

GCspy: An Adaptable Heap Visualisation Framework

Tony Printezis

tony.printezis@sun.com

Sun Microsystems,
1 Network Drive,
Burlington, MA01801,
USA

Richard Jones

r.e.jones@ukc.ac.uk

Computing Laboratory,
University of Kent,
Canterbury, Kent, CT2 7NF,
UK

Motivation

GCs have (almost) *chaotic* behaviour!

Visualisation allows

- concise insight into GC's behaviour
- profiling / debugging

Avoid collecting / analysing *very* large traces

- “*a picture is worth a thousand awk invocations!*”

T.Printezis

GC Debugging / Profiling

```
printf("obj %08x, class ptr %08x",_addr,  
      _class_ptr);
```

```
obj f000defa, class ptr f0008620  
obj f000df40, class ptr f00026a0  
obj f000ef00, class ptr f00a600a  
obj f000ef50, class ptr f00a600a  
obj f000ff00, class ptr f0008620  
. . .
```

Advanced Tools



Demo

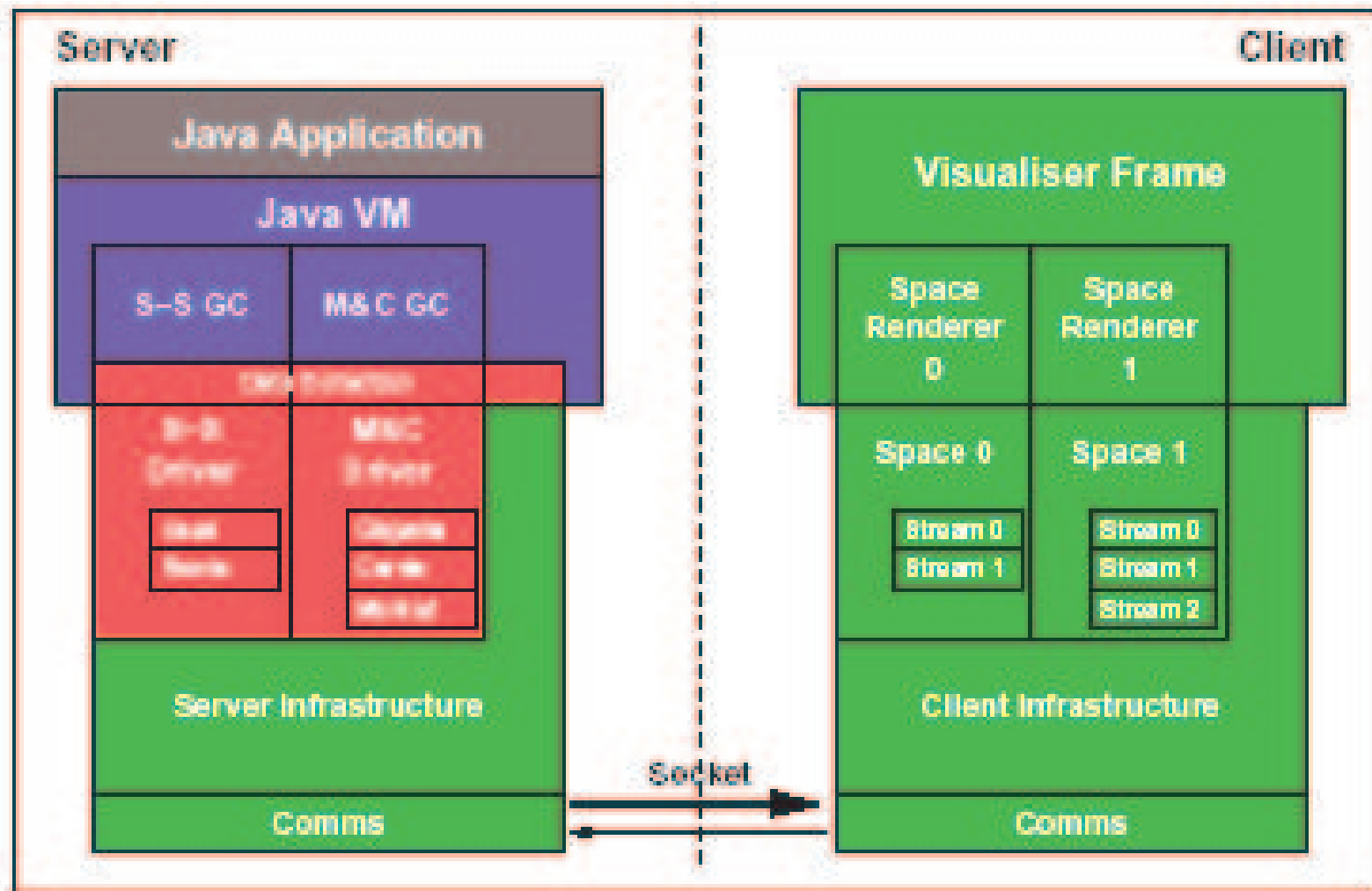
Requirements

1. Portability
2. Adaptability
3. Extensibility
4. Scalability
5. Responsiveness?

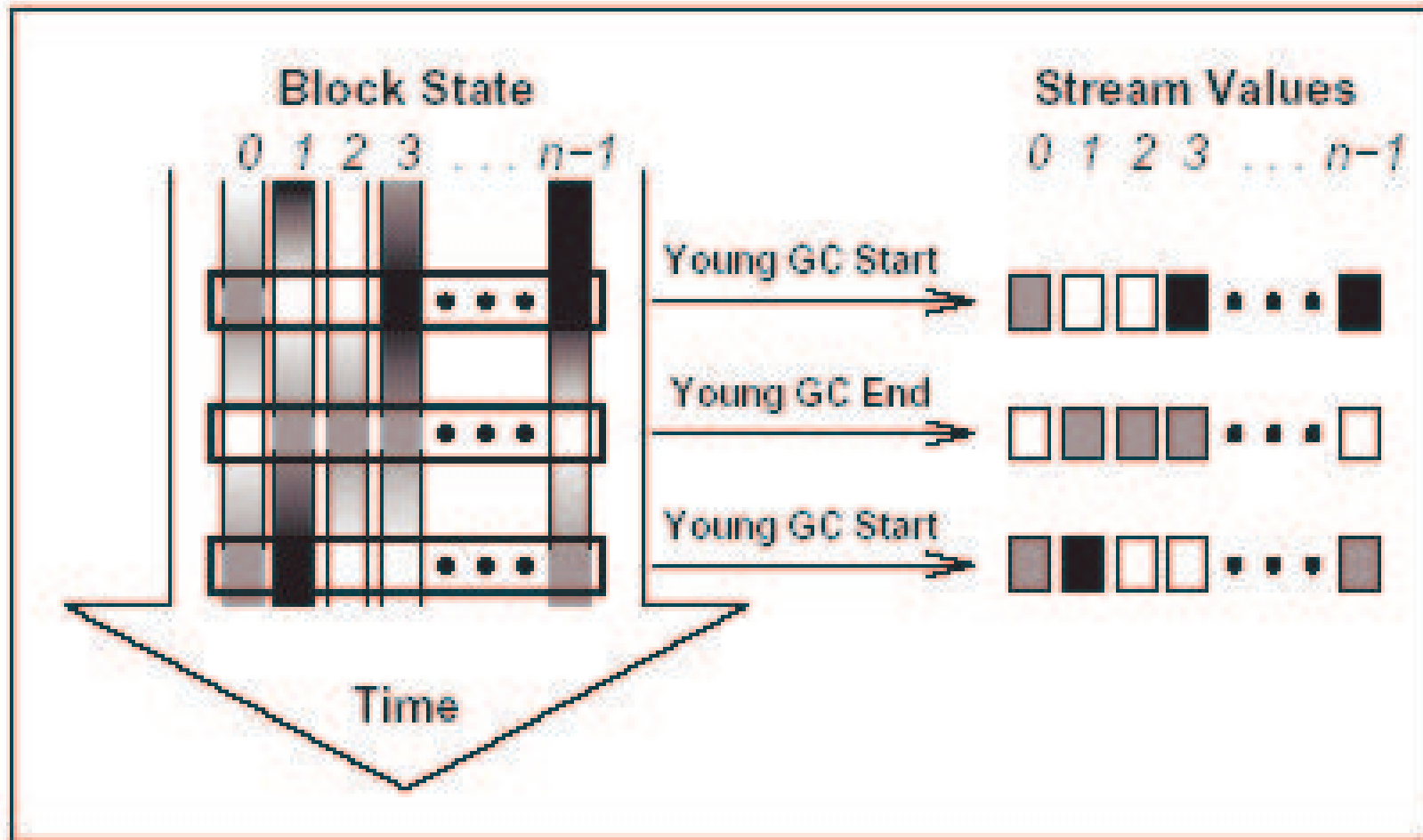
Abstractions

| GC | Visualiser |
|-------------------------------------|-------------------|
| Component e.g. generation | Space |
| Block e.g. memory block | Tile |
| Attribute e.g. used space | Stream |

