



The
PYTHIAN
GROUP™



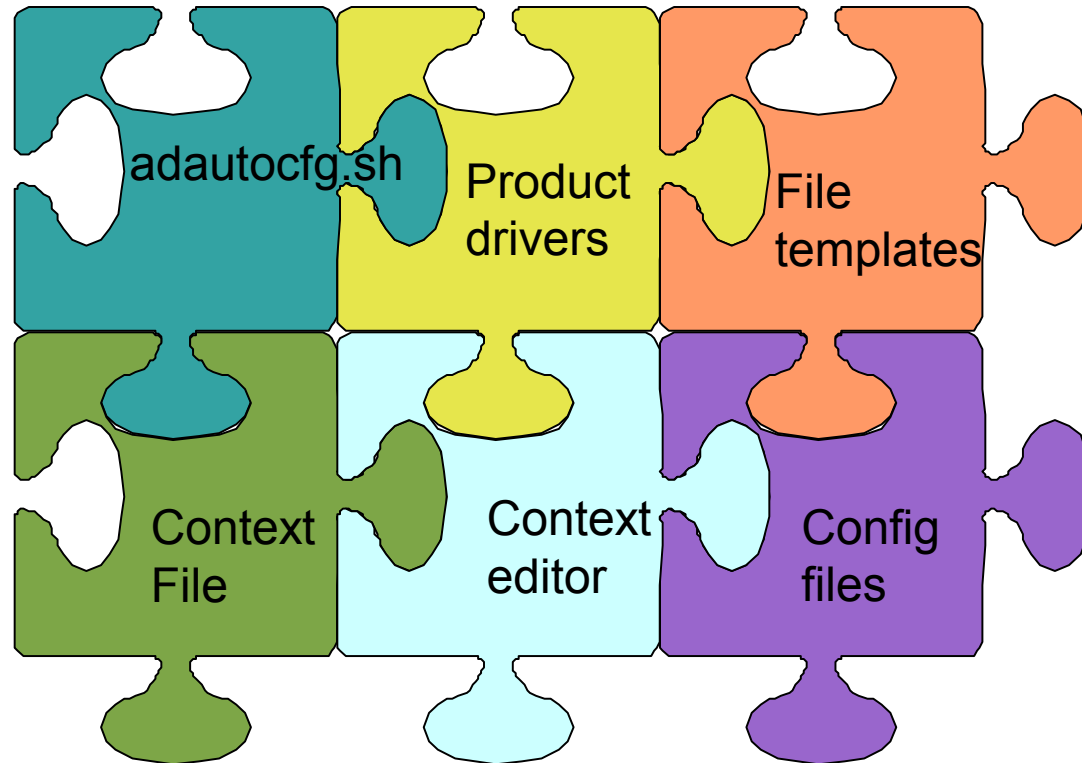
Autoconfig customization & best practices

By Lukas Vysusil, Applications DBA

Topic introduction

- Putting autoconfig into context
- Down into the template customization
- Maintaining customized templates
- Tools and features
- Best practices

How it fits together?



Why customize?

- Integration of custom products
- Integration of 3rd party products
- Integration of Oracle products
- Special needs and workarounds
- Automation



Degrees of customization

- Light customization
modification of existing file template
- Medium customizations
adding your own custom variables or new templates
- Heavy customizations
create your own custom products, driver files, templates and context variables

Answer for yourself before starting

What is the scope of customization?

Add reference to a startup/shutdown script, for our custom service.

Is there a need for new context variables?

No, we will reuse existing ones.

What context variables will I use?

s_contextname, s_logdir, s_inst_base

What template files will I need to modify?

`$AD_TOP/admin/template/adstpall_ux.sh`

`$AD_TOP/admin/template/adstrtal_ux.sh`

Customizing templates step #1

- Copy files which will be modified to the custom directory:

```
[applmgr@blackbox ~]$ cd $AD_TOP/admin/template
[applmgr@blackbox template]$ mkdir custom
[applmgr@blackbox template]$ cp adst*_ux.sh custom
[applmgr@blackbox template]$ cd custom
[applmgr@blackbox custom]$ ls
adstpall_ux.sh  adstrtal_ux.sh
```

Where should I put my customization?

Never edit files directly in:

- \$PRD_TOP/admin/template



Always edit copy of the templates in:

- \$PRD_TOP/admin/template/custom

Building custom block step #2

- Example of path translation into context variables

PATH which we need to translate

```
d01/PVISII/inst/apps/PVISII_blackbox/admin/scripts
```

Context variable we can reuse:

```
<INST_BASE oa_var="s_inst_base" d01/PVISII/inst</INST_BASE>
```

```
<oa_context_name
```

```
oa_var="s_contextname" PVISII_blackbox</oa_context_name>
```

Rewriting the path into path with context:

```
%s_inst_base%/apps/%s_contextname%/admin/scripts/xxstart_myservice.sh
```

Customizing template step #2

- Append your custom block to the template(s)

```
$AD_TOP/admin/template/custom/adstrtal_ux.sh:
...
#Beginning of a custom section
CUSTOM_LOGFILE=%s_logs_dir%/xxstart_myservice.log
  printf "Starting custom services\n"

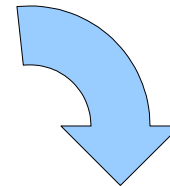
%s_inst_base%/apps/%s_contextname
%/admin/scripts/xxstart_myservice.sh  >> $CUSTOM_LOGFILE

if [ $exit_code -eq 0 ];then
  printf "Custom service for this node is started.\n";
fi
  printf "Check logfile $CUSTOM_LOGFILE for details\n\n"
#End of a custom section
...
```

