Drawing the Line: 
Teaching the Semantics of Binary Class Associations

James H. Paterson, School of Engineering and Computing, Glasgow Caledonian University, james.paterson@gcal.ac.uk
John Haddow, School of Computing, University of the West of Scotland, john.haddow@uws.ac.uk
Ka Fai Cheng, School of Engineering and Computing, Glasgow Caledonian University, k.cheng@gcal.ac.uk

Introduction
Binary class associations and message passing are key concepts in the design and development of object-oriented software, but appear to be poorly understood by students[1]. Students often neither appreciate the exact meaning of class associations which they have identified nor understand how messages should be passed between instances. When it comes to implementing the classes, programming languages such as Java do not have syntax to express associations directly. The association must be expressed in code using the available tools: classes, attributes and methods[2]. This requires thought about exactly what is implied about the classes, and understanding of how to map that meaning to code.

This poster presents the use of a customizable code pattern tool, patternCoder[3], developed by the authors, to provide support for teaching of binary class associations based on research on the semantics of associations.

Implementing an association
This example shows patternCoder working within BlueJ in assisting in the implementation of a dynamic association. Here, the student has identified that two classes called Customer and Shop are required and that they are associated in some way.

Classifying associations
Stevens[2] proposes classifying associations with stereotypes <<static>> which may be implemented with a class attribute, or <<dynamic>>, a temporary association where objects may exchange a message without an attribute being defined, for example:

```
class P {
  private Q myQ;
  public void foo() {
    myQ.bar();
  }
}
```

```
class P {
  public void foo(Q myQ) {
    myQ.bar();
  }
}
```

Associated object is an attribute

Associated object is a parameter

References