Pragmatic Type Interoperability

S. Baehni, P.Th. Eugster, R. Guerraoui

Distributed Programming Laboratory

P. Altherr

Programming Methods Laboratory

Swiss Federal Institute of Technology (EPFL)

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

LPD Distributed Programming Laboratory

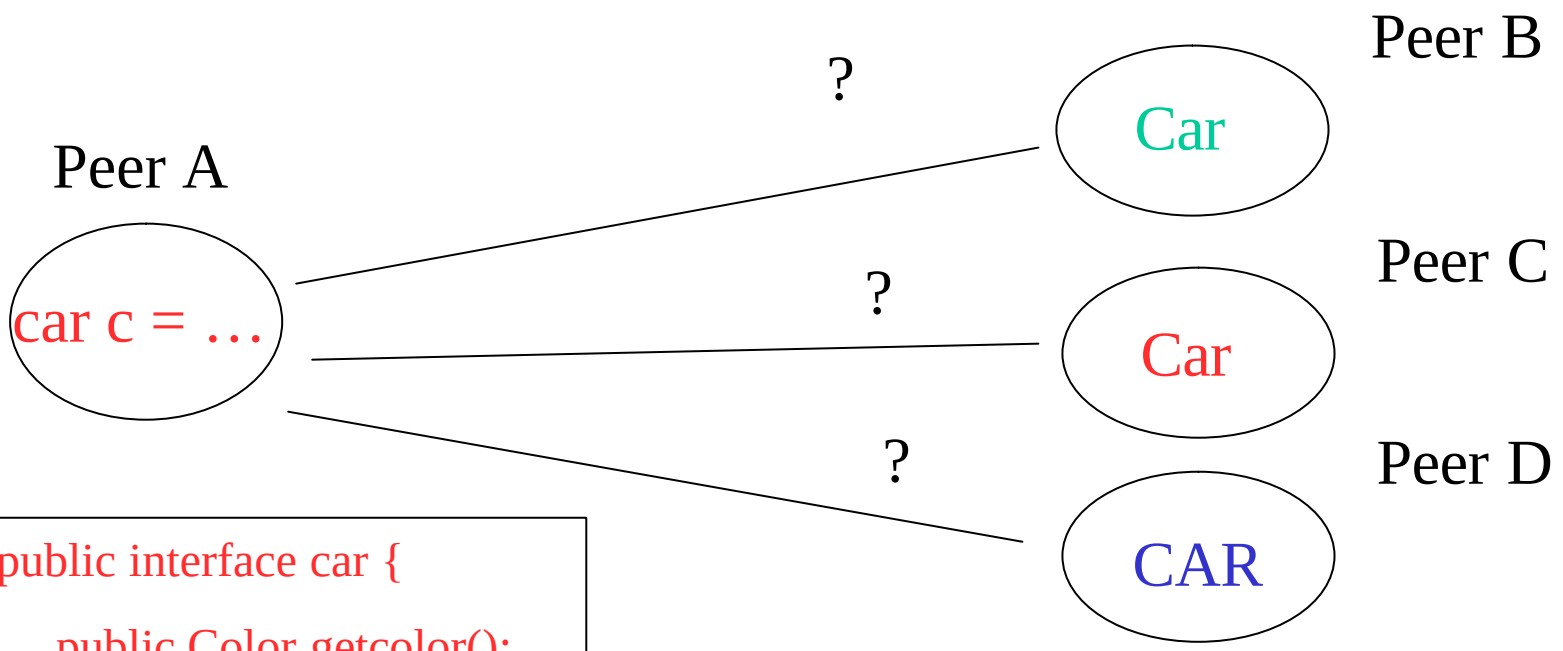
Roadmap

- Context & motivation
- Faces of interoperability
- Object passing semantics & seminal work
- Simplified protocol & implementation issues
- Overhead
- Conclusions & future work

Context & Motivation

- Middleware for peer-to-peer (p2p) object sharing
  - *Borrow/lend* (BL) abstraction, publish/subscribe variant
- Type safety
  - Ideally static within component/processes
- Scalability
  - Abstraction, algorithms
- Interoperability

