STEP: A Framework for the Efficient Encoding of General Trace Data

Rhodes Brown, Karel Driesen, David Eng, Laurie Hendren,
John Jorgensen, Clark Verbrugge and Qin Wang

Sable Research – McGill University
http://www.sable.mcgill.ca

November 18, 2002

Outline

- The problems with current tracing methods.
- A solution: STEP
  - Framework overview
  - Example: JVMPI allocation data
  - Uses of STEP data

Problem #1 With Traces:
Formats are too event-specific

...look at the Java run-time environment:

- Virtual method dispatch
- Garbage collection
- Explicit concurrency

$\Rightarrow$ Need an openly extensible format

Problem #2 With Traces:

They are very large

- Generic compression methods are not good enough
- Specialized schemes can lead to better compression
- Often require tuning to specific data set

$\Rightarrow$ Need adjustable encoding strategies

The STEP System

STEP: a framework for encoding a variety of trace events

Main Features:

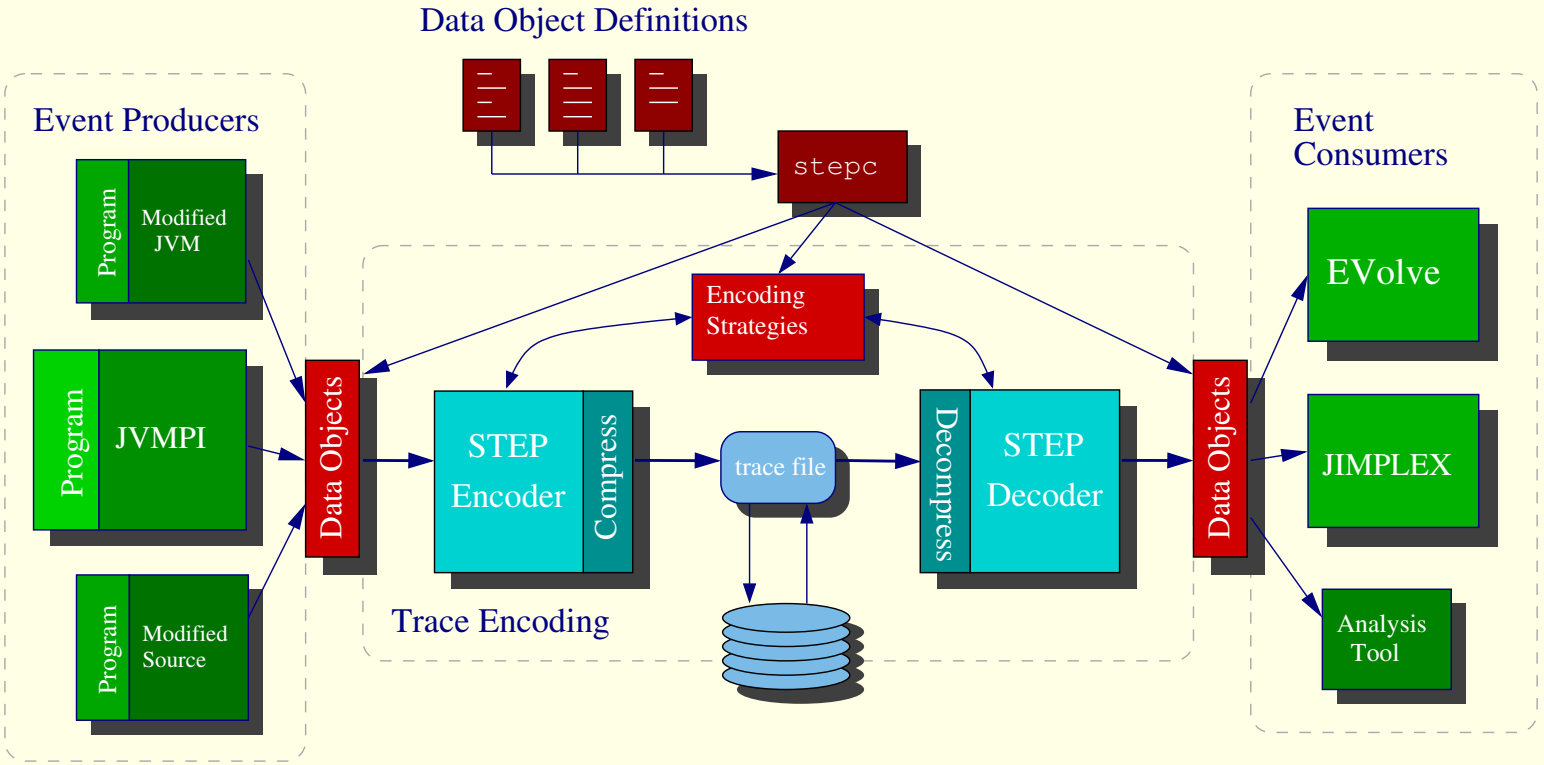
- An event definition language & compiler to generate interface elements

