DiSL: instrumentation framework (language)

Lukas Marek
marek@d3s.mff.cuni.cz
Introduction

What is DiSL?
- domain specific language for java bytecode instrumentation - dynamic analysis

Dynamic analysis
- Time profiling
- Allocation profiling
- Data structure usage
- ...

State of the art
- AspectJ as “high-level” tool (framework, language)
- ASM as “low-level” tool (framework, language)
Motivation

- **Why not use AspectJ?**
  - Lack of features required for dynamic analysis
  - Lack of run-time efficiency

- **Why not use ASM?**

```java
// System.out.println("Hello world")
Type typeSystem = Type.getType(System.class);
Type typePS = Type.getType(PrintStream.class);
Type typeString = Type.getType(String.class);
instructions.add(new FieldInsnNode(Opcodes.GETSTATIC, typeSystem.getInternalName(),
                                    "err", typePS.getDescriptor()));
instructions.add(new LdcInsnNode("Hello world"));
instructions.add(new MethodInsnNode(Opcodes.INVOKEVIRTUAL, typePS.getInternalName(),"println",(" + 
                                    typeString.getDescriptor() + ")V"));
```
Goal

Expressiveness (efficiency)

Aspect-Oriented Programming (AspectJ)

DiSL

ASM

tool development effort
“Language” presentation
Why to use DiSL
Senseo - performance evaluation

Measurement settings
• geo. mean { for each DaCapo “bach” benchmark: median(15 runs) }
<table>
<thead>
<tr>
<th>Lines of code</th>
<th>AspectJ</th>
<th>DiSL</th>
<th>ASM</th>
</tr>
</thead>
<tbody>
<tr>
<td>44</td>
<td>74</td>
<td>489</td>
<td></td>
</tr>
</tbody>
</table>
Did we succeeded?

In the context of dynamic analysis

Expressiveness (efficiency)

tool development effort

Aspect-Oriented Programming (AspectJ)

DiSL

ASM
Questions?