Workflow Configuration Import and Validation for AliECS

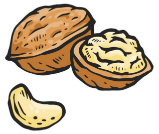
Progress Report
22nd July, 2020

By : Ayaan Zaidi
Mentored by : Teo Mrnjavac

Goals

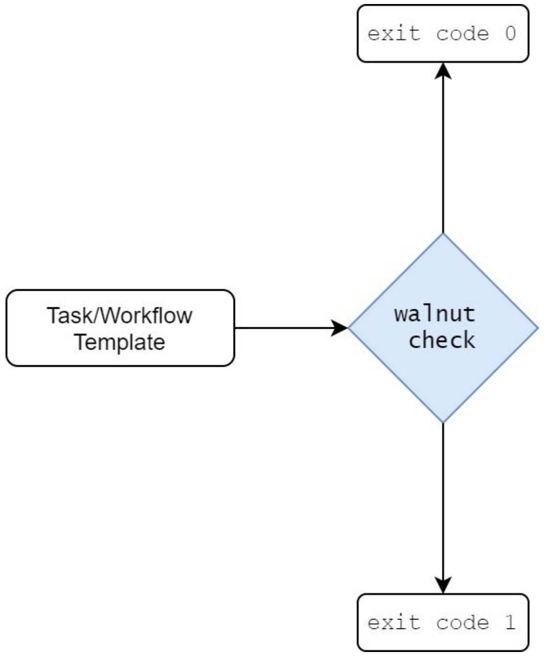
- Convert a DPL Dump generated by O2/DPL into required number of task templates and one workflow template
- Design JSON schemas that describe a structure/pattern for these templates
- Develop a **package to validate** said templates against the schemas without conversion from YAML to JSON or vice versa

All of the above being developed in a package called `walnut` - **W**orkflow **A**dministration and **L**i**n**ting **Ut**ility

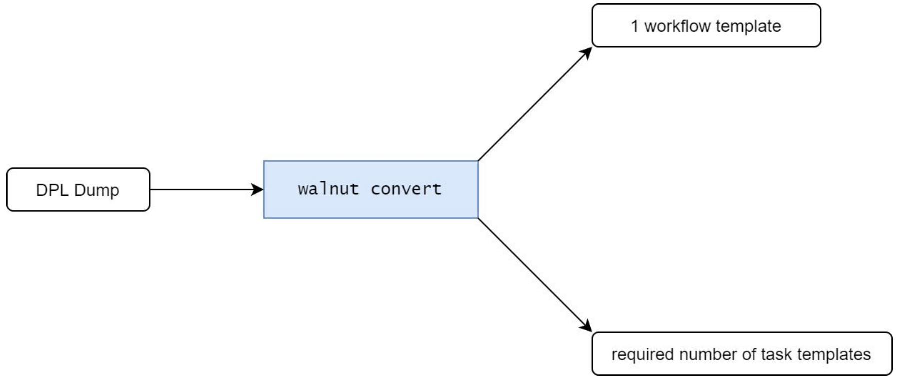


Goals

`walnut check`



`walnut convert`



Validation

Requirements

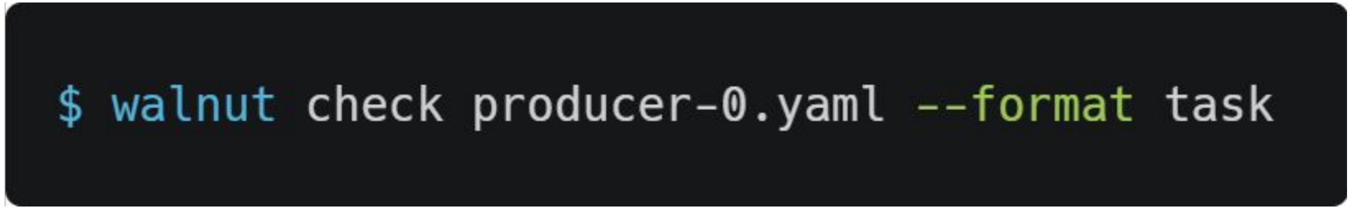
- Define **formal schemas** for AliECS workflow configuration formats (task templates and workflow templates, both of which weren't subject to a formal schema until now).
- Build a package that makes use of these schemas to perform **validation of workflow and task templates** provided as input.

