NATIONAL RESEARCH
SOUTH URAL STATE
UNIVERSITY

Mjolnirr

Providing Integration of UNICORE Services in Private PaaS Platform

Gleb Radchenko, Dmitry Savchenko
gleb.radchenko@susu.ru
South Ural State University, Russia

UNICORE Summit, Leipzig, 24.06.2014

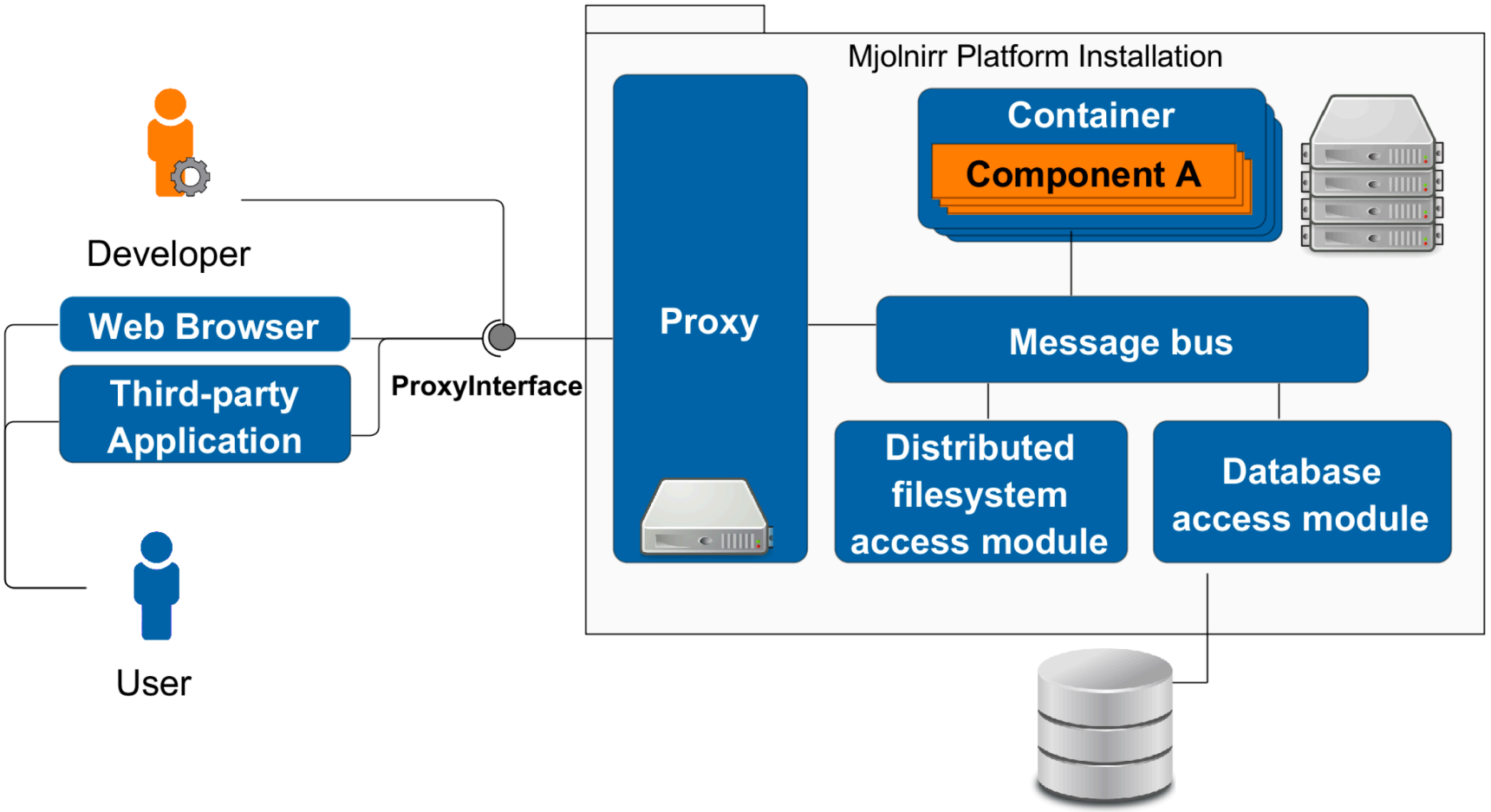
Problem definition

- Cloud computing enables resource providers to reduce support and integration costs, using elastic resource management
- But public cloud platforms raise a security concern: data is stored and processed remotely
- Private clouds are the only option for the company that want to provide computing resources inside the company
  - But most of existing private cloud solutions provide IaaS level of clouds that often require complicated procedures for support and usage of resources

