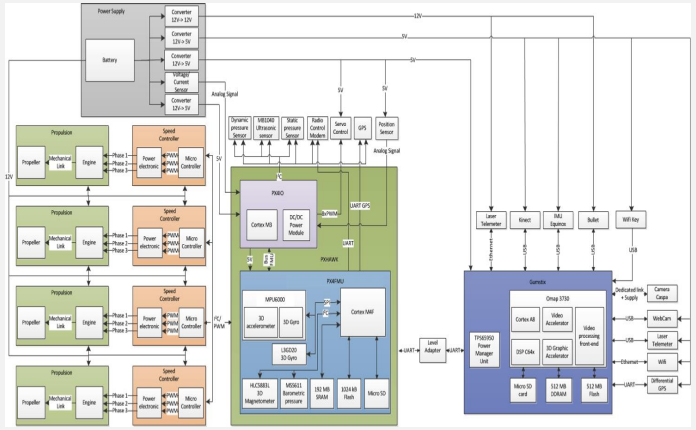
MULTI-VIEW ARCHITECTURE DESCRIPTION AND ENFORCEMENT

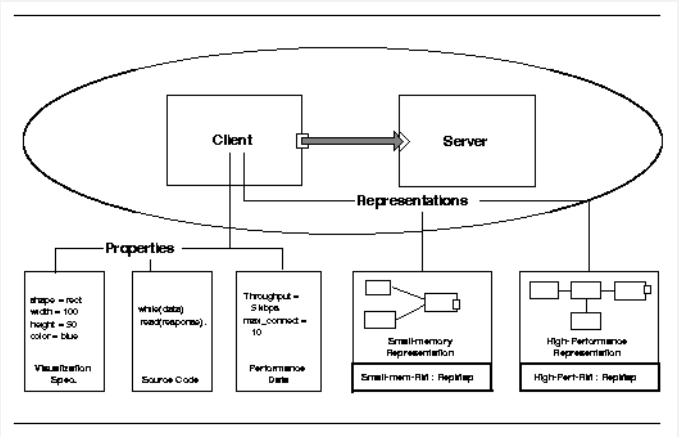
Amanda Liu
Columbia University
Selva Samuel
Carnegie Mellon University
Prof. Jonathan Aldrich
Carnegie Mellon University

Architecture Description Languages (ADLs)

- Formal syntax and semantics for performance-critical systems
- Analyzable model for large scale systems
- Separation of concerns



AADL



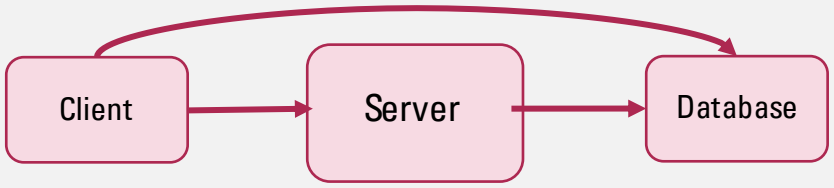
ACME

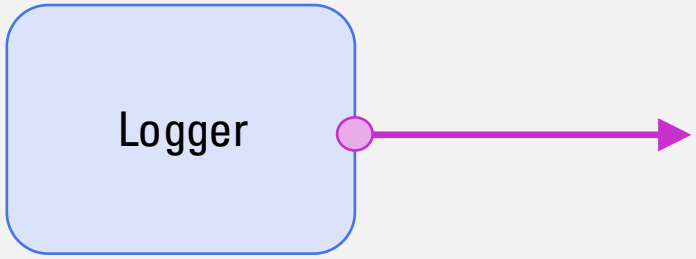
```
public component class Filter = { ... }

public component class Lower extends Filter {
   public port input {
      provides void processChar (int c) {}
   }
   public port output {
      requires void processChar (int c) {}
   }
}

public component class Capitalize extends Filter {
   private final Lower lower = new Lower ();

   connect lower.output, merge.input1;
   ...
}
```

ArchJava

Architectural Drift and Erosion

- As-intended vs. as-implemented gap






```
client.socket(databaseAddr, 80);
```

```
import FileIO;
```

WYVERN

- Capability safety
- Modules and module functors
- Metaprogramming

Wyvern Modules

```
module def Logger(write: FileIO)

def log(): String

...
```

ArchWyvern

A Wyvern ADL

Wyvern

- Capability Safety
- Metaprogramming

ArchWyvern

- Structural security
- Architectural integrity in runtime semantics

ArchWyvern Components

```
component Client
   port getInfo: requires CSIface
```

```
module def Client(getInfo: CSIface)
   ...
```

ArchWyvern Connectors

```
connector JSONCtr
   val host: String
   val prt: Int
```

```
type JSONCtr
   val host: String
   val prt: Int
      metadata new
         ...
```

