ArchJava

- ArchJava: Java extension – architectural features
  - components, ports and connections
- Benefits
  - Better program understanding
  - Reliable architectural reasoning about code
  - Keeping architecture and code consistent as they evolve

ArchJava

- Approach: add architecture to language
  - Control-flow communication integrity
    - Enforced by type system
  - Architecture updated as code evolves
  - Flexible
    - Dynamically changing architectures
    - Common implementation techniques

A Parser Component

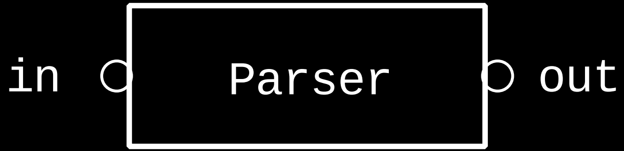
Parser

```
public component class Parser {
```

Component class

- Defines architectural object
- Must obey architectural constraints

A Parser Component

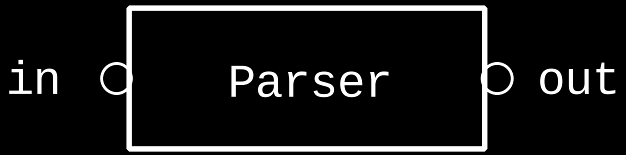


```
public component class Parser {
  public port in {
    requires Token nextToken();
  }
  public port out {
    provides AST parse();
  }
```

Components communicate through *Ports*

- A two-way interface
- Define *provided* and *required* methods

A Parser Component

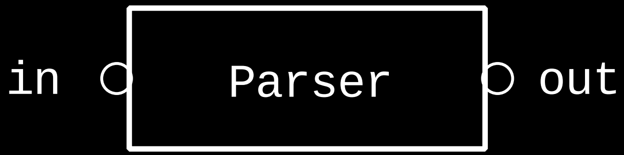


```
public component class Parser {
  public port in {
    requires Token nextToken();
  }
  public port out {
    provides AST parse();
  }
```

Ordinary (non-component) objects

- Passed between components
- Sharing is permitted
- Can use just as in Java

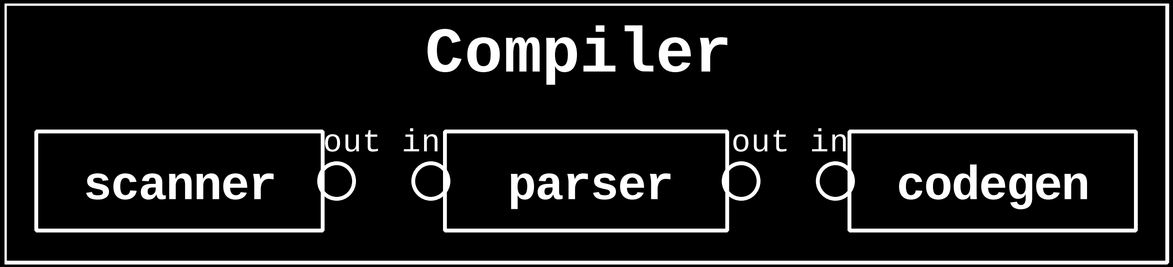
A Parser Component



```
public component class Parser {
  public port in {
    requires Token nextToken();
  }
  public port out {
    provides AST parse();
  }
  AST parse() {
    Token tok=in.nextToken();
    return parseExpr(tok);
  }
  AST parseExpr(Token tok) { ... }
  ...
}
```

Can fill in architecture with ordinary Java code

Hierarchical Composition

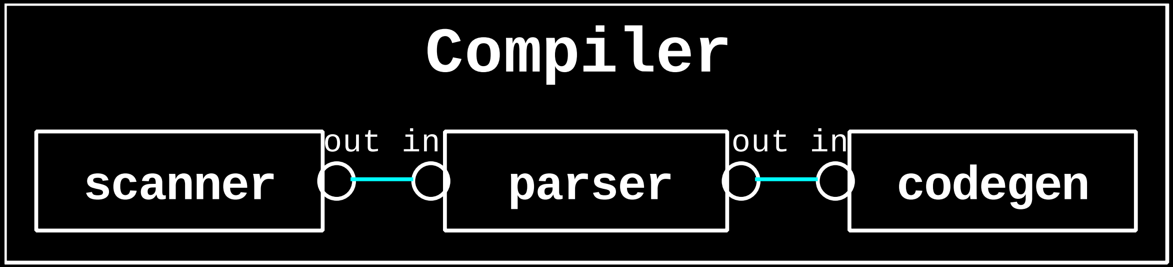


```
public component class Compiler {
  private final Scanner scanner = new Scanner();
  private final Parser parser = new Parser();
  private final CodeGen codegen = new CodeGen();
```

