a fresh-water leak" column of Table 15 does; the leak lockout of item 3 still applies.
5. Only a person ends the mode, after checking the system, and the mode starts again while a sensor still reports a leak [OAP-AU Section 14.3].
6. A leak aborts a water change (Section 9.11), and locks out the top-off and the RO/DI system (Sections 8.13 and 10.4).

```
{ "id": "leak-emergency", "kind": "emergency", "name": "Leak", "start": { "when": "leak-any" },
  "outputs": { "return-pump": { "state": "stop" }, "heater": { "state": "stop" }, "skimmer": { "state": "stop" },
    "ato-pump": { "state": "stop" }, "ro-feed": { "state": "stop" }, "drain-pump": { "state": "stop" },
    "fill-pump": { "state": "stop" } } }
```

11.3 Response by Equipment Category

Table 15 gives the RECOMMENDED action of a leak emergency mode for each Data Model equipment category: `stop`, or `follow`, which leaves the output to the other layers. It agrees with the emergency actions of the Automation presets [OAP-AU Table 29], including the fan preset, whose category is `fan` [OAP-DM Table 24]. It is published in machine-readable form in `checks.json`.

Table 15 — Recommended response to a leak by equipment category

Equipment category	On a leak	On a fresh-water leak	Notes
<code>return-pump</code>	<code>stop</code>	<code>follow</code>	Stops moving water out of the sump. The display then drains down to its overflow, so the sump must have room for that water.
<code>circulation-pump</code>	<code>follow</code>	<code>follow</code>	Flow within a vessel does not move water out of it, and keeps the water oxygenated while the return pump is stopped.
<code>air-pump</code>	<code>follow</code>	<code>follow</code>	Aeration keeps the water oxygenated while the return pump is stopped.
<code>heater</code>	<code>stop</code>	<code>follow</code>	A leak may lower the water below the heater.
<code>chiller</code>	<code>stop</code>	<code>follow</code>	Its pump and hoses may be the source of the leak. Condensation from a chiller can wet a sensor with fresh water.
<code>light</code>	<code>follow</code>	<code>follow</code>	
<code>doser</code>	<code>stop</code>	<code>follow</code>	Stops adding liquids while a person investigates.
<code>auto-top-off</code>	<code>stop</code>	<code>stop</code>	Top-off would replace the water that leaks away, and keep the leak going. Fresh water on the floor often comes from its reservoir or tubing.
<code>protein-skimmer</code>	<code>stop</code>	<code>follow</code>	The sump level changes when the return pump stops, and a skimmer can overflow.
<code>filter</code>	<code>stop</code>	<code>follow</code>	External canister filters and their hoses often leak. A filter below the tank can keep siphoning after its pump stops, unless it has a shut-off or anti-siphon valve.

Equipment category	On a leak	On a fresh-water leak	Notes
reactor	stop	follow	
uv-sterilizer	stop	follow	It needs the flow that stops with the return pump.
ozone-generator	stop	follow	
co2-system	stop	follow	
feeder	stop	follow	
algae-scrubber	stop	follow	Its pump moves water out of a vessel.
analyzer	follow	follow	It draws only small samples.
ro-di	stop	stop	An open feed valve supplies water from the mains without limit.
water-changer	stop	stop	Its pumps move water into and out of the system. A running water change aborts (Section 9.11).
alarm-signal	follow	follow	It signals the leak's alarm (Alarms and Notifications Section 15).
fan	follow	follow	It moves air, not water.
other	stop	follow	When in doubt, stop, and check what the equipment needs.

11.4 Water Type

A leak sensor with the `water-type` feature tells fresh water from saltwater [OAP-DC Section 11.3]. In a marine or brackish system, fresh water on the floor usually comes from the RO/DI system, the top-off reservoir, or their tubing, rather than from the system itself.

1. In a marine or brackish system, an owner MAY give fresh-water leaks a smaller response, the "On a fresh-water leak" column of Table 15, which stops only supply equipment (Appendix D, issue 4). It is off until the owner chooses it: a controller proposes only the full response. The full response then applies to every leak whose water type is not known to be fresh.
2. A leak whose water type is `unknown`, or whose `waterType` is unavailable, MUST get the full response.
3. In a freshwater system, the system's own water is fresh, so water type MUST NOT reduce the response.
4. The alarm for a leak SHOULD show its water type, to direct the owner to its likely source.

Because a condition is unknown while its input is unavailable, the full response's start condition is written so that only a known fresh-water reading excludes it:

```
[
  { "id": "type-fresh", "type": "compare", "input": "floor-leak-type", "equals": "fresh" },
  { "id": "type-known", "type": "not", "of": "type-unavailable" },
  { "id": "type-unavailable", "type": "unavailable", "of": "floor-leak-type" },
  { "id": "known-fresh", "type": "all", "of": ["type-fresh", "type-known"] },
  { "id": "not-known-fresh", "type": "not", "of": "known-fresh" },
  { "id": "leak-full", "type": "all", "of": ["floor-leak", "not-known-fresh"] }
]
```

`known-fresh` is false, not unknown, while the water type is unavailable (Automation Table 12), so `leak-full` follows the leak alone.

11.5 Unavailable Leak Sensors

1. A leak sensor that reports `wiring-fault`, or whose device is lost, raises an alarm [OAP-AN Section 8.4], and its leak is unknown.
2. An unknown leak neither starts nor ends an emergency mode [OAP-AU Section 14.3].
3. An RO/DI system or a water change that lists it in `Leak` does not start, and an RO/DI run stops (Sections 9.4 and 10.4).
4. A top-off continues, because its own limits bound what it can add (Section 7.4).

11.6 Overflow Sensors

A level switch whose `position` is `overflow` detects water about to overflow its vessel, and raises the `overflow` alarm by itself [OAP-AN Section 8.7].

1. Each overflow sensor SHOULD hold stopped, with a latching `block` interlock [OAP-AU Section 13.2], the equipment that adds water to its vessel:
 - for any vessel, the outputs of top-off, RO/DI, and water change fills whose `vessel` it is; and
 - for a display, also the return pump, which fills it from the sump.
2. The interlock on supply equipment SHOULD have `onUnavailable block`. The interlock on a return pump SHOULD have `onUnavailable allow`, because a return pump is vital equipment [OAP-AN Section 7.2], at the risk of filling a display whose drain is blocked while its overflow sensor is lost (Appendix D, issue 3).
3. A sump's overflow sensor MUST NOT stop the return pump: stopping it would add the display's drain-down to the sump.

11.7 Siphons and Drain-Down

This section is informative.

Stopping a pump does not stop a siphon. When a return pump stops, water can siphon back from the display through the return line until the outlet clears the surface, and an external filter below the tank can keep siphoning through a split hose. A sump must have room for the display's drain-down and for any siphon-back, or stopping the return pump on a leak can overflow it. Anti-siphon holes near the surface on return outlets, check valves, and shut-off valves on filter hoses limit what a leak response cannot control.

11.8 Safety Checks

Table 16 — Leak and overflow safety checks

Check	Presence	Applies to	What it requires	Section
Leak lockout	REQUIRED	Controllers	Whatever the configuration, a leak locks out every top-off, RO/DI, and water change of its system, at the precedence of interlocks, until a person clears the lockout after the sensor is dry.	11.2
Leak response without radio	REQUIRED	Controllers	A system's leak response has a path that uses no radio link for an output that it stops while wired leak sensors alone can stop it, and the output is wired to the controller or stops by itself on loss of contact within a <code>failSafeTimeout</code> of at most 300 s. While it has none, the controller shows its owners a standing warning that names the system, the outputs, and its sensors, raises <code>safety-path-missing</code> , and still runs the configuration.	11.1
Leak emergency	RECOMMENDED	Configurations	Every leak sensor starts an emergency mode that applies the response of Section 11.3 to the equipment of the systems it protects. Only a person ends the mode, after the sensor is dry.	11.2
Water type	OPTIONAL	Controllers, configurations	In a marine or brackish system, an owner may give a fresh-water leak a smaller response that stops only the equipment that brings fresh water. A leak of unknown type, and every leak in a freshwater system, gets the full response.	11.4
Unavailable leak sensors	REQUIRED	Controllers	An unavailable leak sensor raises an alarm, and RO/DI systems and water changes that use it in <code>Leak</code> do not start.	11.5
Overflow sensors	RECOMMENDED	Controllers, configurations	An overflow sensor holds stopped, with a latching interlock, the equipment that adds water to its vessel: top-off, RO/DI, and water change fills, and, for a display, the return pump.	11.6

12 Low Water

12.1 Levels That Matter

1. The water must stay above a heater's cover level, as its maker marks it, and above a pump's intake.

- Each vessel that holds a heater or a pump's intake SHOULD have a low sensor, a `level-switch` with `position low`, above the higher of the cover levels of its heaters and the intake levels of its pumps, with a margin for waves.
- The low sensor SHOULD be in the same chamber as the equipment it protects.

