

CS249A Midterm Grading Rubric

Fall 2009

Problem 1

Point distribution

- Attributes (3)
- Remove verbs (3)
- No built-in types (3)
- RudderAngle composite attribute (2)
- Change display() into either functor with double-dispatch or notifications (4)

Common mistakes

- Using mass and weight without justification (-1)
- Returning non-const references to data members (-1)
- Not using pointers for entity reference (-1)
- Unclear explanation of functor and double-dispatch (-2)
- Turning display into an attribute (-3)

Problem 2

Point distribution

- (i) Selective notifications incur extra space/time cost, e.g in keeping a notified list for each notifying attribute and maintaining it (4)
- (ii) Unnecessary complexity and unclear how many parameters need to be passed, so reactor should implement specific requirements like storing old values (4)
- (iii) Invalid argument because with an attribute-only interface, the only changes are state changes (4)
- Using a one-size-fits-all notification mechanism helps separate development across interfaces. More specifically, designers of notifiers need not know what notifieds might be, but are ensured that the notification framework is uniform and supported by all classes. (3)

Common mistakes

- (i) Just saying “don’t override some on() methods” (-2)
- (ii) Mentioning general caching, but not performing caching in reactor specifically (-1)
- (iii) Implying that the client can perform actions on objects which then results in state changes (-1)
- Good reasons, but not the best, for one-size-fit-all notifications (-1)

Problem 3

Point distribution

- Definition of revised class with clear description of graphics primitives (Points, Lines, Rectangles, etc.), notification support and call-up interface (8)
- Any two advantages (4+3)
 - Portability
 - Efficiency, e.g. H/W optimization, less virtual method calls
 - Simplicity, only need a single machine-independent interface for object model

Common mistakes

- Description not specific enough, e.g. a vague description with no code (-3)
- Not mentioning notifications (-2)
- Mentioning notifications but not explaining how they work (-1)
- Improper graphics primitives. In particular, it is not valid to assume the HAL understands high-level concepts like GameCanvas and CharacterPosition (-2)
- Only mention one advantage (-3)

Problem 4

Point distribution

- Part (a) (7.5)
 - Problems with Paten's design – any one will do (4)
 - Disregards referent system
 - Cannot model complex behavior
 - Other reasonable answers
 - Entity examples (2 + 1.5)
 - Airplane
 - ControlTower
 - Runway
 - AirTrafficController
 - Other reasonable answers
- Part (b) (7.5)
 - Problems with Saten's design – significant performance overhead and complex system design. (4)
 - Value examples (2 + 1.5)
 - Controller alertness as an abstraction of text messages
 - Probability of failure as an abstraction of rivets on wings
 - Other reasonable answers

Common mistakes

- Did not mention performance issue in part (b) (-2)

Problem 5

Point distribution

- Part (a): Any one of the following (4)
 - Defining equality
 - Shared description
 - Do not actually refer to referent system objects
- Part (b): Any one of the following (4)
 - Referenced by smart pointers
 - Named descriptions have identity
 - Shared description
- Examples (3.5 for each part)

Common mistakes

- Bad example explanation (-2)
- For part (b): Saying “Named description can have multiple attributes while value types cannot” (-4)