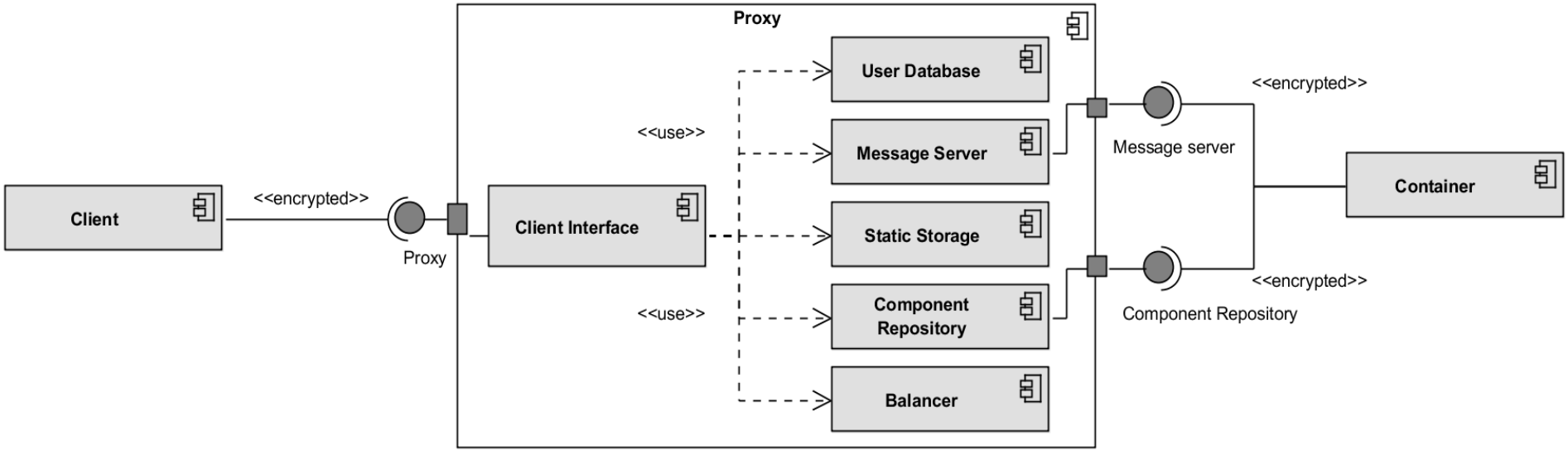
Mjolnirr platform

- Mjolnirr platform – solution for Java-based private PaaS systems deployment:
  - Provide an API to enable programmers to write new modules easily
  - Supports component-oriented loose-coupled system architecture
  - Provides automation of components distribution and deployment
  - Component containers can work not only on server hardware, but on end-user PCs
  - Provides integration with the UNICORE grid services

