

Государственное бюджетное образовательное учреждение  
высшего образования Московской области  
Университет «Дубна»

# Сервер устройства Анализатора сигналов *Rhode&Schwarz FSV-7* в стандарте

---

## **TANGO**

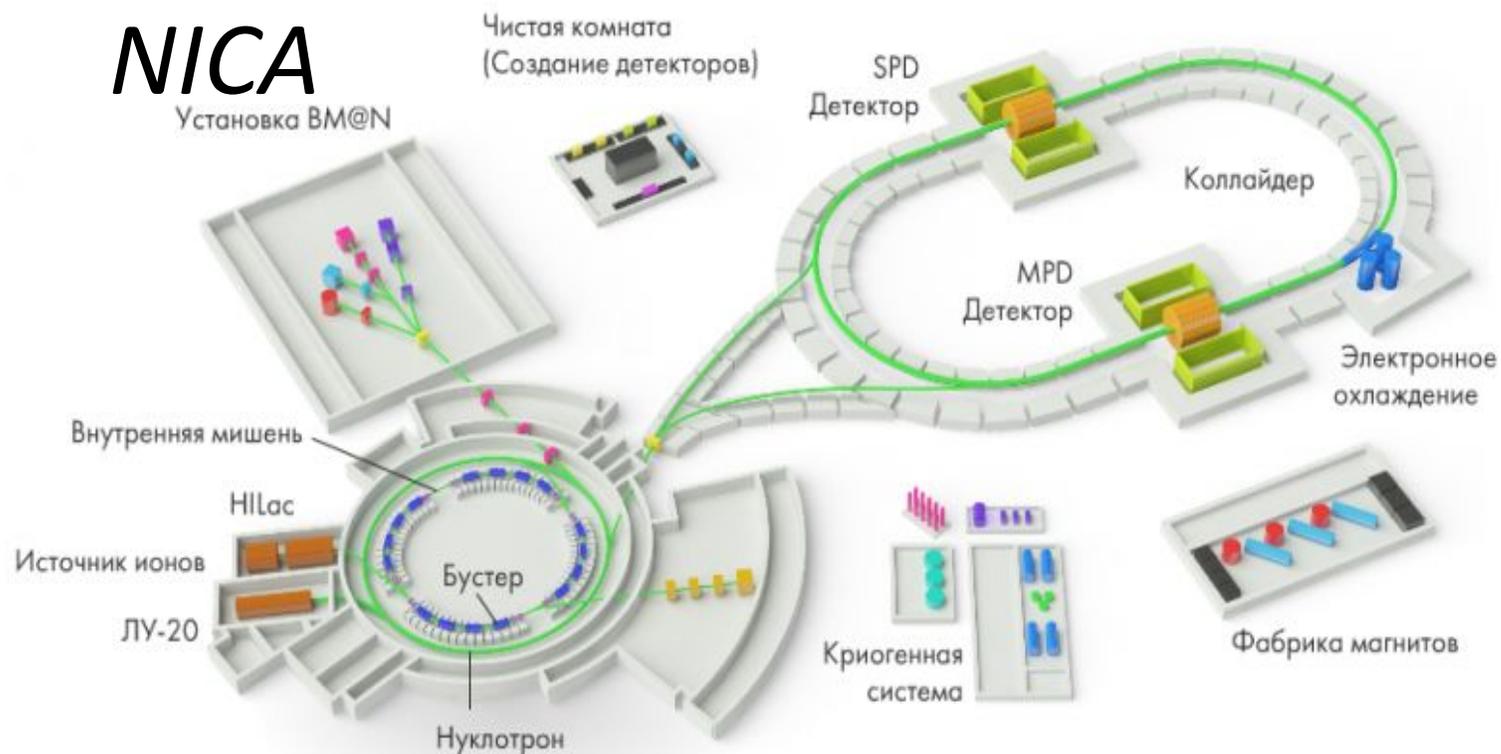
Автор ВКР: Лукьянов М.А.