Illustration

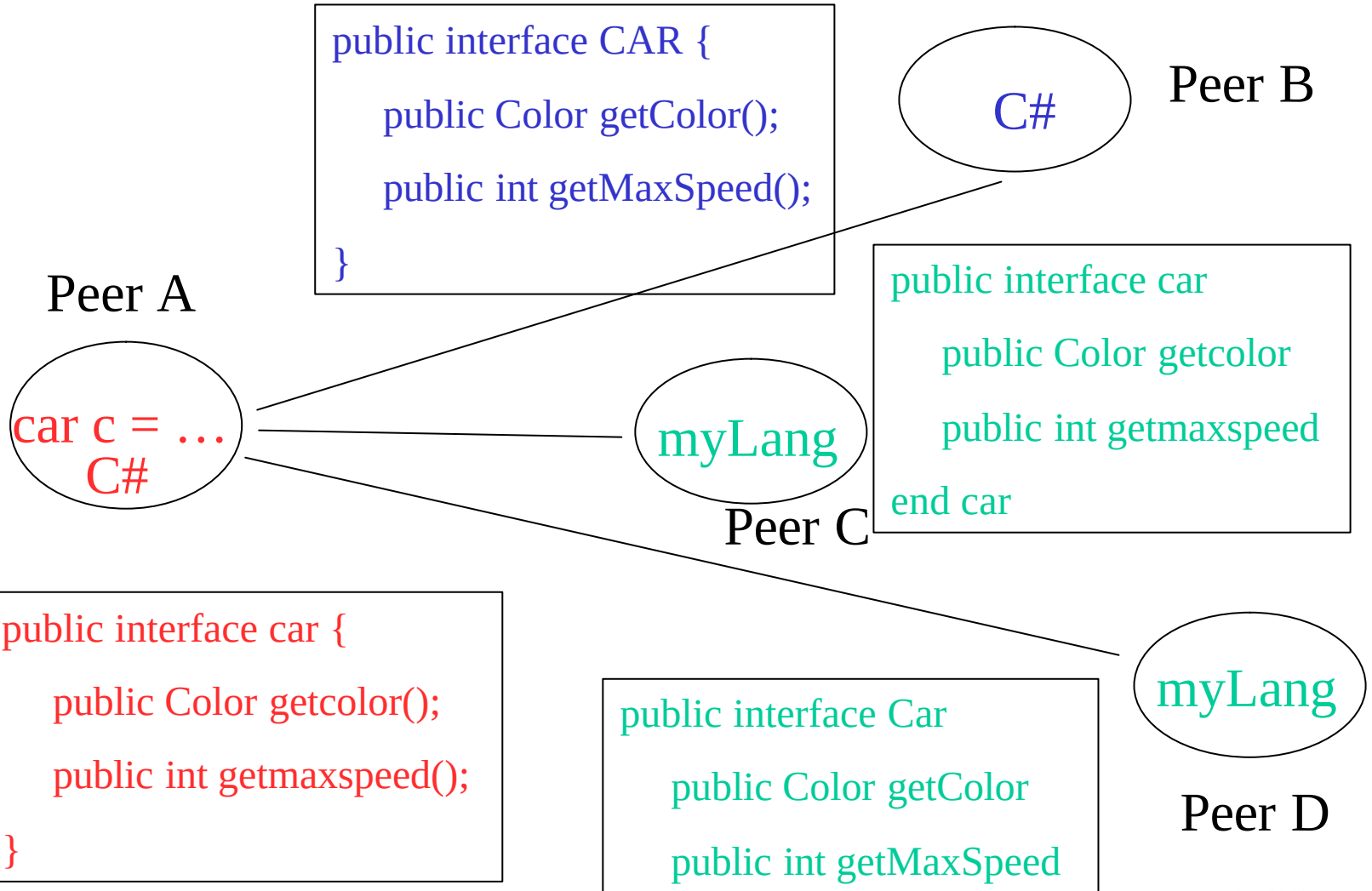



```
public interface car {
    public Color getcolor();
    public int getmaxspeed();
}
```

Interoperability

- LI: “language interoperability”
  - Same type, different languages
  - Usually explicit agreement
- CI: “component interoperability”
  - Similar type, same language
  - cf. *implicit, structural conformance*
    - Single language centralized settings
- LI & CI
  - Similar type, potentially different languages

