

ТЕХНОЛОГИИ В ОБРАЗОВАНИИ УНИВЕРСИТЕТ

МИКРОЭЛЕКТРОНИКА
ИННОВАЦИИ
КАТАЛИТИЧЕСКИЕ
МАТЕРИАЛЫ
ДИЗАЙН
ЛЕКАРСТВ
НАУЧНАЯ
ЛАБОРАТОРИЯ
ГЕОХИМИЯ
ИНЖИНИРИНГ
ГЕОФИЗИКА

ГИБРИДНЫЕ
МАТЕРИАЛЫ
ЭНЕРГОСБЕРЕЖЕНИЕ

ВЫСОКИЕ
ЭНЕРГИИ
БИОТЕХНОЛОГИИ
МОДЕЛИРОВАНИЕ
НАНОТЕХНОЛОГИИ
СЕМІОТИКА

НАУКА МОЗГА АРКТИКА
КОГНИТИВНЫЕ ТЕХНОЛОГИИ
МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

N* Новосибирский
государственный
университет
*НАСТОЯЩАЯ НАУКА

ЭЛЕМЕНТАРНЫЕ
ЧАСТИЦЫ
ГЕОЛОГИЯ

КВАНТОВЫЕ
ТЕХНОЛОГИИ
БИОЛОГИЯ

ТЕМНАЯ
МАТЕРИЯ
ФОТОНИКА
БИОМЕДИЦИНА
ПРИКЛАДНЫЕ
ИССЛЕДОВАНИЯ
РАЗВИТИЕ

АСТРОНОМИЯ
ГЛОБАЛЬНЫЕ ПРИОРИТЕТЫ

АСТРОФИЗИКА
БИОИНФОРМАТИКА
**ЛАЗЕРНАЯ
ФИЗИКА**
АРХЕОЛОГИЯ
ЭКОНОМИКА
ЗНАНИЙ
СОТРУДНИЧЕСТВО

IT
DEEP
LEARNING
ИЗУЧЕНИЕ

МИНОБРНАУКИ РОССИИ ФЕДЕРАЛЬНОЕ
ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»



09.03.01 Информатика и вычислительная техника.
Программная инженерия и компьютерные науки

Разработка прототипа облачной IDE для системы LuNA

Выполнил: Александров Евгений Александрович
студент ФИТ НГУ

Руководитель ВКР: Малышкин Виктор Эммануилович
зав. каф. ПВ ФИТ д.т.н., профессор

Соруководитель ВКР: Перепёлкин Владислав Александрович
ст. преп. каф. ПВ ФИТ НГУ

Рецензент: Черкасов Александр Владимирович
руководитель проектов, ООО «Скайтек»

Новосибирск, 2021

* Актуальность проблемы

Проблемы установки необходимого ПО, хранения файлов, обновления софта актуальны при разработке программ в наше время. Облачная IDE решает эти проблемы.

Создание облачной IDE - это комплексная проблема, включающая в себя разработку архитектуры облачной IDE, требований, которые удовлетворяют потребностям пользователей, а также удобного, интуитивно понятного для пользователя интерфейса.



* LuNA

LuNA (*Language for Numerical Algorithms*) - система фрагментированного программирования, предназначенная для поддержки параллельной реализации больших численных моделей для суперкомпьютеров. LuNA нуждается в такой облачной IDE.

В группу пользователей суперкомпьютеров входят специалисты в своих предметных областях, которые используют численное моделирование для решения поставленных перед ними задач.

Для специалистов нужен продвинутый функционал: запуск на кластерах, использование системы очередей, хранение файлов, отладчик, взаимодействие с системой визуализации фрагментированной программы.

