CS4125
SYSTEMS ANALYSIS
SPRING SEMESTER 2010-2011

J.J. Collins
Dept of CSIS
University of Limerick

Lecture S – W12L2

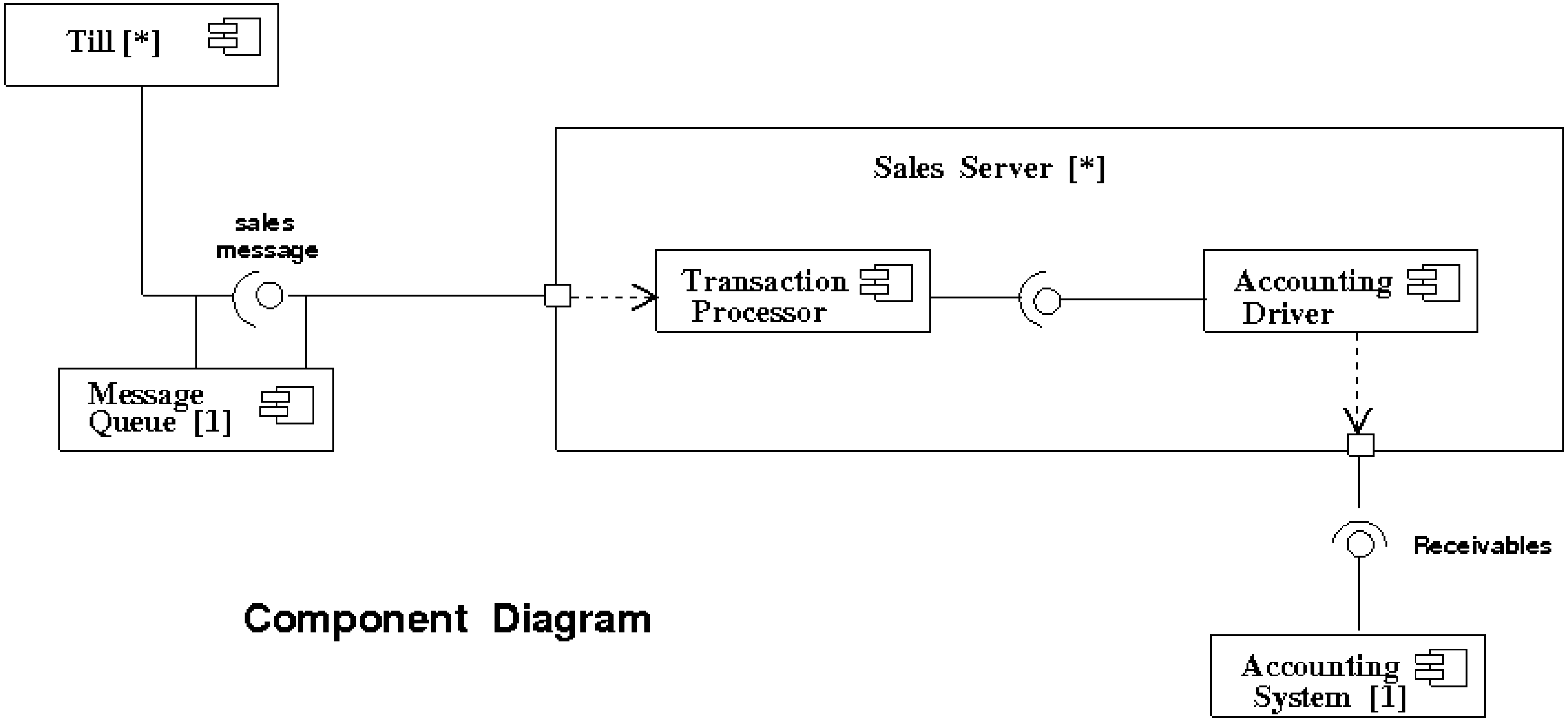
(a) Component and Deployment Diagrams (b) Recap (c) Final Exam (d) Midterm

1. UML: Component and Deployment Diagrams

- Most UML based CASE tools can generate skeleton code from the design.
- The implemented system has to run on hardware, while meeting its non-functional requirements such as performance, reliability and safety.
- UML defines two models to describe how the system is implemented:
  - The component model: shows dependencies between parts of the code. Forms part of the development view (code view). Of interest to designers and maintainers.
  - The deployment model: shows structure of runtime system, hardware configuration etc. Forms the physical view and the process view.

