

MODELS 2016

Saint-Malo, France, October 2-7, 2016.

ACM/IEEE 19th International Conference
on Model Driven Engineering Languages and Systems



Software and
Systems Modeling



Visual Modeling of RESTful Conversations with RESTalk

Ana Ivanchikj, Cesare Pautasso
Università della Svizzera italiana (USI)
Lugano, Switzerland

Università
della
Svizzera
italiana

Faculty
of Informatics

Silvia Schreier
innoQ Deutschland GmbH
Monheim, Germany

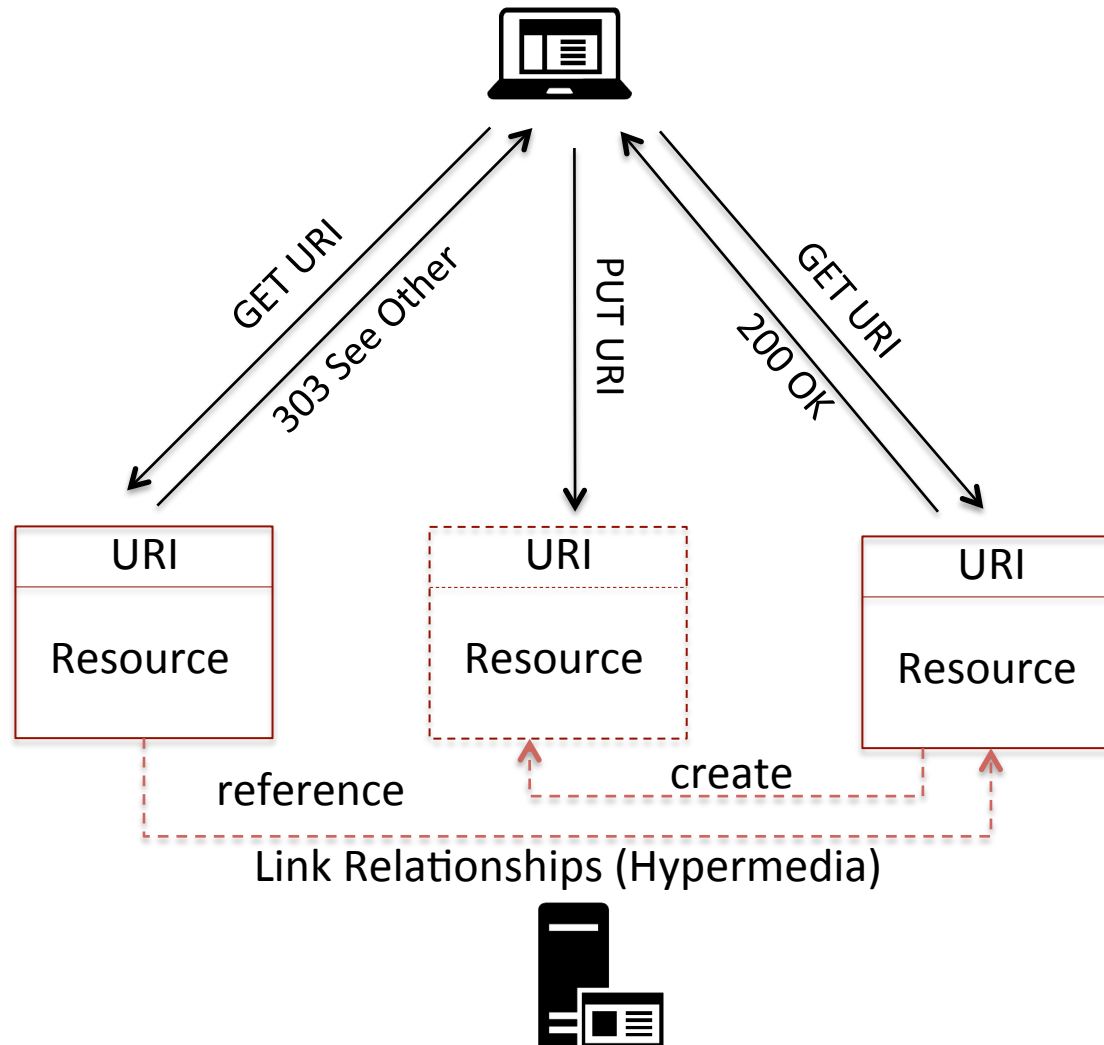
innoQ

MOTIVATION

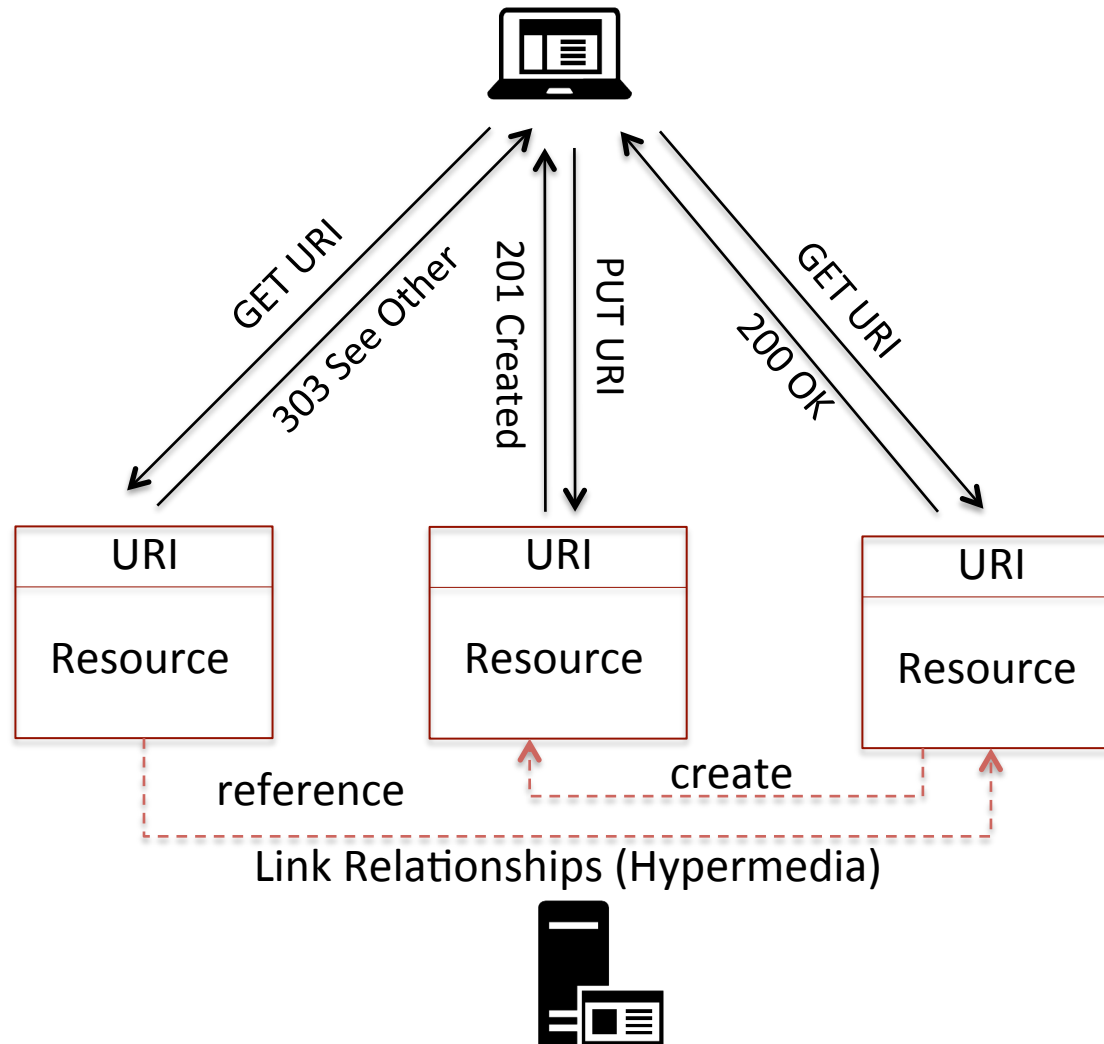
What is a RESTful conversation?

Why do we need a Domain Specific Language to model it?

RESTful conversation



RESTful conversation



REST API structure - RAML

* api.raml

```

1  #%RAML 0.8
2  ---
3  title: World Music API
4  version: v1
5  baseUri:
6  ▶ schemas:⌘
37 ▶ resourceTypes:⌘
96 ▶ traits:⌘
120 ▾ /songs:
121   type: { collection: { schema: song } }
122   is: [ secured ]
123 ▾ /{songId}:
124   type: { member: { schema: song } }
125   is: [ secured ]

```

RAMLeditor

Save

World Music API

/songs

Type: collection

GET POST

Description

No description.

Parameters

Response

200

OK

JSON SCHEMA

```

{ "$schema": "http://json-schema.org/draft-03/schema",
  "type": "object",
  "description": "The canonical song representation",
  "properties": {
    "title": { "type": "string" },
    "artist": { "type": "string" }
  }
  "required": [ "title", "artist" ]
}

```

RAML SPECIFICATION (4)

displayName

securedBy

uriParameters

baseUriParameters

RESTFUL ELEMENTS (7)

get

post

put

delete

head

patch

options

REST API structure- Swagger

POST
/pet
Add a new pet to the store

Parameters
OFF

Parameter	Value	Description	Parameter Type	Data Type
body	<pre>{ "id": 0, "category": { "id": 0, "name": "string" }, "name": "doggie", "photoUrls": [</pre> <div>Parameter content type: application/json</div>	Pet object that needs to be added to the store	body	<div>Model Model Schema</div> <pre>{ "id": 0, "category": { "id": 0, "name": "string" }, "name": "doggie", "photoUrls": ["string"], "tags": [{ </pre> <div>Click to set as parameter value</div>

Response Messages

HTTP Status Code	Reason	Response Model	Headers
405	Invalid input		

Try it out!