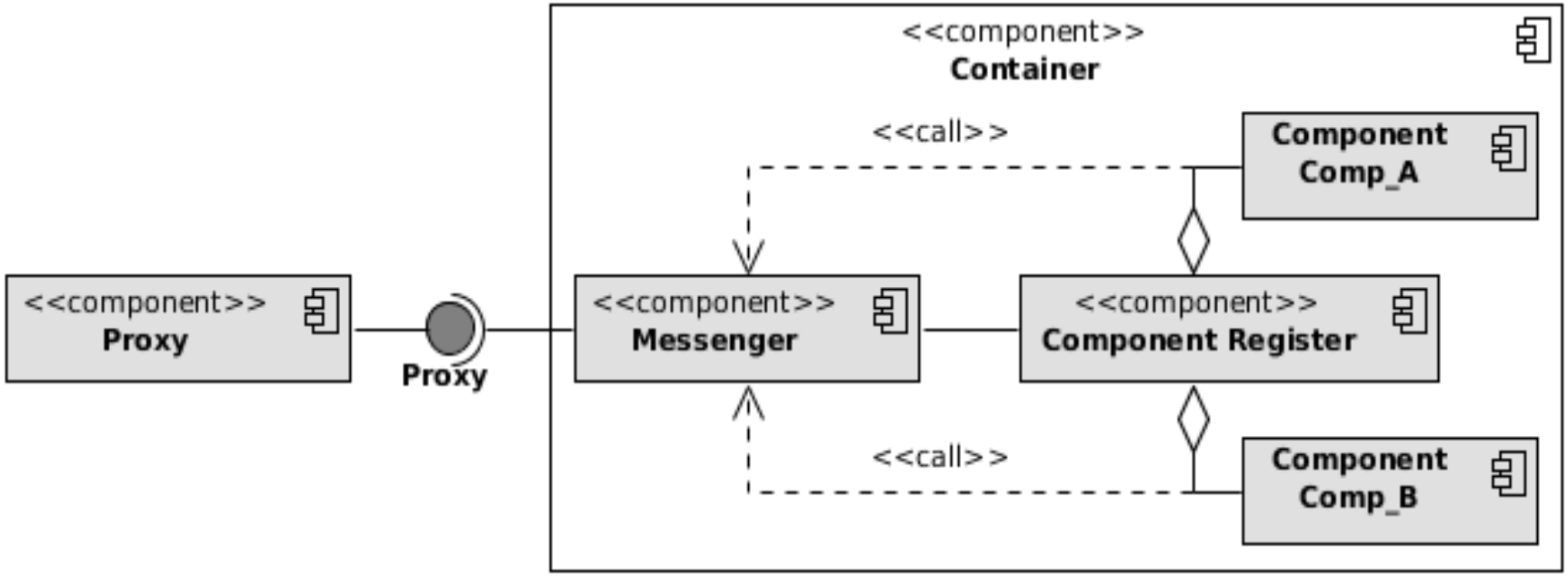
Mjolnirr platform Architecture



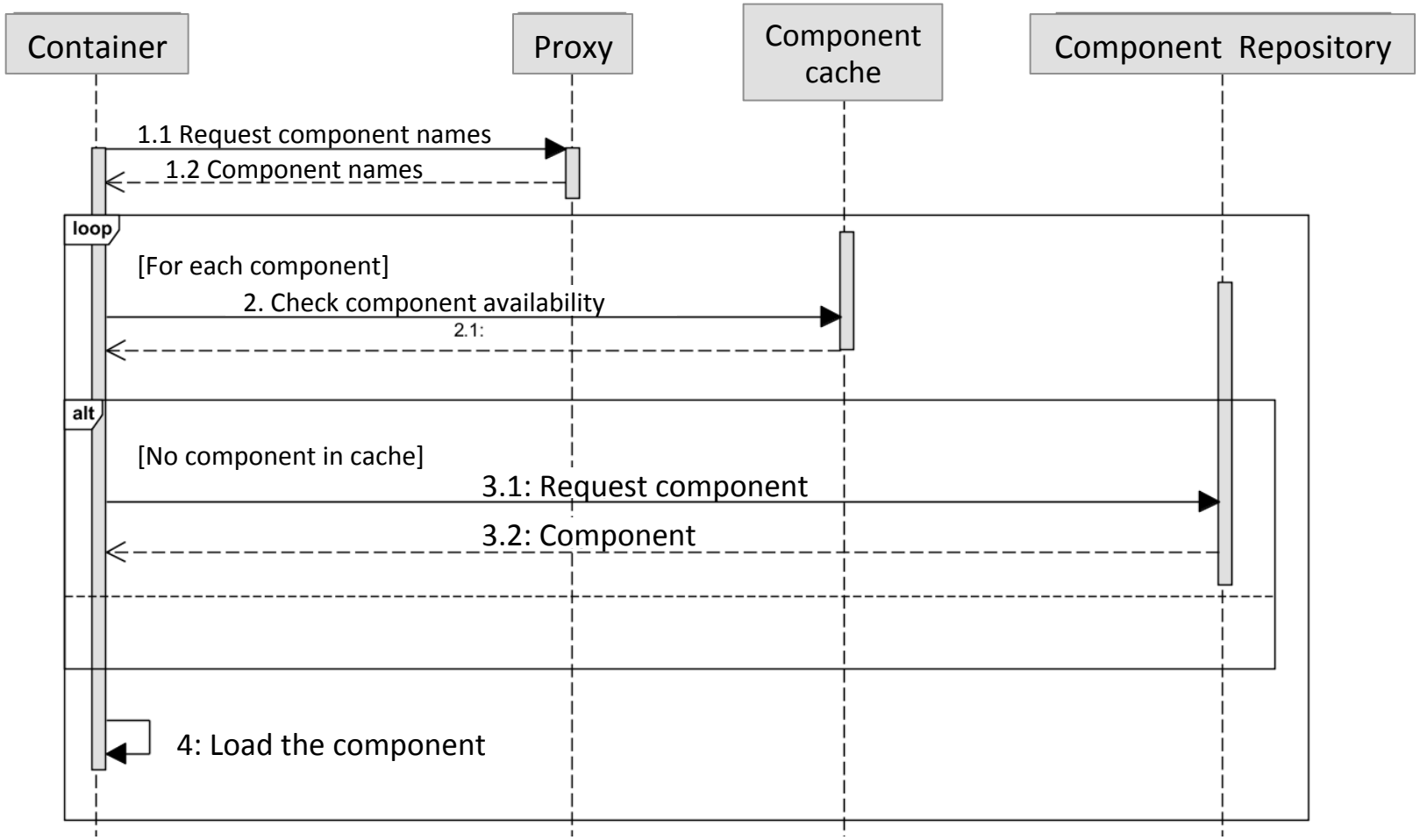
Architecture: Proxy



Architecture: Container



Container deployment



Development: Components

- Two types of custom components:
  - *Application component* provides **user interface**, **scripts and styles** as static files, as well as processing logic.
  - *Module component* represents **a single entity** in the application domain.

- Developer:
  - Creates a components on the basis of provided API
  - Uploads the component to a Proxy, using the web-interface
- The component instances are deployed on containers automatically

Development: Component interface

```
@MjolnirrComponent(
        componentName = "calculator",
        instancesMaxCount = 1,
        memoryVolume = 128,
        interfaces = {
                @MjolnirrInterface(pageNameWildCard = "main", allowedUsers = { "privileged" } )
        }
)
public class Calculator extends AbstractApplication {
    private ComponentContext context;

    @MjolnirrMethod(maximumExecutionTime = 30)
    public String calculate(String expression) throws Exception {
        if (expression.length() > 0) {
            // check syntax, evaluate and display results if correct
            if (CalculatorHelper.checkSyntax(expression)) {
                return String.valueOf(CalculatorHelper.evaluate(expression));
            }
        }

        throw new Exception("Expression missing!");
    }

    @Override
    public void initialize(ComponentContext context) {
        this.context = context;
    }
```