Subcomponents

- Component instances inside another component
- Communicate through connected ports

Hierarchical Composition



```
public component class Compiler {
  private final Scanner scanner = new Scanner();
  private final Parser parser = new Parser();
  private final CodeGen codegen = new CodeGen();
  connect scanner.out, parser.in;
  connect parser.out, codegen.in;
```

Connections

- Bind required methods to provided methods

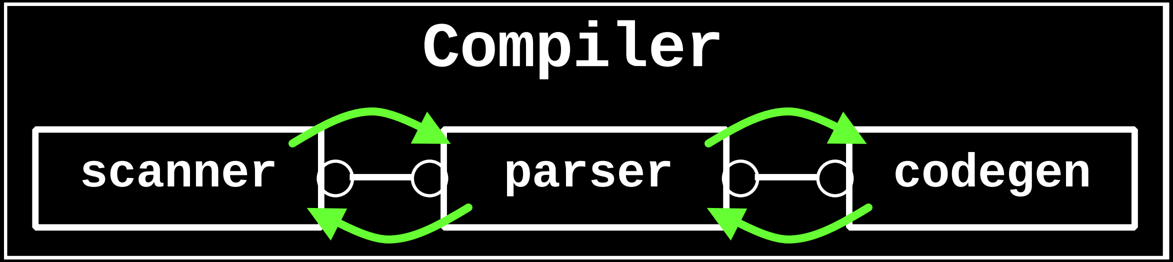
Architecture and Implementation

- Does code conform to architecture?

- Communication Integrity
  - Consistency Property

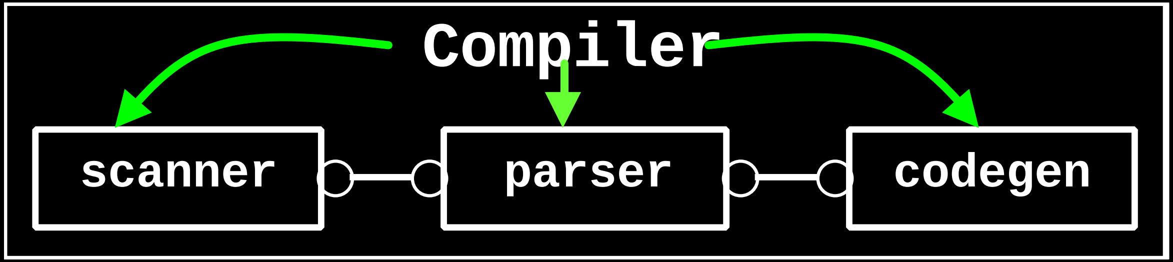
A component may only communicate with the components it is connected to in the architecture

Control Communication Integrity



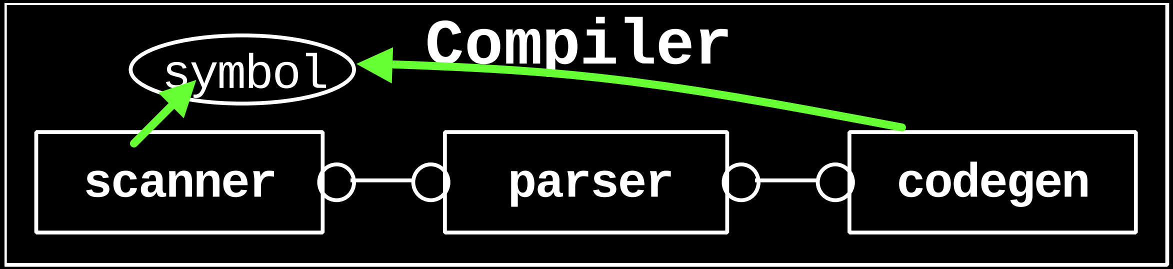
- Architecture allows
  - Calls between connected components