Begin/End Customizations block

\$AD_TOP/admin/templates/custom/adovars_ux.env

```
...  
#BEGIN CUSTOMIZATIONS  
MY_CUSTOM_VARIABLE=%s_dbSid%  
export MY_CUSTOM_VARIABLE  
#END CUSTOMIZATIONS  
MY_CUSTOM_VARIABLE=%s_dbSid%  
export MY_CUSTOM_VARIABLE
```



adautoCfg.sh

\$APPL_TOP/admin/adovars.env

```
...  
MY_CUSTOM_VARIABLE=PVISII  
export MY_CUSTOM_VARIABLE  
#BEGIN CUSTOMIZATIONS  
MY_CUSTOM_VARIABLE=%s_dbSid%  
export MY_CUSTOM_VARIABLE  
#END CUSTOMIZATIONS
```

Customizing templates step #3

- Run adcheckcfg.sh

```
[applmgr@blackbox ~]$ $AD_TOP/bin/adchkcfg.sh
Enter the full path to the Applications Context file:
/d01/PVISII/inst/apps/PVISII_blackbox/appl/admin/PVISII_blackbox.xml
Enter the APPS password:
...
Differences text report is located at:
/d01/PVISII/inst/apps/PVISII_blackbox/admin/out/11261943/cfgcheck.txt
```

- Check the difference report

```
[applmgr@blackbox ~]$ cd /d01/PVISII/inst/apps/PVISII_blackbox/admin/
out/11261943
[applmgr@blackbox template]$ cat x_*
```

Customizing templates step #3

- Verify that context evaluated properly:

```
Checking differences between
Existing file(<) : /d01/PVISII/inst/apps/PVISII_blackbox/admin/scripts/adstrtal.sh
New file(>)      : /d01/PVISII/inst/apps/PVISII_blackbox/admin/out/11262025/adstrtal.sh

=====
Differences
=====
201a201
>
> #Beginning of a custom section
> CUSTOM_LOGFILE=/d01/PVISII/inst/apps/PVISII_blackbox/logs/xxstart_mysevice.log
>   printf "Starting custom services\n"
>
> /d01/PVISII/inst/PVISII_blackbox/admin/scripts/xxstart_mysevice.sh &gt;&gt;
$CUSTOM_LOGFILE 2&gt;&1
>
> if [ $exit_code -eq 0 ];then
>   printf "Custom service for this node is started.\n";
> fi
>   printf "Check logfile $CUSTOM_LOGFILE for details\n\n"
> #End of a custom section
```

* as you can see adcheckcfg.sh is not perfect

Customizing templates step #4

- We're ready to go:

```
[applmgr@blackbox ~]$ cd $ADMIN_SCRIPTS_HOME
[applmgr@blackbox scripts]$ adautocfg.sh
Enter the APPS user password:

The log file for this session is located at:
/d01/PVISII/inst/apps/PVISII_blackbox/admin/log/11261540/adconfig.log
...
```

- Check log, check resulting file, test out the customized script

What tools do we have? (in ADX.F)

- `$AD_TOP/bin/adtmplreport.sh`

List all templates and target files

```
./adtmplreport.sh
  contextfile=/d01/PVISII/inst/apps/PVISII_blackbox/appl/admin/PVISII_blackbox.xml -verbose
```

List all files with customizations

```
./adtmplreport.sh
  contextfile=/d01/PVISII/inst/apps/PVISII_blackbox/appl/admin/PVISII_blackbox.xml listcustom -verbose
```

List template for specific target file

```
./adtmplreport.sh
  contextfile=/d01/PVISII/inst/apps/PVISII_blackbox/appl/admin/PVISII_blackbox.xml target=$ADMIN_SCRIPTS_HOME/adstrtal.sh -verbose
```



More tools?

- `$AD_TOP/bin/adcustomizer.sh`
 - It migrates customizations to new templates

```
./adcustomizer.sh
contextfile=/d01/PVISII/inst/apps/PVISII_blackbox/appl/admin/PVISII_blackbox.xml
```

- **Caveat**
 - this script migrates only BEGIN/END CUSTOMIZATIONS blocks, which as we've learned, does not evaluate context variables

Keep in mind



- Any patch has a potential to deliver new version of a template
- Adpatch will warn you, BUT migration to new template is up to you

Correct way to customize

- Template files of the **same version** should be generic
- You can migrate custom templates between instances
- Cloning should not break customization if it's written correctly

Autoconfig etiquette

- Never edit configuration files directly (especially in multi-DBA environment)
- Always run `adchkcfg.sh` first
- Review and correct the differences
- Avoid manual modification of context file
- Use OAM context editor (OAM.H)
- Know how to revert changes (`restore.sh`)

Questions?





The
PYTHIAN
GROUP™



Thank you!

Lukas Vysusil: vysusil@pythian.com

Backup slides

- This box is empty.

Resources

Subject: Customizing an AutoConfig Environment

Doc ID: Note:270519.1

Subject: Using AutoConfig to Manage System Configurations in Oracle E-Business Suite Release 12

Doc ID: Note:387859.1

Subject: Autoconfig FAQ

Doc ID: Note:218089.1

R12 Autoconfig phases

- # There are five phases which autoconfig recognizes and performs in order.
They are in order
INSTALL (Instantiate the template during Install)
=> if the file exists, do not replace it
- # INSTE8 (To instantiate files)
=> create new files from templates, files marked as INSTALL in driver files will not be instantiated
- # INSTE8_SETUP (Instantiate and run in setup phase)
=> executing scripts after instantiation phase
- # INSTE8_PRF (Instantiate and run in profiles phase)
=> profiles phase mostly consists of running scripts to update profile options
- # INSTE8_APPLY (Instantiate and run in apply phase)
Services phase
=> updating of AD_APPL_TOPS table with accurate information