Руководитель: ст. преп. Андреев В.

А.

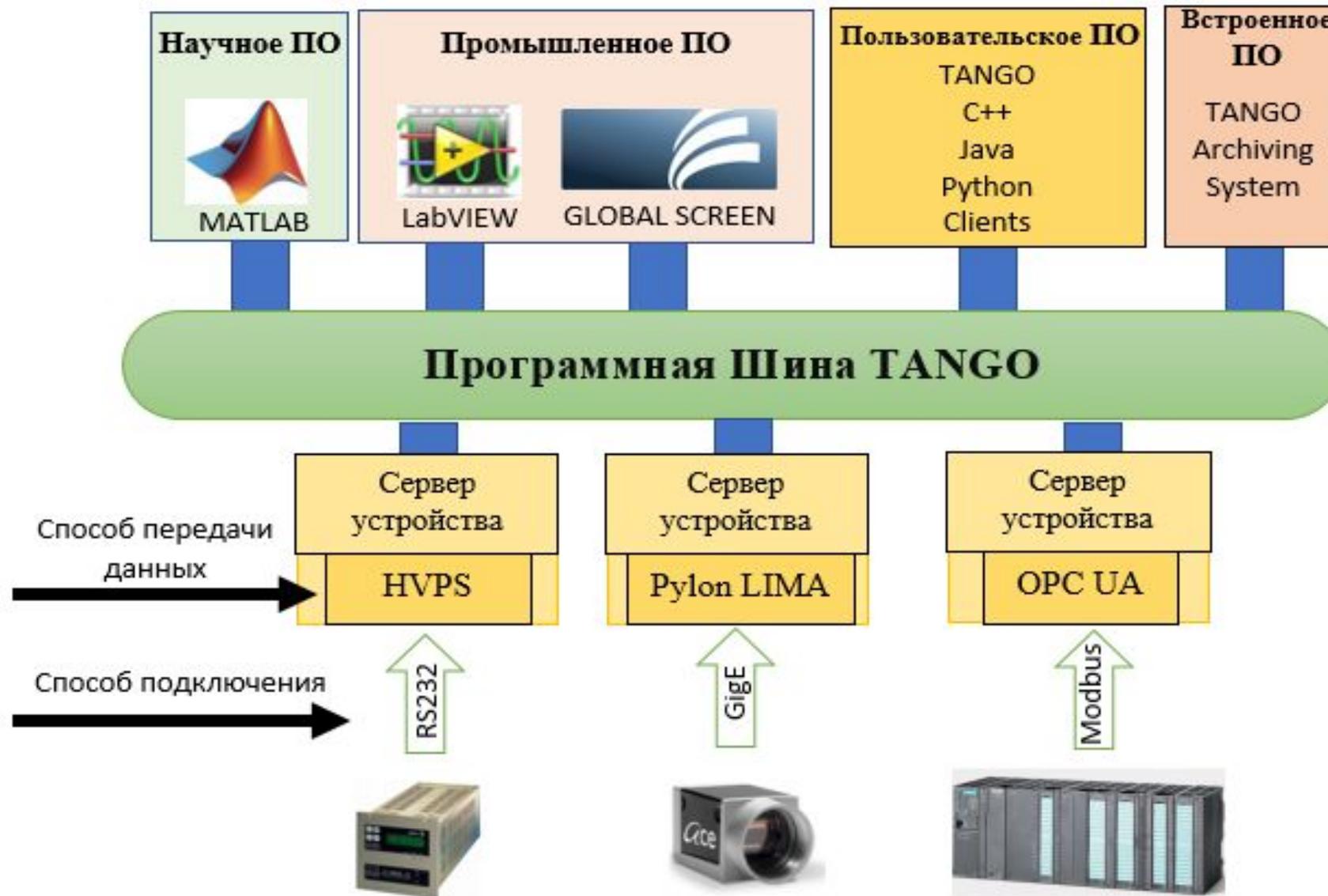
# Ускорительный комплекс

## *NICA*

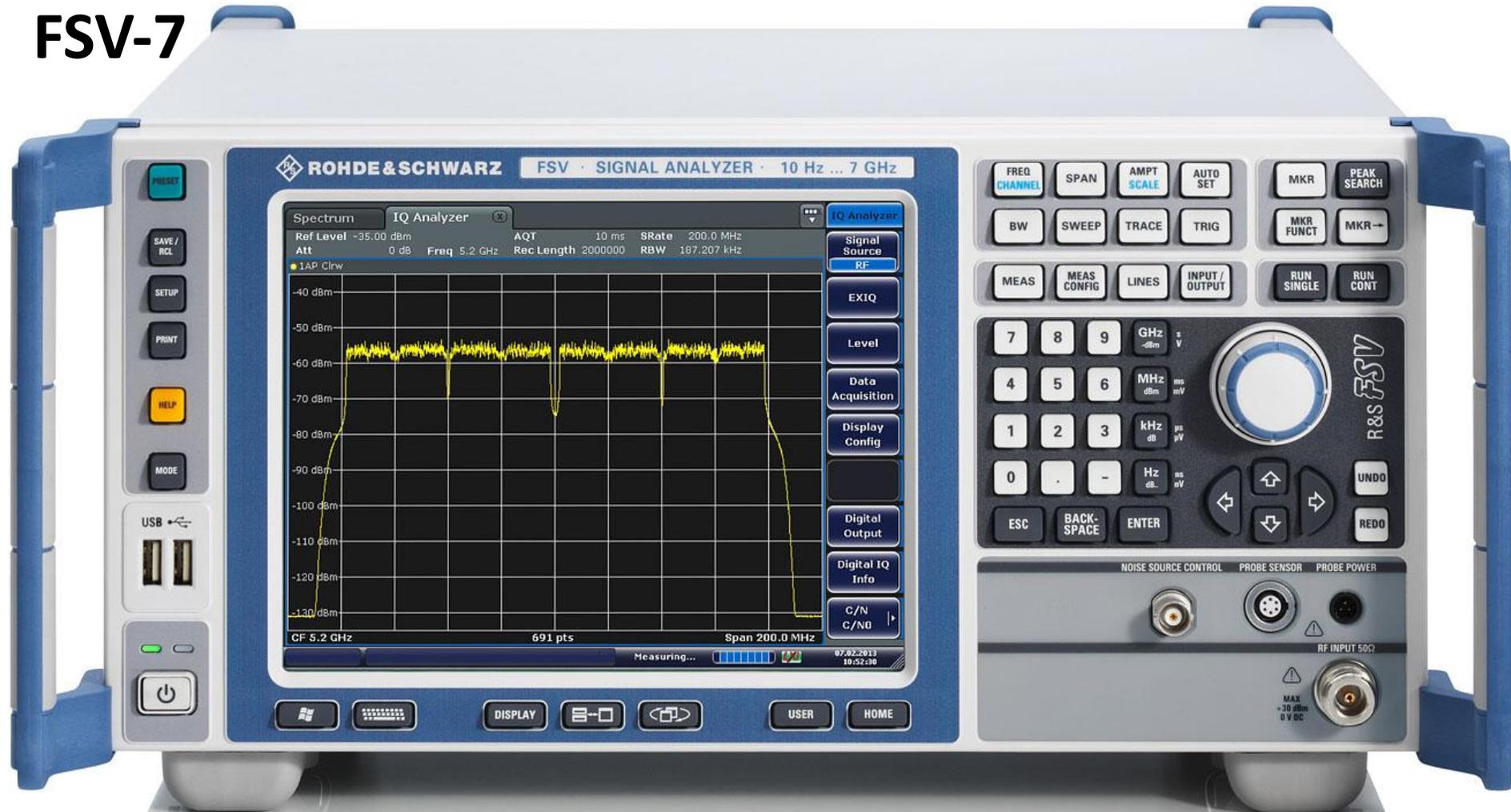


Проект класса «мега-сайенс», реализуется на базе Объединенного института ядерных исследований (ОИЯИ). В комплексе *NICA* задействовано множество измерительных и управляющих устройств различного предназначения и функционала. Каждое из этих устройств может иметь свой протокол подключения, характер работы, формат вычисляемых значений и т.д.