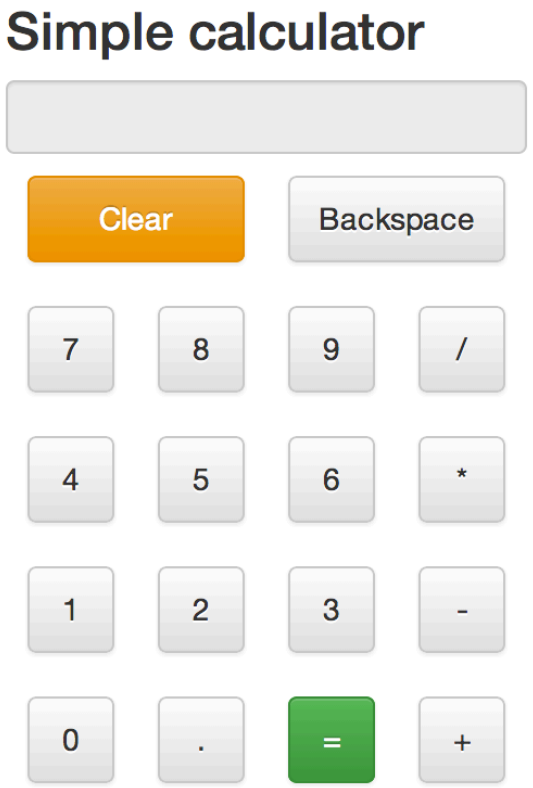
Component interface

Method interface

Component Initialization

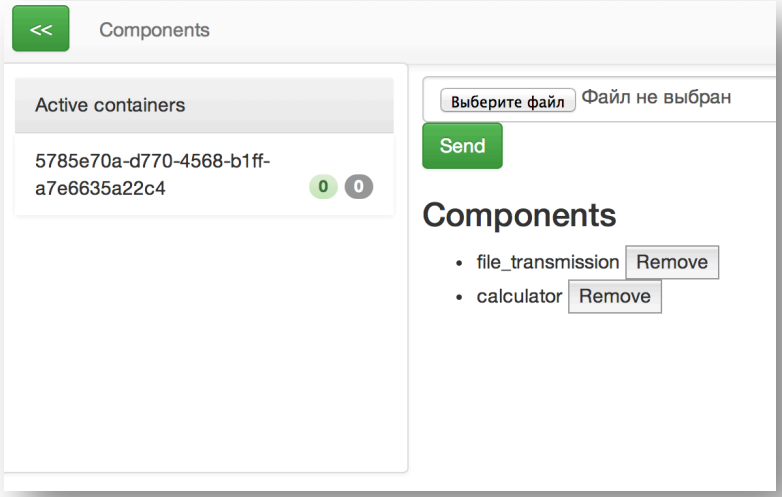
Development: Component UI

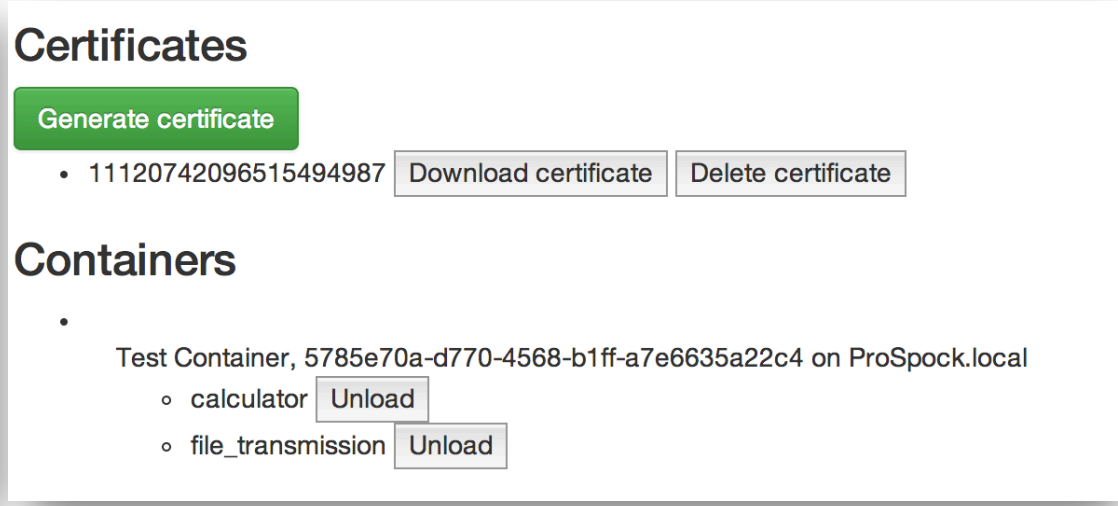
- You can use jade as web-template engine and JavaScript to develop interactive UI

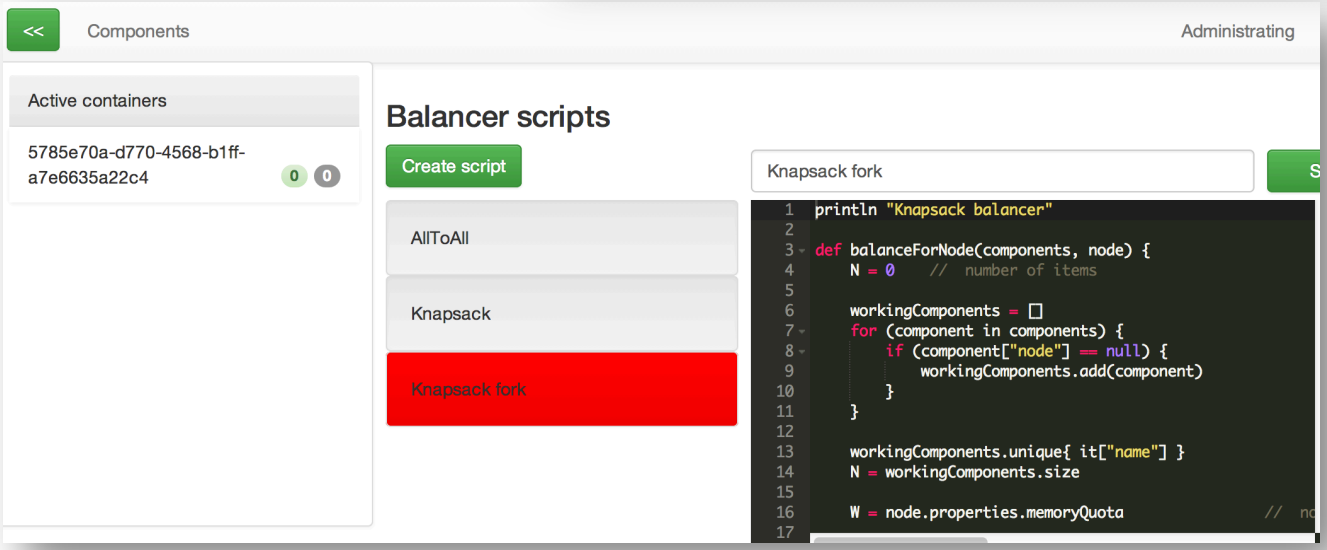



```
function calc() {
    var inputField = $("#calculator-string");
    try {
         inputField.val(callRemoteMethodSync({
         method: "calculate"
         , args: [ inputField.val() ]}));
        } catch (err) {
         bootbox.alert(err);
 }}
```

