



EXPOSING A FLEXIBLE, COMPOSABLE & EXTENSIBLE REST API



Thierry Delprat
td@nuxeo.com
<https://github.com/tiry/>

AGENDA

- Quick introduction
 - *provide some context*
- API design constraints & principles
 - *explain the problem we want to solve*
- Building Nuxeo API
 - *REST + Automation + Composition*
- Design consequences
 - *price of flexibility*

SOME CONTEXT

What we Do and What Problems We Try to Solve



NUXEO

nuxeo

*we provide a Platform that developers can use to
build highly customized Content Applications*

*we provide components, and the tools to assemble them
everything we do is open source*

<https://github.com/nuxeo>

various customers - various use cases



Track game builds



Electronic Flight Bags



Central repository for Models



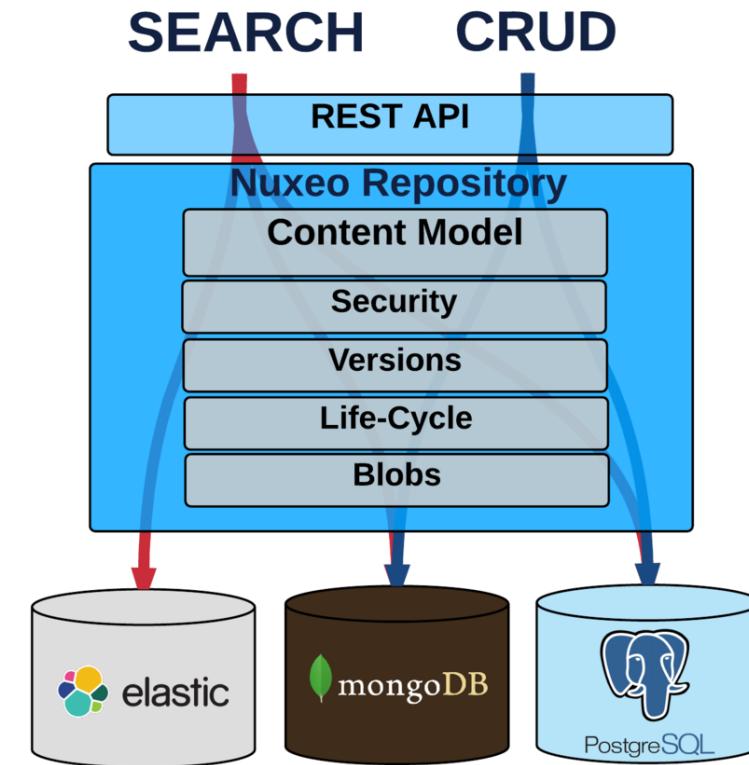
Food industry PLM

me: developer & CTO - joined the Nuxeo project 10+ years ago

NUXEO PLATFORM

nuxeo

Repository

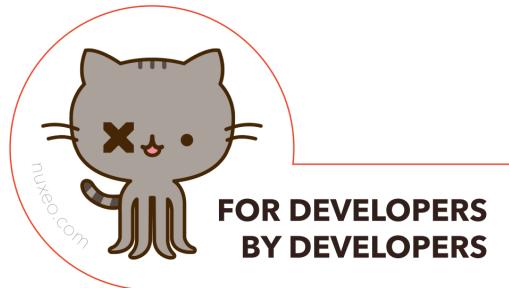


Services

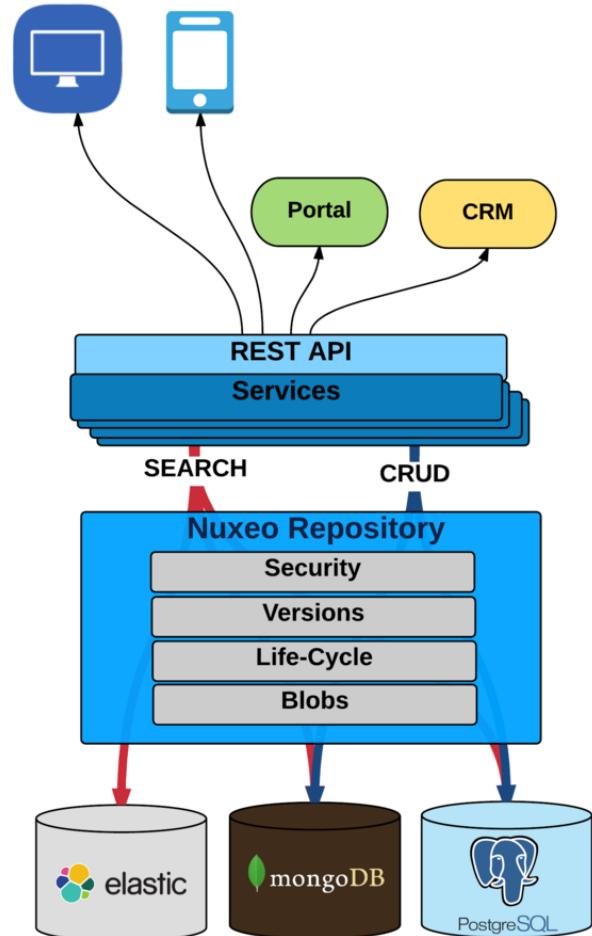
- *Workflows, Conversions, Diff, Notifications, Activity ...*

WHY API IS KEY FOR US

nuxeo



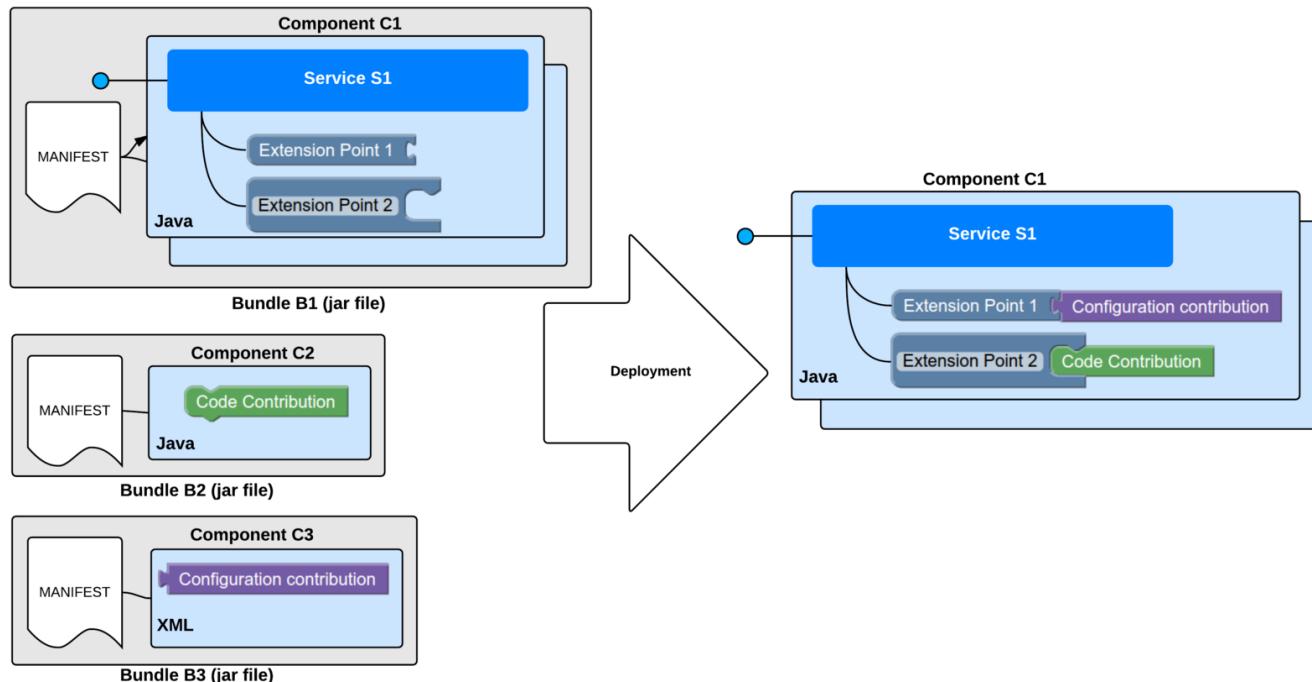
- Nuxeo Repository is a **backend**
 - Portals, Mobile Apps, ERP, CRM ...
- API is **UI**
 - *for the developers*
 - HTML5/JS



PLUGABILITY CHALLENGE

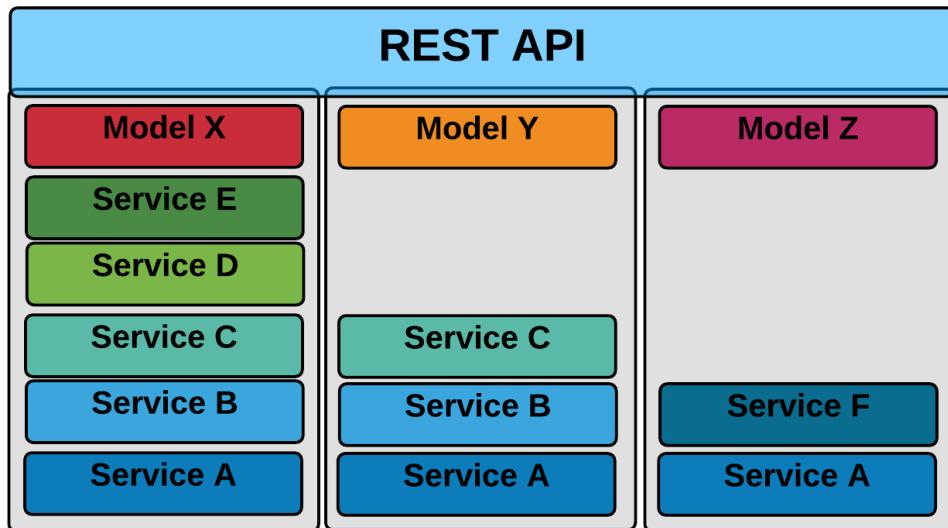
nuxeo

- In Nuxeo architecture **everything is a plugin**
 - Nuxeo Server can provide a single service or 100's of services
- Everything is **configurable**
 - Logic and Data Structures depends on configuration



API CHALLENGE

Expose a Platform: not an application



"One API"

but

Multiple combinations
of
services, plugins
and Domain Models

*developers using the platform
want to expose the API of their Application*

NEED TO FIND A SOLUTION

One Platform → "One API"





