COAR Notify Documentation

UNMAINTAINED. This documentation has been moved to the official DSpace documentation at COAR Notify - LDN Services

- Introduction
- How to enable
- Relation with the Quality Assurance Correction Service
- LDN Autodiscovery mechanism
- LDN Services Directory (Registering Services)
- LDN Inbox queueing
- Notify status boxes
- Configuring automatic QA evaluation using the Level of Trust
- Sending LDN Notifications during the submission of an item
- Automatic Inbound pattern triggering
- Item filters for Inbound pattern
- The LDN Consumer
- Understanding the structure of the LDN Notification
  - @context
  - actor
  - context
  - id
  - inReplyTo
  - object
  - origin
  - target
  - type
- Testing
  - Request Endorsement (LDN Message on item submission)
  - Testing Announce Review against current repository
- Postman Collection for testing COAR-Notify Feature

Introduction

blocked URL

The Arcadia-funded COAR Notify Project is developing and accelerating community adoption of a standard, interoperable, and decentralised approach to linking research outputs hosted in the distributed network of repositories with resources from external services such as overlay-journals and open peer review services, using linked data notifications. As part of this project, COAR is funding the development of platforms and systems to support the exchange of linked data notifications across partner organisations and the workflows to manage notifications in those platforms and systems. As the largest adopted repository platform in the World one of the first platforms to be addressed is DSpace the implementation of which has been entrusted to 4Science.

The application can receive and send LDN messages concerning items with external systems. The LDN system is the protocol of message exchanging; the Quality Assurance system is the mechanism used to approve or reject item updates.

How to enable

Configuration properties involved:

ldn.enabled = true|false	**true** => message is received and managed the server responds with an HTTP 202 code **false** => message is refused and the server responds with an HTTP 404 code
ldn.notify.inbox = <ldn inbox endpoint>	where the ldn inbox rest service is mapped on the current DSpace instance. IE: *${dspace.server.url}/ldn/inbox*
coar-notify.ip-range.enabled = true|false	enables the validation against the IP of received ldn message against the registered range

Relation with the Quality Assurance Correction Service

The LDN system, as a message exchanging i/o system, has an inbox and an outbox. Every LDN message refers to a Notify Service: all the Notify Services are configured manually from the admin application form. A Notify Service is just like an authority labelled on LDN messages.

The Quality Assurance system is the implementation of item metadata updates operations. A Quality Assurance Event contains informations for item metadata updates: QAEvent are stored into QAEvent solr collection. All of the QAEvent are shown on an administration form. Every QAEvent can be accepted, ignored or removed: if accepted some metadata of the referred item are modified, if ignored or removed nothing about the item is modified. Every QAEvent has a property named **source** : qa events created by processing an LDNMessage has ***source="coar-notify"***.

To process an LDN message means to create a QAEvents; as soon as the QAEvent is accepted the referred item is updated. We do it by routing LDN Messages from the queue to the LDN Router. The LDN Router is designed into the spring file */dspace/config/spring/api/ldn-coar-notify.xml* on the bean *ldn Router*. The router de-multiplex two attributes read from the LDN Message (two string values inside the array called ***type***) content to java classes called Processors: see **org.dspace.app.ldn.processor.*** package to view them all.

The match between LDN message ***type*** and the QAEvent **topic** (suddenly created by the evaluation of the ldn message from the queue) is configured onto *ldn-coar-notify.xml* spring file. Every Processor owns a list of actions. An action is often an email to be sent and an action to create the qa event. The LDN processor receives the QAEvent topic as a parameter and creates the relative QAEvent.

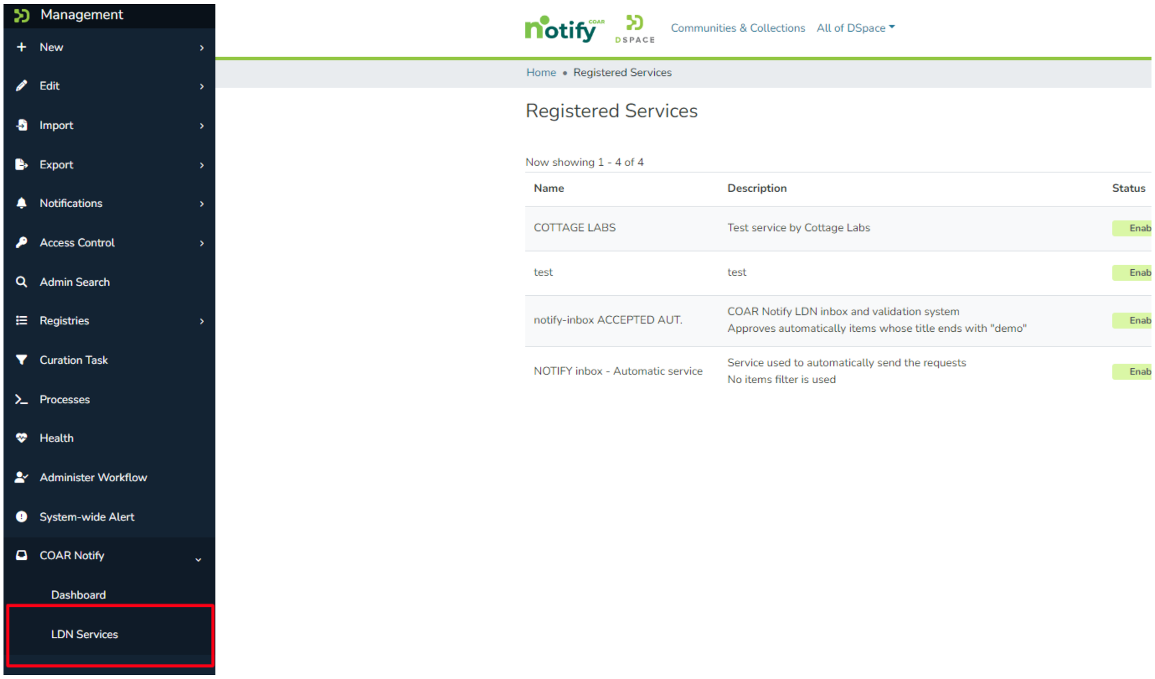
LDN Autodiscovery mechanism

Third party system can retrieve the location of the repository LDN InBox via the LDN autodiscovery mechanism, nevertheless to be able to interact with DSpace they need to be approved by a Repository Administrators and listed in the LDN Services Directory (see next paragraph); otherwise their messages will be flagged as untrusted and not processed at all.

LDN Services Directory (Registering Services)

You need to register the external services to allow the handling of the incoming LDN messages.

These services are also used to send LDN notification during the submission process of an item (Please take a look at LDN Inbound pattern)