Implementation

- Two schemas (one for WFTs and one for TTs) were defined. These adhere to the requirements defined by AliECS. Currently in the final stages of development.
- Package `schemata` was built that allows the user to verify if a workflow or task template adheres to the aforementioned schema **without conversion** from YAML to JSON.
- Available on the walnut branch of AliceO2Group/Control.

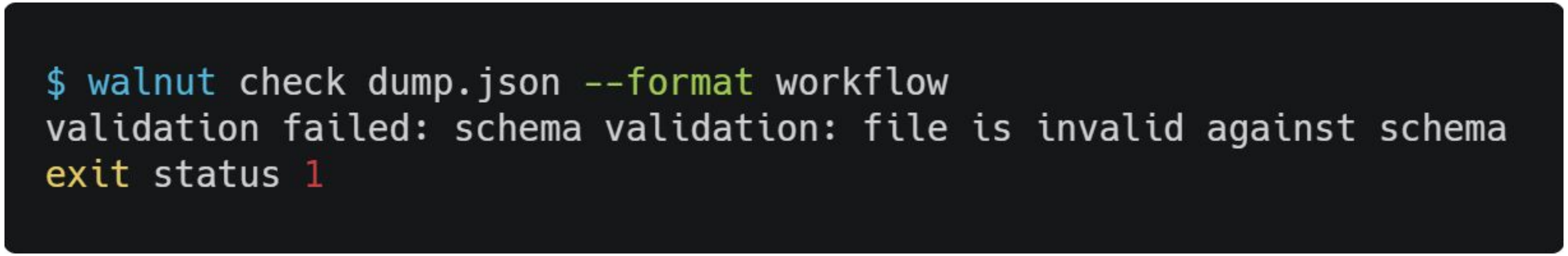
Validation – Example

The user provides a workflow or task template:



```
$ walnut check producer-0.yaml --format task
```

Upon successful validation, the process exits cleanly. If validation fails, walnut exits with exit code 1.

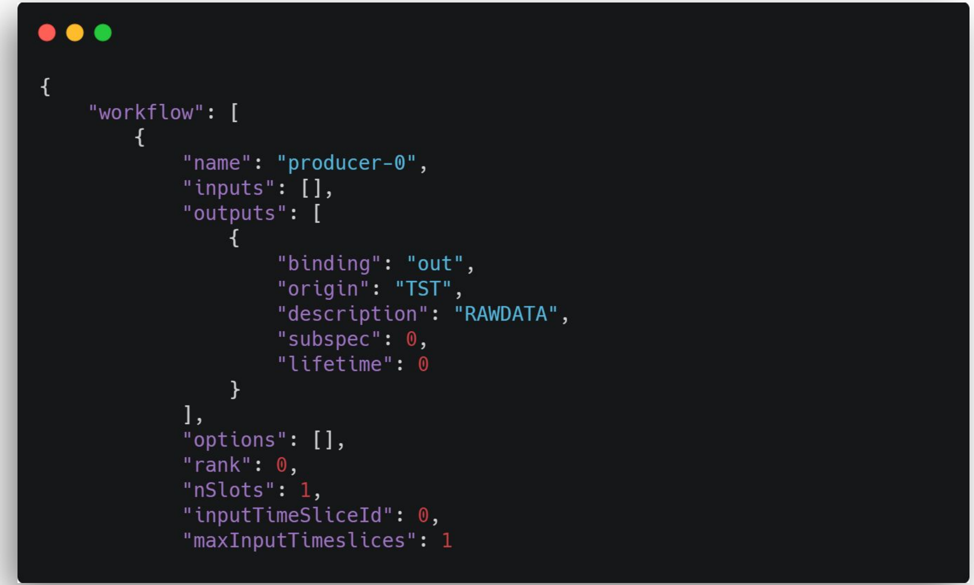


```
$ walnut check dump.json --format workflow
validation failed: schema validation: file is invalid against schema
exit status 1
```

Conversion

Requirements

- Convert an input DPL dump to workflow and task template formats that AliECS can work with.
- Ensure that any DPL dump can be converted with **minimal or no additional input** from the user.

