Системы управления крупными ускорительными установками, основанные на технологии Tango Controls

На примере системы управления ускорительного комплекса NICA.

Седых Георгий Сергеевич, georgy.sedykh@gmail.com, https://t.me/cedbix

Ускорительный комплекс NICA

Система управления :: Статус

Бустер:

- Циклозадающая аппаратура
- Диагностика инжекции
- Термометрия
- Измерение орбиты
- Коррекция орбиты
- Измерение тока пучка
- Q-метр
- Вывод
- Синхронизация
- Вакуум
- ВЧ
- Защита от переходов
- Электронное охлаждение
- СБИС

Нуклотрон:

- Циклозадающая аппаратура
- Диагностика инжекции
- Термометрия
- Измерение орбиты
- Коррекция орбиты
- Измерение тока пучка
- Вывод
- ВЧ

Коллайдер:

- Циклозадающая аппаратура
- Диагностика инжекции
- Термометрия
- Измерение орбиты
- Коррекция орбиты
- Измерение тока пучка
- Q-метр
- Синхронизация
- SLM мониторы
- ВЧ
- Электронное охлаждение
- Вакуум

Отдельно

- DocDB
- Сохранение состояния системы

KRION + ЛУТИ:

СУ работает автономно с возможностью интеграции

Каналы перевода:

СУ строится на базе Tango, все будет полностью интегрировано в общую СУ

- Сделано
- Интегрировано
- Возможно интегрируется
- В работе

Основные задачи системы управления

1. Взаимодействие с аппаратными средствами.
2. Взаимодействие с пользователем.
3. Реализация логики работы, присущей данной конкретной установке.

Дополнительные задачи системы управления

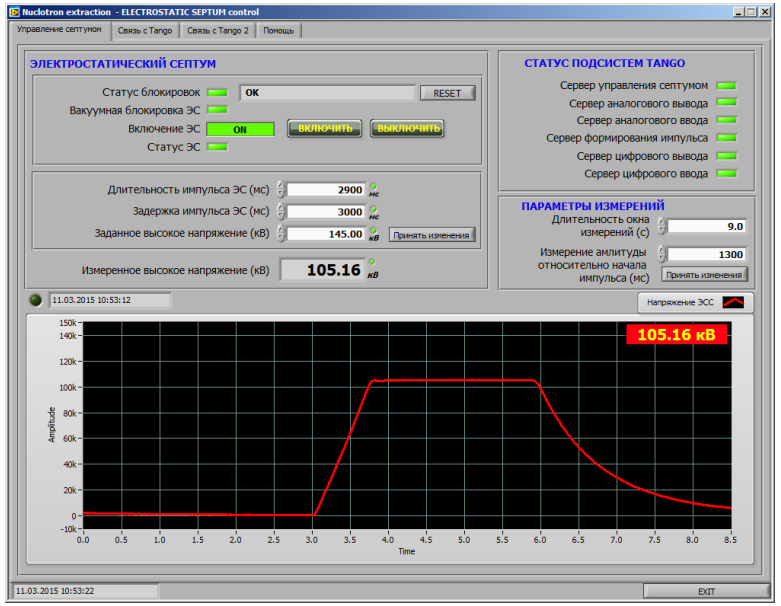
- Средства обеспечения работоспособности;
- Авторизация и разграничение прав доступа;
- Стандартизованная система конфигурации элементов;
- Автоматический запуск и остановка;
- Архивация и извлечение архивных данных;
- Логирование;
- Мониторинг;
- Стандартизованная обработка ошибок;
- Библиотеки и наборы инструментов для создания клиентских приложений