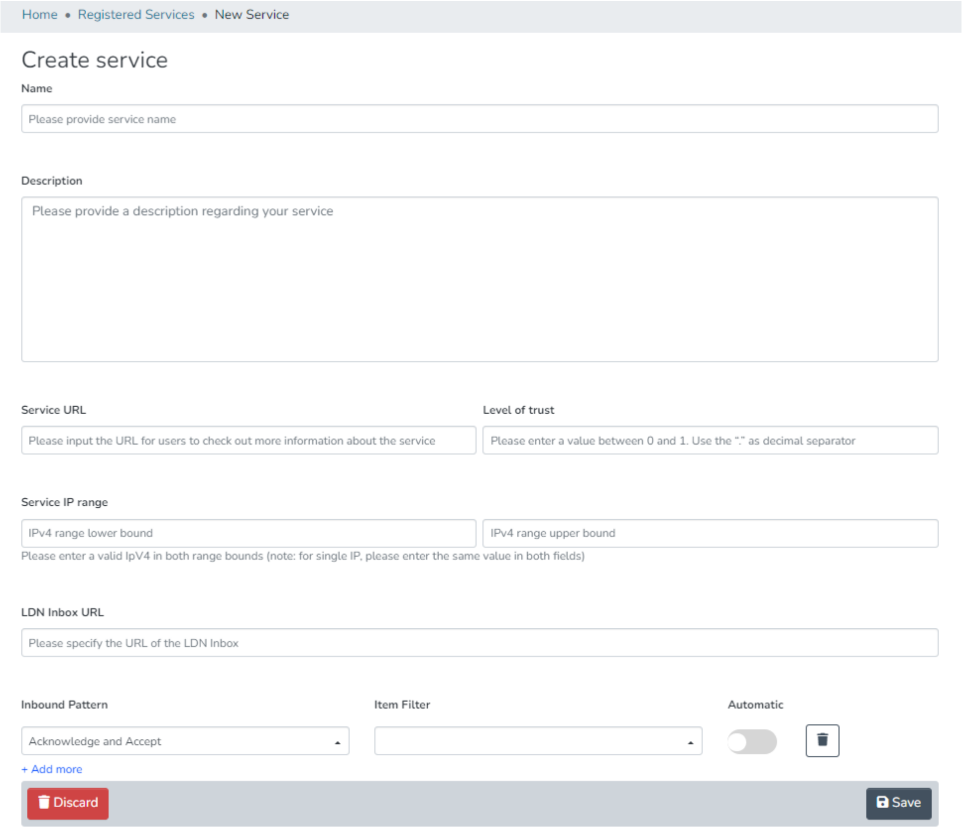
Registered Services

Now showing 1 - 4 of 4

Name	Description	Status
COTTAGE LABS	Test service by Cottage Labs	Enab
test	test	Enab
notify-inbox ACCEPTED AUT.	COAR Notify LDN inbox and validation system Approves automatically items whose title ends with "demo"	Enab
NOTIFY inbox - Automatic service	Service used to automatically send the requests No items filter is used	Enab

Administrators can manage services using the menu **LDN Services**.

The following page is used to submit a new LDN Service:


Home • Registered Services • New Service

Create service

Name

Please provide service name

Description

Please provide a description regarding your service

Service URL

Please input the URL for users to check out more information about the service

Level of trust

Please enter a value between 0 and 1. Use the "." as decimal separator

Service IP range

IPv4 range lower bound

IPv4 range upper bound

Please enter a valid IpV4 in both range bounds (note: for single IP, please enter the same value in both fields)

LDN Inbox URL

Please specify the URL of the LDN Inbox

Inbound Pattern	Item Filter	Automatic
Acknowledge and Accept		

Name: a label for the Notify Service used on the UI

Description: the description of the Notify Service - add details here

Service URL: the url of the remote Notify Service. This is mostly used as a descriptive URL when sending notification to an external system.

The service URL must not be confused with the inbox url. As said this url is descriptive so we expect the main application URL to be added here

Level of Trust: floating point number value accepted between 0 and 1.

This value is used to assign to the service a TRUST value that describes how much reliable the external service is.

This value is not used if the automatic processing of QA Events is disabled

Check the configuration file *dspace/config/spring/api/qaevents.xml* to enable the automatic approval

When the automatic approval is configured the trust value of the LDN Service is check against configured score thresholds.

Thresholds for automatic Approval/Rejection/Ignoring can be set.

Also within a specified range QA Events will need a manual approval

Service IP Range: two IP addresses expected as minimum and maximum.

This range is used to validate the received LDN Messages.

When the LDN notification doesn't match the configured IP Range the notification is stored as UNTRUSTED

LDN Inbox URL: this is the url used to send the LDN Notifications.

This url is also used when a notification is received to retrieve the registered LDN service it belongs to

This URL must is unique among the registered LDN Services

Inbound Pattern: the section for Inbound pattern is the section which describes what operations/actions are supported by the external service

The **pattern** itself can be selected from the dropdown as there's a list of pattern. To better understand pattern usage please refer to the official documentation https://notify.coar-repositories.org/patterns/

The **item filter** is used to describe which item can be processed for the current LDN Service/Inbound pattern.
if the item filter is not set any item is allowed. If the filter is selected only for matching items the LDN Notification will be sent
(in case of automatic LDN Notification the notification is not sent, In case of NON-automatic service a validation will prevent requests to be sent to the external service)

The **automatic** flag when set to true the ldn message exchange is performed automatically right after the item submission has finished.
this Automatic workflow generates an Outgoing LDN message targeting the Notify service for the just submitted item.
The automatic flag involves only the submission phase of an item.
If no item filter is set - the LDN notification is generated for any submitted item.

LDN Inbox queueing

LDN incoming messages are stored into the **ldn_message** database table. As far as the property **ldn.enabled** is true and the incoming json is valid, the LDN Message is stored on the table. Together with the storage of the record, the queue status of the message is initialized. The *queue_status* column of the table contains the status of the LDN message inside the queue. All the possible queue_status values are described into the java class ***org.dspace.app.ldn.LDNMessageEntity*** as integer constants.

Status name	Value in DB	Description
QUEUE_STATUS_UNTRUSTED_IP QUEUE_STATUS_UNTRUSTED	0 5	Message must not be routed as it is not trusted. This may occur if the IP address of the notifications' sender doesn't match the provided "IP Range" or if the service inbox URL in the origin section of the message doesn't match with any registered LDN Service entry's Inbox URL.
QUEUE_STATUS_QUEUED	1	Message is waiting in the queue to be processed by the Extractor
QUEUE_STATUS_PROCESSING	2	Message is currently being processed by the Extractor
QUEUE_STATUS_PROCESSED	3	Message has been **successfully** processed by the Extractor
QUEUE_STATUS_FAILED	4	Message has been evaluated but its routing has failed

If all validations are run successfully the LDN Notification status is set as **QUEUE_STATUS_QUEUED** and the notification **will** be processed as soon as the extractor retrieves it from the queue.

The LDN Message logical queue is managed by:

- The **LDN Message Extractor** is an asynchronous DSpace task which retrieves the oldest processable LDN message.
  A LDN notification is processable by the Extractor only if its queue status is **QUEUE_STATUS_QUEUED** other status will not be considered by the extractor.
  The extractor process ends as soon as the LDN extracted message is routed and processed (either with success of failure).

