

ALLOY  
FIELDS AND TYPES

CURT CLIFTON  
ROSE-HULMAN INSTITUTE OF TECHNOLOGY



FIELDS

# FIELDS DECLARE RELATIONS

■ **sig** A {f: e} DECLARES

■ A SIGNATURE A

■ AND A RELATION  $f: A \rightarrow e$ , SUCH THAT  
**all this: A | this.f in e**

■ **sig** A {f: m e} SAYS **all this: A | this.f in m e**

■ EXAMPLES...

MULTIPLICITY



# FIELD EXAMPLES

- **abstract sig** Object { }  
**sig** Directory **extends** Object { contents: **set** Object }  
**one sig** Root **extends** Directory { }  
**sig** File **extends** Object { }  
**sig** Alias **extends** File { to: Object }
- **sig** Forecast { weather: City -> **one** Weather }  
**sig** City, Weather { }  
**one sig** Rainy, Sunny, Cloudy **extends** Weather { }
- **sig** TrafficLight { color: Color **some** -> Time }  
**abstract sig** Color { }  
**one sig** Red, Green, Yellow **extends** Color { }  
**sig** Time { }



Q1,2

# GROUPING FIELDS

■ A CAT HAS THREE NAMES:

■ **sig** Cat { **disj** daily, peculiar, ineffable: Name }  
**sig** Name { }

MUTUALLY DISJOINT



# DEPENDENT DECLARATIONS

- A CAT STILL HAS THREE NAMES:

- **sig** Cat {  
  daily: Name,  
  peculiar: Name - daily,  
  ineffable: Name - (daily + peculiar)  
}

