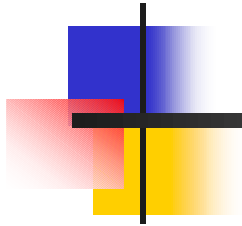
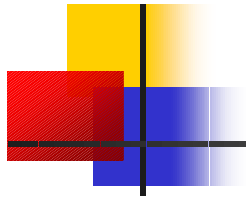


Verteidigung Diplomarbeit

Christian Wurbs



Modellierung .NET basierter Anwendungssysteme mit der UML



Aufgabenstellung / Ziel

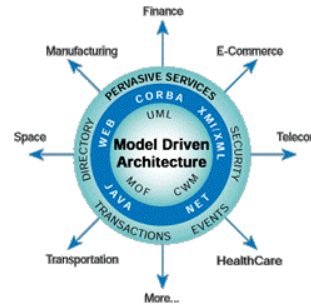
Erstellung eines Konzeptes zur Modellierung von .NET basierten Anwendungen mittels der UML

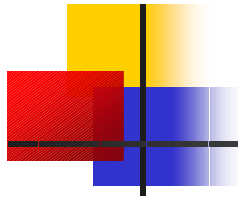
Teilaufgaben:

- Gegenüberstellung .NET-Komponenten vs. EJB-Komponenten
- Einordnung in aktuelle OMG-Bemühungen
- Erstellung eines UML Profile für .NET
- Abbildung des Profiles auf C#
- Nachweis der Anwendbarkeit anhand eines Onlineshop

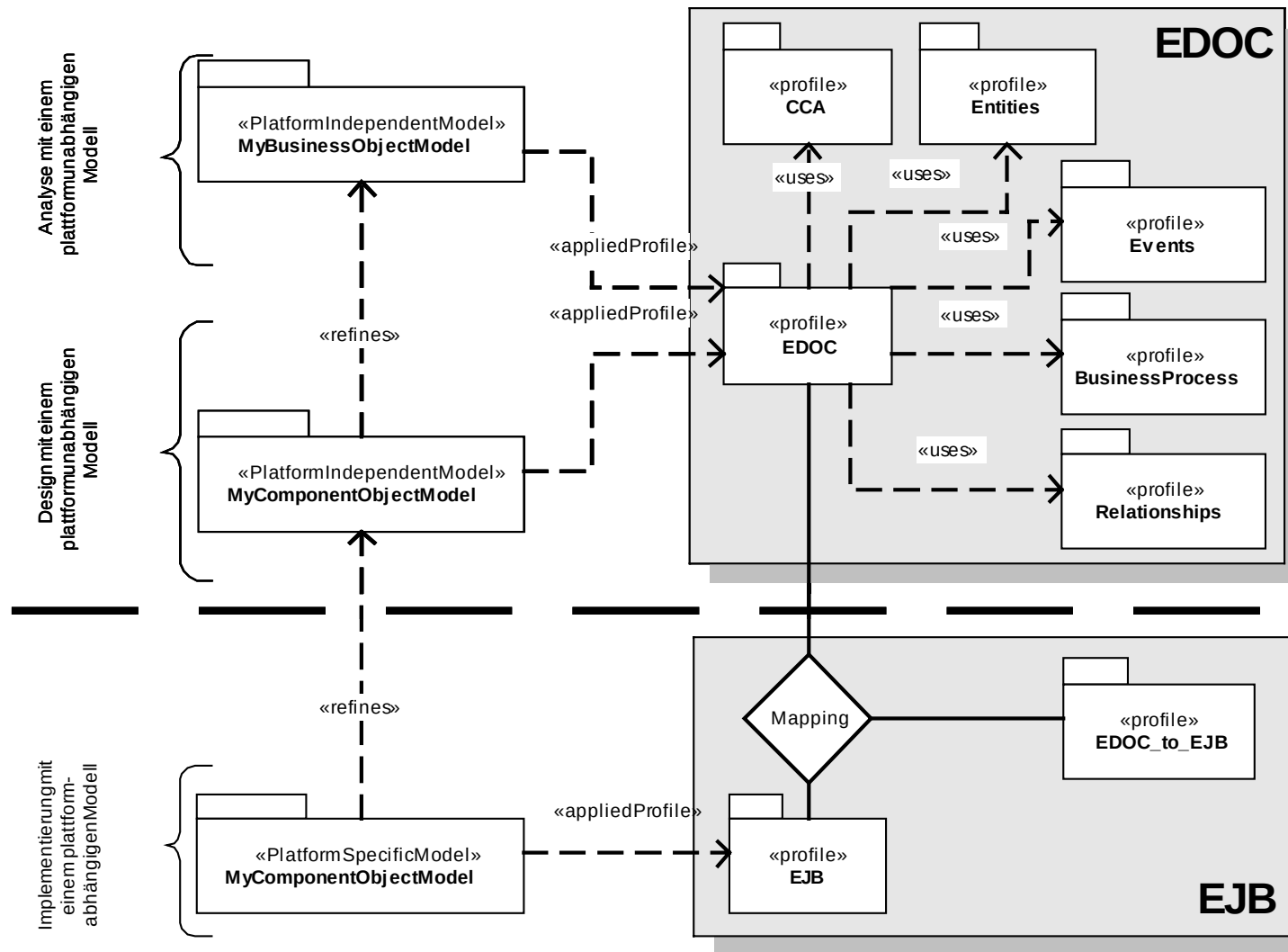
Überblick

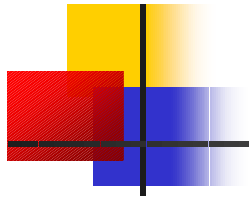
- OMG Bemühungen
- .NET Profile
- EDOC Abbildung
- Onlineshop
- Ergebnis
- Ausblick





