Functional Requirements for an Open Source Overlay Journal Platform utilising COAR Notify

Version 1 - November 20, 2025

The document was produced as part of the COAR Notify Project, which is led by COAR and funded by Arcadia.

Acknowledgments

These requirements were developed based on a number of interviews with journal editors conducted in 2023-2024 by the COAR Notify team members, Eloy Rodrigues and Kathleen Shearer. They were originally compiled by Adam Hyde at Kotahi and have been adapted with input from the COAR Notify Team. Our thanks to the journal editors who participated in the interviews and Colleen Cressman from Harvard Library for reviewing the requirements.

Introduction

This document outlines the core functional requirements for a software platform designed to support the workflows of an overlay journal. The guiding principle is that an overlay journal adds a layer of review, curation, and endorsement on top of research artefacts that are hosted and maintained in external repositories. The platform, therefore, must excel at communication and interoperability, treating the external repository as the authoritative source for the research artefact itself.

The requirements are platform-agnostic and are categorised into essential functions (**must** have) for an overlay model, strongly suggested (**should** have) and optional features (**may** have).

Three possible workflows are considered (in descending order of desirability):

1. The repository initiates the process based on one of its own pre-prints, by sending a request to the overlay journal.
2. The Overlay journal initiates the process based on a pre-print in an open-access repository.
3. The overlay journal initiates the process based on a pre-print which it has received directly, and which has not yet been deposited in any open-access repository. If the overlay journal decides to review the preprint, then its first action is to to deposit it into an open-access repository, and then follow workflow 2.

1. Core Workflow Management The platform must provide a comprehensive system for managing the entire lifecycle of a review process, from initial receipt of a request to the final editorial decision.
1.1 Submission Triage: Editors **must** have an interface to view, assess, and assign all incoming review requests.
1.2 Role Assignment: The system **must** allow editors to assign roles (e.g., Handling Editor, Reviewer) to different users for each submission.
1.3 Peer Review Management: The platform **must** facilitate the peer review process, allowing reviewers to access submission materials (via links to the repository) and submit their evaluations through a dedicated interface.
1.4 Decision Making: Editors **must** be able to record editorial decisions (e.g., Accept, Reject, Revise) based on reviewer feedback.
1.5 Configurable Workflows: Administrators **must** be able to define and customise editorial workflows to suit the journal's specific practices.
1.6 Reminder notices: The system **may** be able to send a reminder to reviewers if the preprint has not been reviewed after a certain period of time.

2. Interoperability, Communication, and Versioning

The platform must be able to communicate seamlessly with external systems, such as institutional and preprint repositories, using COAR Notify (and other standard protocols if/when necessary and suitable).

2.1 Receiving Review Requests: The system **must** have a designated inbox to receive and process incoming Offer notifications (requests for review) from external repositories. Processing **must** entail validating the provided URI to the version of the artefact subject to review.

2.2 Acknowledging Submissions: The system **must** be able to send acknowledgement notifications back to the originating repository, including:

- TentativeAccept: To confirm a submission has been accepted into the review process.
- Reject: To inform the repository that a submission has been rejected and no further action will be taken.
- TentativeReject: To handle "revise and resubmit" scenarios.

2.3 Announcing Outcomes: Upon completion of the editorial process, the system **must** automatically send Announce notifications to relevant external systems. The notification **must** refer to the proper version of the artefact that was subject to review. This includes support for various outcomes:

- **Announcing a Review:** Notifying that a peer review is complete and available.
- **Announcing an Endorsement:** Notifying that a submission has been formally endorsed by the journal.
- **Announcing a Relationship:** Declaring a formal link between a published review and the preprint it evaluates.

2.4 Handling Retractions: The system **must** be able to process Undo notifications from a repository to retract a submission

2.5 Versions: The platform **must** identify and process specific versions of repository artefacts. Any resource created by the system which pertains to a repository artefact must link to the specific version of that artefact.

3. Submission and Intake

The Platform **must** support one of the three artefact intake workflows mentioned in the introduction above the table:

1. The repository initiates the process based on one of its own pre-prints, by sending a request to the overlay journal.
2. The Overlay journal initiates the process based on a pre-print in an open-access repository.
3. The overlay journal initiates the process based on a pre-print which it has received directly, and which has not yet been deposited in any open-access repository. If the overlay journal decides to review the preprint, then its first action is to to deposit it into an open-access repository, and then follow workflow 2).

