Building YaST Modules

Thomas Yeung & Stanislav Visnovsky
Software Engineers
Novell, Inc.

Novell®

Contents

Novell.

What is YaST?

YaST (Yet another Setup Tool) is installation and configuration system

Features:

- modular design
- user interface independence
- hardware architecture independence
- incorporated knowledge about SUSE Linux-based systems

Novell.

Why YaST?

More user friendly interface (both text and graphics modes) to installation and configuration on Linux

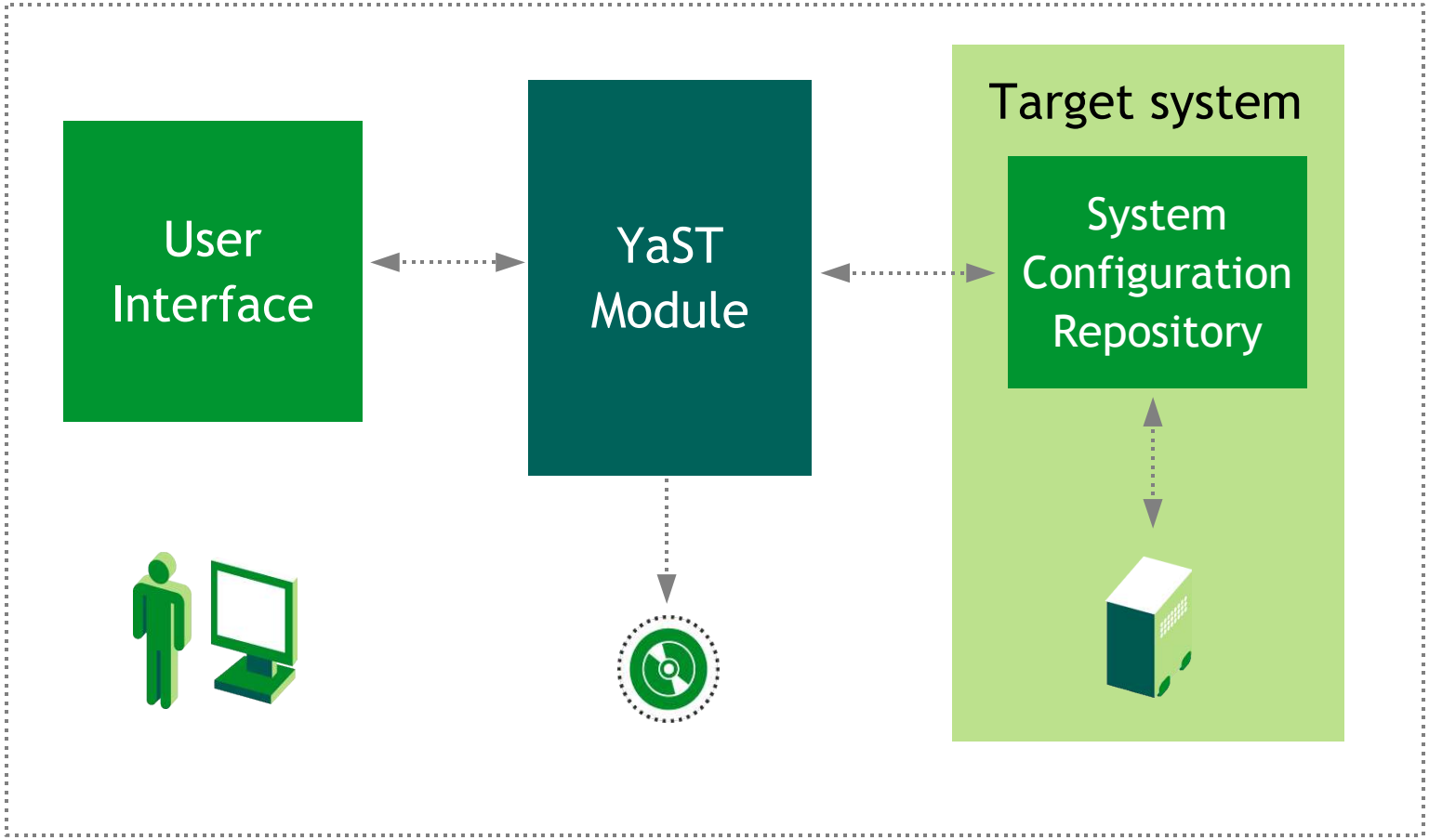
A configuration task is accomplished by a predefined workflow

