Automatic Compiler Testing

Christian Lindig – Dagstuhl

Math Table

…

3 + 4 = 7

3 * 4 = 12

3 + 5 = 8

3 * 5 = 15

…

Program

```
…
printf(“3 + 4 = %d\n”, 3+4);
printf(“3 * 4 = %d\n”, 3*4);
printf(“3 + 5 = %d\n”, 3+5);
printf(“3 * 5 = %d\n”, 3*5);
…
```

```
…
3 + 4 = 7
3 * 4 = 12
3 + 5 = 8
3 * 5 = 15
…
```

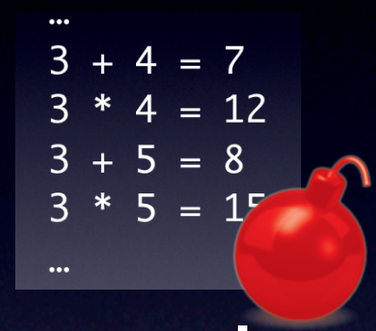
Functions

```
void f(int x, int y)
{
  printf(“%d + %d = %d\n”, x, y, x+y);
  printf(“%d * %d = %d\n”, x, y, x*y);
}
…
f(3,4);
f(3,5);
…
```

```
…
3 + 4 = 7
3 * 4 = 12
3 + 5 = 8
3 * 5 = 15
…
```

Compilation

source code



result

application

CPU

Compiler Bugs

- A compiler bug affects all programs compiled with it
- Operating systems (Linux, Windows, MacOS) are compiled from C
- Resulting programs in embedded systems (car, cell phone, …) cannot be fixed
- Compiler bugs are expensive

Self-Checking Programs

```
void f(int i)
{
    assert(i == 1234);
}

void main()
{
    f(1234);
}
```

i is checked to contain 1234; programs halts otherwise

1234 is passed to f

Random Testing

1. Quest generates random self-checking program test.c
2. Compile test.c to test.exe
3. Execute test.exe

∞

until test.exe fails

GNU Compiler Bug #18742

```
struct A {long long  a1; int a2;}    aa = { 1L , 2 };
struct B {double     b1; int b2;}    bb = { 3.0, 4 };
struct C {short int  c1; }           cc = { 5      };
```

```
int main() {
    f(aa, bb, cc);
    return 0;
}
```

fails

```
void f(struct A a, ...) {
    struct B b;
    struct C c;
    va_list  ap;
    va_start(ap, a);
    b = va_arg(ap, struct B);
    c = va_arg(ap, struct C);
    va_end(ap);
    assert(c.c1 == cc.c1);
}
```

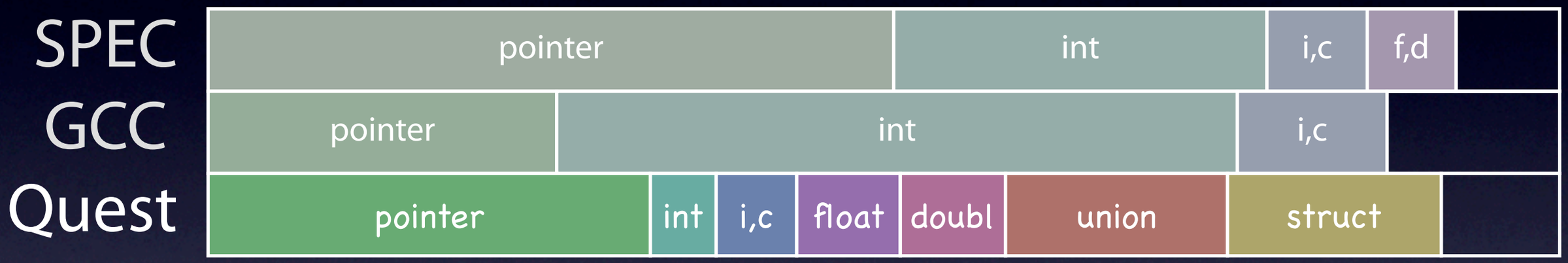
14 Bugs found with Quest

compiler	IRIX MIPS	SunOS 5 SPARC	Linux x86	MacOS PPC
MipsPro	1			
PathCC			3	
LCC 4.2			1	
ICC 8.1			1	
GCC 2.x		1	1	
GCC 3.x	1		2	1
GCC 4.x				2