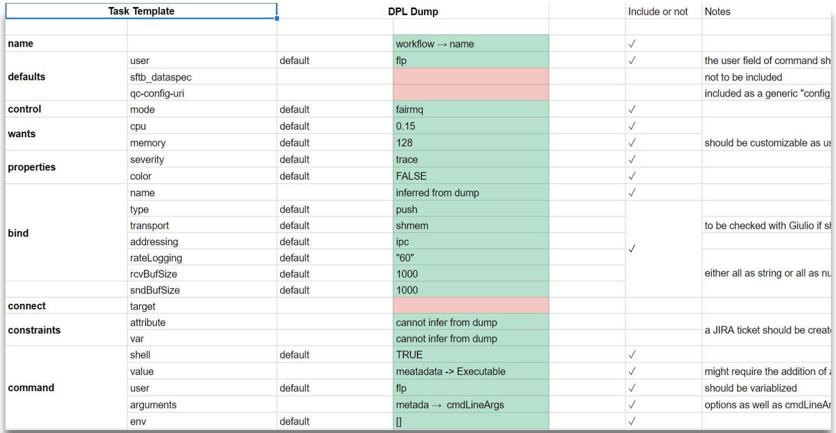


```
{
    "workflow": [
        {
            "name": "producer-0",
            "inputs": [],
            "outputs": [
                {
                    "binding": "out",
                    "origin": "TST",
                    "description": "RAWDATA",
                    "subspec": 0,
                    "lifetime": 0
                }
            ],
            "options": [],
            "rank": 0,
            "nSlots": 1,
            "inputTimeSliceId": 0,
            "maxInputTimeslices": 1
```

Conversion – Implementation

Implementation

- Began with matching each item in a DPL dump with its equivalent in the AliECS task template format.
- Deliberate on each value: what we need, what we get, what we don’t, what we can default and what we cannot.
- Giulio Eulisse extended the DPL dump format on request, to provide additional information such as channel names, which we could then use during conversion.

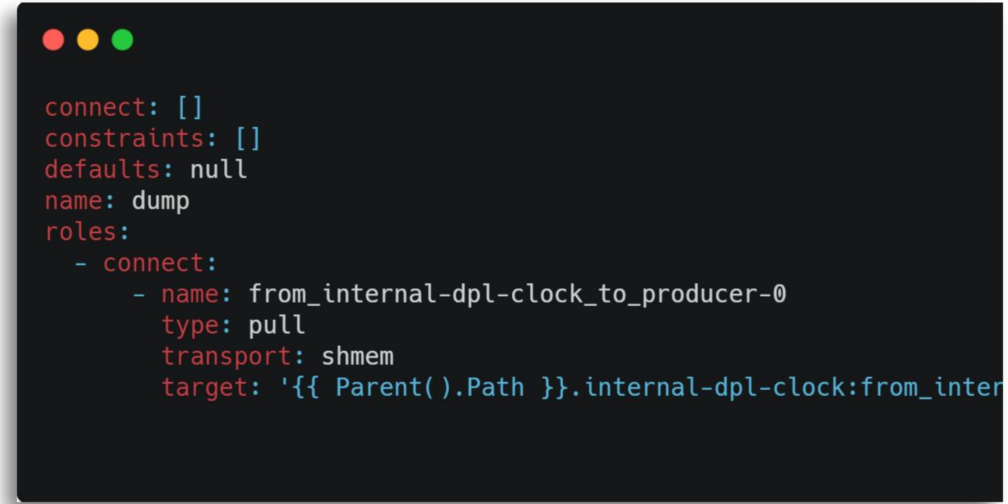


Task Template			DPL Dump		Include or not	Notes
name			workflow → name		✓	
defaults	user	default	flp		✓	the user field of command sh
	sftb_dataspec		🟥			not to be included
	qc-config-uri		🟥			included as a generic "config
control	mode	default	fairmq		✓	
wants	cpu	default	0.15		✓	
	memory	default	128		✓	should be customizable as us
properties	severity	default	trace		✓	
	color	default	FALSE		✓	
bind	name		inferred from dump		✓	
	type	default	push		✓	
	transport	default	shmem			to be checked with Giulio if sh
	addressing	default	ipc			
	rateLogging	default	"60"			
	rcvBufSize	default	1000			either all as string or all as nu
	sndBufSize	default	1000			
connect	target		🟥			
constraints	attribute		cannot infer from dump			a JIRA ticket should be creat
	var		cannot infer from dump			
command	shell	default	TRUE		✓	
	value		meatadata -> Executable		✓	might require the addition of a
	user	default	flp		✓	should be variablized
	arguments		metada → cmdLineArgs		✓	options as well as cmdLineAr
	env	default	[]		✓	