  $\Rightarrow$ flexibility, extensibility, & documentation

- An encoding engine to implement various reduction schemes

  $\Rightarrow$ encapsulation & compactness

The STEP Framework: Overview

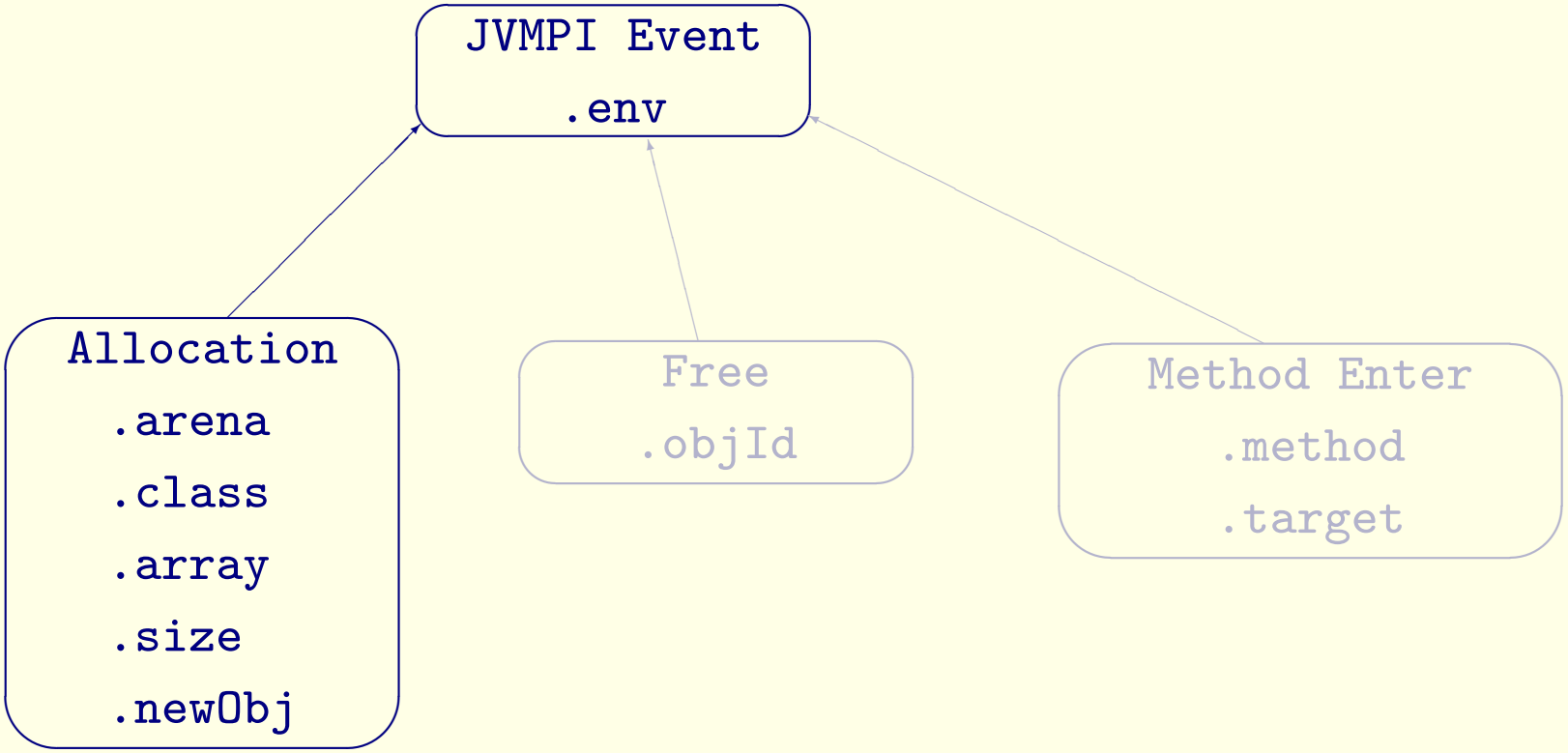


Example: Encoding JVMPI Data

Benchmark: `_213_javac`

- 6319542 allocation records
- naïve (verbatim) format: 150.67 MB
- compressed (`.gz`): 23.18 MB $\rightarrow$ 15.38%

