



PHYSICS

INTRO AND CHALLENGES

2nd PHYSICS Hackathon, 24/5/2023-7/6/2023



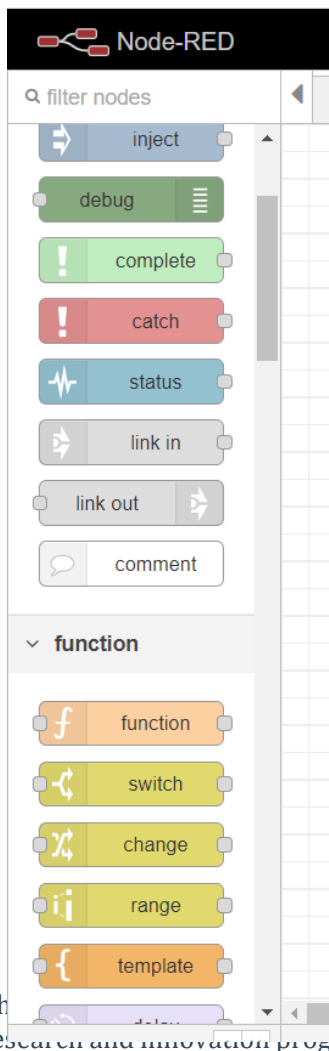
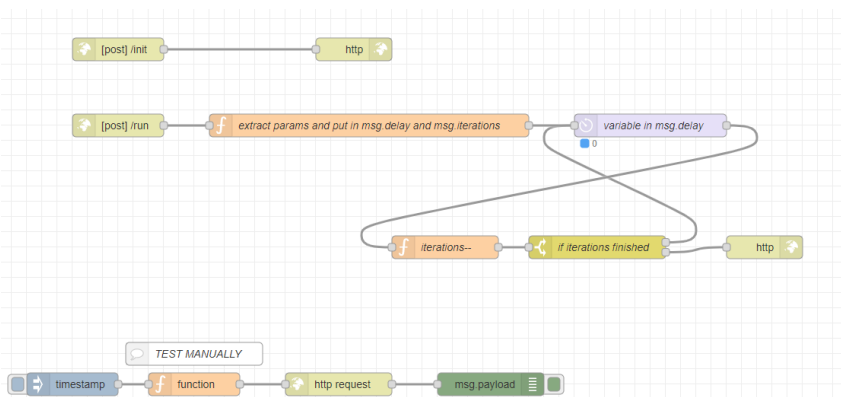
This project has received funding from the European Union's horizon 2020 research and innovation programme under grant agreement no 101017047



Baseline Technologies

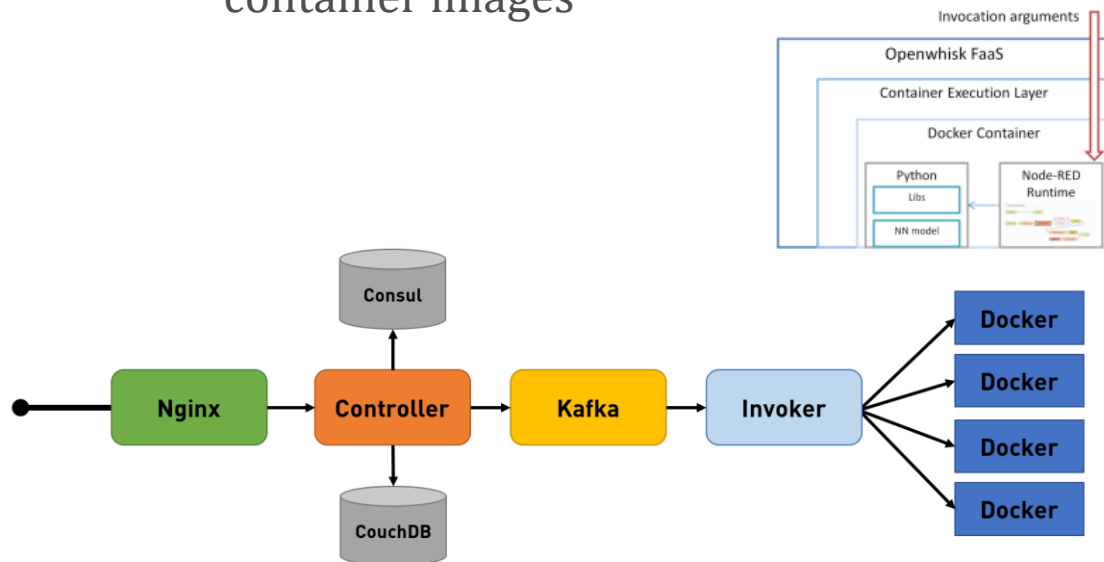
❖ Node-RED

- ❖ Programming environment for event driven applications
 - ❖ Built-in nodes
 - ❖ NPM extension nodes
 - ❖ Subflows (groups of functions) as nodes
- ❖ Combined workflow orchestration and function execution abilities
- ❖ Used as the main execution **runtime** and function **choreographer** for PHYSICS functions



❖ Openwhisk

- ❖ Open source FaaS platform
- ❖ Supported by IBM
- ❖ Main tool behind IBM Cloud Functions Service
- ❖ Can execute functions based on custom container images





Openwhisk API

- ❖ <https://petstore.swagger.io/?url=https://raw.githubusercontent.com/openwhisk/openwhisk/master/core/controller/src/main/resources/apiv1swagger.json#/>

