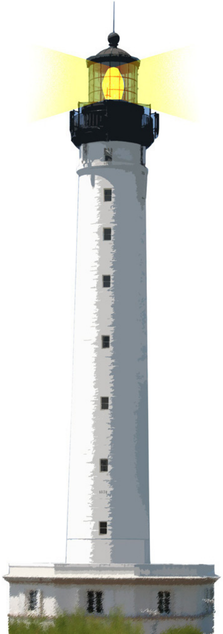
Runtime Architecture

Damien Cassou, Stéphane Ducasse and Luc Fabresse

W6S06

http://www.pharo.org



Execution Model

Pharo virtual machine (VM) executes compiled code

- The virtual machine and its plugins are platform specific (different versions for different OSes)
- VMs exist for MacOS, Windows, Linux (different versions), iOS, ARM, Android

Multiple Stage Compilation

1. Pharo code is compiled to bytecodes (platform neutral instructions)
2. The virtual machine dynamically transforms bytecodes to assembly

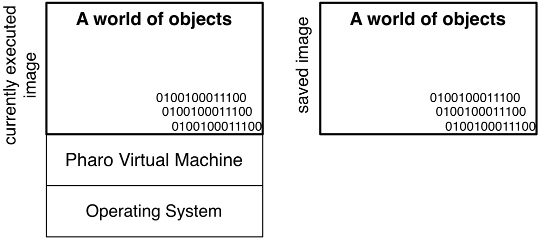
Virtual Machine

- Pharo.exe, Pharo.app... are the virtual machines
- There are two modes:
  - from command-line or in interactive (UI) mode
- It executes compiled code / generates on the fly assembly
- Compiled code is packaged/stored in an image (memory snapshot)
- The virtual machine only needs the image to execute programs

Image Files: Memory Snapshots

.image files is a cache of objects:

- Simple objects (points, strings ...)
- But also **compiled** classes and **compiled** methods
- Each time we save the image, all objects are saved to disc
- At startup we get back all the objects we saved
- PC (program counter) is also saved and restored
    - frozen execution is restarted at launch time



Change Files: Change Tape

.changes file is a tape of all the changes performed to the system

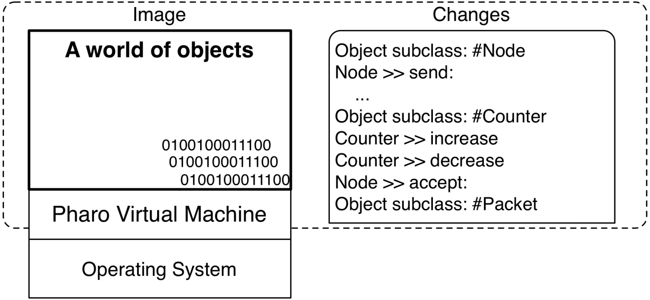
- Logs class creation/deletion, method addition/removal, actions...
- Used to browse versions
- Can replay/undo actions

A change is associated to an image

- To display class/method definition, tools look in the changes file associated to the current image

Image/Change Files

- A change is associated to an image
- Image contains all the objects in binary form. Can be executed without the changes file
- Changes file simply contains the textual representation of the changes made to the image



Image

A world of objects

0100100011100
0100100011100
0100100011100

Pharo Virtual Machine

Operating System

Changes

```
Object subclass: #Node
Node >> send:
   ...
Object subclass: #Counter
Counter >> increase
Counter >> decrease
Node >> accept:
Object subclass: #Packet
```

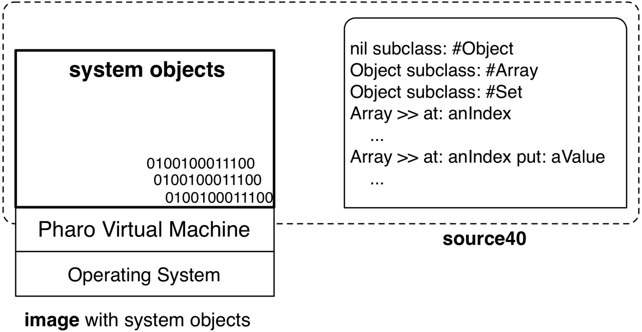
Save your code using a package and version control system

- Change and image are handy to develop
- But **they are not a software engineering artefact**
- Always have a loading script that takes an image, load your code, run the tests, build your application
- Usually
  - save code using a Version Control System (monticello, git)
  - use an integration server to build automatically applications

About the Source/Changes Files

PharoXX.sources

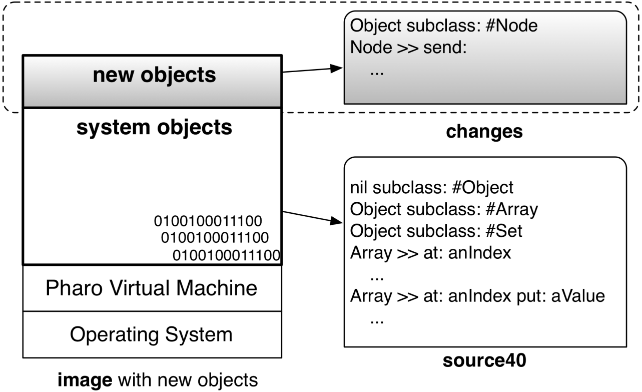
- Contains the textual definition of **system** classes and predefined objects
- Is read-only
- Created during release of new Pharo versions
- Shared to all the users (images)



When you Define New Classes

During development or code loading

- New objects are compiled in the image
- New definitions are added to the changes file
- Still you can browse the definition of the system class (stored in the PharoXX.sources)



Change Management

- Tools>Code Changes
  - relies on the changes file and the recording mechanism
  - support replay changes
- Tools>Iceberg
  - Integration with Git and other modern distributed version control systems
- New ways to produce images (e.g. Bootstrapping)

Conclusion

- Powerful deployment
- Fast boot-time
- Support micro commits
- Modern version control

A course by

Inria informatiques mathématiques

and

UTOP
Université de Technologie
Ouverte Pluripartenaire

in collaboration with

uni sciel

Fondation unit
Université Numérique
Ingénierie et Technologie

Université
de Lille

IMT Lille Douai
École Mines-Télécom
IMT-Université de Lille