JVMPI Data Definitions



STEP-DL for Allocation Events

```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId  <property:"address"><encoding:"size=4">;
  int isArray  <property:"unsigned">;
  int size     <property:"unsigned">;
  int newObjId <property:"address"><encoding:"size=4">;
}
```

STEP-DL for Allocation Events

```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId  <property:"address"><encoding:"size=4">;
  int isArray  <property:"unsigned">;
  int size     <property:"unsigned">;
  int newObjId <property:"address"><encoding:"size=4">;
}
```

- Structure

STEP-DL for Allocation Events

```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId  <property:"address"><encoding:"size=4">;
  int isArray  <property:"unsigned">;
  int size     <property:"unsigned">;
  int newObjId <property:"address"><encoding:"size=4">;
}
```

- Structure

- Inheritance

STEP-DL for Allocation Events

```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId  <property:"address"><encoding:"size=4">;
  int isArray  <property:"unsigned">;
  int size     <property:"unsigned">;
  int newObjId <property:"address"><encoding:"size=4">;
}
```

- Structure
- Inheritance
- Attributes

STEP-DL for Allocation Events

```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId  <property:"address"><encoding:"size=4">;
  int isArray  <property:"unsigned">;
  int size     <property:"unsigned">;
  int newObjId <property:"address"><encoding:"size=4">;
}
```

- Structure
- Inheritance
- Attributes

Baseline Comparison

	raw size	`.step` size	ratio
original	150.67 MB	102.51 MB	68.04%
compressed	23.18 MB	18.21 MB	78.56%
ratio	15.38%	17.76%	12.08%

Baseline Comparison

	raw size	`.step` size	ratio
original	150.67 MB	102.51 MB	68.04%
compressed	23.18 MB	18.21 MB	78.56%
ratio	15.38%	17.76%	12.08%

- Record size reduction

Baseline Comparison

	raw size	`.step` size	ratio
original	150.67 MB	102.51 MB	68.04%
compressed	23.18 MB	18.21 MB	78.56%
ratio	15.38%	17.76%	12.08%

- Record size reduction
- Compression ratio

Baseline Comparison

	raw size	`.step` size	ratio
original	150.67 MB	102.51 MB	68.04%
compressed	23.18 MB	18.21 MB	78.56%
ratio	15.38%	17.76%	12.08%

- Record size reduction
- Compression ratio
- Overall reduction

Remove Constant Values

```
record JVMPI_Event {
  int envId <property:"address"><encoding:"size=4">

}
```

Remove Constant Values

```
record JVMPI_Event {
  int envId <property:"address"><encoding:"size=4">
    <encoding:"repeat">;
}
```

Remove Constant Values

```
record JVMPI_Event {
  int envId <property:"address"><encoding:"size=4">
    <encoding:"repeat">;
}
```

	raw size	.step size	ratio
original	150.67 MB	78.41 MB	52.04%
compressed	23.18 MB	17.35 MB	74.87%
ratio	15.38%	22.13%	11.52%

The Identifier Technique

```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId <encoding:"identifier">;
  int isArray;
  int size;
  int newObjId;
}
```

The Identifier Technique

```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId <encoding:"identifier">;
  int isArray;
  int size;
  int newObjId;
}
```

The Identifier Technique

```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId <encoding:"identifier">;
  int isArray;
  int size;
  int newObjId;
}
```

	raw size	`.step` size	ratio
original	150.67 MB	63.58 MB	42.20%
compressed	23.18 MB	17.24 MB	74.38%
ratio	15.38%	27.11%	11.44%

The Difference Technique

```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId;
  int isArray;
  int size;
  int newObjId <encoding:"delta=1048576">;
}
```