12.2 Heaters

- A heater SHOULD be held stopped by a `block` interlock [OAP-AU Section 13.2] while the water is below its cover level, with `onUnavailable block`, so that it also stops while the level is unknown.
- The interlock SHOULD keep the heater stopped until the water has covered it for 600 s (provisional), through an `off-delay` timer [OAP-AU Section 10.5].
- A controller MUST warn the owner about a heater output that no low-water interlock protects.
- An inline heater, in the plumbing rather than in a vessel, SHOULD require the pump that feeds it, with an Automation `requires` interlock.
- Where the low sensor cannot be mounted well above the heater's cover level, the interlock SHOULD latch, so that a person checks the heater before it heats again.

```
[
  { "id": "sump-low-wet", "type": "endpoint", "class": "level-switch", "equipment": "sump-low-sensor", "vessel": "sump" },
  { "id": "sump-low", "type": "not", "of": "sump-low-wet" },
  { "id": "heater-uncovered", "type": "timer", "kind": "off-delay", "input": "sump-low", "duration": 600 },
  { "id": "heater-low-water", "type": "block", "outputs": ["heater"], "when": "heater-uncovered", "onUnavailable": "block" }
]
```

Heater makers say a heater that has run out of the water must not be put back in until it has cooled for an hour, and that it should cool in the water for 30 minutes before it is taken out. A controller cannot stop the water returning, but it can keep a heater from running dry, which is why the low sensor sits above the cover level. Owners SHOULD switch heaters off before draining below them by hand, which the maintenance and water-change mode presets do. Some heaters have a thermal cut-out that switches them off when they run dry, but switches them on again as they cool, so it is not a substitute for the interlock.

12.3 Pumps

- A return pump, and any other pump that draws from a vessel, SHOULD be held stopped by a `block` interlock while the water is below its intake level, and SHOULD start again only after the water has covered its intake for 120 s (provisional), through an `off-delay` timer.
- The interlock on a return pump SHOULD have `onUnavailable allow`, so that the return pump keeps running while its sensor is unknown, because a return pump is vital equipment and its recommended fail-safe state is running [OAP-FS Table 3] (Appendix D, issue 3). The controller still raises `input-unavailable`.
- The interlocks on other pumps SHOULD have `onUnavailable block`.

When a return pump stops because the sump is low, the display drains down into the sump, the sump's level rises, and the pump starts again and lowers it. The delay limits how often this repeats, and the `low-water` alarm (Section 12.5) calls a person, because the cause, usually a top-off that has stopped, needs one.

12.4 Dry-Run Detection

- A pump that reports `dry-run` [OAP-DC Section 9.3] SHOULD be held stopped by a `block` interlock on an Automation `fault` input [OAP-AU Section 9.4], and tried again only after a delay, through an `off-delay` timer.
- A pump on a monitored output, without dry-run detection of its own, MAY be held stopped the same way by a `compare` condition on its `power`, through an `on-delay` timer that ignores the surge as it starts. Such a condition MUST also require that the pump runs, through an `output` input of the pump without `attribute` [OAP-AU Section 9.3], combined with the `compare` in an `all` condition before the `on-delay`. A stopped pump draws no power, so a `compare` on its `power` alone stays true once the interlock has stopped it, and never lets it run again.
- For a return pump, such an interlock SHOULD have `onUnavailable allow`, as in Section 12.3.

For example, a return pump that normally draws about 60 W is held stopped for 300 s when it draws less than 20 W for 10 s while it runs:

```
[
  { "id": "return-running", "type": "output", "output": "return-pump" },
  { "id": "return-power", "type": "output", "output": "return-pump", "attribute": "power", "unit": "W" },
  { "id": "return-power-low", "type": "compare", "input": "return-power", "below": 20 },
  { "id": "return-dry", "type": "all", "of": ["return-running", "return-power-low"] },
  { "id": "return-dry-for", "type": "timer", "kind": "on-delay", "input": "return-dry", "duration": 10 },
  { "id": "return-dry-hold", "type": "timer", "kind": "off-delay", "input": "return-dry-for", "duration": 300 },
  { "id": "return-dry-run", "type": "block", "outputs": ["return-pump"], "when": "return-dry-hold", "onUnavailable": "allow" }
]
```

Once the interlock has stopped the pump, `return-dry` is false, and the pump runs again when the `off-delay` ends; if it is still dry, it stops again 10 s later.

12.5 Low-Water Alarms

1. Every `level-switch` endpoint whose `position` is `low`, of a device the controller controls, raises `low-water` while it reports `wet` false, after a delay of 30 s (provisional).
2. As with leak and overflow alarms, this does not depend on the automation configuration [OAP-AN Section 8.7].
3. While an automatic water change of its system drains, fills, or settles, `low-water` is an expected condition (Section 9.13): it is raised, shown, and recorded as usual, but its sound and its notifications are held until the water change ends or aborts, or an interruption holds it, when it sounds at once if it is still active. The system of a sensor is the `system` of the inputs that read it, or of its Data Model equipment.
4. While a supervised sensor reports `wiring-fault`, its state is unknown, so it neither raises nor clears `low-water`.
5. `low-water` is `critical` by default (Table 5). An owner who mounts a low sensor as an early warning can lower it to `warning`, but cannot turn it off, because it is a protected alarm [OAP-AN Section 7.3] (Appendix D, issue 7).

12.6 Safety Checks

Table 17 — Low-water safety checks

Check	Presence	Applies to	What it requires	Section
Heaters stay covered	RECOMMENDED	Configurations, controllers	A heater stops when the water falls below the level that covers it, and while that level is unknown. It starts again only after the water has covered it for a delay. A controller warns the owner about a heater that no low-water interlock protects.	12.2
Pumps stay covered	RECOMMENDED	Configurations	A return pump, and any other pump that draws from a vessel, stops when the water falls below its intake level, and starts again after the water has covered it for a delay. While that level is unknown, the return pump keeps running.	12.3
Pump dry-run detection	OPTIONAL	Controllers, configurations	A pump that reports <code>dry-run</code> , or whose power falls below its dry-run threshold while it runs, is held stopped, and is tried again only after a delay. A condition on a pump's power also requires that the pump runs, so that it does not hold the pump stopped for good.	12.4
Low-water alarm	REQUIRED	Controllers	Every level switch whose <code>position</code> is <code>low</code> raises <code>low-water</code> while it reports <code>dry</code> . While an automatic water change of its system drains, fills, or settles, the alarm is raised and shown, but its sound and notifications are held until the water change ends or aborts, or an interruption holds it, when it sounds at once if it is still active.	12.5

13 Fail-Safe Behavior

13.1 Outputs

Table 18 gives the RECOMMENDED fail-safe configuration [OAP-FS Section 13] of the outputs of each function. It follows the Fail-Safe Behavior recommendations for the categories `auto-top-off`, `water-changer`, `ro-di`, and `doser` [OAP-FS Table 3].

Table 18 — Recommended fail-safe configuration of outputs

Output	On loss of contact	After power-up	On battery	Notes
Top-off pump or valve, driven by a controller	<code>stop</code>	<code>fail-safe</code>	<code>stop</code>	Its runs are limited by <code>revertAfter</code> as well (Section 7.5).
Top-off unit	<code>continue</code>	<code>last</code>	<code>stop</code>	It runs its own logic, sensors, and limits while it is enabled, and keeps its last commanded <code>enabled</code> , so a unit that its controller disabled stays disabled (Section 8.14.7).
Top-off unit whose water carries an additive	<code>state</code> , with <code>failSafeState {"enabled": false}</code>	<code>last</code>	<code>stop</code>	Only its controller enforces the additive's limits, so it stops filling when it loses its controller, at its <code>failSafeTimeout</code> (Section 8.16.7).
Water change drain and fill outputs	<code>stop</code>	<code>fail-safe</code>	<code>stop</code>	An interrupted water change resumes as Section 9.10 says.
Dosing heads of a water change	<code>stop</code>	<code>fail-safe</code>	<code>stop</code>	A dose that loss of contact or a power cut interrupts does not resume [OAP-DC Section 7.14.1].
RO/DI feed valves and booster pumps	<code>stop</code>	<code>fail-safe</code>	<code>stop</code>	A feed valve should also be closed when unpowered (Section 10.8).
RO/DI flush and rinse valves	<code>stop</code>	<code>fail-safe</code>	<code>stop</code>	

13.2 Lost Inputs

Each function defines what it does when an input is unavailable, as Fail-Safe Behavior Section 10.3 requires. Table 19 summarizes them.

Table 19 — What each function does when an input is unavailable

Function	Input	Behavior	Section
Top-off	<code>level</code> , <code>high</code>	Counts as true: no fill; resumes when available.	8.8
Top-off	<code>reservoir.empty</code>	Counts as true: no fill.	8.12
Top-off	<code>dryRun.input</code>	The dry-run check is suspended; fills continue under the other checks.	8.11
Water change, checking	Any input it needs	Skipped.	9.5
Water change, draining	<code>drain.until</code> , <code>drain.limit</code>	The drain ends; the water change goes on to fill.	9.6
Water change, draining	<code>fill.until</code> , <code>fill.limit</code> , <code>fill.supplyEmpty</code>	Aborts.	9.6
Water change, draining	<code>leak</code>	The drain ends; the water change goes on to fill.	9.6
Water change, filling	<code>fill.until</code> , <code>fill.limit</code> , <code>fill.supplyEmpty</code>	Aborts.	9.7
Water change, filling	<code>leak</code>	Continues to fill.	9.7
RO/DI	<code>full</code> , <code>high</code> , <code>leak</code>	The run stops; no run starts.	10.4
RO/DI	<code>low</code> , <code>when</code>	No run starts.	10.4
RO/DI	<code>tds.input</code>	The TDS check is suspended.	10.7

Function	Input	Behavior	Section
Leak response	A leak sensor	Neither starts nor ends an emergency mode; alarm.	11.5
Low water	A heater's low sensor	The heater stops.	12.2
Low water	A return pump's low sensor	The return pump keeps running; alarm.	12.3

When the device of an output is lost, its fail-safe behavior stops it (Section 7.5.5): a top-off fill or an RO/DI run ends, and a water change pauses once the controller takes the output to be stopped (Section 9.9.4).