Administrative UI

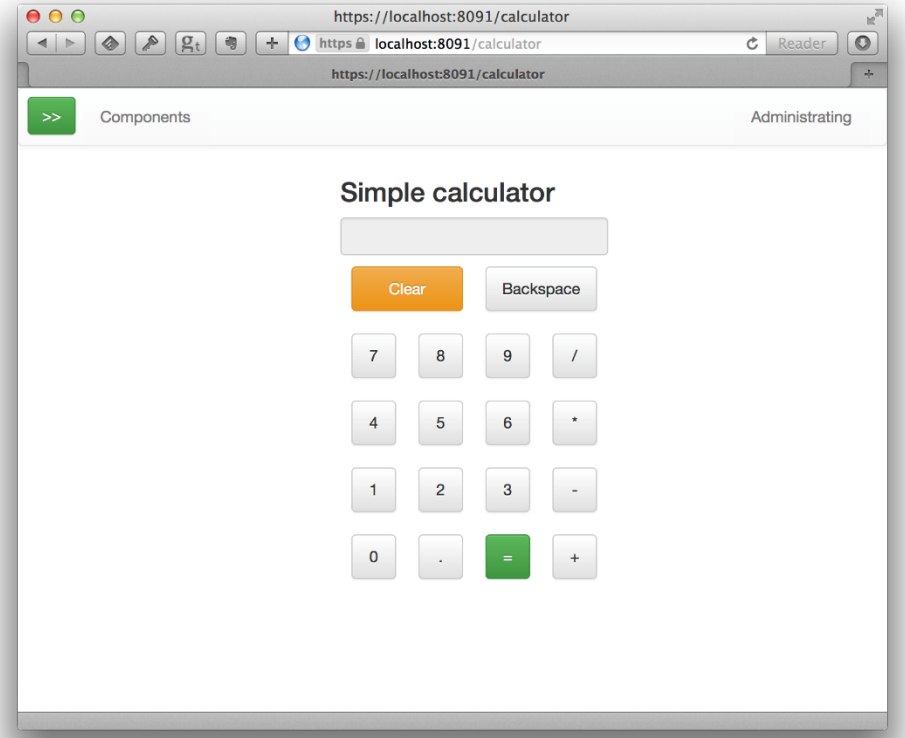


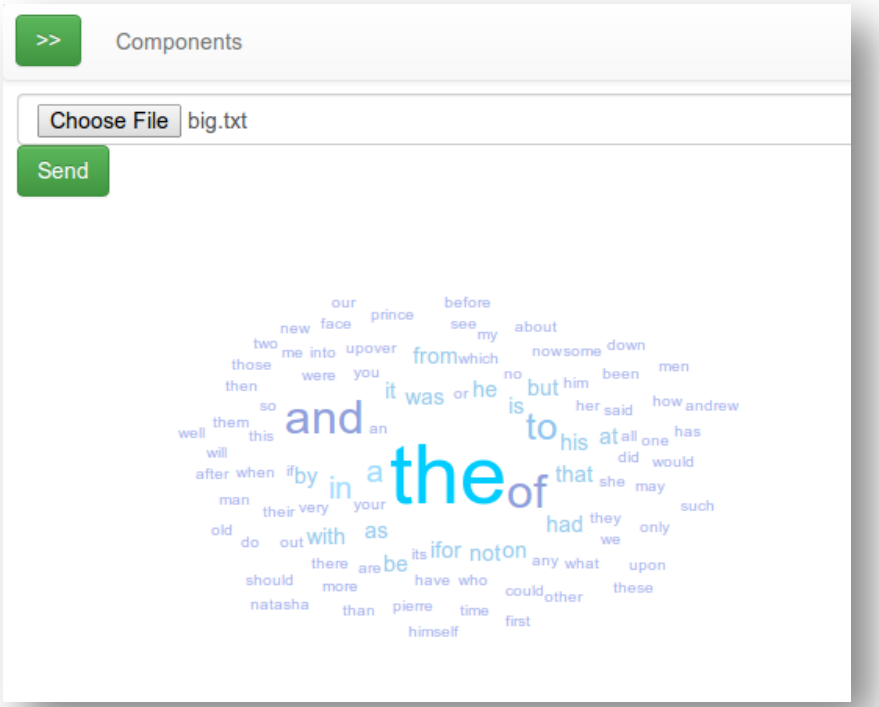




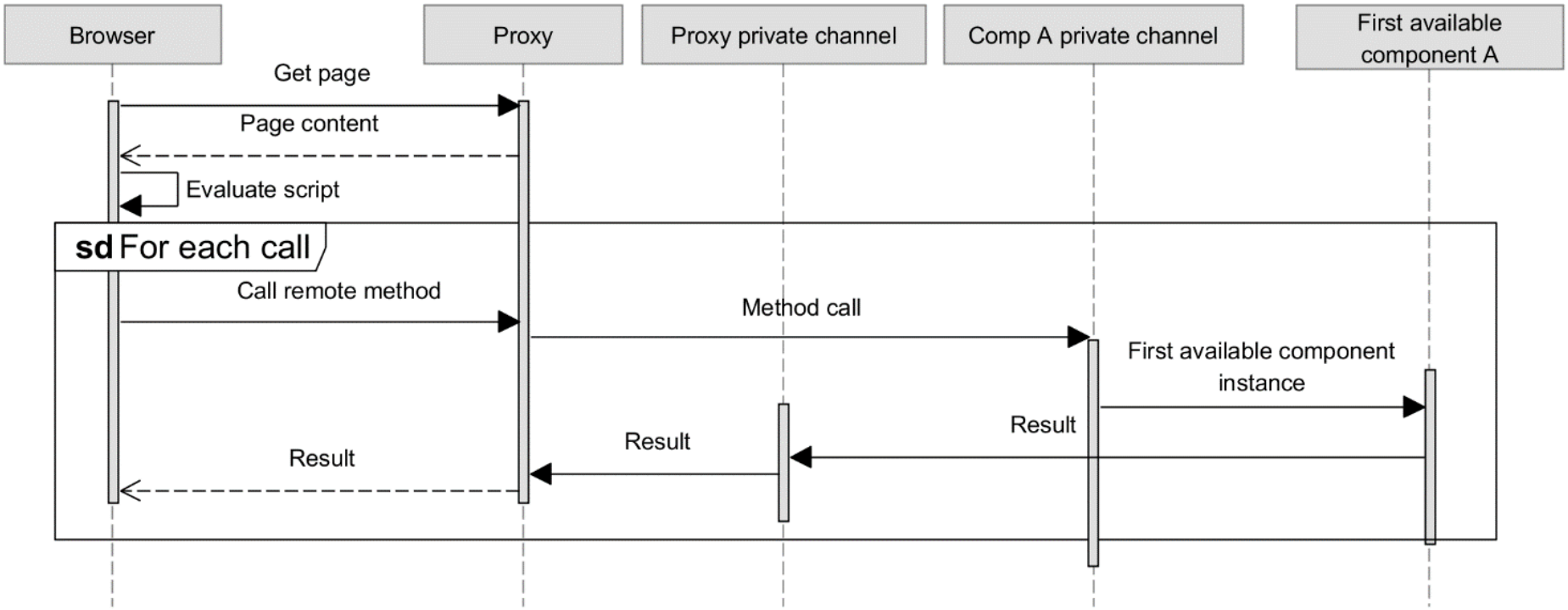
Applications

- Any application is available using the following address: https://HOSTNAME/APP_NAME/PAGE_NAME
  - HOSTNAME – Proxy host name
  - APP_NAME – Application name
  - PAGE_NAME – Name of the page of the application