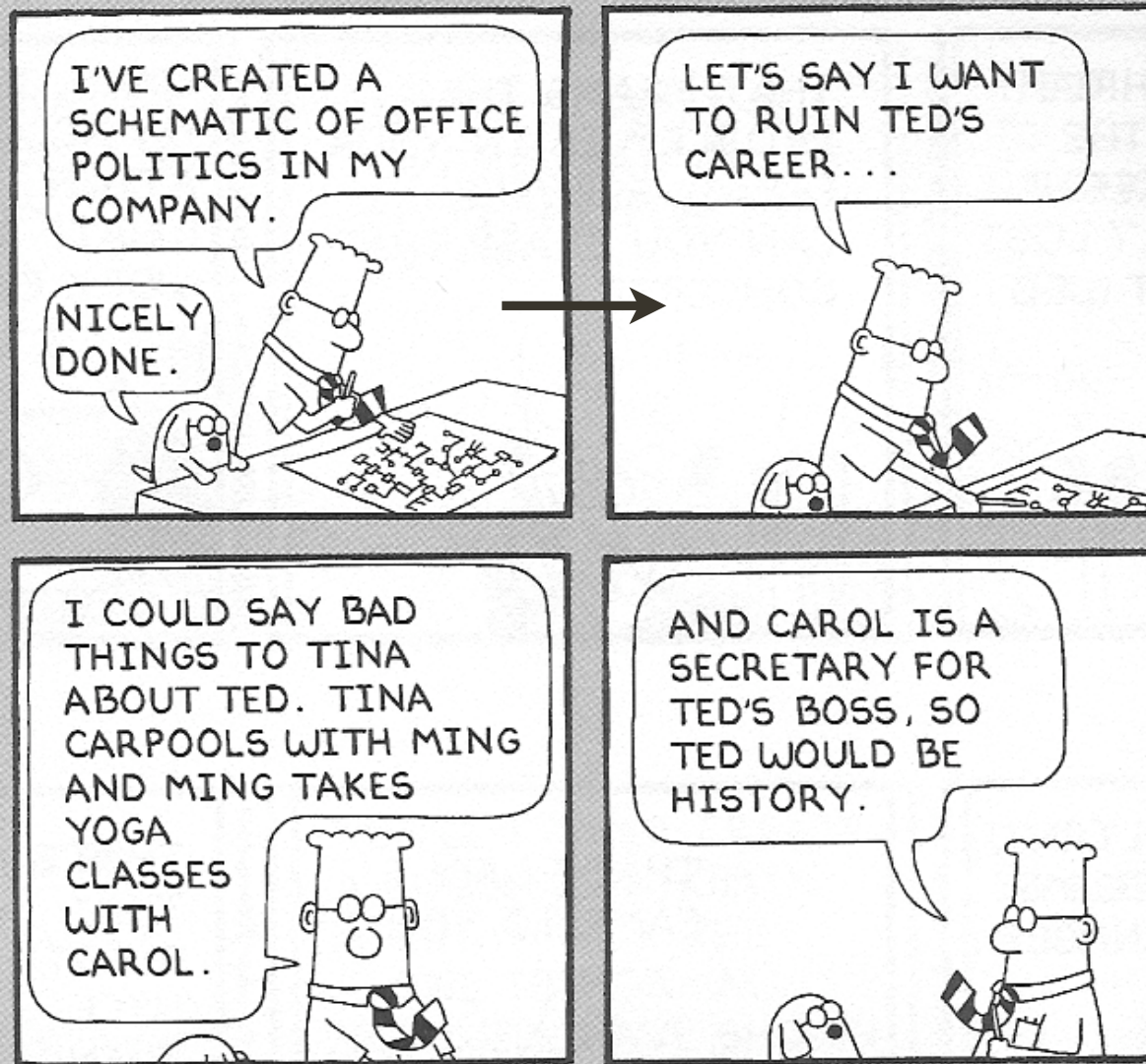
CAN REFERENCE  
EARLIER FIELDS AND  
SUPERTYPE FIELDS

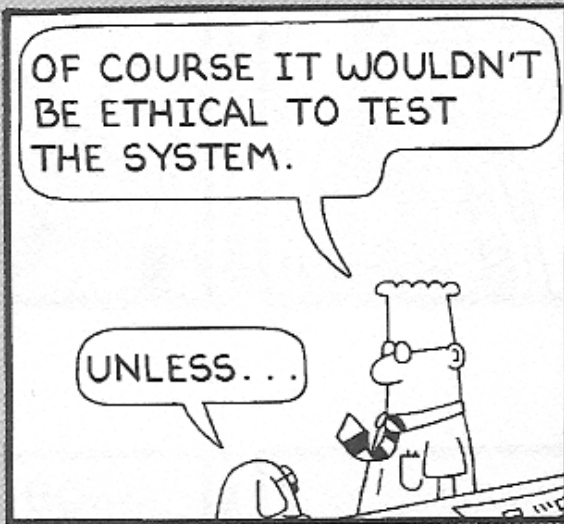


- RADIO STATION WITH TRANSLATORS:

- **sig** RadioStation {  
  owns: **set** Freq,  
  freq: Location -> **one** owns  
}  
**sig** Freq, Location {}

# CARTOON OF THE DAY





9/23/01 ©2001 United Feature Syndicate, Inc.



TYPES

# TYPES IN ALLOY

- TWO PURPOSES:
  - DETECT REDUNDANCY – USUAL INDICATES A MISCONCEPTION
  - RESOLVE OVERLOADING
- DIFFERENT THAN MOST PROGRAMMING LANGUAGES:
  - NO FALSE ALARMS AND NO GUARANTEES

# BASIC TYPES

- INTRODUCED BY SIGNATURES

- CAN OVERLAP:

- TWO BASIC TYPES OVERLAP IF ONE IS A SUBTYPE OF THE OTHER

- EXAMPLE:

- SIG OBJECT {}

- SIG FILE, DIRECTORY EXTENDS OBJECT {}

- SIG TEMP IN OBJECT {}

DOESN'T HAVE OWN TYPE



# RELATIONAL TYPES

- EVERY EXPRESSION GETS A RELATIONAL TYPE

- A UNION OF PRODUCTS:

- $A_1 \rightarrow B_1 \rightarrow \dots +$

- $A_2 \rightarrow B_2 \rightarrow \dots +$

- $A_3 \rightarrow B_3 \rightarrow \dots +$

- ...

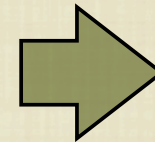
- WHERE  $A_1, B_1, \dots$ , ARE BASIC TYPES

- INFERRED AUTOMATICALLY

# TYPE INFERENCE

## 1. GETS BASIC TYPES FROM SIGNATURES

- **abstract sig** Object { }  
**sig** Directory **extends** Object { ... }  
**one sig** Root **extends** Directory { }  
**sig** File **extends** Object { }  
**sig** Alias **extends** File { ... }  
**sig** Temp **in** Object { }



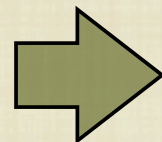
**BASIC TYPES:**  
Object,  
Directory, Root,  
File, Alias

# TYPE INFERENCE

1. GETS BASIC TYPES FROM SIGNATURES

2. EACH FIELD IS GIVEN A TYPE

■ ...  
**sig** Directory **extends** Object {  
  contents: **set** Object  
}  
...  
**sig** Alias **extends** File {  
  to: Object  
}  
...