# Объектно-ориентированная распределенная система Tango Controls

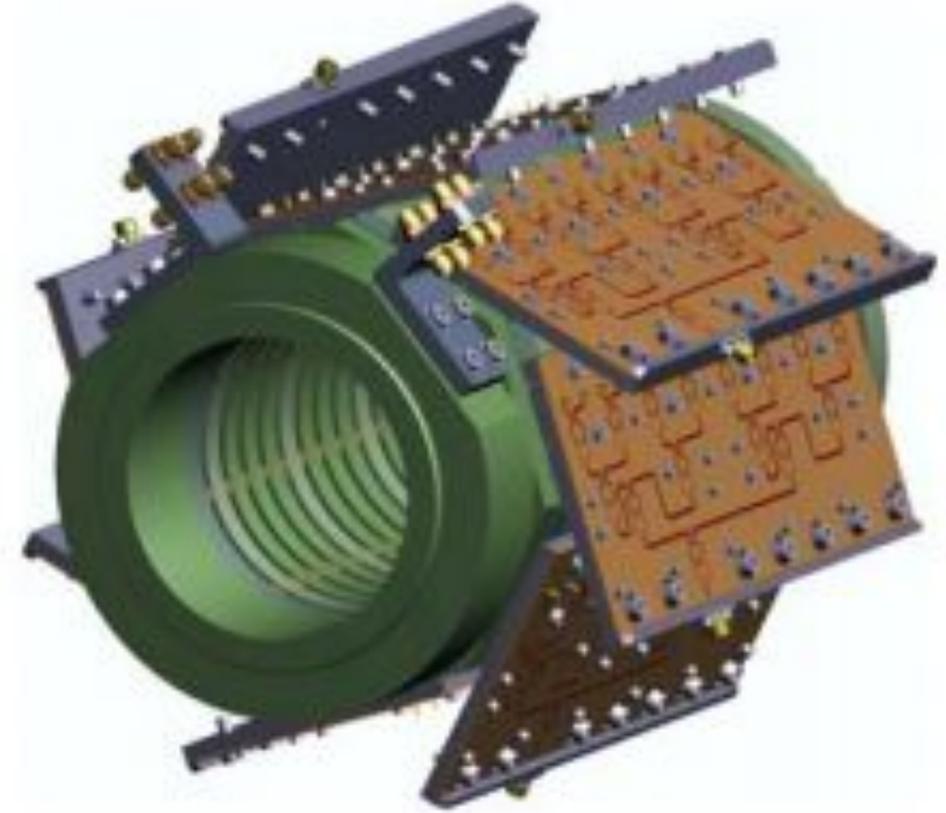
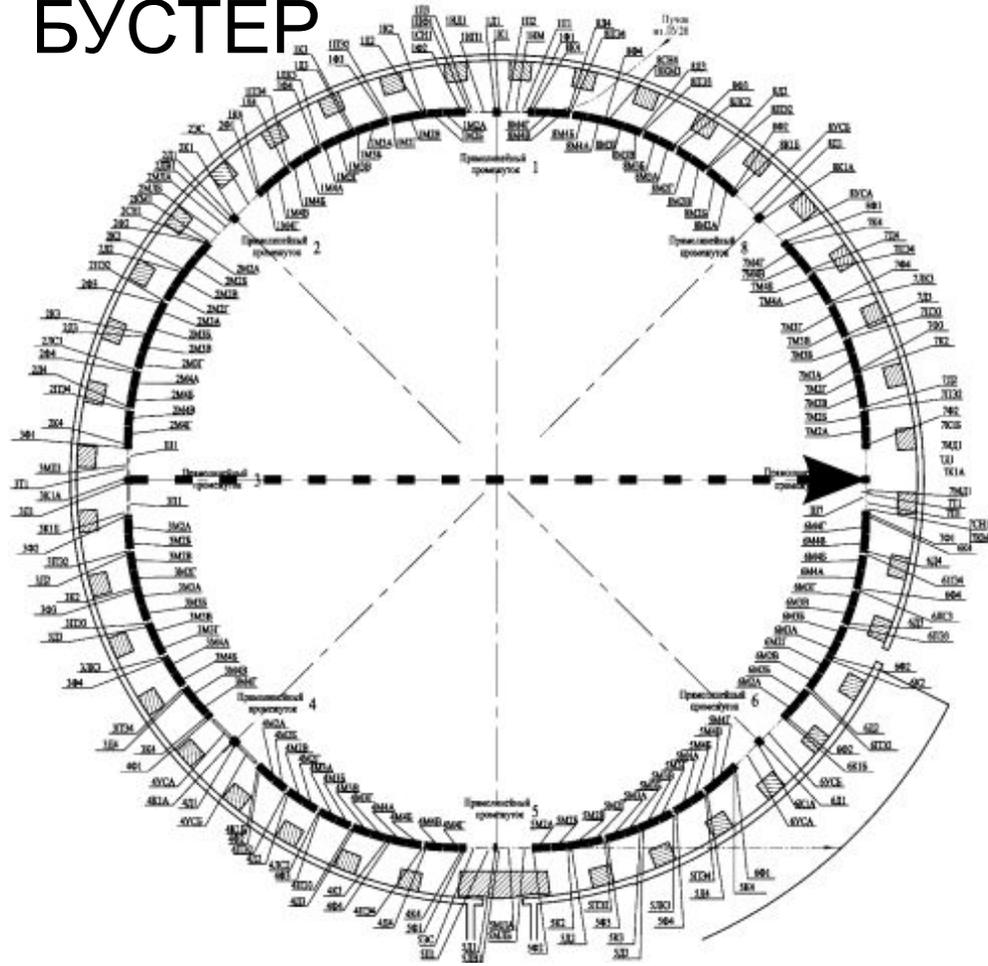