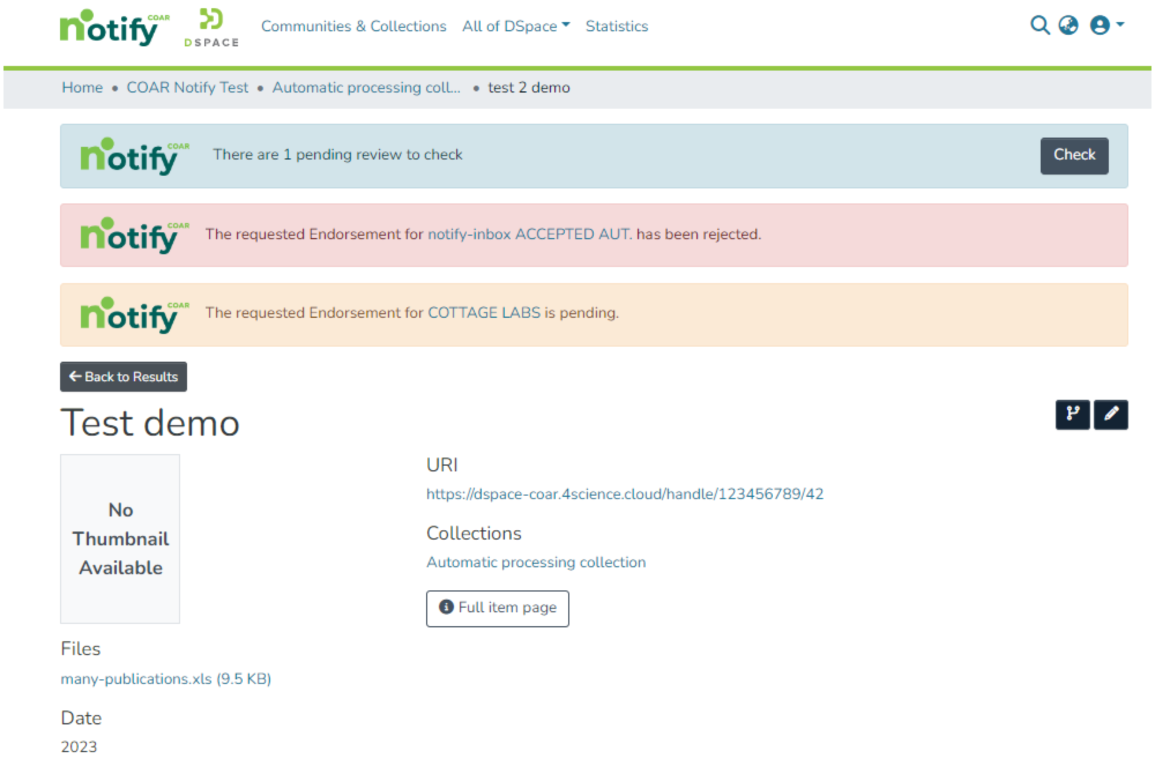
- The **LDN Message Timeout Checke**r is an asynchronous DSpace task that looks for timed-out messages with attempts less than Y, where Y= the value of the configuration property *ldn.processor.max.attempts*. Cron configuration at property *ldn.queue.timeout.checker.cron*
  Each notification if not successfully processed is retrieved in the next execution of the Checker and attempts are kept up to date (increased by 1 and the timeout is increased by X minutes), where X= the value of the configuration property *ldn.processor.queue.msg.timeout* | defaulted to 60.

Please consider that this means that the corresponding QAEvent is not automatically created as soon as the LDN Message is received.
The QA Event related to the LDN Notification will be created only once the notification is successfully processed by the extractor (LDN queue status set to **QUEUE_STATUS_PROCESSED**).

Notify status boxes

Considering these possible scenarios here at: COAR Notify Protocol: Example Scenarios

We have to keep the user updated about the item situation. We do it with colored boxes on its landing (handle) page.


Communities & Collections
All of DSpace
Statistics
Home • COAR Notify Test • Automatic processing coll... • test 2 demo
Check
Back to Results
Full item page

There are 1 pending review to check

The requested Endorsement for notify-inbox ACCEPTED AUT. has been rejected.

The requested Endorsement for COTTAGE LABS is pending.

Test demo

No Thumbnail Available

URI
https://dspace-coar.4science.cloud/handle/123456789/42

Collections
Automatic processing collection

Files
many-publications.xls (9.5 KB)

Date
2023

When the Offer message (being it review, endorsement or ingest) has been sent as an outgoing LDN message, and nothing else about it has been received, the yellow box is shown.

When the Offer message has been followed by a related incoming Acknowledgement message: if the ack is a tentative rejct the box shown is red.

When Offer message has been followed by a related Announce incoming message, the box shown is blue! It is blue because receiving an Announce means - then the message is extracted - we produce a new QA Event. As every QA Event it is shown on Item Page and myDSpace page.

Configuring automatic QA evaluation using the Level of Trust

Score is a number 0 < # < 1.
If the proper configuration is enabled an automatic check for approval of QAEvents is run once the LDN message is extracted.
On *qaevents.xml* file the bean ***qaScoreEvaluation*** we can configure three different boundaries to automatically approve/reject/ignore the level of trust:

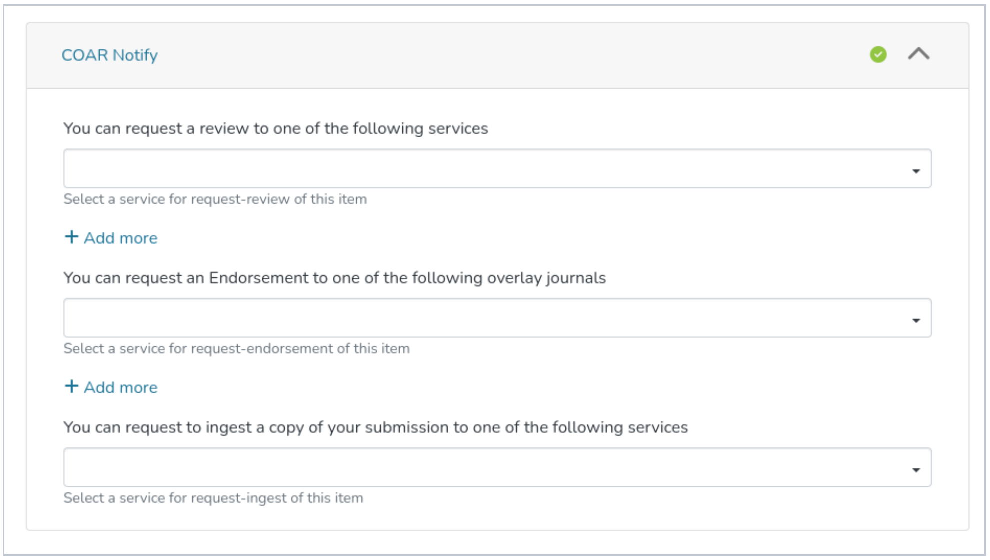
```
<property name="scoreToReject" value="0.3" />
<property name="scoreToIgnore" value="0.5" />
<property name="scoreToApprove" value="0.8" />
```

- if score <= rejection deletes the QAEvent;
- if score <= ignore discards the QAEvent;
- if score >= approval accepts the QAEvent automatically
- if score respects the following: `scoreToIgnore` < score < `scoreToApprove`
  no automatic action is performed and the user must check manually the QA event

This feature must not be confused with the automatic triggering for LDN Inbound pattern

Sending LDN Notifications during the submission of an item

In the example below it is shown the submission of an Item with a new section for COAR-Notify.
In this section we can manually choose **ldn-services** (*which were not marked as automatic*) in order to ask for Review, Endorsement and Ingest to external services.
If No (non-automatic) pattern has been configured for the LDN Service this service will not be displayed in any of the dropdown.



COAR Notify

You can request a review to one of the following services

Select a service for request-review of this item

+ Add more

You can request an Endorsement to one of the following overlay journals

Select a service for request-endorsement of this item

+ Add more

You can request to ingest a copy of your submission to one of the following services

Select a service for request-ingest of this item

Automatic Inbound pattern triggering

When adding new LDN Inbound pattern to a LDN Service we can select if a pattern has to be automatic or not.

Flagging a pattern as automatic means that the request described by the current pattern is sent automatically to the external service:

- No action is required during the submission to select the LDN Service for a specific pattern
- The service in not even listed in the COAR notify step of the submission

Item filters for Inbound pattern

When adding inbound pattern to a service we have the possibility to add an item-filter for each described pattern

