Computer Science
UNIVERSITY OF TORONTO

lecture 9:
robustness analysis

csc302h

winter 2014

Robustness Analysis

Good Object Oriented Design

Robustness Analysis
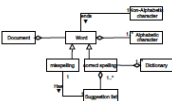
Allocating Behaviour

1

Starting Point

You've done the Requirements Analysis

You have:

A set of Use Cases
(explaining how users will use the system)

A Domain Model
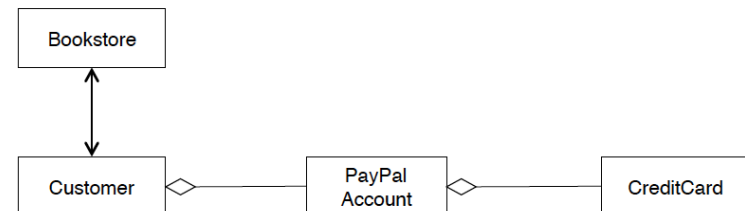**(to keep track of key domain concepts)**

Stakeholder Goal Models
(explaining how the use cases will meet the stakeholders' real needs)

Challenge:

Allocate responsibility for the use cases to classes in the system

2

Diversion: What's wrong with this?

Bookstore — Customer — PayPal Account — CreditCard

```
class Bookstore {
    …

    void settlebill (int total) {
        …
        Customer.PayPalAccount.CreditCard.subtract(total)
        …
    }
}
```

3

the Law of Demeter

Basically:
"Only talk to your friends"

More specifically:
A method, m, of an object, O, can only call methods of:
1. O itself
2. m's parameters
3. any object created by m
4. O's direct component objects
[m cannot call methods of an object returned by another method call]

Programmer's rule of thumb:
"use only one dot"
e.g. instead of: Customer.PayPalAccount.CreditCard.subtract(total)
use: Customer.GetPayment(total)

Robustness Analysis

Boundary Objects
Used by actors when communicating with the system
Only these can initiate events
(usually major User Interface elements, eg screens)

Entity Objects
Usually objects from the domain model
Things we need to keep track of

Control Objects
The "glue" between boundary objects & entity objects
Capture business rules and policies
(note: often implemented as methods of other objects)

Why do Robustness Analysis?

Bridges the gap between Requirements and Design

Sanity Check
Tests the language in the Use Case description
Nouns from the Use Case get mapped onto objects
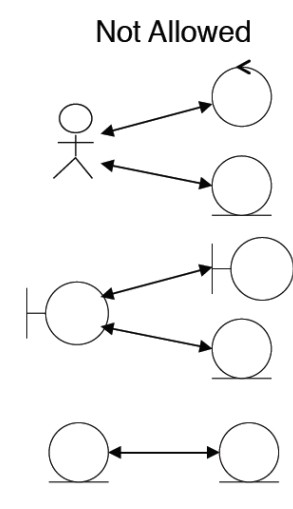Verbs from the Use Case get mapped onto actions

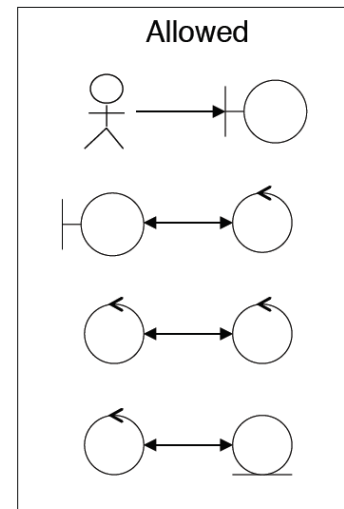**Completeness Check**
Discover the objects you need to implement the use cases
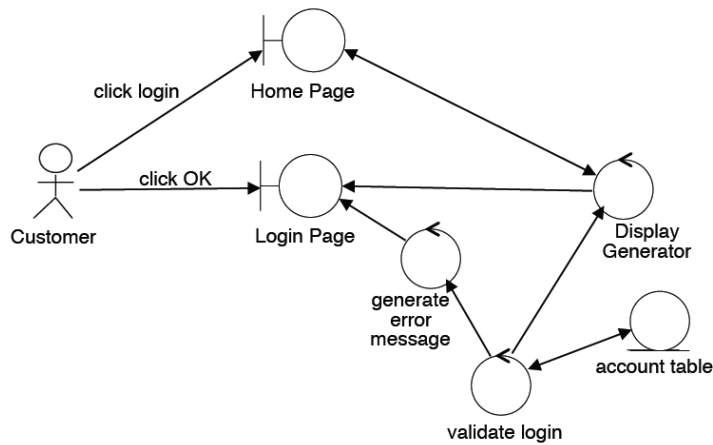Identify alternative courses of action