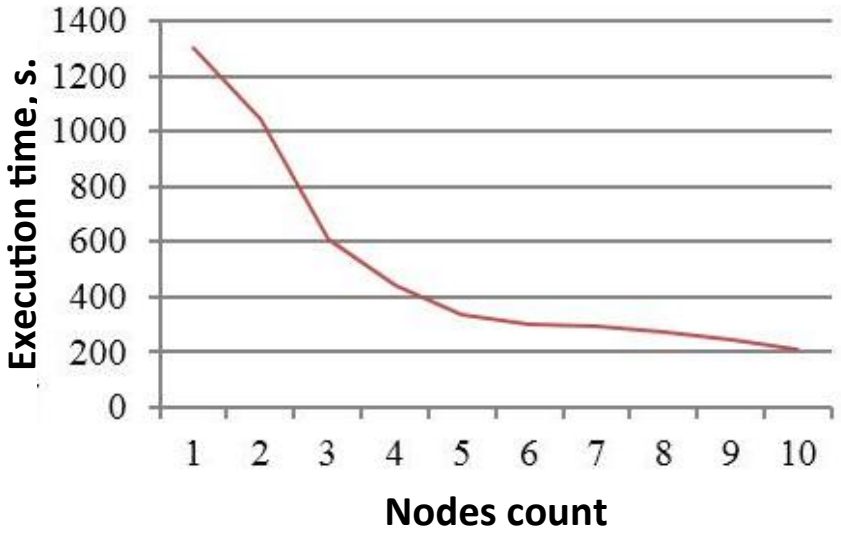
Application execution

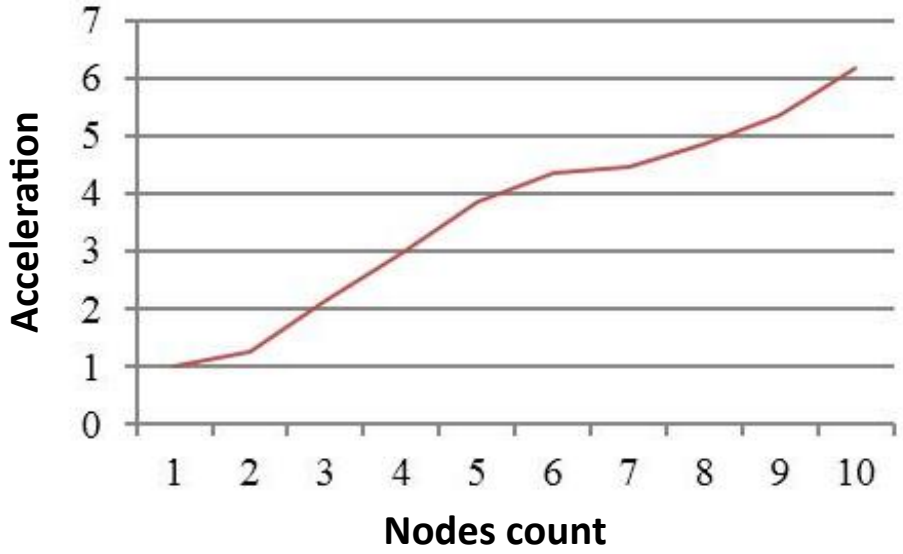


Performance evaluation

- 1 gigabyte of text data was divided on 100 parts and sent to all available worker components for processing.
- Each worker divide text on words and count a frequency of each unique word. Pieces of work were distributed automatically – each worker polled Message Bus to receive new task.

