

# A Language Approach for Multicore and Distributed Programming

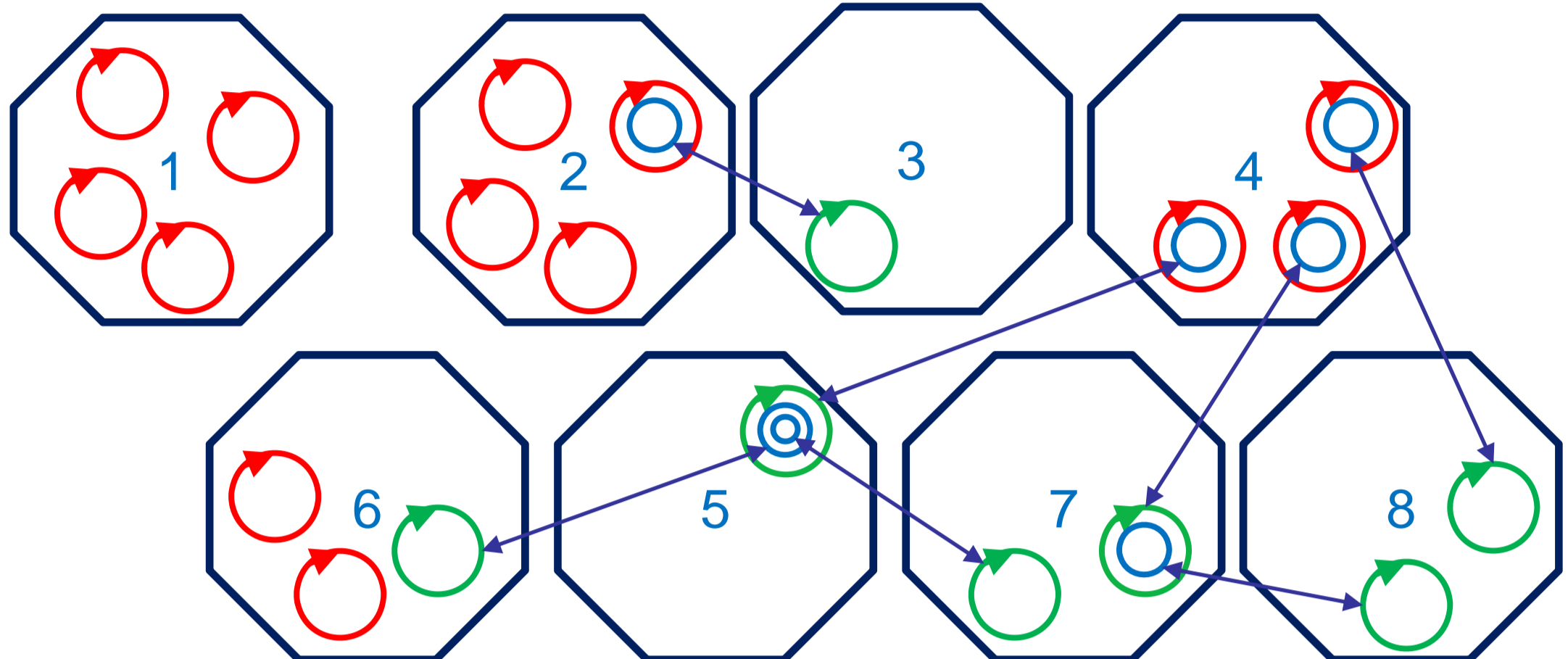
Zonnon Programming language, ETH Zonnon native compiler, ETH Zonnon compiler for .NET

www.zonnon.ethz.ch

## Objective

This project is aimed at increasing developers productivity and reducing entry barriers to programming highly concurrent and distributed applications by means of a novel programming model. It is accompanied by a language tool capable to enforce certain properties desired for the model. The key property addressed by the model is locality that enables the compiler to perform clustering of the application into loosely coupled components.