REST API DESIGN PRINCIPLES

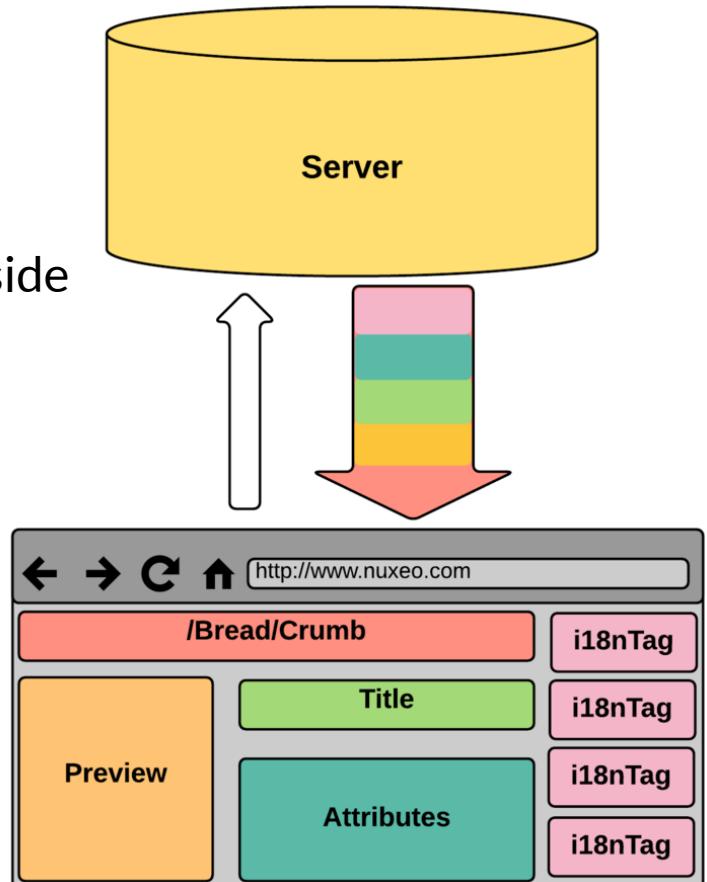
what we want to have



BE EFFICIENT

nuxeo

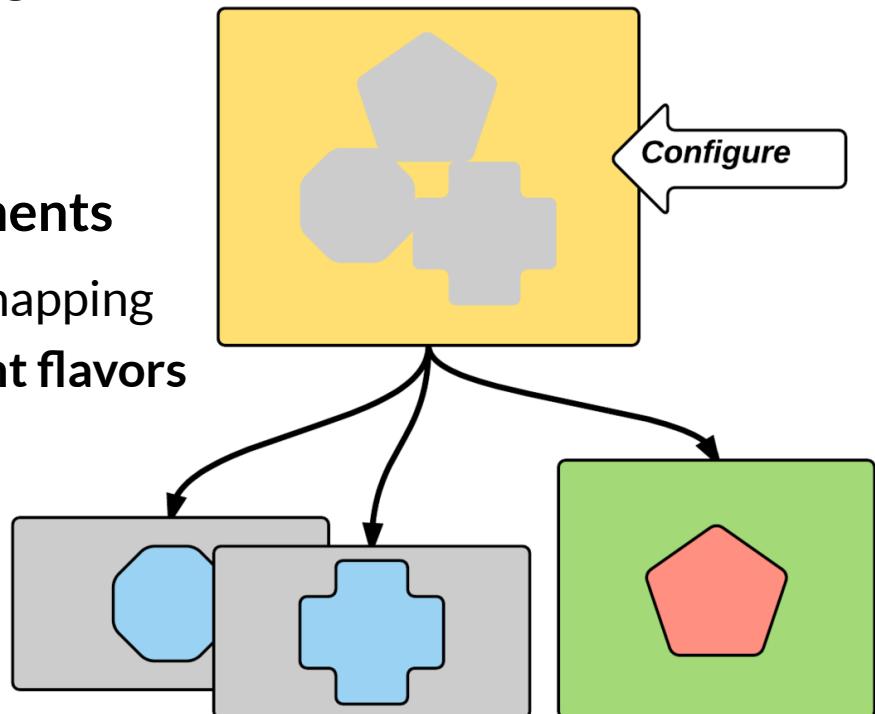
- Avoid **round trips**
 - Get all needed data in one call
 - Resolve some data on the server side
- Avoid fetching too much data



BE FLEXIBLE

nuxeo

- Adapt to the server side **configuration**
 - Domain model definition
- Adapt to client side **requirements**
 - Provide data for the screen mapping
 - Application can have **different flavors**



KEEP IT CLEAN

nuxeo

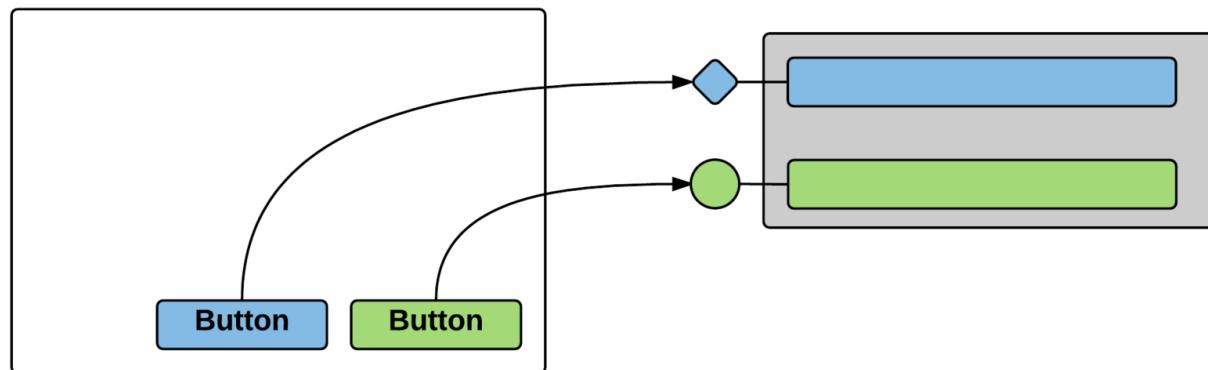
- Work between **transaction boundaries**
 - do all the work in one call
- Ensure **isolation**
 - Other users should not see inconsistent data
- Maintain **encapsulation**
 - Client should not make assertion on server implementation

Client consumes a service, it does not build the service.

BE EXTENSIBLE

nuxeo

- Expose *any meaningful business API*
 - Make API clean and application maintenance easy
- Adapt API granularity to the target Applications
 - one API behind each single button

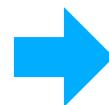


We can not build the target Business API: users/devs will do it

BALANCE CLIENT/SERVER ROLES

nuxeo

"one-size-fits-all" does not work

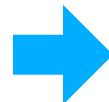


Client driven

CLIENTS

Ask for the data they need
Use custom API

"open bar" seems too messy



Server controlled

SERVER

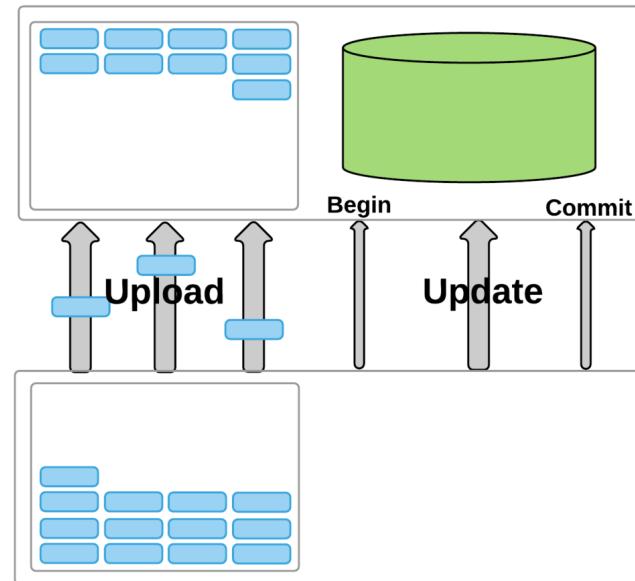
Manage the meta-model
Choose what API is exposed
(*versioned software artifact*)

BLOB FRIENDLY

nuxeo

- Chunked & Out of band upload

Content-Type: multipart/mixed

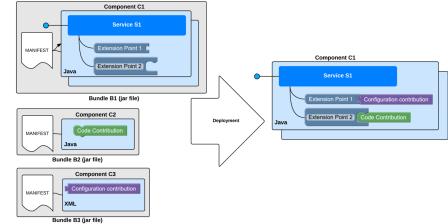


- Cachable and Seekable download

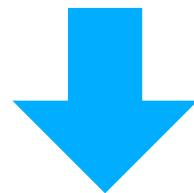
BE SENSIBLE

nuxeo

- Do not **lose our soul**
 - *fight to keep the dynamicity of the platform!*
- No REST integrism
 - *Useful is more important than Beautiful*
- Dogfooding is key
 - *if this is not good enough internally, this is not good*
- Building API is **part of the development cycle**
 - *adding http API should never be a task for later*



PRINCIPLES



BUILD SOMETHING THAT WORKS

DISCLAIMER

nuxeo



**THIS FILM HAS BEEN MODIFIED
FROM ITS ORIGINAL VERSION.
IT HAS BEEN FORMATTED TO
FIT YOUR TV.**

Actually, just the chronology has been adjusted !



BUILDING THE REST API

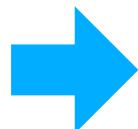
Exposing Resources



EXPOSING RESOURCES

Expose Use Cases !?

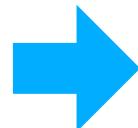
PLATFORM



Target use cases are not defined

Expose the Domain Model !?

CONFIGURABLE



Target Domain Model is unknown

Expose raw technical resources !

EXPOSE SIMPLE RESOURCES

Expose raw resources as EndPoint with REST Bindings

Documents

```
GET /repo/{repoId}/path/{docPath} HTTP 1.1
```

```
GET /repo/{repoId}/id/{docId} HTTP 1.1
```

Users & Groups

```
GET /user/{userName} HTTP 1.1
```

```
GET /group/{groupName} HTTP 1.1
```

```
GET /directory/{directoryName}/{entryId} HTTP 1.1
```

Tasks & Workflows

```
GET /workflowModel/{modelName} HTTP 1.1
```

```
GET /workflow/{workflowInstanceId} HTTP 1.1
```

```
GET /task/{taskId} HTTP 1.1
```

EXPOSE SIMPLE RESOURCES

path : Access documents by their path

Show/Hide List Operations Expand Operations Raw

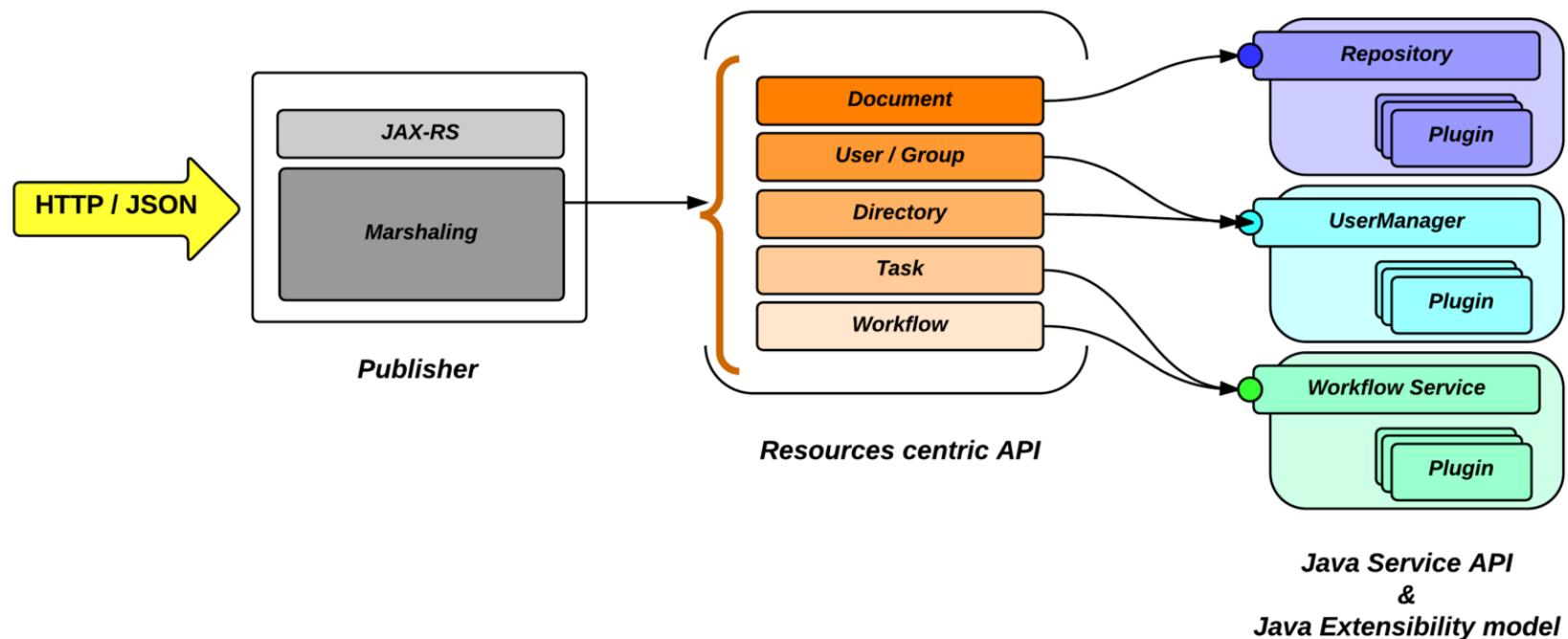
GET	/path/{docPath}	Find a document by its path
PUT	/path/{docPath}	Updates a document by its path
DELETE	/path/{docPath}	Deletes a document by its path
POST	/path/{docPath}	Creates a document by its parent path
GET	/repo/{repoid}/path/{docPath}	Find a document by its path
PUT	/repo/{repoid}/path/{docPath}	Updates a document by its path
DELETE	/repo/{repoid}/path/{docPath}	Deletes a document by its path
POST	/repo/{repoid}/path/{docPath}	Creates a document by its parent path

