

Translating Chalice into SIL

Bachelor's Thesis

Christian Klauser

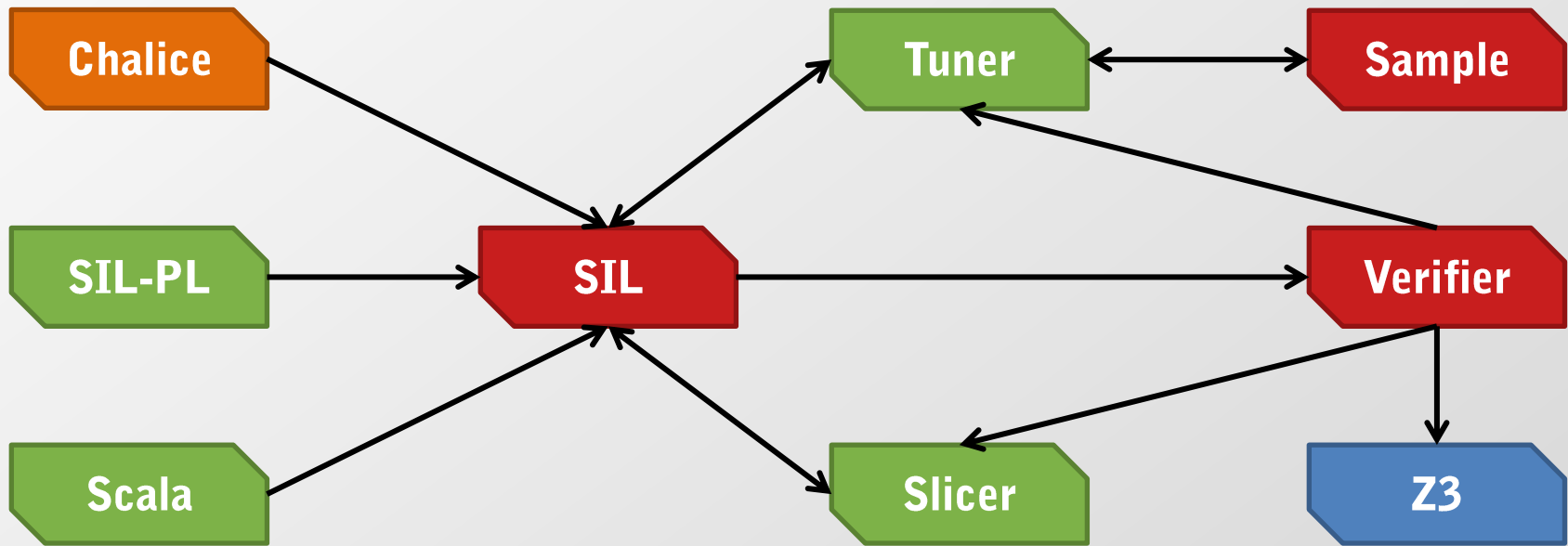
Supervisor: Dr. Alex Summers

The Semper Project

- Long term project
- Automatic program verifier for Scala
 - verify concurrent programs
 - reduce annotation overhead
 - deal with functional features (e.g., closures)



Semper Architecture Design



Chalice

- Annotated Methods

```
class Cell {  
    var v: int;  
  
    method inc(d: int)  
        requires 0 < d;  
        requires acc(v);  
        ensures v == old(v) + d;  
    {    v := v + d; }  
}
```

Chalice

- Annotated Methods
- Monitors

```
class Cell {  
    var v: int;  
    invariant acc(v) && 0 <= c;  
}  
  
class Program {  
    method main() {  
        var c:Cell := new Cell;  
        c.v := 3;  
        share c;  
  
        acquire c; call c.inc(2); release c;  
    }  
}
```

Chalice

- Annotated Methods
- Monitors
- Predicates/Functions

```
class Cell {  
    var v: int;
```

```
    predicate valid  
    { acc(this.v) && 0 <= this.v }
```

```
    function add(d:int) requires valid;  
    { unfolding valid in this.v + d; }
```

```
    ...  
}
```

Chalice

- Annotated Methods
- Monitors
- Predicates/Functions
- Fork-Join

```
class Cell { ... }  
class Program {  
  method main() {  
    var c1:Cell := new Cell;  
    var c2:Cell := new Cell;  
    c1.v := 0; c2.v := 5;  
    fork tk1 = c1.inc(3);  
    fork tk2 = c2.inc(1);  
    join f1 := tk1;  
    join f2 := tk2;  
  }  
}
```

Chalice2SIL

- First front-end for Chalice
- Help establish and test the tool chain
- Ideally no changes to Chalice
- If enough time is left
 - Predicates and functions
 - Deadlock avoidance
 - Channels (Actor model)

Thank you

QUESTIONS?