13.3 Restarts and the Clock

1. After a restart, each function restores its saved state (Section 7.9), and follows Fail-Safe Behavior Section 9.4: it commands nothing until its inputs have current values or have been found unavailable.
2. A top-off counts its minimum interval from the restart (Section 8.6). An RO/DI run in progress ends, and a new one starts when its conditions allow. A water change follows Section 9.10.
3. Latched faults stay latched, whatever the clock.
4. Daily limits use elapsed time (Section 7.7). Scheduled water changes do not start while the clock is not valid, because their schedules are unknown [OAP-AU Section 10.6], and an RO/DI function whose `when` is a schedule starts no run.

13.4 Firmware Updates

1. A controller MUST NOT start its own firmware update, or a restart that a person or an automatic update has scheduled, while a water change drains or fills, or an RO/DI system makes water. It SHOULD wait until they finish, or ask the person.
2. A device does not start a firmware update while one of its endpoints fills or takes part in a water change [OAP-FS Section 11.2].

13.5 A Failed Controller

When the controller fails:

1. every run it started ends by its `revertAfter`, or earlier at its outputs' fail-safe timeout (Section 7.5);
2. a top-off unit with `failSafeAction continue` keeps its last commanded `enabled` [OAP-FS Section 7.3]: an enabled unit continues on its own sensors and limits; one the controller had disabled for a water change or the leak lockout stays disabled until a person ends the hold; and one disabled with `revertAfter`, such as for a feed mode, or by its `requires` interlock on a return pump that a feed mode paused, fills again when the revert falls due (Section 8.14.7); a unit whose water carries an additive, with `failSafeAction state`, stops filling at its `failSafeTimeout` (Section 8.16.7);
3. an RO/DI system stops; and
4. a water change stops wherever it was, and the system may stay short of water until the controller returns and resumes it (Section 9.10).

A failed controller cannot raise an alarm, so the alarm outputs and heartbeats of the Alarms and Notifications specification tell the owner [OAP-AN Section 18].

14 Recommended Values

Table 20 gives the RECOMMENDED values that do not depend on the size of a system. They are provisional, and are published in machine-readable form in `checks.json`. Values that depend on the system, such as `maxFillTime`, the phase time limits, and `maxRunTime`, are set by the owner, with the help of Sections 8.15, 9.14, and 10.9.

Table 20 — Recommended values

Function	Value	Recommended	Why
Top-off	<code>startDelay</code>	10 s	Rides out waves and splashing at the normal sensor before a fill starts. One common top-off kit waits 10 s.

Function	Value	Recommended	Why
Top-off	High-sensor window	60 s	A high sensor that becomes wet this soon after a fill ends was wetted by the fill (Section 8.7).
Top-off	Return pump delay	300 s	The delay of the top-off's requires interlock on the return pump (Section 8.13): the sump settles after the return pump restarts.
Top-off	learning.factor	3	One common top-off kit uses three times its learned run. Fill times vary with the season and the lights.
Top-off	learning.samples	10	The median of the last 10 normal fills; learning starts after 5.
Top-off	dryRun.delay	5 s	Long enough to ignore the current surge as a pump starts; short enough to protect it.
Top-off	reservoir.lowDays	2 days	Leaves the owner a weekend to refill the reservoir.
Top-off	Additive pH limit	8.45	The pH above which a top-off that carries kalkwasser stops (Section 8.16); makers' examples stop dosing kalkwasser above pH 8.40 or 8.45.
Every function	Margin of revertAfter	2 s	Lets the controller, not the device, normally end a run (Section 7.5).
Automatic water changes	settle	600 s	Lets the level steady and the new water mix before top-off and the other equipment resume. One commercial controller waits 10 minutes.
Automatic water changes	maxPause	900 s	Covers a feed mode; a longer pause leaves the system short of water for too long.
Automatic water changes	check.salinity.maxDifference	1.0 psu	About 0.00075 in specific gravity. Owners often aim for this; no maker publishes a tolerance.
Automatic water changes	check.temperature.maxDifference	2.0 °C	Owners often aim for 1 to 2 °C; no maker publishes a tolerance, and many thermometers are accurate only to ±0.5 °C.
Automatic water changes	Progress saved every	10 s	Bounds the volume a restart can miscount (Section 9.10).
Automatic water changes	Rest after a dose	1 times	The rest after each dose of a water change's dosing head, as a multiple of the dose's running time, as Dosing recommends for the parts of a long dose (Section 9.12).
Automatic water changes	Daily limit of a dosing head	1.5 times	A water change's dosing head's maxDailyVolume, as a multiple of the volume the water change moves in a day (Section 9.12).
RO/DI systems	flush.before	60 s	One RO/DI maker advises 1 to 2 minutes before and after making water.
RO/DI systems	flush.interval	3600 s	With flush.duration 60 s, as one commercial controller flushes during long runs.
RO/DI systems	tds.delay	600 s	TDS is high for the first minutes of a run while the membrane recovers; one RO/DI maker advises running 10 minutes before testing.
Low water	Heater restart delay	600 s	The water must have covered the heater, and settled, before it heats again.
Low water	Pump restart delay	120 s	Keeps a return pump from cycling as the display drains down and refills.
Low water	Low-water alarm delay	30 s	Rides out waves at the sensor.

15 Conformance

15.1 Controllers

A conforming controller:

1. runs top-off, water change, and RO/DI functions as Sections 7 to 10 define, including the checks that act on every run (Section 7.3), unknown values (Section 7.4), `revertAfter` (Section 7.5), latched faults (Section 7.6), saved state (Section 7.9), additives (Section 8.16), and water changes by dosing heads (Section 9.12);
2. uses top-off units as Section 8.14 defines;
3. carries out leak, overflow, and low-water responses configured as Sections 11 and 12 describe, raises `Low-water` (Section 12.5), holds only the alarms that a water change makes expected (Section 9.13), and warns its owners while a top-off or a system's leak response has no path that uses no radio link (Sections 8.17 and 11.1.6);
4. raises the codes of Table 4, and shows what Section 7.11 requires;
5. meets Section 13; and
6. passes the test procedures of Section 15.5 that apply to controllers.

15.2 Top-Off Units

A conforming top-off unit:

1. runs Sections 8.4 to 8.12 with its own sensors and settings, as Section 8.14 defines;
2. implements the Device Classes `auto-top-off` class and the Fail-Safe Behavior specification [OAP-DC Section 13.1]; and
3. passes the test procedures of Section 15.5 that apply to units, with no controller connected and again under a controller's control.

15.3 Configuration Documents

A conforming configuration document is a conforming Automation configuration document [OAP-AU Section 20.1] whose functions of this Specification's types are valid against their schema, and meet every requirement of Sections 7 to 12 that applies to documents, including those a schema cannot express, such as references, roles, and units.

15.4 Schema and Tools

`schema/water-level.schema.json` is normative for the structure of the functions and their saved state. Where it and this document differ, this document governs.

`checks.json` is normative for the safety checks, the test procedures, the leak responses, and the recommended values, and Tables 7, 11, 14, 15, 16, 17, 20, and 21 are generated from it by `tools/render_spec_tables.py`. That tool also checks that every safety check has a test procedure, that every MUST and MUST NOT of Sections 7 to 16 is checked by a test, and that the leak responses agree with the Automation presets. Table 5 is generated from `conditions.json`.

The validator, `tools/validate_water_level.py`, checks configuration documents against the Automation validator, the schema, and the rules a schema cannot express, and warns about configurations that leave out a recommended protection. It is informative.

The `examples` directory contains complete configurations (Appendix B).

15.5 Test Procedures

Tests use a controller with modules or simulated devices whose level switches, leak sensors, and power readings the tester can set, whose power the tester can remove, whose links the tester can make wired or radio, and whose clock the tester can set or stop, and a top-off unit whose sensors the tester can wet and dry. "Within 1 s" allows for the controller's evaluation interval of 1 s [OAP-AU Section 17.1]. Every safety check of Tables 7, 11, 14, 16, and 17 has a test in Table 21, which follows the checks with tests of the other requirements. Each test names the requirements it checks, as sections or as numbered items of sections, and every MUST

and MUST NOT of Sections 7 to 16 is checked by at least one test. A pass criterion that checks what a SHOULD asks applies to a product that does it.

