Building Structural UML Models in Visual Studio 2010

Richard Seroter

www.pluralsight.com

pluralsight

see what you can learn

Outline

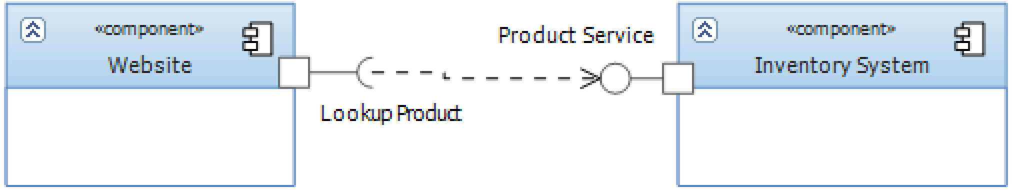
- Training Course Scenario Review
- Building a Component Diagram
- Building an Class Diagram
- Summary

Training Course Scenario Review

- **Watson's Pet Store**
  - They are a growing local store that provides pet supplies and services.
  - Their customers want to be able to order products and manage their service appoints online.

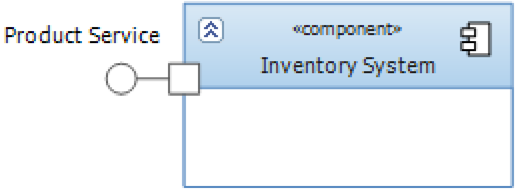
Building a Component Diagram

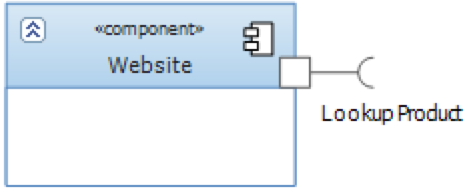
- **Component diagrams show physical structure of the system architecture.**
  - System Dependencies
  - Functional Decomposition
- **The core elements of a component diagram are:**
  - Components
  - Interfaces
  - Relationships



Building a Component Diagram - Interfaces

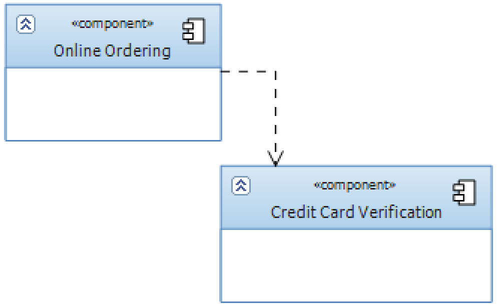
- Component interfaces are used to show how "block boxes" describe what they share with the outside world, and what they need from it.
- Provided Interfaces show what component capabilities can be used by other components.
- Required Interfaces describe the capabilities that a component sends to other components.
- There are generally accepted style guidelines.
  - Name interfaces after the protocol used, or …
  - Name the interface after the collection of capabilities (e.g. WSDL name), or …
  - Name the interface after the relevant operation

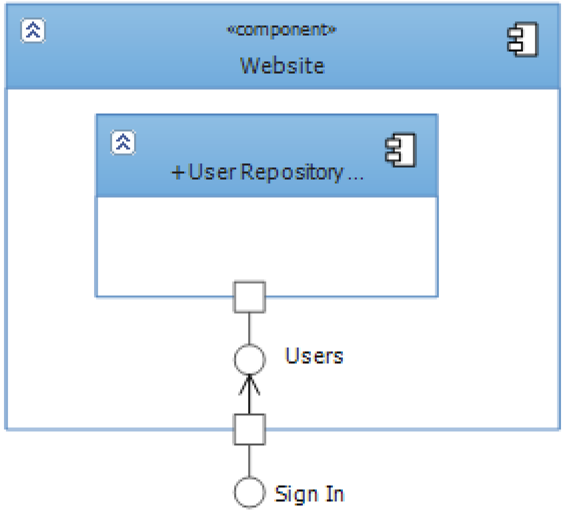




Building a Component Diagram - Relationships

- Dependency relationships demonstrate that one component (or interface of a component) relies on another.
- Generalization is used to indicate that one component inherits from another component.
- Delegation is applied when a component's port is linked to an internal component.
- Part Assembly is used within a component to connect the required and provided ports of sub-components.