OMG Bemühungen

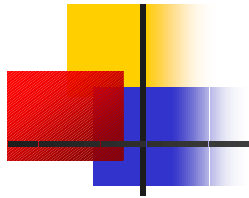




Anforderungen

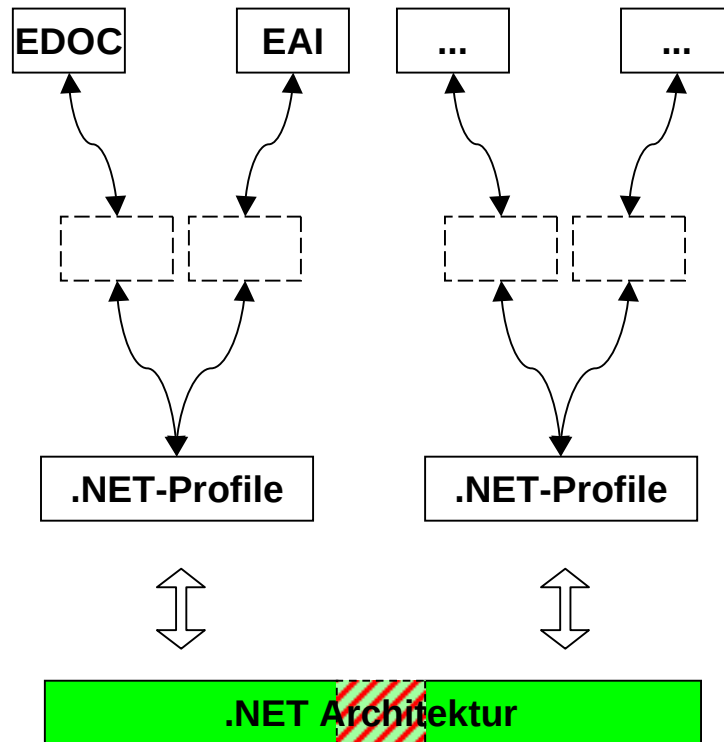
- Allgemein (.NET-Profile):
 - *allgemeine Anwendbarkeit gewährleisten*
 - *nicht auf einen Anwendungsbereich beschränkt sein*
 - *Unterstützung des Softwareentwicklungsprozesses in mehreren Bereichen*

- Speziell (EDOC → .NET):
 - *Möglichkeiten der .NET Plattform bestmöglich nutzen*
 - *Programmiersprachenunabhängigkeit im Rahmen von .NET gewährleisten*
 - *Verwendung von .NET-Attributen*