The Difference Technique

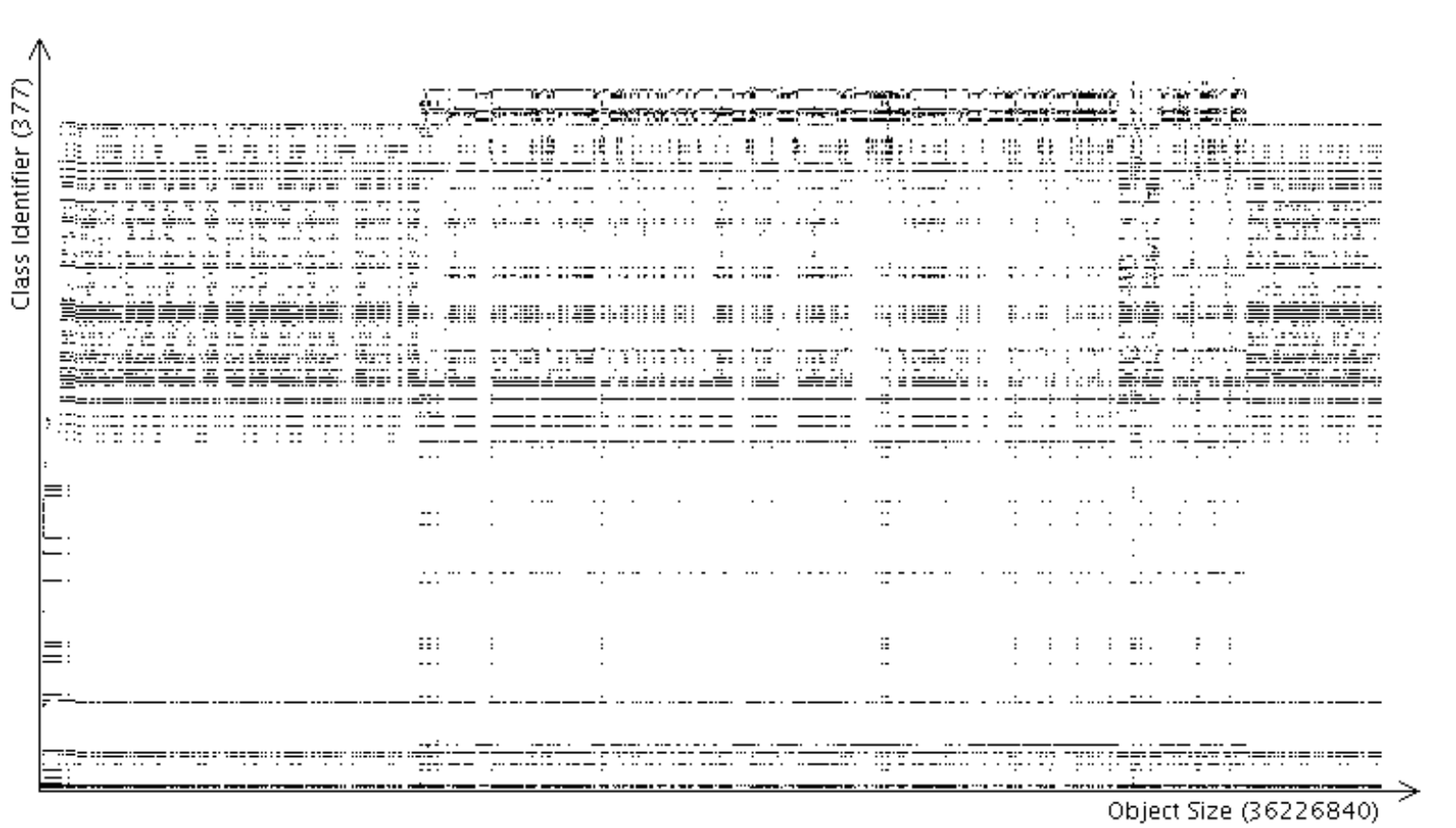
```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId;
  int isArray;
  int size;
  int newObjId <encoding:"delta=1048576">;
}
```