id : Access documents by their id

Show/Hide List Operations Expand Operations Raw

EXPOSE SIMPLE RESOURCES

nuxeo



GET A DOCUMENT

nuxeo

```
GET /nuxeo/api/v1/path/movies/star-wars HTTP/1.1
```

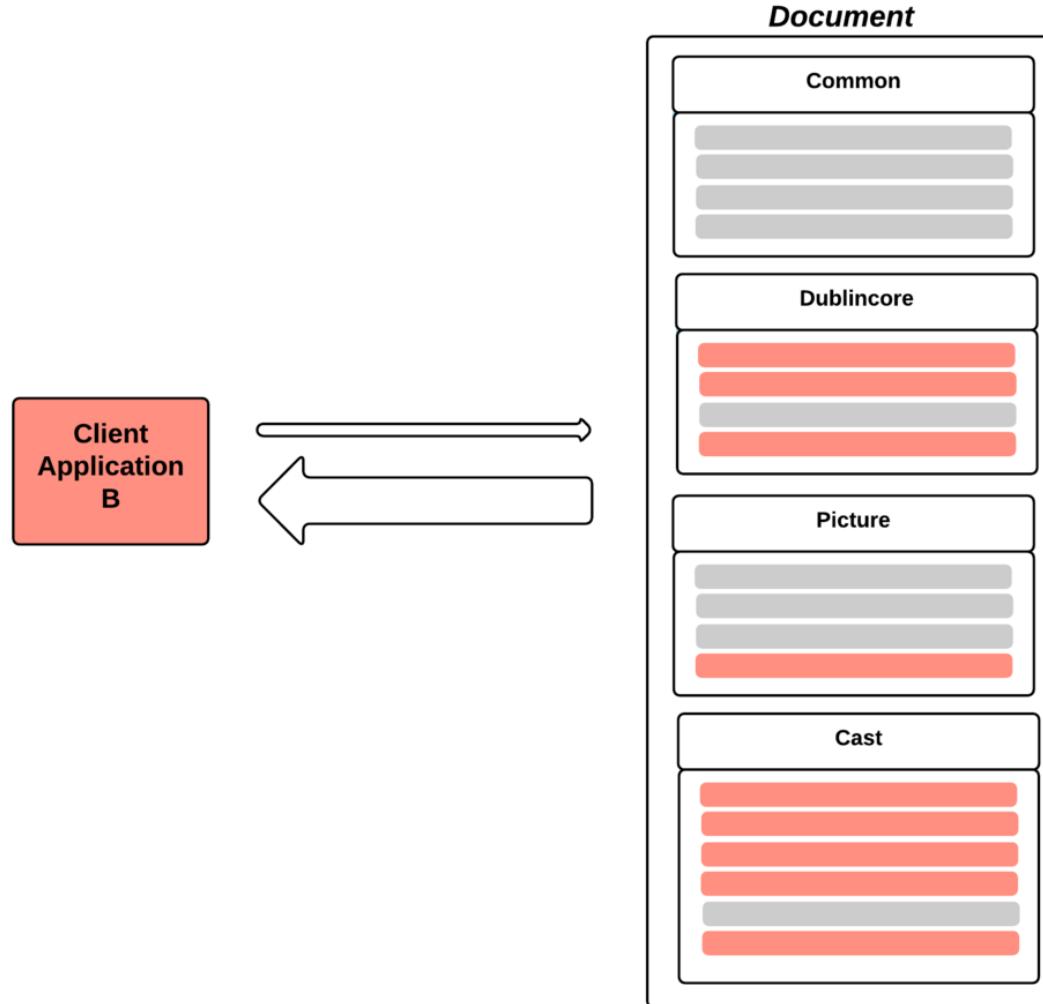
```
{  
  "entity-type": "document",  
  "repository": "default",  
  "uid": "5b352650-e49e-48cf-a4e3-bf97b518e7bf",  
  "path": "/movies/star-wars",  
  "type": "MovieCollection",  
  "isCheckedOut": true,  
  "title": "Star Wars",  
  "facets": [  
    "Folderish"  
  ]  
}
```

Server returns a minimal payload

ADAPTATIVE MARSHALING

nuxeo

Client need to control what **data schemas** are sent



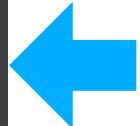
ADAPTATIVE MARSHALING

- Control what data schemas are sent to the client

```
GET /nuxeo/api/v1/path/movies/star-wars HTTP/1.1
```

```
X-NXProperties dublincore, common
```

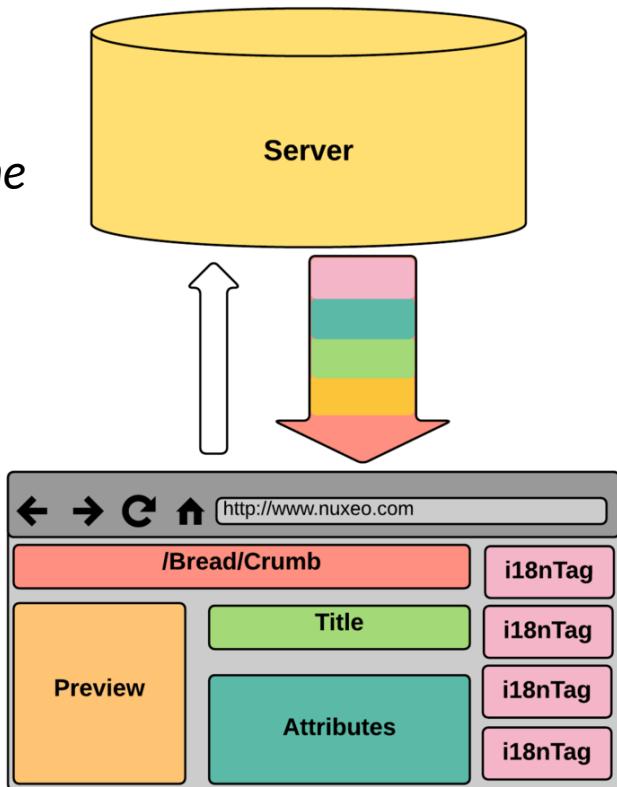
```
{  
    "entity-type": "document",  
    "repository": "default",  
    "uid": "5b352650-e49e-48cf-a4e3-bf97b518e7bf",  
    "path": "/movies/star-wars",  
    "type": "MovieCollection",  
    "isCheckedOut": true,  
    "title": "Star Wars",  
    "properties": {  
        ...  
        "common:icon": "/icons/movieCollection.png",  
        "dc:description": "Star Wars collection",  
        "dc:creator": "tiry",  
        "dc:modified": "2015-10-22T02:12:59.07Z",  
        "dc:lastContributor": "tiry",  
        "dc:created": "2015-10-22T02:12:59.07Z",  
        "dc:title": "Star Wars",  
        ...  
        "dc:contributors": [tiry, "system"]  
    },  
    "facets": [  
        "Folderish"  
    ]  
}
```



FETCHING CONTEXTUAL DATA

nuxeo

- Client may require **more data**
 - *get Document **children** at the same time*
 - *get the **breadcrumb** data*
 - *get **thumbnail** or **preview url***
 - ...
- Client ask for the data
 - *using **Headers***
 - *using **Query String parameters***

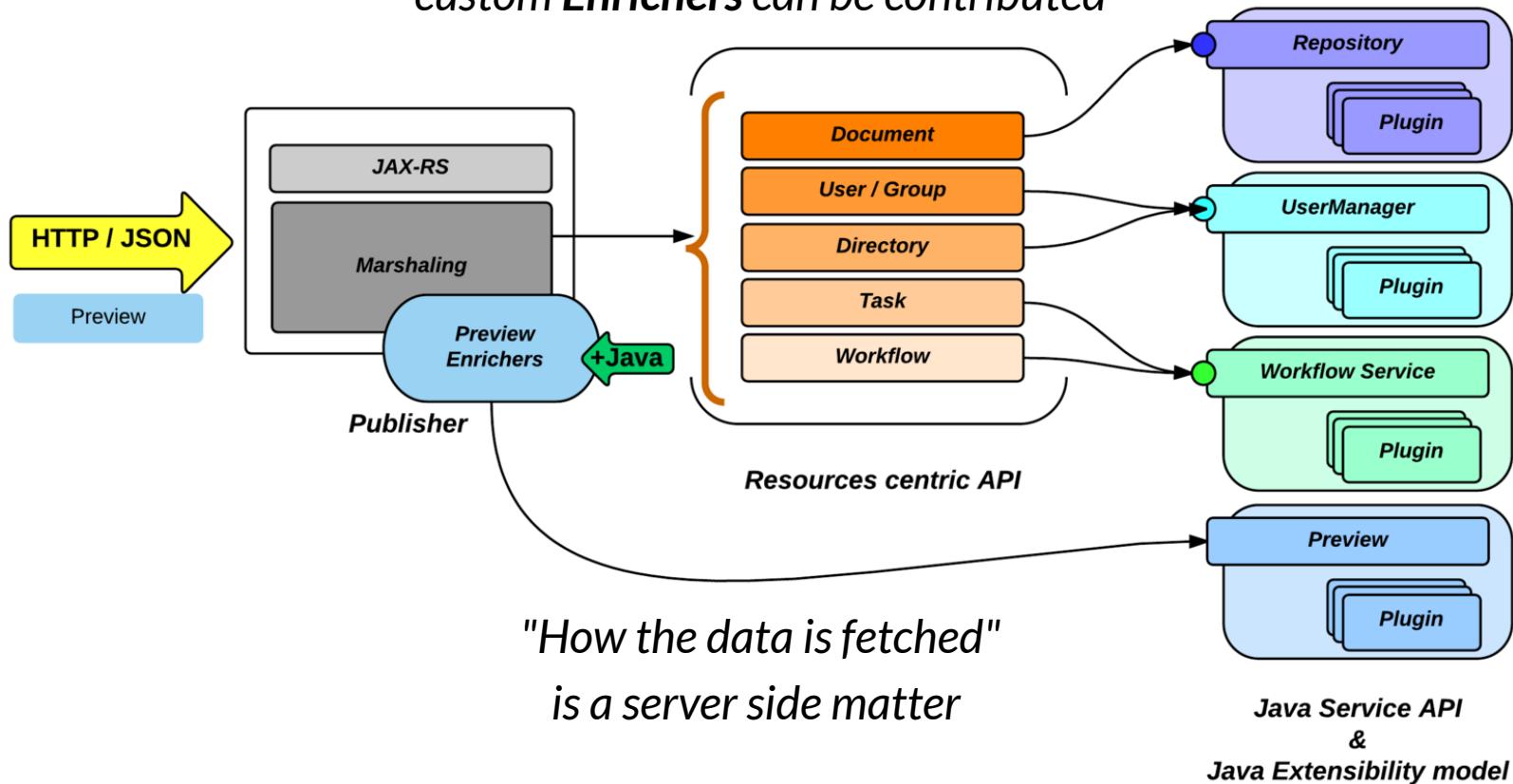


FETCHING CONTEXTUAL DATA

nuxeo

Marshaling registry is pluggable

*custom **Enrichers** can be contributed*



FETCHING CONTEXTUAL DATA

nuxeo

```
GET /nuxeo/api/v1/path/movies/star-wars?enrichers.document=thumbnail H
```

```
GET /nuxeo/api/v1/path/movies/star-wars HTTP/1.1
```

```
X-NXenrichers.document: thumbnail
```

```
{  
    "entity-type": "document",  
    "repository": "default",  
    "uid": "5b352650-e49e-48cf-a4e3-bf97b518e7bf",  
    "path": "/movies/star-wars",  
    "type": "MovieCollection",  
    "isCheckedOut": true,  
    "title": "Star Wars",  
    "contextParameters":  
    {  
        "thumbnail":  
        {  
            "url": "/nuxeo/nxthumb/default/5b352650-e49e-48cf-a4e3-bf97b518e7bf"  
        }  
    },  
    "facets": [  
        "Folderish"  
    ]  
}
```