REST API dynamics

- Redirect -

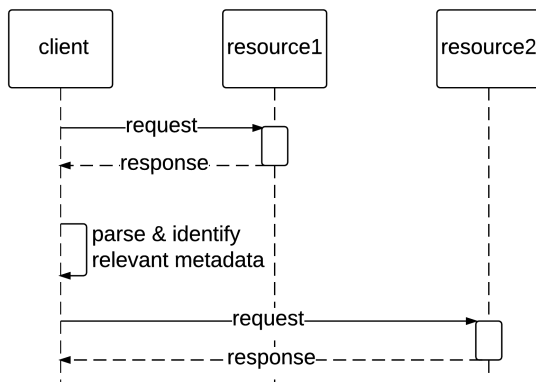
GET /resource 1 HTTP/1.1

HTTP/1.1	303 See Other
Location:	/resource2

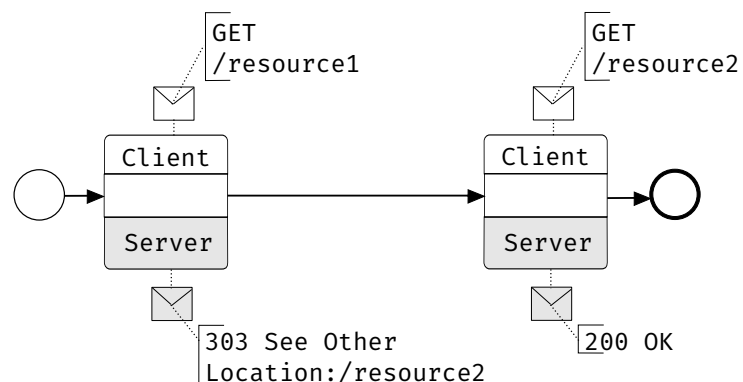
GET /resource2 HTTP/1.1

HTTP/1.1	200 OK
----------	--------

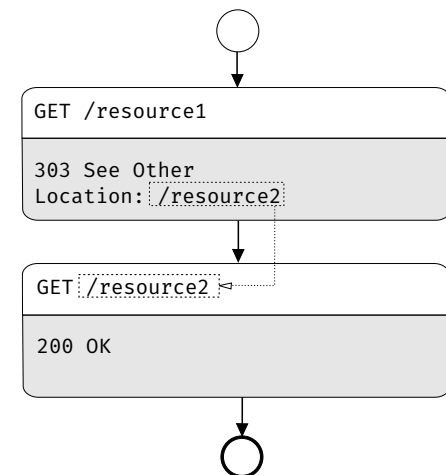
UML Sequence diagram



BPMN Choreography diagram



RESTalk

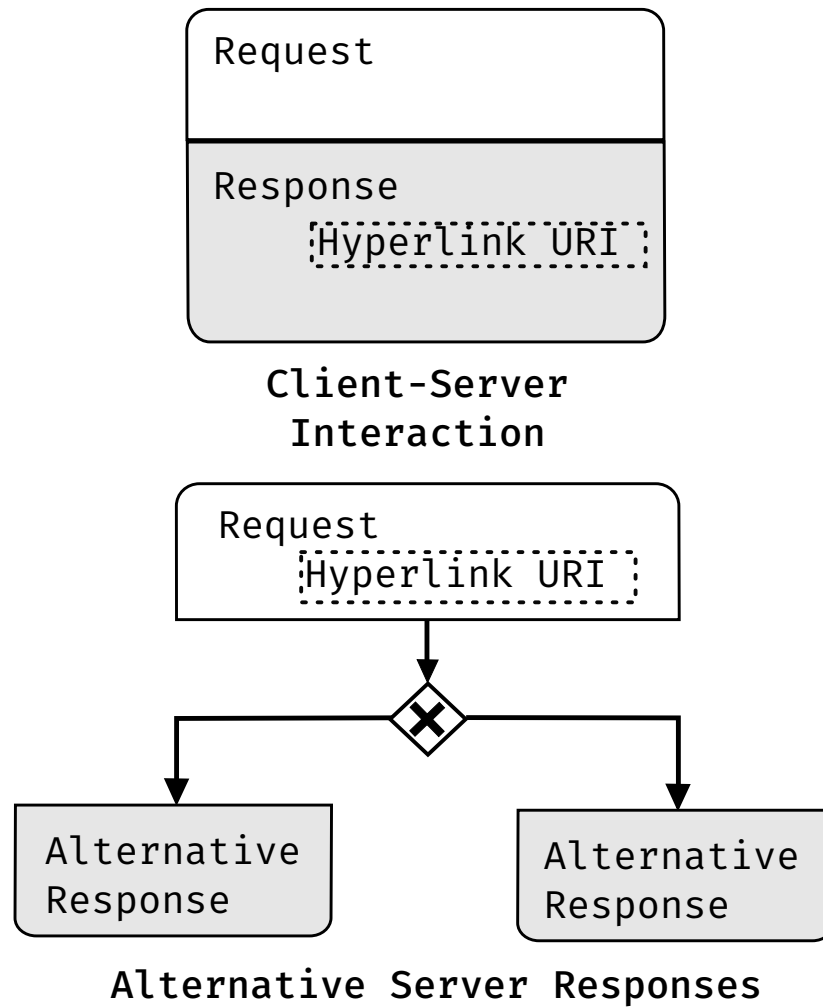


RESTalk

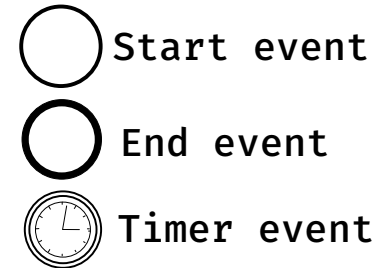
RESTalk constructs

Long-running request modeled with RESTalk

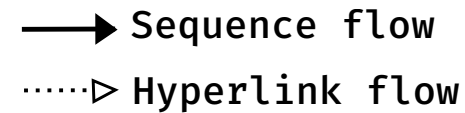
RESTalk



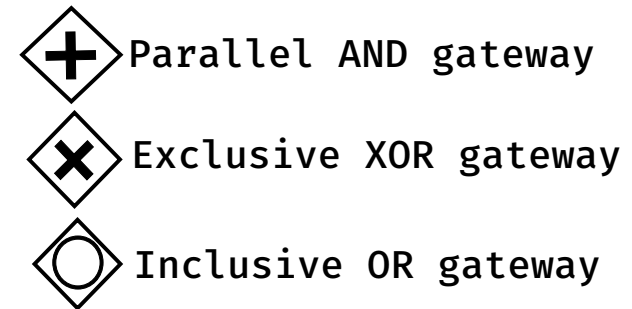
Events



Flows



Gateways



Long-running Request

-RESTful Conversation Pattern Example-
-Visualized with RESTalk-

e.g.: <http://docs.aws.amazon.com/amazonglacier/latest/dev/job-operations.html>

Long-running Request

- Happy path -

Create job

POST /job HTTP/1.1

HTTP/1.1 202 Accepted
Location: /job/42

Poll

GET /job/42 HTTP/1.1

HTTP/1.1 200 OK

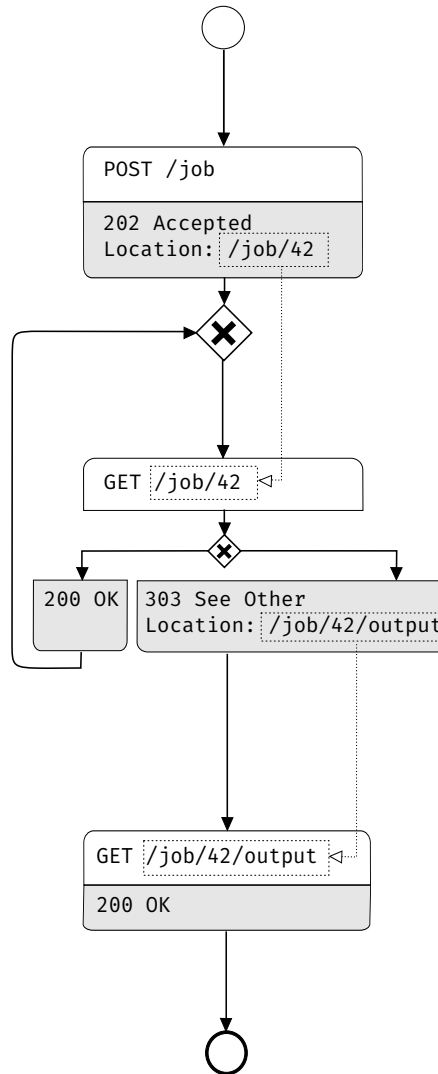
GET /job/42 HTTP/1.1

HTTP/1.1 303 See Other
Location: /job/42/output

Read results

GET /job/42/output HTTP/1.1

HTTP/1.1 200 OK



Long-running Request

- Resending the request -

Create job

POST /job HTTP/1.1

POST /job HTTP/1.1

HTTP/1.1 202 Accepted
Location: /job/42

Poll

GET /job/42 HTTP/1.1

HTTP/1.1 200 OK

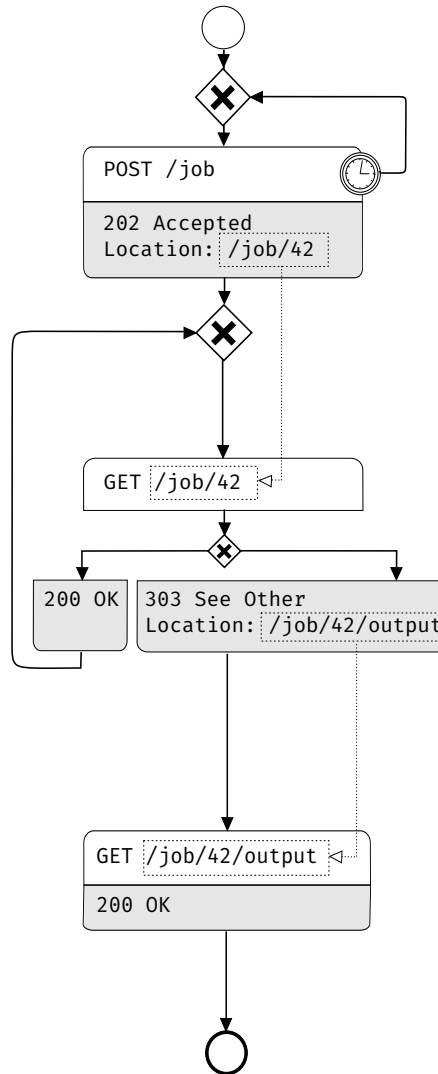
GET /job/42 HTTP/1.1

HTTP/1.1 303 See Other
Location: /job/42/output

Read results

GET /job/42/output HTTP/1.1

HTTP/1.1 200 OK