Performance evaluation

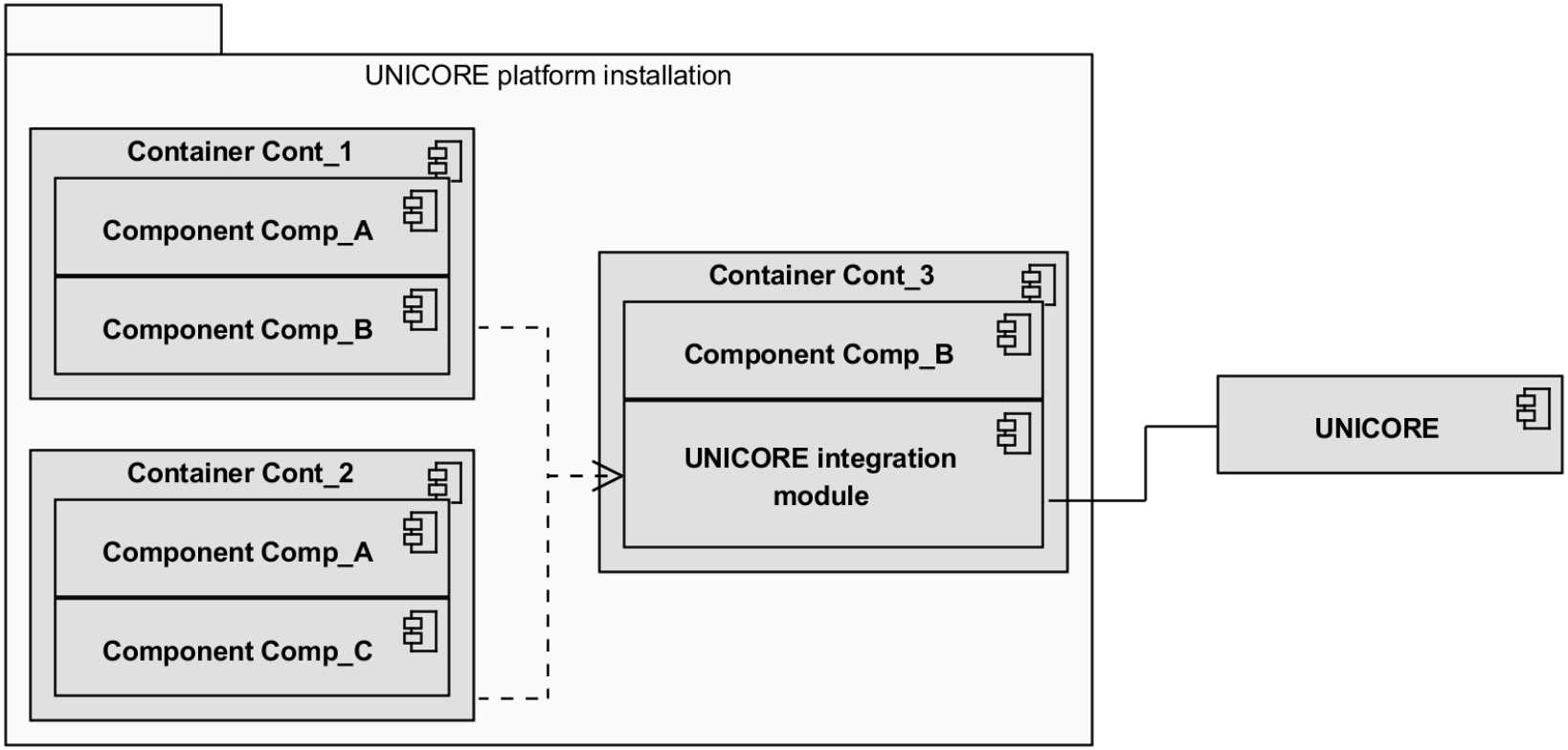




Experiments have shown that the platform is stable. Average execution time on **10** containers was **208** seconds. Thus, acceleration of parallel word frequency counter task was **6.3**.

UNICORE integration

- Communication with UNICORE based on module, which uses standard Mjolnirr client API



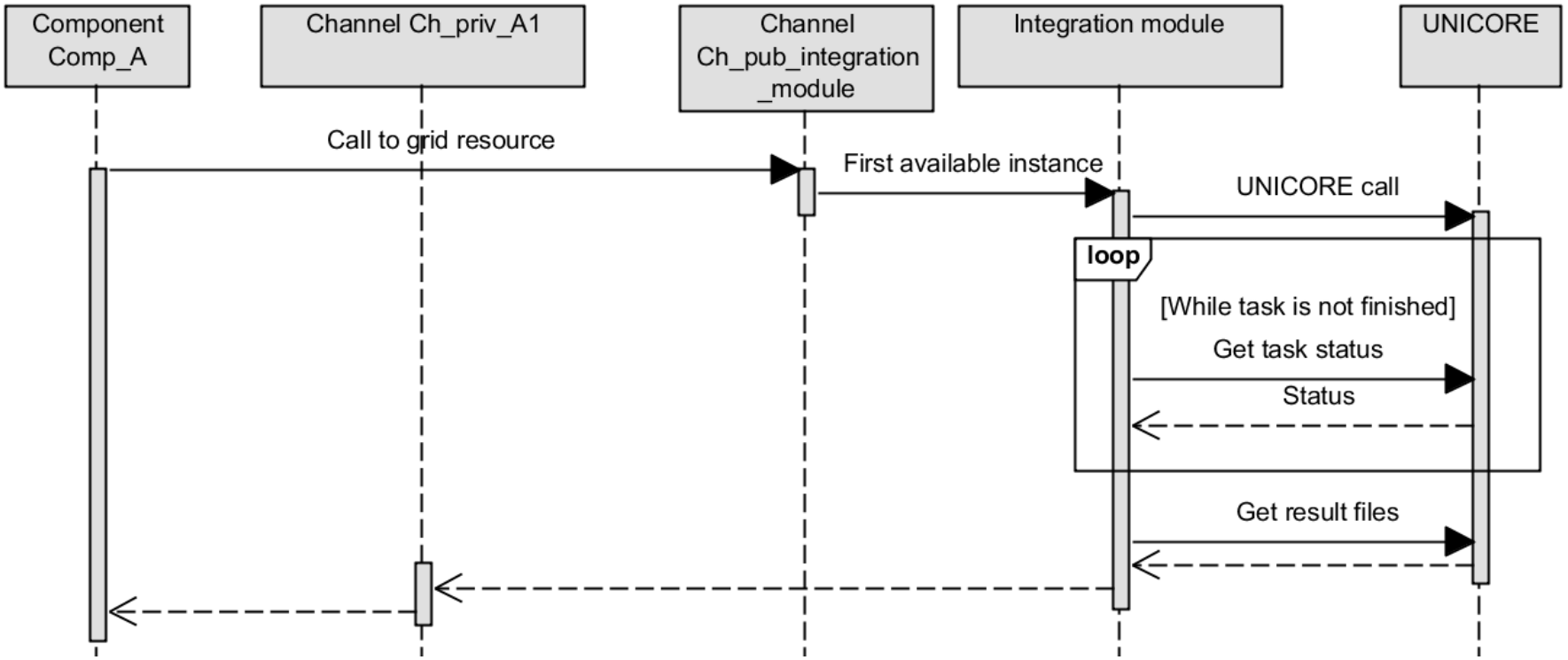
UNICORE call

- UNICORE module call allows to submit the custom task into the grid environment. This module returns all the required result files.

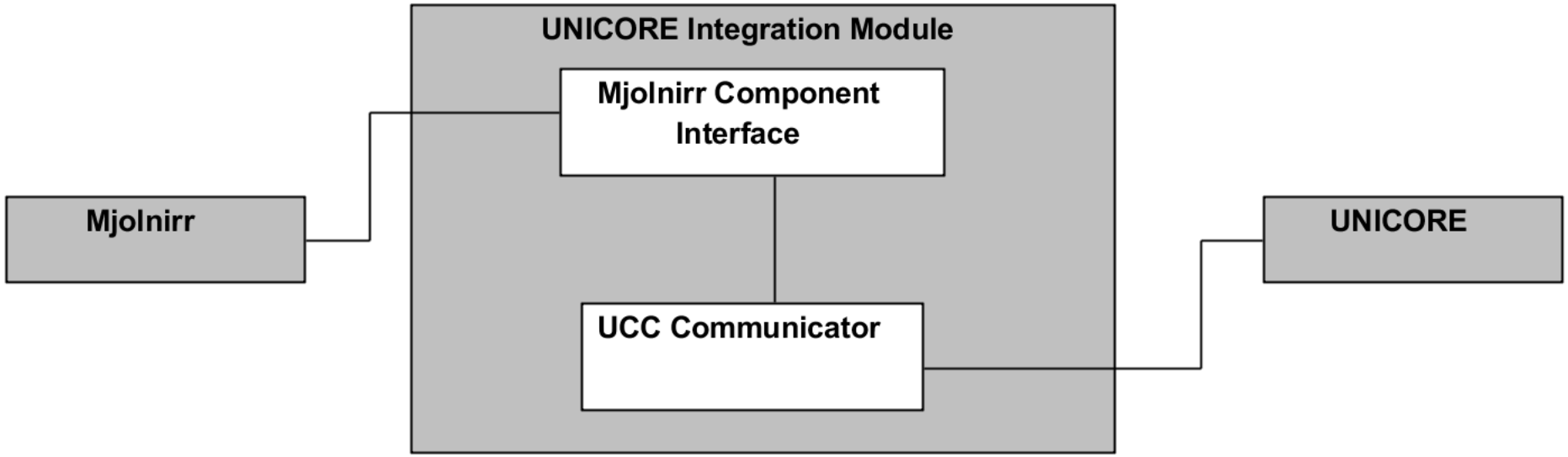
```
testFiles = new HornetCommunicator().sendSync(context,
    "unicore", "run", new ArrayList<Object>() {{
        Map<String, String> params = new HashMap<String, String>();
        List<String> inputs = new ArrayList<String>();
        List<String> outputs = new ArrayList<String>();
        outputs.add("*");

        add("Date");
        add("1.0");
        add(params);
        add(inputs);
       add(outputs);
}}, List.class);
```