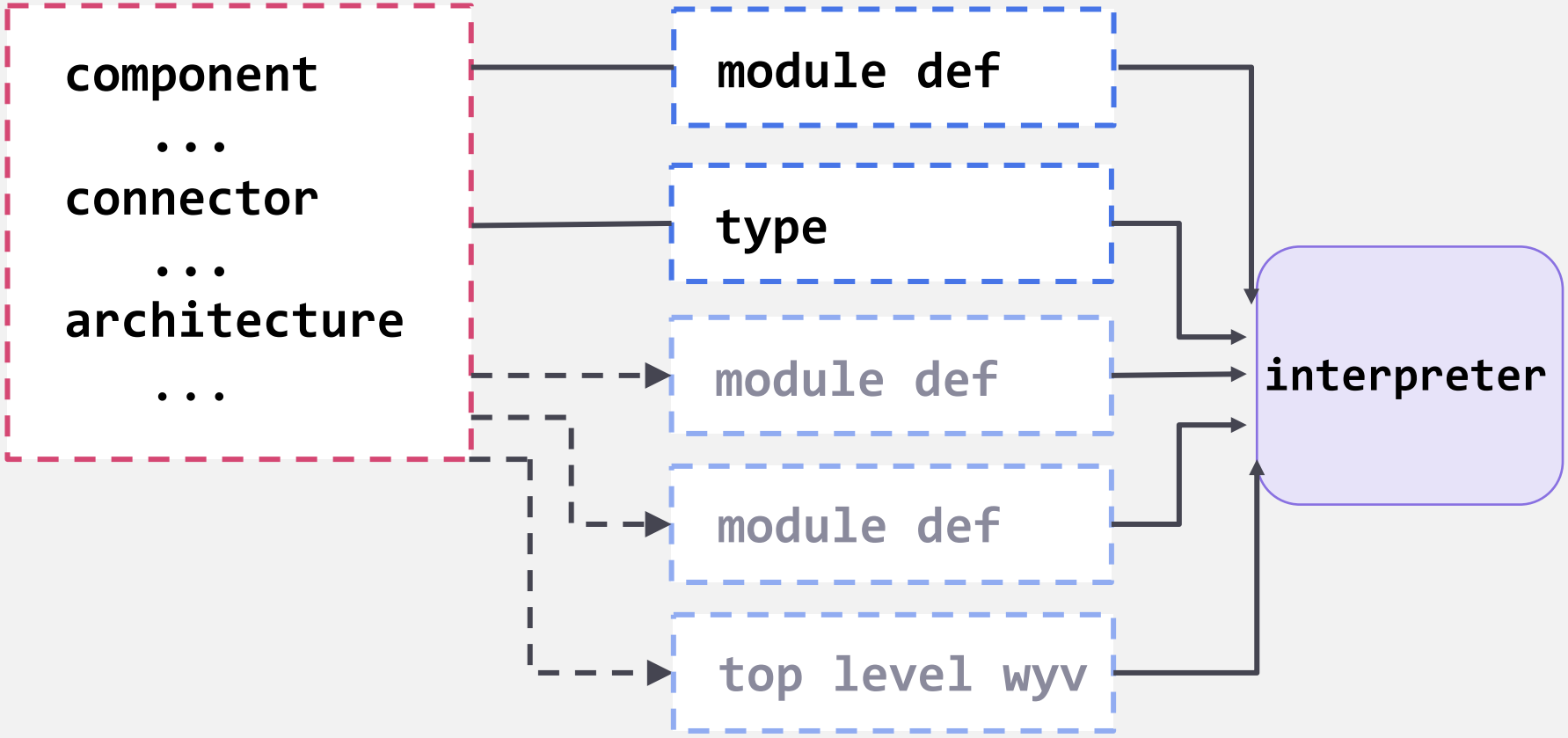
ArchWyvern Architecture Description

```
architecture TwoTier
   components
      Client client
      Server server
   connectors
      JSONCtr jsonCtr
   ...
```

Code Generation and Metaprogramming

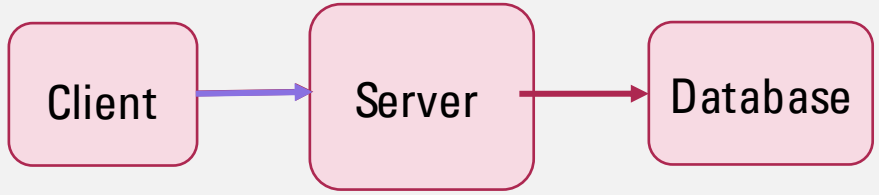
```
type JSONCtr
   val host: String
   val prt: Int
      metadata new
         ...
```

ArchWyvern Pipeline



In Summary

- As-intended architectural integrity at deployment
- High-level description tied to implementation
- Increased ease of software evolution
- Increased formalization and empirical work in the future




```
module def TCPClient(net: Network, con: JSONCtr,
                     client: Client)
def entryPoint(): Unit
     net.socket(client.addr);
     net.connect(con.serverAddr)
     ...
```

Thank You!