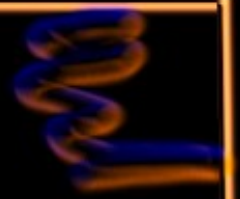
Generativní programování s podporou pro formální verifikaci

Marek Paška

Přehled

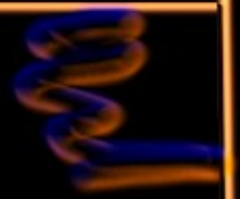


1. motivace a cíle
2. nástroje, které používám
3. optimalizace pro formální verifikaci

Software ve vestavěných zař.

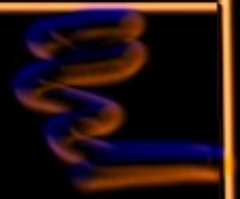
- omezené zdroje (tlak na cenu zařízení)
- spolehlivý
  - selhání může mít vážné následky
  - chyba se těžko opravuje
- reaguje na podněty z okolí
  - reaktivní
  - reálný čas

Přístup k vývoji



- desktopový / serverový software:
  - čas programátora vždy dražší než HW zdroje
  - nové a nové vrstvy abstrakce (VM, skriptovací jazyky)
- vestavěný software:
  - roli hrají výrobní náklady
  - extrémně konzervativní (assembler, čisté C)
  - Ada?
  - Java? (režie, JIT, RT)

Zbraně vývojáře

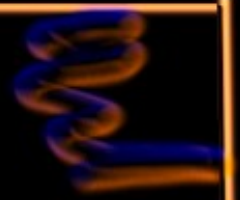


- generativní programování
  - od časů Fortranu používáme všichni
- formální metody
  - mocné ale „těžké“ - speciální nástroje
  - výsledný kód nutně nemusí mít vlastnosti ukázané na formálním modelu
- pokročilé praktiky softwarového inženýrství
  - aspektově orientované programování
  - design by contract

Idea: zlepšit vývojový proces

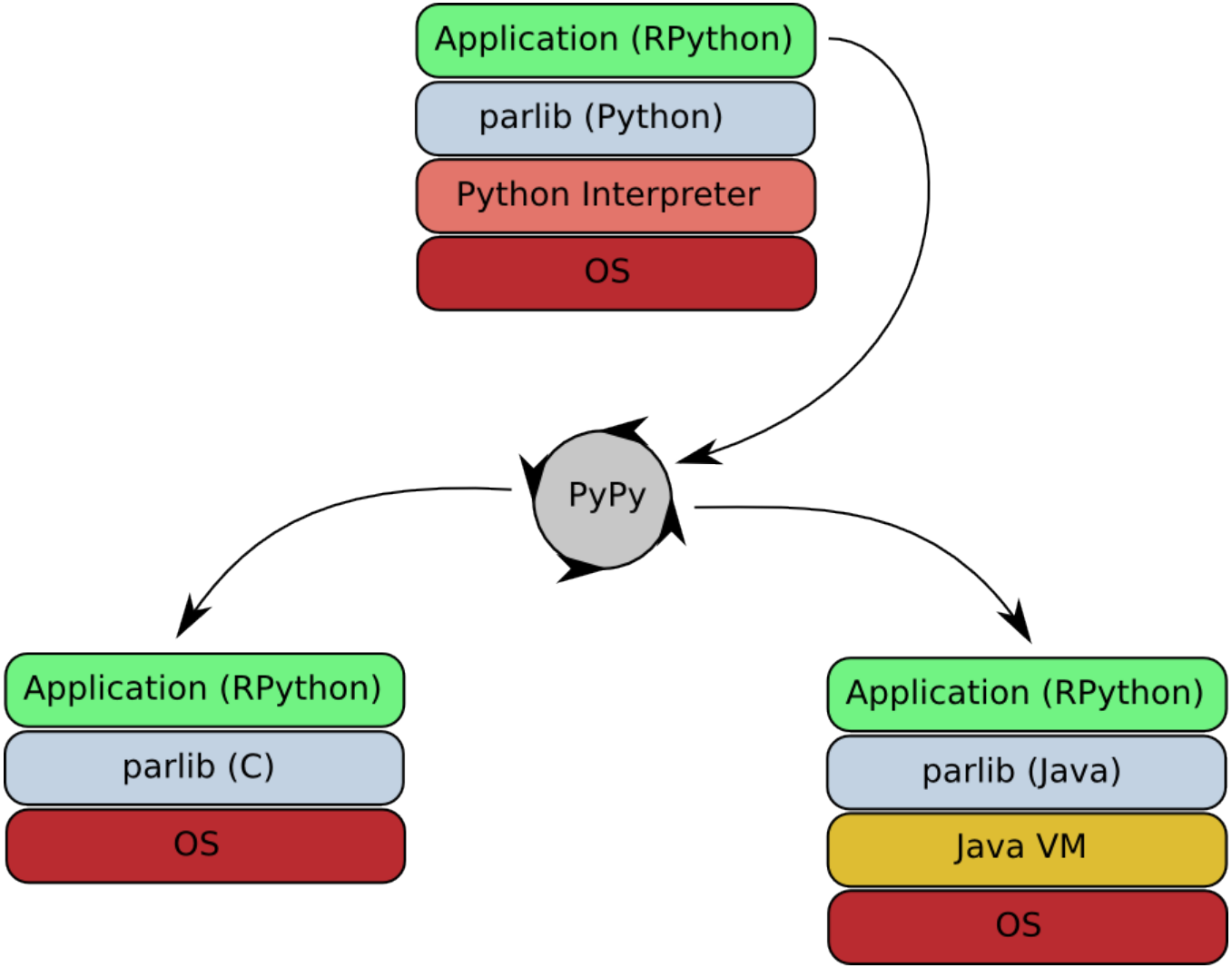
- popsat zamýšlený program v nějakém „vhodném“ jazyce
  - vhodná úroveň abstrakce
  - dostatečná vyjadřovací schopnost
  - příjemné na používání
- přátelské k formálním metodám
- efektivní kód vygenerovat