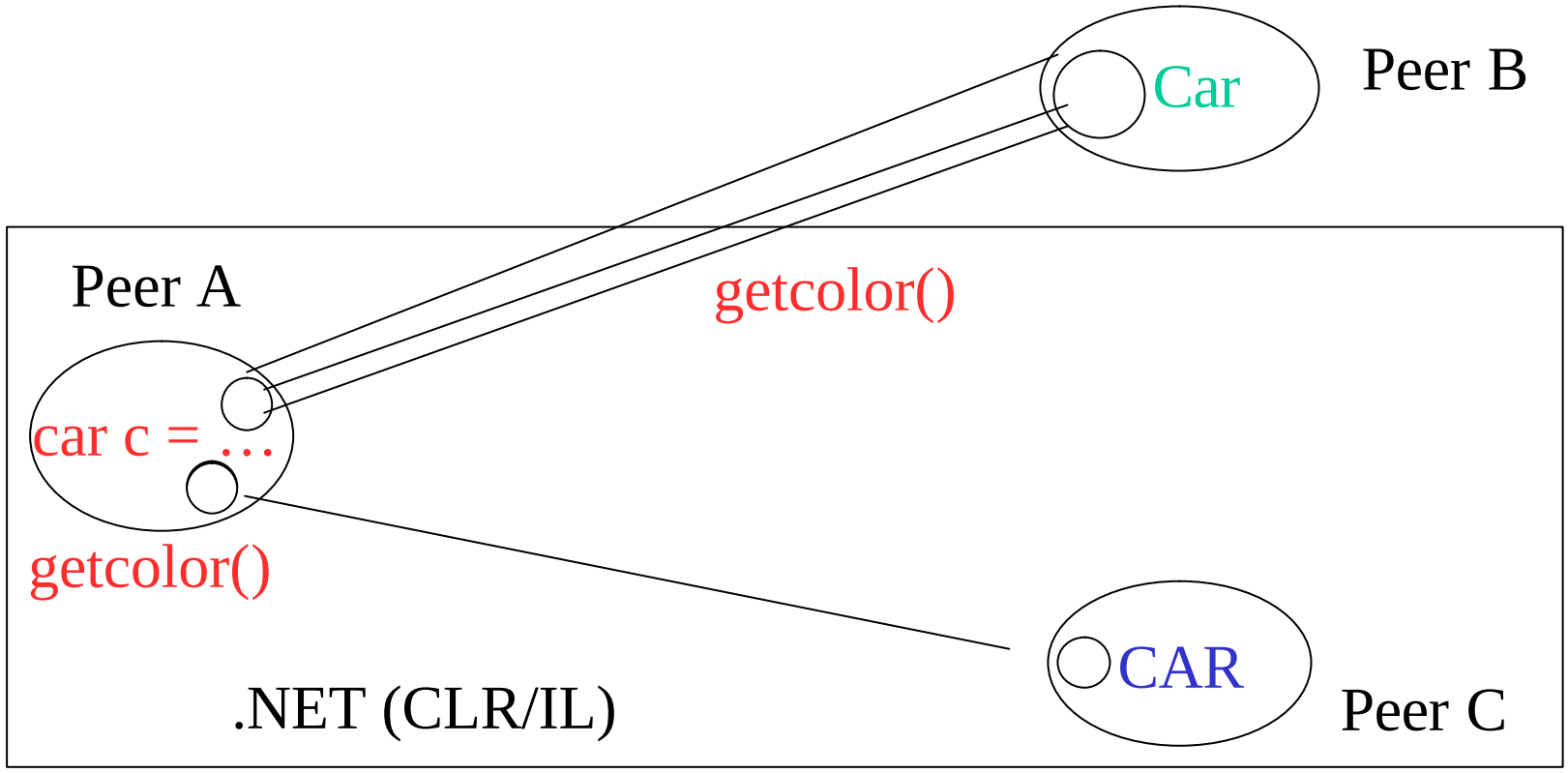
Illustration



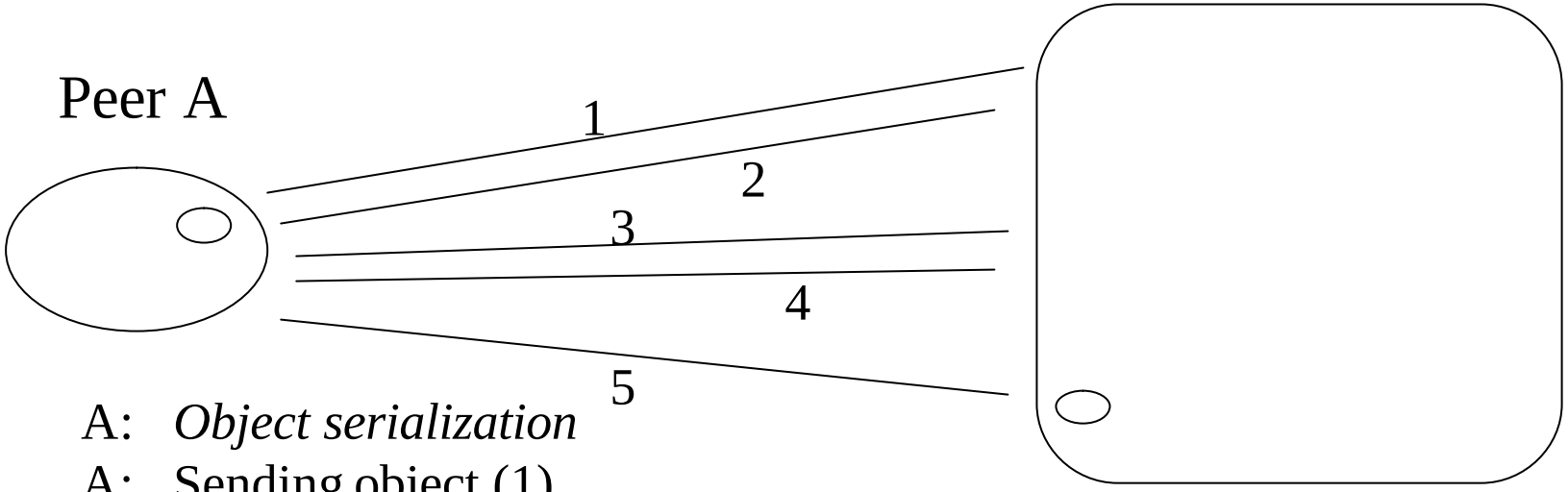
Semantics & Seminal Work

- RS: pass-by-reference semantics
  - Objects invoked remotely (RPC et al.)
  - Proxies and instances of primitive types “transferred”
  - CORBA / .NET (LI), Renaissance [Muckelbauer&Russo] (LI&CI)
- VS: pass-by-value semantics
  - Objects are transferred
