WSDL-DSL + WSToolKit

full automatization of Web Services testing

Laura M. Castro

University of A Coruña

Jun 10, 2015

EUC (tutorial)

What is a ‘web service’?

- Nowadays, anything and everything claims to be a WS
- Conservative characterization:
  - Integration mechanism with third-party components/systems
  - Interaction over the internet (HTTP)
  - Data exchange using standards like JSON/XML

How are WS used/tested?

- You are given an API description in a computable format, plus a with clear definition of its semantics, and both are stable and work as indicated
  - From the API description you can (automatically) generate your integration stubs, then implement the logic according to said semantics

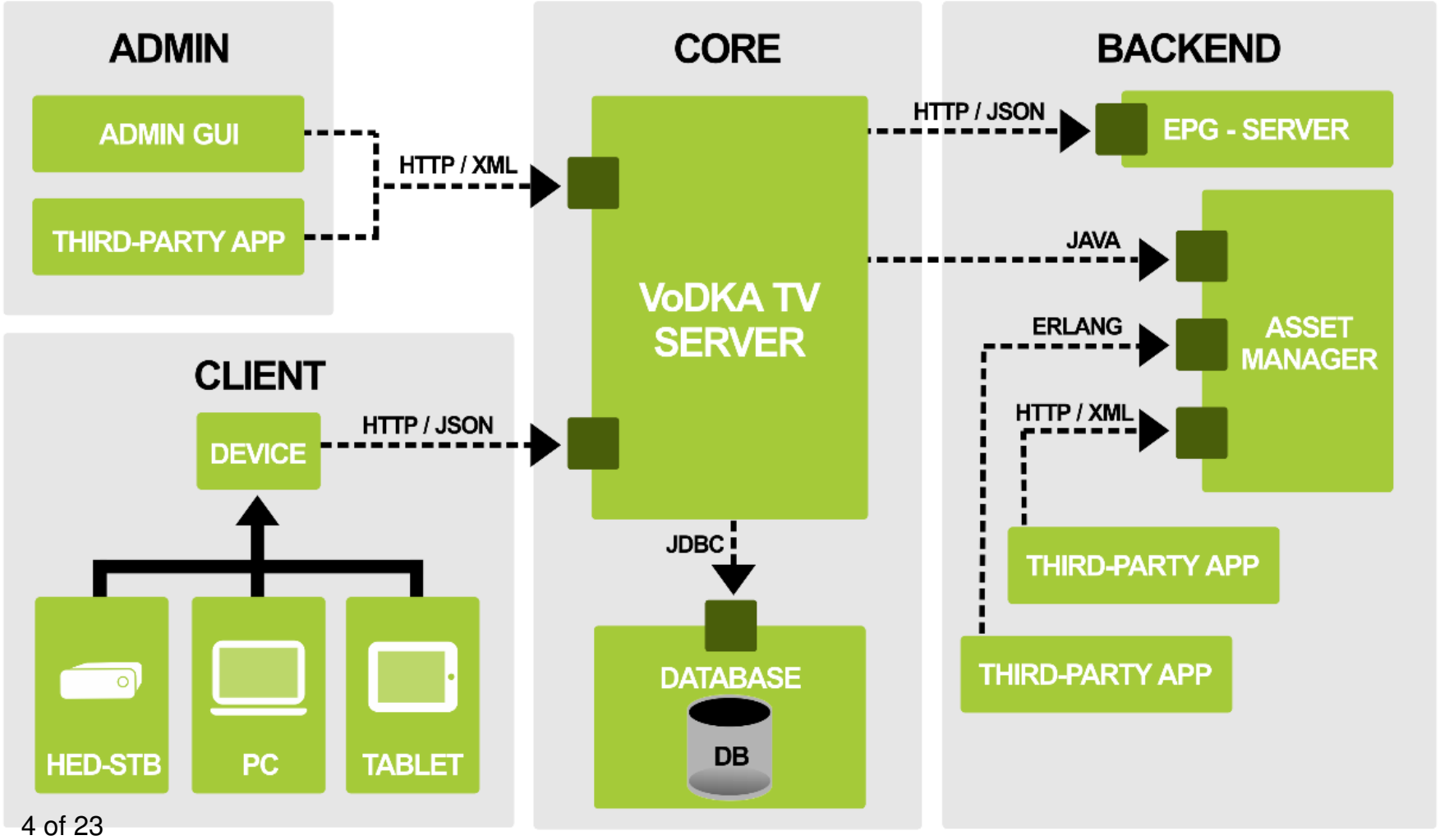
How are WS used/tested?

- You are given an API description in a computable format, plus a with clear definition of its semantics, and both are stable and work as indicated
  - From the API description you can (automatically) generate your integration stubs, then implement the logic according to said semantics
  - We want at least the same for testing!

