

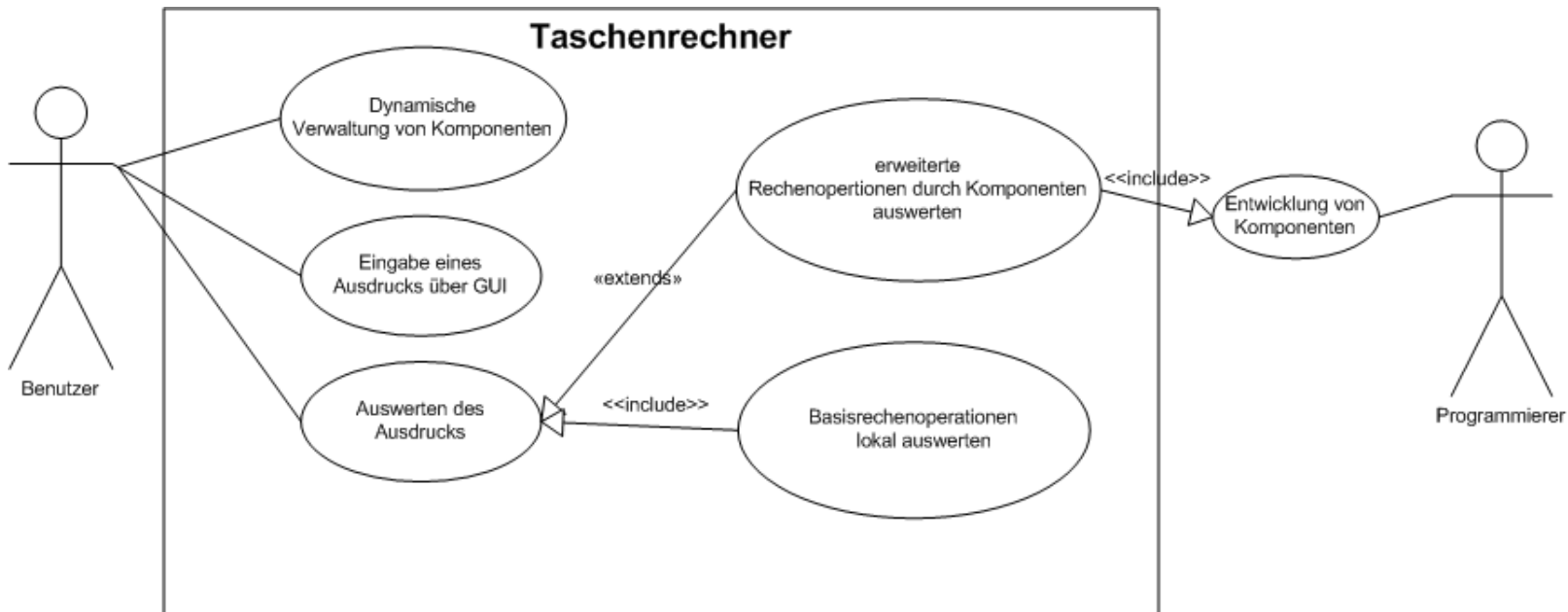
Komponentenbasierter Taschenrechner .NET-Framework