Table 21 — Test procedures

Test	Applies to	Requirements	Method	Pass criteria
Maximum fill time (8.5)	Controllers, top-off units	7.3, 7.6.7, 8.5.1, 8.5.2, 8.14.3	Keep the normal level sensor dry, so that a fill starts and never reaches the level. While <code>timeout</code> is latched, remove and restore the power of the controller or unit. Then clear <code>timeout</code> , and let the level be reached.	The output stops within 1 s of <code>maxFillTime</code> after the fill started, and <code>timeout</code> is raised. It stays latched after power returns, and no fill starts, whatever the sensors do, until it is cleared. After it is cleared, fills start again.
Minimum fill interval (8.6)	Controllers, top-off units	8.6.1, 8.6.2, 8.14.2	Let a fill end at the normal level, then dry the normal sensor at once. Then restart the controller or unit with the sensor dry.	Each new fill starts no earlier than <code>minFillInterval</code> , within 1 s, after the previous fill ended or after the restart.
Backup high sensor (8.7)	Controllers, top-off units	7.3, 8.7.2, 8.7.3, 8.7.4	Wet the high sensor during a fill. Separately, wet it while the normal sensor is dry and no fill runs. Separately, wet it while the normal sensor is also wet and no fill has run for longer than the window of Section 8.7. Then override the top-off output to run, or send a unit <code>fill</code> , with the high sensor wet.	The first stops the output within 1 s and raises <code>high-level</code> , which stays until cleared; so does the second, without a fill. The third raises nothing and starts no fill. The override or manual fill does not run the output.
Unavailable sensor means water (8.8)	Controllers, top-off units	7.4.1, 7.4.3, 8.8	During a fill, open the wiring of the supervised normal sensor; then short it; then make the high sensor's device lost. On a controller, also stop the normal sensor's device from reporting, without disconnecting it, for longer than the input's maximum age.	Each stops the fill within 1 s of the input becoming unavailable, and raises <code>input-unavailable</code> on a controller or <code>wiring-fault</code> on a unit. No fill starts until the input is available again, and fills then resume without a person.
Fills end by themselves (8.5)	Controllers	7.1.3, 7.5.1, 7.5.2, 7.5.3, 7.5.5, 8.5.2	Start a fill, and record the commands the controller sends, then stop the controller's evaluations while its devices stay powered and connected. Repeat, but start a feed mode that stops the top-off during the fill before stopping the evaluations. Then bind the top-off to an output whose fail-safe configuration does not stop it.	One command starts each fill. In the first case, the output stops no later than <code>maxFillTime</code> plus 2 s after the fill started, and the output's device applies its fail-safe behavior once its <code>failSafeTimeout</code> has passed. In the second, the output does not run again when the feed mode would have ended. In the third, the controller does not run the top-off, and tells the owner what to change.
No fill during a water change (8.13)	Controllers	8.4, 8.13.1	Run a water change, and dry the top-off's normal sensor during each phase. Then abort a water change during its fill. Then abort a water change by dosing heads that drained 250 mL of a 200 L system. Then, with a top-off unit, stop the controller's evaluations during a water change.	No fill starts until the water change ends, or, after the first abort, until a person resumes or abandons it. After the second abort, the top-off fills again as its sensors ask. The top-off unit stays disabled after the controller stops.
Leak lockout (8.13)	Controllers, top-off units	8.13.2, 8.14.4, 11.2.3	With no emergency mode configured, wet a leak sensor of the system during a fill, then dry it. For a unit, use its own leak sensor, with no controller connected.	The output stops within 1 s. It stays stopped after the sensor is dry, until a person clears the lockout or <code>resetFault</code> is sent.
Wait for the return pump (8.13)	Configurations	8.13.3	With the normal sensor dry, stop the return pump, then start it again.	No fill starts while the return pump is stopped, nor until <code>delay</code> after the return pump reports running.

Test	Applies to	Requirements	Method	Pass criteria
Daily limit (8.9)	Controllers, top-off units	7.7, 8.9.2, 8.9.3, 8.14.9	With a short <code>minFillInterval</code> , make fills that each end at the level, until their total reaches <code>maxDailyFillTime</code> . Restart the controller or unit. Then wait until the first of the fills is more than 24 hours old.	The fill that reaches the limit stops within 1 s of reaching it, and <code>daily-limit</code> is raised. The limit still holds after the restart. No fill starts until enough fill time has left the 24-hour window, and fills then resume without a person.
Learned fill time (8.10)	Controllers, top-off units	8.10.1, 8.10.2, 8.10.4, 8.10.5	With <code>factor</code> 2 and <code>samples</code> 10, make 10 fills of 20 s that end at the level, then a fill that does not reach the level.	The last fill stops 40 s after it started, within 1 s, and <code>timeout</code> is raised and stays until cleared. A fill that ended by a limit or was interrupted does not change the learned fill time. The controller shows the learned fill time of 20 s.
Dry-run detection (8.11)	Controllers, top-off units	7.6.7, 8.11.2, 8.11.3	During a fill, lower the pump's power reading below the threshold. Then record a reservoir refill. Then clear the fault.	The output stops within <code>delay</code> plus 1 s, and <code>dry-run</code> is raised. No fill starts after the refill is recorded, until a person clears the fault; fills then resume.
Reservoir tracking (8.12)	Controllers, top-off units	8.12	Set the reservoir's capacity and the pump's flow rate, run fills that add up to the capacity, then record a refill. Separately, make the reservoir's empty condition true.	The estimate falls by each fill's volume, within 5 %. <code>low</code> is raised at the threshold. While the empty condition is true, no fill starts and <code>empty</code> is raised. Recording a refill resets the estimate to the capacity.
Holds of a top-off unit (8.14)	Controllers	8.14.5, 8.14.7, 8.14.10, 13.5.2	With a top-off unit whose <code>failSafeAction</code> is <code>continue</code> , which requires the return pump with a <code>delay</code> of 300 s, start a water change, and ask the controller, as a person, for a manual fill; then send the unit <code>fill</code> from a test tool; then stop the controller's evaluations while the unit stays powered. Repeat with a leak that locks the top-off out, and with a feed mode of 300 s that stops the return pump and the top-off. Then cut the unit's power while it is disabled for the water change, and restore it; and cut it during another feed mode, and restore it after the feed mode would have ended.	The controller does not send the manual fill, and says that the water change holds the top-off; the unit refuses the <code>fill</code> of the test tool. After the water change and the leak, the unit stays disabled, and does not fill however low the level falls, until the controller returns and a person resumes or abandons the water change, or clears the lockout. After the feed mode, the unit fills again when its revert falls due, 600 s after the mode started, the end of the return pump's pause plus the interlock's <code>delay</code> , within 2 s. After the power cut during the water change, the unit is still disabled; after the one during the feed mode, it fills again as its sensors ask.
Additive limits (8.16)	Controllers, configurations	7.7, 8.16.2, 8.16.3, 8.16.4, 8.16.7	Give a top-off with <code>flowRate</code> an additive whose <code>maxDailyVolume</code> is two and a half fills, and a <code>block</code> interlock on pH. Make fills that end at the level until the limit is reached. Then raise the pH above its limit during a fill, then make the pH reading unavailable. Then remove <code>flowRate</code> . Repeat the fills with a top-off unit that reports <code>volumeToday</code> . Then give that unit <code>failSafeAction</code> <code>continue</code> ; then <code>state</code> , with <code>failSafeState</code> <code>{"enabled": false}</code> , and stop the controller's evaluations while it fills.	The third fill stops when its volume reaches what the limit leaves, within 1 s, and <code>additive-limit</code> is raised, and not <code>daily-limit</code> ; no fill starts until older fills leave the 24-hour window. The fill stops within 1 s of the pH passing its limit, and no fill starts while the pH is high or unavailable. Without <code>flowRate</code> , the controller does not run the top-off, and tells the owner what to set. The controller disables the unit within 1 s of its <code>volumeToday</code> reaching what the limit leaves. While the unit's

Test	Applies to	Requirements	Method	Pass criteria
				<code>failSafeAction</code> is <code>continue</code> , the controller warns its owners; with <code>state</code> , the unit stops filling at its <code>failSafeTimeout</code> , and fills no more until the controller returns.
Paths without radio (8.17)	Controllers	8.17.3, 8.17.4	Bind a top-off's level sensors to devices reached by Thread, and its pump to an outlet reached by Wi-Fi whose <code>failSafeAction</code> is <code>state</code> , with its inactive state as <code>failSafeState</code> , and whose <code>failSafeTimeout</code> is 120 s. Then change the <code>failSafeTimeout</code> to 3600 s, and restart the controller. Then bind the pump to a module on the module bus. Then use a top-off unit reached by Wi-Fi.	With a <code>failSafeTimeout</code> of 120 s, no warning is shown. With 3600 s, before and after the restart, the controller runs the top-off, shows every owner a standing warning, which cannot be dismissed, naming the top-off and its output, and raises <code>safety-path-missing</code> ; within a minute of the change. On the module bus, and with the top-off unit, the warning ends.
Phase time limits (9.6)	Controllers	7.3, 9.6.2, 9.7.4	Hold the drain's end condition false. In a second water change, hold the fill's end condition false. Then, while idle, override a drain output to run for longer than <code>drain.maxTime</code> .	The drain stops at <code>drain.maxTime</code> , within 1 s, with <code>timeout</code> , and the fill starts. The fill stops at <code>fill.maxTime</code> , within 1 s, and the water change aborts, with <code>timeout</code> and <code>water-change-aborted</code> . The overridden output stops at <code>drain.maxTime</code> .
Backup low limit (9.6)	Controllers	7.3, 9.6.2, 9.6.3	During a drain, make <code>drain.limit</code> true. Then override a drain output to run while it is still true.	The drain stops within 1 s, <code>drain-limit</code> is raised, and the fill starts. The override does not run the output.
Backup high limit (9.7)	Controllers	7.3, 9.7.3	During a fill, make <code>fill.limit</code> true. Then request another water change.	The fill stops within 1 s, and <code>high-level</code> is raised and stays. The water change settles and ends. The next one is skipped until <code>high-level</code> is cleared.
Stop on a leak (9.11)	Controllers	9.6.4, 9.7.4, 9.11.1, 9.11.4	Make <code>leak</code> true, or start a leak emergency mode, during a drain, and again during a fill.	Every output of the water change stops within 1 s, and <code>water-change-aborted</code> is raised. No output of the water change runs until a person resumes it.
Resume without draining more (9.10)	Controllers	9.10	Remove the controller's power during a drain, then restore it. Repeat during a fill. Repeat both with a power-loss mode that does not list the water change's outputs, instead of a power cut, and with the clock not valid after the restart.	The power-loss mode stops the water change's outputs within 1 s of starting. After each interruption, no drain output runs. The fill runs until its end condition, and for no longer than the fill time it had left. The water change then settles and ends.
Unavailable inputs (9.6)	Controllers	9.6.2, 9.6.4, 9.7.4, 13.2	During a drain, make <code>drain.until</code> unavailable; in another, <code>drain.limit</code> . During a fill, make <code>fill.until</code> unavailable; in another, <code>fill.limit</code> .	In each drain case, the drain stops within 1 s, <code>input-unavailable</code> is raised, and the fill starts. In each fill case, the fill stops within 1 s and the water change aborts.
Conditions to start (9.4)	Controllers	9.4.3, 9.4.4, 9.12.4, 9.12.6	Request a water change: with <code>fill.limit</code> unavailable; with <code>leak</code> true; with <code>leak</code> unknown; with <code>fill.supplyEmpty</code> true; during an emergency mode; during a power-loss mode; with <code>high-level</code> latched; while another water change of the same system runs; with a dosing head without <code>maxDailyVolume</code> ; and with a dosing	In each case, no output of the water change runs, <code>water-change-skipped</code> is raised, and the water-change mode ends.