UNICORE call



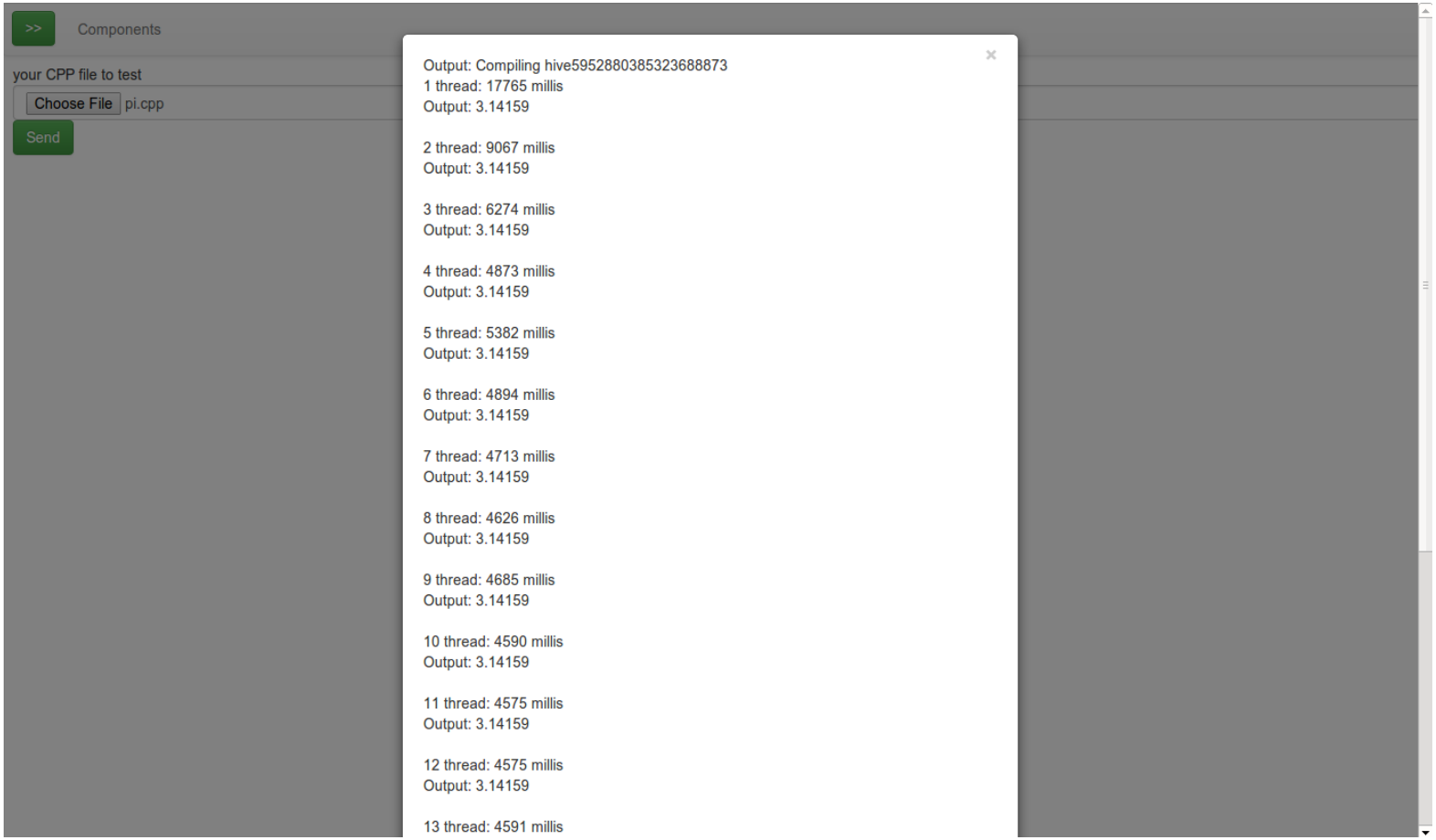
UNICORE Integration module structure



UNICORE: OpenMP test stand example

- UNICORE OpenMP test stand works as follows:
  - Client uploads C or CPP source into the cloud
  - Test stand application sends this source into UNICORE and invokes OpenMP application in the UNICORE installation
  - OpenMP application is implemented as the Python script which compiles the source and executes the binary on N threads for N=1..16

OpenMP test stand example



Results

- We developed an architecture and implementation of the Mjolnirr platform
- The tests shown, that the system is stable, provides effective loose coupling components development
- We developed a module for integration with the UNICORE grid system and provided a test application for it.

- As a development of this project, we are planning to provide:
  - Application-level migration support to provide system stability;
  - Resource monitoring for flexible load balancing;
  - Global component store to reduce the number of the duplicate applications;
  - Integration modules for DBMS and distributed file-management systems.

- All sources are available on BitBucket:
  - https://bitbucket.org/mjolnirr/mjolnirr/src

- Contact: gleb.radchenko@susu.ru