Rami Eid-Sabbagh, Ingmar Lemke,
Stephan Müller, Christian Tinnefeld

Inhaltsübersicht

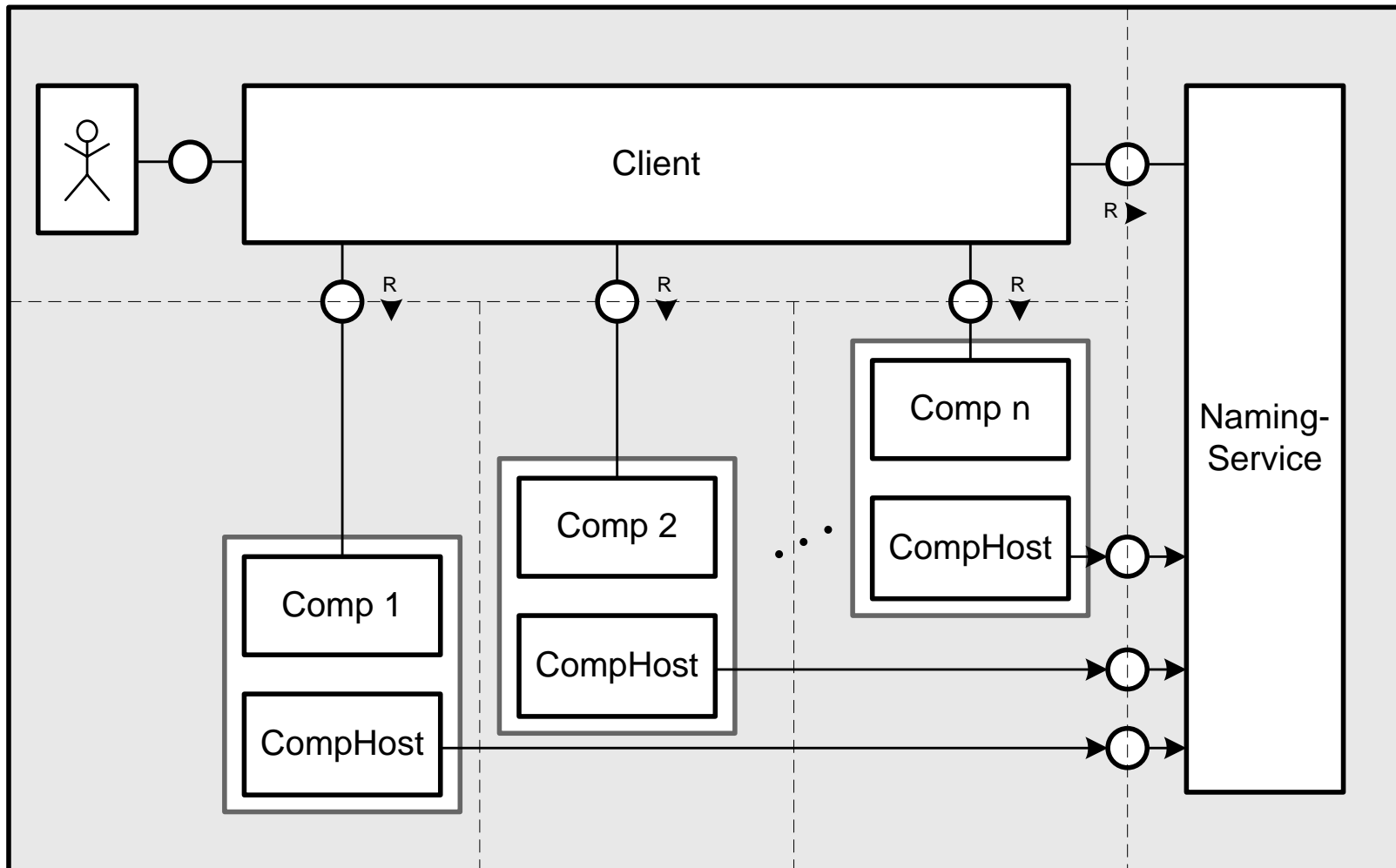
- Anforderungsdefinition
- Architektur
 - Client / coco/R
 - NamingService
 - Komponenten
- Ablauf
- Implementation / Tests
- Live Demo
- Frameworks im Vergleich
- Erweiterungen...

Motivation / Anforderungsdef.



