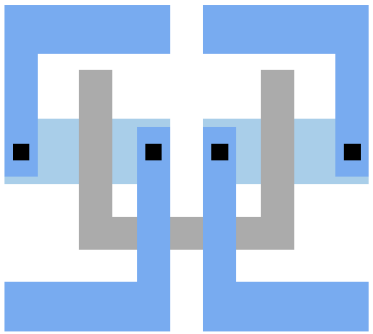
UNIVERSITÄT
HEIDELBERG

„Preprocessed OPIs“



Michael Ritzert

michael.ritzert@ziti.uni-heidelberg.de

VXD SC Telco

17.09.2015

Introduction

- Purpose:
  Work around the „macros cannot be redefined“ problem in CSS.
- At the same time, provide stable link targets for entry to the OPI hierarchy from e.g. the alarm system.

- Proposal:
  „Preprocess“ skeleton OPIs by statically adding the macro definitions to the OPI:
  selector_rates.opi becomes selector_rates@O02S1.opi, selector_rates@O02S2.opi, etc.
- Provide „make“ rules to automatically run the conversion.

Calling Side

```
<action type="OPEN_DISPLAY">
 <path>basic.opi</path>
  <macros>
  <include_parent_macros>↵
    true</include_parent_macros>
    <DEVICE>O01M1</DEVICE>
    <OPI_FILE>merger_rates.opi↵
      </OPI_FILE>
  </macros>
  <replace>0</replace>
  <description/>
</action>
```

```
<action type="OPEN_DISPLAY">
  <path>merger_rates@O01M1.opi</path>
  <macros>
    <include_parent_macros>↵
      true</include_parent_macros>
  </macros>
  <replace>0</replace>
  <description/>
</action>
```

merger_rates.opi vs. merger_rates@O01M1.opi

```
<macros>
  <include_parent_macros>true↵
    </include_parent_macros>
  <FWTYPE>Merger</FWTYPE>
</macros>
```

```
<macros>
  <include_parent_macros>true↵
     </include_parent_macros>
  <FWTYPE>Merger</FWTYPE>
  <INPUT2>DC</INPUT2>
  <INPUT1>HLT</INPUT1>
  <OUTPUT1>O02S1</OUTPUT1>
  <OUTPUT2>O02S1</OUTPUT2>
  <TOP>ONSEN</TOP>
  <DEVNAME>Merger</DEVNAME>
  <NEXT>O01M1</NEXT>
  <IPMIID>I123</IPMIID>
  <DEVICE>O01M1</DEVICE>
  <PREV>O01M1</PREV>
</macros>
```

Conversion Flow

- Creating XSLT instructions to put the right macros for each device.
  - Created from create_system_dyn and lut_macros in the case of ONSEN.
- Python-Script to run the conversion.
- Input: Name of a top-level OPI.
  - Scans for linked OPIs from there.
    - Builds the names when OPI_FILE and DEVICE macros are used.
  - Puts each linked OPI in the list of OPIs to create.
  - Uses XSLT for the DEVICE to convert the linked OPI.
- Works fine as long as the links are in the OPI file itself.
- When scripts create the links, the process has to be restarted to consider these links.
  (For ONSEN, I modified create_system_dyn.**py** to print the links.)

O01M1.xslt

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:output method="xml"/>
    <xsl:template match="node()|@*">
        <xsl:copy>
            <xsl:apply-templates select="node()|@*"/>
        </xsl:copy>
    </xsl:template>

    <xsl:template match="/display/macros">
        <xsl:copy>
           <xsl:apply-templates select="node()|@*"/>
           <INPUT2>DC</INPUT2>
           <INPUT1>HLT</INPUT1>
           <OUTPUT1>O02S1</OUTPUT1>
           <OUTPUT2>O02S1</OUTPUT2>
           <TOP>ONSEN</TOP>
           <DEVNAME>Merger</DEVNAME>
           <NEXT>O01M1</NEXT>
           <IPMIID>I123</IPMIID>
           <DEVICE>O01M1</DEVICE>
           <PREV>O01M1</PREV>
       </xsl:copy>
    </xsl:template>
</xsl:stylesheet>
```

Open Items

- Is it possible to re-write python-scripts included in OPIs to allow running the from the shell.
  - Add #!/usr/bin/python
  - On start, check if sys.argv is defined, and change mode if yes.
  - Commands
    - Print list of link targets that are dynamically created.
    - Print list of macro definitions passed into an OPI.
- Fix and finalize the python converter script. Possibly add another wrapper around it.
- How to design the OPIs?

Thank you!