Brane
A Framework for Programmable Orchestration of Multi-Site Applications

Onno Valkering, Reginald Cushing, Adam Belloum
o.a.b.valkering@uva.nl

Multiscale Networked Systems (MNS)
University of Amsterdam

https://mns-research.nl
https://epi-project.nl
https://github.com/onnovalkering/brane
Table of Contents

• Multi-site applications
  • Context
  • Challenges

• The Brane framework
  • Separation of concerns
  • Runtime system

• Programming Model
  • Packages
  • DSLs

• Future work
A single infrastructure provider might not be able to fulfill all project’s compute and storage requirements.

Privacy or security constraints might restrict centralized aggregation and demand federated data processing.

https://process-project.eu

https://epi-project.nl
Multi-site applications

Challenges

• Brane aims to streamline and simplify development and deployment of complex multi-site scientific applications.

  • How to develop locally, but run it distributed on heterogeneous infrastructures?
  • How can globally dispersed research teams and organizations collaborate?
  • How to monitor and control applications that run on a dispersed resource pool?
  • How to appropriately and securely handle sensitive data in a multi-domain setting?

• If not addressed, this may lead to delayed or missed innovation!
The Brane framework

Separation of concerns

• Focus on both **technical** and **organizational** challenges.
• Accommodate different roles with different tools.

1) **Engineers**: prepare building blocks
2) **Scientists**: compose applications
3) **Brane**: deploy (almost) anywhere

---

Domain-specific languages (DSLs)

---

HPC Cloud

- **netherlands Scientific Center**
- **XENON**
- **slurm**
- **docker**
- **kubernetes**
The Brane framework
Separation of concerns

• Focus on both **technical** and **organizational** challenges.
• Accommodate different roles with different tools.

---

1) **Engineers:** prepare building blocks
2) **Scientists:** compose applications
3) **Brane:** deploy (almost) anywhere
The Brane framework

Runtime system

- The runtime system is a collection of loosely coupled microservices.

- The centralized runtime system relies, by default, on direct control of compute and storage resources.
The Brane framework

**Runtime system**

- The runtime system is a collection of loosely coupled **microservices**.

- The **centralized** runtime system relies, by default, on direct control of compute and storage resources.

- If direct control is not desired, an **ambassador** service can be used on-site as a **layer of indirection**.
Programming model

Packages

- Brane currently offers four **package builders**.
- Packages are **self-contained** and in the OCI image format.

Package builders

- **Config**: Code, CWL, OpenAPI, DSL

BuildKit

Function definitions

Functionality implementation

branelet

Brane (runtime system)

Config

Runtimes

Orchestration

Functionalities

Packages
Programming model

**DSLs**

- Two domain-specific languages as glue between Brane packages.
  - **Bakery** requires limited to no programming experience.
  - **BraneScript** is a more complete and C-like language.
Programming model

**DSLs**

- Two domain-specific languages as glue between Brane packages.
  - Bakery requires limited to no programming experience.
  - BraneScript is a more complete and C-like language.
Programming model

DSLs

```java
// Brings the relevant functions into scope
import filesystem, lofar-lta, prefactor;

// ID of the LOFAR observation is the input
observation := 246403;
directory := new_directory;

// Files are staged from tape drives to a cache (remote)
staging := stage observation files;
wait until staging status = "success";

archives := download observation files to directory
measuresets := extract archives to directory;

skymap := calibrate measuresets;
return skymap; // The sky map is the output
```

Bakery
Programming model

**DSLs**

```branescript
import filesystem, lofar-1ta, prefactor;

on "location_1" {
  f();
}

on queryLocations("AMS") {
  g();
}

skymap := calibrate measuresets;
return skymap; // The sky map is the output
```

```branescript
// Brings the relevant functions into scope
import distributed_dl;

let world_size := 3;

on "node1" {
  // The master service is running on node 1
  let master := startMaster(world_size);
  master.waitForStarted();

  // After the master service is ready, workers
  // are started, in parallel, on nodes 2 and 3
  parallel {
    on "node2" {
      let w1 := startWorker(
          world_size, 1, master.address
        );
      w1.waitForDone();
    },
    on "node3" {
      let w2 := startWorker(
          world_size, 2, master.address
        );
      w2.waitForDone();
    }
  }

  // Let the application run until completion
  master.waitForDone();
}
```

Bakery
Future work

• The goal is **complete full-stack programmable orchestration**.
  • Attaching UIs to the runtime system.
  • Inter-domain networking with NFs.
  • Checking applications against policies.

• A **hub** for sharing Brane packages.
• Investigating a **decentralized** setup.

• Additional and deeper **integrations**.
  • Interactive (rich) JupyterLab notebook widgets.
  • Integration with WfMS, Spark, AsyncAPI, …

---

Currently hiring a scientific programmer to work on Brane. Contact: **Adam Belloum. a.s.z.belloum@uva.nl**
Thank you!

Onno Valkering, Reginald Cushing, Adam Belloum
o.a.b.valkering@uva.nl

Multiscale Networked Systems (MNS)
University of Amsterdam

Currently hiring a scientific programmer to work on Brane. Contact: Adam Belloum. a.s.z.belloum@uva.nl