* Обзор существующих IDE (1 из 2)

The screenshot shows the Python.org homepage. At the top, there's a dark blue header with tabs for Python, PSF, Docs, PyPI, Jobs, and Community. Below the header is a search bar with a magnifying glass icon and a 'GO' button. To the left of the search bar is a yellow 'Donate' button. The main content area features the Python logo and the word 'python™'. Below the logo is a navigation bar with links for About, Downloads, Documentation, Community, Success Stories, News, and Events. A large central section contains a code snippet demonstrating Python 3 syntax:

```
# Python 3: Simple output (with Unicode)
>>> print("Hello, I'm Python!")
Hello, I'm Python!

# Input, assignment
>>> name = input('What is your name?\n')
>>> print('Hi, %s.' % name)
What is your name?
Python
Hi, Python.
```

To the right of the code snippet is a yellow button with a right-pointing arrow. Below the code, a section titled "Quick & Easy to Learn" states: "Experienced programmers in any other language can pick up Python very quickly, and beginners find the clean syntax and indentation structure easy to learn. [Whet your appetite](#) with our Python 3 overview." At the bottom of this section are five numbered buttons (1, 2, 3, 4, 5). At the very bottom of the page, a footer message reads: "Python is a programming language that lets you work quickly and integrate systems more effectively. [» Learn More](#)".

python.org – официальный сайт, где можно познакомиться с Python. Сразу бросается в глаза самый большой элемент на сайте – командная строка с пятью примерами, с помощью которых, можно познакомиться с синтаксисом, операциями, работой со списками, определением функций.

* Обзор существующих IDE (2 из 2)

The Go Playground

Run

Format

Imports

Share

Hello, playground ▾

About

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 func main() {
8     fmt.Println("Hello, playground")