Система управления с единой программой

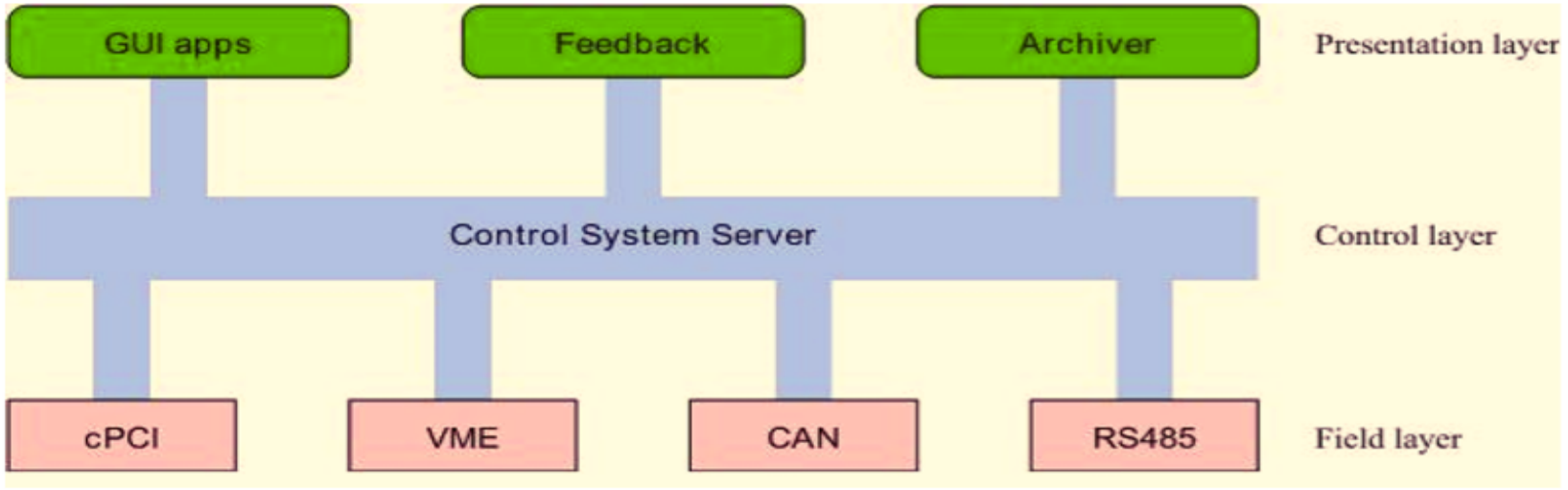
Для небольших, «настольных» установок все три функции обычно выполняются единой программой, которая и реализует графический интерфейс пользователя, взаимодействует с аппаратурой и содержит некую специфичную для конкретной установки логику (управление рабочим циклом установки, обратные связи, слежение за различными параметрами, и т. п.).

Недостатки:

- Трудно обслуживать большое количество электроники одним компьютером и одной программой.
- Большой объем кода и повышенная сложность программы.
- Трудно обеспечить доступ разных программ к одному устройству.



Трехуровневая архитектура ПО системы управления



Достоинства:

- разграничение функций;
- распределенность;
- разрешение конфликтов доступа.

Следствием этих достоинств является масштабируемость – одно из основных требований, предъявляемых к крупным системам управления.

– связующая технология

- Предназначена для построения систем управления крупными физическими и промышленными установками;
- Основана на принципе распределённых tango-устройств;
- Скрывает сетевое взаимодействие;
- Бесплатное и мультиплатформенное ПО с открытым исходным кодом;
- Набор инструментов для централизованной настройки, контроля и управления всеми подключёнными tango-устройствами;
- Серверные и клиентские библиотеки для языков C++, Python, Java;
- Создание клиентских приложений на web, Qt, Taurus, LabView, MatLab,…