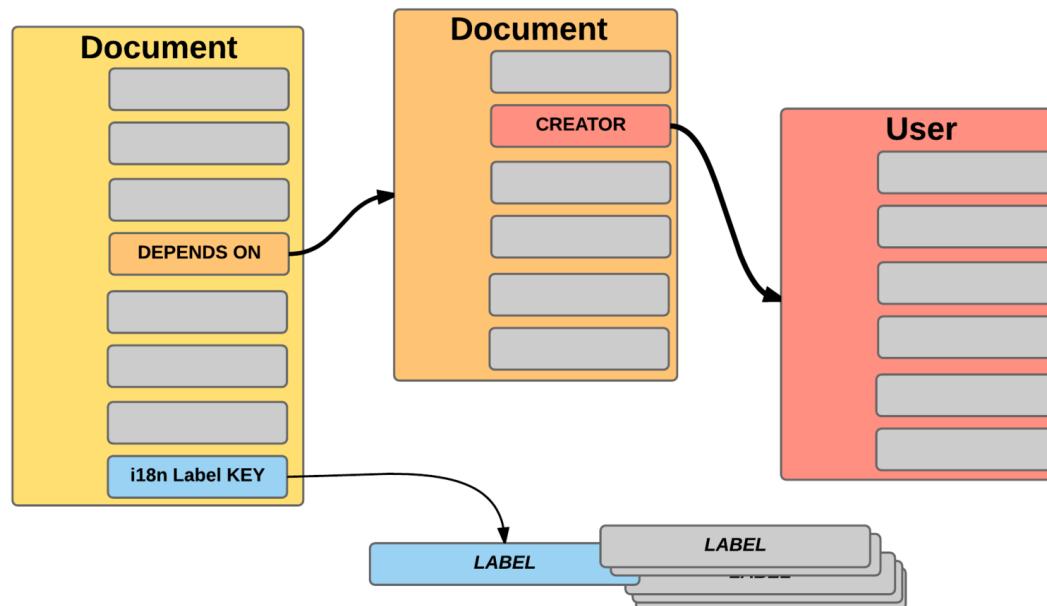


RETRIEVE LINKED DATA

- Resolve entity fields

- *pointing to a label*
- *pointing to an other Document*
- *pointing to a User*
- ...

Implicit JOIN



RETRIEVE LINKED DATA

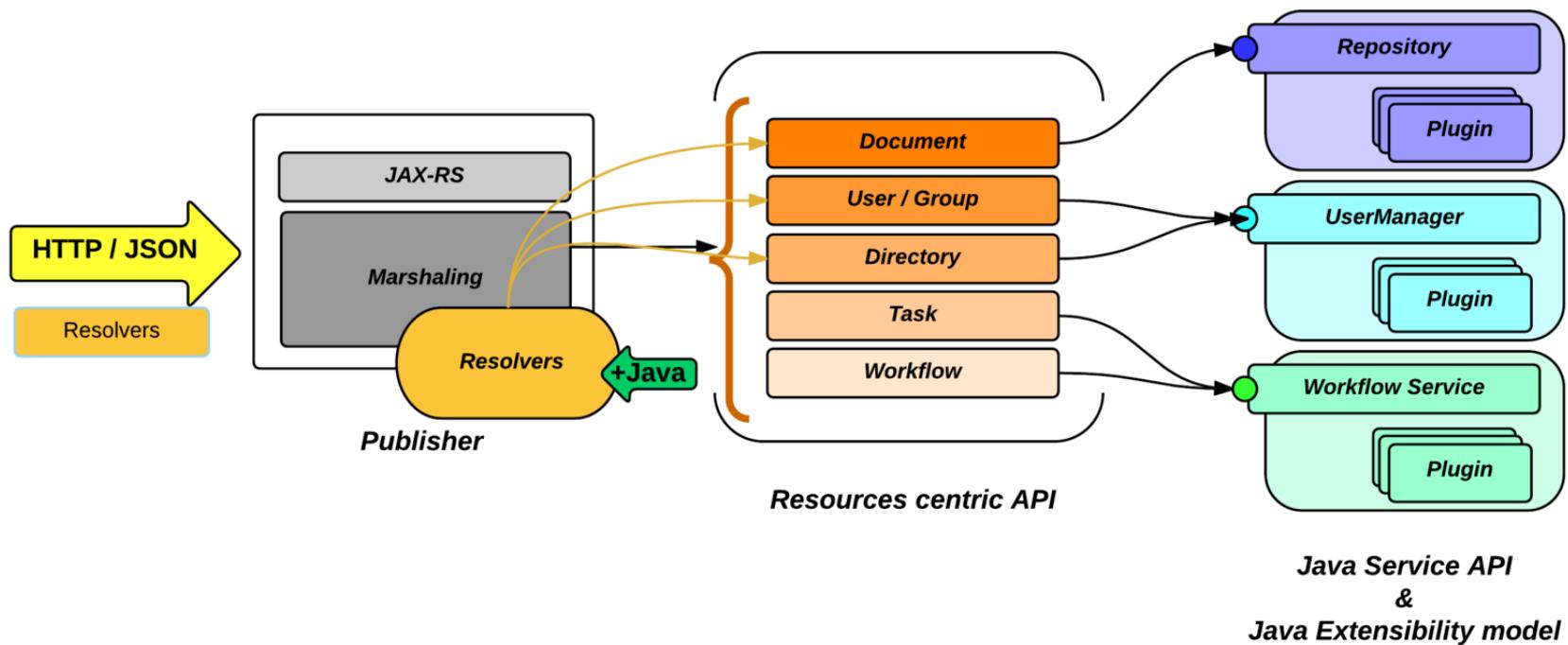
- Use client side parameter to know what to resolve
 - *header*
 - *QueryString parameter*

```
fetch.objectType=fieldToFetch  
  
translate.objectType=fieldToTranslate  
  
depth=children
```

- Can be **recursive**
 - *client need to control that too!*

RETRIEVE LINKED DATA

nuxeo



ADAPTERS



- Change the **return type**
 - *get only ACLs or History info about the Document*
 - *get the tasks associated to document*

```
GET /nuxeo/api/v1/path/movies/star-wars@acl HTTP/1.1
```

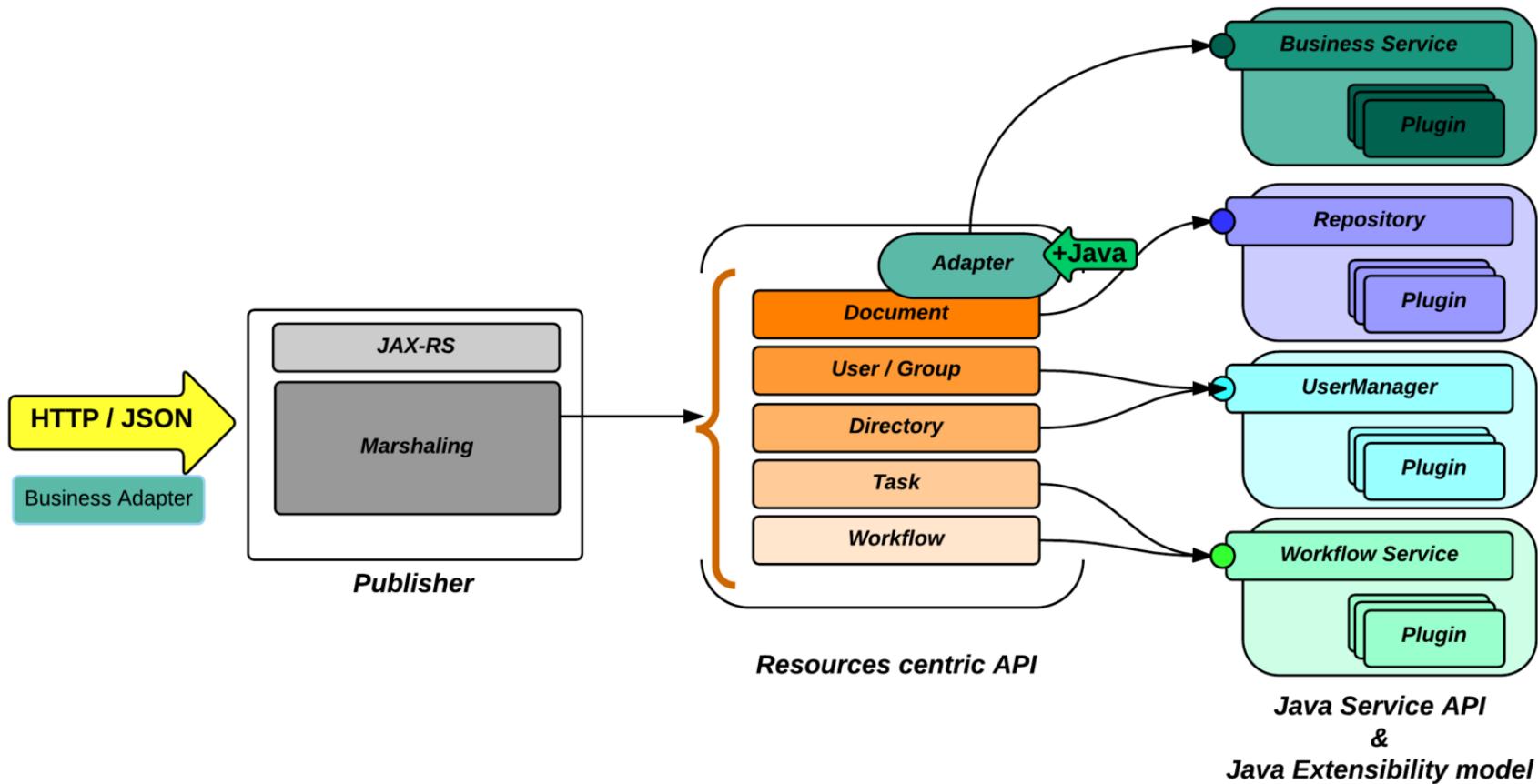
```
GET /nuxeo/api/v1/path/movies/star-wars@audit HTTP/1.1
```

- Use your own **business object**
 - use **business Adapters**
 - wrap document or documents
 - provide custom marshaling

```
GET /nuxeo/api/v1/path/movies/star-wars@bo/MyBusinessObject HTTP/1.1
```

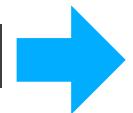
ADAPTERS

nuxeo

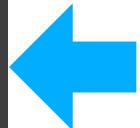


ADAPTERS

```
GET /nuxeo/api/v1/path/movies/star-wars@bo/MovieCollection HTTP/1.1
```



```
{  
  entity-type: "MovieCollection"  
  id: "5b352650-e49e-48cf-a4e3-bf97b518e7bf",  
  "title": "Star Wars"  
  "episodes": 7  
}
```