  - CORBA (LI limited), .NET (LI)

Illustration



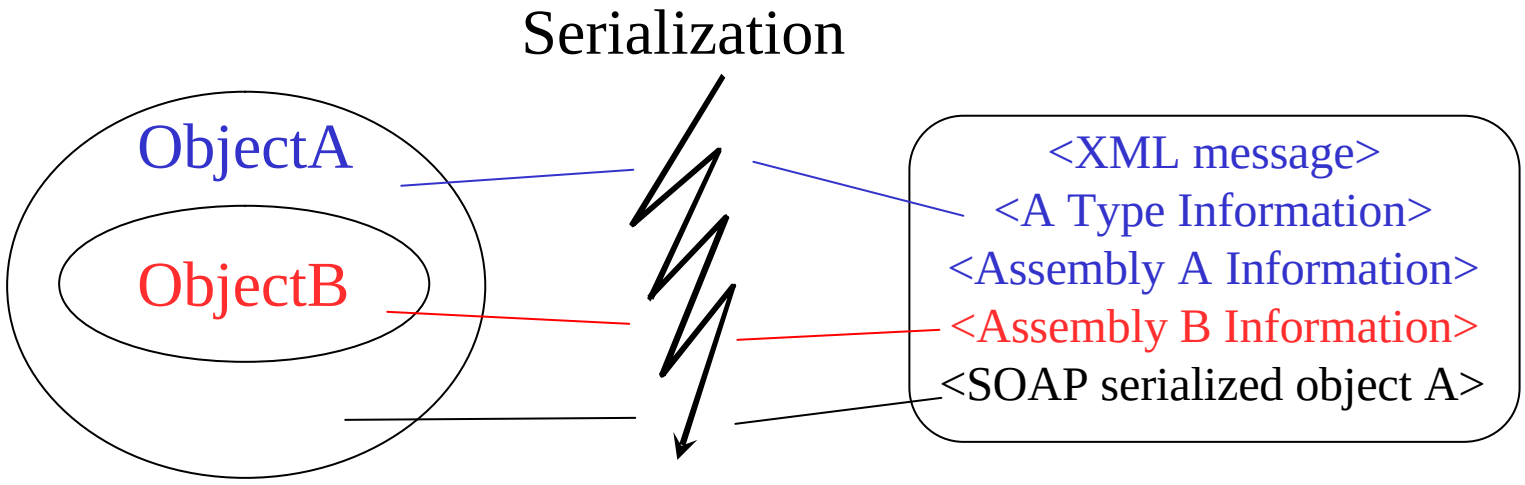
Simple Protocol & Issues (VS)



A: *Object serialization*
A: Sending object (1)
B: Type lookup
B: Asking for new type information (2)
A: Sending *type representation* (3)
B: Applying *type conformance rules*
B: Asking for code, i.e., IL assembly (4)
A: Sending code (5)
B: Object deserialization
B: *Object access*