Базовые понятия и принципы технологии Tango Controls

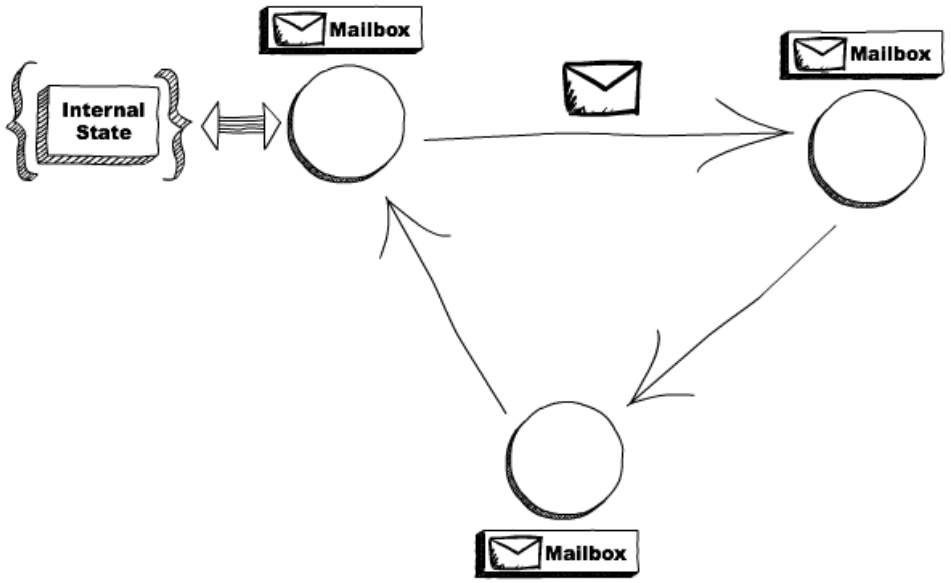
1. Система Tango
2. Устройство
3. Атрибуты
4. Команды
5. Свойства
6. Модели взаимодействия
7. События
8. Машина состояний
9. Класс устройства
10. Сервер устройства
11. База данных Tango
12. Утилиты
13. Клиентские технологии

Система Tango

Tango = Actors + Microservices

Модель Акторов

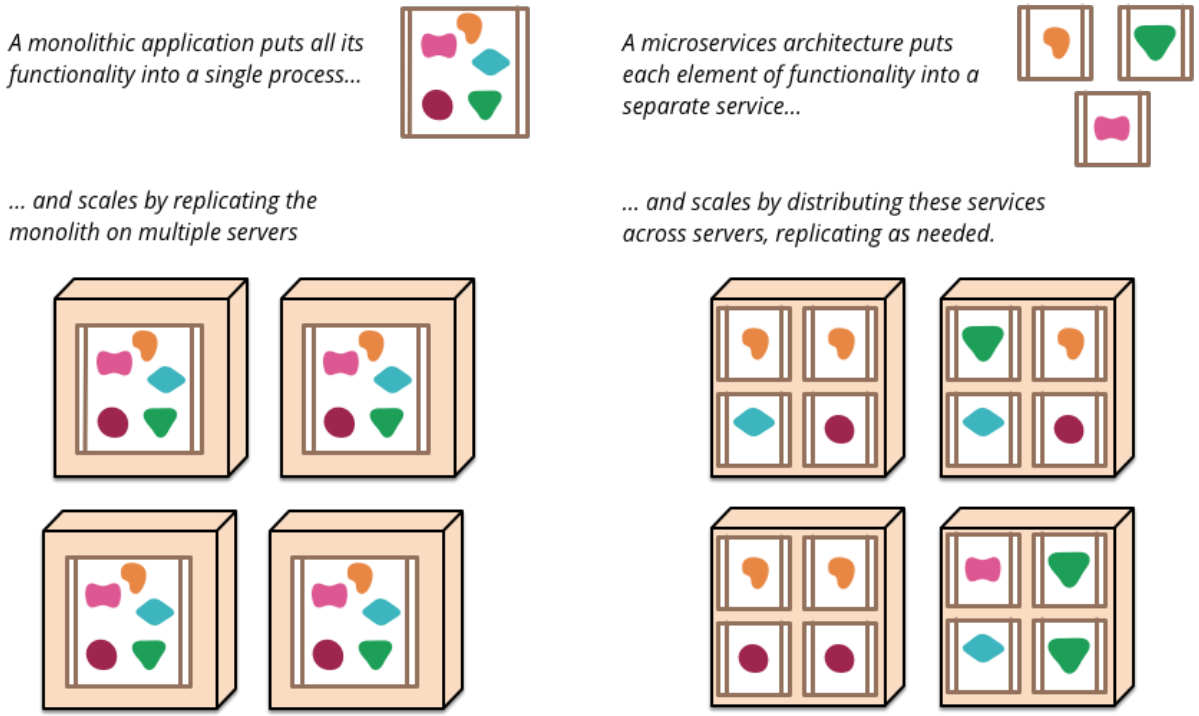
Актор — примитивная единица, получающая сообщения и делающая примитивные вычисления, основанные на этих сообщениях. Акторы изолированы друг от друга и имеют внутреннее состояние которое не может быть изменено извне.