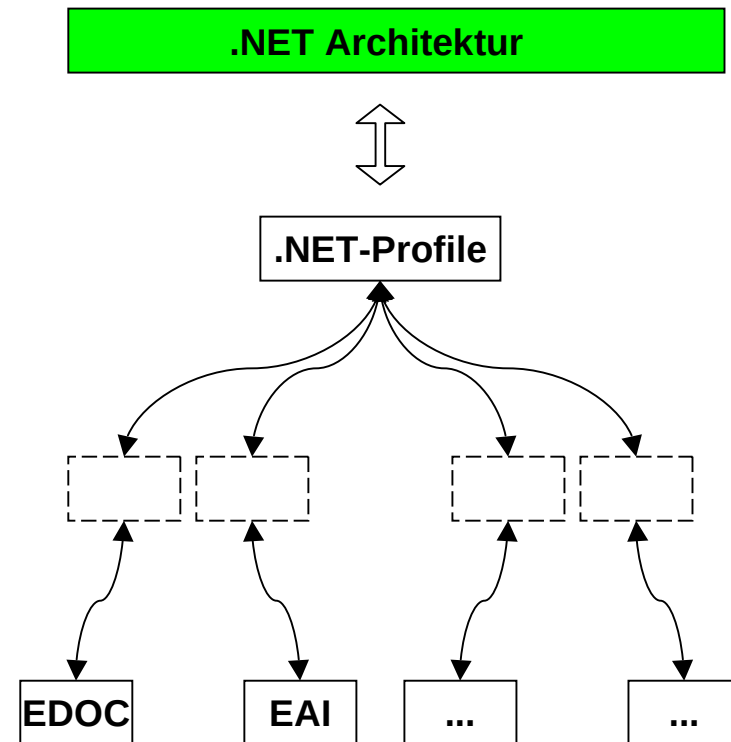


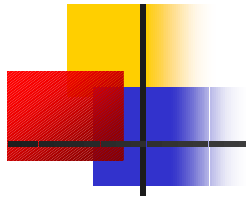
.NET Profile Ansatz

„top down“



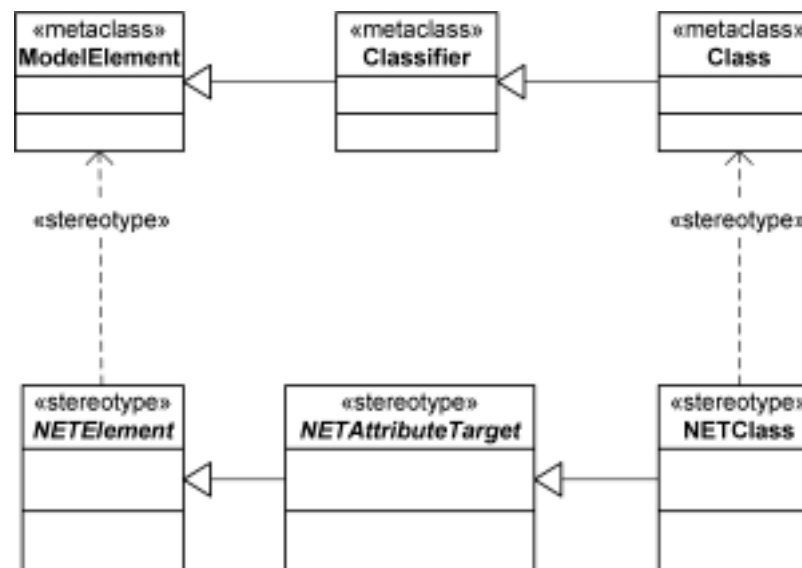
„bottom up“

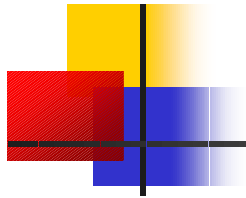




.NET Profile Umsetzung