1. UML: Component and Deployment Diagrams

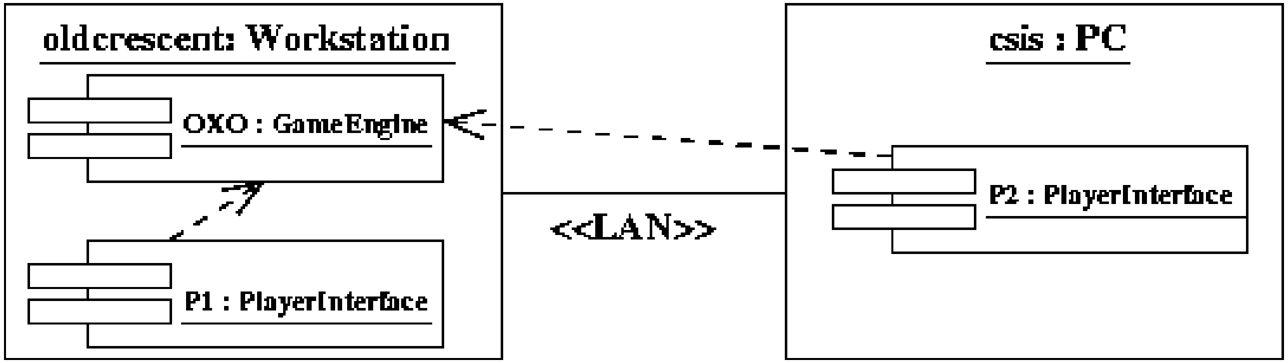
- A component may (and should) realise one or more interfaces.
- Components should depend on interfaces of other components, & not on their internals.



Component Diagram

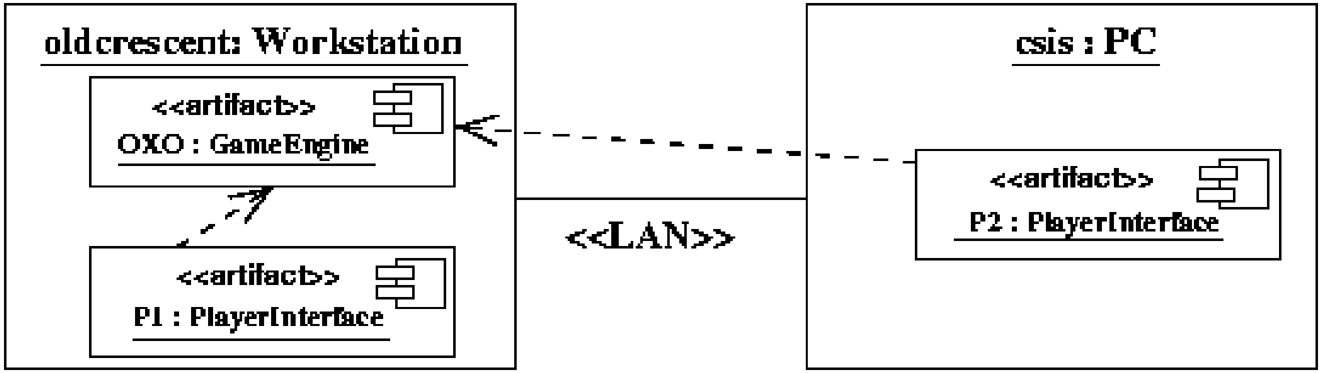
1. UML: Component and Deployment Diagrams

UML 1.x



Deployment Diagram

UML 2.0



Deployment Diagram

1. UML: Component and Deployment Diagrams

- Bennett et al. (Fourth Edition): sections 3 and 4 in chapter 19.
- Stevens and Pooley: chapters 13 and 14.