Микросервисы

Микросервисная архитектура - подход к разработке программного обеспечения, основанный на использовании насколько это возможно небольших, слабо связанных и легко заменяемых модулей - микросервисов.

Микросервисы обычно оснащены стандартизированными интерфейсами и взаимодействуют по стандартным протоколам.



Система Tango

- Tango основана на принципе **Распределенных Устройств.**

Это является реализацией **Модели Акторов.**

- Каждое устройство в системе Tango представляет собой **Микросервис.**

- Бесплатное, свободное программное обеспечение

- Использует технологии CORBA и ZMQ в качестве канального уровня

- Активно разрабатывается и поддерживается сообществом инженеров и операторов физических установок (ESRF, Elettra, SOLEIL, MAX IV, …)

Устройство (Device)

- **Устройство** – это объект, представляющий собой микросервис в системе Tango. Оно является программным отображением реального устройства или любой логической сущности системы управления.
- Tango ID: **domain/family/member**
- **Состояние**.
- **Интерфейс**: атрибуты, команды, свойства
- Устройства являются экземплярами **Классов** (Device Class) и выполняются в **Серверах Устройств** (Device Servers).

- Классы устройств могут быть реализованы на языках C++, Python, Java

Атрибуты (Attributes)

- Атрибуты являются частью интерфейса устройства. Они представляют поля данных. Клиентские приложения могут прочитать их, записать или подписаться на события.
- Типы (логический, строка, целые числа, дробные числа, символы, ..)
- Размерность (скалярные, спектральные (1D), картинки (2D))
- Самоописание (единицы измерения, максимальные и минимальные значения, предупреждения, формат отображения, ..)
- Атрибуты могут быть прочитаны один за другим или сразу несколько.

Команды (Commands)

- Команды являются частью интерфейса устройства. Они представляют действия, которые клиентское приложение может выполнить. Команды могут изменять внутреннее состояние устройства.
- Команды могут принимать один входной параметр и возвращать один выходной. Параметры могут быть любого из типов данных системы Tango.

Свойства (Properties)

- Свойства – это данные, хранящиеся в базе данных, и используемые для конфигурации Устройства при его запуске. Свойства могут быть любых типов данных Tango. Механизм свойств позволяют создавать обобщенные Классы Устройств.
- Свойства обычно редактируются при помощи утилиты Jive.
- Примеры: ip адрес, количество каналов, начальные значения токов, ..

Модели взаимодействия

1. Синхронное. Основной поток после запроса блокируется и ждет, пока не придет ответ или наступит таймаут.
2. Асинхронное. Основной поток после запроса продолжает свою работу. Результат проверяется спустя некоторое время. Также удобно использовать механизм с функцией обратного вызова.
3. Событийное