Object Serialization

- Own XML for *description* of object (type T)
  - Built-in XML serialization only for public fields
  - Binary/SOAP require T at deserialization
- SOAP for object *representation*



Type Representation

- XML serialization with own introspection types
  - Introspection to learn about a type T, but
  - Instances of built-in introspection types (fields, methods, etc.) require type T on target peer

```
public interface CAR {
    public Color getColor();
    public int getMaxSpeed();
}
```

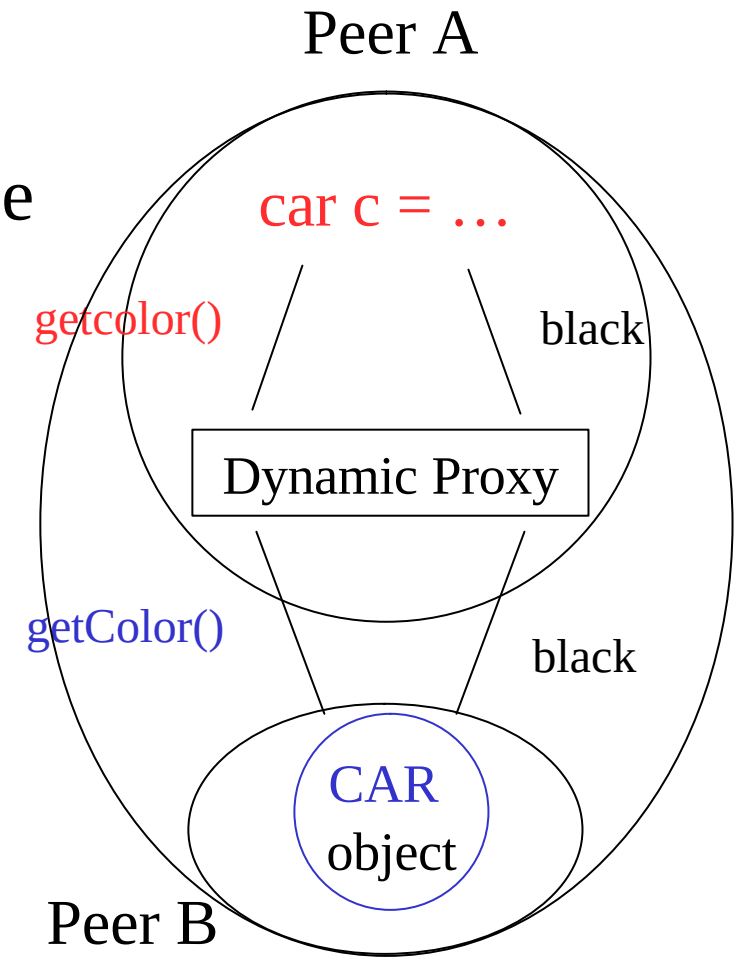
```
<XML message>
<TypeDescription
<classes> <classInfo name="CAR"/></classes>
 <superClasses></superClasses>
 <fields></fields>
 <methods>
        <methodInfo name="getMaxSpeed"
          modifier="public" returnType="int"/>
        …
 </methods>
 <constructors></constructors>
</TypeDescription>
```

Conformance Rules (Informal)

- T conforms to T' explicitly or implicitly
- T implicitly conforms to T' iff
  - Every method in T' (supertypes) finds a method in T (supertypes):
    - Same name (case-insensitive)
    - Parameter types (same order) conform explicitly or implicitly
  - Every field in T' (st) finds a field in T (st):
    - Same name (case-insensitive)
    - Field types conform explicitly or implicitly
- Default (no-argument) constructors

Object Access

- Dynamic proxies
  - Pass-by-reference
  - Pass-by-value



Overhead

Simple types (no recursion)

- Object serialization:
  - Serialization: 16.68 [ms]
  - Deserialization: 1.32 [ms]
- Type representation:
  - Creation and serialization: 6.14 [ms]
  - Deserialization: 2.34 [ms]
- Conformance rules application: 12.66 [ms]
- Object access:
  - Direct invocation: 0.0000142 [ms]
  - Indirect invocation (dynamic proxy): 0.03 [ms]

Conclusions & Future Work

- .NET provides many useful mechanisms
  - Ready-to-use for many scenarios in LI (RS & VS)
  - More effort for achieving CI as not declared goal
  - Reflection costly, need “deeper” integration
- Using conformance (“subtyping”) relationships
  - Object routing
  - Membership management
- Variations on “type conformance”
  - More precise/flexible rules
  - Behavioral conformance for avoiding accidental conformance

Questions and Answers