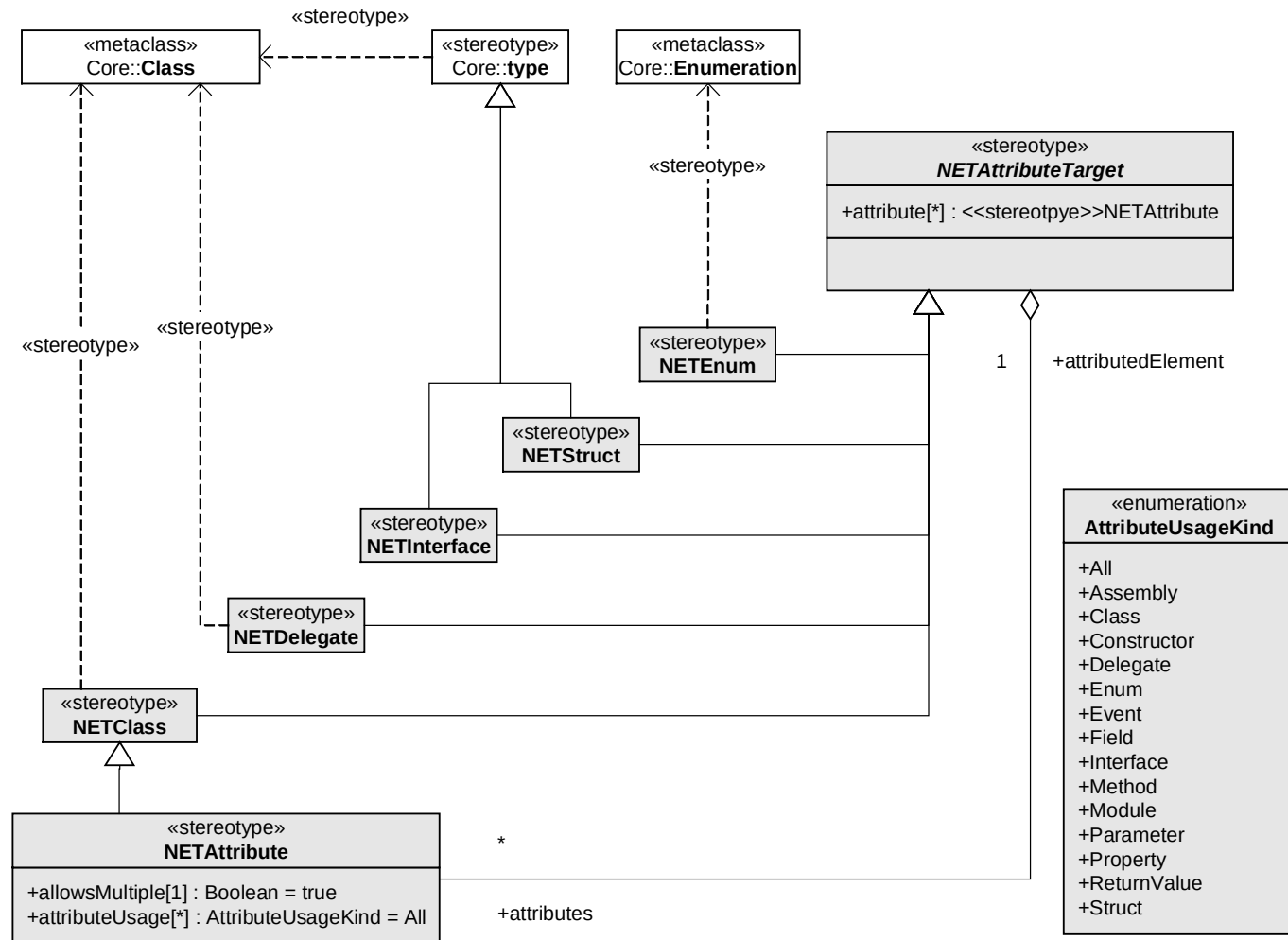
- Beachtung grundlegender Eigenschaften (Attribuierbarkeit von .NET-Elementen)
- Anhand C# Spezifikation (ECMA 334)
- Abbildung der Sprachelemente (Klasse, Methode, Properties ...)
→ UML-Metamodellelemente