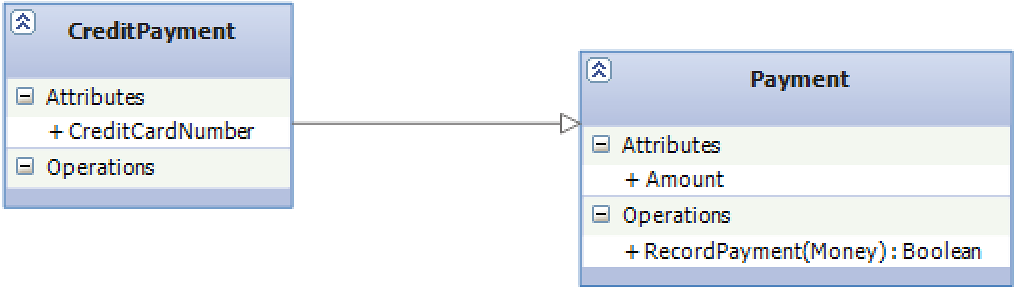


Building a Component Diagram DEMO

- **Add new (system dependency) diagram**
- **Drag components for website, inventory system, credential store, payment verification system**
  - Connect with dependencies; color the website different
- **Add new (functional decomposition) diagram**
- **Drag website, inventory system, Order Payment, and Reservation Service to diagram**
  - Add interface to inventory system (Product Svc) and required to website (Lookup Product)
  - Create other payment components (CC, Gift Cert) and do generalization
  - Add User Repo to Website; Add Users interface and delegate to Sign In on Website
  - Add Resv Repo and Resv Mgmt components to Resv Service
    - Add Reservations interface to Repo and connect to required Create Resv and Update Resv included on Resv Management component

Building a Class Diagram

- A class diagrams are used to model both the business and construction elements of a software solution.
- The core elements of a class diagram are:
  - Classes
  - Attributes
  - Interfaces
  - Enumerations
  - Packages
  - Relationships



CreditPayment

Attributes

+ CreditCardNumber

Operations

Payment

Attributes

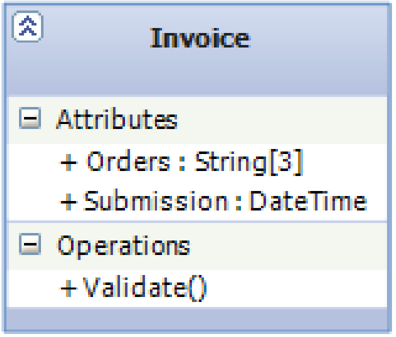
+ Amount

Operations

+ RecordPayment(Money) : Boolean

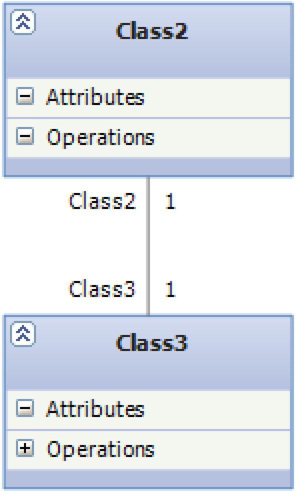
Building a Class Diagram - Classes

- Classes are made up of attributes and operations.
- Attributes and operations can have:
  - Data types
  - Visibility settings
  - Static designations
- Attributes may also have multiplicity and default values.
- Interfaces define a part of the externally visible behavior.
- Enumerations define a set of literal values.



Building a Class Diagram - Relationships

- Associations demonstrate a meaningful relationship between classes.
- Dependency shows that changes in one class may influence the implementation of the other class.
- Inheritance can take the form of “generalization” between types or “realization” between a class and interface.
- Aggregation describes a group of objects that form a whole.
- Composition is a stronger type of aggregation where the parts are physical part of the whole.
- Package Import describes a relationship between distinct packages.



Building a Class Diagram DEMO

- **Create new class diagram for a domain model**
  - Drag elements and do associations with multiplicity
    - Can type in own multiplicity
    - Add association labels
- **Create new class diagram for construction model**
  - Drag class, enumeration, and interface (operation) and set values
    - Ianimal with Type that points at Enum;
    - Do inheritance and switch to lollipop (show action tag)
  - Drag 3 classes and do generalization (show param wizard on operations)
    - Override operations on each
  - Drag “Branch” and “District” and connect left to right; remove navigation
    - Aggregation property (on role) to shared to composite
  - Drag Shopping cart item and shopping cart and connect
  - Packages and classes get scoped to a package; import between packages

Summary

- Training Course Scenario Review
- Building a Component Diagram
- Building an Class Diagram
- Summary