Test	Applies to	Requirements	Method	Pass criteria
			head bound since its last calibration.	
Runs end by themselves (9.6)	Controllers	7.5.1, 7.5.3, 9.6.1, 9.7.1	Start a drain, then stop the controller's evaluations while its devices stay powered and connected. Repeat during a fill.	The outputs stop no later than the phase's <code>maxTime</code> plus 2 s after the phase started.
Pauses (9.9)	Controllers	9.9	During a drain, start a feed mode that stops a drain output, then let it end. During a fill, override a fill output to stop for longer than <code>maxPause</code> . During a drain of 300 s with a drain output whose <code>failSafeTimeout</code> is 60 s, make its device lost 100 s into the drain, and restore it 200 s later. During another drain, stop a drain output from a home app through a Matter bridge, then send it <code>On</code> .	The drain resumes when the feed mode ends, and its time limit excludes the pause. The fill aborts <code>maxPause</code> after the override began, within 1 s. With the lost device, the drain counts its time until the controller takes the output to be stopped, about 60 s after its last contact, and then pauses, so that its outputs run for no longer in all than its <code>maxTime</code> . The home app's stop aborts the water change, with <code>water-change-aborted</code> , and its <code>On</code> neither resumes nor abandons it.
Volume balance (9.12)	Controllers	9.12.1, 9.12.3	Run a water change with dosing heads. Then run another and empty the new-water head's container during the fill.	The first fill delivers the volume the drain removed, within the heads' calibration. In the second, the fill stops, the head raises <code>empty</code> , and the water change aborts.
Dosing heads (9.12)	Controllers	7.5.4, 9.12.1, 9.12.2, 9.12.6, 9.12.7, 9.12.11	With heads whose <code>maxContinuousRun</code> is 60 s, one of which has the <code>local-program</code> feature and runs a program that an earlier configuration gave it, run a water change whose drain needs 150 s of running. Remove the controller's power during a drain dose, then restore it. Then make the fill head report <code>dosing</code> for 10 s with no dose sent.	Within a minute of the configuration, the head with <code>local-program</code> holds a program that runs no doses of its own and carries the local time zone, as Dosing Section 15.3.2 says. Each dose runs for at most 60 s, and the next starts no sooner than the previous one ran. The interrupted dose is not sent again, and the fill adds no more than the drain removed, within the heads' calibration. The controller sends the fill head <code>stop</code> within 1 s of the threshold, raises <code>stuck-on</code> , and the water change aborts; no water change starts, across a restart, until a person clears it.
New water matches (9.5)	Controllers, configurations	9.5.2, 9.5.3, 9.5.4, 9.5.6	Set the new water's salinity 2 psu above the system's, with <code>maxDifference</code> 1 psu. Then set its temperature 3 °C below, with <code>maxDifference</code> 2 °C. Then make the new water's salinity input unavailable. Then import a water change of a 200 L system without <code>check.temperature</code> , whose drain ends at <code>until</code> and has no <code>flowRate</code> ; and one whose drain has a <code>volume</code> of 8 L; and one whose drain has a <code>volume</code> of 2 L.	Each water change is skipped before any drain output runs: the first two with <code>new-water-mismatch</code> , the third with <code>input-unavailable</code> . The controller and the validator reject the first two imports, saying why, and accept the third.
Supply and waste ready (9.5)	Controllers	9.5.1, 9.6.4, 9.7.4	Request a water change with a <code>check.ready</code> condition false. Then, during a drain, make <code>fill.supplyEmpty</code> true; and again during a fill.	The first is skipped before any drain output runs. In the second, the drain stops within 1 s and the water change aborts, with <code>water-change-aborted</code> . In the third, the fill stops within 1 s,

Test	Applies to	Requirements	Method	Pass criteria
				empty is raised, and the water change aborts.
Maximum run time (10.5)	Controllers	7.3, 10.5.1	Keep the reservoir's full condition false.	The feed outputs stop within 1 s of <code>maxRunTime</code> after the run started, and <code>timeout</code> is raised. No run starts until it is cleared.
Backup high sensor (10.5)	Controllers	7.3, 10.5.2	During a run, make <code>high</code> true. Then override the feed valve to open while it is still true.	The feed outputs stop within 1 s, and <code>high-level</code> is raised and stays. The override does not open the valve.
Unavailable sensor means full (10.4)	Controllers	10.4.6	During a run, make <code>full</code> unavailable; in another run, <code>high</code> .	Each stops the feed outputs within 1 s and raises <code>input-unavailable</code> . No run starts until the input is available again.
Stop on a leak (10.4)	Controllers	10.4.4, 10.4.7, 11.2.3	During a run, make <code>leak</code> true, then false. In another run, make <code>leak</code> unknown. Then, with no emergency mode configured, wet a leak sensor of the system that <code>leak</code> does not use.	Each stops the feed outputs within 1 s. After each leak, no run starts until a person clears the lockout. While <code>leak</code> is unknown, no run starts.
Runs end by themselves (10.5)	Controllers	7.5.1, 10.5.4	Start a run, then stop the controller's evaluations while its devices stay powered and connected.	The feed outputs close no later than <code>maxRunTime</code> plus 2 s after the run started.
Feed closed without control (10.8)	Controllers	7.5.5, 10.8.1, 10.8.2	Bind a feed output whose fail-safe configuration does not stop it, then a feed valve whose <code>normallyOpen</code> is true. Then make the feed valve refuse to open while the booster pump is a feed output.	The controller does not run the function with the first, and tells the owner what to change. It warns the owner about the second. The booster pump does not run while the valve does not report open.
Daily limit (10.5)	Controllers	10.5.3	Make runs that each end at the full level, until their total reaches <code>maxDailyRunTime</code> .	The run that reaches the limit stops within 1 s of reaching it, and <code>daily-limit</code> is raised. No run starts until enough run time has left the 24-hour window.
Membrane flushing (10.6)	Controllers	10.6	Configure each kind of flush, and make a run that lasts longer than <code>interval</code> .	Each flush opens the flush outputs, and any rinse outputs, with the feed outputs running, for its time, within 1 s, and at the times configured. The run's limits still apply during flushes.
TDS monitoring (10.7)	Controllers	10.7	Raise the TDS reading above <code>max</code> in the first <code>delay</code> seconds of a run, then after them, with <code>stop</code> false and then true. Configure a rinse and start a run.	Nothing is raised during <code>delay</code> . After it, <code>tds-high</code> is raised; the run continues with <code>stop</code> false and stops with <code>stop</code> true. The rinse outputs open for <code>rinse.time</code> at the start of the run.
Leak lockout (11.2)	Controllers	7.3, 7.6.6, 11.2.3	With no emergency mode configured, wet a leak sensor of the system while a top-off fills, an RO/DI system runs, and a water change drains. Dry the sensor, then override each of their outputs to run, then clear the lockout.	Every output of the three functions stops within 1 s, and stays stopped after the sensor is dry, overrides included, until a person clears the lockout.
Leak response without radio (11.1)	Controllers	11.1.5, 11.1.6	With a leak emergency mode that starts on any of a system's leak sensors and stops a return pump on a module of the module bus, bind every leak sensor to devices reached by Wi-Fi or Thread, then restart the controller. Then bind	While every sensor is on radio, before and after the restart, the controller runs the configuration, shows every owner a standing warning, which cannot be dismissed, naming the system, the return