# АНАЛИЗАТОР СИГНАЛОВ ROHDE&SCHWARZ FSV-7



# СВЕРХПРОВОДЯЩИЙ БУСТЕР

# ПИКА



**Задача:** необходимо диагностировать интенсивность пучка с помощью пикапа экспериментальной системы охлаждения пучка

## Устройство, класс и сервер устройства в стандарте *TANGO*

«**Устройство**» — оборудование (анализатор сигналов и спектров), набор оборудования (4 двигателей, управляемых одним контроллером) или группа устройств, представляющих подсистему.

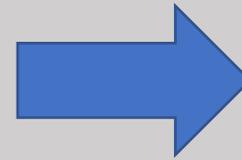
«**Класс устройства**» — абстрактная модель, определяющая интерфейс и реализацию управления устройством.

«**Сервер устройства**» — исполняемая программа (процесс), которая может создавать экземпляры устройств определенных классов.

# Последовательность шагов корректной разработки сервера устройства для анализатора сигналов ROND&SCHWARZ FSV-7

Определение интерфейса для сервера устройств. Создание модели класса сервера устройств и генерация шаблона кода.

Утилита *POGO*



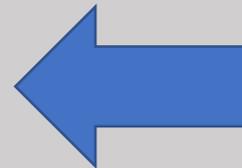
Установка дополнительных модулей. Программная реализация команд и методов чтения и записи атрибутов.

Язык *Python*



Первоначальный запуск и отладка сервера для проверки корректности его работы.

ПО *TANGO*+клиент *TANGO*



Регистрация класса сервера устройств и имени устройства в статической централизованной базе данных *TANGO*.