Long-running Request

- Reading the results -

Create job

POST /job HTTP/1.1

POST /job HTTP/1.1

HTTP/1.1 202 Accepted
Location: /job/42

Poll

GET /job/42 HTTP/1.1

HTTP/1.1 200 OK

GET /job/42 HTTP/1.1

HTTP/1.1 303 See Other
Location: /job/42/output

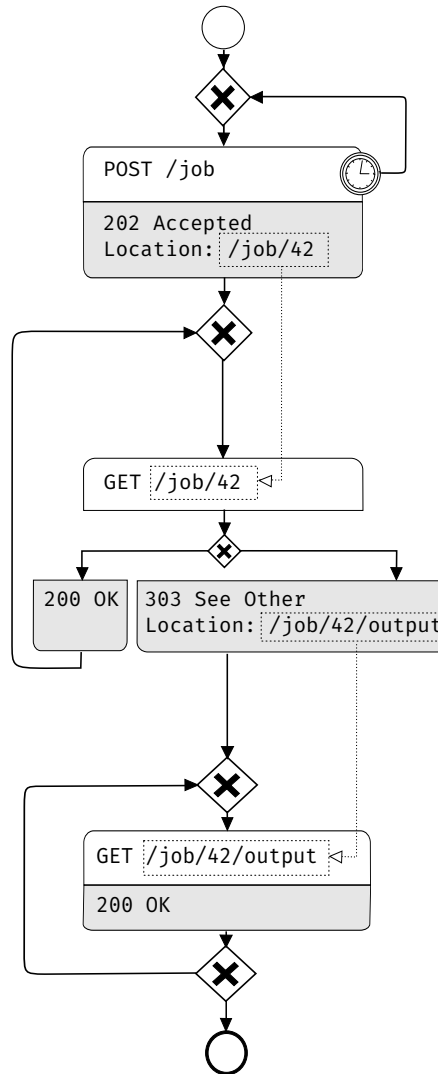
Read results

GET /job/42/output HTTP/1.1

HTTP/1.1 200 OK

GET /job/42/output HTTP/1.1

HTTP/1.1 200 OK



Long-running Request

- Deleting the output resource -

Create job

POST /job HTTP/1.1

POST /job HTTP/1.1

HTTP/1.1 202 Accepted
Location: /job/42

Poll

GET /job/42 HTTP/1.1

HTTP/1.1 200 OK

GET /job/42 HTTP/1.1

HTTP/1.1 303 See Other
Location: /job/42/output

Read results

GET /job/42/output HTTP/1.1

HTTP/1.1 200 OK

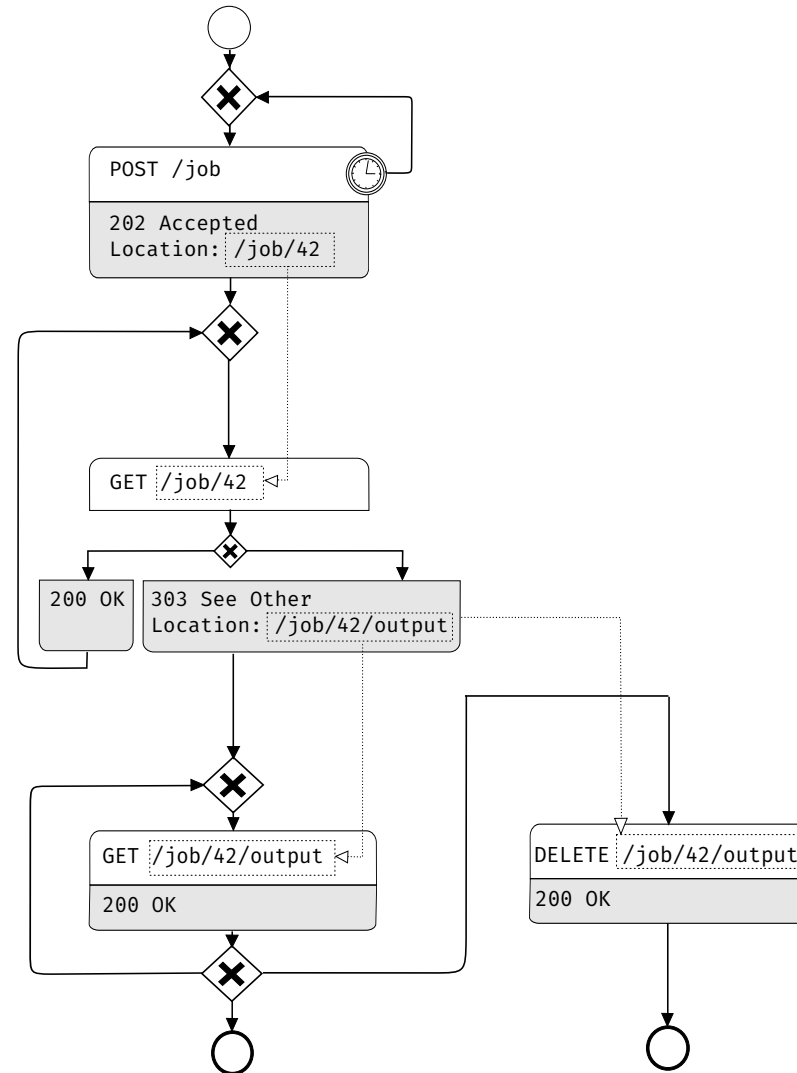
GET /job/42/output HTTP/1.1

HTTP/1.1 200 OK

Delete output

DELETE /job/42/output HTTP/1.1

HTTP/1.1 200 OK



Long-running Request

- Deleting the output resource -

Create job

POST /job HTTP/1.1

POST /job HTTP/1.1

HTTP/1.1 202 Accepted
Location: /job/42

Poll

GET /job/42 HTTP/1.1

HTTP/1.1 200 OK

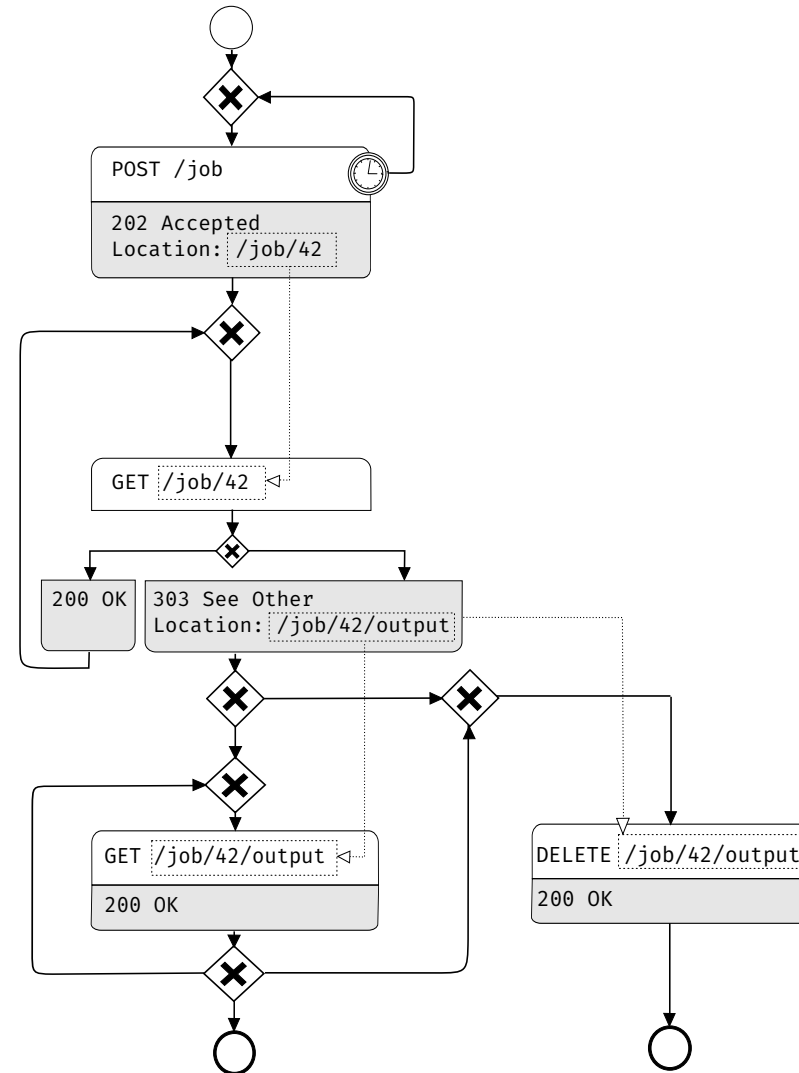
GET /job/42 HTTP/1.1

HTTP/1.1 303 See Other
Location: /job/42/output

Delete output

DELETE /job/42/output HTTP/1.1

HTTP/1.1 200 OK



Long-running Request

- Deleting the job resource -

Create job

POST /job HTTP/1.1

POST /job HTTP/1.1

HTTP/1.1 202 Accepted
Location: /job/42

Poll

GET /job/42 HTTP/1.1

HTTP/1.1 200 OK

GET /job/42 HTTP/1.1