Conversion – Implementation

Implementation

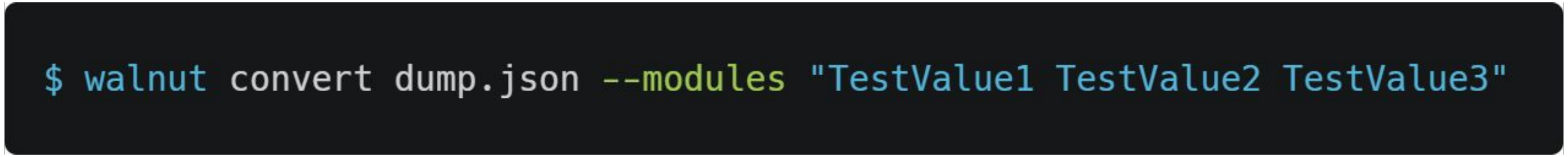
- The implementation of workflow template generation takes advantage of the prior effort on task templates.
- Rather than creating new handlers for WFTs, reused handlers built for conversion of TTs.
- Successful conversion of DPL dump to workflow and task templates was achieved.



```
connect: []
constraints: []
defaults: null
name: dump
roles:
  - connect:
      - name: from_internal-dpl-clock_to_producer-0
        type: pull
        transport: shmem
        target: '{{ Parent().Path }}.internal-dpl-clock:from_inter
```

Conversion – Example

The user provides one or more DPL dumps (as well as some additional flags to provide information which the DPL dump doesn't contain, like `alienv` modules):



```
$ walnut convert dump.json --modules "TestValue1 TestValue2 TestValue3"
```

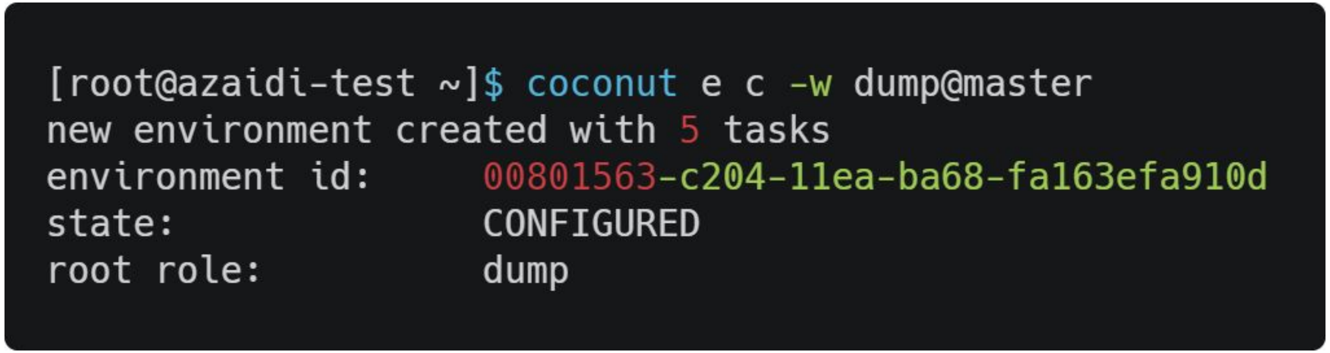
A successful conversion will result in:

- One unified directory for all DPL dumps provided
- Each folder will have subdirectories for tasks and workflows

All the code can be found at AliceO2Group/Control.