The Difference Technique

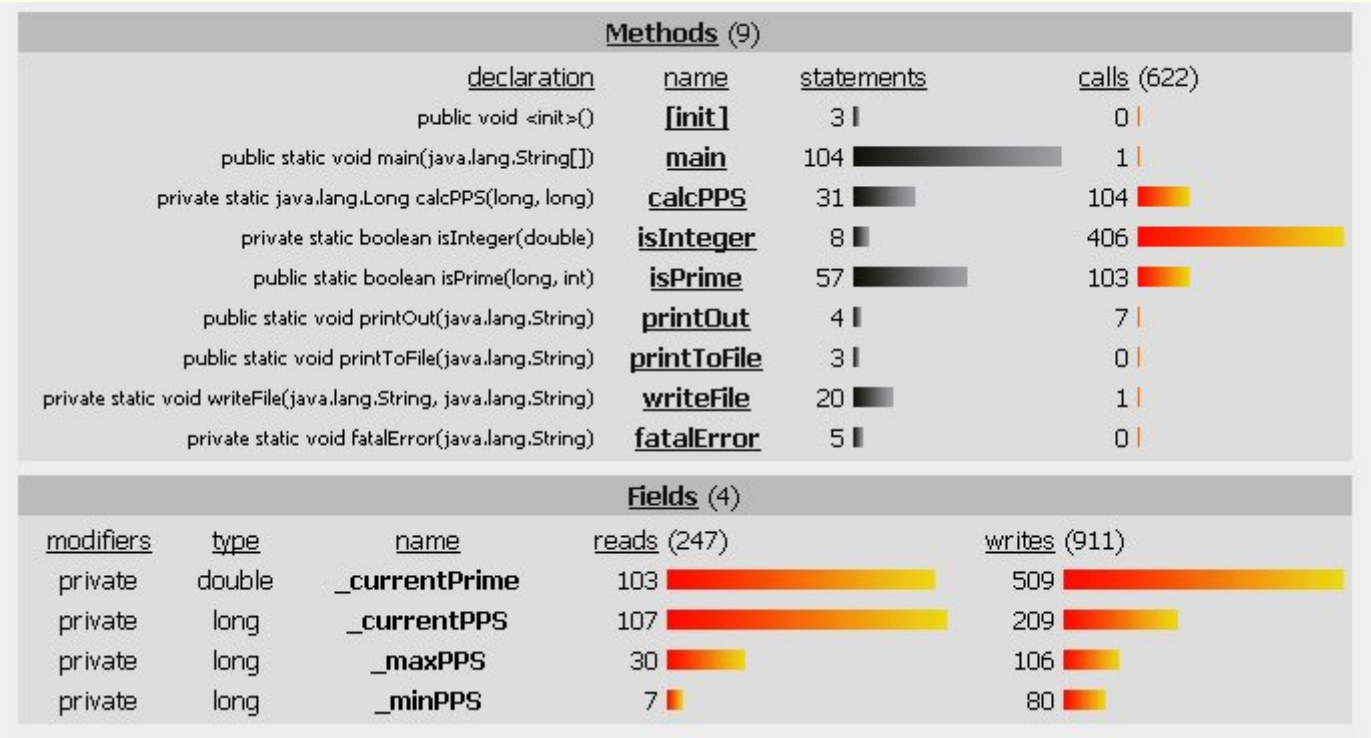
```
record OBJECT_ALLOC extends JVMPI_Event {
  int arenaId;
  int classId;
  int isArray;
  int size;
  int newObjId <encoding:"delta=1048576">;
}
```

	raw size	.step size	ratio
original	150.67 MB	46.01 MB	30.54%
compressed	23.18 MB	1.97 MB	8.49%
ratio	15.38%	4.28%	1.31%

Example Use I - EVolve



Example Use II - JIMPLEX



Methods (9)

declaration	name	statements	calls (622)
public void <init>()	[init]	3	0
public static void main(java.lang.String[])	main	104	1
private static java.lang.Long calcPPS(long, long)	calcPPS	31	104
private static boolean isInteger(double)	isInteger	8	406
public static boolean isPrime(long, int)	isPrime	57	103
public static void printOut(java.lang.String)	printOut	4	7
public static void printToFile(java.lang.String)	printToFile	3	0
private static void writeFile(java.lang.String, java.lang.String)	writeFile	20	1
private static void fatalError(java.lang.String)	fatalError	5	0

Fields (4)

modifiers	type	name	reads (247)	writes (911)
private	double	_currentPrime	103	509
private	long	_currentPPS	107	209
private	long	_maxPPS	30	106
private	long	_minPPS	7	80

Future Directions

- Pattern analysis
  - Extraction of "hot streams"
- New reduction strategies
- Integration with other tracing tools

Summary

- The problems with current tracing methods.
- A solution: STEP
  - Framework overview
  - Example: JVMPI allocation data
  - Uses of STEP data

Links to More Information

STEP	http://www.sable.mcgill.ca/step/
EVolve	http://www.sable.mcgill.ca/evolve/
JIL (JIMPLEX)	http://www.sable.mcgill.ca/jil/
Sable Research	http://www.sable.mcgill.ca