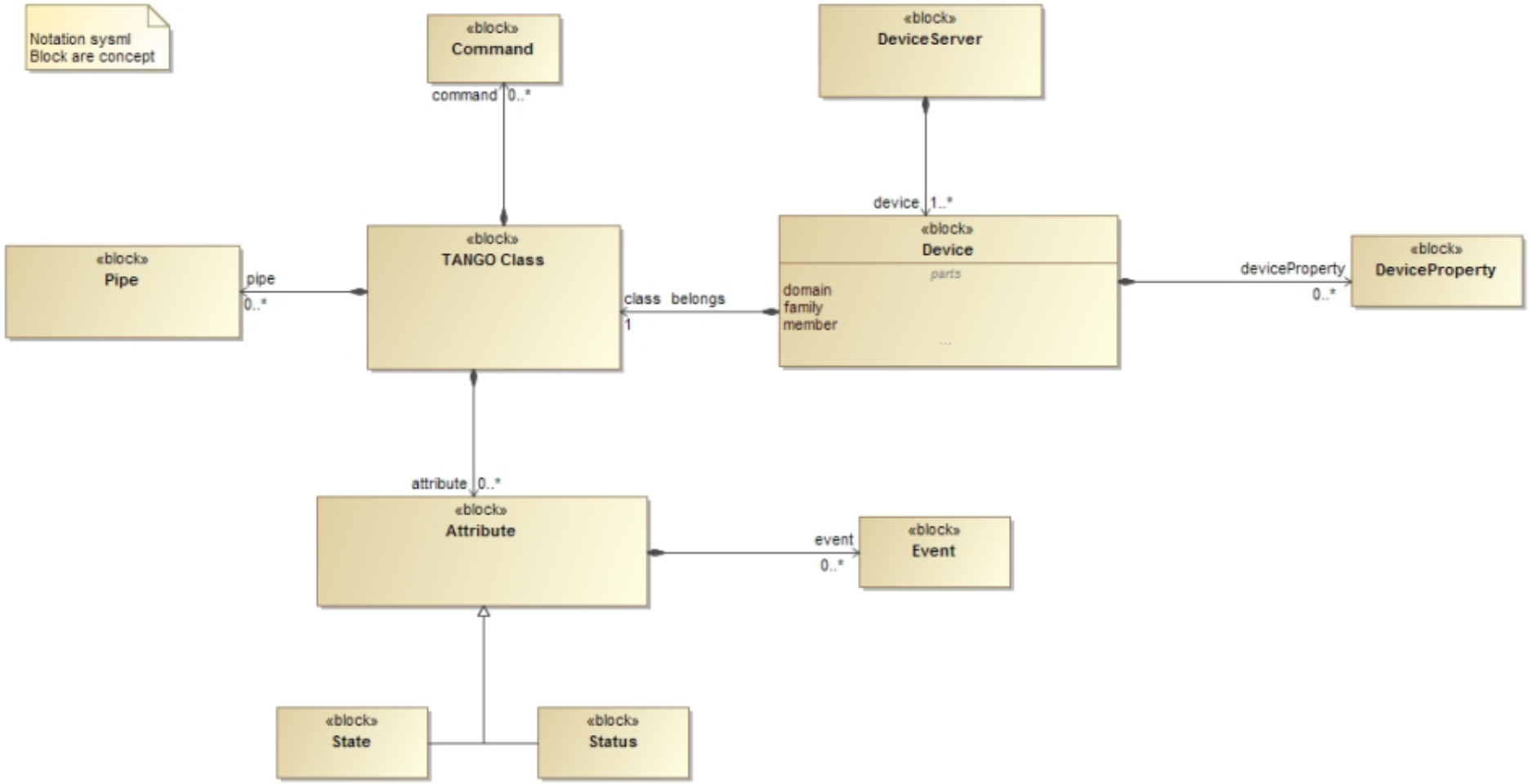
События (Events)

- События Tango – модель взаимодействия между клиентом и сервером в формате издатель-подписчик. События могут работать только для Атрибутов или Пайпов.
- Типы событий: по изменению (change), периодические (periodic), архивные (archiving), пользовательские (user), ..
- Пример: Посылать событие если значение атрибута изменяется на 1% или на 1 единицу измерения.
- Механизм событий использует библиотеку ZMQ и является наиболее эффективным способом взаимодействия.
- События можно использовать через периодическую проверку наступивших событий (pull), либо используя механизм функций обратного вызова (push)

Машина состояний (State machine)

- Все Устройства имеют внутренние состояния. В системе Tango существует 14 различных значений состояния (ON, OFF, FAULT, MOVING, OPEN, CLOSED, STANDBY, UNKNOWN, ..). Каждый Класс Устройств содержит механизм смены состояний.
- Состояния являются очень мощным механизмом для защиты Устройства и для передачи информации клиенту.

Упрощенная модель Устройства



Tango Controls :: Device- устройство

- Программное отображение реального устройства или логической сущности в системе управления;
- Уникальный идентификатор domain/family/member;
- Имеет внутреннее состояние (ON, OFF, FAULT, …);
- Имеет статус (понятное текстовое описание состояния);
- Взаимодействуют через интерфейс:
- Атрибуты (поля данных);
- Команды (действия);
- Настраивается при помощи свойств;
- Взаимодействуют синхронно, асинхронно или по событиям;
- Принадлежит к определенному tango-классу;
- Выполняется внутри сервера устройств tango.