3.1 Primary Intake Channel (Repository initiated): The platform **may** support the primary intake workflow outlined above. If it does, it **must** receive review requests via the COAR Notify Protocol.

3.2 Secondary Intake Channel (Overlay Journal initiated): The platform **may** support the secondary intake workflow outlined above. If it does, it **may** facilitate the workflow by providing a direct submission interface for authors.

3.3 Tertiary Intake Channel (Overlay Journal initiated): The platform **may** support the tertiary intake workflow outlined above. If it does, it **may** facilitate the workflow by providing a direct submission interface for authors

4. Metadata and Configuration The platform must be flexible in its handling of metadata and allow for deep, journal-specific configuration.
4.1 Metadata Retrieval: The system **must** be able to fetch rich metadata for a submission, via HTTP, from the repository that is hosting the artefact under consideration for review. The repository may decide how to make this metadata available, but the overlay journal may assume that the metadata is machine-readable and available via HTTP.
4.2 Discovering Metadata for Retrieval via *signposting*: The system **must** be able to determine the source of rich, machine-readable metadata for a given repository resource via *signposting*, if the repository supports this.
4.3 Discovering Metadata for Retrieval via other means: If the repository does not support *signposting*, then other mechanisms **may** be used, including (but not limited to) the interrogation of metadata associated with a PID (e.g. ARKs, DOIs etc), or some direct repository API.
4.4 Flexible Metadata Mapping: The system **must** allow administrators to define a flexible data model for submissions and map incoming metadata fields from external sources to this internal model. This mapping must be configurable per journal and not hardcoded.
4.5 Configurable Form Builder: The platform **should** provide a user-friendly, graphical form-building tool for administrators to create and customise all forms used in the workflow (e.g., submission, review, and decision forms).

4.6 Secure Authentication: The system **must** support secure authentication for establishing trusted communication with partner systems.

5. Administration and Monitoring

The platform must provide administrators with the tools needed to manage the system, monitor its health, and troubleshoot integration issues.

5.1 Communication Monitoring: Administrators **must** have access to a dashboard that provides a searchable, real-time log of all incoming and outgoing machine-to-machine messages. This interface should display message status (e.g., processed, error), raw payloads, and any processing errors.

5.2 System Policies: The platform **should** allow administrators to configure system-level policies, such as setting the default open content license (e.g., CC BY 4.0) for all published reviews and editorial materials.

5.3 Graceful Error Handling:

- When an incoming notification is malformed, the system **must** automatically send a structured error message back to the sender.
- If a valid notification fails to process internally, the system **must** preserve all incoming data and place the submission in a recoverable state for manual intervention.

6. Dissemination of Review and Curation Outputs

The platform must support the publication and dissemination of its own editorial and review outputs, ensuring they are clearly linked back to the original research artefact.
6.1 Publishing Review Materials: The platform **must** have a mechanism to publish review materials, endorsements, decision letters, and other curated editorial content to a public-facing web page.
6.2 Journal Policies: The platform **must** support the public posting of information about the journal's policies (as per DOAJ requirements)
6.3 Linking to Source: All published review content **must** prominently and as persistently as possible link back to the specific version of the reviewed research artefact in its source repository.
6.4 Closed Reviews for Authors: The platform **may** provide a mechanism for authors to read their reviews that are not made public, either because they delay publishing reviews until later or do not publish the reviews
6.5 Generation of Review Outputs: The platform **may** provide tools to generate formatted outputs of its own processes, such as downloadable PDFs, of peer review reports or signed endorsement statements.

7. User Management
7.1 Internal Team Management: The system **must** provide tools for administrators to create and manage user accounts and permissions for the internal editorial team (e.g., Administrators, Editors-in-Chief, Handling Editors, Reviewers).

7.2 Author Accounts: The system **may** support the creation of author accounts and the association of those accounts with their corresponding submissions.

8. Other
8.1 The system **must** track statistics such as number of submissions, number of reviews received, average time from submission to publication
8.2 The system **should** support a multilingual interface