Contains powerful scripting language for rapid development

Provides system abstraction

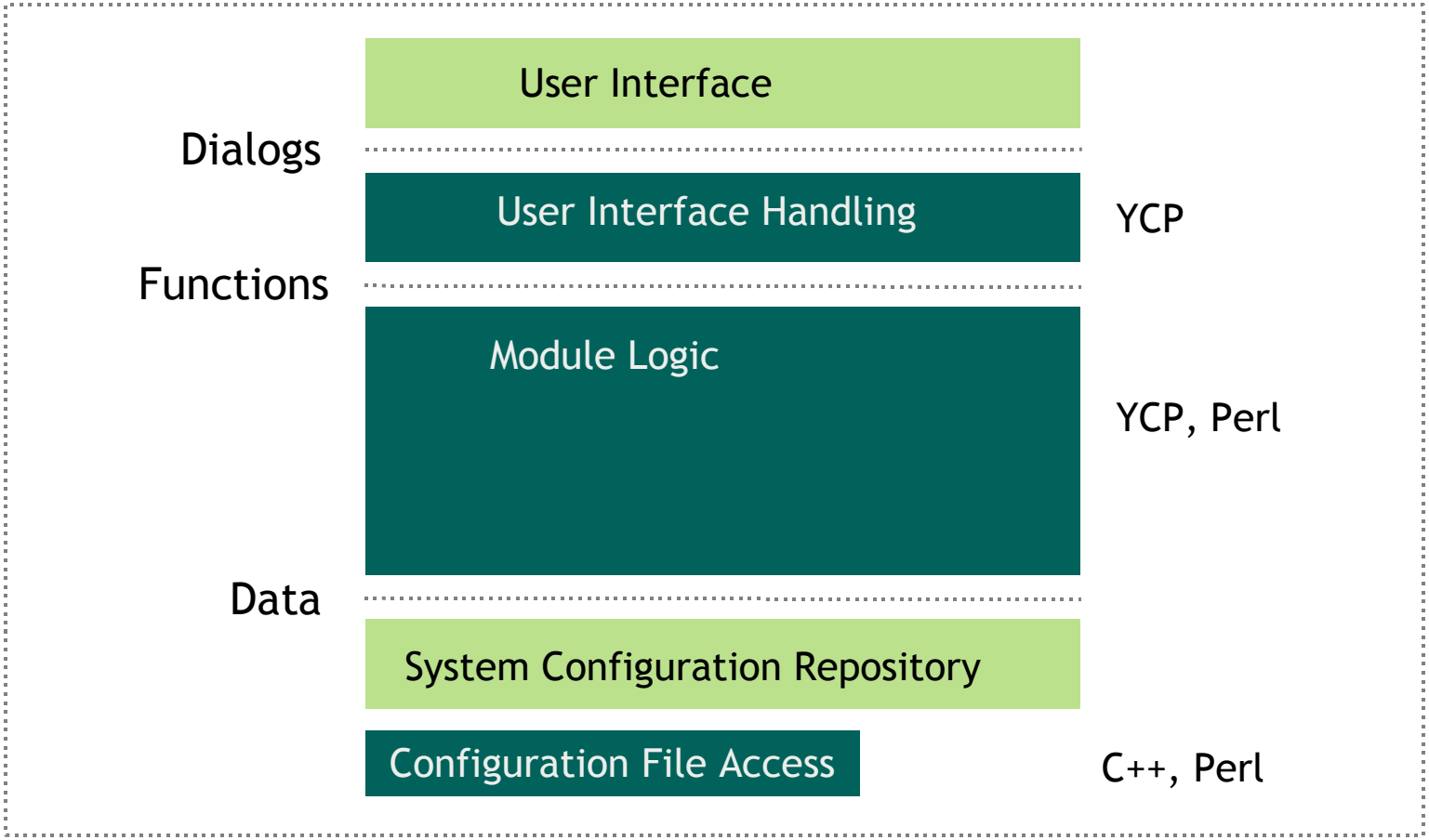
Novell.

Overall architecture



Novell.

YaST Module



Novell.

Programming Languages

Programming languages

YaST supports multiple programming languages

- "Choose the right language" approach
- YCP
  - YaST "native" scripting language
- Perl

Novell.

YCP

YCP = YaST Control Protocol

“Native” YaST programming language

Used for development of user interface and logic

Features:

- C-like interpreted scripting language (can be precompiled)
- modular, imperative
- statically typed
- full range of types tailored for configuration manipulation
- default values

Novell.