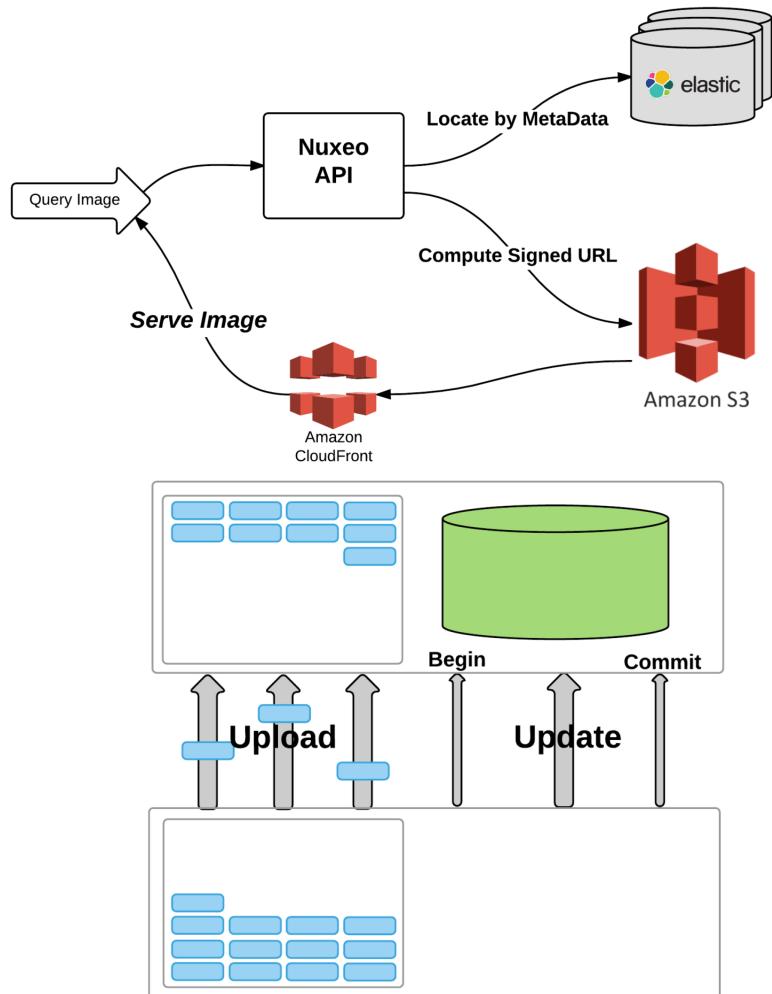
BLOBS

nuxeo

- Sent as links

- Digest
- CDN

- Uploaded out-of-band
 - chunking
 - reference in JSON



BLOB UPLOAD



- Upload EndPoint

```
POST /api/v1/upload/{batchId}/{fileIdx} HTTP/1.1
X-Upload-Chunk-Index 0
X-Upload-Chunk-Count 5
```

- Reference Blobs from JSON Payload

```
PUT /nuxeo/api/v1/path/movies/star-wars HTTP/1.1
```

```
{"entity-type": "document",
  "properties": {
    {
      "file:content" : {
        "upload-batch" : "0b0061d48f69b072",
        "upload-fileId" : 0,
        "type" : "blob"
      }
    }
  }
}
```

ARE WE HAPPY WITH THAT ?

nuxeo



- **Efficiency**
 - *we can get all data in one call*



- **Flexibility**
 - *we can configure the data we want*



- **Extensibility: partial**
 - *enrichers, resolvers & adapters are not always enough*



- **Coverage: poor**
 - *100+ services and only 5 endpoints*
 - *not everything is CRUD*

NEED A WAY TO MAP 100+ SERVICES

Without creating 100 endpoints!

Need an other paradigm !



AUTOMATION API

Exposing service API over HTTP



PRINCIPLES

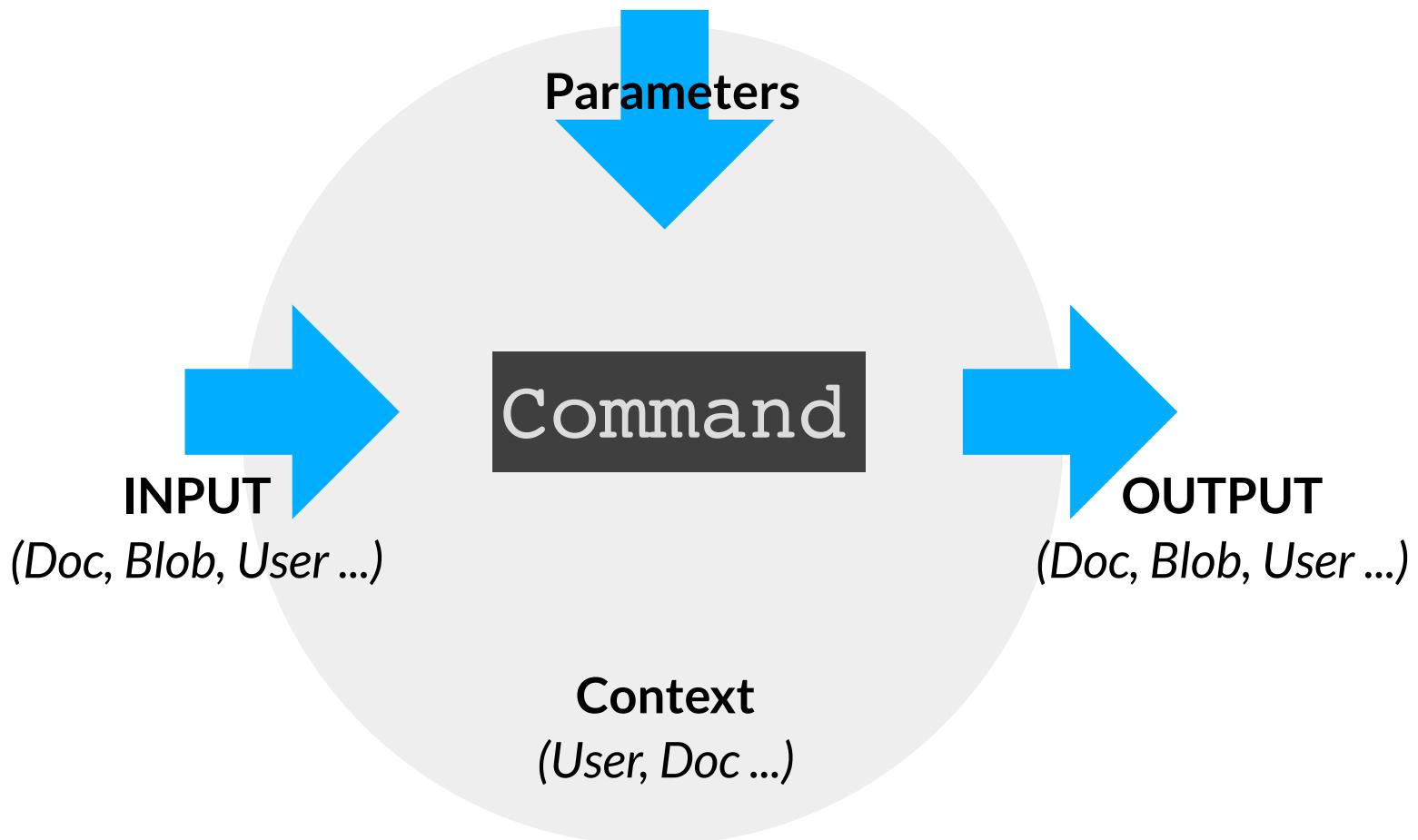


- Build a **coarse gained API** on top of service Java API
 - select simple Commands
- **Shell like** commands !
 - *each service can contribute*

```
> commandA(p1,p2) | commandB(p3,p4)
```

COMMAND SYNOPSIS

nuxeo



COMMANDS

WebUI.AddErrorMessage WebUI.AddInfoMessage WebUI.AddMessage Document.AddPermission Document.AddToCollection Document.MultivaluedProperty.addItem
Task.ApplyDocumentMapping Blob.AttachOnDocument BlobHolder.AttachOnCurrentDocument AttachFiles Audit.QueryWithPageProvider Blob.ImportClipboard
d Blob.ImportWorklist Blob.RunConverter Document.BlockPermissionInheritance WorkflowModel.PullRestartInstances Business.BusinessCreateOperation
Business.BusinessFetchOperation Business.BusinessUpdateOperation Navigation.GoBack WorkflowInstance.Cancel Navigation.ChangeCurrentTab Document
t.CheckIn Document.CheckOut Update.WFISetCurrentWorkflow Document.CheckoutWorkflow Document.CheckoutWorkflow SelectedDocuments WebUI.ClearWorklist WorkflowTa
sk.Complete Blob.ConcatenatePDFs Context.FetchDocument Context.FetchFile Blob.ToPDF Blob.Convert Document.Copy Document.Create FileManager
.Import UserWorkspace.CreateDocumentFromBlob Seam.CreateDocumentInUI Picture.Create Document.CreateLiveProxy Document.AddRelation Collection.C
reate Workflow.CreateRoutingTask Task.Create Directory.CreateEntries Document.Delete Document.DeleteRelation Directory.DeleteEntries WebUI.D
estroySeamContext Repository.GetDocument Document.Export WebUI.DownloadFile Blob.ExportToFS Document.FetchByProperty Blob.CreateFromURL File
Manager.ImportInSeam FileManager.ImportWithMetaData FileManager.ImportWithMetaDataInSeam Document.Filter Document.FollowLifecycleTransition Comm
ent.Moderate Document.GetBlobs Document.GetChildren Document.GetChildrenInCollection Document.GetBlobsByProperty User.GetUserWorkspace Doc
ument.GetLinkedDocuments Proxy.GetSourceDocument User.Get Document.GetParent Context.GetEmailsWithPermissionOnDoc Context.GetTaskNames Context
.GetUsersGroupIdsWithPermissionOnDoc Document.GetVersions Directory.Projection Collection.Suggestion User.GetCollections Directory.Entries Dir
ecitory.SuggestEntries Collection.GetDocumentsFromCollection Favorite.GetDocuments Document.Routing.GetGraph Picture.GetView Workflow.GetOpenTasks
Tag.Suggestion Task.GetAssigned UserGroup.Suggestion Document.GetRendition Blob.PostToURL Image.Blob.Resize WebUI.InitSeamContext JsonSt
ack.ToggleDisplay Actions.GET GetRepositories Document.Lock Log Audit.LogEvent Auth.LoginAs Auth.Logout Document.Move Document.PublishTo
Sections NRD-AC-PR-ChooseParticipants-Output NRD-AC-PR-LockDocument NRD-AC-PR-ValidateNode-Output NRD-AC-PR-force-valid
ate NRD-AC-PR-storeTaskInfo WebUI.NavigateTo NuxeoDrive.SetActiveFactories NuxeoDrive.AddToLocallyEditedCollection NuxeoDrive.AttachBlob Nuxeo
Drive.CanMove NuxeoDrive.CreateFile NuxeoDrive.CreateFolder NuxeoDrive.CreateTestDocuments NuxeoDrive.Delete NuxeoDrive.FileSystemItemExists N
uxeoDrive.GenerateConflictedItemName NuxeoDrive.GetRoots NuxeoDrive.GetChangeSummary NuxeoDrive.GetChildren NuxeoDrive.GetClientUpdateInfo Nuxeo
Drive.GetFileSystemItem NuxeoDrive.GetTopLevelFolder NuxeoDrive.GetTopLevelChildren NuxeoDrive.Move NuxeoDrive.SetSynchronization NuxeoDrive.Ren
ame NuxeoDrive.SetVersioningOptions NuxeoDrive.SetupIntegrationTests NuxeoDrive.TearDownIntegrationTests NuxeoDrive.UpdateFile NuxeoDrive.WaitFo
rElasticsearchCompletion NuxeoDrive.WaitForAsyncCompletion Repository.PageProvider Context.PopDocument Context.PopDocumentList Context.PopBlob
Context.PopBlobList Document.PublishToSection Context.PushDocument Context.PushDocumentList Context.PushBlob Context.PushBlobList Context.Pus
hDocument Context.PushDocumentList Context.PushBlob Context.PushBLOBList WebUI.AddClipboard WebUI.PushDocumentToSeamContext WebUI.AddToWorkl
ist LocalConfiguration.PutSimpleConfigurationParameters LocalConfiguration.PutSimpleConfigurationParameter Repository.Query Audit.Query Reposito
ry.ResultSetPageProvider WebUI.RaiseSeamEvents Blob.ReadMetadata Context.SetMetadataFromBlob Directory.ReadEntries WebUI.Refresh WebUI.Refresh
Document.RemoveACL Services.RemoveDocumentTags Document.RemoveEntryOfMultivaluedProperty Blob.RemoveFromDocument Document.RemovePermission Do
cument.RemoveProperty Collection.RemoveFromCollection Render.Document Render.DocumentFeed TemplateProcessor.Render Document.ReplacePermission
Document.Reload Picture.Resize Context.RestoreDocumentInput Context.RestoreDocumentsInput Context.RestoreBlobInput Context.RestoreBlobsInput D
ocument.RestoreVersion Context.RestoreBlobInput Document.RestoreDocumentInput Context.RestoreDocumentInputFromScript Context.Resta
reDocumentsInputFromScript Repository.ResultsetQuery Document.Routing.ResumeStep Workflow.ResumeNode Counters.GET RunOperation RunDocumentOpe
ration Context.RunDocumentOperationInNewTx RunFileOperation RunOperationOnList RunOperationOnProvider RunOperationOnListInNewTx RunInputScript
RunScript WebUI.RunOperationInSeam Document.Save Seam.SaveDocumentInUI Repository.SaveSession SeamActions.GET Document.Mail Event.Fire
Document.AddACE Context.SetVar Context.SetInputAsVar LocalConfiguration.SetSimpleConfigurationParameterAsVar Document.Routing.SetRunningStepFromTa
sk Document.SetBlob Document.SetBlobName WebUI.SetJSFOutcome Workflow.SetNodeVariable Document.Routing.Step.Done Document.Routing.BackToReady
Document.Routing.EvaluateCondition Context.SetWorkflowVar WebUI.ShowcreateForm Document.CreateVersion Context.StartWorkflow Search.SuggesterSL
auncher Services.TagDocument Traces.Get Traces.ToggleRecording Document.SetMetadataFromBlob Seam.GetChangeableDocument Seam.FetchFromClipboard
Seam.GetCurrentDocument Seam.GetListeDocuments Seam.GetUserWorkflowDocument Seam.GetSelectedDocuments Seam.GetDocumentsFromSele
ctionList Seam.FetchFromWorklist Document.Lock Document.Unlock Document.Update Document.Update Document.SetProp
erty Document.Routing.UpdateCommentsInfoOnDocument Directory.UpdateEntries Workflow.UserTaskPageProvider VersionAndAttachFile VersionAndAttachFi
les Blob.SetMetadataFromDocument Blob.SetMetadataFromContext Blob.CreateZip acceptComment addCurrentDocumentToWorklist blobToPDF cancelWorkf
low conditionalTask decideNextStepAndSimpleValidate downloadFilesZip evaluateCondition followLifeCycleTransition followLifeCycleTransitionTask
initInitiatorComment logInAudit nextAssignee notifyInitiatorEndOfWorkflow publishDocument publishTask reinitAssigneeComment rejectComment
Workflow.RemoveRoutingTask sendTaskCreatedNotificationMail setDone setNextStep setTaskDone simpleChooseNextOption1AndDone simpleChooseNextO
ption2AndDone simpleRefuse simpleTask simpleUndo simpleValidate terminateWorkflow undoRunningTask updateCommentsOnDoc validateDocument v
oidChain xmlExportRendition zipTreeExportRendition