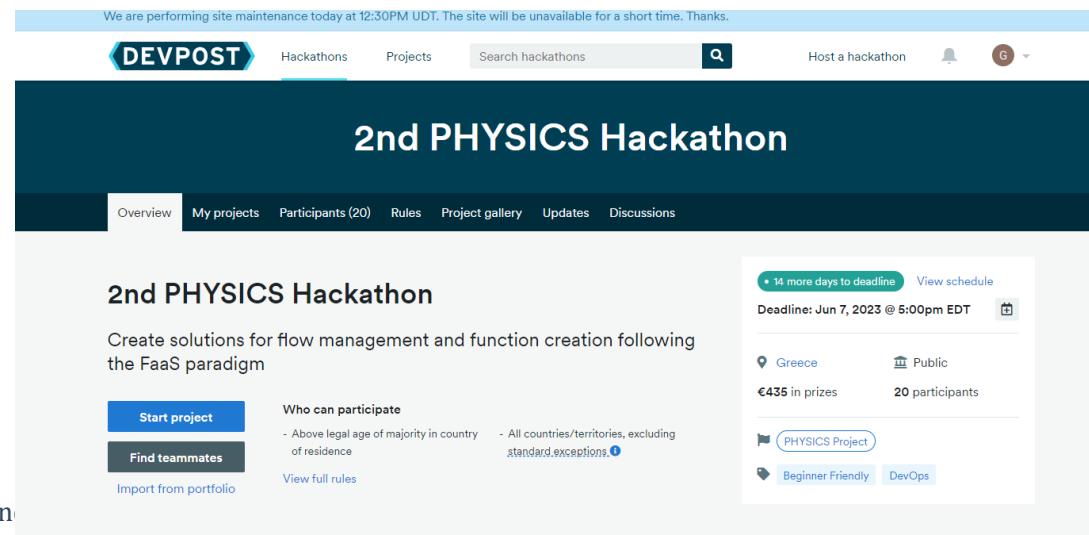
Actions		
GET	/namespaces/{namespace}/actions	Get all actions
GET	/namespaces/{namespace}/actions/{actionName}	Get action information
PUT	/namespaces/{namespace}/actions/{actionName}	Create or update an action
DELETE	/namespaces/{namespace}/actions/{actionName}	Delete an action
POST	/namespaces/{namespace}/actions/{actionName}	Invoke an action
GET	/namespaces/{namespace}/actions/{packageName}/{actionName}	Get action information
PUT	/namespaces/{namespace}/actions/{packageName}/{actionName}	Create or update an action
DELETE	/namespaces/{namespace}/actions/{packageName}/{actionName}	Delete an action
POST	/namespaces/{namespace}/actions/{packageName}/{actionName}	Invoke an action
GET	/web/{namespace}/{packageName}/{actionName}.{extension}	
PUT	/web/{namespace}/{packageName}/{actionName}.{extension}	
DELETE	/web/{namespace}/{packageName}/{actionName}.{extension}	
POST	/web/{namespace}/{packageName}/{actionName}.{extension}	





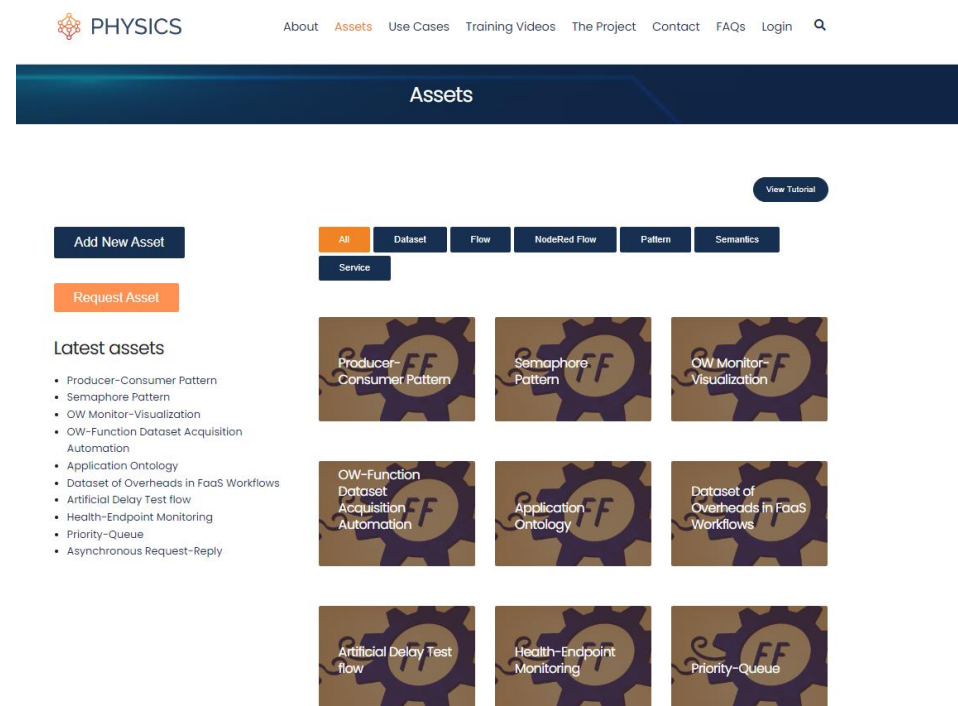
Submission process

- ❖ Individual or team participation
- ❖ Submission through the DevPost form including
 - ❖ Link to a github repo with the outputs or relevant (e.g. Docker image in dockerhub, subflow file in Node-RED repo: <https://flows.nodered.org/>)
 - ❖ Small presentation and/or video with the produced results and screenshots/demo
- ❖ Submission deadline
 - ❖ 7/6/2023
- ❖ Decisions
 - ❖ 8-9/6/2023



Prizes

- ❖ 4 amazon gift card prizes for successfully submitted and substantial artefacts/completed challenges
 - ❖ 1st place: 140
 - ❖ 2nd place: 130
 - ❖ 3rd place: 75
 - ❖ 4th place: 70
- ❖ Certificate of participation for all submissions
- ❖ Working artefacts can also be uploaded in the project marketplace for better visibility
 - ❖ <https://marketplace.physics-faas.eu/assets>