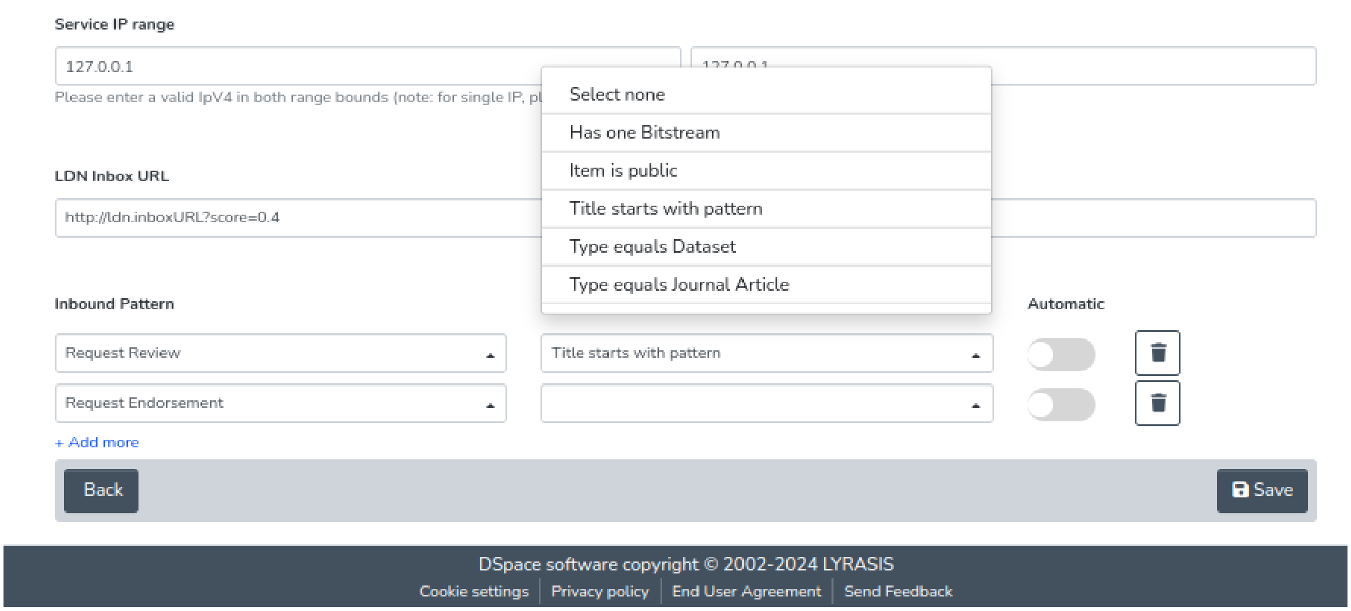
An item filter allows to filter items according to specified criteria.

- If the pattern is flagged as automatic the item filter is evaluated automatically
  - if the item filter is respected the LDN Notification is stored and queued for sending
  - if the item filter is not respected the LDN Notification is not even generated an nothing will be sent to the external service
- If the pattern is **not** flagged as automatic the item filter is evaluated when choosing the service from the dropdown for Notify in the submission
  - if the pattern is not respected an error message will be displayed saying the selected LDN service cannot be used for the current item
  - if the pattern is respected no error message is displayed

Item Filter is a non-mandatory field. It activates a filter to be applied to the item during the submission.
If the item matches the item filter condition the submission will be successful: otherwise the submission will fail with a talkative message.
Visible item filters are configured into a static list mapped into *item-filters.xml* file - on a bean named *ldnItemFilters*. We established for the following item filter to be shown:

Has one Bitstream = the item must have exactly one bundle named "ORIGINAL";
Item is public = anonymous user can see/read the item;
Title starts with pattern = the dc.title metadata of the item must start with 'Pattern';
Type equals dataset = the dc.type metadata of the item must be 'Dataset';
Type equals dataset = the dc.type metadata of the item must be 'Journal Article';

item-filter.xml has all item filters declarations: only the ones declared inside the *ldnItemFilters* map are shown.



Service IP range

127.0.0.1 | 127.0.0.1

Please enter a valid IpV4 in both range bounds (note: for single IP, pl

LDN Inbox URL

http://ldn.inboxURL?score=0.4

Inbound Pattern | Automatic

Request Review | Title starts with pattern

Request Endorsement |

DSpace software copyright © 2002-2024 LYRASIS

The LDN Consumer

The LDN Consumer is the consume responsible for generating the **outgoing** LDN Notification.

This consumer is triggered any time an Item is successfully installed and a check against the registered LDN Services/patterns is done to generate the appropriate LDN Notification

This consumer is responsible for the handling of both automatic a non-automatic patterns

```
event.consumer.ldnmessage.class = org.dspace.app.ldn.LDNMessageConsumer
event.consumer.ldnmessage.filters = Item+Install
```

Understanding the structure of the LDN Notification

The LDN Notification includes some important section many of these if improperly set might led to errors in evaluations on the notification

Let's take a deeper look at the LDN Notification structure

@context

The context section is the same for any LDN Notification

```
  "@context": [
    "https://www.w3.org/ns
/activitystreams",
    "https://purl.org/coar/notify"
  ]
```

actor

This section describes the actor which performs the request
This is a descriptive section and doesn't include important information

```
  "actor": {
    "id": "https://review-
service.com",
    "name": "Review Service"
,
    "type": "Service"
  }
```

context

This is one of the most important section.
In this section we have a description of the Item being involved in the LDN Notification

- **id** is the identifier URL of the item in the repository
  this field is really important to identify and link the Item to the current notification

other fields are descriptive fields for the item

```
  "context": {
    "id": "https://research-organisation.org/repository/preprint/201203/421/",
    "ietf:cite-as": "https://doi.org/10.5555/12345680",
    "type": "sorg:AboutPage",
    "url": {
      "id": "https://research-organisation.org/repository/preprint/201203/421
/content.pdf",
      "mediaType": "application/pdf",
      "type": [
        "Article",
        "sorg:ScholarlyArticle"
      ]
    }
  }
```

id

The identifier of the LDN notification (Each notification has a different id)

```
"id": "urn:uuid:94ecae35-dcfd-4182-8550-22c7164fe23f"
```

inReplyTo

This field is used to reply to other LDN notification so that the system knows if the current notification is a response

```
"inReplyTo": "urn:uuid:0370c0fb-bb78-4a9b-87f5-bed307a509dd"
```

object

This is a descriptive section of the received Review, Endorsement etc..

- **id** is the URL of the review/endorsement etc.

```
  "object": {
    "id": "https://review-service.com/review/geo
/202103/0021",
    "ietf:cite-as": "https://doi.org/10.3214
/987654",
    "type": [
      "Document",
      "sorg:Review"
    ]
  }
```

origin

This is an important section since it is used (for incoming notifications) to determine the service among the registered ones.
If the service is not recognized as registered the notification is **untrusted**

The field being used for this check is **inbox**

```
  "origin": {
    "id": "https://review-service.com/system",
    "inbox": "https://review-service.com/inbox/",
    "type": "Service"
  }
```

target

This section is used to describe the target system involved in the notification workflow

```
  "target": {
    "id": "https://generic-service.com/system",
    "inbox": "https://generic-service.com/system/inbox/",
    "type": "Service"
  }
```