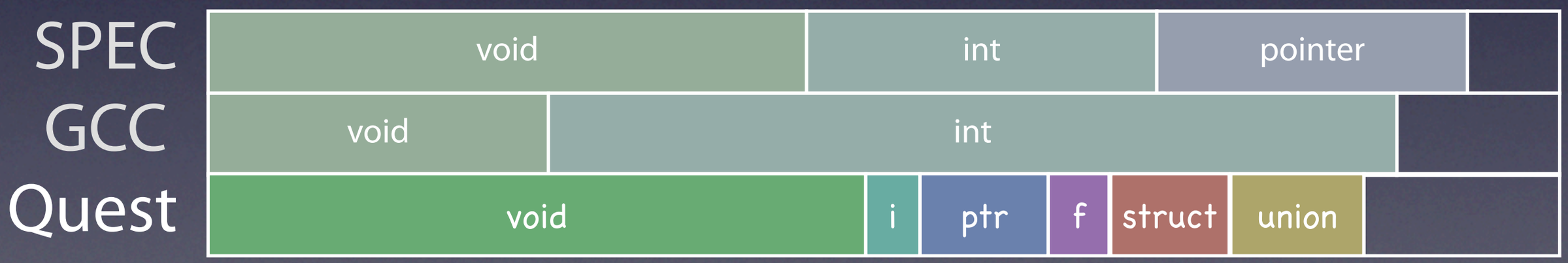
All C compilers are mature and used for production