Источник питания

- Состояние
- Статус
- Ток
- Напряжение
- Режим

Атрибуты

- Включить вывод
- Выключить вывод

Команды

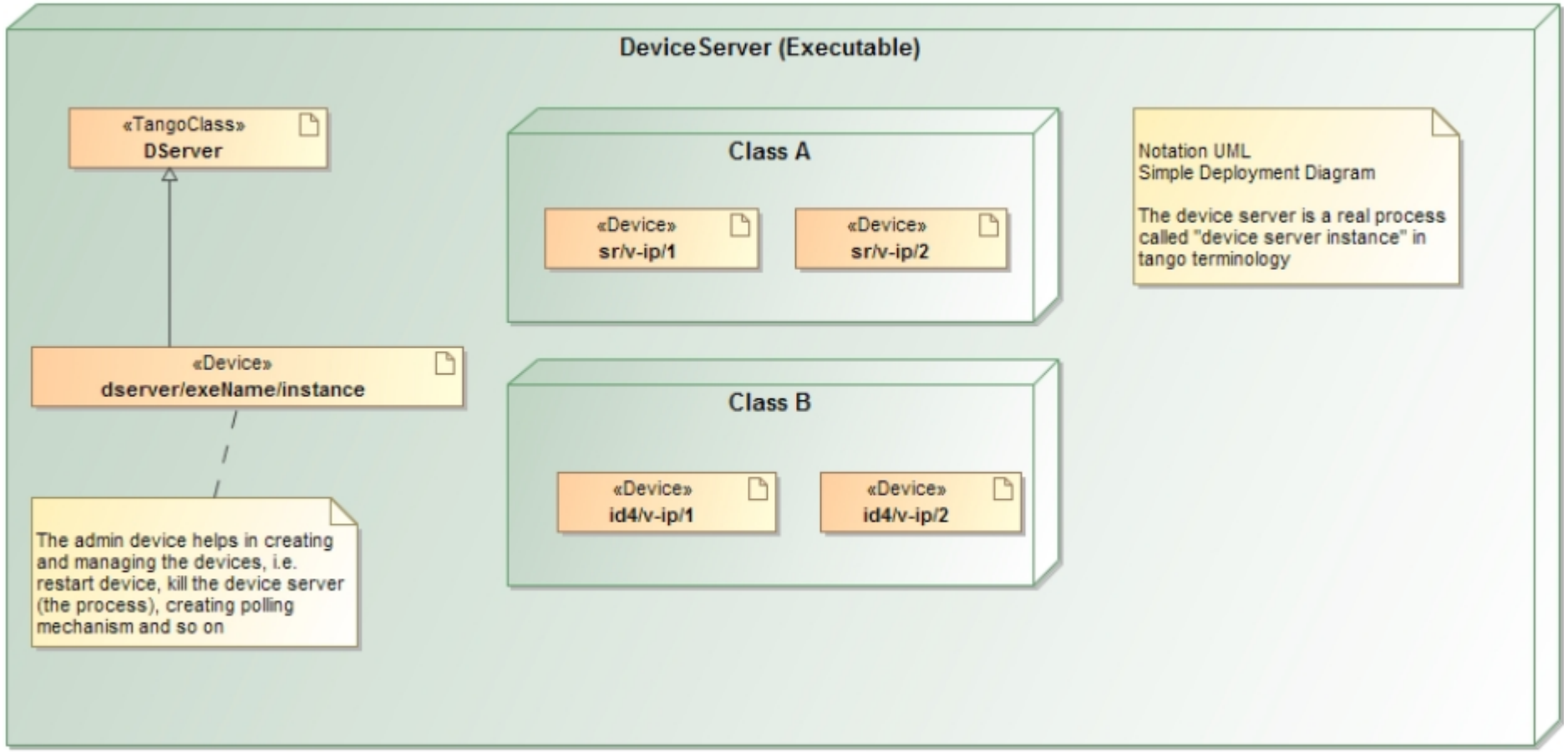
- Адрес
- Порт

Свойства

Класс Устройства (Device Class)

- Каждое Устройство системы Tango является представителем какого-либо Класса Устройств. Класс Устройства реализует обобщенное поведение Устройств. При помощи Свойств происходит конфигурация конкретного Устройства.
- Примеры: MyPowerSupply, SerialLine, Modbus, ..
- Разработчики Серверов Устройств фактически разрабатывают именно Классы Устройств.
- Они могут быть реализованы используя языки C++, Python, Java.