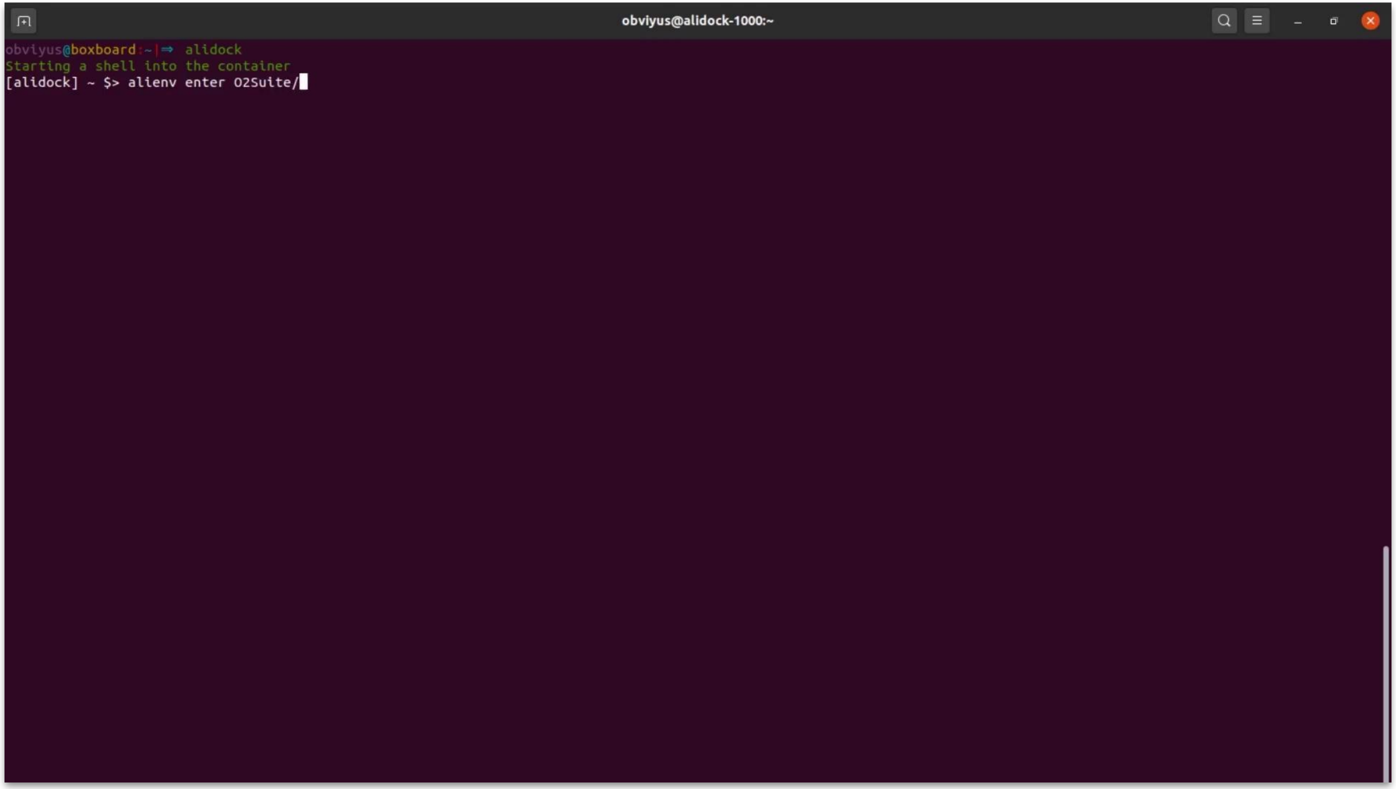
Workflow Deployment

Once the converted WFTs and TTs are placed into a git repository, committed and pushed, they can be accessed from `coconut`. From here, they can be used to create environments:



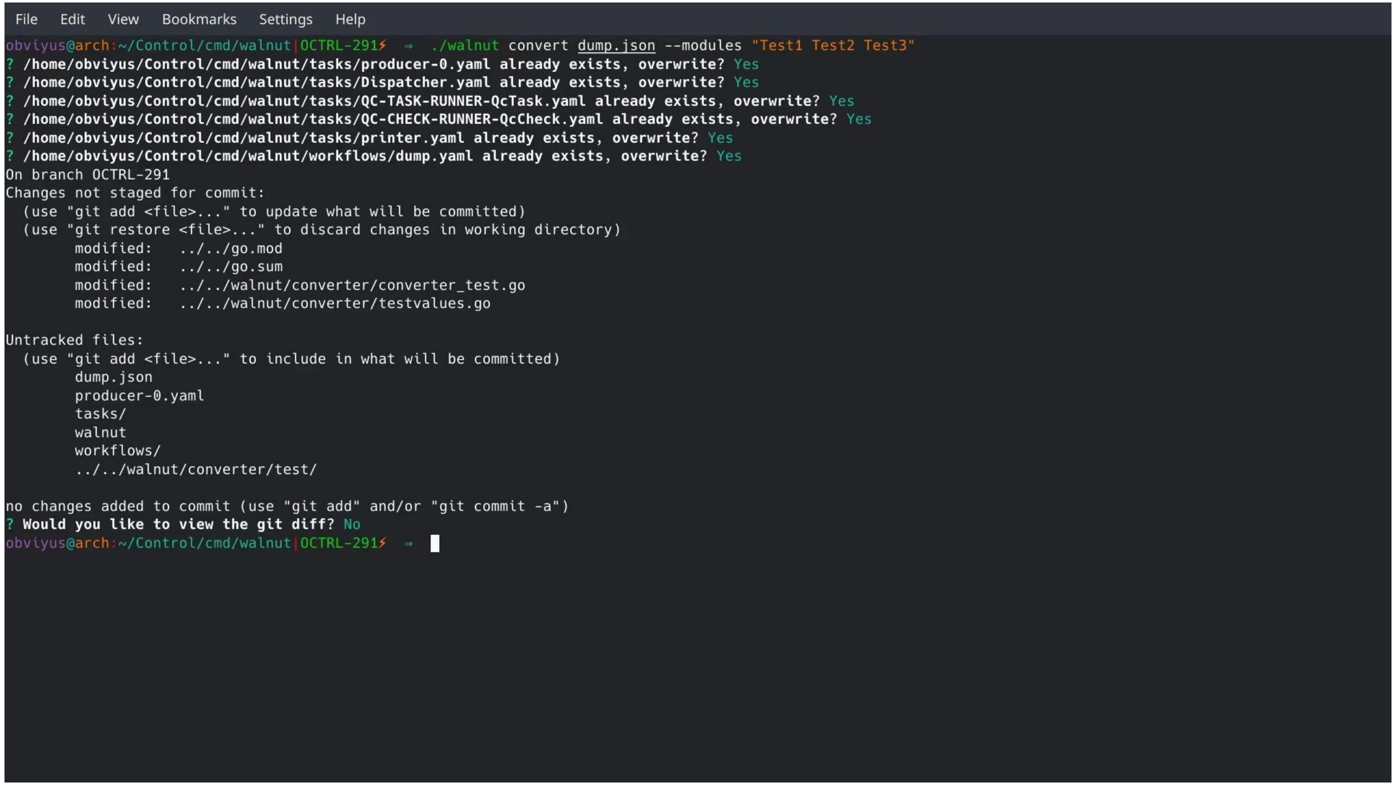
```
[root@azaidi-test ~]$ coconut e c -w dump@master
new environment created with 5 tasks
environment id:     00801563-c204-11ea-ba68-fa163efa910d
state:              CONFIGURED
root role:          dump
```