type

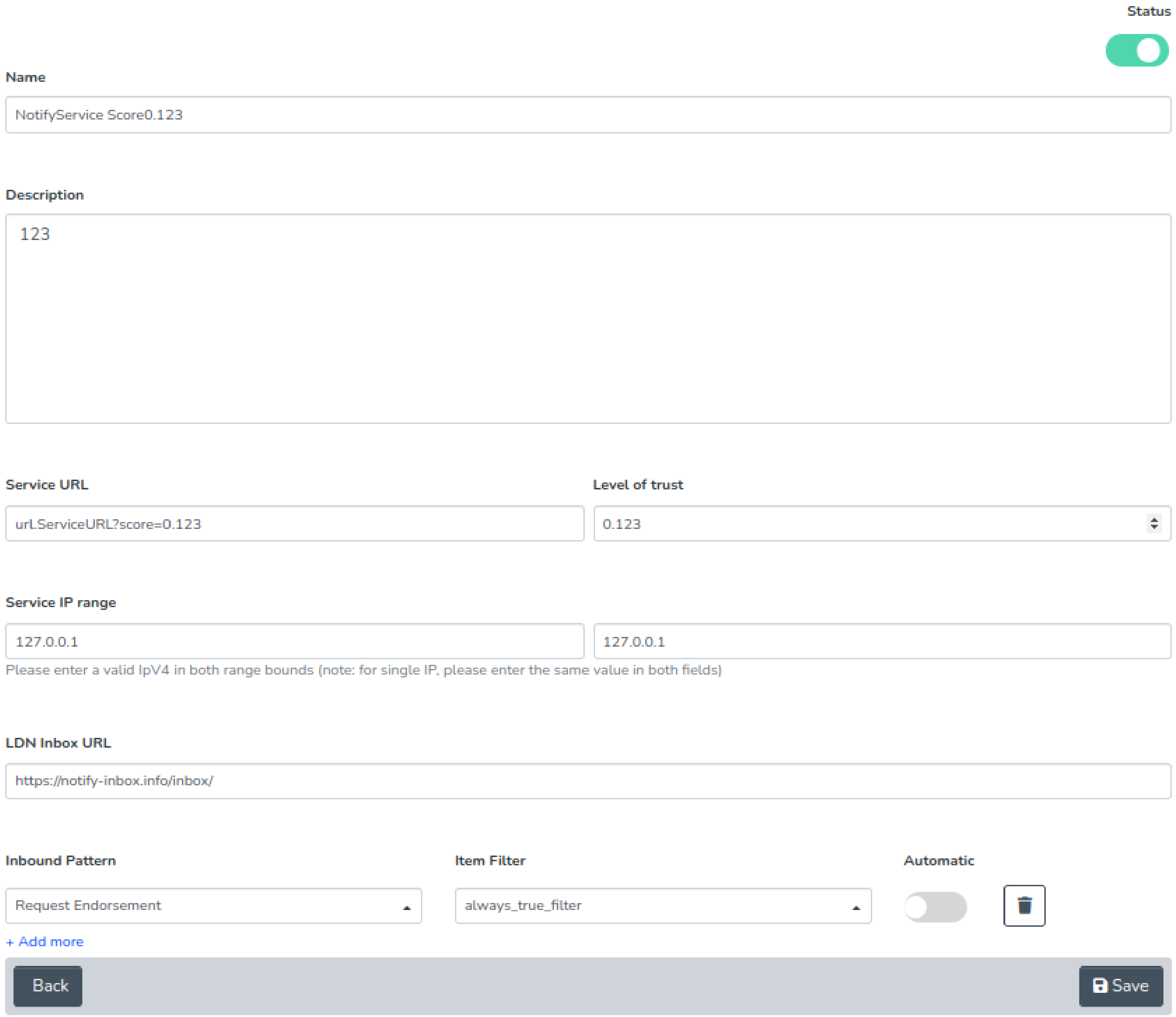
This type section is important since it's determining how to process/evaluate any received notification

```
  "type": [
    "Announce",
    "coar-notify:ReviewAction"
  ]
```

Testing

Request Endorsement (LDN Message on item submission)

Let's configure a new Notify Service such as:



Status

Name

NotifyService Score0.123

Description

123

Service URL

url.ServiceURL?score=0.123

Level of trust

0.123

Service IP range

127.0.0.1

127.0.0.1

Please enter a valid IpV4 in both range bounds (note: for single IP, please enter the same value in both fields)

LDN Inbox URL

https://notify-inbox.info/inbox/

Inbound Pattern

Request Endorsement

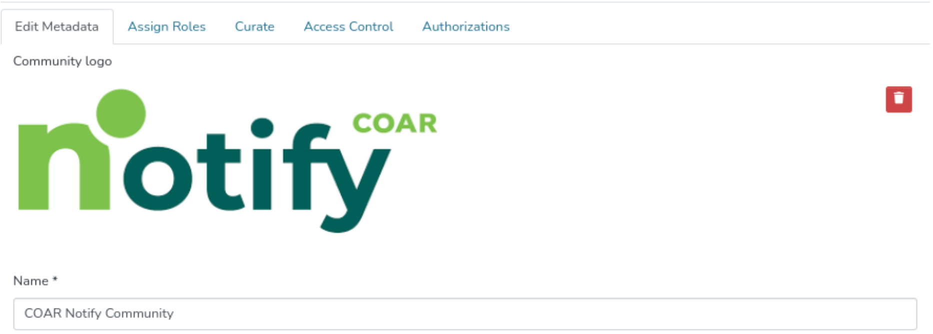
Item Filter

always_true_filter

Automatic

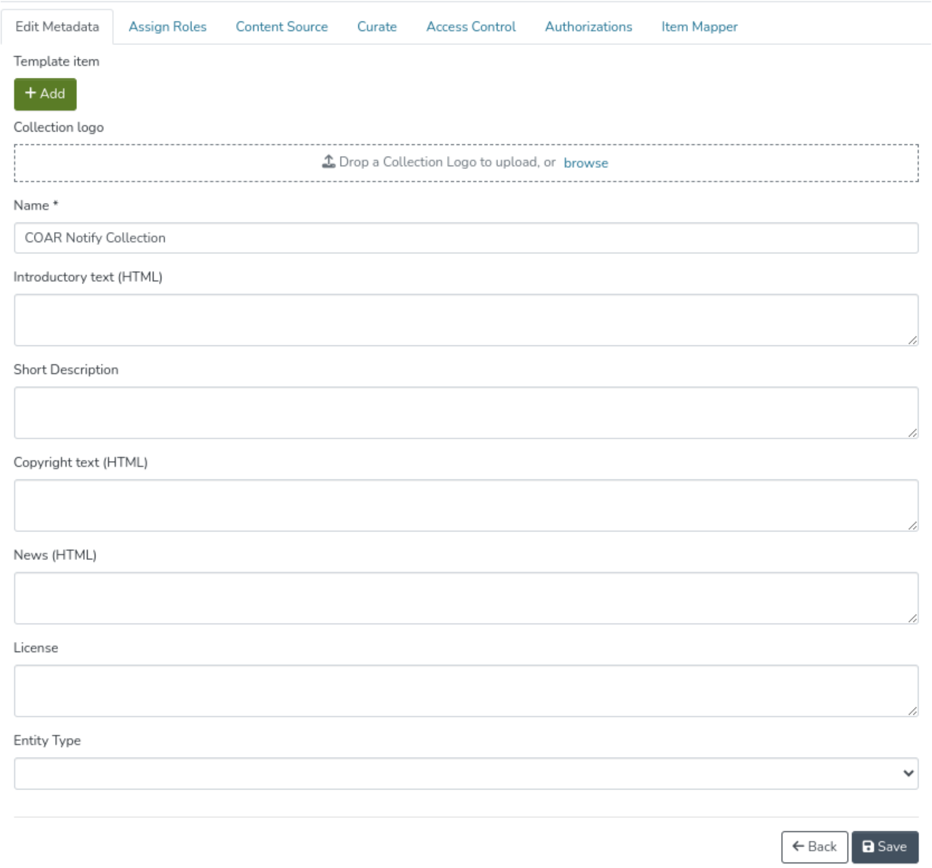
and save.

(NOTE: If you plan to test with Docker, the "Service IP Range" should likely include 172.23.0.1 – this is the IP that your Host machine will likely appear as when running the backend in Docker.)

Create a new Community:	 Community logo Name * COAR Notify Community

Create a new Collection inside the community created above.

It is important for this community to have no EntityType set.