2. Module Themes

1. Characteristics of good software
   - Modules with interfaces
   - Low coupling, high dependency
   - Abstraction, encapsulation, information hiding
2. The UML as a modelling notation to support a generic OOAD methodology
   - Characteristics of OO paradigm – objects and classes, generalisation, polymorphism, templates
   - UML diagrams
   - DDD versus RDD, etc.
   - Program to interfaces, not implementations.
3. Software architecture and quality attributes
4. Interfaces and contracts, pre and post conditions
5. Patterns: architectural and design
6. All leading to
   1. REUSE and
   2. DEPENDENCY MANAGEMENT (maintenance and evolution support).

3. Final Exam 2010 / 2011

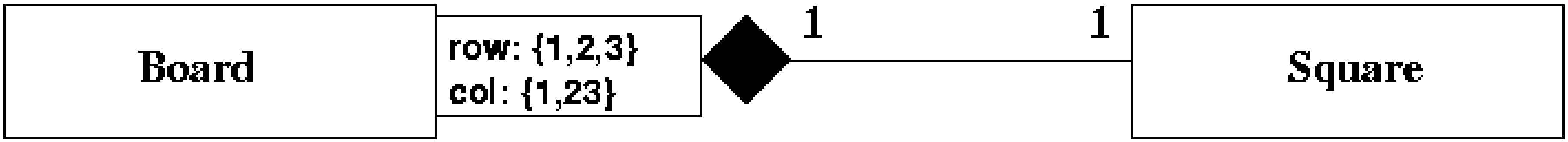
- Answer Q1 AND Any three others
- Q1 based on project and is worth 40 marks
- Other ***5*** questions worth 20 marks each
- Take cognizance of marks per part, varies per question.
- Exam is 2.5 hours
- Exam contributes 50% towards module grade
- Revision: last years exam paper, midterm paper.

4. Projects Interviews

- Submit projects to Nuala Kitson – CSIS Administrator, in CS1-005.
- Will take place on Tue 26th, Wed 27th, and Thur 28th April.
- Schedule will be emailed on Monday 18th April.
- You will be required to submit an electronic version of report. Bring it with you on a USB key.
- Will email instructions on using the Report Generator later this evening.

5. Midterm 2010-2011

- Q1: decomposition into modules, managed dependencies, maximise cohesion, interfaces at the correct level of abstraction, encapsulation and information hiding, all leading to architecture centric pluggable components.
- Q2



Midterm 2010-2011

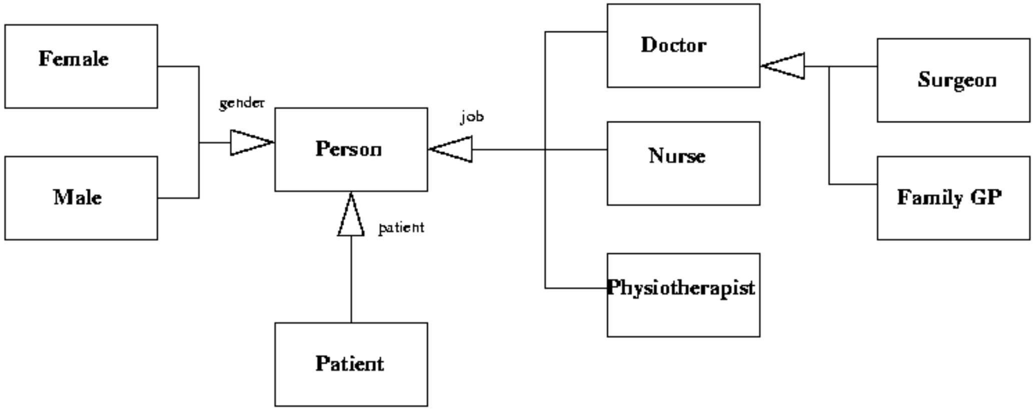
- Q3: polymorphism
  - Define - one interface multiple implementations.
  - Benefits
    - List whose type is a superclass, and contains objects instantiated from subclasses. Dynamic binding determines which implementation to execute at runtime
    - Supports extensibility – see example from chapter1 of Fowler's text on refactoring.
  - Code – must have a collection.
    - Note: sample code handed out demonstrates two mechanisms – known as supertypes, to support polymorphism: inheritance and interfaces.

