Appendix: EXPRESS Model (1 page)

One and a half hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Model-Based Software Design

Tuesday 22nd January 2008
Time: 14:00 – 15:30

Please answer any TWO questions from the FOUR questions provided
This is a CLOSED book examination

The use of electronic calculators is NOT permitted.
1. For parts a)-f), consider the EXPRESS model given in appendix A.

a) Explain the meaning of the OPTIONAL keyword on line 31. (1 mark)

b) Explain what INVERSE attributes, such as those on lines 20 and 21, are and their advantage over alternative ways of achieving the same effect. (2 marks)

c) Explain the QUERY statement on line 18. (1 mark)

d) Explain the constraint on line 5. (1 mark)

e) Explain the constraint on line 23. (1 mark)

f) Explain the effects on instance values of the UNIQUE clause on lines 32-33? (1 mark)

Write an EXPRESS model that captures a domain model for the following situation:

Mr. Green is a commercial vegetable and fruit grower and has 20 greenhouses and five outdoor vegetable beds. The fruit plants that he grows have a minimum and maximum temperature that they can tolerate, so he only grows these in the greenhouses and carefully monitors the temperature in each of the greenhouses. To maximise the range of fruit that he grows, each greenhouse has a different temperature. The vegetable plants that Mr Green grows are more tolerant of temperature changes, so he only grows these in the outdoor beds. The people that Mr Green supplies are very fussy and always want to know the Latin name of the plant from which the fruit or vegetable has come, and when it was planted.

The model that you define should capture any constraints (invariants) that are appropriate. (10 marks)

h) Discuss how appropriate the EXPRESS modelling notation is for capturing models during software development and if there are any restrictions on the sorts of models that it can capture. (3 marks)
2. a) In relation to ISO STEP, explain the role of the following:

i) EXPRESS

ii) Part21

iii) Standard Data Access Interface (SDAI); defined by Part 22

iv) SDAI language bindings

(4 marks)

b) By using a small example EXPRESS model, show how the elements of this model are related to instances in Part 21.

(3 marks)

c) Consider the following EXPRESS ENTITY:

ENTITY Building;
    address : STRING;
    size : REAL; -- square meters
    rooms : SET[1:?] OF Room;
END_ENTITY;

Assuming that an implementation in Java is to be produced for this, discuss how both an early and late bound implementation would appear, and how these would be used to create an instance and set the value of the address or size attributes.

(8 marks)

d) Consider the following EXPRESS;

ENTITY Person ABSTRACT SUPERTYPE OF (Staff ANDOR Student));
    forename : STRING;
    age : REAL;
END_ENTITY;
ENTITY Staff SUBTYPE OF (Person);
    staffId : STRING;
END_ENTITY;
ENTITY Student SUBTYPE OF (Person);
    studentNumber : STRING;
END_ENTITY;

Discuss the affects of the ANDOR in the SUPERTYPE clause on what instances can exist, how this is handled in Part 21 and how it effects the identification of instance types when creating instances in a programming environment.

(5 marks)
3. a) Describe the issues that are important when going from a model, without implementation details, to an implementation. (3 marks)

b) Outline the role of each of the following types of transformations:

i) model 2 text (m2t)  
ii) model 2 model (m2m)  
iii) text 2 model (t2m)  
(3 marks)

c) For an m2t transformation, describe some of the issues that a transformation tool must allow a user defining a transformation to describe. (3 marks)

d) As part of a development flow, a transformation that takes an object with attributes and transforms them into a bean style Java class (private attributes, and public get and set accessor methods) is required; outline an implementation of this transformation. (4 marks)

e) For the creation of instances that are consistent with a model, discuss the merits of using a t2m tool, like TCS, over an implementation using just a basic parser toolkit. Your answer should include consideration of reference resolution, semantic checking and meaningful error messages. (5 marks)

f) Describe the role of meta-models in any of the types of transformation mentioned in section b) (2 marks)

4. a) Define the following terms:

i) Abstract syntax  
ii) Concrete syntax  
iii) Domain Specific Language (DSL)  
iv) Platform Independent Model (PIM)  
v) Platform Specific Model (PSM)  
vi) Target platform  
(6 marks)

b) Outline a model-driven software development flow from a single specification model to an implementation in multiple technologies. (5 marks)

c) Describe the role of meta-models and meta-meta-models in model-driven software development and how their use affects what model-specific tooling is required to work with models. (5 marks)

d) Many models and transformations can be required in a model-driven software development cycle. The ones used in the early stages of the design cycle may be application specific, however, the ones used in the later stages tend to be technology specific. Discuss the merits of using a model-driven software development approach and how the cost of using it could be minimised. (4 marks)

END OF EXAMINATION

Appendix: EXPRESS Model (1 page)
Appendix A

1. SCHEMA Museum_Schema;

2.

3. TYPE CatalogNumber = INTEGER;

4. WHERE

5. SELF >= 0;

6. END_TYPE;

7.

8. TYPE TitleString = STRING;

9. END_TYPE;

10.

11. TYPE ItemCategories = ENUMERATION OF (Painting, Sculpture, Ceramic);

12. END_TYPE;

13.

14. ENTITY Museum;

15. displayedItems : SET[0:?] OF CatalogItem;

16. DERIVE

17. itemsOnLoan : SET[0:?] OF CatalogItem :=

18. QUERY(item <* ownedItems | EXISTS(item.loanedTo));

19. INVERSE

20. ownedItems : SET[0:?] OF CatalogItem FOR ownedBy;

21. borrowedItems : SET[0:?] OF CatalogItem FOR loanedTo;

22. WHERE

23. x : displayedItems <= ownedItems - itemsOnLoan + borrowedItems;

24. END_ENTITY;

25.

26. ENTITY CatalogItem;

27. itemNumber : CatalogNumber;

28. category : ItemCategories;

29. title : TitleString;

30. ownedBy : Museum;

31. loanedTo : OPTIONAL Museum;

32. UNIQUE

33. uniqueId : ownedBy, itemNumber;

34. END_ENTITY;

35.

36. END_SCHEMA;