Test	Applies to	Requirements	Method	Pass criteria
			one of them to a module on the module bus. Then bind the return pump to an outlet reached by Wi-Fi whose <code>failSafeAction</code> is <code>state</code> , with its inactive state as <code>failSafeState</code> , and whose <code>failSafeTimeout</code> is 3600 s.	pump, and its leak sensors, and raises <code>safety-path-missing</code> . Once one sensor is wired, the warning ends. With the outlet on Wi-Fi, the warning returns, naming the return pump, and ends when its <code>failSafeTimeout</code> is 300 s or less.
Leak emergency (11.2)	Configurations	11.2.1, 11.2.2, 11.2.5	Wet a leak sensor. End the emergency mode while the sensor is wet, then after it is dry.	Within 1 s, each output takes the action of Section 11.3 for its equipment category, or the owner's configured action. Ended while the sensor is wet, the mode starts again at once. Ended after the sensor is dry, it stays ended.
Water type (11.4)	Controllers, configurations	11.4.1, 11.4.2, 11.4.3	In a marine system configured with the fresh-water response, wet a sensor with RO water, then with saltwater, then make its water type unavailable while it reports a leak. Then wet a sensor of a freshwater system with RO water.	RO water starts only the fresh-water response. Saltwater, and the leak of unavailable type, start the full response. In the freshwater system, RO water starts the full response.
Unavailable leak sensors (11.5)	Controllers	11.5	Open the wiring of a supervised leak sensor used in the <code>leak</code> of an RO/DI function and of a water change, then request each.	<code>wiring-fault</code> is raised. Neither the RO/DI function nor the water change starts. No emergency mode starts or ends because of it.
Overflow sensors (11.6)	Controllers, configurations	11.6.1, 11.6.2, 11.6.3	Wet a sump's overflow sensor during a top-off fill, then dry it. Wet a display's overflow sensor while the return pump runs, then dry it.	Within 1 s, the top-off stops, and for the display the return pump stops. Each stays stopped after the sensor is dry until a person clears the interlock. The sump's overflow sensor does not stop the return pump.
Heaters stay covered (12.2)	Configurations, controllers	12.2.1, 12.2.2, 12.2.3	While the heater heats, dry the low sensor of its vessel, then wet it again. Then make the sensor unavailable. Then remove the interlock from the configuration.	The heater stops within 1 s of the sensor drying. It does not start again until the sensor has been wet for the delay. While the sensor is unavailable, the heater stays stopped. Without the interlock, the controller warns the owner.
Pumps stay covered (12.3)	Configurations	12.3	While the return pump runs, dry the low sensor of the sump, then wet it again. Then make the sensor unavailable.	The return pump stops within 1 s of the sensor drying, and starts again once the sensor has been wet for the delay. While the sensor is unavailable, the return pump runs and <code>input-unavailable</code> is raised.
Pump dry-run detection (12.4)	Controllers, configurations	12.4.1, 12.4.2	Make a running pump report <code>dry-run</code> , then clear it. Separately, configure a monitored pump as the example of Section 12.4 does, and lower its power below the threshold while it runs, and keep it low.	Each pump stops within 1 s, or within the on-delay plus 1 s, and starts again no sooner than the configured delay. The monitored pump runs again when its off-delay ends, and stops again within the on-delay plus 1 s while its power stays low.
Low-water alarm (12.5)	Controllers	9.13.2, 9.13.3, 9.13.4, 12.5.1, 12.5.3, 12.5.4	Dry a low sensor. Then dry it while an automatic water change drains, and keep it dry after the water change ends. Repeat, and abort the water change during its fill.	The first raises <code>low-water</code> after its delay, and it sounds. During the automatic water change, <code>low-water</code> is raised and shown, but neither sounds nor is

Test	Applies to	Requirements	Method	Pass criteria
			Repeat, and start a power-loss mode during the drain. Then dry it during a water-change mode without a water change function. Then open the wiring of a supervised low sensor.	notified, and the low-water interlocks still stop the heater; when the water change ends, or aborts, or the power-loss mode starts, with the sensor still dry, it sounds and is notified at once. The third raises <code>low-water</code> , and it sounds, after its delay. The open wiring neither raises nor clears <code>low-water</code> .
Invalid configurations	Controllers, configurations	7.2, 7.2.4, 7.2.5, 7.2.6, 7.2.7, 7.8.1, 8.5.3, 8.6.3, 8.7.1, 8.9.1, 8.10.3, 8.14.6, 8.15, 9.2.1, 9.2.5, 9.4.1, 9.5.2, 9.5.6, 9.12.5, 12.4.2	Import configurations that each break one rule for documents: a function whose vessel belongs to another system; an output in a role but not in <code>outputs</code> ; a function with <code>onUnavailable</code> ; a reference to a number; a function that fails its schema; a volume in a unit of flow; a top-off without <code>maxFillTime</code> ; a <code>maxFillTime</code> or <code>minFillInterval</code> of 0; a top-off on a controller without <code>high</code> ; a <code>maxDailyFillTime</code> less than <code>maxFillTime</code> ; a <code>learning.factor</code> of 1; a top-off unit's function with <code>level</code> ; a drain without <code>until</code> , <code>volume</code> , or <code>time</code> ; a water-change mode that lists its water change's output; a water change that follows a feed mode, and two that follow one mode; a salinity check whose inputs measure different parameters; a water change that can remove more than 5 % of its system's water without a temperature check; a fill of dosing heads without <code>until</code> or <code>limit</code> ; and a <code>block</code> interlock on a pump whose condition compares its power without requiring that it runs.	The controller and the validator reject each one, saying why, and the running configuration is unchanged.
Visibility	Controllers	7.6.4, 7.11	With a top-off that has learning, an additive, and a reservoir, a water change, and an RO/DI system, let each run, latch a fault of each, and start a water change that holds <code>low-water</code> .	The controller shows, for each function, each item of Section 7.11, and, for each latched fault, the function, the fault, why it latched, and when.
Saved state	Controllers	7.9.1, 7.9.3	While a top-off has fills in the last 24 hours, a water change fills by dosing heads, and an RO/DI system has a latched fault, read the saved state every second, then remove the controller's power, restore it, and export the configuration. Import the export on another controller.	Each function's saved state is in <code>state.functions</code> , under its identifier, and valid against <code>\$defs/savedState</code> , changes within 1 s of each change of the functions, and, while the water change fills, at least every 10 s and after each dose. After the restart, and on the other controller, each function continues from its saved state.
Top-off unit description	Top-off units	8.14.1, 8.14.2	Read the top-off unit's device description through each binding it supports.	It has <code>level-switch</code> endpoints whose <code>position</code> is <code>normal</code> and <code>high</code> , and an <code>auto-top-off</code> endpoint that implements <code>minFillInterval</code> .
Settings of a top-off unit	Controllers	8.14.8, 8.14.9	Change a top-off function's <code>maxFillTime</code> , <code>minFillInterval</code> , <code>reservoir.capacity</code> , <code>startDelay</code> , <code>overrun</code> , <code>maxDailyFillTime</code> , and	Within a minute, each unit's attributes hold the new values of every setting it implements. For the unit of revision 0.1, the

Test	Applies to	Requirements	Method	Pass criteria
			Learning, first with a unit of Device Classes revision 0.2 that implements them all, then with a unit of revision 0.1. With the second, make fills that reach <code>maxDailyFillTime</code> .	controller tells the owner that the unit uses its own start delay, overrun, and learning settings, and disables the unit when its fills reach <code>maxDailyFillTime</code> , until the window allows a fill.
Mode ended early	Controllers	9.4.5, 9.4.8	With a water-change mode that stops the heater and the return pump, end the mode by hand during the drain of its water change. Then repeat, and stop the controller's evaluations during the fill, while its devices stay powered and connected.	The drain stops within 1 s, and the fill starts. The heater and the return pump stay stopped until the water change returns to idle, and start again only then. With the evaluations stopped, they stay stopped until the fill time that was left, and <code>maxPause</code> , have passed, and no longer than that plus 2 s.
Firmware updates	Controllers	13.4.1	Offer the controller a firmware update, and schedule a restart, while a water change drains, while it fills, and while an RO/DI system makes water.	The controller starts neither the update nor the restart until the run has finished, or a person who has been asked confirms it.
Imported configurations	Controllers	16.6	Import a configuration with a top-off that carries an additive, a water change, and an RO/DI function.	Before it runs any of them, the controller shows the owner the limits of each function, including the additive's daily limit, and runs them only once the owner confirms.

16 Security Considerations

- Water functions can flood a room.** Changing their configuration, including their limits, needs the `change-configuration` operation, and changing a top-off unit's limits needs `change-device-settings` [OAP-SA Table 6], both of the `full-control` role.
- Operating them.** Starting or ending a water change needs `start-mode`, a manual fill of a top-off unit `run-action`, and clearing a latched fault, or resuming or abandoning a water change, `clear-latch`, all of the `override` role. A manual fill of a top-off unit whose water carries an additive adds a chemical, so it needs `dose-now`, of the `full-control` role, as a manual dose does, rather than `run-action`, which runs only actions that add no chemicals [OAP-SA Table 6]. Priming a water change's dosing head needs `dose-now`, and calibrating it `calibrate`, both of the `full-control` role. Recording a reservoir refill, which changes no output, needs `record-maintenance`, of the `override` role [OAP-SA Table 6]. A held top-off unit refuses a manual fill, and a controller sends none to it (Section 8.14.10), so no operation gets around a hold.
- Clearing faults from afar.** A latched fault usually means water went somewhere it should not. A person who clears it remotely restarts water flowing without anyone having seen the system. A controller SHOULD say so before a remote user clears a latched fault, and SHOULD ask the user to confirm that someone has checked the system.
- Radio links.** A radio link is never the only path for a safety function [P0001] [OAP-FS Section 13.4]. Top-off shutoff and leak response are safety functions, and so are the low-water interlocks that stop heaters [OAP-AU Section 13.7]. A controller checks each for a path that uses no radio link, as Fail-Safe Behavior Section 13.4 judges one, and warns its owners while one has none, as Automation Section 13.7 and Sections 8.17 and 11.1.6 require. A backup sensor SHOULD be wired to the device whose output it stops, as on a top-off unit, or reach the controller over the module bus [OAP-MB]. An output reached by radio is protected by `revertAfter` and its own fail-safe behavior if the link fails (Section 7.5).
- False readings.** Anyone who can forge a level or leak reading can start or stop water flowing. Network bindings authenticate devices and their messages [OAP-SA Section 10.3], and the backup sensors and time limits bound what a forged reading can do.

6. **Imported configurations.** A configuration from another owner can set limits that do not suit the system it is imported into. A controller **MUST** show the owner the limits of every imported top-off, water change, and RO/DI function before running it, including an additive's daily limit, as Automation Section 21, item 2, requires for imported automation.
7. **Privacy.** Water change schedules, refills, and manual actions show when someone is at home, and **SHOULD** be treated as the owner's data.

