COLONIES - COMPUTE CONTINUUMS ACROSS PLATFORMS

A PREPRINT

Johan Kristiansson
Department of Computer Science
RISE Research Institutes of Sweden
Luleå, Sweden
johan.kristiansson@ri.se

Thomas Ohlson Timoudas
Department of Computer Science
RISE Research Institutes of Sweden
Luleå, Sweden
thomas.ohlson.timoudas@ri.se

Henrik Forsgren
Department of Computer Science
RISE Research Institutes of Sweden
Luleå, Sweden
thomas.ohlson.timoudas@ri.se

March 27, 2023

ABSTRACT

This paper presents a novel framework for managing computational workload across heterogeneous platforms. Colonies is based on a loosly coupled microservice architecture where complex workflows are broken down in composable functions that are executed by independently deployable executors using an API service. In this way, workflows can be expressed declarative in any computer language and be executed across platforms by so-called executors independently deployed in the cloud, edge, devices, or even in browser applications, creating compute continuums across platforms. The paper

Keywords Serverless computing · Parallel computing · Workflow orchestration

1 Introduction

TODO

2 The Colonies framework

2.1 Architecture

TODO

2.1.1 Workflows

TODO

Table 1: Function Specifications

Function Spec	Function	Executor Type	Priority	Max Exec Time	Max Retries
F_1	gen_nums()	Edge	1	200 s	5
F_2	square()	Cloud	1	200 s	5
F_3	square()	Cloud	1	200 s	5
F_4	sum()	Browser	1	200 s	5

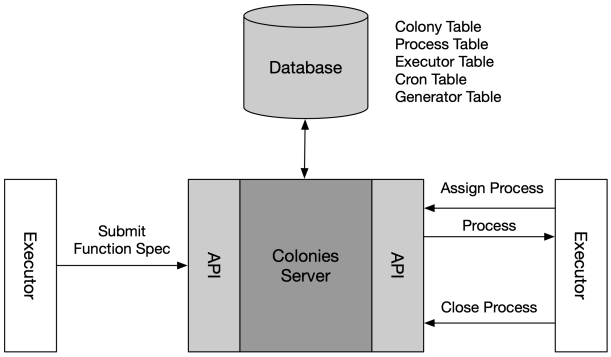

Figure 1: cron management

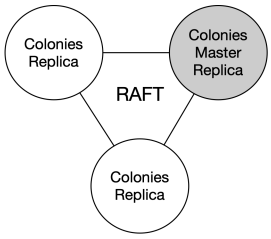

Figure 2: cron management

dt = -1000000000 * 60 * 60 * 24 process.PriorityTime = int64(process.FunctionSpec.Priority)*dt + submission-Time.UnixNano()

2.1.2 Cron

TODO

2.1.3 Generators

TODO

2.1.4 Zero-trust security

TODO

3 Evaluation

3.1 Implementation

```
gen_nums = Function(gen_data, colonyid, executortype="edge")
square1 = Function(square, colonyid, executortype="cloud")
square2 = Function(square, colonyid, executortype="cloud")
sum = Function(square, colonyid, executortype="browser")

wf = ColoniesWorkflow("localhost", 50080, colonyid, executor_prvkey)
wf >> gennums
gennums >> square1
gennums >> square2
[square1, square2] >> sum
res = wf.execute()
```

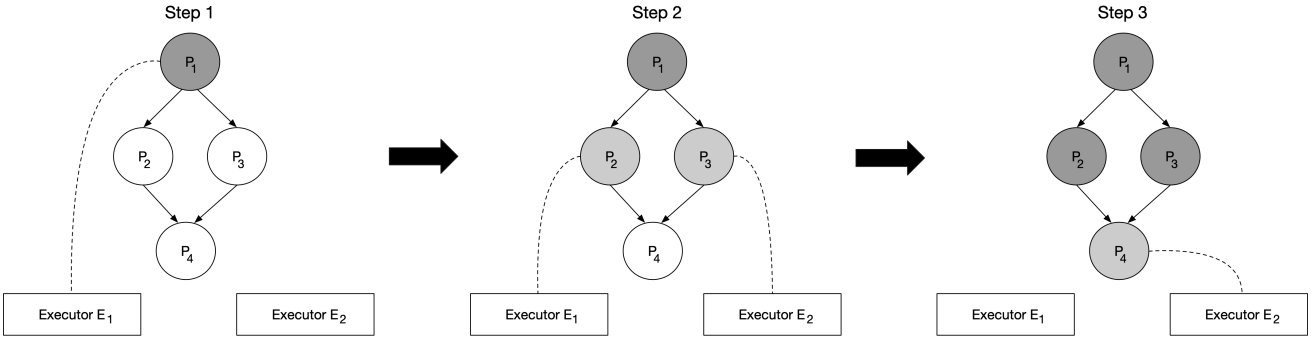

Figure 3: cron management

Table 2: Snapshot of Process Table as in Step 2

Process Id	Function Spec	Wait for Parents	Assigned Executor Id	State	Priority Time
P_1	F_1	$False$	E_1	Successful	1679906715352024000
P_2	F_2	$False$	E_1	Running	1679906715353453000
P_3	F_3	$False$	E_2	Running	1679906715354286000
P_4	F_4	$True$	-	Waiting	1679906715355188000

3.2 References

TODO

Table 3: Dependency Table

Process Id	Name	Dependencies
P_1	$Task_1$	-
P_2	$Task_2$	$Task_1$
P_3	$Task_3$	$Task_1$
P_4	$Task_4$	$Task_2$, $Task_3$

Table 4: Input/Output Table

Process Id	Input	Output
P_1		[2,3]
P_2	2	4
P_3	3	9
P_4	[4,9]	13

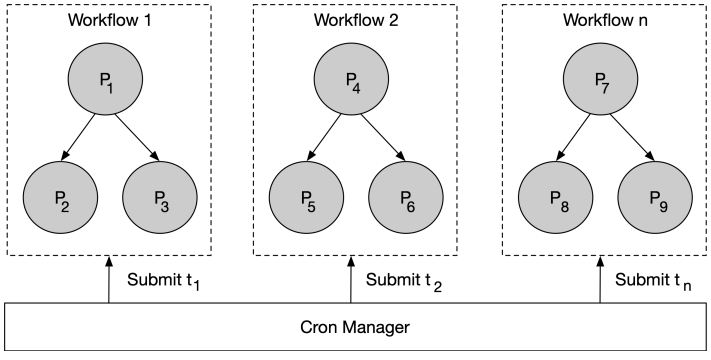

Figure 4: Sample figure caption.