“Hello, World!” in YCP

```
/**
 * File:    client/hello.ycp
 * Module:  hello world configuration
 * Summary: display hello world message
 * Author:  tyeung <tyeung@novell.com>
 * $Id: hello.ycp
 */
{
     string message = “Hello, World!”;

     UI::OpenDialog(
          `Vbox(
                    `Label( message ),
                    `PushButton(“&OK”)
               )
          );
     UI::UserInput();
     UI::CloseDialog();
}
```


Novell.

ycp compiler

ycpc is a ycp bytecode compiler

ycpc -Eq hello.ycp

- Check YCP syntax and don't print any progress

ycpc -c hello.ycp

- Compile the ycp file into a byte code file

Novell.

Lab Environment Setup

SUSE 9.2 running on VMWare

Installed with the following packages:

- yast2-devtools – automake/autoconf, helper scripts
- yast2-core-devel – documentation for system agents
- yast2-devel – library documentation
- yast2-testsuite – support for automatic testsuites

Novell.

Lab 1

Create a Hello World ycp program

1

Novell.

y2tool

Use y2tool to create a skeleton framework:

y2tool create-new-package [-dsv] <skeleton> <name> <maintainer> <email>

skeleton – the one which should be used (config, trans, ...)

name – of the components. A package name will be constructed like yast2-skeleton-name

maintainer – his name

email – maintainer's email

Novell.

y2tool

For example:

y2tool create-new-package config test tyeung tyeung@novell.com

Novell.

Lab 2

Use y2tool to create a skeleton project.

Examine the files and directory structure of the skeleton project.

2

Novell.

YCP Data Types

- void
- symbol
- boolean
- integer
- float
- string
- byteblock
- list
- map
- term
- path
- block
- symbol
- any

Novell.

YCP Operators

Comparison Operators - ==, <, >, <=, >=, !=

Boolean Operators - &&, ||, !

Bit Operators - &, |, ~, <<, >>

Math Operators - +, -, *, /, %, unary -

Triple Operator – (condition ? expression : expression)

Novell.

Conditional Loops

If (condition) then_part [else else_part]

while() loop

do..while() loop

repeat..until() loop

break

continue

return

Novell.

Include and Import Statements

The include statement allows you to insert contents of a file at the given place in the current file. If the current file is a module, the contents of the included file will become a part of the module.

The import statement is liked the java import statement. It imports a namespace to the file.

Novell.

User Interface

User Interface

- interactive user interface
- same code for textual and graphical user interface
- predefined set of widgets
- using libyui library to provide the UI abstraction layer
- intelligent automatic layout management
- simplified event handling

UI Example

```
`HBox(
      `HWeight( 30, `RichText( "Help text") ),
      `HWeight( 70, `VBox(
                        ...   // the dialog contents
                        `HBox(
                        `PushButton( "Back"),
                        `HCenter( `PushButton( "Abort
                                 Installation") ),
                        `PushButton( "Next")
                             )
                          )
                )
      )
```

The YaST2 Event Model

Typical UI Event Model uses one central event loop and lots of callbacks

With the YaST2 Event Model, the flow control remains in the interpreted YCP code

This of course means that there is no single one central "waiting point" in the program (like the event loop in the event-driven model), but rather lots of such waiting points spread all over the YCP code within each UserInput() or WaitForEvent() statement.

Novell.

UI Events Example

```
{
    UI::OpenDialog(
            `VBox(
                ... // Some input fields etc.
                `HBox(
                    `PushButton(`id(`back ), "Back" ),
                    `PushButton(`id(`next ), "Next" )
                    )
                )
            );
```

UI Events Example (cont.)

```
    symbol button_id = UI::UserInput();
    if ( button_id == `next )
    {
        // Handle "Next" button
    }
    else if ( button_id == `back )
    {
        // Handle "Back" button
   }
   UI::CloseDialog();
}
```


Novell.

Lab 3

Create the UI part for the sampled ycp program

3

Novell.