HTTP/1.1 303 See Other
Location: /job/42/output

Delete output

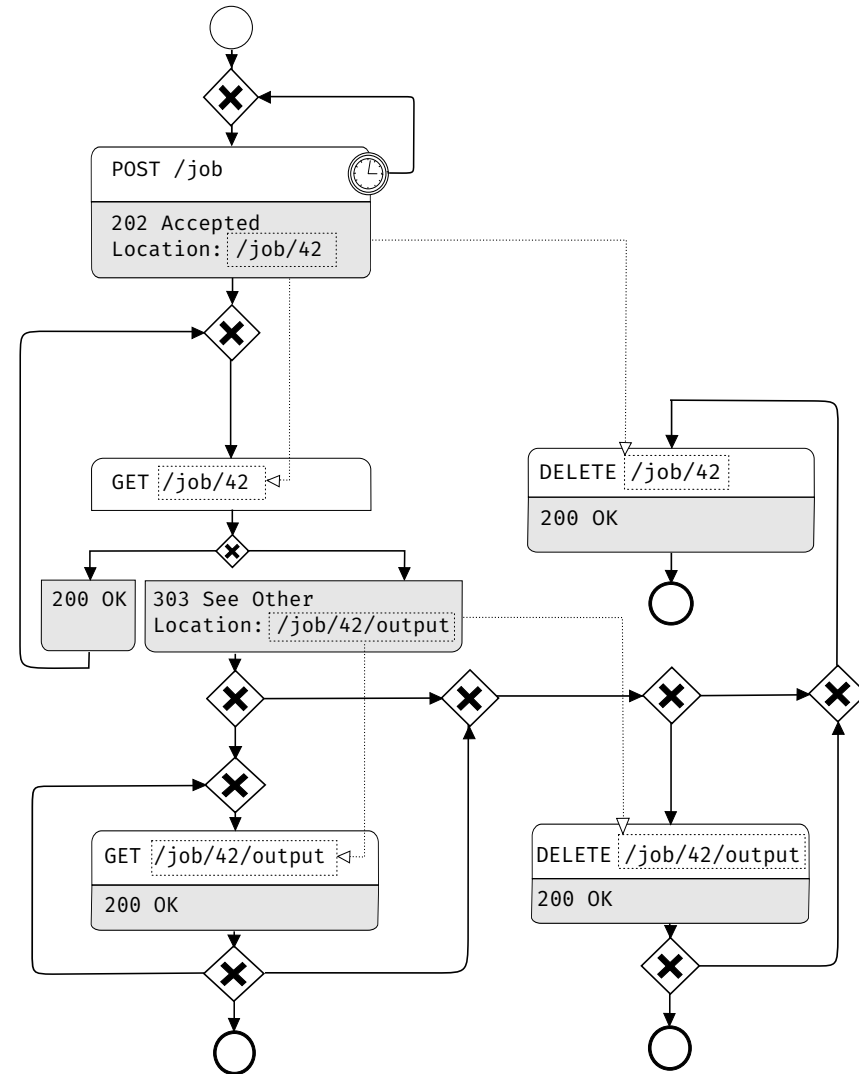
DELETE /job/42/output HTTP/1.1

HTTP/1.1 200 OK

Clean up

DELETE /job/42 HTTP/1.1

HTTP/1.1 200 OK



Long-running Request

- Deleting the job resource -

Create job

POST /job HTTP/1.1

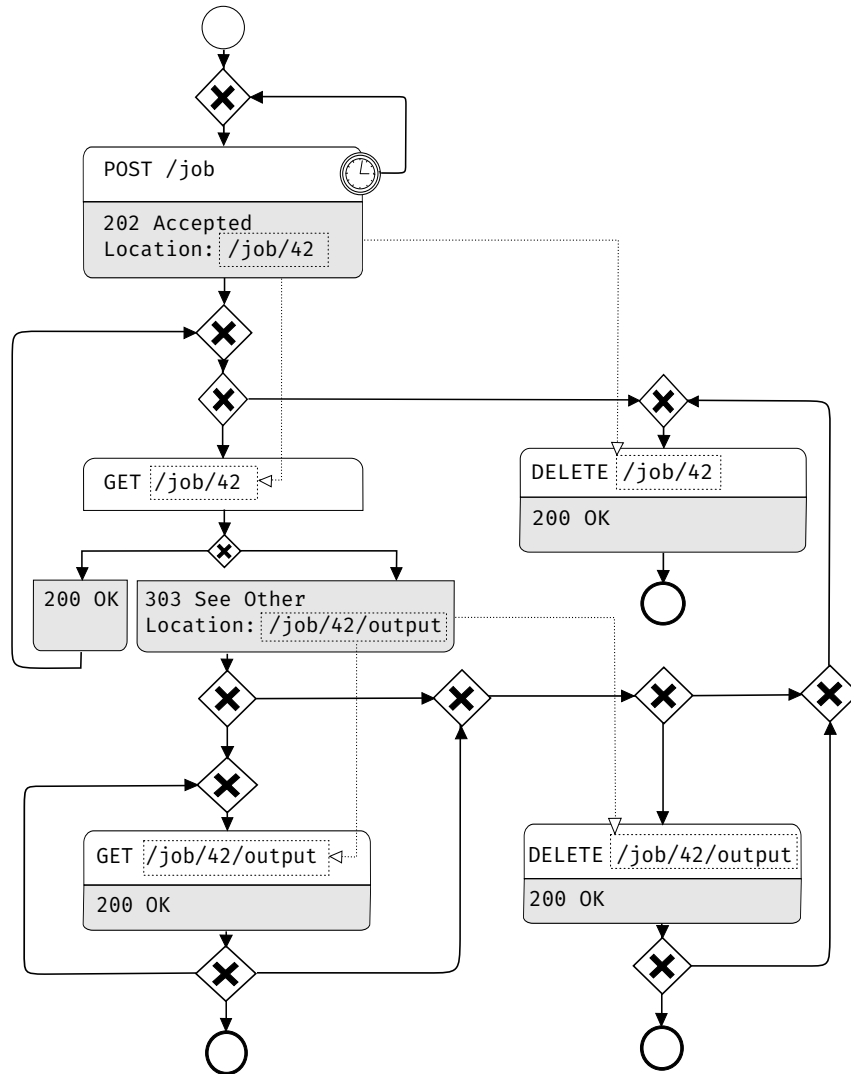
POST /job HTTP/1.1

HTTP/1.1 202 Accepted
Location: /job/42

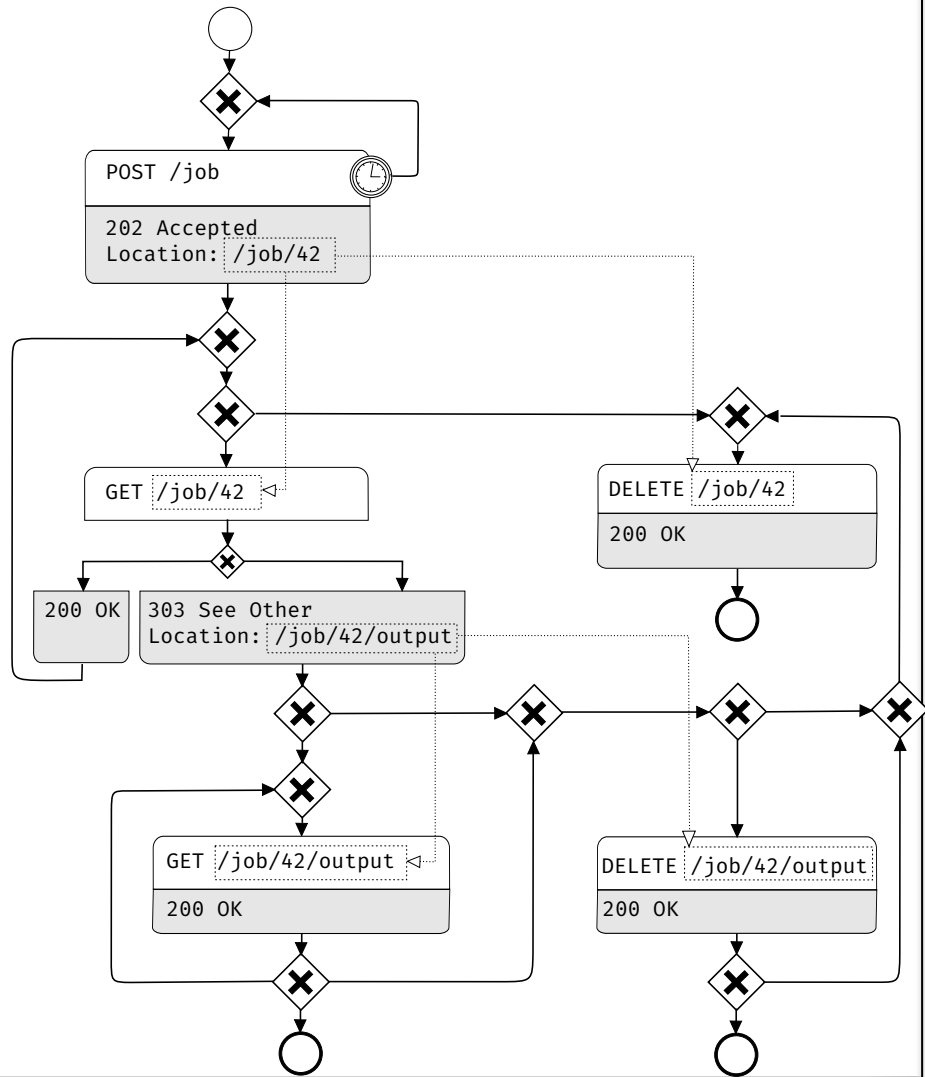
Delete job

DELETE /job/42 HTTP/1.1

HTTP/1.1 200 OK



Long-running Request



Long-running Request

- Short happy path -

Create job

POST /job HTTP/1.1

HTTP/1.1 202 Accepted
Location: /job/42

Poll

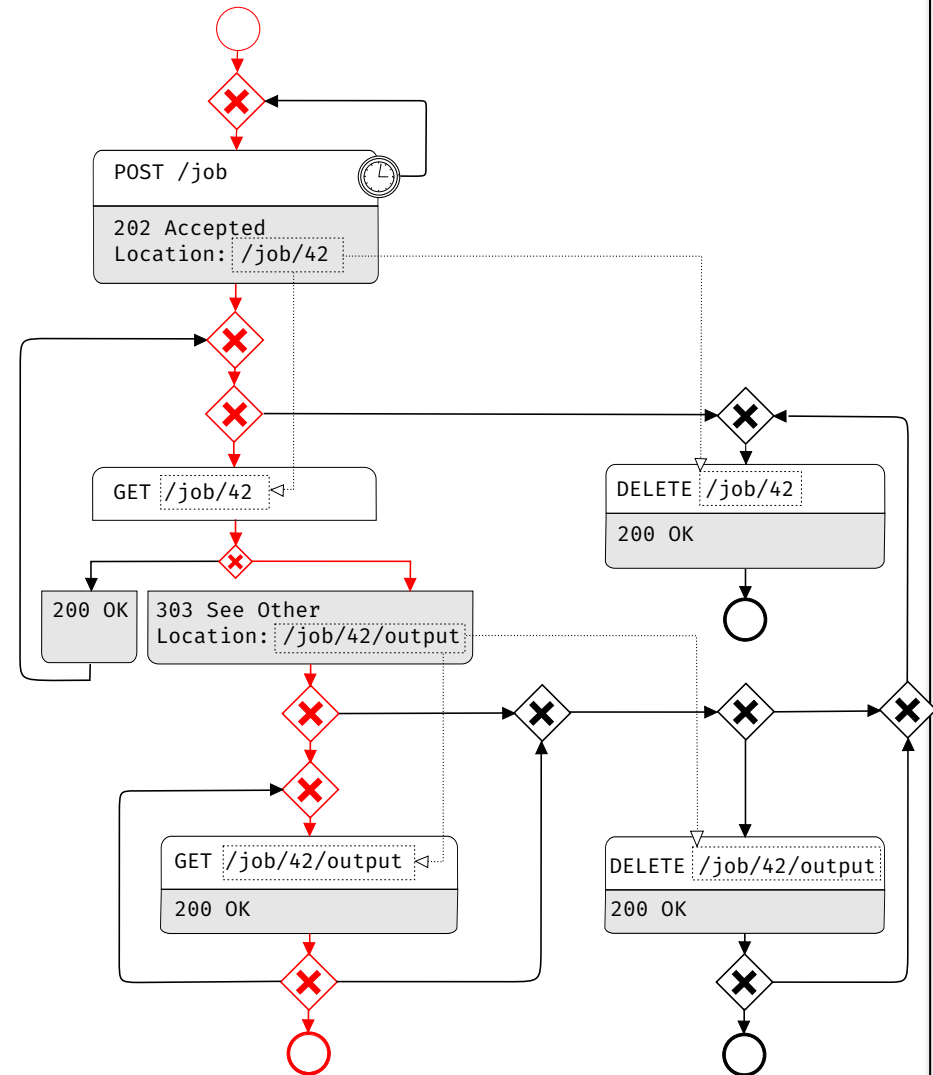
GET /job/42 HTTP/1.1

HTTP/1.1 303 See Other
Location: /job/42/output

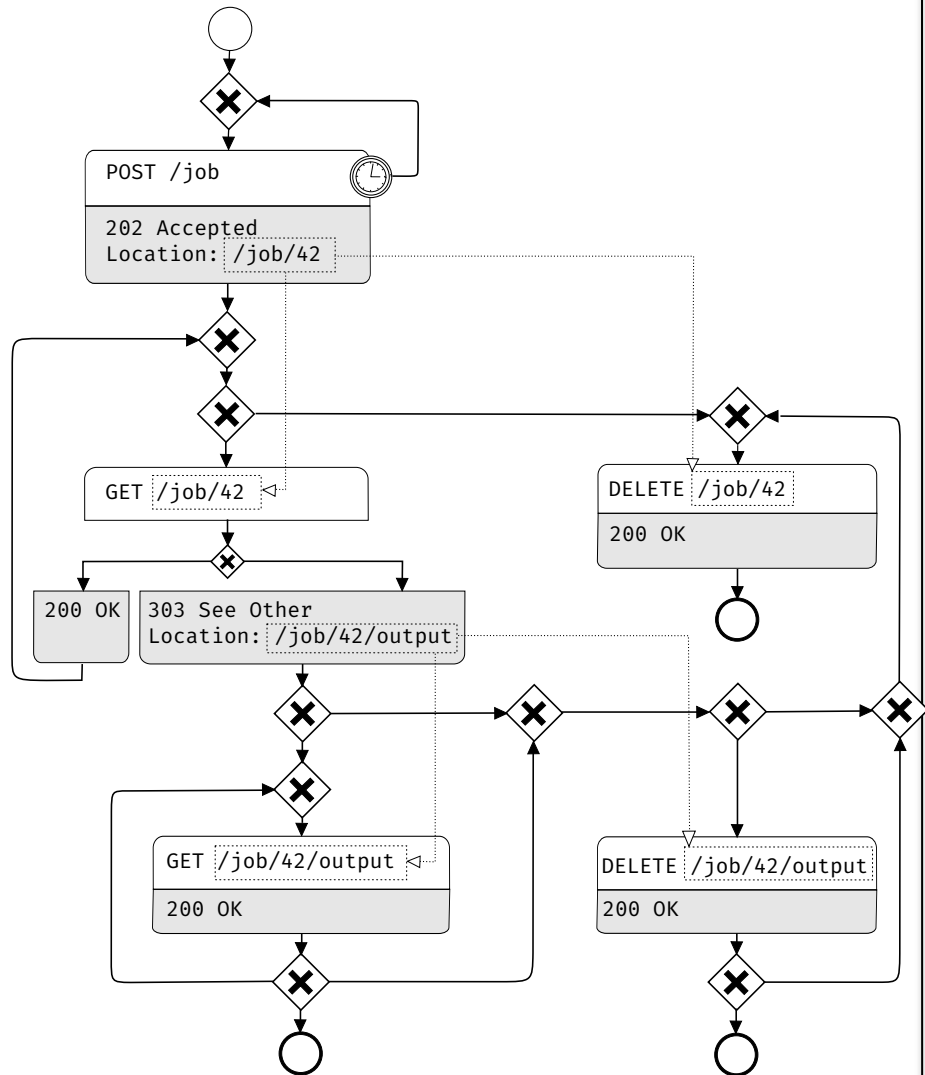
Read results

GET /job/42/output HTTP/1.1

HTTP/1.1 200 OK



Long-running Request



Long-running Request

- Long path -

Create job

POST /job HTTP/1.1

POST /job HTTP/1.1

HTTP/1.1 202 Accepted
Location: /job/42

Poll

GET /job/42 HTTP/1.1

HTTP/1.1 200 OK

GET /job/42 HTTP/1.1

HTTP/1.1 200 OK

GET /job/42 HTTP/1.1

HTTP/1.1 303 See Other
Location: /job/42/output

Read results

GET /job/42/output HTTP/1.1

HTTP/1.1 200 OK

GET /job/42/output HTTP/1.1

HTTP/1.1 200 OK

Delete output

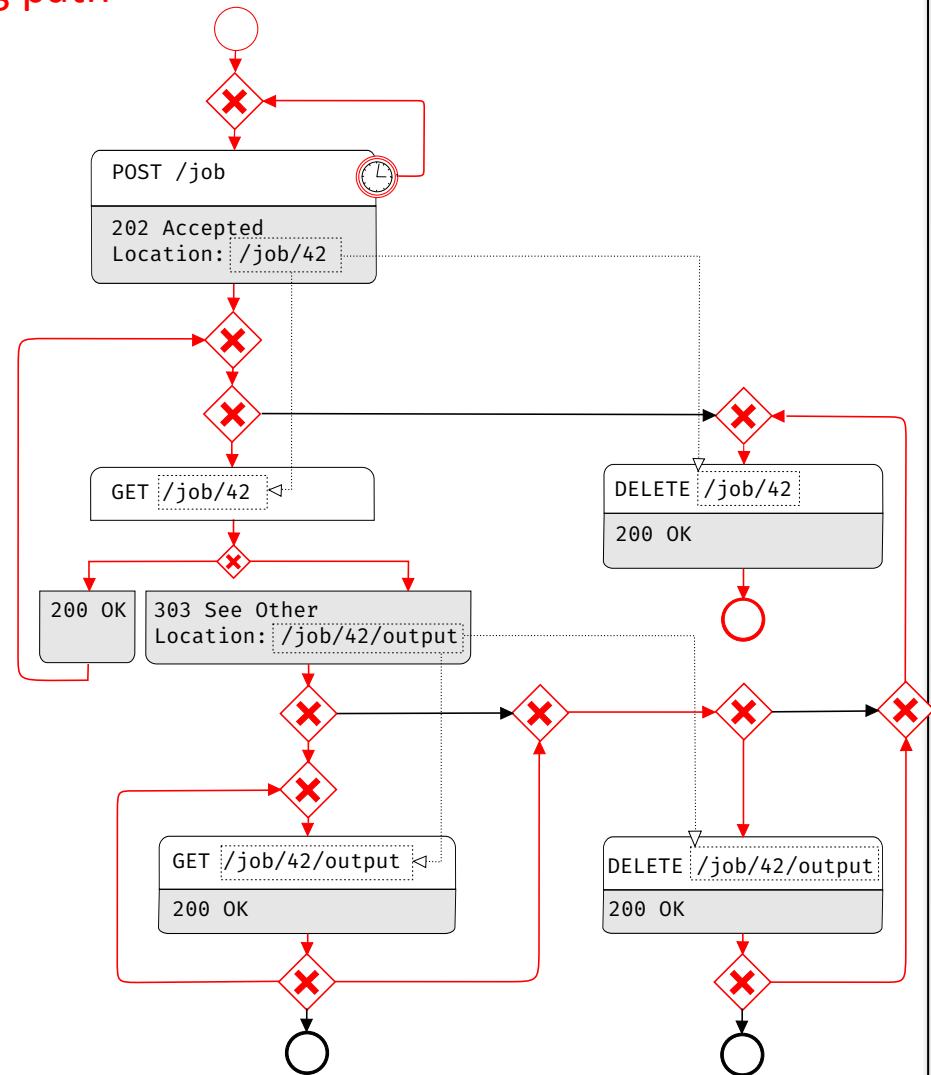
DELETE /job/42/output HTTP/1.1

HTTP/1.1 200 OK

Delete job

DELETE /job/42 HTTP/1.1

HTTP/1.1 200 OK



RESTalk Exploratory Survey

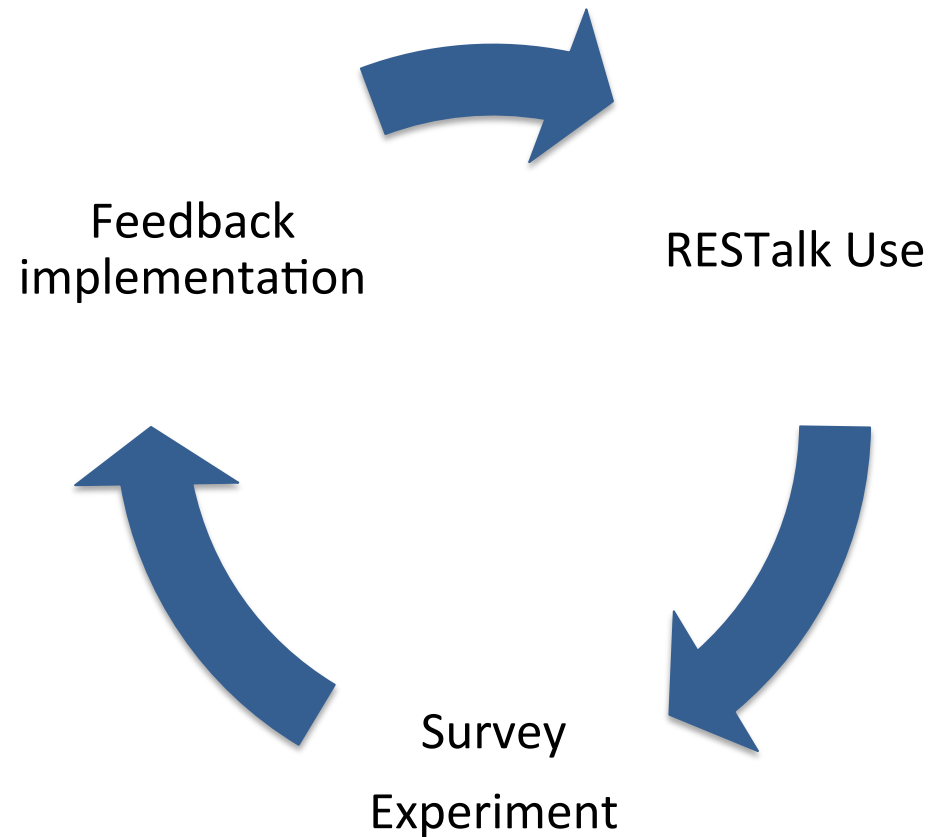