Template item

Collection logo

Name *

COAR Notify Collection

Introductory text (HTML)

Short Description

Copyright text (HTML)

News (HTML)

License

Entity Type

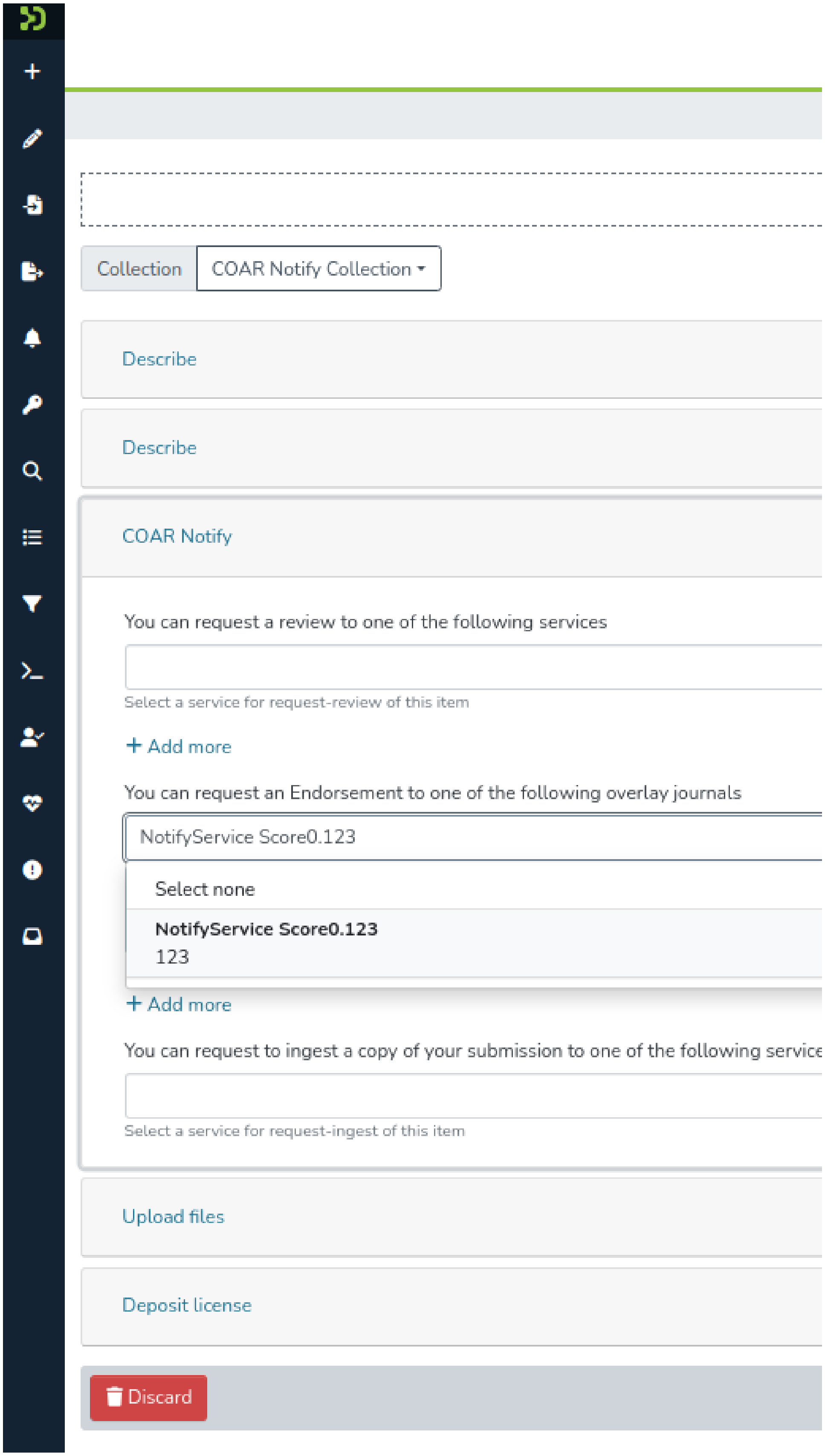
Take a look into the *item-submission.xml* file for the "***traditional***" step sequence configuration. Here we've set the coarnotify-step! The traditional step sequence is used as the default, and the default is used for Collection with no entity type associated.

If the coar-notify step is needed for every collection possible,

the entry-step must be added onto every ***<submission-process>***

described into *item-submission.xml*	
Create a new Item, select the collection created above. As expected we see the COAR Notify section: here it is the	

ldn request configuration: comboes for Review, Endorsement and Ingest are filled with the corresponding configured LDN Services.



Collection | COAR Notify Collection ▾

Describe

Describe

COAR Notify

You can request a review to one of the following services

Select a service for request-review of this item

+ Add more

You can request an Endorsement to one of the following overlay journals

NotifyService Score0.123

Select none

NotifyService Score0.123
123

+ Add more

You can request to ingest a copy of your submission to one of the following service

Select a service for request-ingest of this item

Upload files

Deposit license

Discard

For this example we see only one value for the Endorsement

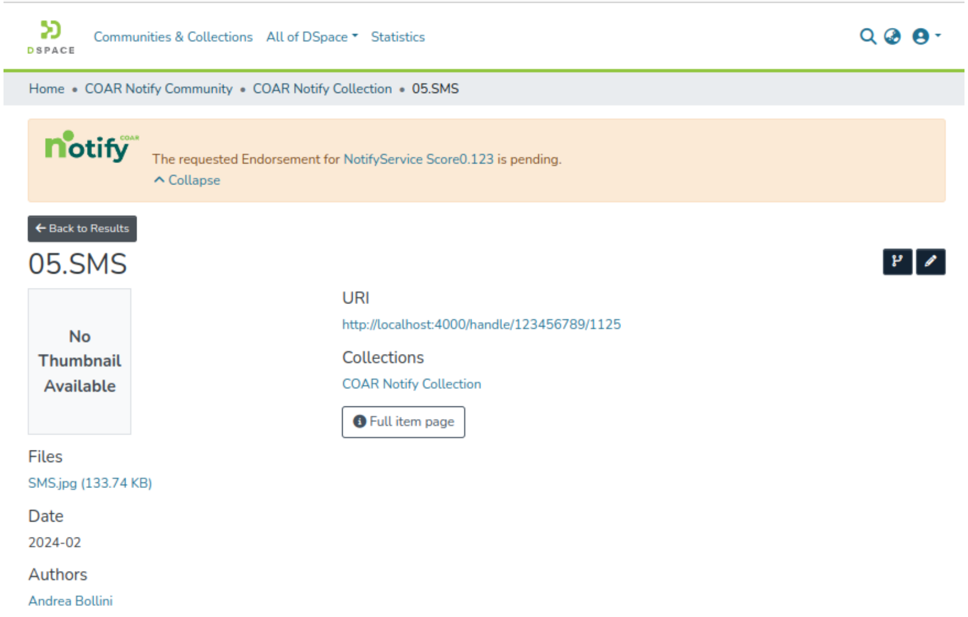
combo (see the Inbound Pattern configured onto the NotifyService

namedNotifyServiceScore0.123 - first screenshot of this paragraph) Deposit the item. A new LDN Message of type **Offer:EndorsementAction**

is stored and enqueued! As soon as the Extractor manages it, it is enrouted to the Processor configured at *ldn-coar-notify.xml* (as one of the <*outcomingProcessors*>). The processor is the bean named outcomingAnnounceEndorsementAction, which lists the SendLDNMessageAction. So, it is only at this point, when the

action SendLDNMessageAction is executed - that the notification is sent out to the external NotifyService.

To whom the notification is sent out? To the url configured as the LDN Inbox property of the NotifyService

(see above: https://notify-inbox.info/inbox/).	
as soon as our Offer is correctly received on the Notify Service side,	Until an Acknowledge or an Announce is received for this item, on the item page you can see the upper yellow window as:  DSPACE Communities & Collections All of DSpace Statistics Home • COAR Notify Community • COAR Notify Collection • 05.SMS notify COAR Collapse Back to Results No Thumbnail Available Full item page The requested Endorsement for NotifyService Score0.123 is pending. 05.SMS URI http://localhost:4000/handle/123456789/1125 Collections COAR Notify Collection Files SMS.jpg (133.74 KB) Date 2024-02 Authors Andrea Bollini

the Notify Service has to provide us - asynchronously as a separate Acknowledge LDN message - the Acceptance or Rejection of the Endorsement we offered.

When Acceptance is received:

The Acknowledge LDN Message - when received - is managed by the Processor configured into the bean named *acceptAckAction*.