Утилита *JIVE* (администратор)



## AnalyzerRS

## Device Properties

- ✓ IP\_address

## Commands

- ✓ Get\_Freq
- ✓ Get\_Name
- ✓ Get\_Points
- ✓ Get\_Span
- ✓ Set\_Freq
- ✓ Set\_Points
- ✓ Set\_Span
- ✓ ValueX
- ✓ ValueY
- ✓ Status
- ✓ State

## States

- ✓ INIT
- ✓ ON
- ✓ FAULT

Команда	Вх.пар.	Вых.пар.	Назначение команды
Get_Name	void	string	Запрос идентификационной информации о приборе
Get_Freq	void	float	Запрос центральной частоты, на которой устройство анализирует мощности сигналов, МГц
Set_Freq	float	void	Установка центральной частоты для анализа мощности сигналов, МГц
Get_Span	void	float	Запрос полосы диапазона частот, МГц
Set_Span	float	void	Установка полосы диапазона частот, МГц
Get_Points	void	string	Запрос кол-ва точек измерения
Set_Points	string	void	Установка кол-ва точек измерения
ValueY	void	string	Запрос значений мощности анализируемых сигналов, Вт
ValueX	void	string	Запрос значений частот для каждого измеряемого сигнала, Гц
State	void	string	Запрос состояния устройства
Get_Status	void	string	Запрос статусной строки устройства

## ШАГ 2

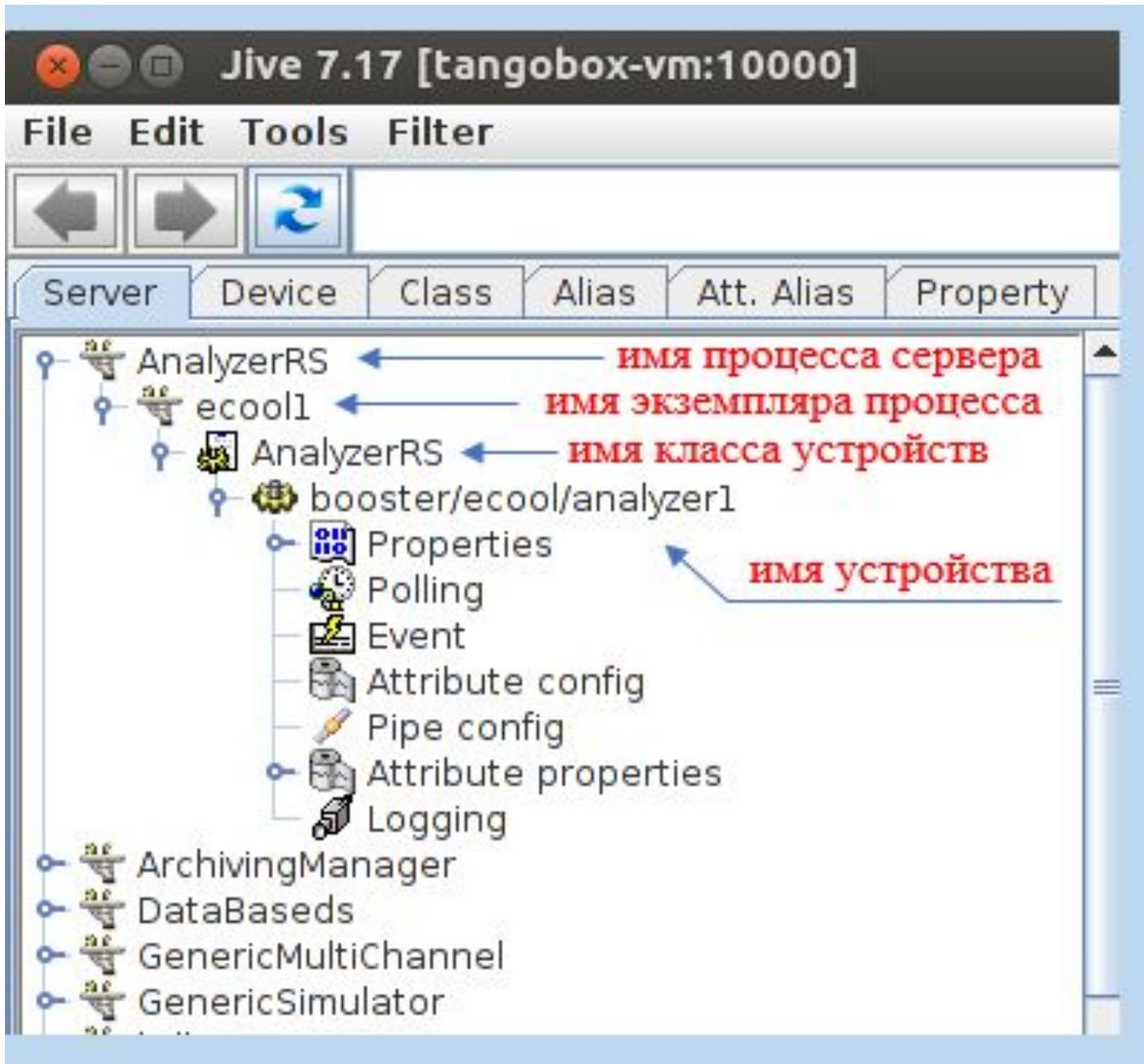
- Программная реализация команд на языке высокого уровня *Python*