Compiler Test Suites

Static Distribution of Argument Types



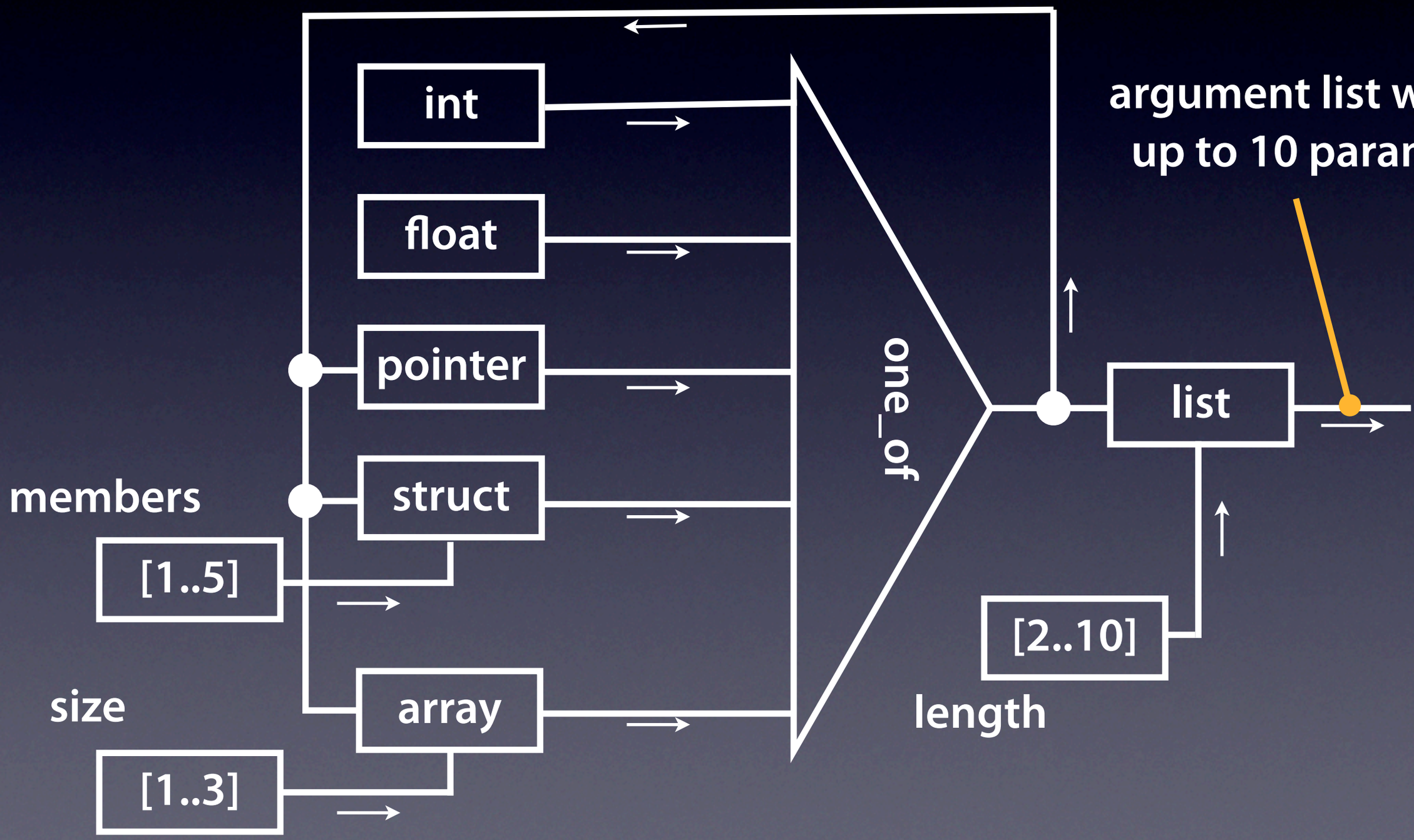
Static Distribution of Return Types



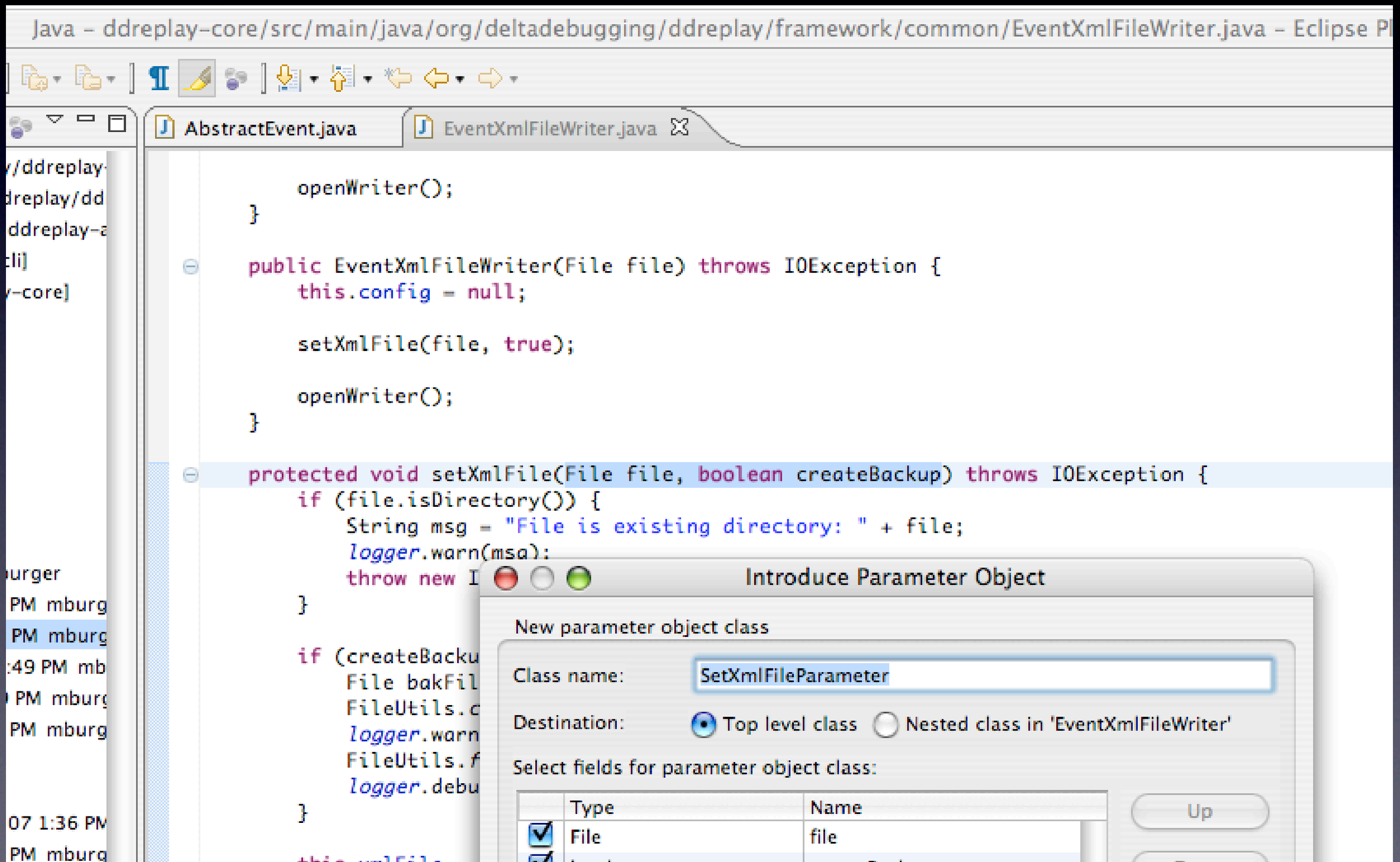
Correctness–Why so Hard?