.NET Profile Umsetzung

- i.W. direkte Abbildung möglich
- einige Ausnahmen:
 - Interface
 - Namespace
- Problempunkte:
 - Attribute
 - Property



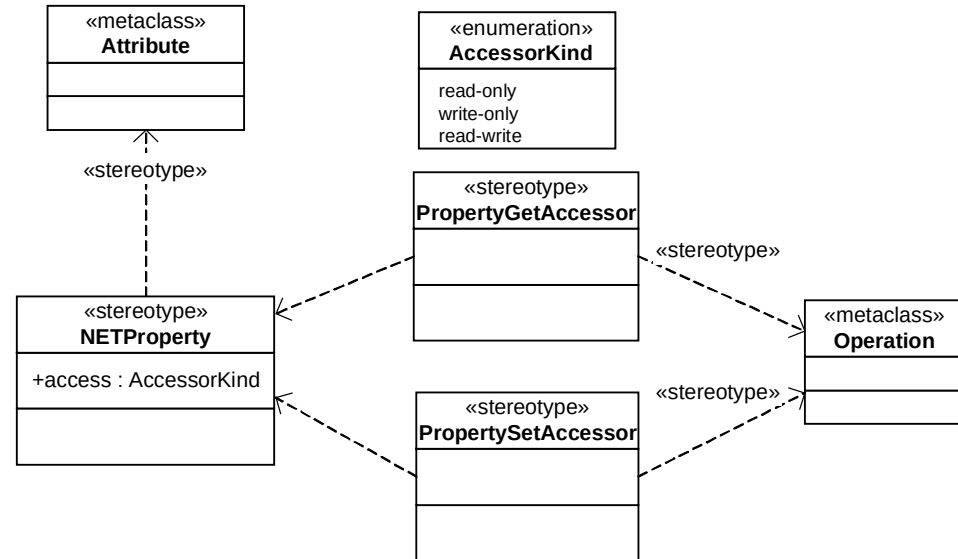
Ausschnitt .NET-Classfier

Problempunkte

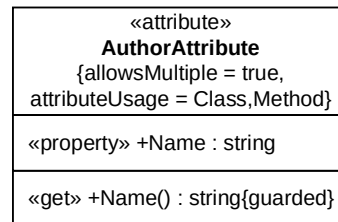
Property

```
class Ticker
{
    public string Symbol;

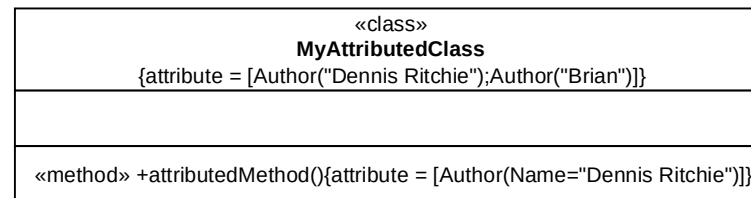
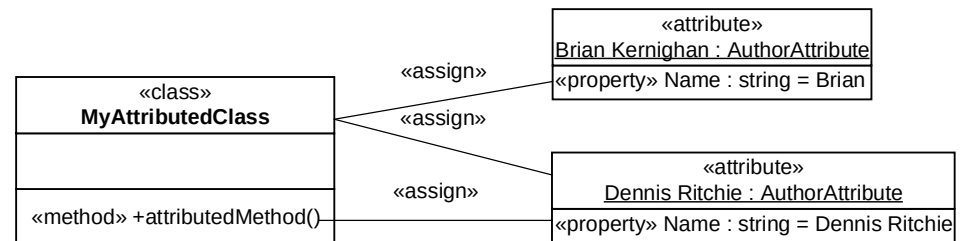
    public decimal Value
    {
        get
        {
            TickerW3Svc s = new TickerW3Svc();
            return s.getValue( this.Symbol );
        }
    }
}
```



Attribute:



```
[Author("Dennis Ritchie");
Author("Brian")]
class MyAttributedClass
{
    [Author(Name="Dennis Ritchie")]
    public void attributedMethod()
    {
        ...
    }
}
```