## Programming Model



- a root activity
- a root activity with a link to its child activity
- an agent activity
- an agent activity with a link to its own child
- an agent activity with two more links to its child activities

- ❑ Applications viewed as constellations of active objects towards support of object structures instead of individual objects
- ❑ Activities extend procedure calls into asynchronous communications and procedure signatures into communication protocols that guard message passing communications
- ❑ Two types of object relations
  - Strong-coupled if objects communicate via method calls
  - Loosely coupled if objects communicate via protocols only
- ❑ Protocols for communication are exposed by object interfaces
- ❑ Objects communicate through channels which ownership is statically ensured and thus compiler can check protocol

## Static analysis

```

protocol Service = (
    Request, Allow, Reject, Enter, Leave,
    park = [Enter] Leave,
    dialog = Request (?Allow park | ?Reject)
);
activity {public} Serve implements Service;
var cmd: Service;
begin
    accept cmd; (* Request *)
    if bookSpace then
        return Service.Allow;
    ...
end Serve;
    
```

```

object Client
activity {public} Drive;
var parking: Server.Serve;
ans: Server.Service;
begin
    parking := new Server.Serve;
    ans := parking(Server.Service.Request);
    if ans = Server.Service.Allow then
        ...
    end Serve;
    
```

### Objects are loosely coupled

Server and Client do not have mutual method calls or field accesses neither directly nor through other objects by transitive closing

### Communicating peers are known

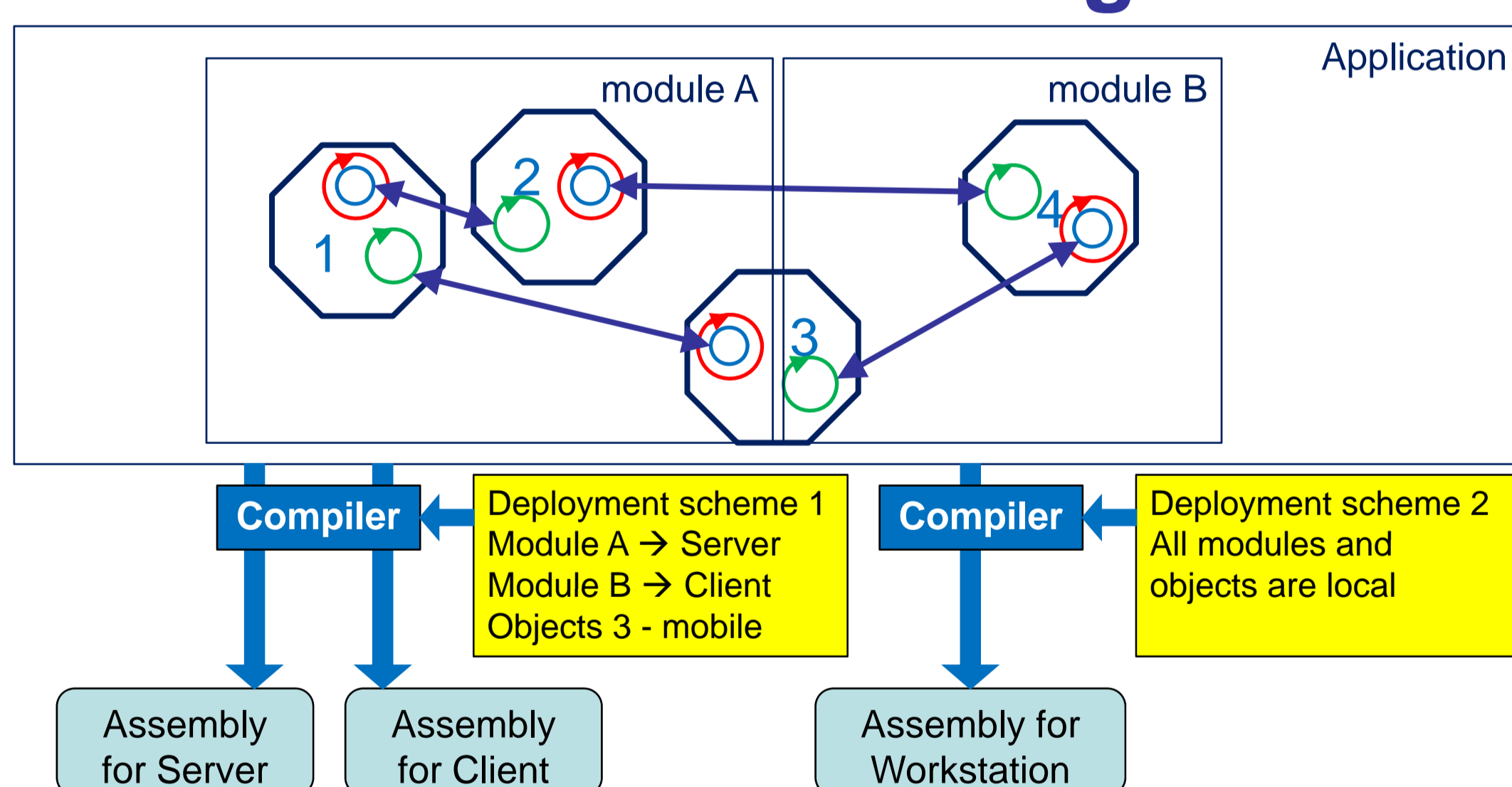
parking can be only a local variable with limited destructive reference transfer or lend opportunities