When Rejection is received:

The Acknowledge LDN Message - when received - is managed by the Processor configured into the bean named *rejectAckAction*.	

at time of writing COAR service for test purposes is responding an HTTP 500

as we sent the Offer. This error did not exist back in December 2023 when we

firstly developed the LDN implementation.	

Testing Announce Review against current repository

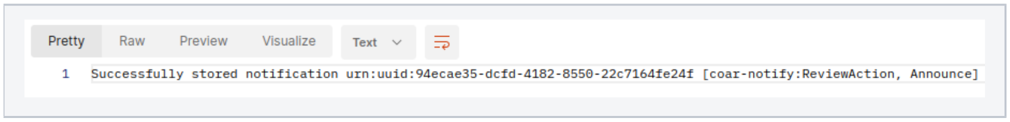
Let's trigger an Request Review (see COAR official pattern documentation here).

For this specific example we can set just the first Inbound Pattern as Request Review and leave the item filter blank. This test is not about item submission. Pick the following POST call from the Postman project (see below to download it and import it to your installed PostMan instance): let's emulate the Notify Service external system sending us an LDN Message!

For this test to succeed you *must update the json below* based on your system configuration. See the "IMPORTANT" lines.

```
{
    "@context":[
        "https://www.w3.org/ns/activitystreams",
        "https://purl.org/coar/notify"
    ],
    "actor":{
        "id":"https://orcid.org/0000-0002-1825-0097",
        "name":"Josiah Carberry",
        "type":"Person"
    },
    "id":"urn:uuid:0370c0fb-bb78-4a9b-87f5-bed307a50dff", //// IMPORTANT!! this MUST be a unique value because it's going to be stored as uuid into the ldn_message table.
    "object":{
        "id":"http://localhost:4000/handle/123456789/1122",   ////IMPORTANT!! this MUST be the URL of an Item in your DSpace repository. It can be either a handle-based URL or a UUID-based URL.
        "ietf:cite-as":"https://doi.org/10.5555/12345680",
        "type":"sorg:AboutPage",
        "url":{
            "id":"https://research-organisation.org/repository/preprint/201203/421/content.pdf",
            "mediaType":"application/pdf",
            "type":[
                "Article",
                "sorg:ScholarlyArticle"
            ]
        }
    },
    "origin":{
        "id":"https://research-organisation.org/repository",
        "inbox":"https://research-organisation.org/inbox",  /// IMPORTANT!! this field MUST match the "LDN Inbox URL" for one of the LDN/Notify Services you've added to your repository. (See previous screenshot)
        "type":"Service"
    },
    "target":{
        "id":"https://review-service.com/system",
        "inbox":"https://notify-inbox.info/inbox",
        "type":"Service"
    },
    "type":[
        "Offer",
        "coar-notify:ReviewAction"
    ]
}
```

See below the successful response:

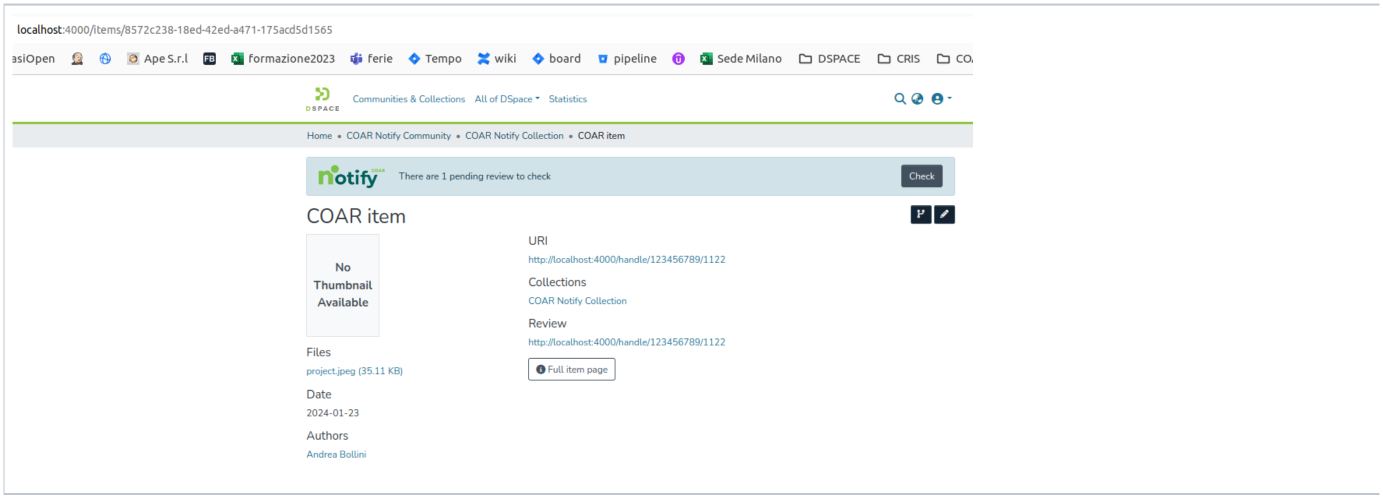


```
Successfully stored notification urn:uuid:94ecae35-dcfd-4182-8550-22c7164fe24f [coar-notify:ReviewAction, Announce]
```

This means the LDN Message has been received.
At this point there's not feedback if the notification has been stored & enqueued or just stored as untrusted.

Please remember the enqueued ldn messages must be evaluated by the Extractor for a QA Event (with source=**coar-notify**) to be created.

As soon as it is processed by the extractor, a blue popup is shown on Item and myDSpace pages:


localhost:4000/items/8572c238-18ed-42ed-a471-175acd5d1565
asiOpen
Ape S.r.l
FB
formazione2023
ferie
Tempo
wiki
board
pipeline
Sede Milano
DSPACE
CRIS
CO
DSPACE
Communities & Collections
All of DSpace
Statistics
Home • COAR Notify Community • COAR Notify Collection • COAR item
notify COAR
Check
Full item page

There are 1 pending review to check

COAR item

No Thumbnail Available

Files
project.jpeg (35.11 KB)

Date
2024-01-23

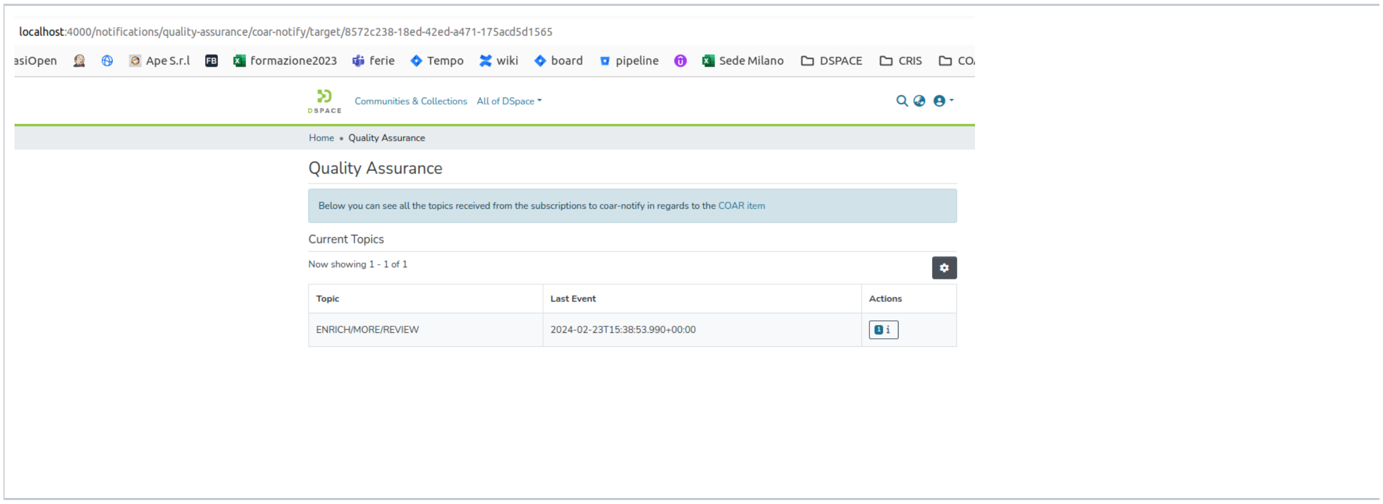
Authors
Andrea Bollini

URI
http://localhost:4000/handle/123456789/1122

Collections
COAR Notify Collection

Review
http://localhost:4000/handle/123456789/1122

by clicking on the Check button we are redirected to the Quality Assurance Page for the source **coar-notify**:


localhost:4000/notifications/quality-assurance/coar-notify/target/8572c238-18ed-42ed-a471-175acd5d1565
asiOpen
Ape S.r.l
FB
formazione2023
ferie
Tempo
wiki
board
pipeline
Sede Milano
DSPACE
CRIS
CO
DSPACE
Communities & Collections
All of DSpace
Home • Quality Assurance