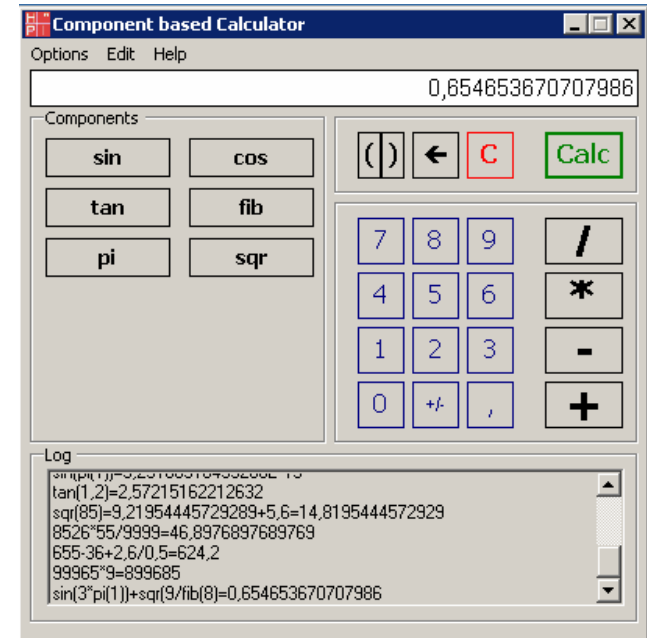
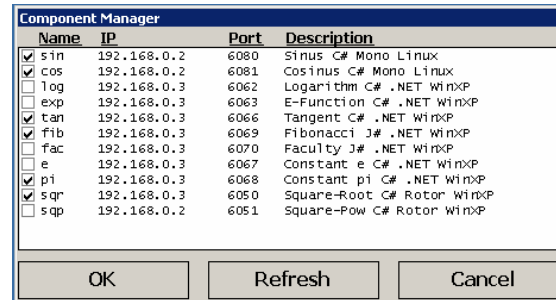
- Taschenrechner mit Grundfunktionen
- Komponenten dynamisch hinzufügen / entfernen
- Interoperabilität
- Erweiterbarkeit

Serviceorientierte Architektur

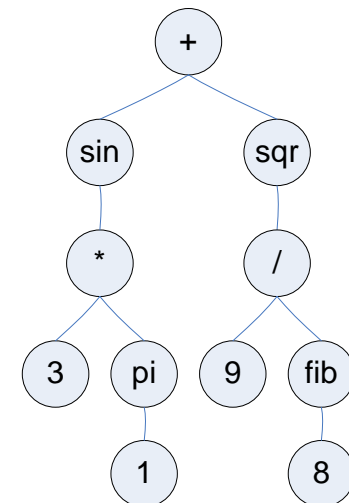


Client

- Eingabe über GUI
- Auswertung der Ausdrücke mit Scanner und Parser
- Basisrechenoperationen vom Client berechnet
- .NET Remoting: Komponenten werden dynamisch geladen, und können dann zur Berechnung verwendet werden



$$\sin(3 \cdot \pi(1)) + \text{sqr}(9 / \text{fib}(8))$$



Grammatik mit coco/R

COMPILER Calc

CHARACTERS

```
letter = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz".
digit = "0123456789".
```

TOKENS

```
ident = letter {letter | digit} .
number = digit {digit}[".", digit {digit}] .
```

PRODUCTIONS

```
Calc                                     (. double expr; .)
=
Expr <out expr> .
```

```
Expr <out double expr>                 (. expr=0; double term; .)
=
Term <out term>                         (. expr = term; .)
{ '+' Term <out term>                   (. expr += term; .)
  | '-' Term <out term>                  (. expr -= term; .)
} .
```

[...]

```
Fact <out double factor>                (. factor = 0; double number;
                                         double expr; string name; .)
```

```
=
Number <out number>                     (. factor = number;. )
| '(' Expr <out expr> ')'                 (. factor = expr; .)
| Ident <out name> '(' Expr <out expr>')' (. Component c = (Component)Activator.GetObject [...].
                                         factor = c.calculate( expr); .)
```

```
Ident <out string name>
= ident                                  (. name = t.val; .)
```

```
Number <out double number>
= number                                  (. number = Convert.ToDouble(t.val); .)
```

END Calc.