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 passed the test procedures in Section 15.5, including a top-off unit and a controller.	Not started	
Every normative reference is publicly available.	Done	
The checks file, schema, and examples are published, and the examples pass the validator.	Done	
Experienced aquarists have reviewed the leak responses and the recommended values.	Not started	
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 — Examples (Informative)

The `examples` directory contains complete Automation configurations of revision 0.3, with the function types and saved state of this revision:

- `reef-sump.json`: a reef system with a sump; a top-off on a controller with a normal and a backup high sensor, a daily limit, learning, dry-run detection, and a tracked reservoir; a water change by pumps that checks the new water's salinity and temperature; an RO/DI system with a booster pump, flushing, rinsing, and TDS monitoring; leak emergency modes for the cabinet, with a smaller fresh-water response, and for the RO/DI system; an overflow interlock on the display; low-water interlocks for the heater and the return pump; and dry-run detection on the return pump from its power.
- `top-off-unit.json`: a nano reef with a top-off unit of Device Classes revision 0.2, which the controller enables and configures, including its start delay and daily limit, held by its return pump and stopped by a leak.
- `kalkwasser-top-off.json`: a reef whose top-off carries kalkwasser, with a daily volume limit and a pH interlock.
- `dosing-water-change.json`: a freshwater system with small water changes every two hours by a two-head doser, whose doses fit the heads' `maxContinuousRun`, whose phases' time limits allow for the rests between the doses, and which are too small to need checks of the new water; and saved state of a water change interrupted during the second part of its fill, with the dose in progress.
- `minimal.json`: one top-off on a controller.

Appendix C — Commercial Features (Informative)

How the features of four widely used commercial controller ecosystems correspond to this Specification.

Table 22 — Commercial features

Feature	In this Specification
Top-off with timeouts, a backup high sensor, and alarms	Sections 8.5 to 8.7.
Top-off that learns the normal fill amount	Learned fill time, Section 8.10.
Pump dry-run detection from power draw	Sections 8.11 and 12.4.
Broken level sensor treated as “level reached”	Section 8.8, for every function in Section 7.4.
Automatic water change: drain, fill, resume after power loss, stop on leak	Section 9, especially Sections 9.10 and 9.11.
Water changes using dosing pumps	Section 9.12.
RO/DI management: flush, reservoir control	Section 10.
Reservoir level forecast	Section 8.12.
Leak response: emergency mode, top-off lockout	Sections 11.2, 11.3, and 8.13.
Leak sensors that detect RO water as well as saltwater	Section 11.1; telling them apart, where a sensor can, Section 11.4. None of their makers documents doing so.

Appendix D — Open Issues (Informative)

Issues 1 to 21 were open in revision 0.1; the Project Lead decided issues 22 to 24 and 28 on 15 October 2026, and issues 1 to 7 and 21 on 16 October 2026. Issues 25 to 27 arose in revision 0.2, and the Project Lead decided them on 16 October 2026, accepting the recommendation in each. On 17 October 2026 the Project Lead froze the scope of revision 1.0, and decided issue 12, and the code of issue 26; revision 0.3 applies those decisions and the decisions of issues 2 and 28, and resolves issue 8, which another specification settled. Issue 29 records the choices that revision 0.3 made where those decisions left them open, which the Project Lead accepted on 18 October 2026. Issues 14 to 20 are deferred beyond revision 1.0, and are tracked outside this Specification. Issue 13 stays open until the bench tests and owners’ experience confirm its numbers. Each issue says how a revision resolves it, or why it stays open. Issue numbers do not change, because other specifications refer to them.

Questions for the Project Lead, decided on 16 October 2026

- Values that depend on the system.** *Decided by the Project Lead on 16 October 2026; resolved in revision 0.2: no defaults.* `maxFillTime`, `minFillInterval`, `maxDailyFillTime`, the water change phase limits, and `maxRunTime` depend on the size of the system, its sump, its pumps, and its evaporation, so this revision gives no defaults, and a controller must not run a function until the owner has set them (Sections 8.15, 9.14, and 10.9). The alternative is a table of conservative defaults by system volume. Defaults are convenient, but a default that suits a 400 L system with a large sump can overflow a nano’s sump in one fill. The Project Lead decided on no defaults, and guided setup from the vessel’s dimensions and the pump’s measured flow, which Section 8.15 now names.
- Matching new water.** *Decided by the Project Lead on 16 October 2026; resolved in revision 0.3.* Checking the new water’s salinity and temperature before draining was RECOMMENDED (Section 9.5), provisionally. Making it REQUIRED would protect livestock from a badly mixed batch, but rules out automatic water changes for owners without probes in their new-water reservoir, and dosing water changes whose volumes are too small to matter. The Project Lead decided that the checks stay RECOMMENDED in revision 0.2, and that the next revision makes them REQUIRED for water changes that replace more than 5 % (provisional) of the system’s volume, so that owners have time to add probes to their new-water reservoirs. Revision 0.3 does so in Section 9.5.6: every water change has both checks, unless the most its drain can remove is known to be no more than 5 % of the system’s water, so that a water change whose volume is unknown, such as one that drains to a level sensor, has them too.

3. **A return pump with an unknown sensor.** *Decided by the Project Lead on 16 October 2026; resolved in revision 0.2: kept.* The low-water and overflow interlocks on a return pump allow it to run while their sensors are unknown (Sections 11.6 and 12.3), because a return pump is vital equipment, at the risk of the pump running dry if the sump empties meanwhile, or of filling a display whose drain is blocked. The alternative is to stop it, as for heaters. The Project Lead decided to keep it running, with an alarm, as this revision says.
4. **Fresh-water leaks.** *Decided by the Project Lead on 16 October 2026; resolved in revision 0.2.* An owner may give fresh-water leaks in a marine system a smaller response (Section 11.4). The alternative is to give every leak the full response, and use water type only to inform. The Project Lead decided to allow it as an owner's choice, off by default (Section 11.4.1).
5. **Single-sensor top-offs.** *Decided by the Project Lead on 16 October 2026; resolved in revision 0.2: kept.* A backup high sensor is REQUIRED (Section 8.7). Some small top-offs have only one sensor. Should they be allowed with stricter limits, such as a short `maxFillTime` and a daily limit, or excluded? The Project Lead decided to keep the backup sensor required, because a sensor that sticks, or grows a film of algae, is a common fault of a top-off (Section 6.1).
6. **Top-off after an aborted water change.** *Decided by the Project Lead on 16 October 2026; resolved in revision 0.2.* The top-off stays stopped until a person resumes or abandons an aborted water change (Section 8.13), so that it does not replace saltwater with fresh water. For a small dosing water change, the shortfall is too small to matter, and stopping the top-off for hours does more harm. The Project Lead decided that the hold applies unless the volume the water change drained is known and is no more than 1 % (provisional) of the system's water volume (Section 8.13.1). Replacing that much with fresh water lowers a marine system's salinity by less than the new-water check allows, and a leak still locks the top-off out whatever the shortfall (Section 11.2.3).
7. **Severity of low-water.** *Decided by the Project Lead on 16 October 2026; resolved in revision 0.2: kept.* `low-water` is `critical`, because it stops heaters and the return pump (Table 5). Owners who mount a low sensor as an early warning would prefer `warning`. From revision 0.2, an automatic water change holds the alarm's sound, and never drops it (Section 9.13), so a water change that drains below the low sensor no longer argues for a lower severity. The Project Lead decided to keep `critical` as the default, which an owner can lower to `warning` but not turn off (Section 12.5.5), and to have the severities of every new code reviewed by experienced aquarists with the others, as Alarms and Notifications Appendix D, issue 2, asks.

For other specifications

8. **Automation.** *Resolved in revision 0.3: the Automation specification's revision 0.3 lists the function types `top-off`, `water-change`, and `ro-di`.* The Automation specification was to list `top-off`, `water-change`, and `ro-di` among its function types (its Section 11.10), in place of the provisional extension names of Section 7.2; give functions a member of the saved state, in place of `state.extensions["org.openaquariumproject.water-level"]` (Section 7.9); add the checks that act on every run (Section 7.3) to its precedence; say that the water change function sends `dose` commands to the heads it drives; describe a water-change mode followed by a function, and allow water-change modes of different systems to be active together (Section 9.4); allow the stops that end these functions' runs, and holds of top-off units, without `revertAfter` (Sections 7.5.3 and 8.14.7); and point its presets to these types. Its revision 0.3 does each, as the Project Lead decided on 17 October 2026 (its Sections 8.3, 11.10, 14.3.3, and 17.8, and Tables 25 and 28), and this revision uses the function types and `state.functions["<function id>"]` (Sections 7.2 and 7.9), reading the names of revision 0.2 as well (Section 7.2.8).
9. **Device Classes.** *Resolved in revision 0.2.* Device Classes revision 0.2 made `minFillInterval` REQUIRED; added `startDelay`, `overrun`, `maxDailyFillTime`, `learningFactor`, and `learningSamples`; made a unit report `leak` when its own leak sensor locks it out, and `daily-limit` at its daily limit; and gave `level-switch` the positions `drain` and `reservoir-empty` (its Section 10.1 and Appendix D, issue 17). Section 8.14 and Table 8 now use them: the controller writes the settings a unit implements, and a unit that implements `maxDailyFillTime` enforces it itself (Section 8.14.9). Section 7.4.6 names the level-switch positions. A water changer class stays deferred until self-contained products exist (Device Classes Appendix D, issue 27).