RPython



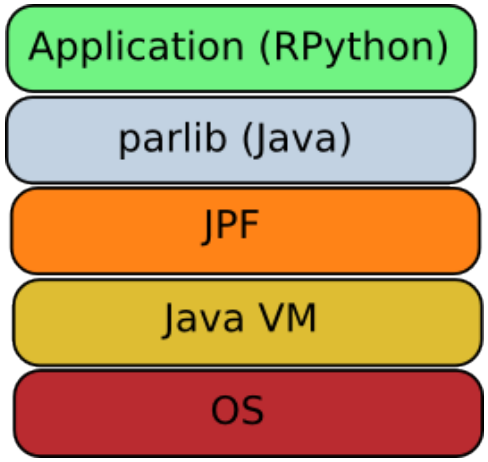
- podmnožina jazyka Python
  - příjemné programování
- výsledek projekty PyPy (ETH Zürich)
  - experimentální interpretr a překladač Pythonu
- dobré vlastnosti dynamicky typovaných jazyků
  - krátký kód (méně chyb)
  - otevřeno pro nová paradigmata (DbC, AOP)
- překlad do různých jiných kódů (C, JVM)

Schéma generování



Java PathFinder

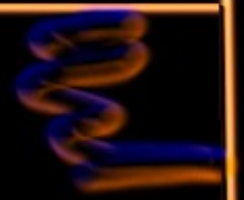
- explicitní model-checker pro Java bytecode
- JVM s backtrackingem
  - uváznutí (deadlock)
  - neodchycené výjimky
  - Linear Temporal Logic
- spíše hodně důkladné testování než formální důkaz správnosti



Nástrahy

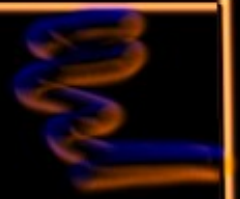
- JPF je mocný, ale citlivý
- partial order reduction
  - nesnáší souběh (i čtení)
  - optimalizované javovskou synchronizaci (monitory)

Synchronizace



- v Pythonu
  - obalené POSIXové mutexy
- v Javě
  - monitory, podpora na úrovni bytecode
- Naivní řešení: parlib implementuje zámky s POSIXovou sémantikou nad javovskými monitory
  - exploduje

Synchronizace: řešení



- vylepšit PyPy tak, aby generovalo nativní javovskou synchronizaci
  - zamčení zámku na začátku metody, odemčení na konci => synchronizovaná metoda

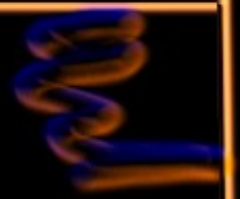
Synchronizace: benchmark

- dvě vlákna inkrementují sdílenou proměnnou chráněnou zámkem
- **public synchronized** inc()
  { **this**.counter++; }

Synchronizace: benchmark

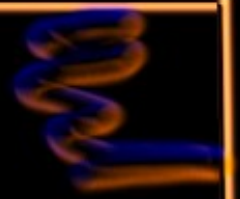
Jazyk	*Zámky*	*N*	*Čas [s]*	*Paměť [MiB]*	*Stavy*
RPython	Python	5	4	9	2939
RPython	Python	10	14	10	11649
RPython	Python	15	30	10	26159
RPython	Java	5	1	8	93
RPython	Java	10	1	8	263
RPython	Java	15	1	8	533
Java	Java	5	1	6	93
Java	Java	10	1	6	263
Java	Java	15	1	6	533

Výhled



- verifikovat Vodárnu
  - uváznutí
  - výjimky
  - LTL (jestliže je nádrž plná, pak se v budoucnu zapne čerpadlo)

Konec



- děkuji za pozornost