Helper Material&Channels

- ❖ <https://github.com/gkousiouris/2ndPHYSICSHackathon>
- ❖ Hello world function available in the existing Openwhisk installation
 - ❖ guest/helloLab
 - ❖ Parameters: name
 - ❖ guest/clustering
 - ❖ Input according to:
 - ❖ <https://flows.nodered.org/flow/4e1fc62b3b7040e2d5b07f3b25297c73/in/HXSkA2JJLcGA>
- ❖ [Slack channel](#)
 - ❖ https://join.slack.com/t/2ndphysicshackathon/shared_invite/zt-1vscl9myd-mq~h5WmQwPj2cnU3lvfiHQ





PHYSICS

EXAMPLE CHALLENGES

2nd PHYSICS Hackathon, 24/5/2023-7/6/2023

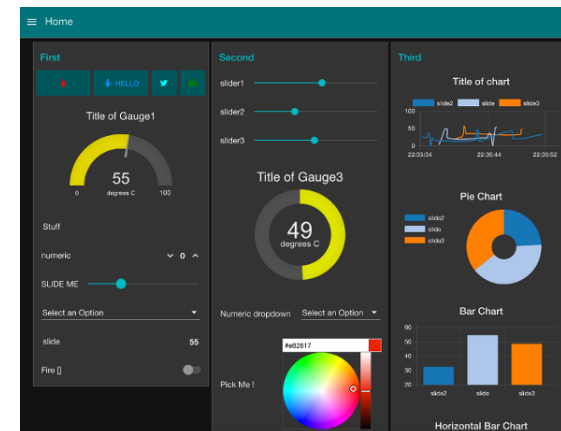


This project has received funding from the European Union's horizon 2020 research and innovation programme under grant agreement no 101017047



Visualization of Performance Data

- ❖ By using the node-red dashboard node (<https://flows.nodered.org/node/node-red-dashboard>) create suitable visualizations that portray a number of performance data files
- ❖ Examples of data inputs are given in the hackathon material
 - ❖ <https://github.com/gkousiouris/2ndPHYSICSHackathon/tree/main/dataFilesForVisualization>
- ❖ Ideally we should have different tabs for each category as well as filters based on the data fields (e.g. function name or needed information)



2ndPHYSICSHackathon / dataFilesForVisualization / loadgen-data.json

vasKatevas added json data

```
Code Blame 187 lines (187 loc) • 1.47 MB

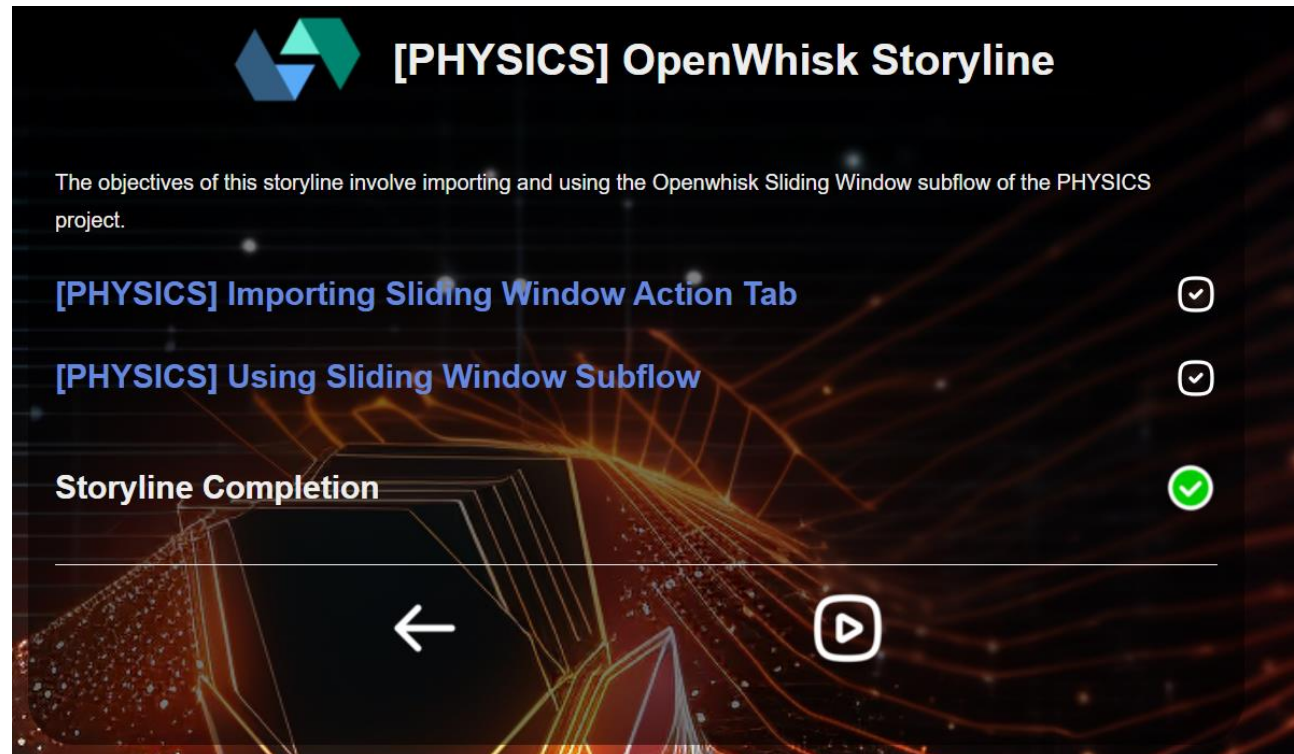
1 {
2   {
3     "output": "\\achievedAverageRate\\:45.27\\", "action": "\\fibonacci\\", "actualStartTime": "1682691462546", "averageDuration": "373.1", "averageInitTime": "0", "averageStartLatency": "17.9", "averageUserSideDelay": "391",
4     "flow": "fibonacci",
5     "activationId": "d3d77059f32401e77059f32001e62"
6   },
7   {
8     "output": "\\achievedAverageRate\\:88.04\\", "action": "\\fileRW\\", "actualStartTime": "1682692433592", "averageDuration": "7.78", "averageInitTime": "0", "averageStartLatency": "21.52", "averageUserSideDelay": "29.3",
9     "flow": "fileRW",
10    "activationId": "0825eab0c14463a5e8e0c144f58a20"
11  },
12  {
13    "output": "\\achievedAverageRate\\:64.4\\", "action": "\\list\\", "actualStartTime": "168269379904", "averageDuration": "147.47", "averageInitTime": "0", "averageStartLatency": "21.67", "averageUserSideDelay": "69.14",
14    "flow": "list",
15    "activationId": "2246d0d08c7476556d0d08c7476552"
16  },
17  {
18    "output": "\\achievedAverageRate\\:36.14\\", "action": "\\sort\\", "actualStartTime": "1682789166132", "averageDuration": "127.51", "averageInitTime": "0", "averageStartLatency": "22.44", "averageUserSideDelay": "49.95",
19    "flow": "sort",
20    "activationId": "18d230e130d4de79230e130d4de792"
21  },
22  {
23    "output": "\\achievedAverageRate\\:73.93\\", "action": "\\fileRW\\", "actualStartTime": "168302670592", "averageDuration": "7.36", "averageInitTime": "0", "averageStartLatency": "17.31", "averageUserSideDelay": "24.66",
24    "flow": "fileRW",
25    "activationId": "3d678f0636a420d978f0636a420d9e"
26  },
27  }
```