- informal specifications
- evolution of C vs. binary compatibility
- lack of orthogonality, many dark corners
- 2 implementations in compiler:
  call site and function definition out of sync
- optimization: PathCC has 300+ flags

Quest Feature: Composable Randomness

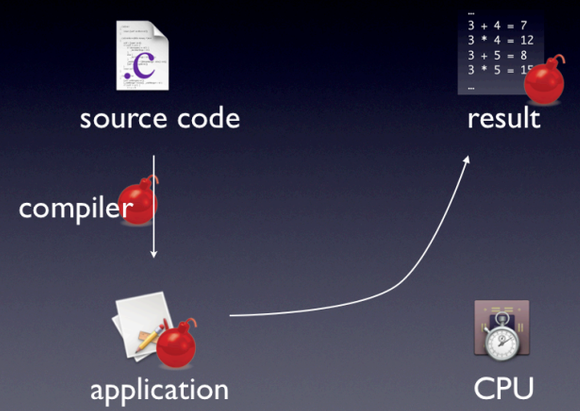


Future Work


Java – ddreplay-core/src/main/java/org/deltadebugging/ddreplay/framework/common/EventXmlFileWriter.java – Eclipse Pl
AbstractEvent.java
EventXmlFileWriter.java
openWriter();
}
public EventXmlFileWriter(File file) throws IOException {
this.config = null;
setXmlFile(file, true);
openWriter();
}
protected void setXmlFile(File file, boolean createBackup) throws IOException {
if (file.isDirectory()) {
String msg = "File is existing directory: " + file;
logger.warn(msg);
throw new I
}
if (createBacku
File bakFil
FileUtils.c
logger.warn
FileUtils.f
logger.debu
}
Introduce Parameter Object
New parameter object class
Class name:
SetXmlFileParameter
Destination:
Top level class
Nested class in 'EventXmlFileWriter'
Select fields for parameter object class:
Type
Name
File
file
Up