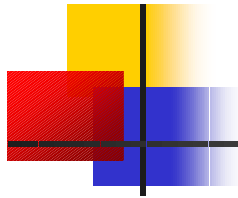
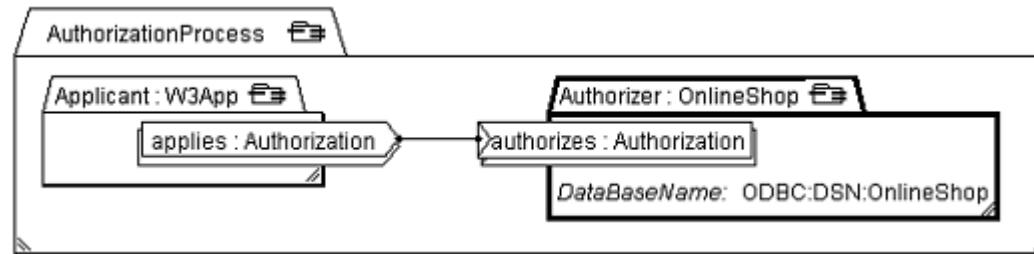
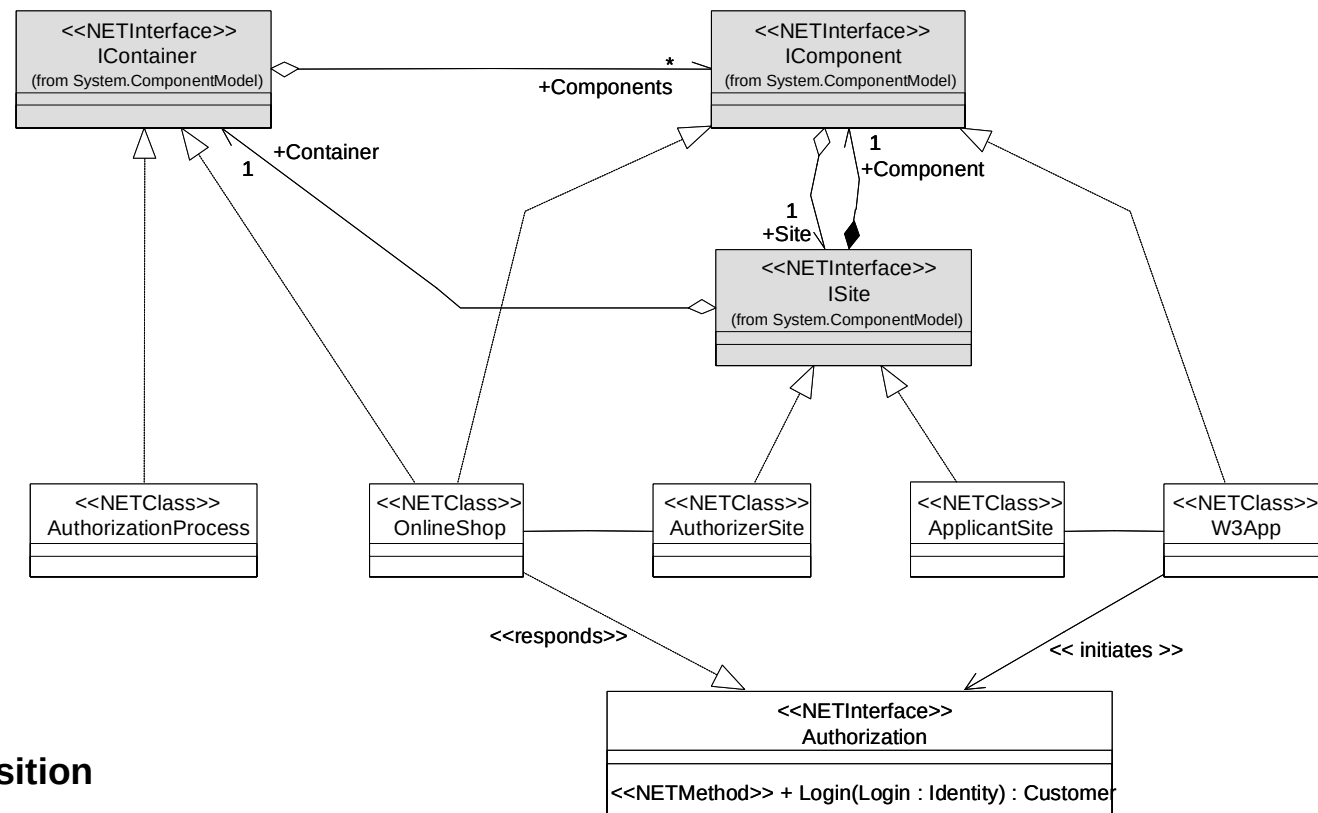