Control Communication Integrity



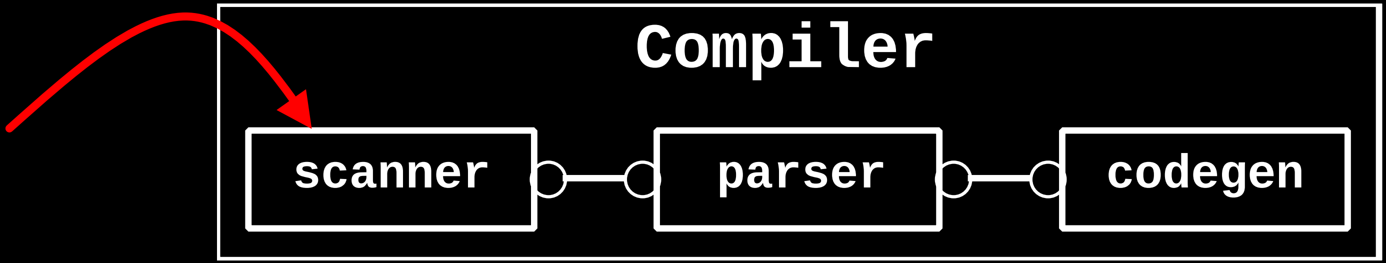
- Architecture allows
  - Calls between connected components
  - Calls from a parent to its immediate subcomponents

Control Communication Integrity



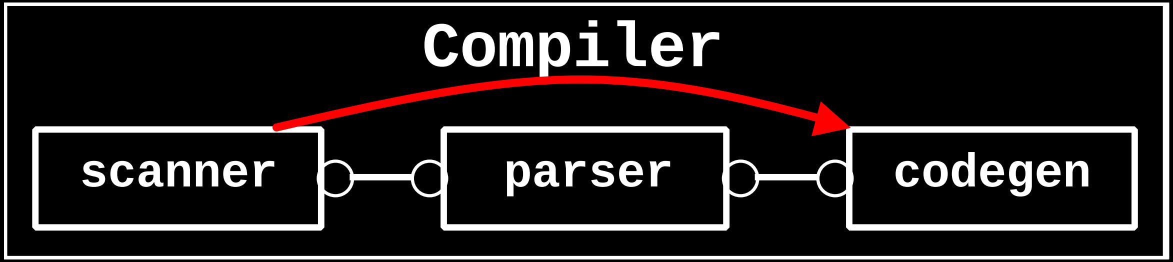
- Architecture allows
  - Calls between connected components
  - Calls from a parent to its immediate subcomponents
  - Calls to shared objects

Control Communication Integrity



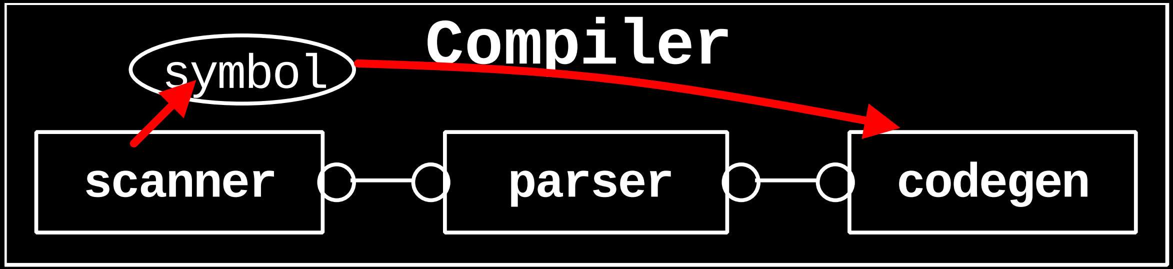
- Architecture allows
  - Calls between connected components
  - Calls from a parent to its immediate subcomponents
  - Calls to shared objects
- Architecture forbids
  - External calls to subcomponents

Control Communication Integrity



- Architecture allows
  - Calls between connected components
  - Calls from a parent to its immediate subcomponents
  - Calls to shared objects
- Architecture forbids
  - External calls to subcomponents
  - Calls between unconnected subcomponents

Control Communication Integrity



- Architecture allows
  - Calls between connected components
  - Calls from a parent to its immediate subcomponents
  - Calls to shared objects
- Architecture forbids
  - External calls to subcomponents
  - Calls between unconnected subcomponents
  - Calls through shared objects

Conclusion

- ArchJava integrates architecture with Java code
- Control communication integrity
  - Keeps architecture and code synchronized

- Formalization of language & properties
  - ArchFJ (Arch Featherweight Java)