**Под программной реализацией** понимается следующее:

- реализация инициализации устройства;
- реализация команд анализатора.

## Пример программной реализации команды для запроса центральной частоты, на которой работает анализатор

```
# Структура, определяющая функцию Python как команду устройства
# В качестве аргумента - выходной параметр (float)
@command(
dtype_out='float',
)
# Структура, позволяющая отслеживать работу команды
@DebugIt()
# Команда для определения центральной частоты устройства
def Get_Freq(self):
    # Отладочная информирующая строка
    self.debug_stream("In Get_Freq()")
    # Переменная, принимающая центральную частоту
    argout = None
    # PROTECTED REGION ID(AnalyzerRS.Get_Freq) ENABLED START #
    # Запрос на устройство, возвращающий значение центральной частоты
    argout=self.inst.query("FREQ:CENT?")
    # PROTECTED REGION END # // AnalyzerRS.Get_Freq
    # Возвращение результата запроса (перевод частоты в MHz)
    return float(argout)/(10**6)
```



## ШАГ 3

### Регистрация

- сервера
  - устройства,
  - класса устройства,
  - имени устройства
- в централизованной

Имя устройства: БД

booster/ecool/analyzer1

## ШАГ 4

### Первоначальный запуск и отладка сервера устройства для анализатора сигналов

- Отладка работы сервера устройства с помощью утилиты

*JIVE*

- Проверка работы сервера устройства с помощью приложения-клиента

Argin value

1800 ← **Входной параметр для команды Set\_Freq**

Argin Type	Argout Type
DevVoid	DevDouble

Get\_Freq  
Get\_Name  
Get\_Span  
Get\_Status  
Init  
Set\_Freq  
Set\_Span  
State

Show description  
Execute  
Plot

Command: booster/ecool/analyzer1/Set\_Freq  
Duration: 0 msec  
Command OK ← **Результат команды**  
Установка цен. частоты

Command: booster/ecool/analyzer1/Get\_Freq  
Duration: 0 msec  
Output argument(s) :  
1800.0 ← **Результат команды**  
Запрос значения частоты

Проверка команд  
Анализатора  
сигналов  
*Rohde&Schwarz FSV-7*  
в отладочном  
режиме работы  
сервера