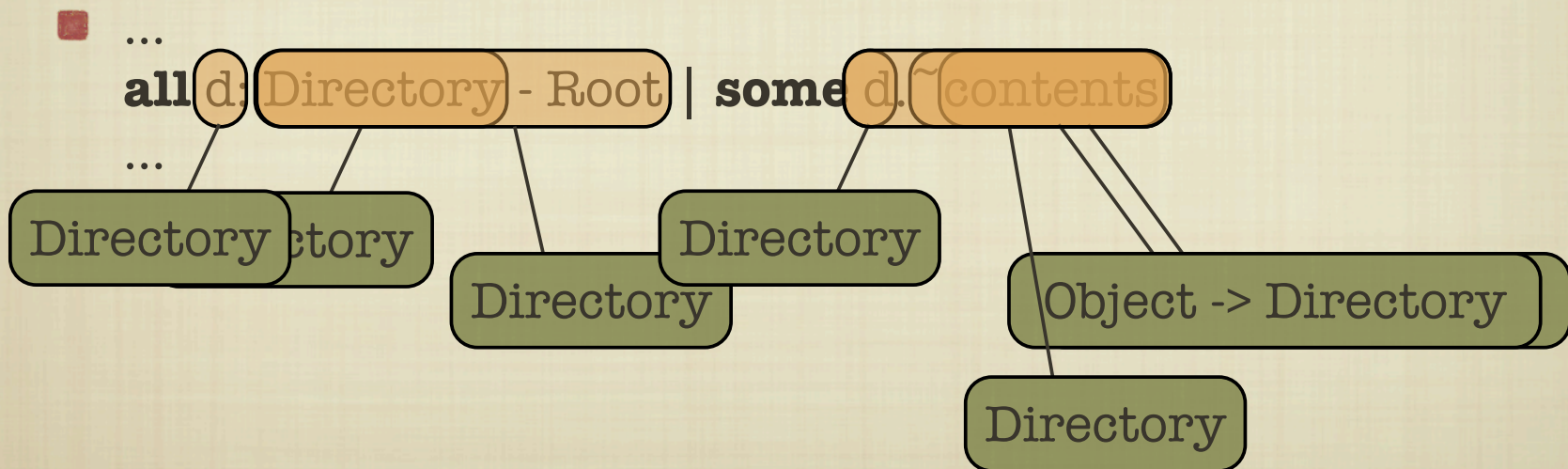
**FIELD TYPES:**  
contents: Directory -> Object  
to: Alias -> Object

# TYPE INFERENCE

1. GETS BASIC TYPES FROM SIGNATURES

2. EACH FIELD IS GIVEN A TYPE

3. EACH EXPRESSION GETS A TYPE



# TWO KINDS OF TYPE ERRORS

- **MIXED ARITY:**

- contents + Directory

- **CAN'T UNION SET OF PAIRS WITH UNARY SET**

- **REDUNDANCY...**

# REDUNDANCY

- **abstract sig** Object {}  
**sig** Directory **extends** Object { contents: **set** Object}  
**one sig** Root **extends** Directory {}  
**sig** File **extends** Object {}  
**sig** Alias **extends** File {to: Object}
- Directory & Alias – **ERROR!**
- Alias.contents – **ERROR!**



FIELD OVERLOADING

# FIELD OVERLOADING

- USING THE SAME FIELD NAME IN DIFFERENT SIGS
- ALLOY USES TYPES TO GUESS WHICH FIELD TO USE

# EXAMPLE

```
sig Object, Block {  
sig Directory extends Object {  
  contents: set Object  
}  
sig File extends Object {  
  contents: set Block  
}  
fact NoEmptyFiles {  
  all f: File | some f.contents  
}  
fact NoSharedBlocks {  
  no disj f, f': File | some f.contents & f'.contents  
}  
fun GetFileOf[b: Block]: File {  
  b.~contents  
}
```

WHICH  
CONTENTS?

# SUBTLE EXAMPLE

```
sig Object, Block {  
sig Directory extends Object {  
  contents: set Object  
}  
sig File extends Object {  
  contents: set Block  
}  
  
fact NoSelfDirectory {  
  no o: Object | o in o.contents  
}
```

WHICH  
CONTENTS?

# USING RESTRICTION

```
sig Node {  
  next: Node,  
  value: Value one -> Time  
}
```

```
sig Time {  
  next: Time  
}
```

```
sig Value {}
```

```
fact NodesInRing { all n: Node | Node in n.^next }
```

```
fact TimeIsAnArrow { no ^next & iden } AMBIGUOUS
```

```
fact TimeIsAnArrow { no ^(Time <: next) & iden }
```