Motivation

Goals

Design

Results

Approach: Agile DSML Design



Final Goal:

Increase acceptance and dissemination of RESTalk

Explorative Survey Goals

- **Goal 1:** Evaluate the need in industry for a domain specific language for modeling RESTful conversations
- **Goal 2:** Evaluate the cognitive characteristics of RESTalk



***Qualitative research technique => No statistical inference**

Explorative Survey Design

English + German



Modeling RESTful Conversations (English version)

Welcome to the RESTful conversations survey. RESTful conversations are complex interactions between client(s) and server(s). For more details on the conversation based approach for modeling RESTful APIs please refer to the following paper: <http://design.inf.usi.ch/sites/default/files/biblio/wicsa2015.pdf>.

**Please note that having in mind the exploratory goal of this survey, going back to the previous question is not an option in the same.*

0% 100%

Usage of BPMN Choreography

Why did you decide to use BPMN Choreography?

Which constructs of BPMN Choreography do you appreciate the most and you find core for depicting RESTful conversations?



Mainly open ended optional questions

7 question groups:

- demographic data
- background on used notations in practice
- RESTalk's intuitiveness
- RESTalk vs. standard BPMN Choreography
- reading task
- modeling task
- RESTalk's evaluation

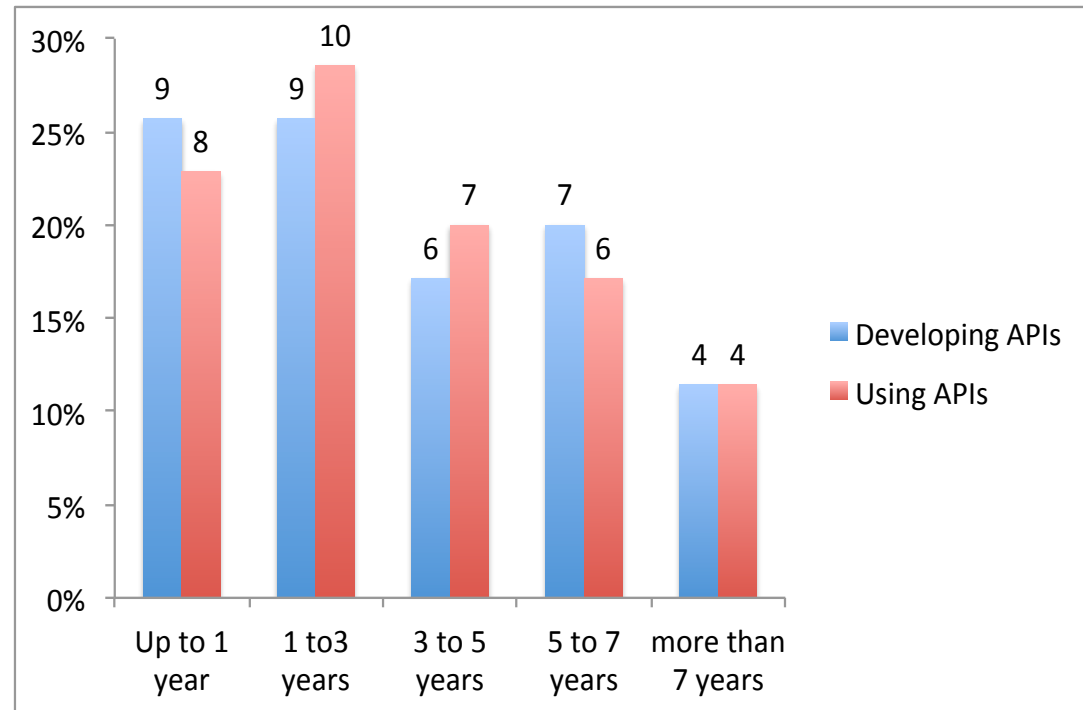
Respondents' Demographic Data

35 respondents:

- 74% industry
- 26% academia

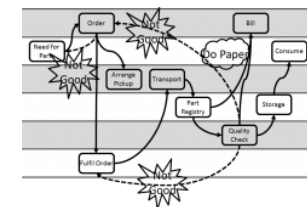
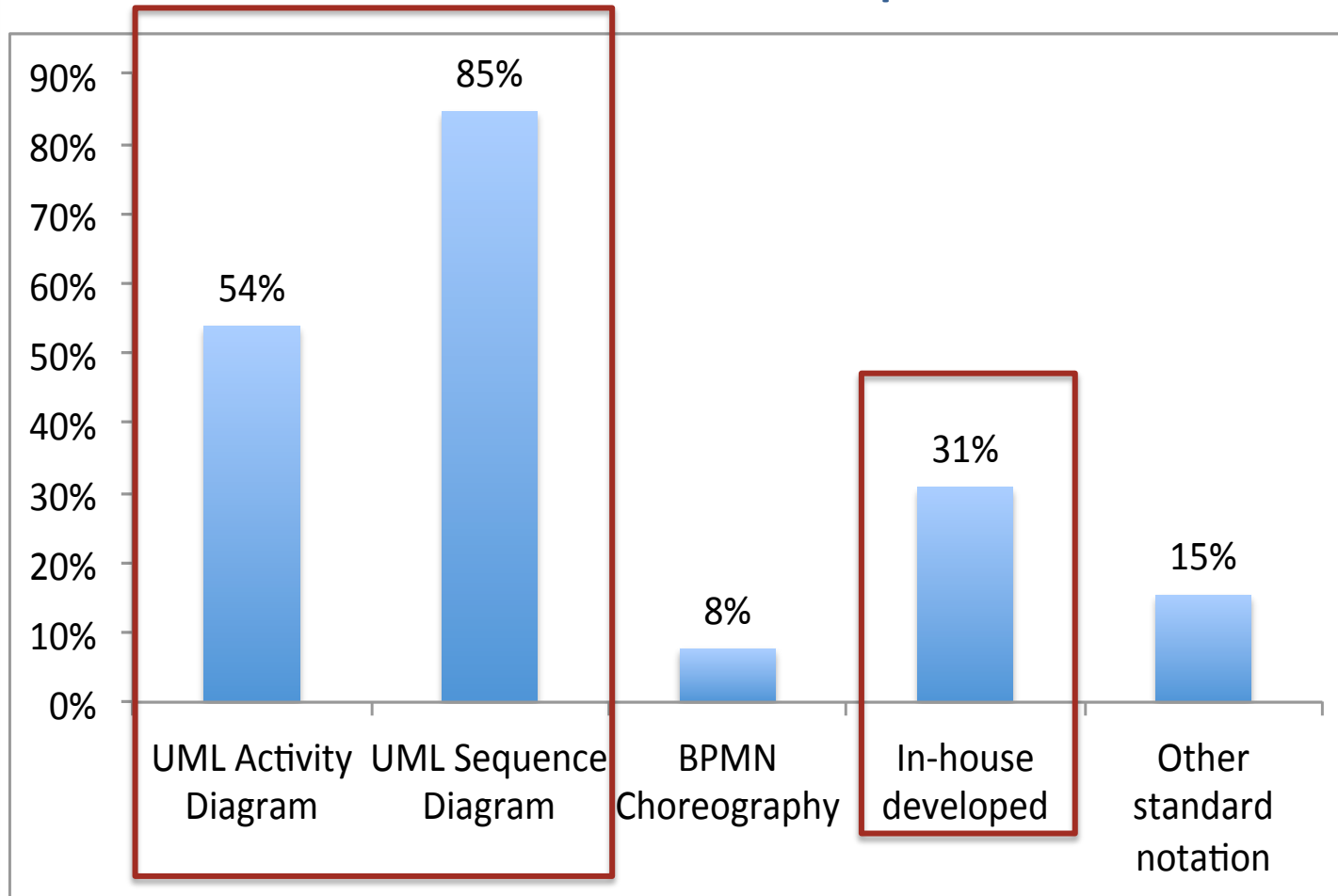
Profile:

- IT consultants
- SW quality engineers
- SW developers
- SW architects
- a CTO
- researchers

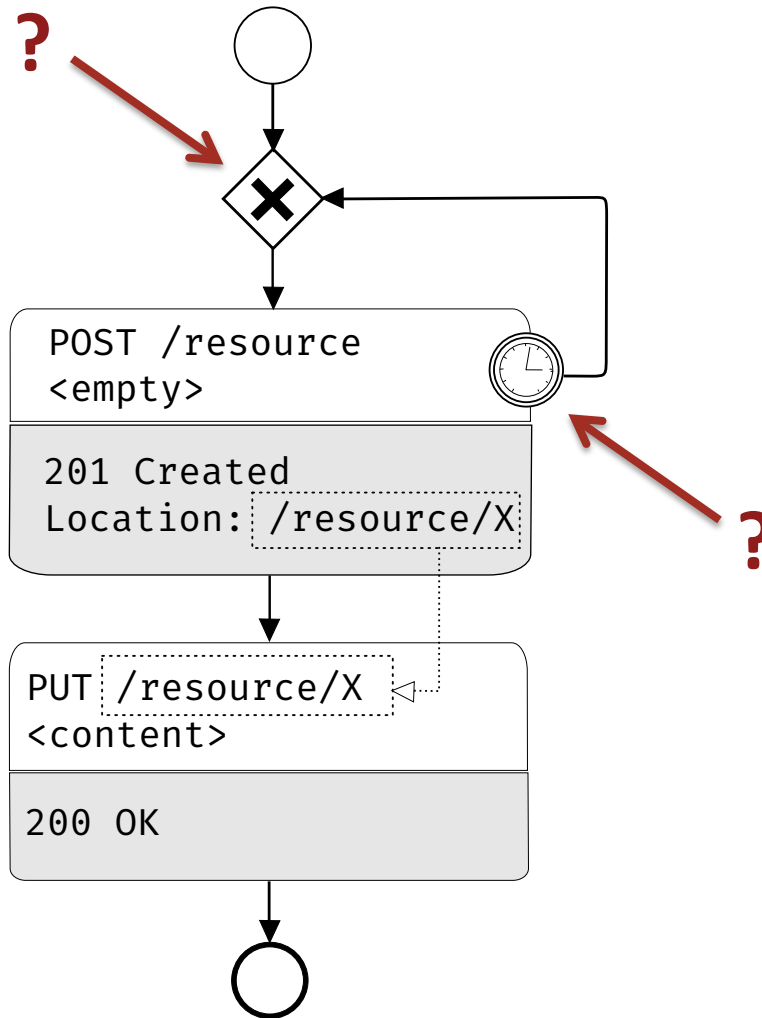


Used notations in practice

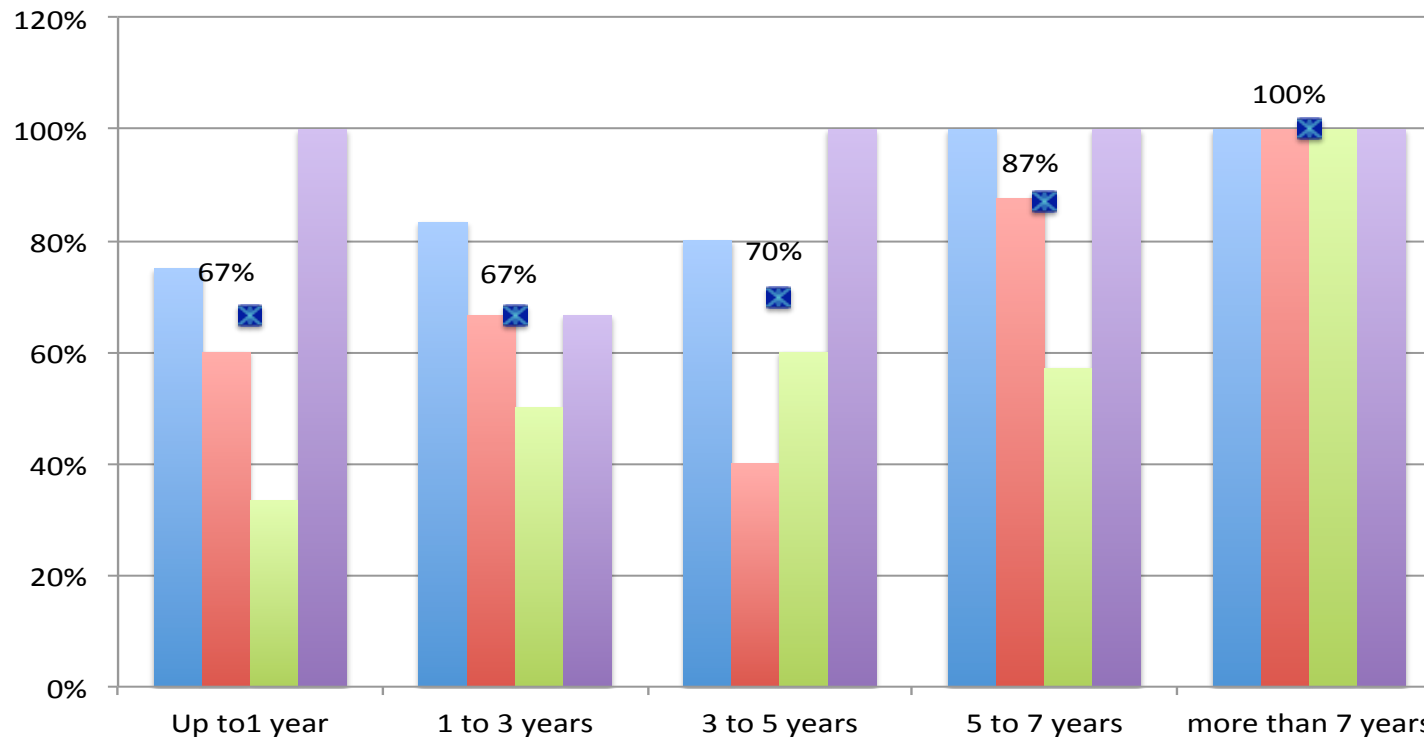
-38% of respondents-



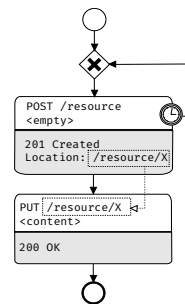
RESTalk's intuitiveness



RESTalk's intuitiveness



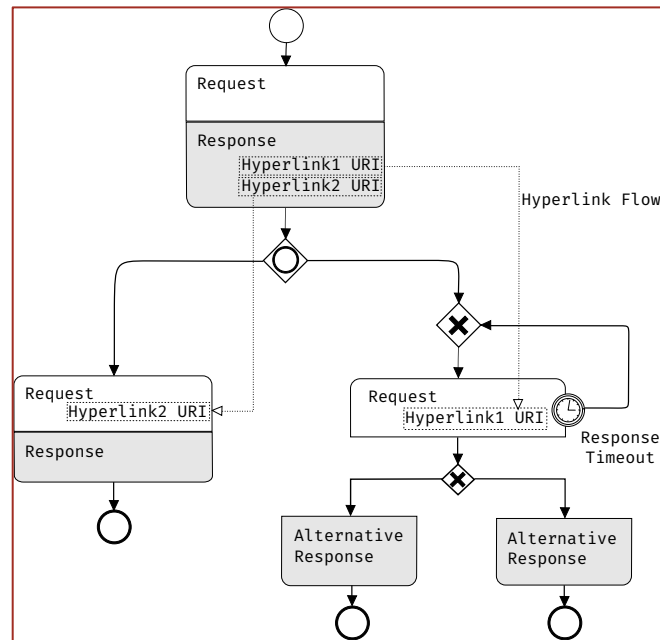
- The goal of this RESTful conversation is creating a new resource
- The client can send the POST request multiple times