9 }
10
11
12
13
```

- Hello, playground
- Tests
- Multiple files
- Display image
- Sleep
- [Clear](#)

play.golang.org – сайт, где можно познакомиться с возможностями этого языка и запустить предлагаемые примеры. Интерфейс гораздо проще чем у python.org, снизу текстовой поле, куда нужно вписывать код, а на панели сверху – несколько управляющих кнопок.

N*

* Выводы по существующим IDE

1. Наличие поля ввода исходного кода
2. Возможность просмотра тестовых примеров
3. Подсветка синтаксиса
4. Возможность перехода к документации
5. Запуск программы и возможность видеть результат её исполнения

* Цели и задачи

Цель работы - разработать облачную IDE для системы фрагментированного программирования LuNA.

Задачи

1. Разработать требования к IDE.
2. Разработать функциональность.
3. Разработать архитектуру IDE.
4. Реализовать front-end часть IDE.
5. Разработать вычислительный модуль.
6. Провести тестирование.

* Устройство работы LuNA-программы

На вход программе подается набор входных файлов одного из двух типов: **файл с расширением ".fa"** (может быть несколько), в нем описывается LuNA программа, и **файл с расширением ".cpp"**, который реализует последовательные процедуры, осуществляющие вычисление выходных из входных данных.

Результат компиляции – это файл libcodes.so - исполняемое представление, скомпилированное обычным компилятором.

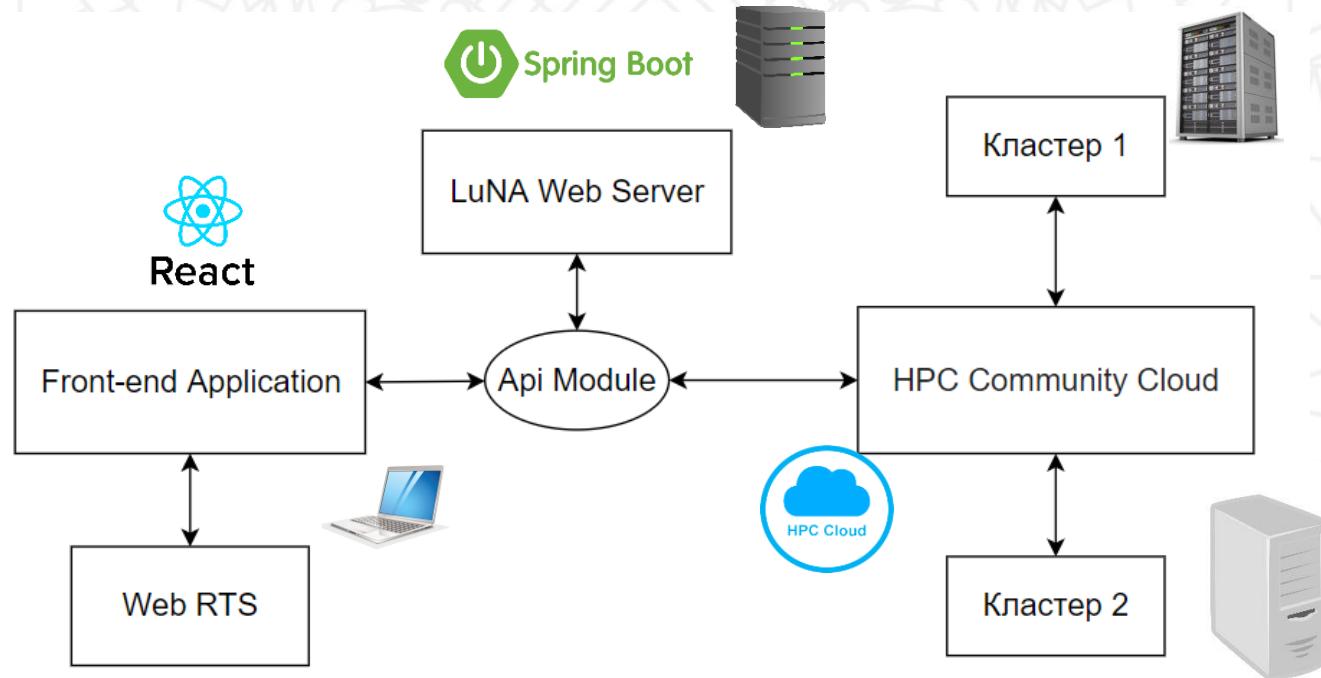
* Требования к облачной IDE

Специфичные требования:

1. Запуск на различных вычислительных системах.
2. Требования к количеству одновременных подключений к IDE.
3. Просмотр и запуск тестовых примеров.
4. Интеграция существующего инструментария
5. Отображение результатов производительности.
6. Возможность расширения.



* Архитектура решения



Реализация Front-end Application

LuNA Cloud IDE

localhost:8000

LuNA Cloud IDE

LuNA Home Documentation Examples ▾ LuNA Web Server ▾ Run

test.fa