System Configuration Repository (SCR)

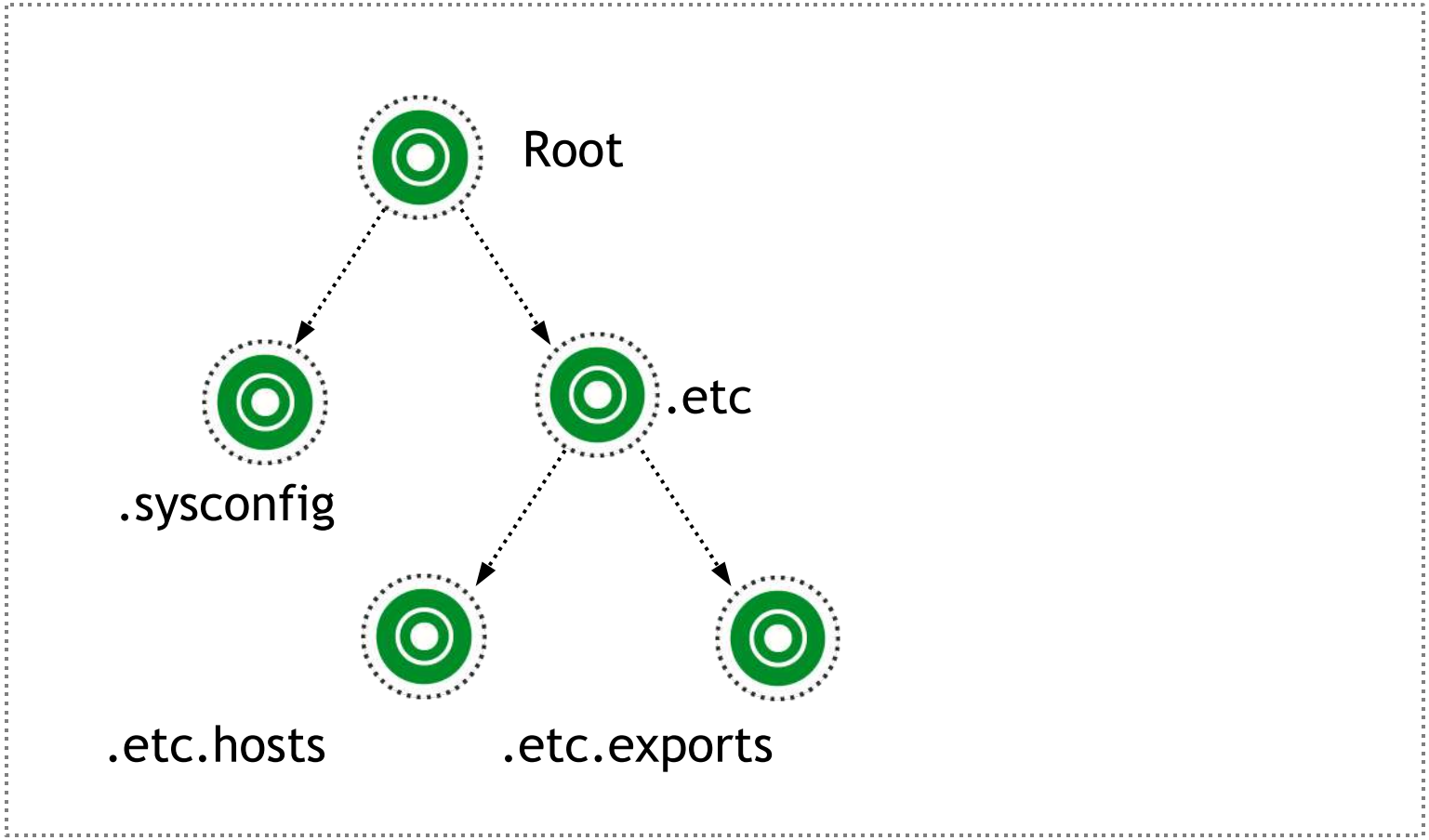
SCR provides a consistent view of the system hardware and its configuration files.

It provides an abstraction of various type of data to be handled

SCR consists of a number of agents, each of them being specialized on a specific task