10. **Alarms and Notifications.** *Resolved in revision 0.2.* This Specification's codes are in `conditions.json` (Table 5), and Alarms and Notifications revision 0.2 holds the conditions that a water change makes expected, as Section 9.13 names them, without dropping them (its Section 12.8.4). Functions of this Specification raise Device Classes codes, such as `timeout`, with the function as their subject (Section 7.10).
11. **Data Model.** *Resolved in revision 0.2.* Data Model revision 0.2 records a day's top-off fills as one event with `occurrences`, a reservoir's refill as a `refill` event, and an RO/DI run as an `ro-di-run` event, with the water sent to the drain as `wasted` (its Sections 8.2 and 8.6, and Appendix C, issue 18). Section 9.15 now uses them.
12. **Security and Access.** *Resolved in revision 0.3, as the Project Lead decided on 17 October 2026.* Recording a reservoir refill used `run-action`, provisionally, and a dedicated operation, or treating it like a calibration, might fit better (Security and Access Appendix D, issue 29). The Security and Access specification's revision 0.3 adds the operation `record-maintenance`, for records that change no output, which this revision uses (Section 16.2).

Details of this Specification

13. **Numbers.** *Deferred until the bench tests of the breadboard prototype, and owners' experience, confirm them.* The start delay, the overrun, the high-sensor window, the learning factor, the dry-run delay and threshold, the settle time, the maximum pause, the new-water tolerances, the TDS delay, the restart delays, and the `low-water` delay need confirming on the breadboard controller and with owners. So do the numbers that revision 0.2 adds (issue 27), among them the 5 % of a system's water above which revision 0.3 requires the new-water checks (Section 9.5.6). Revision 0.3 adds no numbers of its own; the `failSafeTimeout` of 300 s that gives a top-off or a leak response a path without radio is Fail-Safe Behavior's, provisional there (Sections 8.17 and 11.1.5).
14. **Certification.** *Decided by the Project Lead on 16 October 2026; resolved: yes.* Should passing the test procedures of Section 15.5 be required for a future compatibility mark? The project's governance says that a compatibility program, if created, will require conformance testing (its Section 5), and the tests of Table 21 are those of this Specification, as Fail-Safe Behavior Appendix C, issue 7, says of its own. Which products may carry a mark, and how their results are checked, belong to the program, not to this Specification.

Decided by the Project Lead

22. **Wired leak sensors.** *Resolved in revision 0.2.* A leak sensor wired to the controller stays a recommendation (Section 11.1.4). While every leak sensor of a system is on a radio link, a controller shows its owners a standing warning, and does not refuse the configuration (Section 11.1.5), as Fail-Safe Behavior Section 13.4 requires of every safety function. On 17 October 2026 the Project Lead decided that the paths of top-off shutoff and leak response are judged as Fail-Safe Behavior Section 13.4 judges every path, with the `failSafeTimeout` of at most 300 s (provisional) of its revision 0.4; revision 0.3 does so, for each output that a top-off or a leak response stops, and raises `safety-path-missing` while one has none (Sections 8.17, 11.1.5, and 11.1.6).
23. **Expected conditions during a water change.** *Resolved in revision 0.2.* Section 9.13 names the only condition that a water change makes expected, `low-water`, and the phases in which it is expected: draining, filling, and settling. Its alarm is held, never dropped, and sounds as soon as the water change ends or aborts (Sections 9.13.3 and 12.5.3), as Alarms and Notifications Section 12.8 requires, and, from revision 0.3, as soon as an interruption, such as a power-loss mode, holds it. The low-water interlocks act throughout.
24. **Additives through a top-off.** *Resolved in revision 0.2.* A top-off function declares its additive, such as kalkwasser, with `additive` (Section 8.16). A daily volume limit, `maxDailyVolume`, which the controller enforces as it does `maxDailyFillTime`, and a `block` interlock on pH, with `onUnavailable block`, are RECOMMENDED, and a controller warns about a top-off with an additive that lacks either. The sizes of the limits are issue 26. Only the controller enforces those limits, so a top-off unit that carries an additive and follows `continue` dropped them when it lost its controller. On 17 October 2026 the Project Lead decided that such a unit is recommended `failSafeAction state`, with `enabled false`, and that a controller

warns about one configured to `continue`; revision 0.3 does so (Section 8.16.7 and Table 18), as Fail-Safe Behavior Table 3 does.

The following issues arose in revision 0.2:

25. **The leak lockout and dosing.** *Decided by the Project Lead on 16 October 2026; resolved in revision 0.2: kept.* The leak lockout holds the outputs of every top-off, water change, and RO/DI function of a system until a person clears it (Section 11.2.3), but not dosing heads that a dosing function drives [OAP-DO Section 13.2]. A leak emergency mode stops them, as Table 15 recommends, and a controller warns about a leak sensor that starts no emergency mode (Section 11.2.1). Locking out dosing too would stop it even where the owner has configured no emergency mode, but every leak would then miss doses until a person clears the lockout, though dosing heads add little water. The Project Lead decided to keep the lockout to the equipment that moves water, as Section 11.2.3 now says.
26. **Sizing an additive's limits.** *Decided by the Project Lead on 16 October 2026; resolved in revision 0.2: the proposal stays, provisional until experienced aquarists have reviewed it; its code decided on 17 October 2026, and resolved in revision 0.3.* Section 8.16 proposes, provisionally, a `maxDailyVolume` that changes the additive's parameter by no more than Dosing Table 13 allows in a day, or twice the water the system loses on its heaviest day if that is smaller, and a pH limit of 8.45 for kalkwasser. For saturated kalkwasser, the first is about 1.2 % of the system's water a day, which is within the 0.5 to 2.5 % a day that systems are reported to evaporate, so in a dry season the top-off of a system that evaporates more stops at its limit while its salinity rises, until the owner dilutes the kalkwasser. A larger limit keeps the salinity steady, but lets a fault add more kalkwasser before the top-off stops. The Project Lead kept the proposal of Section 8.16, provisional until experienced aquarists have reviewed it, as Appendix A requires before approval. The limit raised `daily-limit`, which is critical for hazardous equipment and protected, and says that water may be leaving the system, so in a dry season sounded a critical alarm every day. On 17 October 2026 the Project Lead decided that the additive's limit raises a code of its own, `additive-limit`, `warning` by default and critical for no equipment group, and that `daily-limit` stays for limits that point to a leak; revision 0.3 does so (Sections 7.10 and 8.16.2, and Table 5).
27. **Numbers that revision 0.2 adds.** *Decided by the Project Lead on 16 October 2026: the values stay, provisional, and are deferred with issue 13.* Revision 0.2 adds these provisional numbers: a rest after each dose of a water change's dosing head as long as the dose ran, and a `maxDailyVolume` of 1.5 times a day's water changes for those heads (Section 9.12); the pH limit of 8.45 and the figures for saturated kalkwasser (Section 8.16); the 1 % of a system's water below which an aborted water change does not hold the top-off (Section 8.13.1); and the 5 % of a system's water above which the next revision requires the new-water checks (Section 9.5.5).

The following issue arose from another specification after revision 0.2:

28. **A home app's stop during a water change.** *Decided by the Project Lead on 15 October 2026; resolved in revision 0.3.* Section 9.9 paused a water change while an override holds one of its outputs, and a home app's stop is an override, so a home app's later `On` would let the paused water change go on, after a person had stopped it for a reason. The `Matter Mapping` specification [OAP-MM] therefore aborts a water change when a home app stops an output of its `drain.outputs` or `fill.outputs`, from the start of its drain to the end of its settling, so that only a person in an Open Aquarium app resumes it (its Section 12.11; its Appendix D, issue 24). Revision 0.3 says so in Section 9.9.3, beside an emergency mode, which also aborts rather than pauses, and among the causes of an abort in Sections 9.6.4 and 9.7.4. Raised by the Matter Mapping specification (Appendix D, issue 24).

The following issue records the choices of revision 0.3:

29. **Choices of revision 0.3.** *Resolved in revision 0.3: the Project Lead accepted each choice on 18 October 2026.* Revision 0.3 made these choices where the decisions it applies left them open:
 - a water change checks its new water's salinity and temperature unless the most its drain can remove, the smallest of its `volume`, and its `time` and `maxTime` each times its `flowRate`, is known to be no more than 5 % of the system's water, so that a drain to a level sensor without a flow rate, whose volume is unknown, needs the checks too (Section 9.5.6);

- a water-change mode that ends before its water change finishes keeps its stops, at its precedence, until the water change returns to idle or aborts, sent with `revertAfter` for the fill time left and `maxPause`, rather than leaving heaters and return pumps to the low-water interlocks (Section 9.4.8), and a mode's `duration` allows a `maxPause` for each of the drain and the fill (Section 9.4.6);
- a phase counts its time while the state of a lost output is unknown, a drain as having drained nothing and a fill as having filled at its flow rate, and pauses only once the output is taken to be stopped (Section 9.9.4);
- a system's leak lockout is part of its leak response, whose path without radio is judged for each output it stops (Section 11.1.5);
- a top-off unit that does not report `volumeToday` may add up to one fill more than an additive's limit, which the controller tells the owner, proposing a limit smaller by one fill (Section 8.16.3);
- a manual fill of a top-off unit whose water carries an additive needs `dose-now`, as a manual dose does (Section 16.2); and
- a top-off unit whose water carries an additive keeps `last` after power-up, as Fail-Safe Behavior recommends for top-off units, so that it fills without its controller only until its `failSafeTimeout` (Table 18).