Midterm 2010-2011

- Q3: problems with use cases
  - Procedural decomposition – danger of developing non OO code
  - Danger of confusing requirements with design
  - Missing requirements – RE + conceptual class modelling (analysis) carried out in parallel
- Q5: abstract class
  - Define – a class that has at least one method with no implementation defined.
  - Purpose:
    - Specify an interface for the generalisation (inheritance hierarchy)
    - Site for default implementations.
  - Boolean property abstract set to true, usually denoted by {abstract} in the class name compartment.

Midterm 2010-2011

- Q6: alt, loop, ref, par, region, sd, etc.
- Q7: see coding fragment at end of handout
- Q8: most diagrams did not have discriminator

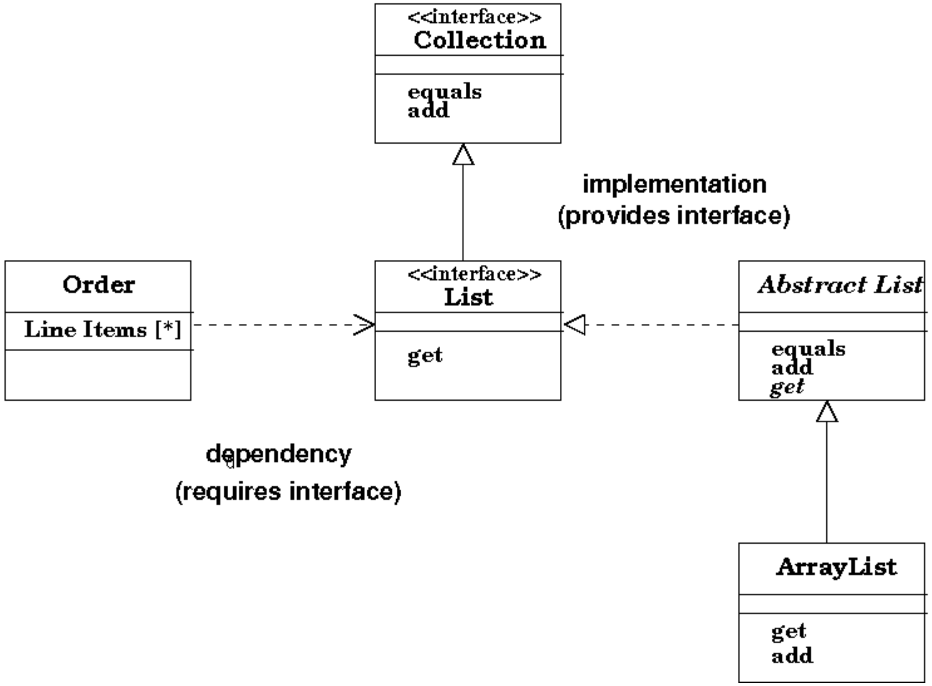


Multiple Classification

- Q9: see coding samples at end of this handout.
  - Common mistake – confusing composition with aggregation!

Midterm 2010-2011

- Q10: program to interfaces, not implementation.



- Benefits: decouples client from server, minimises impact on client when changing/upgrading server.