```
1 /*  
2 Initialization of data fragments and data dependencies.  
3 */  
4  
5 import c_init(int, name) as init;  
6 import c_iprint(int) as iprint;  
7  
8 sub main() {  
9     df x;  
10  
11     cf a: init(7, x) @ {  
12         // req_count x=1;  
13         locator_cyclic: 1;  
14     };  
15  
16     iprint(x) @ {  
17         request x;  
18         locator_cyclic: 2;  
19     };  
20  
21 }  
22 @ {  
23     locator_cyclic x => 3;  
24 }
```

ucodes.cpp

```
1 #include <cstdio>  
2  
3 #include "ucenv.h"  
4  
5 extern "C" {  
6  
7     void c_init(int val, OutputDF & df) {  
8         df.setValue(val);  
9         printf("c_init: %d, size: %d\n", val, (int) df.getSize());  
10    }  
11  
12    void c_iprint(int val) {  
13        printf("c_iprint %d\n", val);  
14    }  
15  
16    void c_print(const InputDF & df) {  
17        printf("c_print: %d\n", df.getValue < int >());  
18    }  
19  
20 }
```

Output

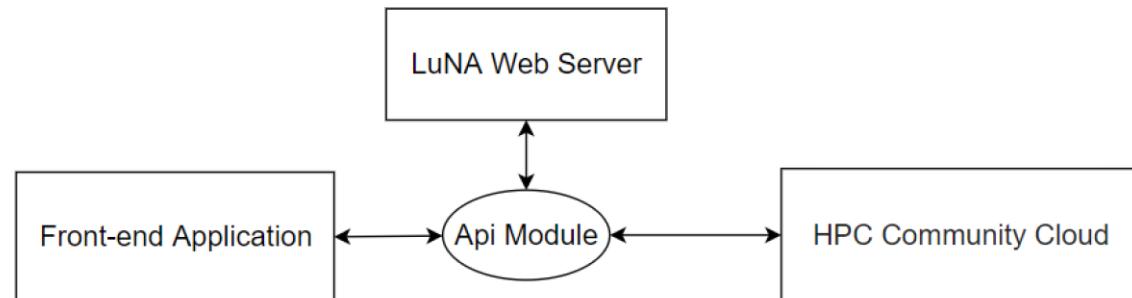
1

Table

Clear

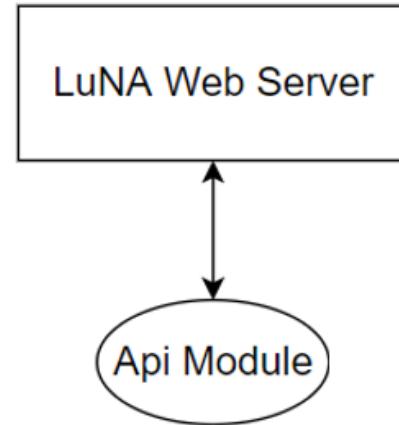
* Api Module

Api Module - модуль через который облачная IDE взаимодействует с вычислительными системами, например, LuNA Web Server. Api Module является точкой расширения, для интеграции других вычислительных систем.



* Реализация LuNA Web Server

LuNA Web Server – вычислительная система с предустановленной LuNA. Реализована на Java с использованием фреймворка Spring. Spring – это фреймворк состоящий из множества модулей для различных задач. Мы используем Spring Boot – подсистема для построения веб-приложения.



test.fa

```

1 /*
2 Initialization of data fragments and data dependencies.
3 */
4
5 import c_init(int, name) as init;
6 import c_iprint(int) as iprint;
7
8 sub main() {
9     df x;
10
11    cf a: init(7, x) @ {
12        // req_count x=1;
13        locator_cyclic: 1;
14    };
15
16    iprint(x) @ {
17        request x;
18        locator_cyclic: 2;
19    };
20
21 }
22 @ {
23     locator_cyclic x => 3;
24 }
```

Output

```

1 luna: creating build dir
2 luna: preprocessing
3 luna: parsing
```

ucodes.cpp

```

1 #include <cstdio>
2
3 #include "ucenv.h"
4
5 extern "C" {
6
7     void c_init(int val, OutputDF & df) {
8         df.setValue(val);
9         printf("c_init: %d, size: %d\n", val, (int) df.getSize());
10    }
11
12    void c_iprint(int val) {
13        printf("c_iprint %d\n", val);
14    }
15
16    void c_print(const InputDF & df) {
17        printf("c_print: %d\n", df.getValue < int > ());
18    }
19
20 }
```

Table

Clear

* Анализ предложенного решения

Облачная IDE для системы LuNA гибкая в выборе вычислительной системы, за счет компонента архитектуры API Module, что делает IDE слабосвязанной от конкретной вычислительной системы и является сильной стороной.

Дизайн облачной IDE в дальнейшем требует доработки, добавления анимации во время выполнения действий.

* Заключение

В результате была разработана архитектура и реализован прототип облачной IDE для системы фрагментированного программирования LuNA.

На защиту выносятся:

1. Требования к облачной IDE для системы LuNA
2. Архитектура облачной IDE
3. Реализация модулей Front-end Application, Api Module, LuNA Web Server

Дальнейшие планы:

1. Интеграция с HPC Community Cloud
2. Реализация задания аргументов программы
3. Интеграция с Web RTS

* Апробация работы

Данная работа была опубликована на 59-й Международной научно-студенческой конференции, г. Новосибирск, 2021 г.

Публикация по теме:

Александров Е. А. Разработка прототипа облачной IDE для системы фрагментированного программирования LuNA / Е. А. Александров // Информационные технологии : Материалы 59-й Междунар. науч. студ. конф. 12–23 апреля 2021 г. / Новосиб. гос. ун-т. – Новосибирск : ИПЦ НГУ, 2021. – С. 110.