Rich internet Applications in Higher Education Admissions Systems

MIRKO STANIĆ   <MIRKO.STANIC@AZVO.HR>

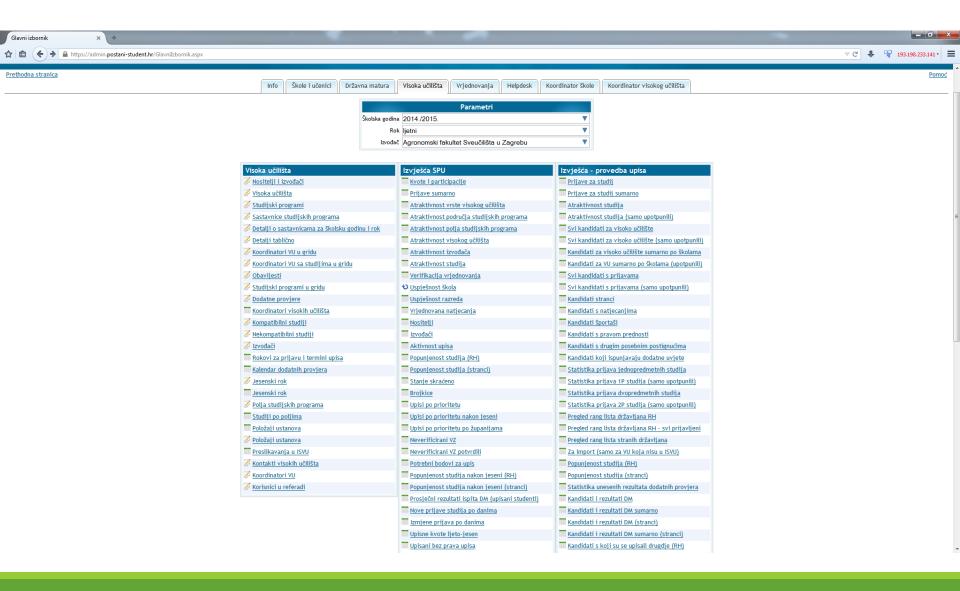IGOR DRVODELIĆ   <IGOR@AZVO.HR>

Introduction

- applications have to adapt to HEI's needs

- changes can happen between two enrollment turns

- implementation deadline can be shorter then 1 year

- system need to be able to adapt quickly

- stop viewing web systems as web sites and view them as web applications

NISpVU

- custom built

- monolithic web system

- each page represents only the current state

- changes require complete page refresh

- heavy load on server

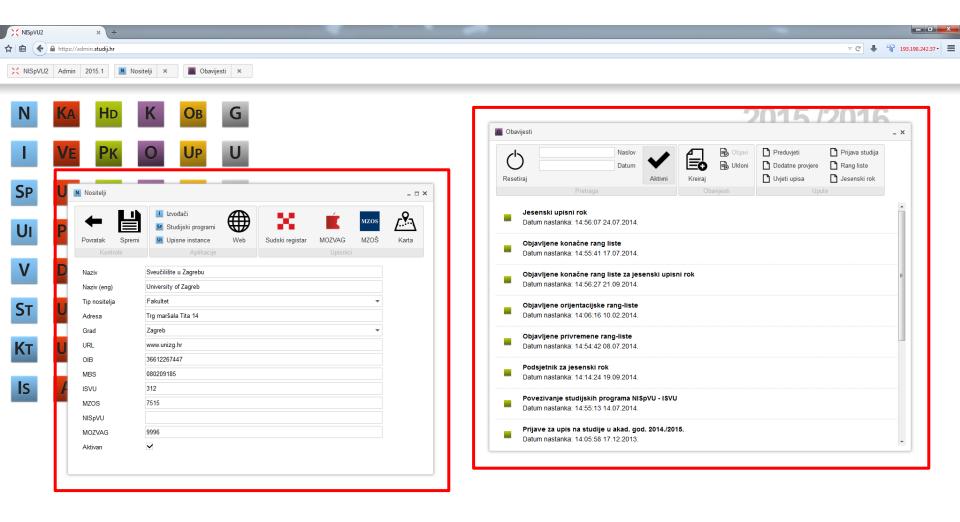- thin client architecture
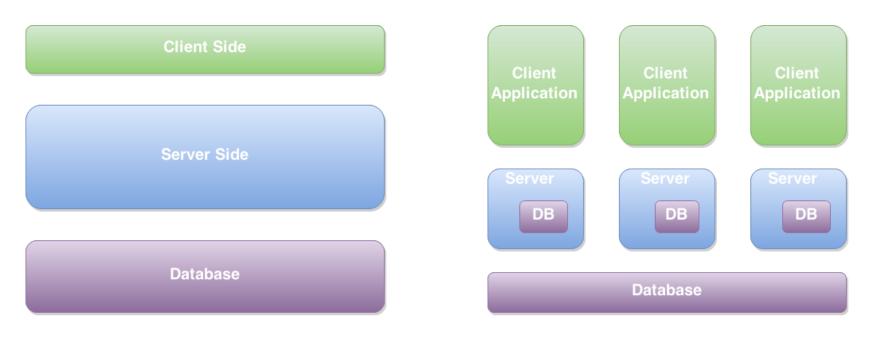
- ASP.NET, MSSQL

NISpVU user interface

NISpVU2

- custom built

- modular web system

- individual features are organized into „applications"

- changes made in one application do not affect other (feature decoupling)

- fine granulation of access control

- client heavy system, most of the work is offloaded onto the browser

- PHP, PostgreSQL

NISpVU2 user interface

Architecture comparison

Data interchange

- comunication between „applications" and client/server thriugh JSON

- ~50 times faster parsing compared to XML

- lower overhead compared to XML

- native to JavaScript

- used to create a unified messaging format for NISpVU2

{system:sysName, application:appName, action:actionName, payload:data}

Data interchange

JSON	XML
- less overhead	- high markup overhead
- native support in JavaScript	- strong semantics

```
{
    "name": "Veleučilište u Rijeci",
    "type": "Javno veleučilište",
    "city": "Rijeka",
    "url": "www.veleri.hr"
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<faculty>
        <name>Veleučilište u Rijeci</name>
        <type>Javno veleučilište</type>
        <city>Rijeka</city>
        <url>www.veleri.hr</url>
</faculty>
```

98 characters
40 markup characters (40%)

165 characters
107 markup characters (65%)

THE END