KEY:
ASSIGN RESPONSIBILITY TO CLASSES
RESPONSIBILITIES (CRC)

Write (on a card) the name of a class, the responsibilities of that class, and the collaborators, other classes that will help carry out each responsibility.

Responsibilities – high level purpose of class, may be realised by many methods

Guideline: each class should not have more than three or four responsibilities – poor cohesion

Guideline: generally class should have only a few collaborators – poor coupling
## CRC CARDS

<table>
<thead>
<tr>
<th>LibraryMember</th>
<th>Responsibilities</th>
<th>Collaborators</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maintain data about copies currently borrowed. Meet requests to borrow and return copies.</td>
<td>Copy</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Copy</th>
<th>Responsibilities</th>
<th>Collaborators</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maintain the data about a particular copy of a book. Inform corresponding Book when borrowed and returned</td>
<td>Book</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Book</th>
<th>Responsibilities</th>
<th>Collaborators</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maintain data about one book. Know whether there are borrowable copies.</td>
<td></td>
</tr>
</tbody>
</table>
METHODS

For each responsibility, determine methods required to realise it.

Consider also use cases – what operations are required to support them.

Refine when doing dynamic/behavioural modelling.
UML

**LibraryMember**
- memberId : Integer

**Copy**
- is a copy of

**Book**
- title : String
- copiesOnShelf() : Integer
- borrow(c: Copy)
- return(c: Copy)

**Journal**
- title : String
- series : Integer
- volume : Integer
- borrow()
- return()

**MemberOfStaff**
- borrow/returns

**MemberOfStaff**
- borrow/returns
GENERALISATION

Identify classes that are generalisations of others or perhaps classes that have a common generalisation.

Leads to opportunities for reuse.

Test: Every A is a B.

A dog is an animal → every dog is an animal ✔

A collie is a breed → every collie is a breed ✗
GENERALISATION

- **LibraryMember**:
  - memberid : Integer

- **Copy**

- **Book**
  - copiesOnShelf() : Integer

- **Item**
  - title : String
  - borrow(c: Copy)
  - return(c: Copy)

- **Journal**
  - series : Integer
  - volume : Integer

- **MemberOfStaff**

borrow/returns

0..1 0..1

0..1 0..* 0..* 0..*

is a copy of

1

1..*
AGGREGATION

Conceptual notion of whole-part: one object is a part of another.

COMPOSITION

Refinement of aggregation: the whole strongly owns the part.

If whole is deleted, then so are all the parts.
MORE “CLASSES”

Interfaces

Abstract classes

Parameterized classes

Association classes – record information about association

As in Java

Diagram:

- **Student** (1..* is taking 6 **Module**)
- **GradeBook**
  - grade : Integer