Сервер Устройства (Device Server)



База данных Tango

- База данных Tango служит для хранения структуры системы и конфигурационной информации для устройств.
- Интерфейс базы реализован в виде специального Сервера Устройства. Клиенты получают доступ к базе данных через стандартизованный клиентский программный интерфейс Tango. Поддерживается только реализация с базой MySQL.
- Каждая база данных имеет свою точку доступа, которая определяется переменной окружения TANGO_HOST=host:port.
- В пределах системы Tango могут одновременно работать несколько баз данных.

Утилиты и инструменты

- Jive – менеджер базы данных Tango.
- Pogo – генератор шаблонов исходных кодов для Классов Устройств
- Astor – утилита для управления основными узлами системы и назначения прав доступа.
- LogViewer – утилита для просмотра отладочной информации.
- HDB++ - система архивации данных.

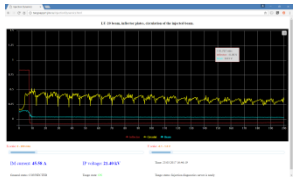
Создание клиентских приложений

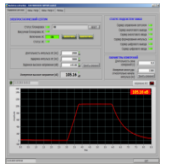
- C++ Device API, QTango
- Python – PyTango, Taurus
- Java Device API, Tango ATK, mTango
- LabView Tango binding
- Matlab Tango binding
- Tango REST API

Клиентский уровень :: Python клиент

```
import tango

try:
    #Подключение к устройству
    device = tango.DeviceProxy("sys/tangotest/3")

    #Чтение атрибута double_scalar
    doubleValue = device.double_scalar
    print(doubleValue)

    #Запись атрибута double_scalar
    device.double_scalar = 123.45

    #Выполнение команды DevDouble
    argout = device.DevDouble(23.89)
    print(argout)

except Exception as e:
    print(str(e))
```

Трёхуровневая структура системы управления комплекса NICA





Веб-клиент

Клиент для ПК

REST, WEBSOCKET

TANGO

Клиентский уровень

TANGO Connecting things together

Танго-устройство, отображающее управляемую систему

Как правило:

*Представляет физические параметры в виде **атрибутов**.*

*Для смены режимов работы или внутреннего состояния приметяются **команды**.*

*Для конфигурации используются **свойства***

Сервисы

- Web API
- Система архивации
- Контроль прав доступа
- Мониторинг
- Логирование

Сервисный уровень

Танго драйвер | Танго драйвер | Танго драйвер | Танго драйвер | Танго драйвер | Танго драйвер | Танго драйвер

Уровень оборудования

СПЕЦИАЛЬНЫЕ БИБЛИОТЕКИ

ПРОТОКОЛЫ СВЯЗИ

Modbus

ДРАЙВЕРЫ И SDK

OPC FOUNDATION

УСТРОЙСТВА -ШЛЮЗЫ

КРЕЙТЫ С ОБОРУДОВАНИЕМ

КОНТРОЛЛЕРЫ

scada

EPICS

АЦП/ЦАП

ОСЦИЛЛОГРАФЫ

TANGO

Ссылки

- Официальный сайт Tango Controls http://www.tango-controls.org/
- Документация по Tango https://tango-controls.readthedocs.io/en/latest/
- Tango Controls Concepts. A brief introduction to Tango Controls Concepts. Andy Gotz, ICALEPCS 2017.
- Модель акторов. Arthur Arsalanov. https://medium.com/@arturarsalanov/%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C-%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%BE%D0%B2-e2da975fff68
- Микросервисная архитектура https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BA%D1%80%D0%BE%D1%81%D0%B5%D1%80%D0%B2%D0%B8%D1%81%D0%BD%D0%B0%D1%8F_%D0%B0%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0