- By sending multiple POST requests multiple resources are being created
- The client knows the link to the created resource before the start of the conversation
- ✕ Average correct answers

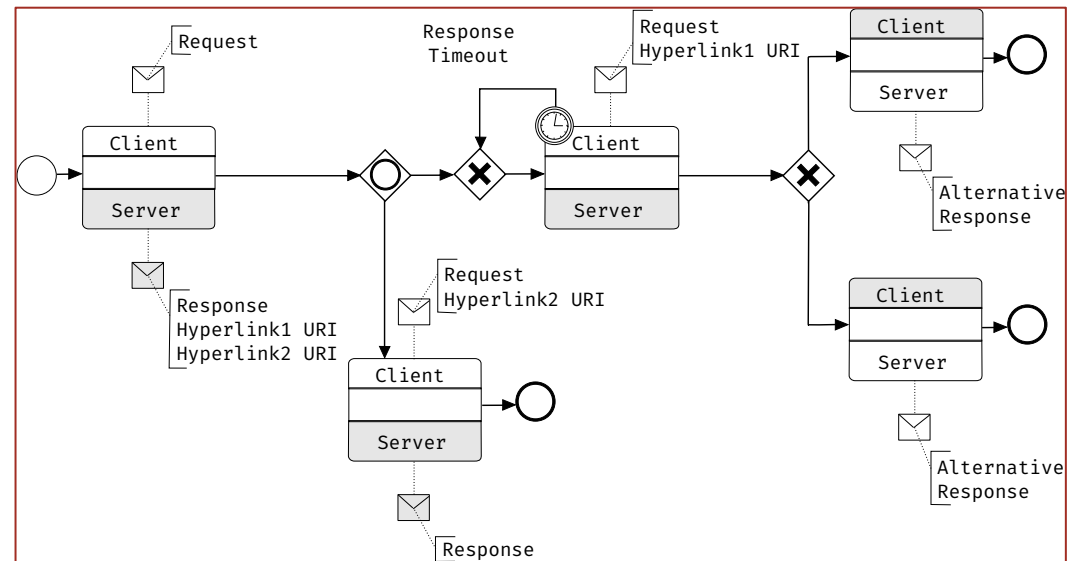
RESTalk vs. Standard BPMN Choreography

-41% of respondents-



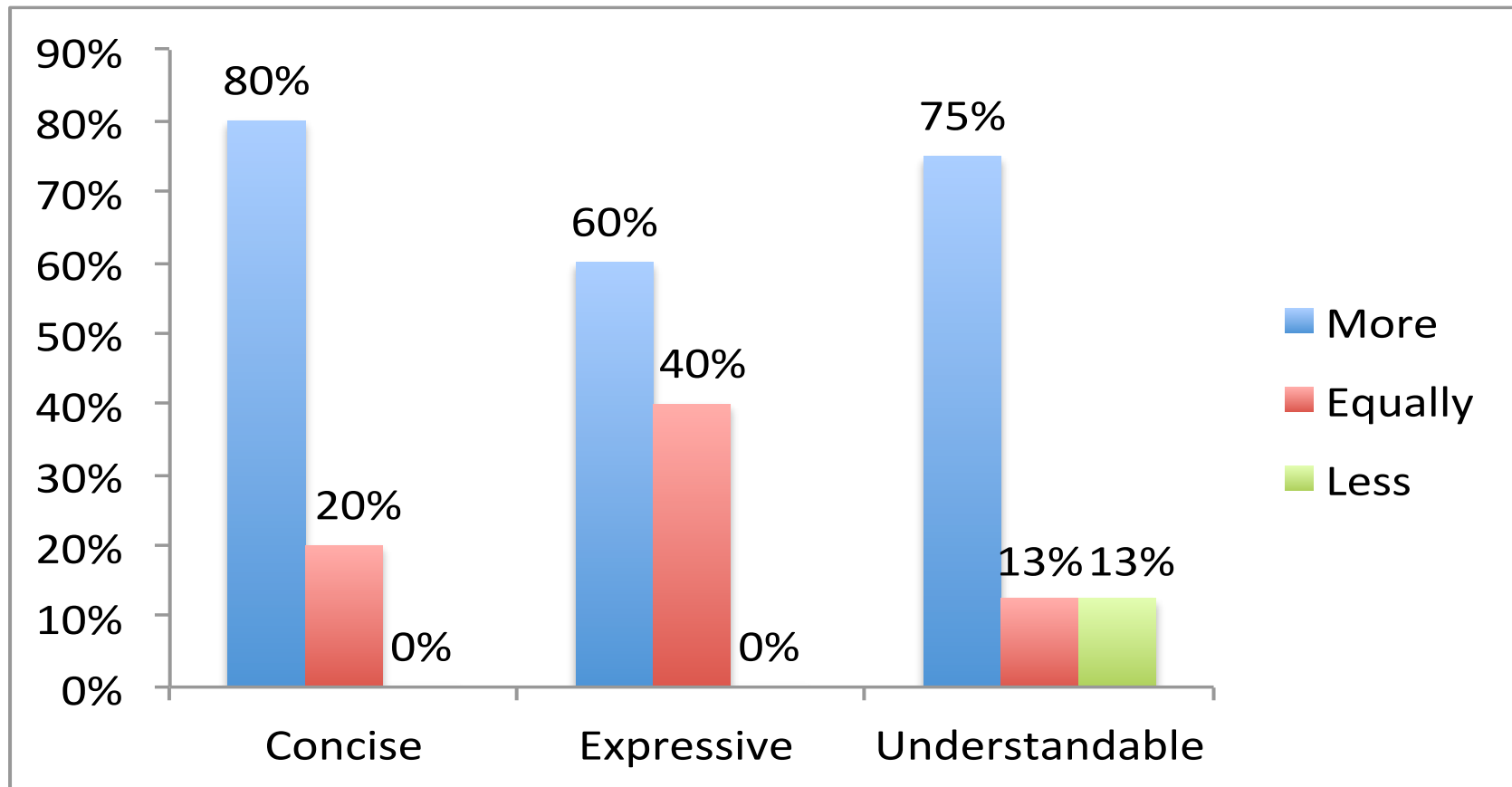
RESTalk

Standard BPMN Choreography



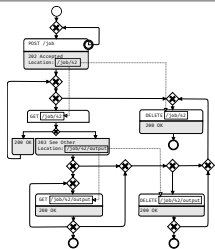
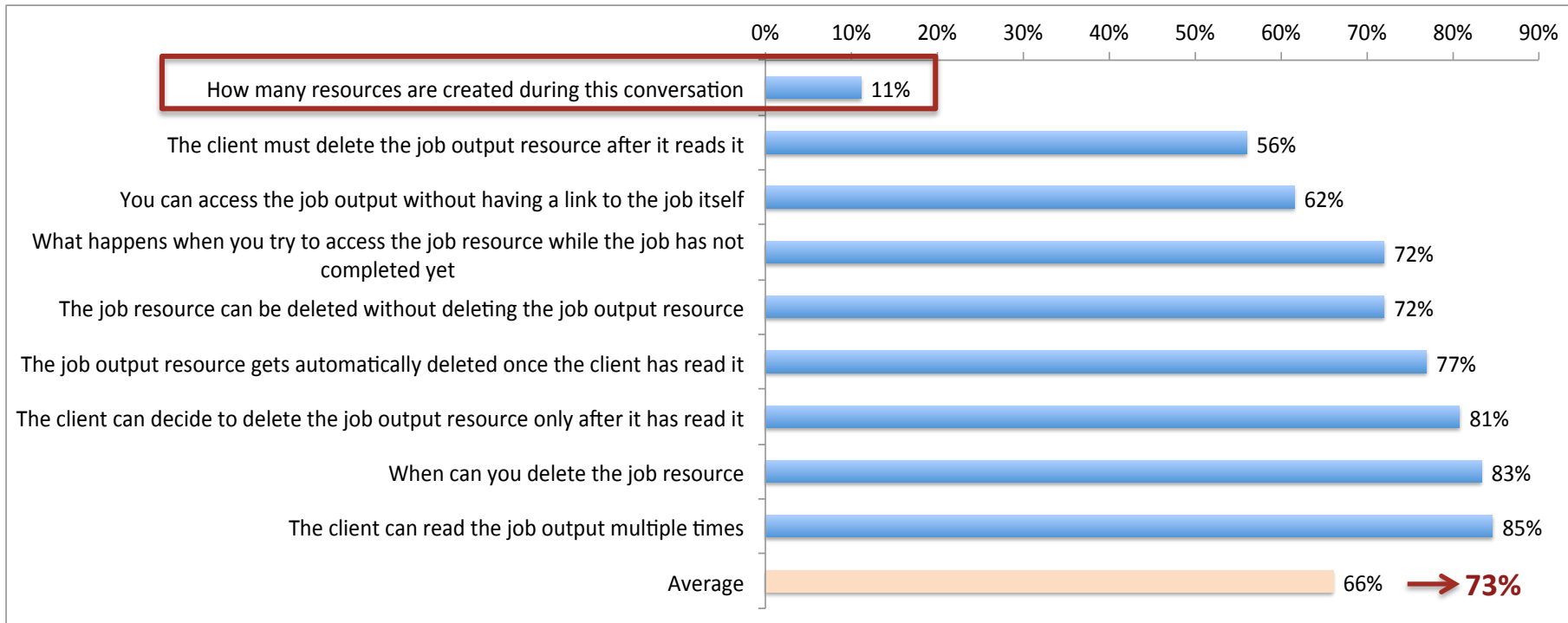
RESTalk vs. Standard BPMN Choreography