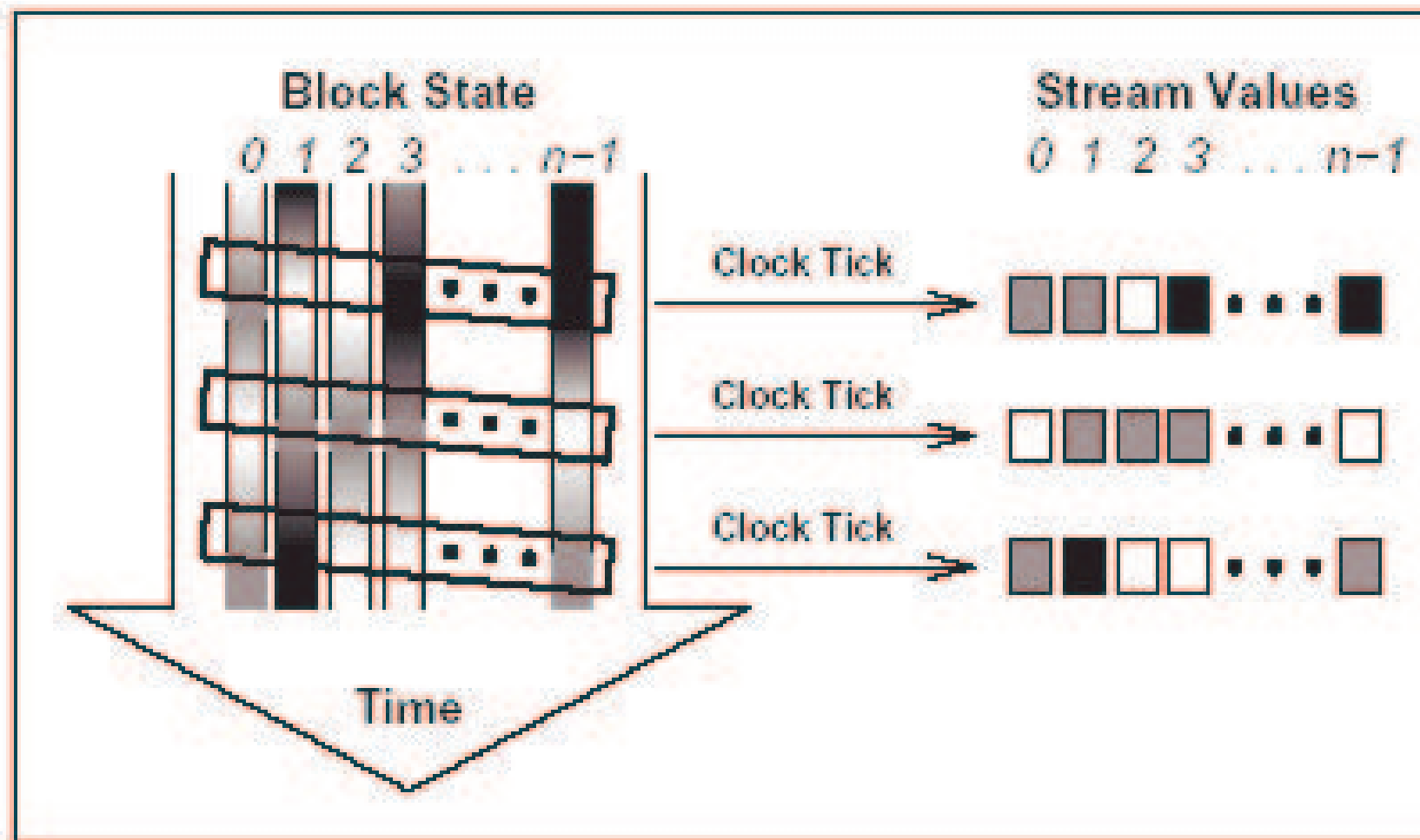
Architecture



Data collection — sync



Data collection — async



Success Stories

1. **Sun Microsystems' Java HotSpot™ Virtual Machine (C server)**
2. **Sun Microsystems Laboratories Virtual Machine for Research (C++ server)**
3. **IBM's Jikes™ Research Virtual Machine (Java server)**
4. **BDW Conservative GC for C and C++ (C server)**



Questions?