Demo: Generating DPL Dumps


obviyus@alidock-1000:~

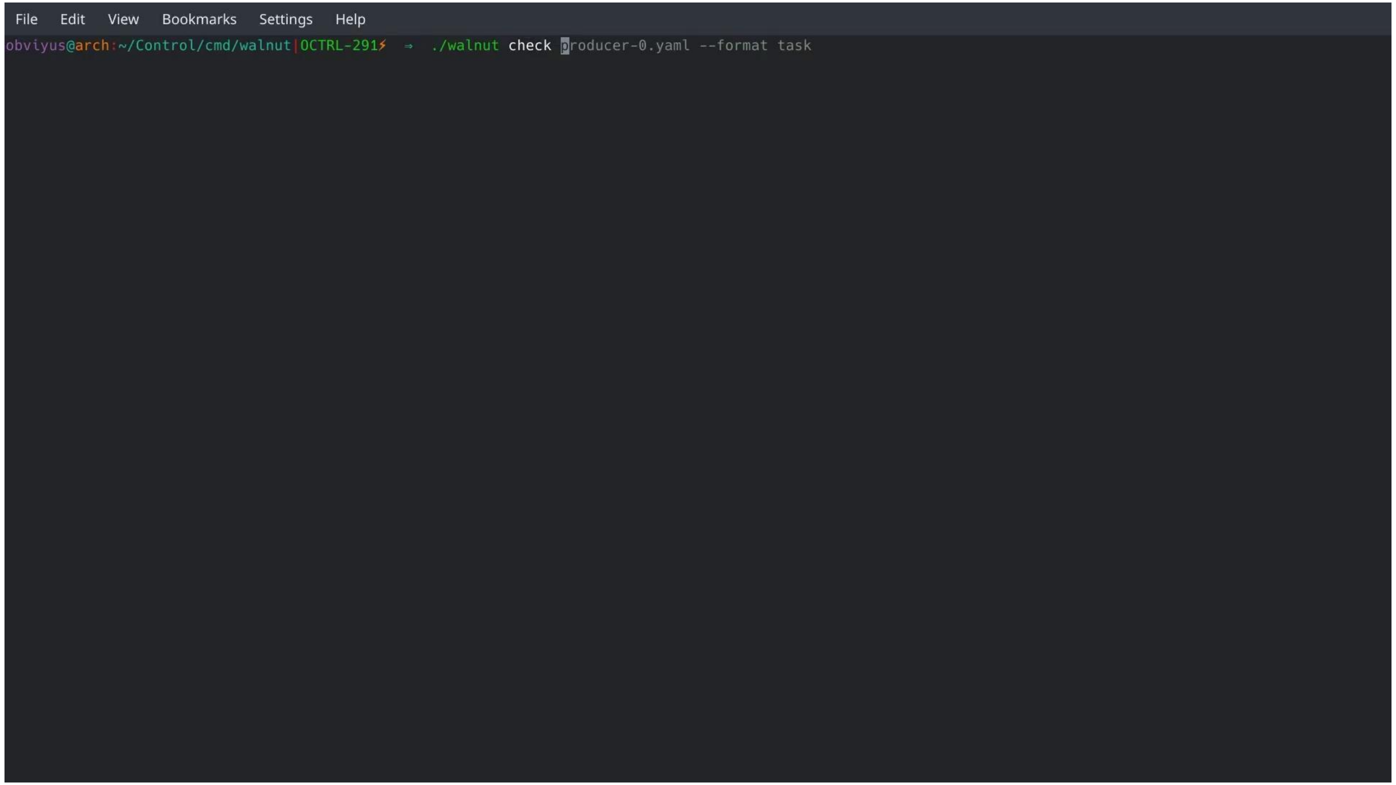
```
obviyus@boxboard:~|⇒  alidock
Starting a shell into the container
[alidock] ~ $> alienv enter O2Suite/
```