-41% of respondents-



Reading task

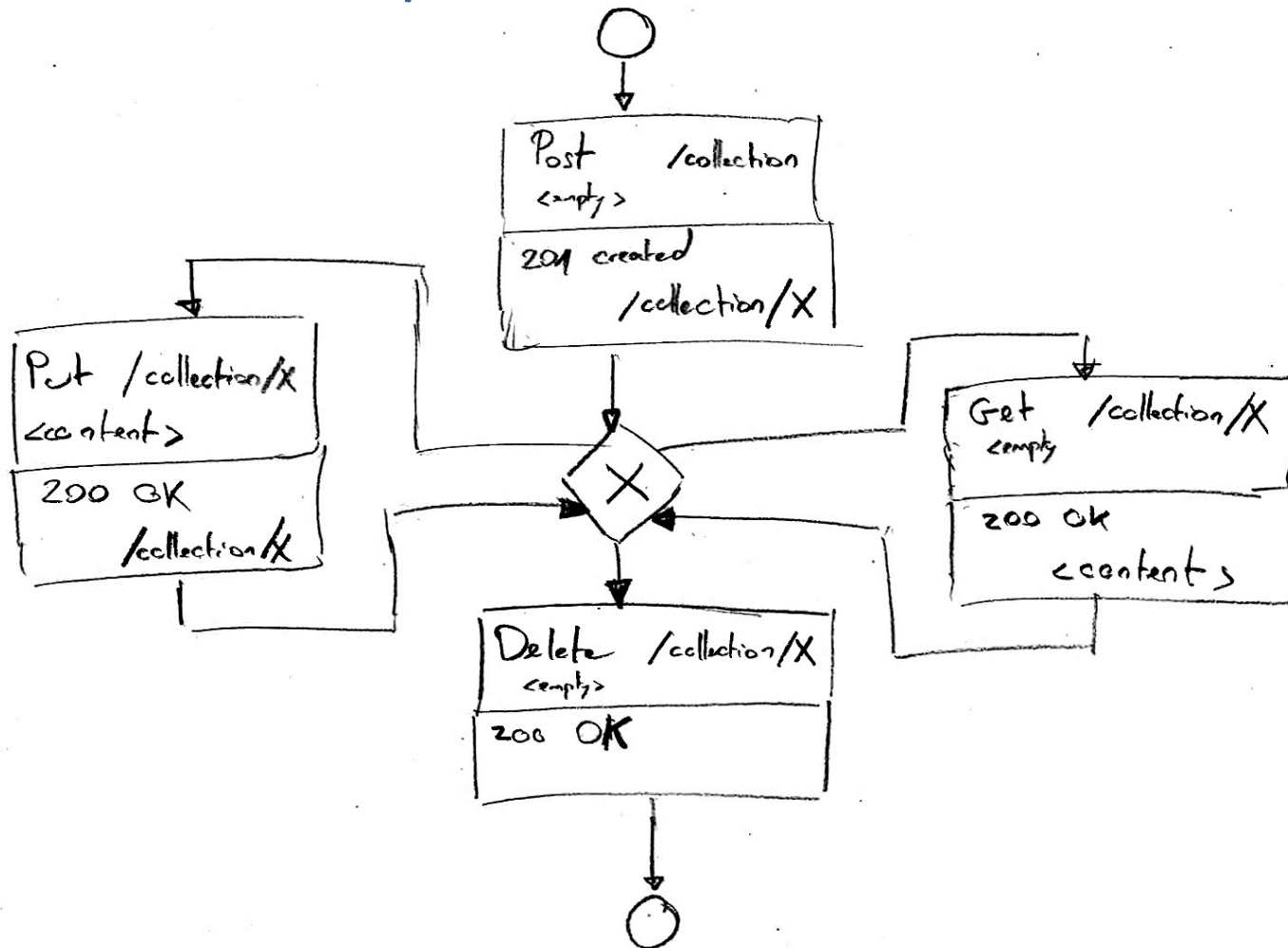
-Long Running Request-



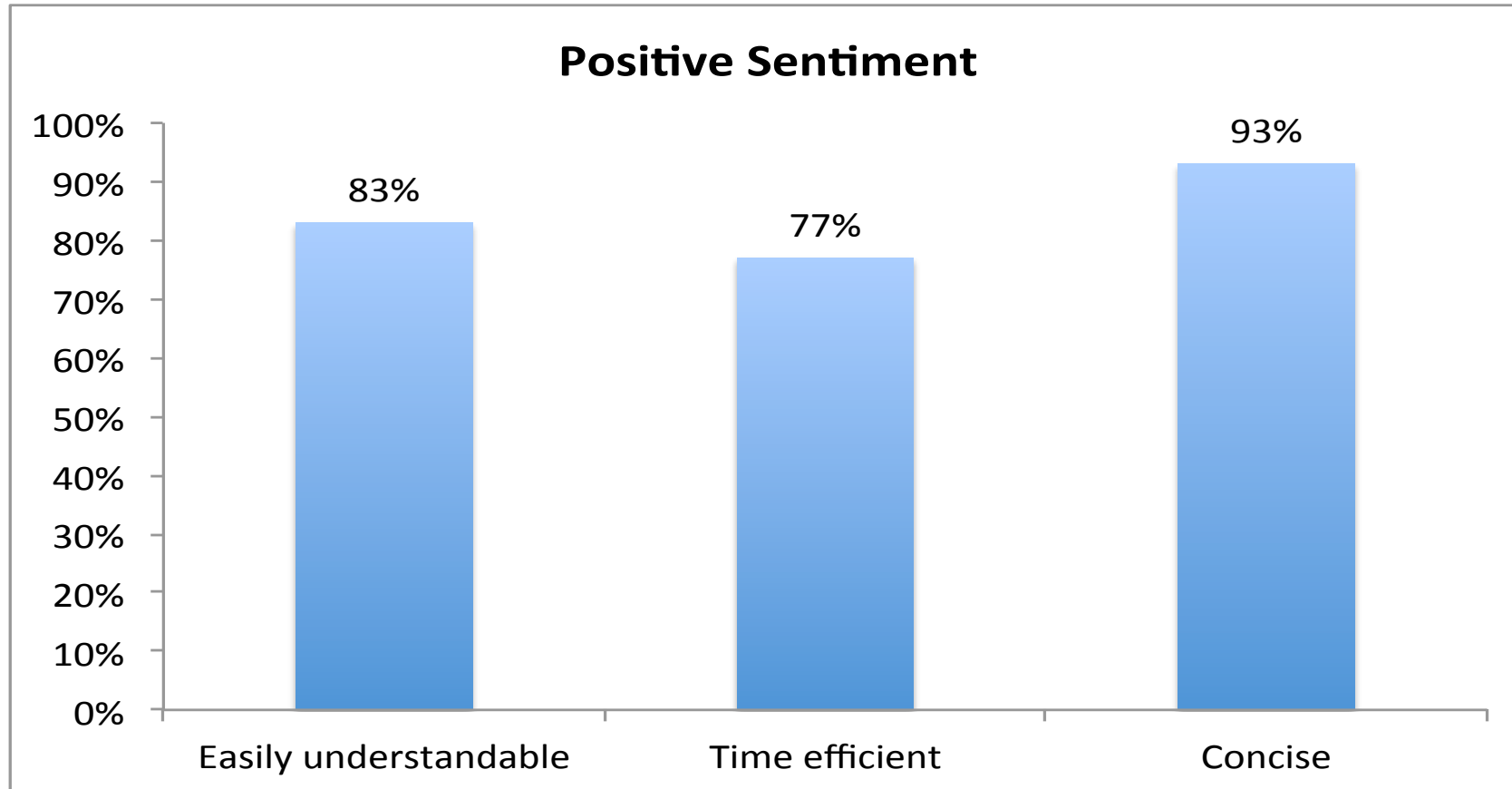
Correct answers by sector:
86% in academia and 68% in industry

Modeling task

-CRUD operations on a resource-



RESTalk's evaluation



Would use RESTalk – 78%
Would prefer a tool – 69%

Conclusions

Take-Aways

Further existing work

Future work

Explorative Survey Goals

- **Goal 1:** Evaluate the need in industry for a domain specific language for modeling RESTful conversations
- **Goal 2:** Evaluate the cognitive characteristics of RESTalk



***Qualitative research technique => No statistical inference**

Take away

-Goal 1: Evaluate the need in industry for a DSML-



38% already using some notation
78% willing to use RESTalk



Take away

-Goal 2: Evaluate the cognitive characteristics of RESTalk-

Intuitiveness:

77% correct answers without prior RESTalk knowledge

Feedback: include the rationale behind server's decisions at XOR gateways



Closeness of mapping to the problem world:

61% found RESTalk more concise than what they are using

Feedback: make state transitions and looping limits explicit in RESTalk

Abstraction gradient:

83% found RESTalk easy or somewhat easy to understand

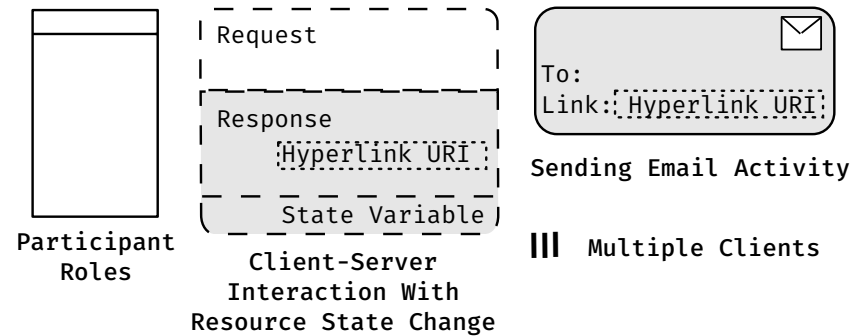
Feedback: use end events only after a DELETE method and make dependencies among resources explicit



Further Existing Work

RESTalk Extension

- roles
- state transitions
- asynch email interactions
- multiparty

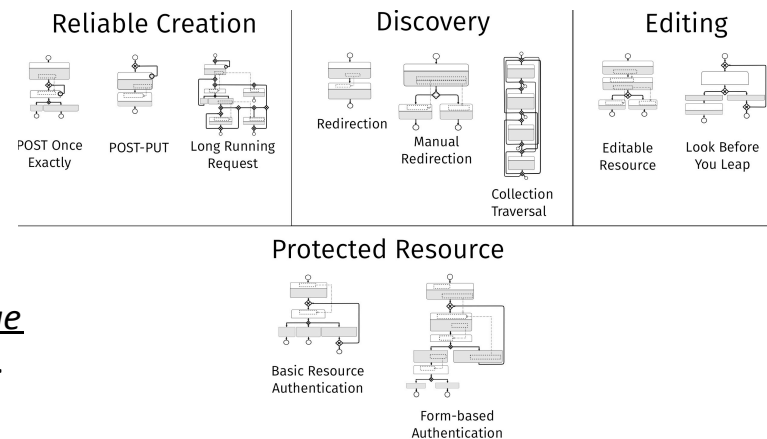


A. Ivanchikj. *RESTful Conversation with RESTalk –The Use Case of Doodle-*. In *Proc. of ICWE*. Springer, 2016

RESTalk Pattern Language

<http://restalk-patterns.org>

C. Pautasso, A. Ivanchikj, and S. Schreier. *A Pattern Language for RESTful Conversations*. In *Proc. Of EuroPLoP*. ACM, 2016.



Future Work

Tool development:

- natural language
- graphical modeling
- code generation

Feedback
implementation

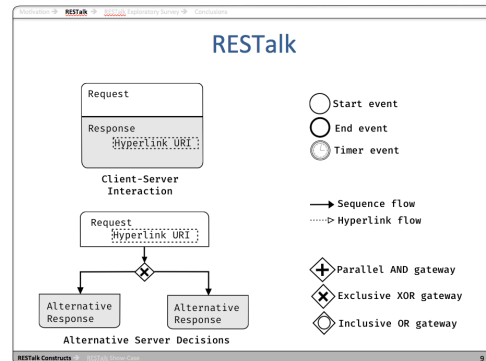
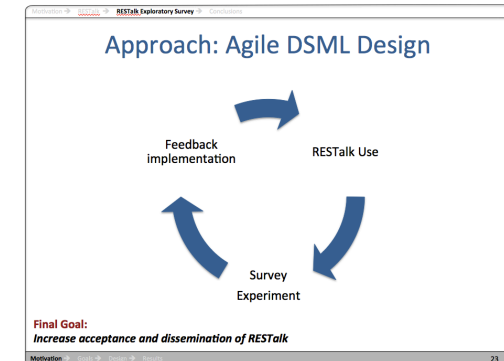
RESTalk Use

Applied:

- on patterns
- on real APIs
- by practitioners

Survey
Experiment

With statistical relevance

Contribute to our research on RESTful conversation patterns!
<http://restalk-patterns.org/contribute.html>

Contact us at:
 ana.ivanchikj@usi.ch

c.pautasso@ieee.org
 silvia.schreier@innoq.com