- kontextfreie LL(1) Grammatik mit semantischen Erweiterungen

- Auszug aus Parser.cs File

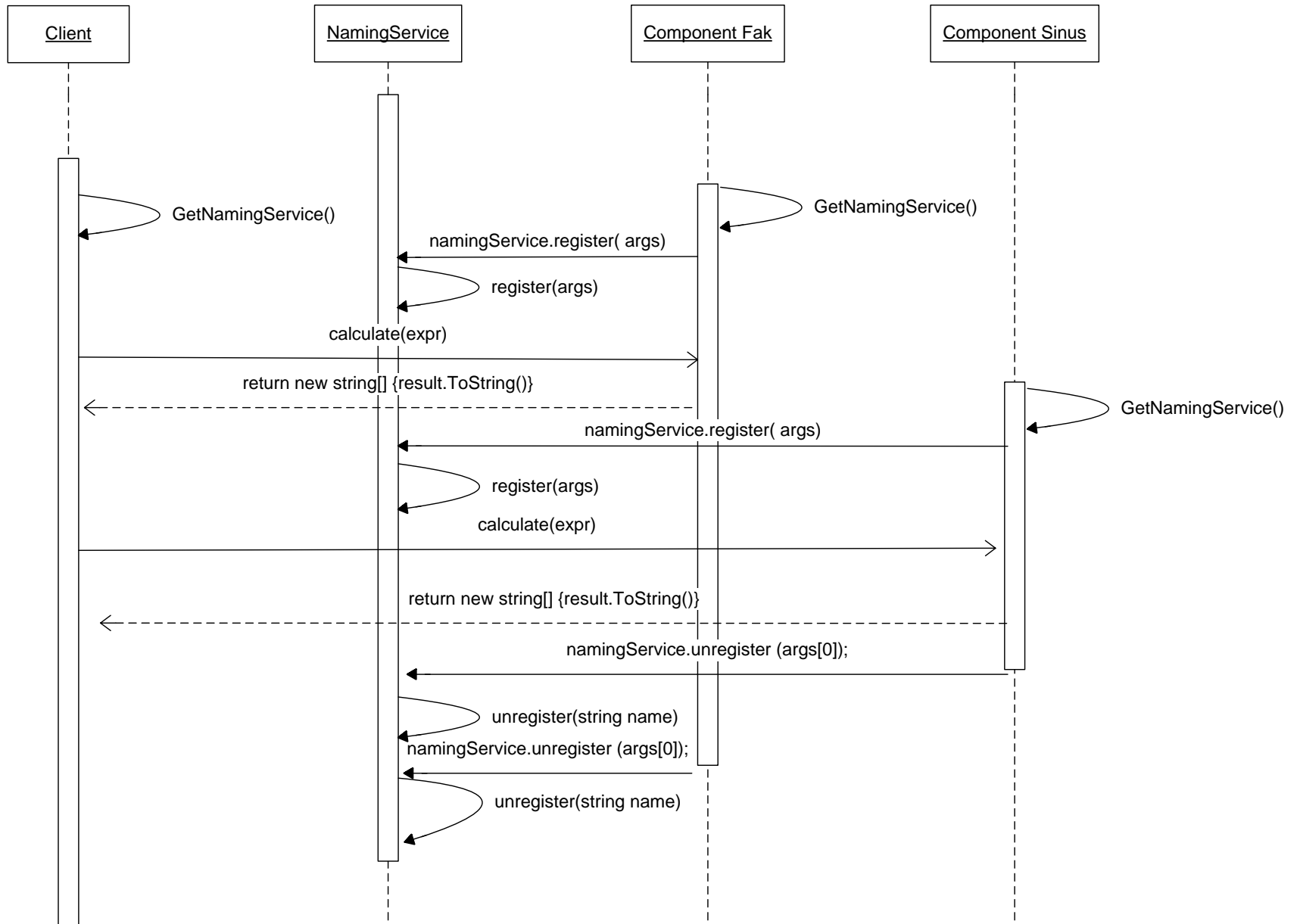
```
static void Expr(out double expr) {
    expr=0; double term;
    Term(out term);
    expr = term;
    while (la.kind == 3 || la.kind == 4) {
        if (la.kind == 3) {
            Get();
            Term(out term);
            expr += term;
        } else {
            Get();
            Term(out term);
            expr -= term;
        }
    }
}
```