Summary

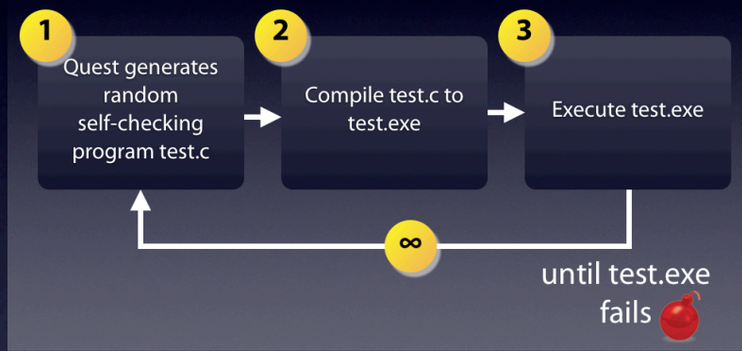
Compilation



Compiler Bugs

- A compiler bug affects all programs compiled with it
- Operating systems (Linux, Windows, MacOS) are compiled from C
- Resulting programs in embedded systems (car, cell phone, …) cannot be fixed
- Compiler bugs are expensive

Random Testing

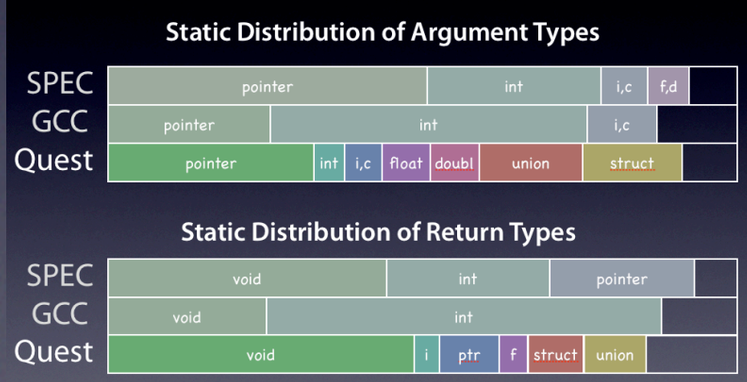


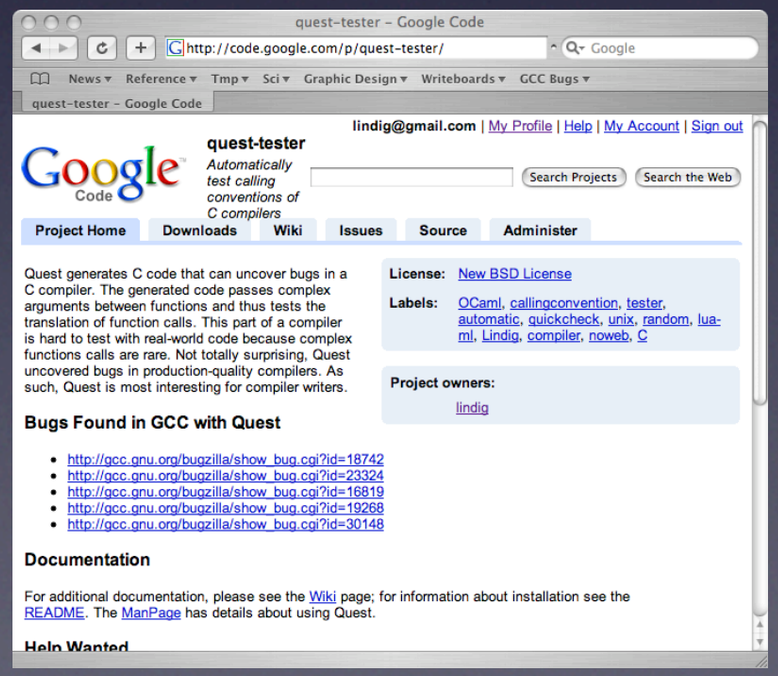
14 Bugs found with Quest

compiler	IRIX MIPS	SunOS 5 SPARC	Linux x86	MacOS PPC
MipsPro	1			
PathCC			3	
LCC 4.2			1	
ICC 8.1			1	
GCC 2.x		1	1	
GCC 3.x	1		2	1
GCC 4.x				2

all bugs are related to var args or struct/unions

Compiler Test Suites





http://code.google.com/p/quest-tester/