Gamification Server

- ❖ Challenge
 - ❖ Complete the scenarios listed in the Game Server
 - ❖ Evaluate the overall structure/functionality of the Game Server





Import of common algorithms as function implementations

- ❖ Example: k-means clustering algorithm

- ❖ Node-RED flow:

- <https://flows.nodered.org/flow/4e1fc62b3b7040e2d5b07f3b25297c73/in/HXSkA2JJLcGA>

- ❖ Docker Image:

- https://hub.docker.com/r/vkatevas/node-red_data_clustering

- ❖ Challenge: Create according images for other common algorithms

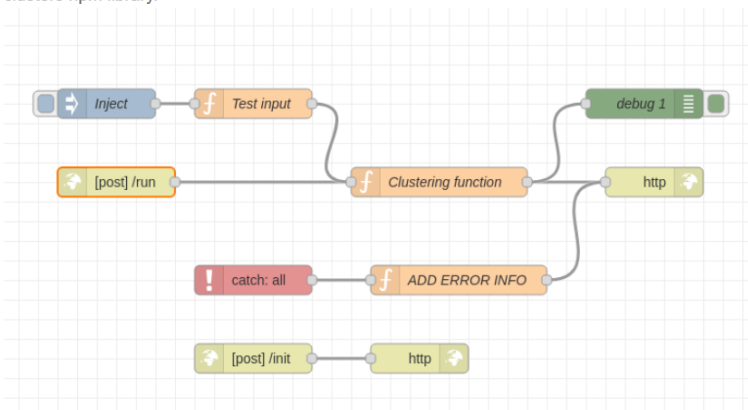
- ❖ Able to be run as Openwhisk functions

- ❖ Taking data as input parameters

- ❖ And potentially algorithm choice

K-means clustering flow

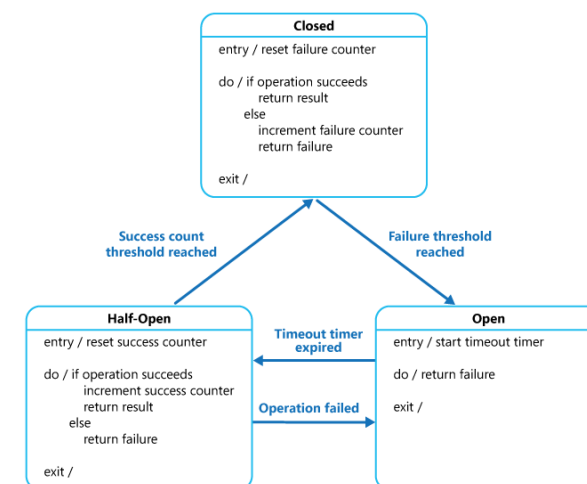
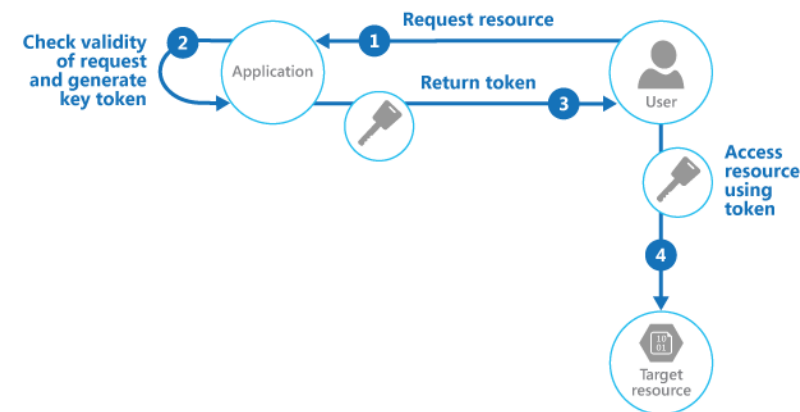
This is a flow that implements a k-means clustering operation as a service. As such it can be executed inside any Node-RED environment in a service manner. It also implements the Openwhisk API specification so that it can be executed directly as a custom docker action of Openwhisk. The inputs include arrays of objects and their values and the output returns clusters with three centroids for any given input, using the k-means implementation provided by the clusters npm library.