Abbildung EDOC → .NET

EDOC



.NET



Beispiel: EDOC-Composition

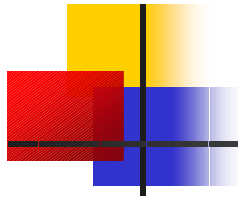


-
- ```
classDiagram
 class CustomerMgmt {
 <<interface>>
 ICustomerManagement
 +Customers: CustomerCollection
 +createCustomer: Customer
 +findCustomer: Customer
 +removeCustomer: void
 }
 class OrderMgmt {
 <<interface>>
 IOrderManagement
 +Orders: OrderCollection
 +FindOrders: Collection
 +PlaceOrder: void
 }
 class ProductManagement {
 <<interface>>
 IProductManagement
 +Categories: ProductCategoryColl
 +Products: ProductCollection
 +ProductDetails: ProductItemColl
 +MinPrice: decimal
 }
 class Address {
 <<datatype>>
 +Street: string
 +Street2: string
 +ZIP: int
 +Town: string
 +Region: string
 +Country: string
 +isBilling: boolean
 +isShipping: boolean
 }
 class Customer {
 <<core>>
 +Adresses: AddressColl
 +ID: int
 +Firstname: string
 +Lastname: string
 +Birthdate: DateTime
 }
 class Order {
 <<core>>
 +ID: int
 +Date: DateTime
 +BillingAddress: Address
 +ShippingAddress: Address
 }
 class ProductCategory {
 <<type>>
 +Name: string
 +Description: string
 }
 class ProductItem {
 <<type>>
 +ID: int
 +Size: int
 +Color: Color
 +Price: decimal
 }
 class Product {
 <<core>>
 +Name: string
 +Description: string
 }
 class OrderItem {
 <<type>>
 +Amount: int
 }

 CustomerMgmt --|> Customer
 OrderMgmt --|> Order
 ProductManagement --|> ProductCategory
 ProductManagement --|> ProductItem
 ProductManagement --|> Product

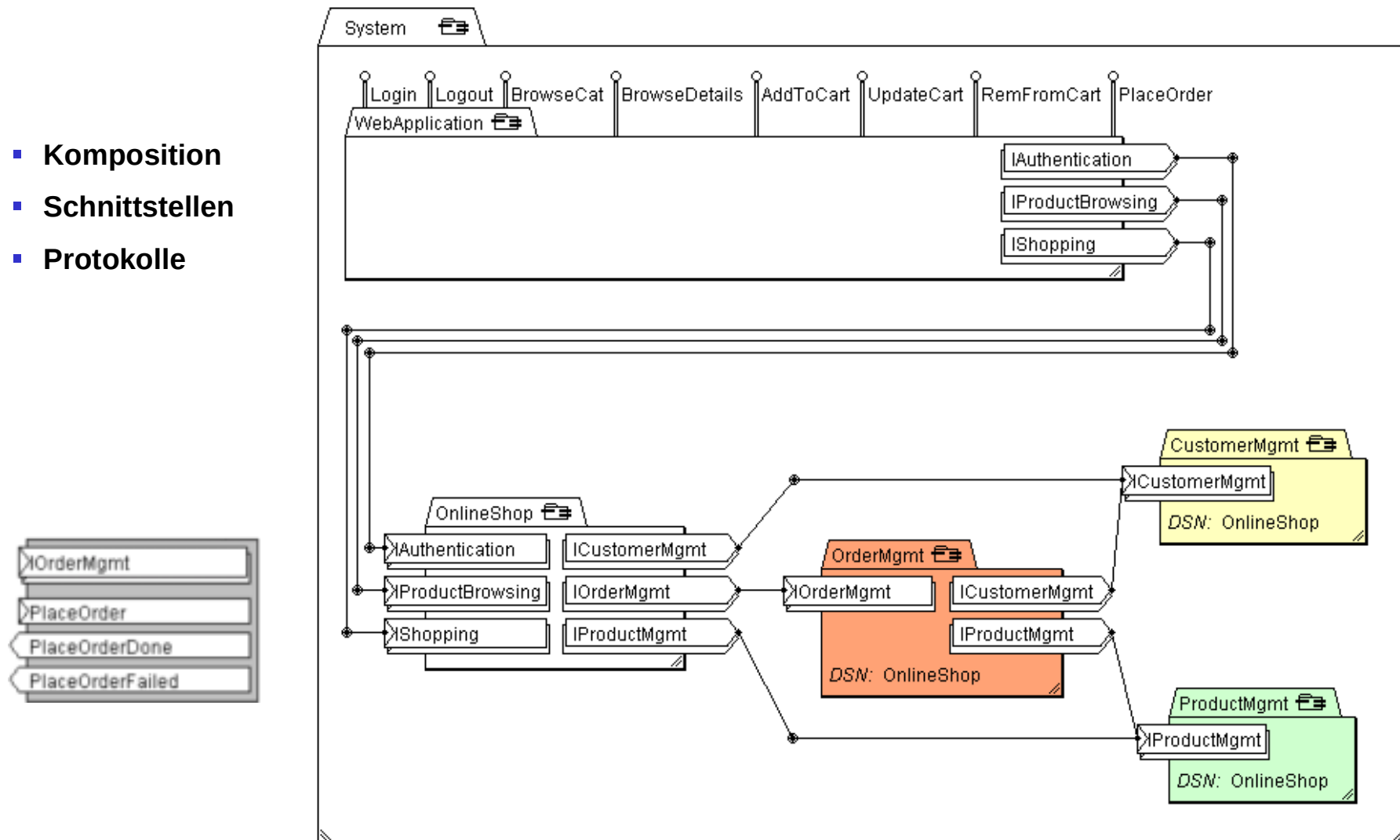
 Customer -- "*" Order : Orderer
 Order -- "1..*" OrderItem
 ProductCategory -- "*" ProductItem
 Product -- "*" ProductItem
 ProductItem -- "1..*" OrderItem
```
- CustomerMgmt**
- Interface: ICustomerManagement**
    - +Customers: CustomerCollection
    - +createCustomer: Customer
    - +findCustomer: Customer
    - +removeCustomer: void
  - Core: Customer**
    - +Adresses: AddressColl
    - +ID: int
    - +Firstname: string
    - +Lastname: string
    - +Birthdate: DateTime
- OrderMgmt**
- Interface: IOrderManagement**
    - +Orders: OrderCollection
    - +FindOrders: Collection
    - +PlaceOrder: void
  - Core: Order**
    - +ID: int
    - +Date: DateTime
    - +BillingAddress: Address
    - +ShippingAddress: Address
  - Type: OrderItem**
    - +Amount: int
- ProductManagement**
- Interface: IProductManagement**
    - +Categories: ProductCategoryColl
    - +Products: ProductCollection
    - +ProductDetails: ProductItemColl
    - +MinPrice: decimal
  - Type: ProductCategory**
    - +Name: string
    - +Description: string
  - Type: ProductItem**
    - +ID: int
    - +Size: int
    - +Color: Color
    - +Price: decimal
  - Core: Product**
    - +Name: string
    - +Description: string
- Address** (datatype)
- +Street: string
  - +Street2: string
  - +ZIP: int
  - +Town: string
  - +Region: string
  - +Country: string
  - +isBilling: boolean
  - +isShipping: boolean
- Relationships:**
- Customer** (1) to **Order** (\*): Orderer
  - Order** (1) to **OrderItem** (1..\*)
  - ProductCategory** (\*) to **ProductItem** (\*)
  - Product** (\*) to **ProductItem** (\*)
  - ProductItem** (1..\*) to **OrderItem** (1)
- Note:** BillingAddress and ShippingAddress needs to be one of the Orderers Adresses
- Business Type Model nach Cheesman und Daniels**