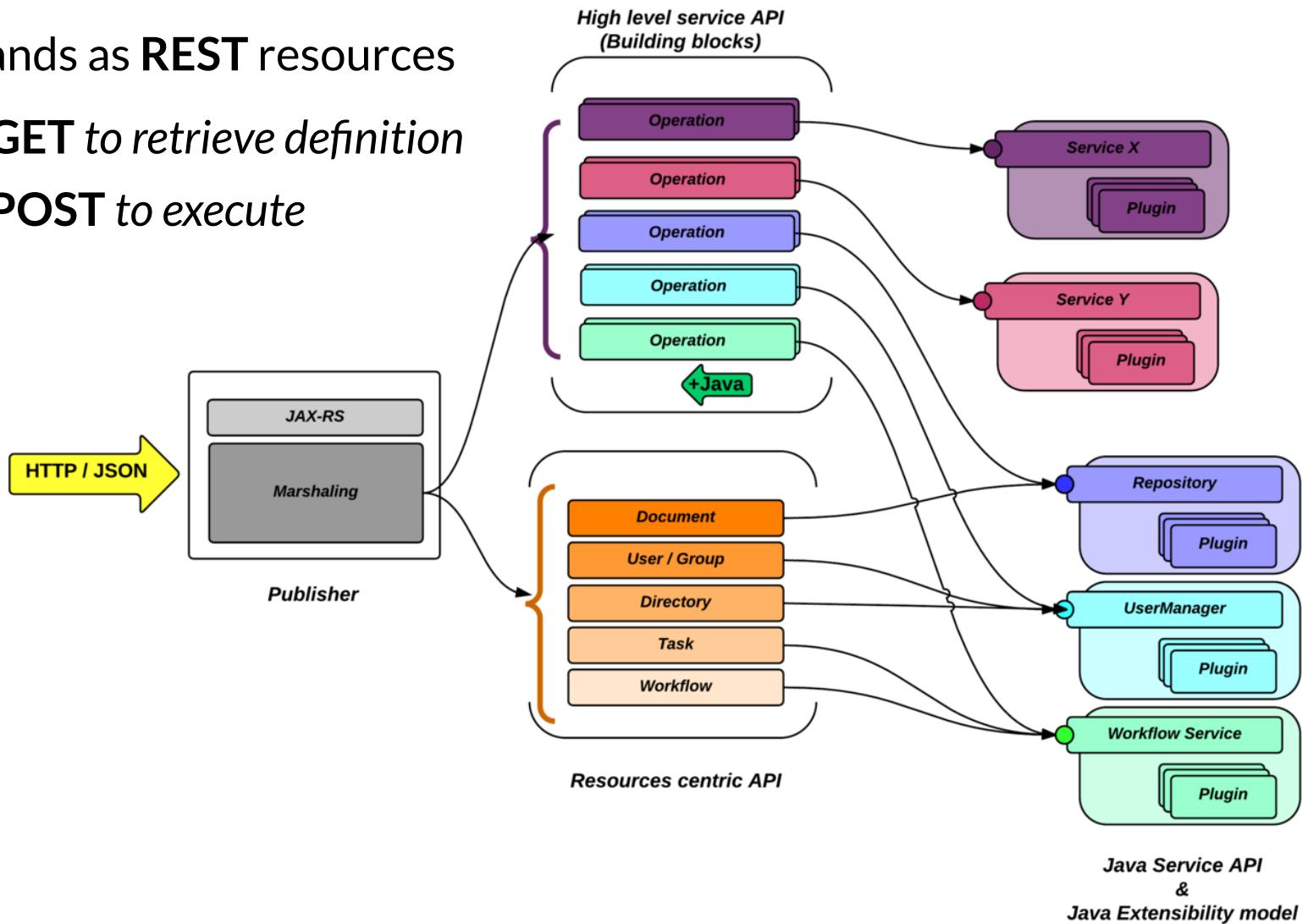
lot of contributed operations

PRINCIPLES

nuxeo

Commands as REST resources

- **GET** to retrieve definition
- **POST** to execute



GET AN OPERATION



```
GET /nuxeo/api/v1/automation/Document.PageProvider HTTP/1.1
```

```
HTTP/1.1 200 OK
```

```
Content-Type: application/json
```

```
{  
    "id": "Document.PageProvider",  
    "label": "PageProvider",  
    "description": "Perform a query ...",  
    "signature": [ "void", "documents" ],  
    "params": [  
        { "name": "page",  
          "type": "integer",  
          "required": false  
        }, {  
            "name": "query",  
            "type": "string",  
            "required": false, },  
            ... ]  
}
```

GET AN OPERATION



Operation Repository.PageProvider (PageProvider)

Description

Perform a query or a named provider query on the repository. Result is paginated. The query result will become the input for the next operation. If no query or provider name is given, a query returning all the documents that the user has access to will be executed.

General Information

Operation id: Repository.PageProvider

Aliases: [Document.PageProvider]

Category: Fetch

Label: PageProvider

Requires:

Since:

Parameters

Name	Description	Type	Required	Default value
currentPageIndex		integer	no	
documentLinkBuilder		string	no	
language		string	no	NXQL
maxResults		string	no	
namedParameters	Named parameters to pass to the page provider to fill in query variables.	properties	no	
page		integer	no	
pageSize		integer	no	
providerName		string	no	
query		string	no	
queryParams		stringlist	no	
sortBy	Sort by properties (separated by comma)	string	no	
sortInfo		stringlist	no	
sortOrder	Sort order, ASC or DESC	string	no	ASC, DESC

Signature

Inputs: void

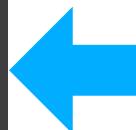
Outputs: documents

RUN AN OPERATION

```
POST /nuxeo/api/v1/automation/Document.PageProvider HTTP/1.1
Content-Type: application/json+nxrequest
{
  "params" :
    {
      "query" : "select * from Note",
      "page" : 0
    }
}
```



```
HTTP/1.1 200 OK
Content-Type: application/json
{
  "entity-type": "documents",
  "pageIndex": 0,
  "pageSize": 2,
  "pageCount": 2,
  "entries": [
    {
      "entity-type": "document",
      "repository": "default",
      "uid": "3f76a415-ad73-4522-9450-d12af25b7fb4",
      ...
    },
    { ... },
    ...
  ]
}
```



RESOURCES & AUTOMATION

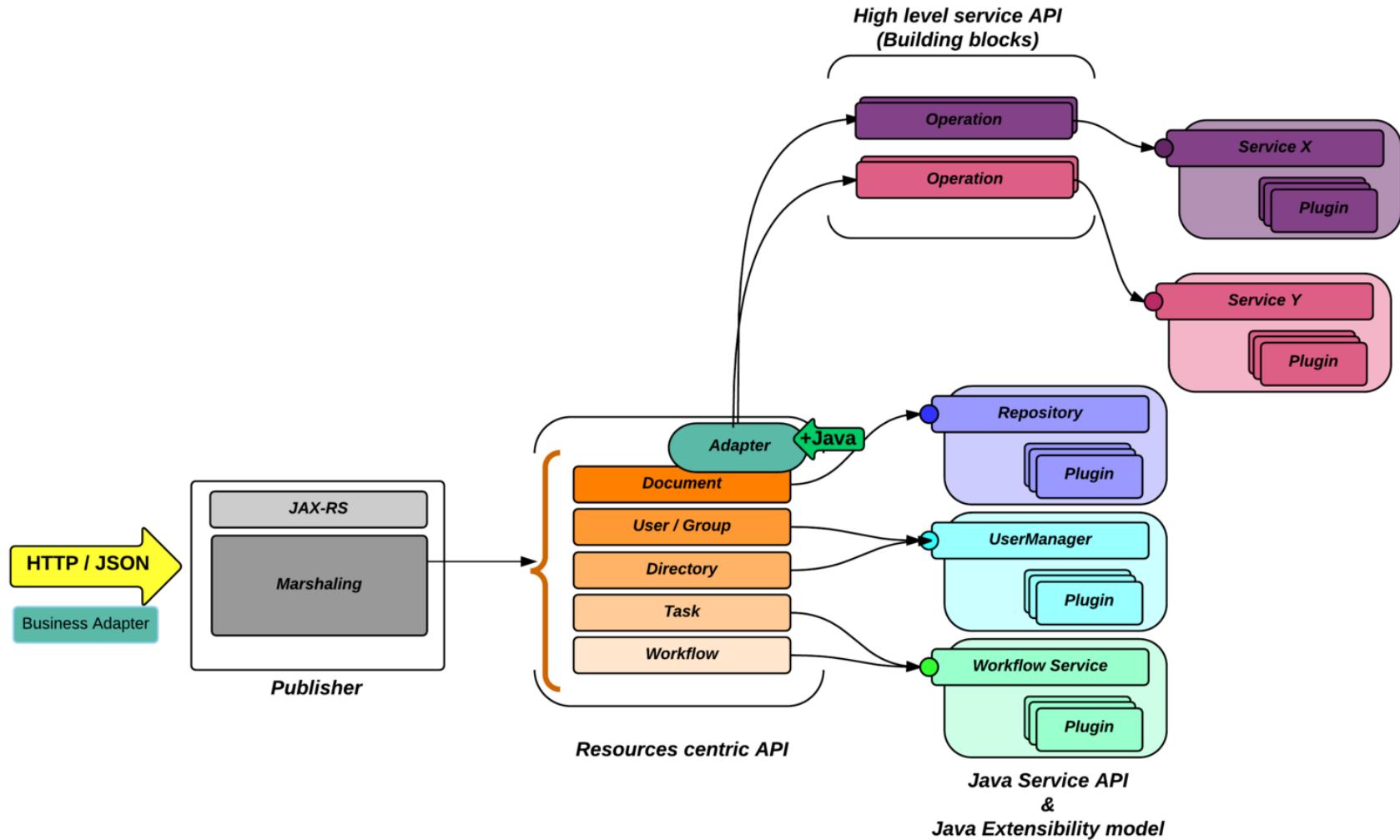


- Share the **marshaling layer** and **extension**
 - *Enrichers, Resolvers are available too*
- Compose Resources and Automation API
 - *Pipe Resources as input for Automation Operation*

```
> cat /doc/path/somedoc | command(p3,p4)
```