Cloud Design Patterns

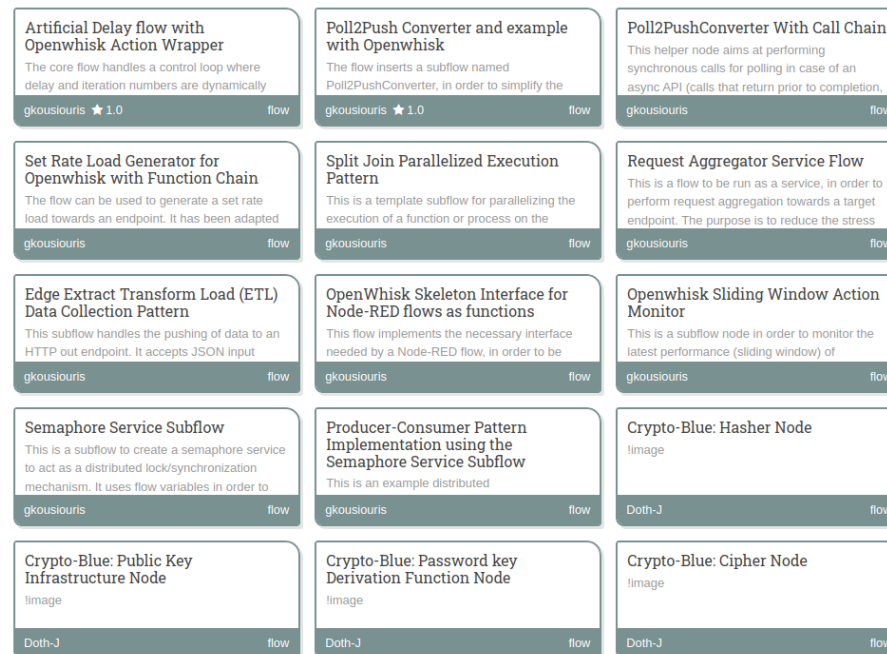
- ❖ Cloud Design Patterns are a way to define strategies that are suitable for cloud environments. Example documentation:
 - ❖ <https://learn.microsoft.com/en-us/azure/architecture/patterns/>
- ❖ Patterns are a parametric/reusable way to solve a particular problem
 - ❖ Design considerations need to take various trade-offs in mind
 - ❖ When to use them or not
 - ❖ How to set the parameters





Combination/extension of PHYSICS Pattern flows

- ❖ PHYSICS Pattern Flows
 - ❖ Includes a number of produced flows
 - ❖ <https://flows.nodered.org/collection/HXSkA2JJLcGA>
- ❖ Challenge goal is to
 - ❖ Extend the collection with other common functionalities as Node-RED subflows
 - ❖ Combine and enrich current patterns to solve a particular problem
 - ❖ Create prototypes of a selected design pattern from:
 - ❖ <https://learn.microsoft.com/en-us/azure/architecture/patterns/>



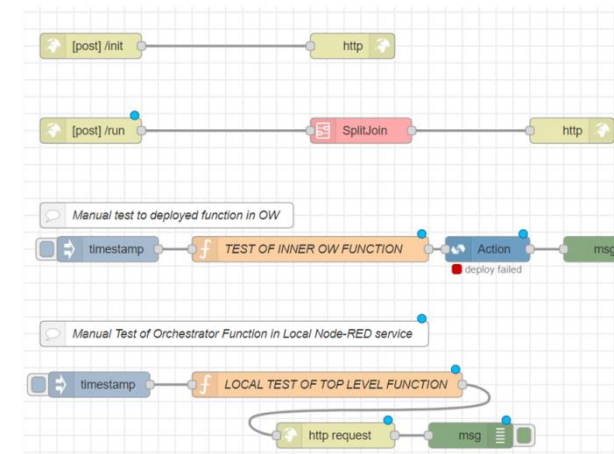


PHYSICS

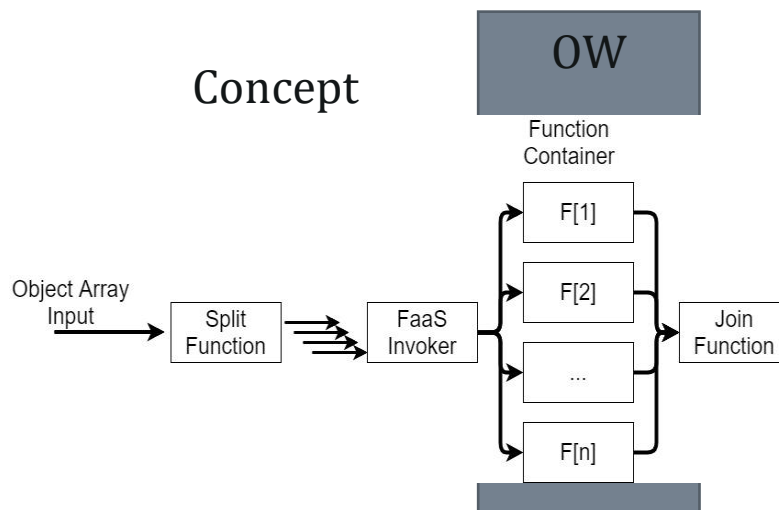
Example of
Fork-Join
parallelization
pattern

