Formal Methods for Java
Lecture 20: Java Pathfinder
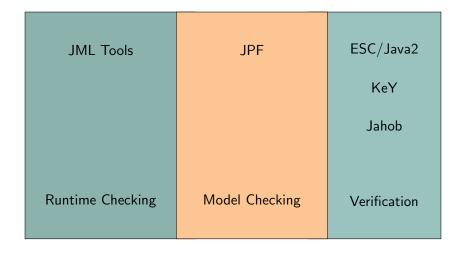
Jochen Hoenicke

Software Engineering
Albert-Ludwigs-University Freiburg

July 10, 2017

Bridging the Gap

JML Tools	JPF	ESC/Java2
		KeY
		Jahob
Runtime Checking	Model Checking	Verification

Java Pathfinder (JPF)

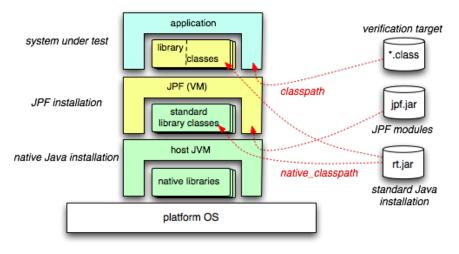**JPF** *.. the swiss army knife of Java™ verification*

http://babelfish.arc.nasa.gov/trac/jpf/wiki

- Developed at NASA Ames Research Center
- One tool – many different usage patterns
- Highly extensible core
- Core implements explicit state model checking on top of a Java VM
- Key concepts:
    - Execution choices as transition breakers
    - State matching
    - Backtracking (restoring previous state)
    - Listeners, Properties, and Publishers

History of JPF

1999 Start as front end for the Spin model checker.

. . . .

2000 Reimplementation as virtual machine

2003 Extension interfaces

2005 Open sourced on Sourceforge

since 2008 Participation in Google Summer of Code

since 2009 Project, extensions, and wiki hosted on NASA servers (still open source)

Obtaining and Building JPF

- Download from `http://babelfish.arc.nasa.gov/trac/jpf`
- Recommendation: use Mercurial repositories

  > `hg clone http://babelfish.arc.nasa.gov/hg/jpf/jpf-core`

- Repository contains everything needed to build jpf-core

  > `ant`

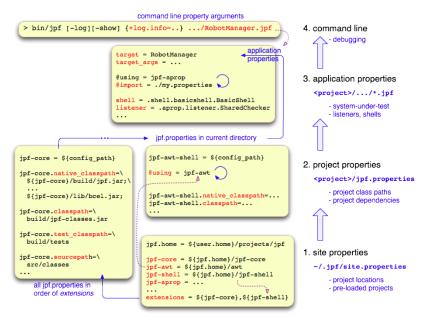- Instructions for Eclipse or NetBeans can be found in the JPF wiki

What We Got

http://babelfish.arc.nasa.gov/trac/jpf/wiki

VM Inside a VM?

- JPF is written in Java $\Longrightarrow$ runs on a JVM
- JPF interprets Java Bytecode $\Longrightarrow$ acts as a JVM
- JPF operates differently:
    - Bytecode of System under Test (SUT) and
    - SUT-specific Configuration produce
    - a report and (possibly) some other artefacts (e.g., test cases)
- JPF might terminate the application if a property is violated

How to Configure JPF

JPF Configuration

July 10, 2017   9 / 13

JPF Configuration Files

- Basically Java properties files:
    - key=value assigns value to key
    - # This is a comment
- Extensions:
    - ${x} expands to current value of variable x
    - key+=value appends value to the value of key
      (No space between key and +=)
    - +key=value prepend value to the value of key
    - ${config_path} expands to the directory of the currently parsed file
    - ${config} expands to the filename of the currently parsed file
    - @using=<project-name> loads project project-name from location
      defined in site.properties with line
      <project-name>=<project-path>
    - ...
- Shortcut for class names: package prefix $gov.nasa.jpf$ can be omitted
- Configuration of JPF can be difficult

Configuring Our Compiled Version

- Switch to your home directory
- Create folder `.jpf`
- Create file `.jpf/site.properties`

  ```
  jpf.home = <Path where you downloaded jpf>

  jpf-core = ${jpf.home}/jpf-core

  extensions = ${jpf-core}
  ```
- This creates the basic configuration
- Add line `jpf-proj = path` to `site.properties` for every additional project you download

Configuring SuTs

- Create configuration file (typically ends with `.jpf`)
- Content:
  - Some `@using` directives (optionally)
  - One line `target = <SuT>`
  - Optional arguments in a line `target_args = <args>`
  - Additional JPF and related project configuration (optional)
  - Optional `classpath` entry to locate the `.class` file
  - Optional `sourcepath` entry to locate the `.java` file

Demo