RESOURCES & AUTOMATION

nuxeo



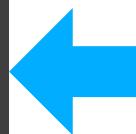
RESOURCES > AUTOMATION

nuxeo

```
POST /nuxeo/api/v1/path/somePath/@op/Blob.ToPDF HTTP/1.1
```



```
HTTP/1.1 200 OK  
Content-Type: application/pdf  
...
```



ARE WE HAPPY WITH THAT ?

nuxeo



- Efficiency
- Flexibility
- Coverage
 - *all services and plugins can contribute*



- Extensibility
 - *good, but limited to Java Developers*



- Consistency
 - *still no way to align Application Transactions*

MORE COMPOSITION

assemble API blocks without having to code

build business API



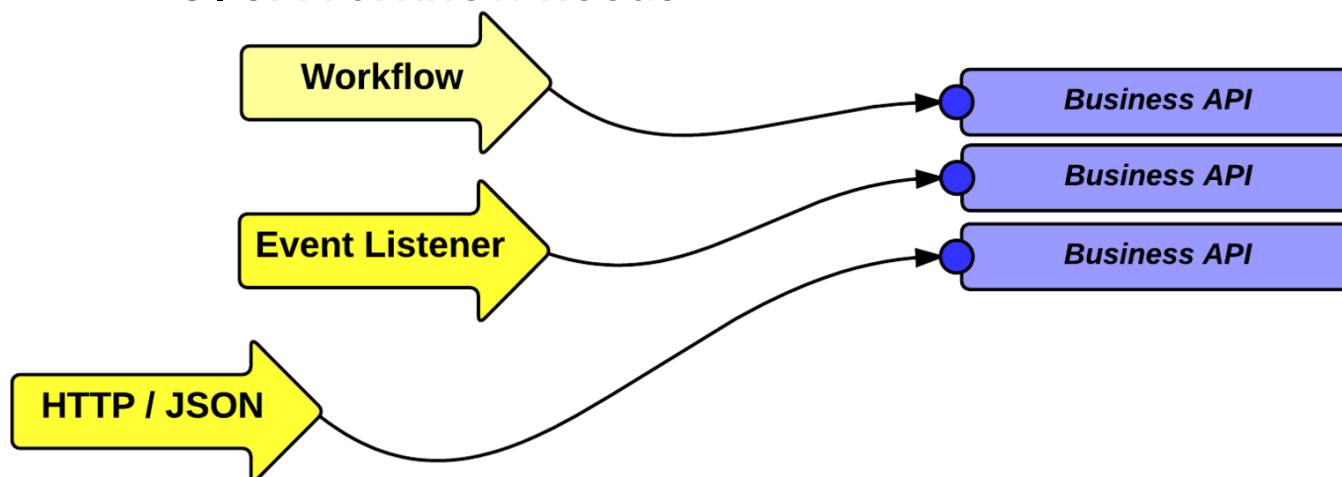
COMPOSABLE API

Expose the API that matches client needs



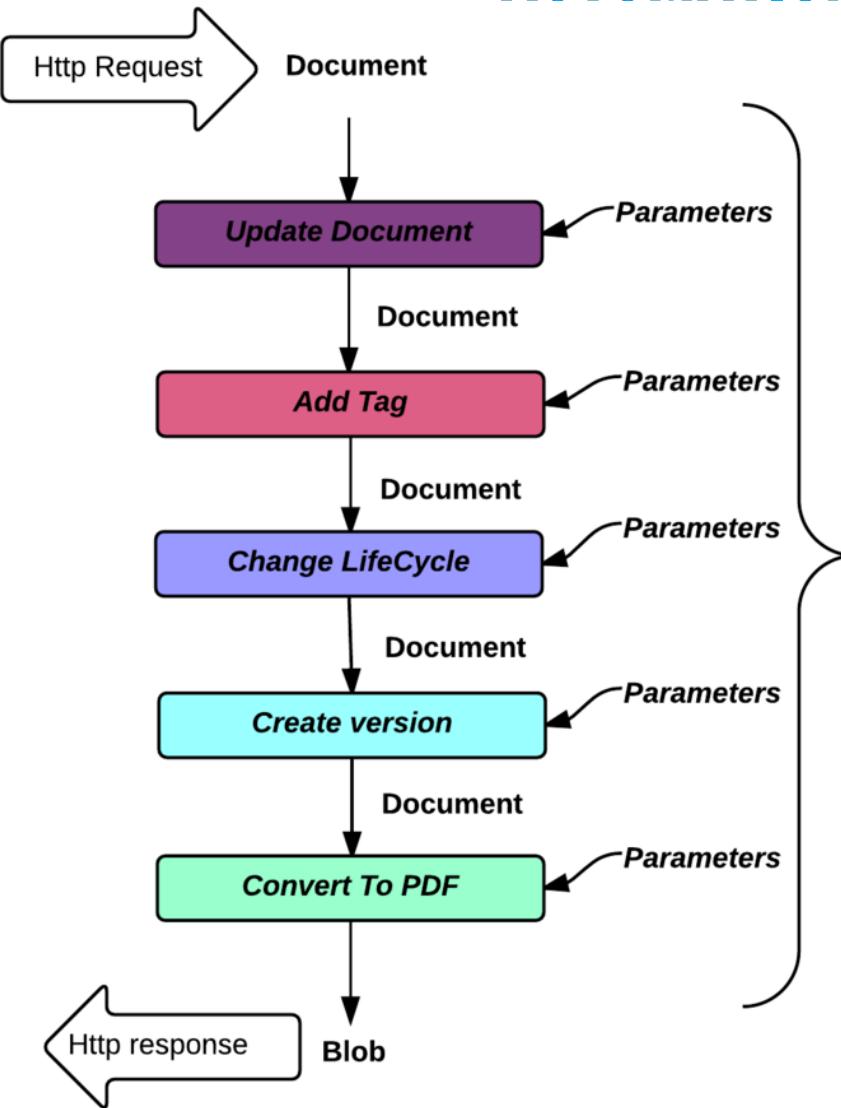
GOALS

- Tailor the API to match application requirements
 - *one API behind every action / button*
- Allow **business analysts** or **UI developers** to tailor the API
 - define what API is exposed
 - UI & Workflow needs



AUTOMATION CHAIN

nuxeo



- **Assemble** operations in a **chain**
- **Pipe** Output / Input
- Give it a **name**
- Call and execute within a **single transaction**

**One
Transaction**

One Context

Server side assembly

ASSEMBLING CHAINS

nuxeo

Chain editor ? Chain parameters ?

▶ Documentation

- Conversion
- Document
- Execution Context
- Execution Flow
- Fetch
- Files
- Local Configuration
- Notification
- Push & Pop
- Scripting
- Services
- User Interface
- Users & Groups
- Workflow Context

Operation	Add
Blob.RunConverter	Add
Concatenate PDFs	Add
Convert To PDF	Add
Convert to given mime-type	Add
Render Document	Add
Render Document Feed	Add

The context document represents either the current document in the user interface or the target document of a repository event.

Edit all operations

Fetch > Context Document(s)	Edit	x	?
Document > Update Property	Edit	x	?
XPath	dc:description		
save	true		
value	Changed		
Services > Tag Document	Edit	x	?
tags	Changed		
Document > Follow Life Cycle Transition	Edit	x	?
Value	validate		
Document > Snapshot Version	Edit	x	?
increment	Minor		
saveDocument	true		
Conversion > Convert To PDF	Edit	x	?

Convert the input file to a PDF and return the new file.

Accepts: bloblist, blob, document

Produces: bloblist, blob, blob

[See Online Help](#)

Drop the next operation.

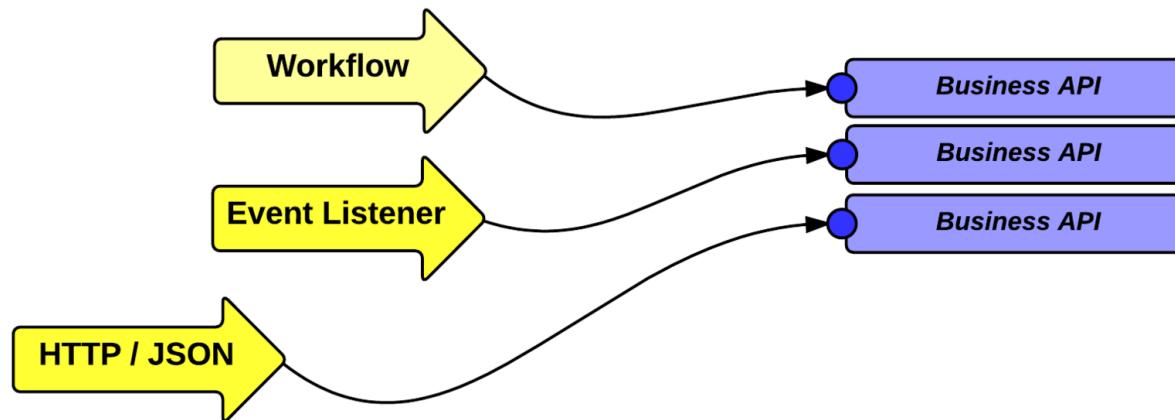
Discard Changes

Save

IT DOES WORK !

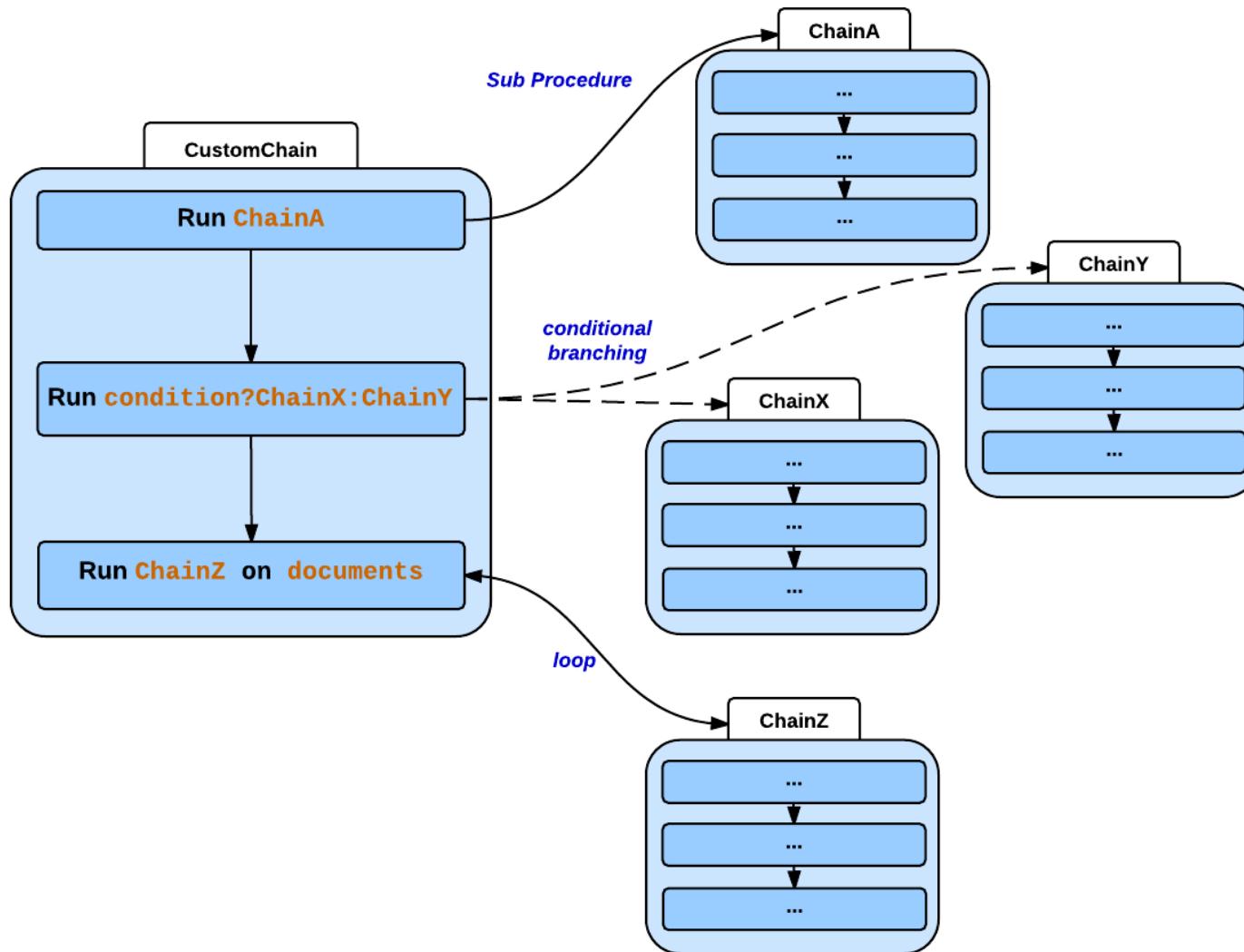
nuxeo

- Business users & Front end developers leverage this
 - to expose custom API for their UI
 - to build custom logic inside their Workflows
 - to add automatic processing (listeners)
- Actually it **works** almost too well
 - users do awfully complicated things
 - chains calling chains calling chains ...