OPTIMIZED HYBRID SPACE-TIME SERVICE CONTINUUM IN FAAS

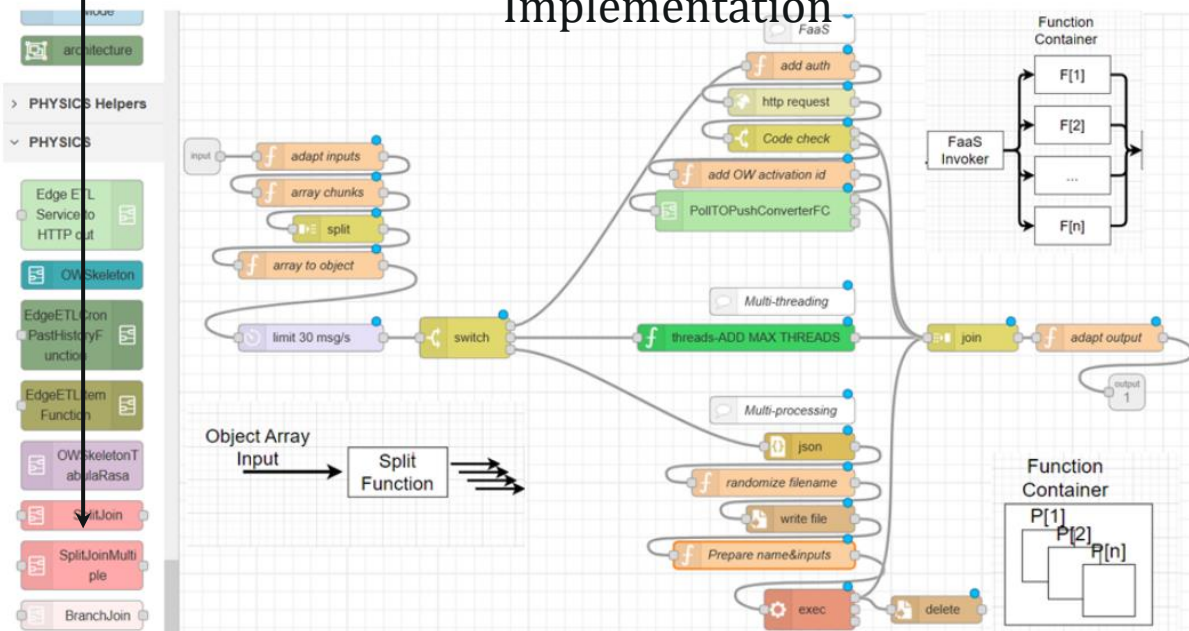
Usage in an Orchestrator Function



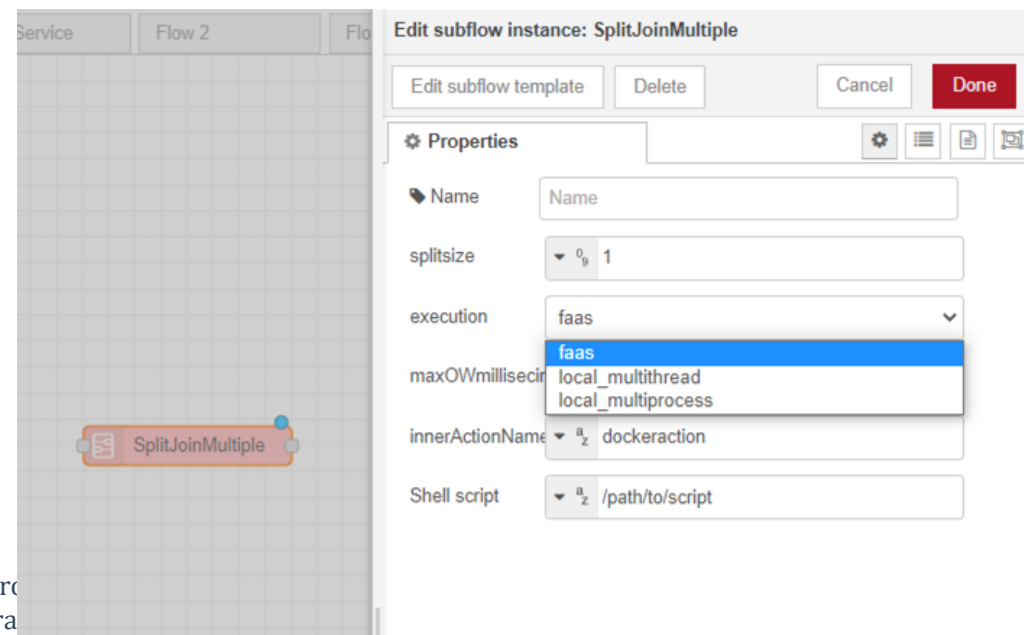
Concept



Implementation



Parameterization



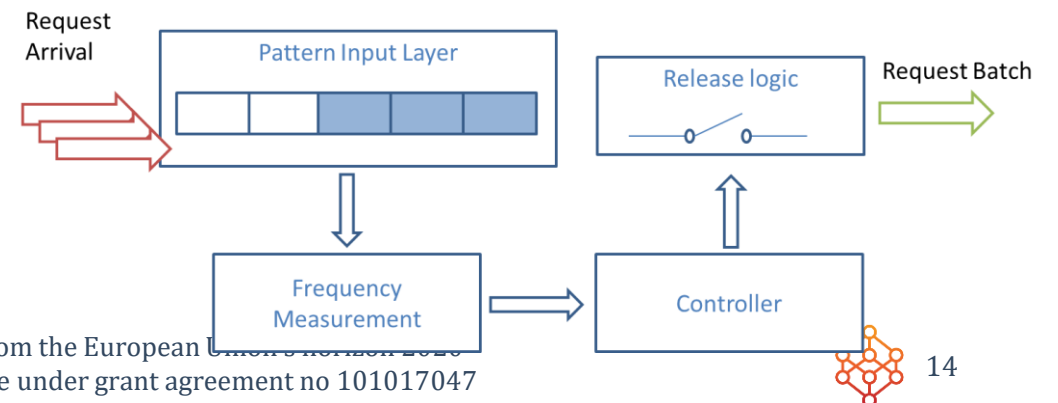
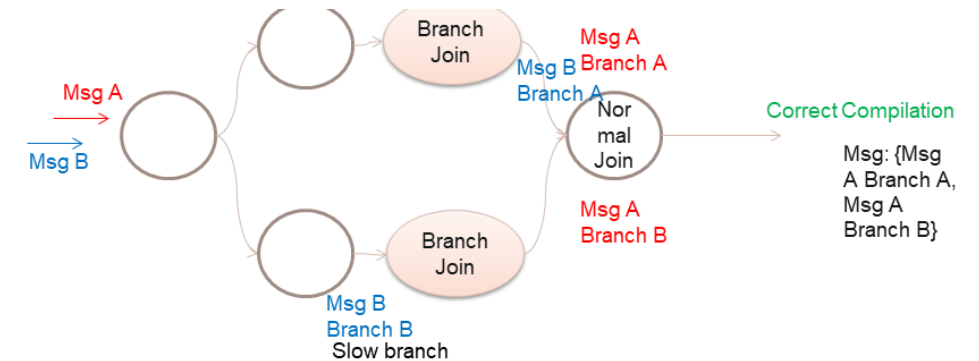
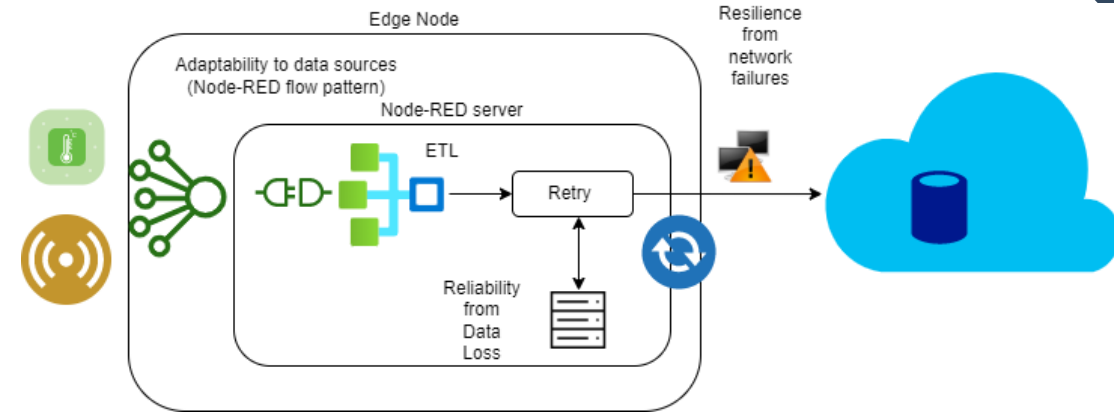
...ing from the Euro
programme under gra



Pattern flow examples

- ❖ Edge Extract-Transform-Load processes
 - ❖ Reliable data collection at the edge and pushing to a central storage
- ❖ Workflow primitives assistance
 - ❖ BranchJoin
- ❖ Request Aggregation
- ❖ Encryption and decryption
- ❖ Async read/writes
- ❖ Poll2Push Converters
 - ❖ For async APIs
- ❖ OW interface and Context Management
- ❖ Load Generation

OPTIMIZED HYBRID SPACE-TIME SERVICE CONTINUUM IN FAAS



Scoring for challenges

- ❖ 1 completed scenario from gaming challenge
 - ❖ 20 points for generic intro scenarios
 - ❖ 40 points for PHYSICS related scenarios
- ❖ Visualization scenario
 - ❖ 30 points per data file
- ❖ Common algorithm as function
 - ❖ 100 points per function
- ❖ Creation of a subflow
 - ❖ Complex subflow: 100 points
 - ❖ Simple subflow: 25 points



PHYSICS

HELPER ENVIRONMENT

2nd PHYSICS Hackathon, 24/5/2023-7/6/2023



This project has received funding from the European Union's horizon 2020 research and innovation programme under grant agreement no 101017047



Openwhisk Installation

- HUA students can use university's server
 - <https://10.100.59.208>
 - Namespace: guest
 - Credentials: 23bc46b1-71f6-4ed5-8c54-816aa4f8c502:123zO3xZCLrMN6v2BKK1dXYFpXlPkccOFqm12CdAsMgRU4VrNZ9lyGVCGuMDGIwP