Demo: walnut convert



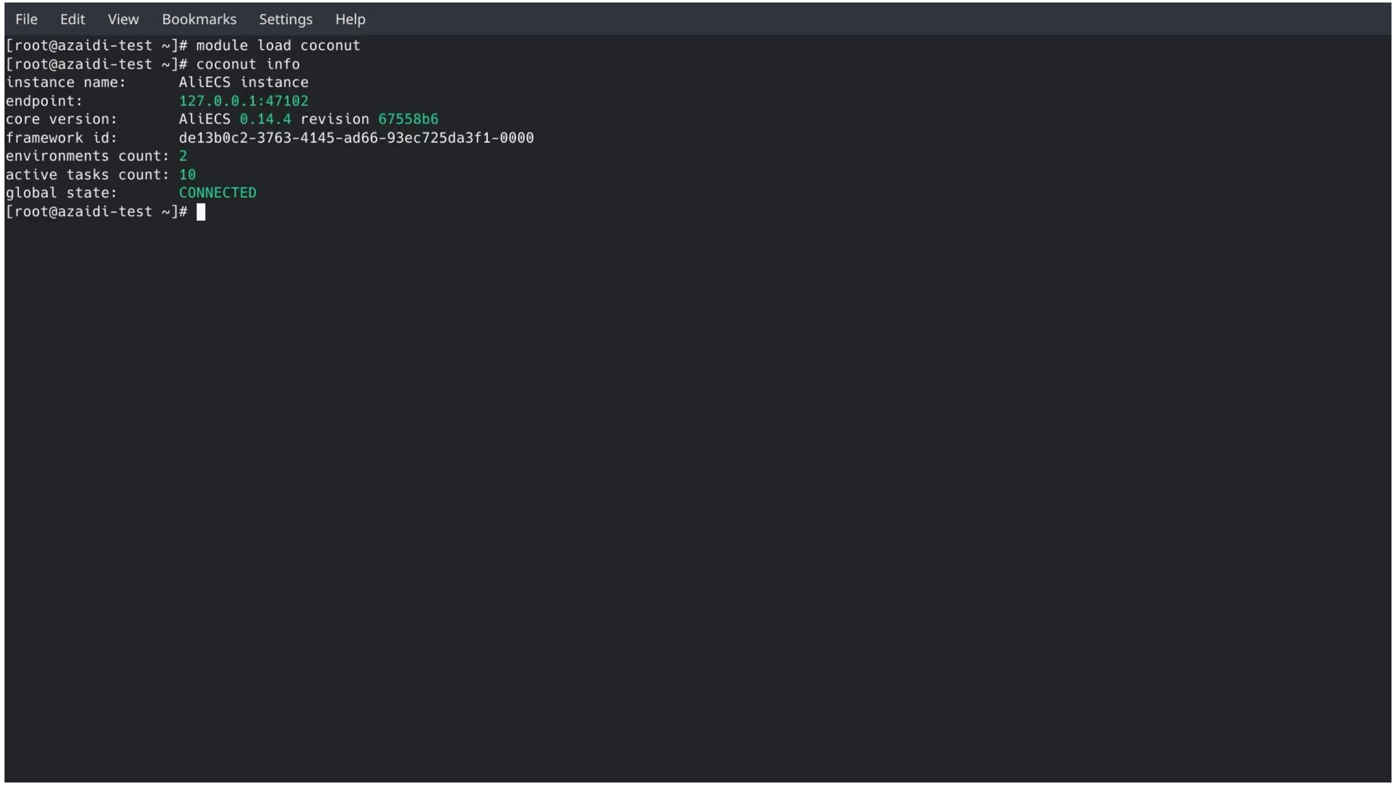
```
obviyus@arch:~/Control/cmd/walnut|OCTRL-291⚡  ⇒  ./walnut convert dump.json --modules "Test1 Test2 Test3"
? /home/obviyus/Control/cmd/walnut/tasks/producer-0.yaml already exists, overwrite? Yes
? /home/obviyus/Control/cmd/walnut/tasks/Dispatcher.yaml already exists, overwrite? Yes
? /home/obviyus/Control/cmd/walnut/tasks/QC-TASK-RUNNER-QcTask.yaml already exists, overwrite? Yes
? /home/obviyus/Control/cmd/walnut/tasks/QC-CHECK-RUNNER-QcCheck.yaml already exists, overwrite? Yes
? /home/obviyus/Control/cmd/walnut/tasks/printer.yaml already exists, overwrite? Yes
? /home/obviyus/Control/cmd/walnut/workflows/dump.yaml already exists, overwrite? Yes
On branch OCTRL-291
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   ../../go.mod
        modified:   ../../go.sum
        modified:   ../../walnut/converter/converter_test.go
        modified:   ../../walnut/converter/testvalues.go

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        dump.json
        producer-0.yaml
        tasks/
        walnut
        workflows/
        ../../walnut/converter/test/

no changes added to commit (use "git add" and/or "git commit -a")
? Would you like to view the git diff? No
obviyus@arch:~/Control/cmd/walnut|OCTRL-291⚡  ⇒
```

Demo: walnut check



```
obviyus@arch:~/Control/cmd/walnut|OCTRL-291⚡  ⇒  ./walnut check producer-0.yaml --format task
```

Demo: workflow deployment



```
[root@azaidi-test ~]# module load coconut
[root@azaidi-test ~]# coconut info
instance name:      AliECS instance
endpoint:           127.0.0.1:47102
core version:       AliECS 0.14.4 revision 67558b6
framework id:       de13b0c2-3763-4145-ad66-93ec725da3f1-0000
environments count: 2
active tasks count: 10
global state:       CONNECTED
[root@azaidi-test ~]#
```

Future Developments

- The **minimum viable product** milestone was achieved, on which we base any further work.
- Future efforts range from:
  - minor bug fixes
  - further enhancement of output structures
  - integration of additional data sources, such as Git, Consul and hardcoded defaults
  - the ability to amend existing workflow templates.
- We are also waiting for some additional changes to the DPL output in order to **replace some potentially-unreliable guesswork** with DPL-provided facts. This will allow us to avoid guessing what each value should hold and/or resorting to the default values.

Thank you.