HAVE WE CREATED A MONSTER ?

nuxeo



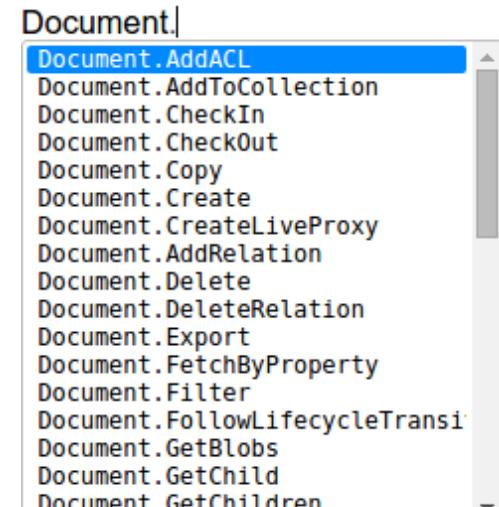
GO FURTHER THAN THE CHAIN MODEL

USE NASHORN TO ASSEMBLE OPERATIONS

- *Operations remain the building blocks*
- *JavaScript is the glue code to assemble them*

 *better control of the flow*

```
1 function run(input, params) {  
2  
3     var docs = Seam.GetSelectedDocuments(input, {  
});  
/* Description: Fetch the documents selected in the current folder listing */  
4  
5     if(docs.length>3){  
6         var index;  
7         for(index=0;index<docs.length;++index){  
8             Document SetProperty(input, {  
/*required:true - type: string*/  
'xpath': "dc:title",  
/*required:false - type: boolean*/  
'save': true,  
/*required:false - type: serializable*/  
'value': "test"  
});  
WebUI.Refresh(input, {});  
}  
else{  
    WebUI.AddMessage(input, {  
/*required:true - type: string*/  
'message': "DISPLAY IT",  
/*required:true - type: string*/  
'severity': "WARN"  
});  
}  
}  
}|
```



Nuxeo Automation Scripting : Blockly Editor

Blockly Editor Generated JavaScript XML Blocks Convert Automation Chains

operation name Filter

Control
Functions
Variables
Text
Math
Logic
▼ Automation
 Helpers
 Business
 Chain
 Conversion
 Document
 Execution Context
 Execution Flow
 Fetch
 Files
 Local Configuration
 Notification
 Push & Pop
 Routing
 Scripting
 Services
 User Interface
 Users & Groups
 Workflow Context

The screenshot shows the Nuxeo Automation Scripting Blockly Editor interface. On the left, there is a sidebar with a tree view of available operations categorized under 'Automation'. The 'Document' category is currently selected and highlighted in blue. The main workspace on the right displays a visual script composed of various blocks. The script starts with a 'for each item doc in list' loop. Inside the loop, there is a 'Document.Query' block with parameters: query = "select * from Document", currentPageIndex = 0, pageSize = 20. Below the loop, there is an 'if' condition block with the condition 'GetDocumentProperty type = "File"'. If the condition is true, it executes a 'do' block which contains a 'Document SetProperty' block. This 'SetProperty' block sets the property 'dc:nature' to 'This is a File' for the document 'doc'. The entire script is enclosed in a green border.

ARE WE HAPPY WITH THAT ?

nuxeo



- Efficiency



- Flexibility



- Coverage



- Extensibility



- Consistency

DYNAMIC AND COMPOSABLE API

What about the trade-off ?



DYNAMIC API COMES WITH A PRICE

nuxeo

- **Documentation** is a challenge
 - *maintain up to date and exhaustive*
- **Clients** needs to deal with the dynamic aspect
 - *data mapping*
 - *discover APIs*
- **Debugging** API calls can be challenging

Introspection is a requirement

INTROSPECTION

nuxeo

- Trace feature
 - Allow client to retrieve traces of nested executions (dev mode)
 - Introspect the Command API
 - GET *retrieve definition + doc*
 - Add REST API to introspect configuration resources
 - Document types, Schemas
 - (Widgets & Forms)
- Provide a **Playground** to test
- connect to a remote server
 - introspect server API and structures

Introspect Data Structures

nuxeo

<http://nuxeo.github.io/api-playground/>

nuxeo API Playground

The screenshot shows the nuxeo API Playground interface. The left sidebar has a 'Data Structures' tab selected, with a search bar and a list of schemas including 'Schemas', 'advanced_search', 'audio', 'authtoken', 'basicauditsearch', 'collection', 'collectionMember', 'comment', 'common', 'conditional_step_folder', and 'conditional_task_step'. The main content area shows the 'dublincore' schema details. The URL in the browser is [/Data Structures / Schemas / dublincore](#). The page title is 'dublincore'. A 'PREFIX: DC' button is present. A table lists the fields and their types:

Name	Type
contributors	null#anonymousType[]
coverage	coverage#anonymousType
created	date
creator	creator#anonymousType
description	string
expired	date
format	string
issued	date
language	string

Introspect Resources

nuxeo

<http://nuxeo.github.io/api-playground/>

nuxeo API Playground

Administrator, connected to <http://demo.nuxeo.com/nuxeo> [SIGN OUT](#)

Resources Endpoints

- ▶ Directory
- ▶ Group
- ▶ Id
- ▼ Path
 - GET** /path/{docPath}
 - PUT** /path/{docPath}
 - DELETE** /path/{docPath}
 - POST** /path/{docPath}
 - GET** /repo/{repoid}/path/{docPath}
 - PUT** /repo/{repoid}/path/{docPath}
 - DELETE** /repo/{repoid}/path/{docPath}
 - POST** /repo/{repoid}/path/{docPath}
- ▶ Query

[☰](#)

[/](#) / Resources Endpoints / Path / GET

GET /api/v1/path/{docPath}

Find a document by its path

Parameters

Name	Value	Description	Type
docPath	<input type="text"/> file	Path of the document, ex: 'default-domain'	string PATH

Click on Run for executing the request. You can also click on the "settings" icon for adding custom headers to your request. Then click on the "CURL request" tab for seeing the details of the request.

[⚙️](#) **RUN**

Introspect Command API

nuxeo

<http://nuxeo.github.io/api-playground/>

nuxeo API Playground

Command Endpoint

Administrator, connected to <http://demo.nuxeo.com/nuxeo> SIGN OUT

User Interface

Document

Add Permission

Add document to collection

Add entry into multi-valued metadata

Check In

Check Out

Copy

Create

Create Proxy Live

Create a collection

Document.COPY

POST /api/automation/Document.Copy

COPY

Copy the input document into the given folder. The name parameter will be used as the copy name otherwise if not specified the original name will be preserved. The target folder can be specified as an absolute or relative path (relative to the input document) as an UID or by using an EL expression. Return the newly created document (the copy).

Input

document

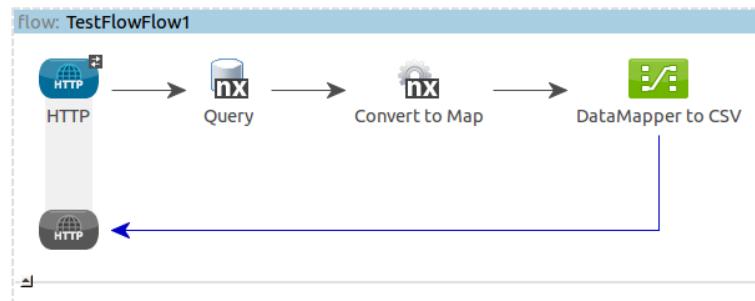
Parameters

Name	Value
target	
name	

INTROSPECTION IN ACTION

nuxeo

muleESB



The screenshot shows the Mule ESB DataMapper interface with the following details:

- Element Mapping:** Set to "For each Note --> Invoice".
- Left Panel (Input Elements):** Shows a list of attributes with their types:
 - dc_mouname : DateTime
 - dc_nature : String
 - dc_publisher : String
 - dc_rights : String
 - dc_source : String
 - dc_title : String
 - dc_valid : DateTime
 - ecm_id : String
 - ecm_path : String
 - ecm_repository : String
 - ecm_state : String
 - ecm_type : String
 - note_mime_type : String
 - note_note : String
 - uid_major_version : Long
 - uid_minor_version : Long
 - uid_uid : String
 - dc_contributors : dc_contributors
 - item : String** (highlighted with a red border)
 - dc_subjects : dc_subjects
- Middle Panel (Mapping Diagram):** A graph showing the mapping between input elements and output attributes. Arrows connect 'item : String' to several output attributes: CUSTOMER_NAME, ACCEPTED_PAYMENTS, CHANNEL, INVOICE-ID, CONTENT, TYPE, REGULATORY_COUNTRY, ACCOUNTANCY_CATEGORY, and AMOUNT.
- Right Panel (Output Arguments):** Shows the structure of the 'INVOICE : INVOICE' XML element, which contains attributes for CUSTOMER_NAME, ACCEPTED_PAYMENTS, CHANNEL, INVOICE-ID, CONTENT, TYPE, REGULATORY_COUNTRY, ACCOUNTANCY_CATEGORY, and AMOUNT.
- Bottom Panel:** Instructions for mapping:
 - Drag an Input attribute to an Output attribute to assign/concatenate.
 - Drag an Input element list to an Output one to map their attributes.

Using introspection from Mule ESB DataSense

CLIENTS LIBS

nuxeo

Dynamic API → more complex to use !?

client libraries



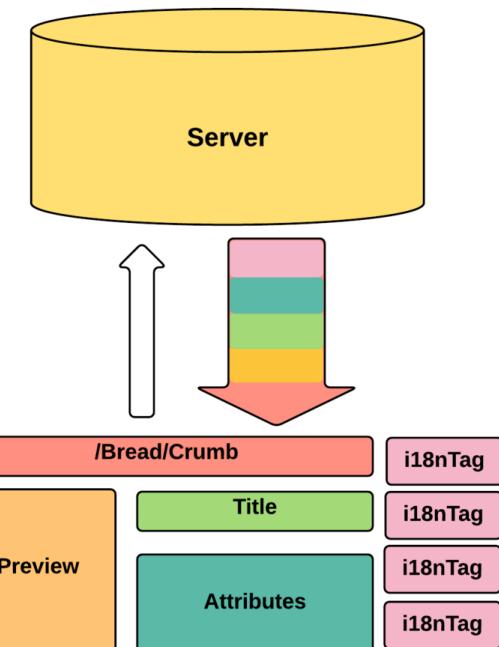
NEXT CHALLENGES

UI INTROSPECTION

nuxeo



- Introspect UI components tree
 - *gather needed data*
- Auto-configure Nuxeo Calls
 - *schemas, enrichers, resolvers ...*
- Post back aggregated data
 - *reverse enrichers and resolvers ?*



SOME LINKS



<http://nuxeo.github.io/api-playground/>

<https://university.nuxeo.io/>

<https://doc.nuxeo.com/display/NXDOC/REST+API>

<https://github.com/nuxeo>

[nuxeo-workshop-restapi](#)



ANY QUESTIONS ?

Thank You !

<http://www.nuxeo.com/careers/>