NamingService

- Selbst implementiert mit .NET Remoting Technologie
- Methoden `register()` und `unregister()`
- Metadatenformat: ComponentInformation enthält alle Infos über Komponenten
- Client kann Komponenteninfos auslesen und dann Komponenten laden

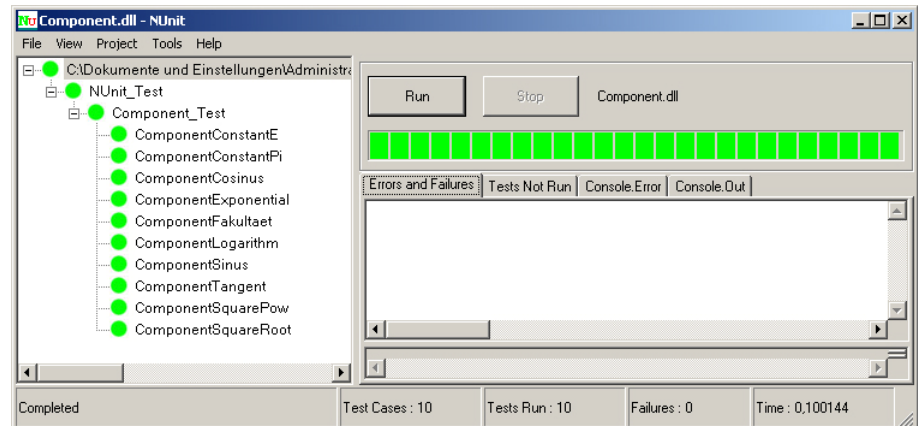
Komponenten

- Generischer ComponentHost wird für jede Komponente exemplarisiert
- Registrierung beim NamingService:
`Componenthost.exe <Name> <Type> <IP>`
`<Port> <Description>`
- Abstrakte Klasse Component
- Berechnung mittels polymorphen Ausdruck:
`Component.calculate()`