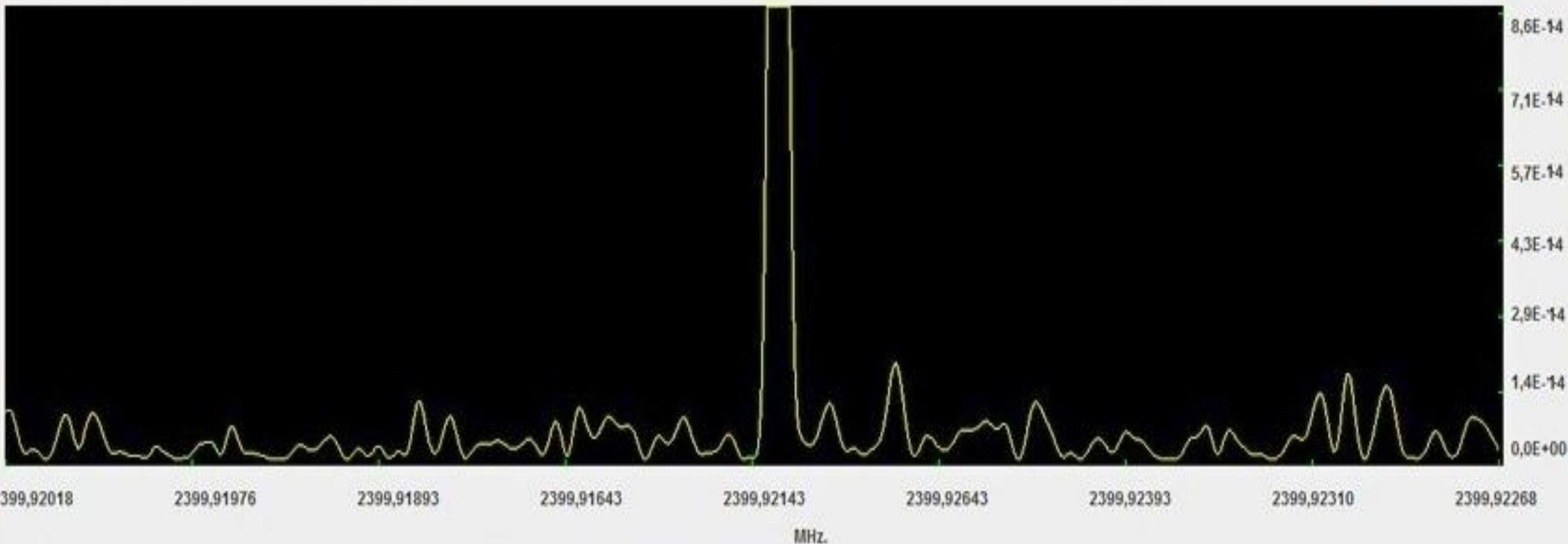
```
Администратор: C:\Windows\System32\cmd.exe - python AnalyzerRS.py ecool1 -v4
D:\User\python\tango\cppserver\pythonHL\AnalyzerRS>python AnalyzerRS.py ecool1 -v4
Ready to accept request
1558093580 [6056] DEBUG booster/ecool/analyzer1 -> AnalyzerRS.Set_Freq(<)
1558093580 [6056] DEBUG booster/ecool/analyzer1 In Set_Freq(<)
Set Center Frequency 1800.0 MHz
1558093580 [6056] DEBUG booster/ecool/analyzer1 <- AnalyzerRS.Set_Freq(<)
1558093614 [6056] DEBUG booster/ecool/analyzer1 -> AnalyzerRS.Get_Freq(<)
1558093614 [6056] DEBUG booster/ecool/analyzer1 In Get_Freq(<)
1558093614 [6056] DEBUG booster/ecool/analyzer1 <- AnalyzerRS.Get_Freq(<)
```

# Контрольные измерения на частотах близких к максимальной чувствительности пикапа 2 ГГц с помощью приложения-клиента

Rohde&Schwarz,FSV-7,1307.9002K07/102110,2.30 SP1

Мощность, Вт

Параметры



# Заключени

- Разработан сервер устройства для анализатора сигналов *Rohde&Schwarz FSV-7* в рамках стандарта *TANGO*;
- Весь требуемый функционал был полностью реализован;
- Проведены регистрация и запуск сервера устройства;
- Работа сервера была отлажена, полученные в ходе тестовых измерений данные соответствуют действительности.

# Спасибо за внимание

Автор ВКР: Лукьянов М.А.

Руководитель: ст. преп. Андреев В.  
А.