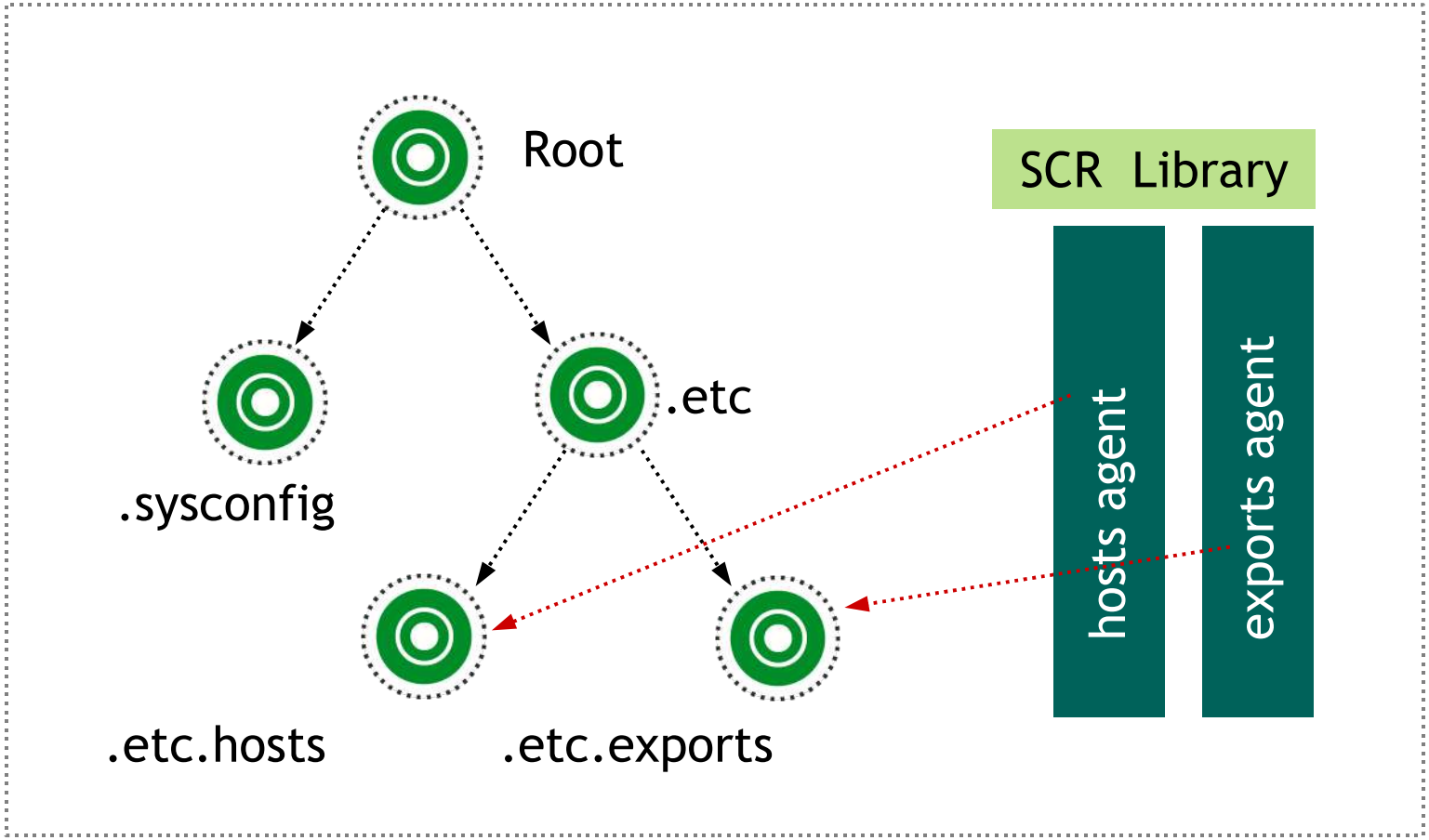
Novell.

System Abstraction



Novell.

SCR Architecture



Novell.

SCR Methods

Reading

- SCR::Read (path)

Writing

- SCR::Write (path, value)

Executing actions

- SCR::Execute (path, value)

Listing contents

- SCR::Dir (path)

Novell.

Standard SCR Access

Generic agents

- line-oriented configuration files (INI-agent)
- sysconfig agent

System agents (.target)

- Execution of commands
- Temporary files
- file/directory status

Novell.

Lab 4

Using SCR library to modify a configuration file

4

Novell.

Debugging

Logging

Different outputs for debugging:

- y2error() – error messages
- y2warning() – warning messages
- y2debug() – debugging messages

If root:

- See /var/log/YaST2/y2log

If non root:

- See $HOME/.y2log

Full Debugging

- Set “export Y2DEBUG=1” in your .profile
- Run “Y2DEBUG=1 yast2”

Novell.

References

yast2.suse.com

yast2-hacker@suse.de

Coolsolutions

Attend DL101 - YaST and YaST Developement

Novell.

Novell®

Novell