- Non-HUA participants can install openwhisk themselves locally with kind or docker compose
 - install-requirements.sh and kind-setup.sh from [here](#) for the kind installation
 - *Docker compose:*
 - <https://github.com/apache/openwhisk-devtools/blob/master/docker-compose/README.md>





Install Compose

- ❖ `sudo curl -L "https://github.com/docker/compose/releases/download/1.28.6/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose`
- ❖ `sudo chmod +x /usr/local/bin/docker-compose`
- ❖ `sudo apt install zip`
- ❖ `git clone https://github.com/apache/openwhisk-devtools.git`
- ❖ `cd openwhisk-devtools/`
- ❖ `cd docker-compose`
- ❖ `make quick-start`
 - ❖ Builds and deploys

- ❖ Stopping
 - ❖ `docker-compose --project-name openwhisk stop`
- ❖ Starting again
 - ❖ `docker-compose --project-name openwhisk start`

- ❖ Το μόνο πρόβλημα με αυτή τη προσέγγιση είναι ότι γράφει στο tmp file
- ❖ Αλλά αν δεν γίνει restart ο node τότε θα κρατηθούν τα διάφορα functions klp



Install Openwhisk cli

- ❖ Download cli targz for platform
 - ❖ <https://github.com/apache/openwhisk-cli/releases>
- ❖ `tar -zxvf <clifile>`
- ❖ Instructions
 - ❖ <https://github.com/apache/openwhisk/blob/master/docs/cli.md>





Cli configuration

- ❖ Compose installation on HUA:
- ❖ `./wsk property set --apihost http://10.100.59.208`
- ❖ `./wsk property set --auth 23bc46b1-71f6-4ed5-8c54-816aa4f8c502:123zO3xZCLrMN6v2BKK1dXYFpXlPkccOFqm12CdAsMgRU4VrNZ9lyGVCGuMDGIwP`
- ❖ Main instructions
 - ❖ `wsk action create -i`
 - ❖ `wsk action invoke -i`
 - ❖ `wsk activation get -i`