Quality Assurance

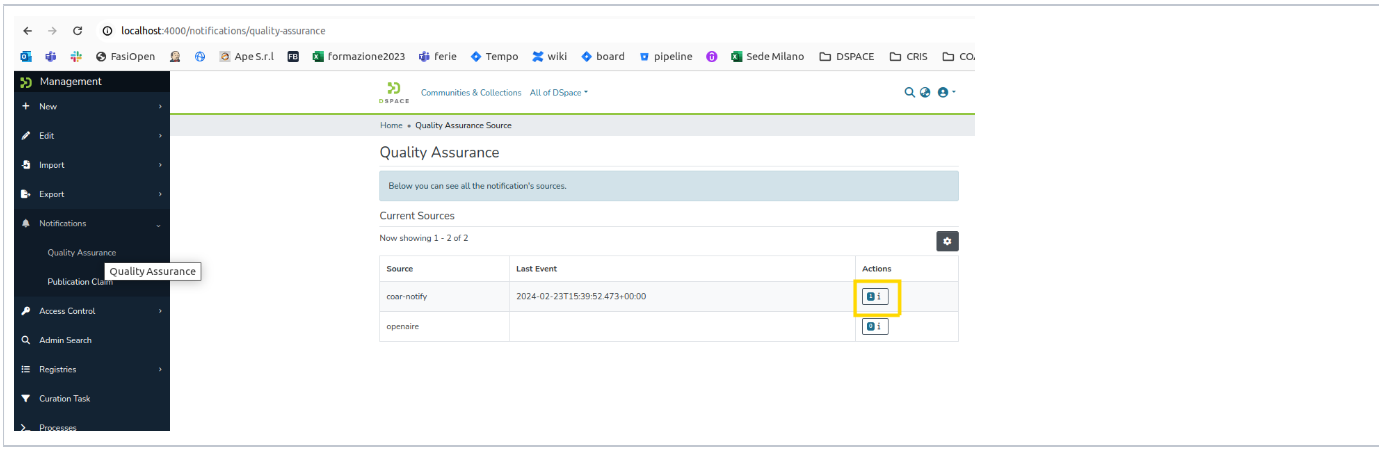
Below you can see all the topics received from the subscriptions to coar-notify in regards to the COAR item

Current Topics

Now showing 1 - 1 of 1

Topic	Last Event	Actions
ENRICH/MORE/REVIEW	2024-02-23T15:38:53.990+00:00	1 i

If we navigate to the previous Quality Assurance page we can see the counter for QA Event of coar-notify source increased by 1!



Quality Assurance

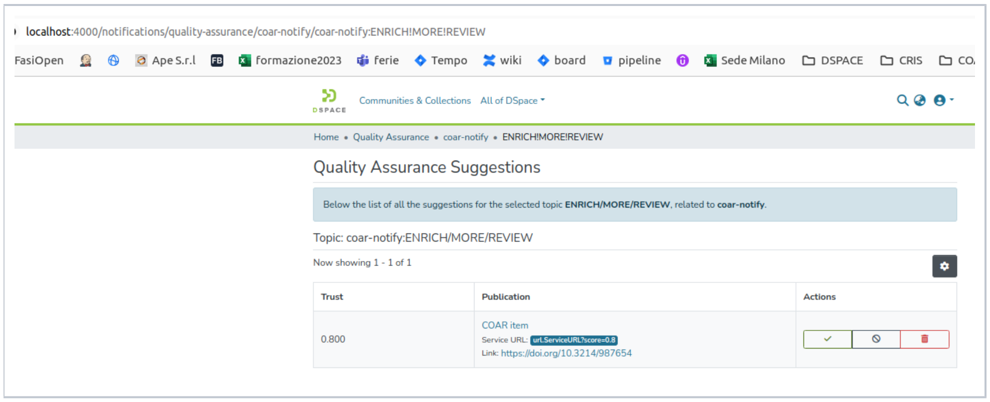
Below you can see all the notification's sources.

Current Sources

Now showing 1 - 2 of 2

Source	Last Event	Actions
coar-notify	2024-02-23T15:39:52.473+00:00	1 i
openaire		0 i

Here it is the page of the specific QA Event in example:



Quality Assurance Suggestions

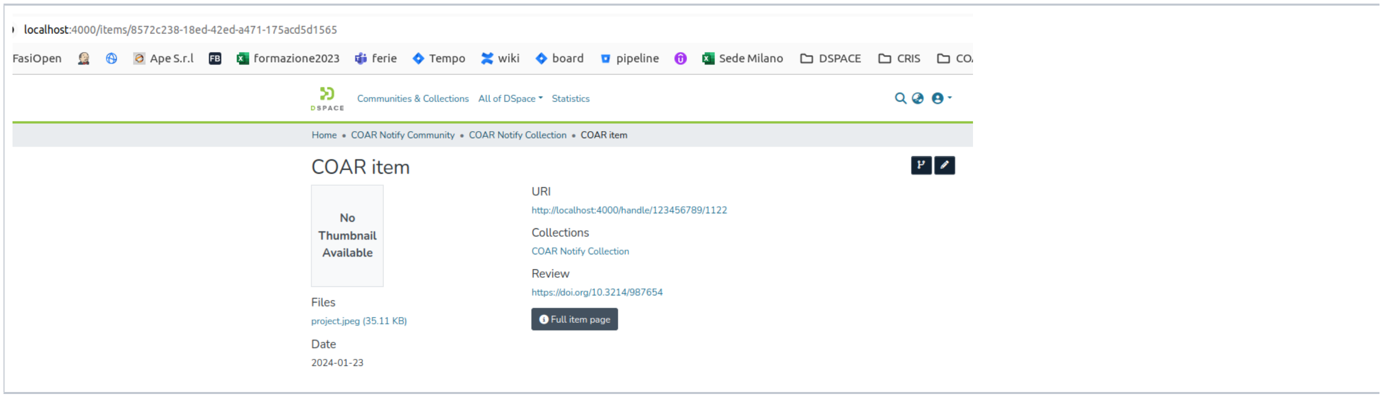
Below the list of all the suggestions for the selected topic **ENRICH/MORE/REVIEW**, related to **coar-notify**.

Topic: coar-notify:ENRICH/MORE/REVIEW

Now showing 1 - 1 of 1

Trust	Publication	Actions
0.800	COAR item Service URL: url.ServiceURL?score=0.8 Link: https://doi.org/10.3214/987654	✓ ⊘ 🗑

Let's accept it with a click on the Green Check.
Go back to the item page to see the new Review metadata!



COAR item

URI
http://localhost:4000/handle/123456789/1122

Collections
COAR Notify Collection

Review
https://doi.org/10.3214/987654

Files
project.jpeg (35.11 KB)

Date
2024-01-23

Postman Collection for testing COAR-Notify Feature

Here's the Postman collection for all test scenarios:

Coar Notif...ction.json