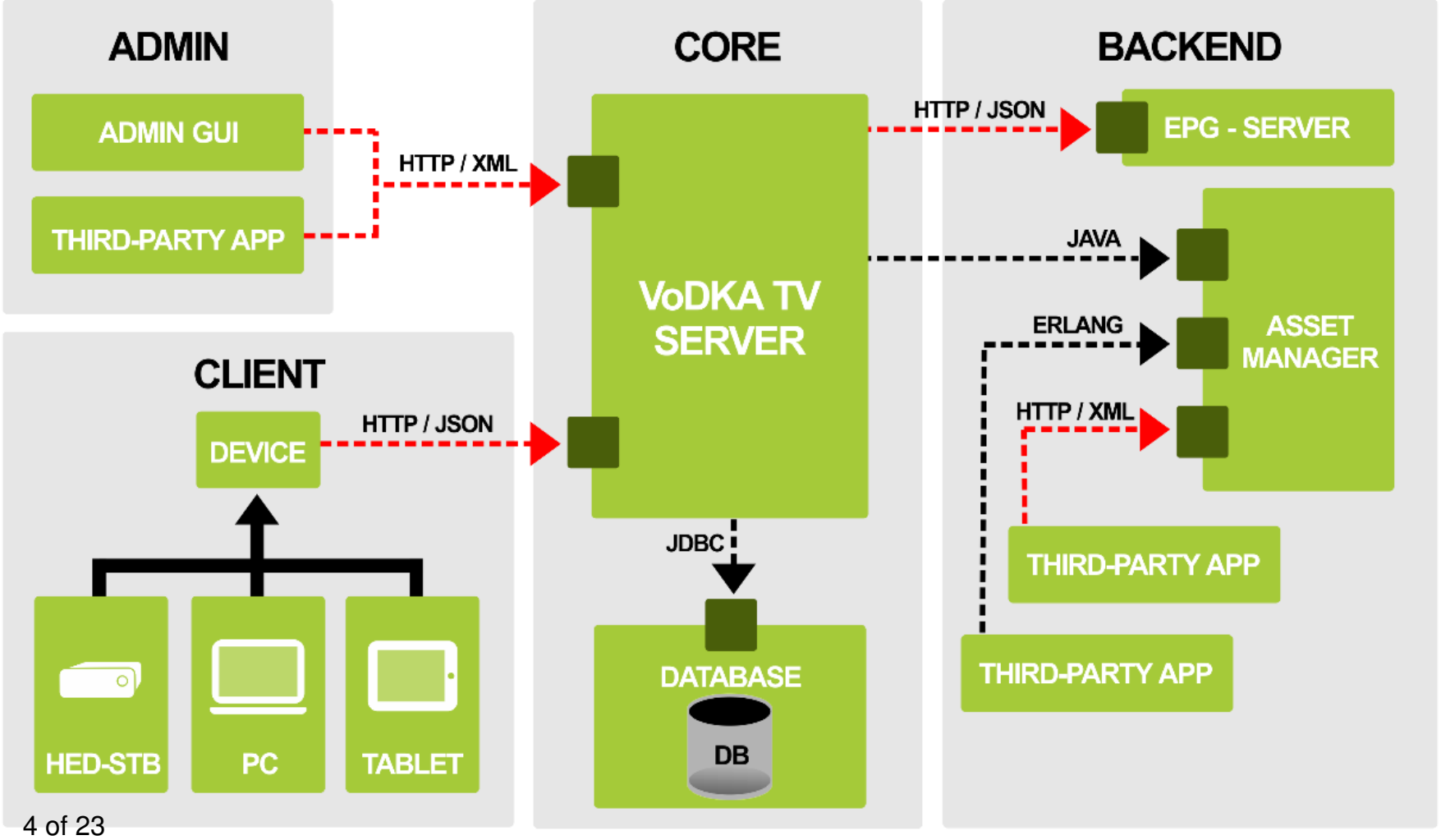
How are WS used/tested?

- You are given an API description in a computable format, plus a with clear definition of its semantics, and both are stable and work as indicated
  - From the API description you can (automatically) generate your integration stubs, then implement the logic according to said semantics
  - We want at least the same for testing!
- You are given some instructions in a PDF (or email text)

Industrial example: VoDKATV



Industrial example: VoDKATV



VoDKATV: the past

- API description and semantics in PDF format
- New PDF version when changes occurred
- Lots of effort put into reviewing document, applying changes, designing & writing unit tests

WSToolkit

- **Offers**: avoid all the boilerplate code

  Push-button technology for property-based testing a WS
  - Saves a lot of effort
  - Improves using QC over unit tests
- **Requires**: an API description in a computable format (WSDL)
- Available at:

  `https://github.com/prowessproject/WSToolkit`

WSToolkit

- **Offers**: avoid all the boilerplate code

  Push-button technology for property-based testing a WS
  - Saves a lot of effort
  - Improves using QC over unit tests
- **Requires**: an API description in a computable format (WSDL [+XSD])
- Available at:

  `https://github.com/prowessproject/WSToolkit`

WSToolkit: VoDKATV demo

1. Use XSD file to obtain HRL with **data definitions**
2. Use WSDL, XSD and HRL files to obtain **wrapper for WS**
3. Use WSDL, XSD to obtain **QC generators**
4. Use WSDL, XSD, Erlang wrapper, and QC generators to obtain **QC test model**

WSToolkit: VoDKATV demo (1)


```
write_hrl:write_hrl_file("vodkatv.xsd",
                         "vodkatv.hrl").
```

WSToolkit: VoDKATV demo (1)

```
write_hrl:write_hrl_file("vodkatv.xsd",
                         "vodkatv.hrl").
```

WSToolkit: VoDKATV demo (2)


```
write_sut_api:write_sut_api("vodkatv.hrl",
                            "vodkatv.wsdl",
                            "vodkatv.xsd",
   "http://10.20.19.102:8082/vodkatv",
                            "vodkatv_sut.erl").
```

WSToolkit: VoDKATV demo (2)


```
write_sut_api:write_sut_api("vodkatv.hrl",
                            "vodkatv.wsdl",
                            "vodkatv.xsd",
   "http://10.20.19.102:8082/vodkatv",
                            "vodkatv_sut.erl").
```

WSToolkit: VoDKATV demo (3)


```
write_data_gen:write_data_generators_to_file(
                                "vodkatv.xsd",
                                "vodkatv.wsdl",
                                "vodkatv.erl").
```

WSToolkit: VoDKATV demo (3)


```
write_data_gen:write_data_generators_to_file(
                                    "vodkatv.xsd",
                                    "vodkatv.wsdl",
                                    "vodkatv.erl").
```

WSToolkit: VoDKATV demo (4)


```
write_eqc_statem:write_eqc_statem(
                                  "vodkatv.wsdl",
                                  "vodkatv.xsd",
                                  "vodkatv.hrl",
                                  "vodkatv_sut",
                                  tuple | non_tuple,
                                  "vodkatv_eqc.erl").
```

WSToolkit: VoDKATV demo (4)


```
write_eqc_statem:write_eqc_statem(
                                  "vodkatv.wsdl",
                                  "vodkatv.xsd",
                                  "vodkatv.hrl",
                                  "vodkatv_sut",
                                  tuple | non_tuple,
                                  "vodkatv_eqc.erl").
```

Beyond WSToolkit

- You still need to *do* something: **semantics**
  - If you have them in OCL, you can close the circle
- You still may want to *improve* something: **data generation**

Beyond WSToolkit

- You still need to *do* something: **semantics**
  - If you have them in OCL, you can close the circle

    (ask us if you're interested!)
- You still may want to *improve* something: **data generation**

Improving data generation

- WS data can include a number of **constraints**:
  - optional information
  - number of ocurrences
  - patterns for content
  - etc.
- We want to take into account these contraints **before** generating the data, **not afterwards**

Improving data generation

```
int_list_gt(N) ->
    list(int_gt(N)).

int_gt(N) ->
    ?SUCHTHAT(M, int(), M>N).

> eqc_gen:sample(sample:int_list_gt(100)).
** exception exit: "SUCHTHAT failed to find
                    a value within 100 attempts."
```

WSDL-DSL

- We created a DSL with a WSDL-flavour to improve data generation for testing web services
- It is a stand-alone add-on to the WSToolkit
  - you can use it even if you only need data generation (i.e. if you write your own properties, or unit tests)
- Available at:

  `https://github.com/prowessproject/WSDL-DSL`

WSDL-DSL: VoDKATV demo

vodkatv.xsd

```
<xsd:element name="rooms">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="roomId" type="xsd:string" />
            <xsd:element name="description" type="xsd:string" minOccurs="0" maxOccurs="1" />
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

WSDL-DSL

```
wsdlType(tag("rooms",
             xlist(tag("room", [                        tag("roomId",      string()),
                                minOccurs(0,maxOccurs(1,tag("description", string()))))])))).
```

WSDL-DSL: VoDKATV demo

WSDL-DSL pretty printing

```
<rooms>
 <room>
       <roomId>82s</roomId>
       <description></description>
 </room>
 <room>
       <roomId>x 26K6pEL </roomId>
       <description>G  </description>
 </room>
 <room>
       <roomId>  2</roomId>
       <description></description>
 </room>
 <room>
       <roomId>fxQel(d3n</roomId>
 </room>
</rooms>
```

Conclusions

- If you have a web service and you have a WSDL, **take advantage** of WSToolkit + WSDL-DSL
- If you have a web service and you don't have a WSDL, ask for it, now you know how **useful** it can be
- If you have a web service, and a WSDL that **changes** WSToolkit can also help with that *(ask us!)*

Conclusions

- Use them, report your issues, contribute!
- and **thanks** for listening!! :-)