### Communication is correct

Compile time partial validation of compliance to the protocol

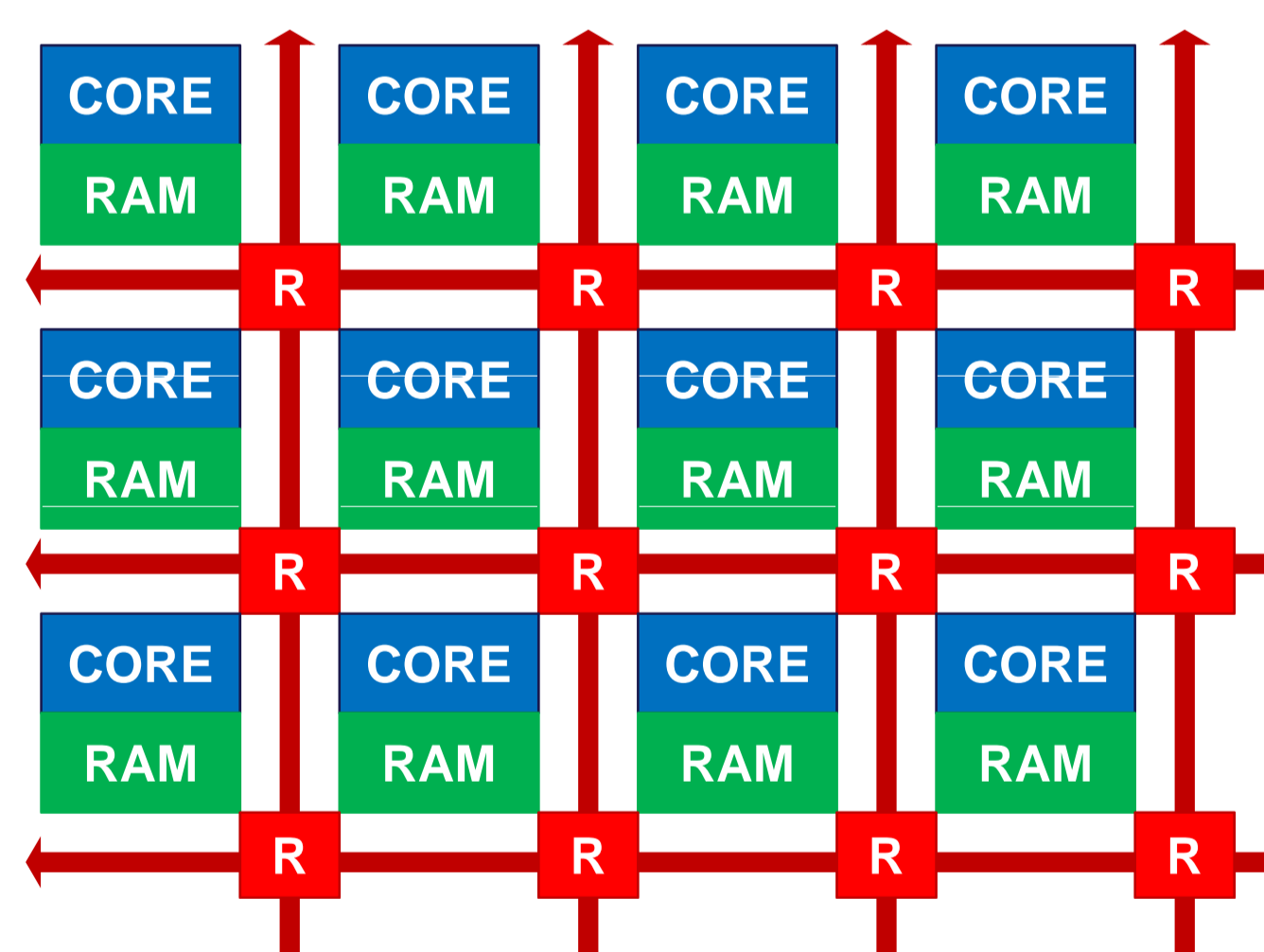
## Target environments

### .NET / .NET Remoting



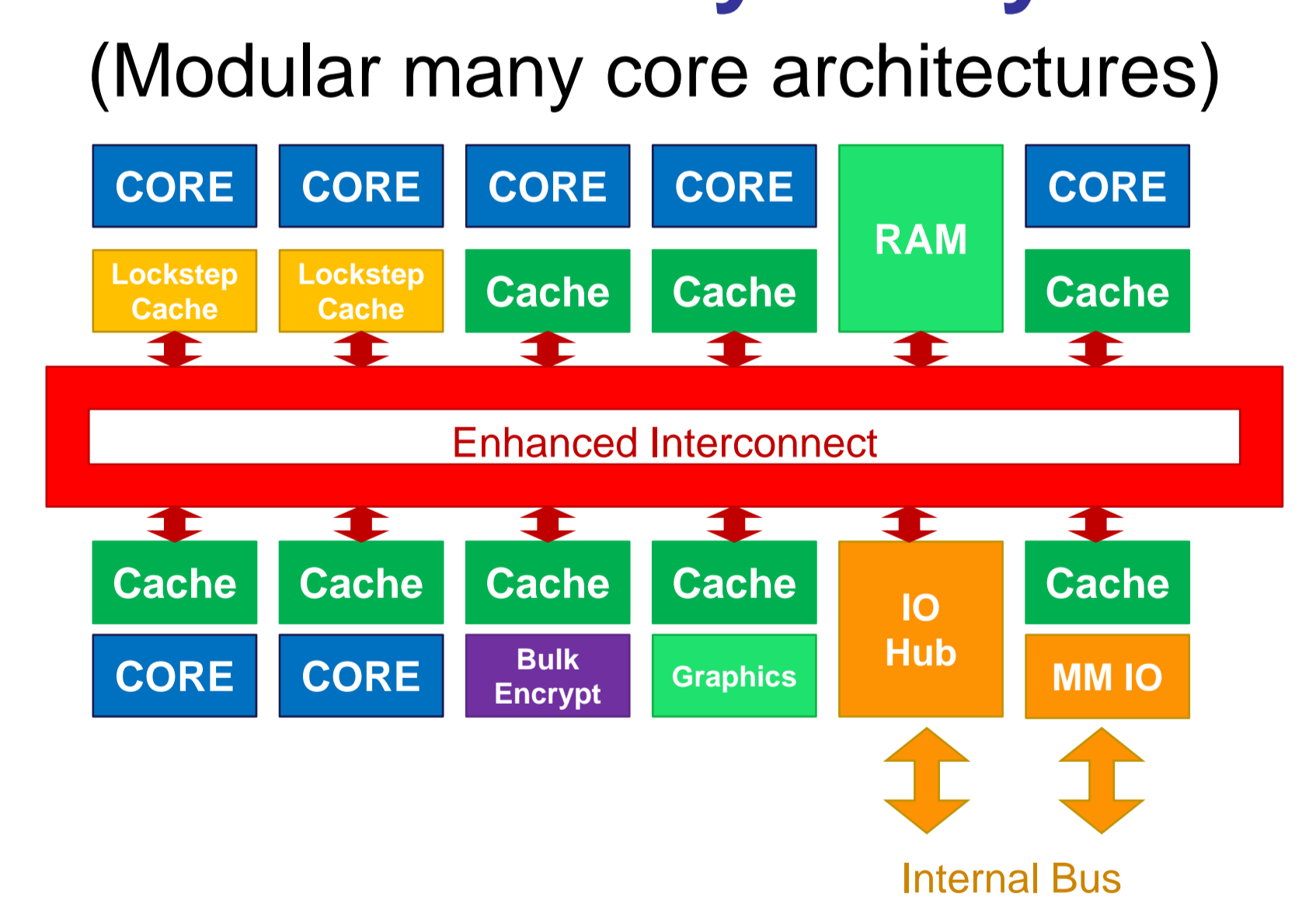
**Goal:** Placing mobile objects close to most active peers by static analysis of protocols

### Array processors (Simple 32, SEAforth)



**Goal:** Placing a group of strong-coupled objects together on a core

### Shared memory many core (Modular many core architectures)



**Goal:** Achieving better locality and hence minimizing trashing of caches

## Compiler architecture

Compiler as a Collection of Resources