PHYSICS

CUSTOM FUNCTION BASED ON NODE-RED RUNTIME TUTORIAL

2nd PHYSICS Hackathon, 24/5/2023-7/6/2023



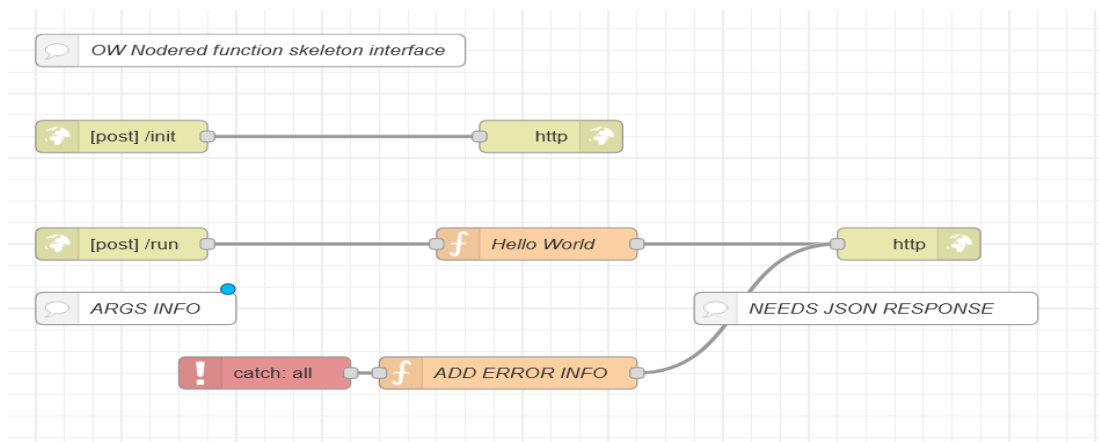
This project has received funding from the European Union's horizon 2020 research and innovation programme under grant agreement no 101017047



Openwhisk Custom Docker Image as Function-Template skeleton flow

- ❖ <https://flows.nodered.org/flow/d010a0a5af458e093342420349d63773/in/HXSkA2JLcGA>

- ❖ Includes instructions on image creation



```
FROM nodered/node-red
RUN npm install <any needed node-red packaged node for your flows from npm>
#move arbitrary flow from local dir
COPY myflow.json /data/flows.json
#optional: add a specific settings.js file
COPY settings.js /data/settings.js
#optional: add a specific flows_cred.js file with credentials. If used then the
#credentialSecret option should be set in the settings.js file during credential
#creation in the development Node-RED server so that credentials are transferable
#to the new image
COPY flows_cred.json /data/flows_cred.json
ENV PORT 8080
EXPOSE 8080:8080
```



Parameter passing considerations

- ❖ Openwhisk intervenes in the way our parameters in the request (or in the body of the POST HTTP call) get passed in the flow
 - ❖ It includes an extra field `msg.payload.value.<param_name>`
 - ❖ This needs to be considered in the follow-up nodes when we try to extract the parameters

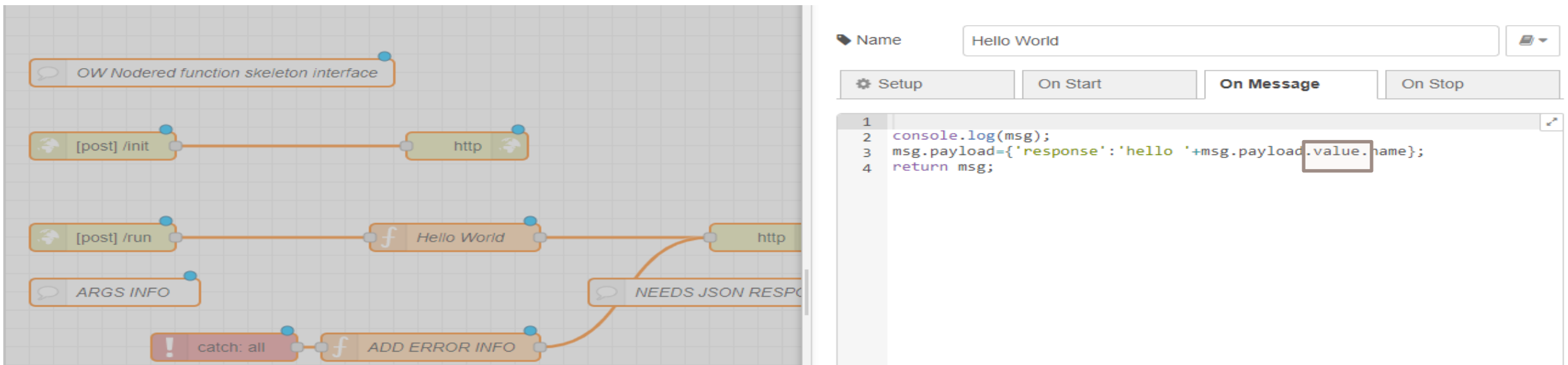




Image building and Action Registration

Then the image can be built (in the same dir):

```
docker build -t <docker_registry_account>/<image_name> .
```

and pushed:

```
docker push <docker_registry_account>/<image_name>
```

Following the creation of the image and its pushing to a registry, it can be registered in Openwhisk as an action through the following command:

```
wsk action create <action_name> --docker <docker_registry_account>/<image_name>
```

and invoked through:

```
wsk action invoke <action_name> --param name george
```



Practice

- ❖ Try to create the image based on the previous slides and the available skeleton flow
- ❖ Register the action in the Openwhisk testbed with a name like `hello_<YOUR_IT>`



PHYSICS

Thank you for the attention!
George Kousiouris, HUA



www.physics-faas.eu



[linkedin.com/company/physicsh2020](https://www.linkedin.com/company/physicsh2020)



<https://twitter.com/H2020Physics>



info@physics-faas.eu



This project has received funding from the European Union's horizon 2020 research and innovation programme under grant agreement no 101017047