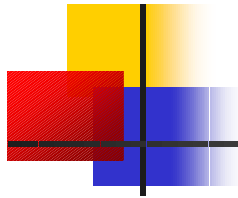
**August 12, 2002**



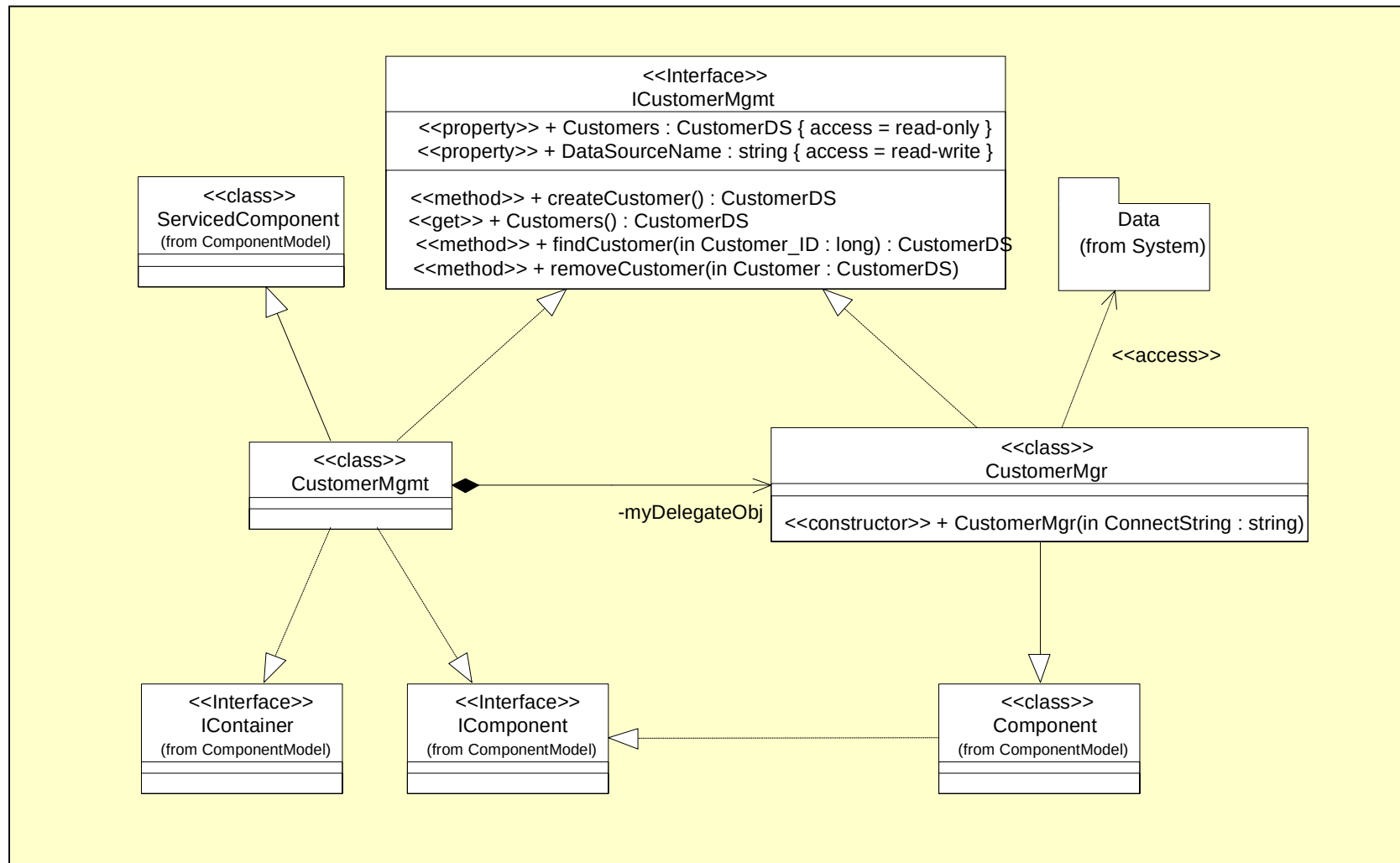
# Onlineshop - EDOC

- Komposition
- Schnittstellen
- Protokolle



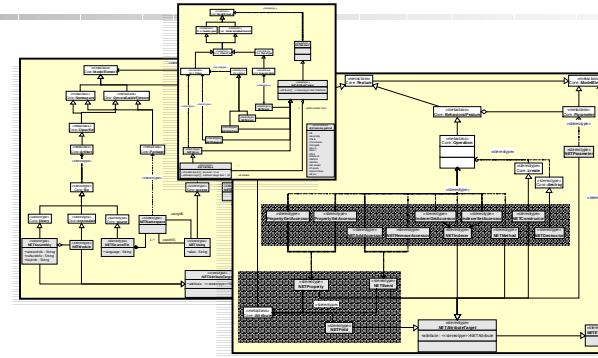


# Onlineshop - EDOC → .NET

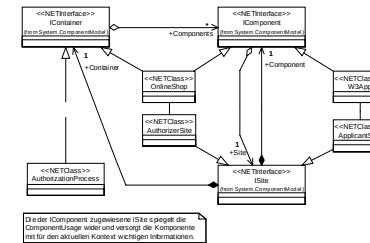
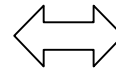
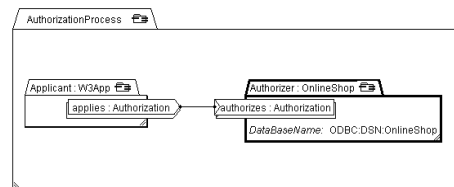


# Ergebnis

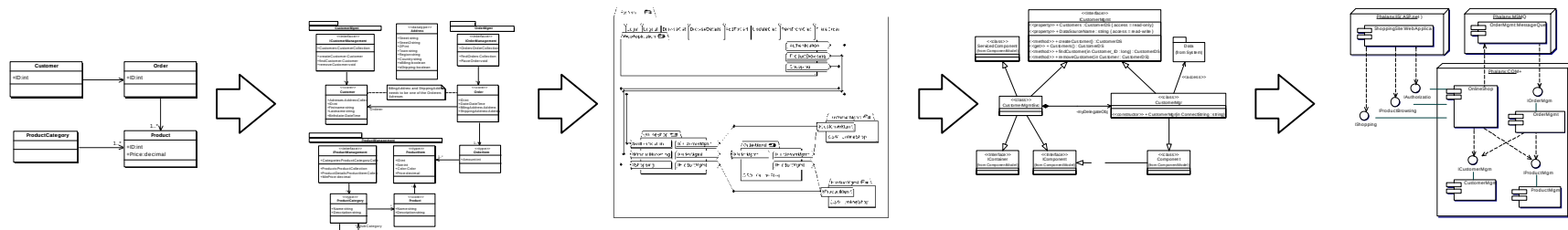
- Entwurf eines .NET-Profiles basierend auf C#

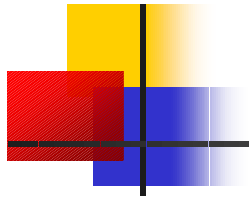


- Abbildung EDOC auf .NET-Profile



- Beispiel einer MDA-basierten Softwareentwicklung





# Ausblick

---

## Ausblick:

- *Akzeptanz bzw. Verbreitung von .NET*
- *Entwicklung der Bemühungen der OMG (MDA, EDOC, UML 2.0)*
- *Toolunterstützung bei der Anwendung von Profiles*

## Weitere Arbeiten:

- *geeignete Transformationsregeln PIM -> PSM*
- *Umsetzbarkeit von Stereotypen, Constraints usw. in .NET (Attribute)*
- *Refactoring zur MSIL*
- *.NET-Sourcecodeerzeugung ( MSIL, C#, Eiffel.NET ...)*