Implementation / Tests

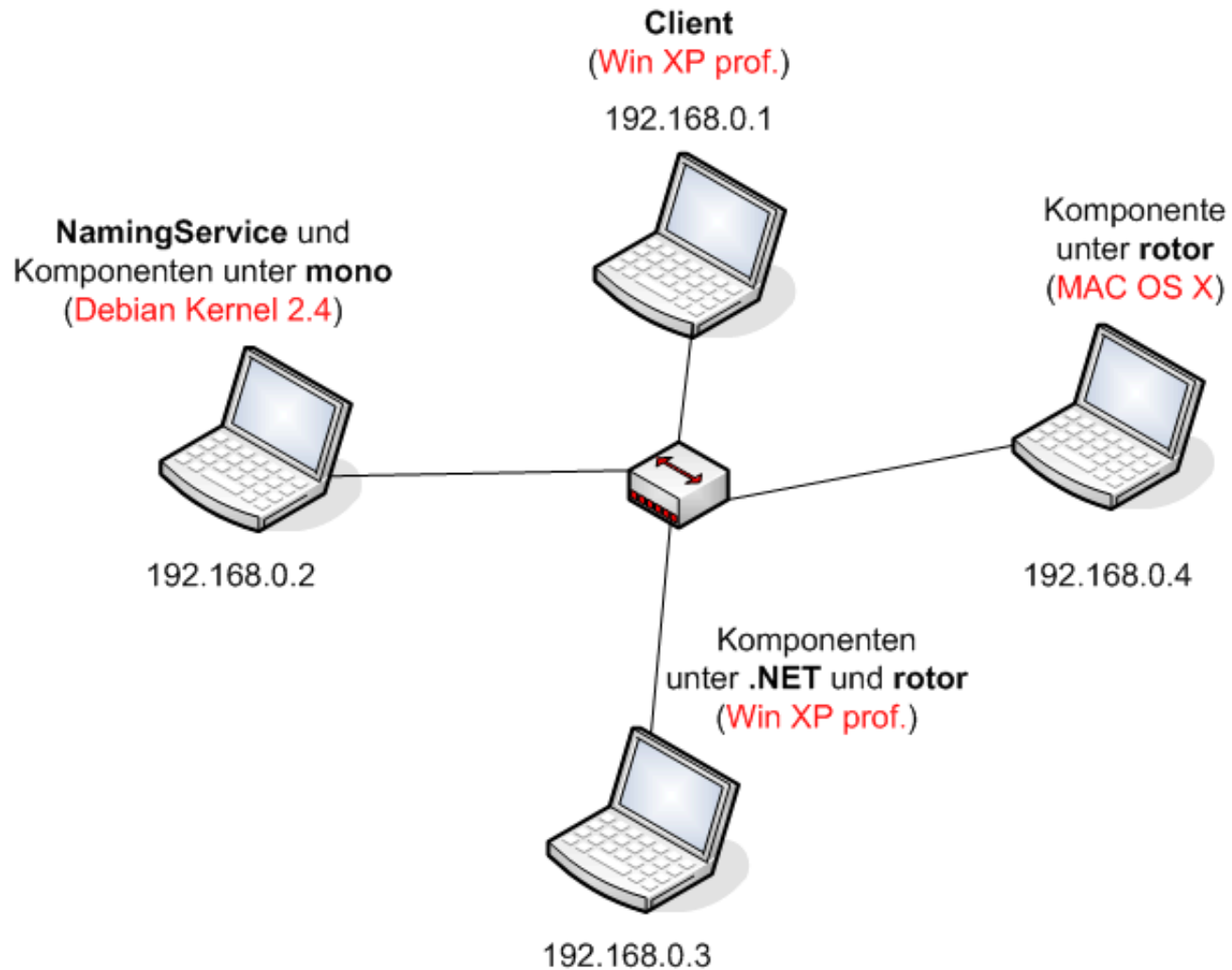
- Entwicklungsablauf
- Quellcodeverwaltung nicht benutzt
- Komponententest mit NUnit
- Dann GUI aufgesetzt
- Priorität auf Funktionalität



Mono und Rotor

- Implementationen der ECMA-334 und ECMA-335 Standards
- Kommandozeilenkompiler
- Nicht alle Klassen verfügbar
- Momentan nur bedingte Kompatibilität
- Nur ComponentHost kompiliert

Live-Demo



Vergleich Frameworks

	.NET	J2EE	CORBA	COM+
location transparency	Ja	Ja	Ja	Ja
platform Independence	Nein	Ja	Ja	Nein
language independence	Ja	Nein	Ja	Ja
runtime Enviroment	Ja	Ja	Nein	Nein
services usability	Servicenutzung erfordert(viel) Coding	Nutzen von Diensten über Konfiguration	Servicenutzung erfordert Coding	Servicenutzung erfordert Coding

Ausblick...

- Erweiterte Fehlerbehandlung
- Komponententauthentifizierung
- Exklusive Komponentenreservierung

Literatur

- Mark Strawmyer (2002): „.NET Remoting“
- Rammer, Ingo (2002): „Advanced .NET Remoting“
- dot.net magazin (2004): „ Service-orientierte Architekturen“
- dot.net magazin (2004): „ Rahmenbedingungen“
- <http://www.go-mono.com>
- <http://www.ssw.uni-linz.ac.at/Research/Projects/Coco/>
- <http://msdn.microsoft.com/net/sscli>
- <http://www.nunit.org>