Object Identification
Decide which methods belong to which objects
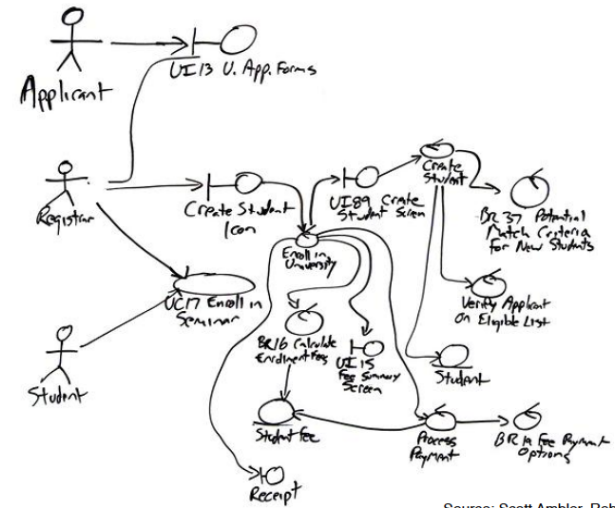

Rules for Robustness Diagrams

Allowed | Not Allowed

Slide 8

Example

click login — Home Page

Customer — click OK — Login Page

Display Generator

generate error message

account table

validate login

8

Slide 9

Intended for the whiteboard…

Source: Scott Ambler, Robustness Diagrams

9

Slide 10

Constructing a Robustness Diagram

Add a boundary element for each major UI element
  (not at the level of individual widgets though!)

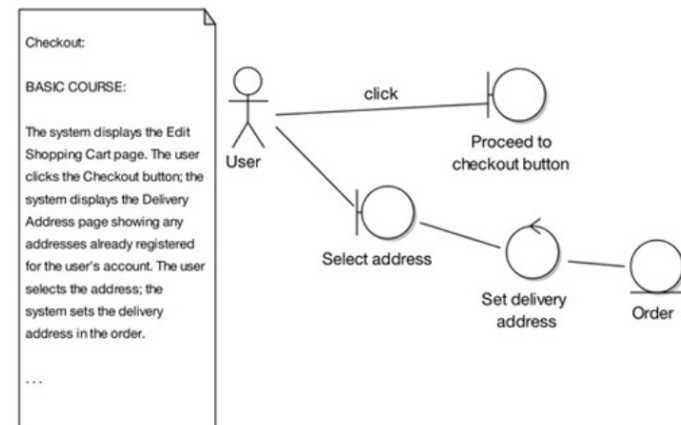**Add controllers:**
  One to manage each Use Case
  One for each business rule
  Another for each activity that involves coordination of several other elements

Add an entity for each business concept
  (most domain objects!)

10

Slide 11

Compare against each Use Case

Checkout:

BASIC COURSE:

The system displays the Edit Shopping Cart page. The user clicks the Checkout button; the system displays the Delivery Address page showing any addresses already registered for the user's account. The user selects the address; the system sets the delivery address in the order.

…

User — click — Proceed to checkout button

Select address

Set delivery address — Order

Source: Matt Stephens, Robustness Analysis in Practice

11

Slide 12

Improved Diagram…

Checkout:

BASIC COURSE:

The system displays the Edit Shopping Cart page. The user clicks the Checkout button; the system retrieves the Delivery Addresses already registered for this user, and then displays the Delivery Address page. The user selects the address to use from the Customer Address list selector, and clicks the Select button. The system sets the Delivery Address in the Order.

…

Source: Matt Stephens, Robustness Analysis in Practice

Slide 13

ICONIX process

Slide 14

Basic Design Steps

Use case text is refined during robustness analysis and reviewed during the preliminary design review.

1. Copy the use case text to the left margin of the sequence diagram.
2. Add the entity objects.
3. Add the boundary objects.
4. Work through the controllers, one at a time, and figure out how to allocate the behavior among the collaborating objects.

Slide 15
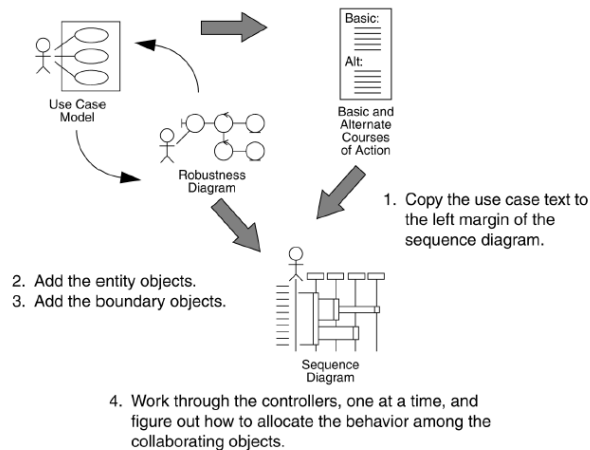
Benefits of Robustness Analysis

1. Forces a consistent style for use cases
2. Forces correct 'voice' for use cases
3. Sanity and completeness check for use cases
4. Syntax rules for use case descriptions
   e.g. actors only talk to boundary objects
5. Quicker and easier to read than sequence diagrams
6. Encourages use of Model-View-Controller (MVC) pattern
7. Helps build layered architectures
   e.g presentation layer, domain layer, repository layer
8. Checks for reusability across use cases before doing detailed design
9. Provides traceability between user's view and design view
10. Plugs semantic gap between requirements and design

next time: v&v