Exame Época Recurso - 2017-02-07 - Duração 90 minutos
Nota Minima 8/20 - **Sem consulta**, v A

isep Instituto Superior de Engenharia do Porto

EDOM - Mestrado em Engenharia Informática

Nome: _____

Numero: _____

= Parte 1 =

Atenção:

1. Todas as justificações devem ser colocadas numa folha em branco identificada como "**Parte 1**".
2. Nas perguntas com justificação esta vale 50%.
3. Deve colocar o seu número, nome e versão do exame no topo da folha em branco.

Pergunta 1 (4v)

Assinale com V as afirmações verdadeiras e com F as falsas. **Cada resposta errada desconta meia certa**.

_____ Podemos ter referências para uma entidade de um "aggregate" mesmo que esta não seja "root".

_____ Os Factory podem ser usados para a reconstituição de uma "entity".

_____ A "Mapping Feature" dos modelos significa que estes devem ser um reflexo completo do sistema que representam podendo inclusive substituir esse mesmo sistema.

_____ Nas abordagens "software product lines" o termo engenharia de aplicação e engenharia de domínio são sinónimos.

_____ A abordagem MDSE é conhecida pela "famosa" equação: Models + Transformations = Objects.

_____ O EMFText é uma ferramenta orientada ao desenvolvimento de sintaxes gráficas para DSLs baseadas em metamodelos ecore.

_____ Em ATL, quando se configura uma "Run Configuration" é necessário especificar o metamodelo de input e o metamodelo de output.

_____ Em UML "multiplicity" refere-se ao intervalo possível para a "cardinality" permitida nas instâncias do elemento em questão.

_____ Pode-se usar transformações ATL para converter instâncias entre versões diferentes de um metamodelo.

_____ O mecanismo de Profile do UML permite alterar o metamodelo do UML através da incorporação de novos elementos.

_____ O ecore permite associações unárias, binárias e ternárias entre classes.

_____ As Entities não devem referenciar Value Objects.

_____ Numa transformação ATL, numa regra "declarativa", pode-se usar uma expressão para especificar em que condições os elementos do modelo origem são usados na regra.

_____ Num modelo de domínio os *Value Objects* devem ser considerados mutáveis.

_____ O elemento EReference do Ecore não têm limites inferior e superior de multiplicidade.

_____ Um elemento Ecore *EEnum* permite definir uma lista de valores e pode ser utilizada numa das extremidades de uma EReference.

_____ Não é possível gerar ficheiro ".ecore" com o Acceleo.

_____ Normalmente existe uma relação de 1 para muitos entre diagramas e modelos.

_____ SQL é um exemplo de general-purpose language.

_____ Pode-se considerar que o ATL é uma DSL.

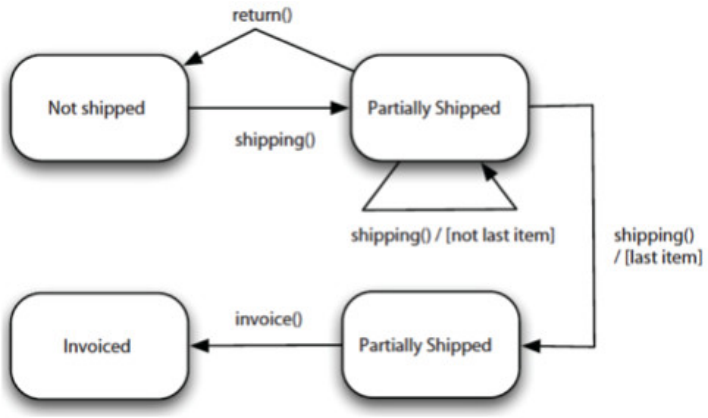
Pergunta 2 (2,5v)

Nas questões seguintes selecione a opção que permite obter uma afirmação correta. **Uma resposta errada desconta meia certa**.

Considere o seguinte *diagrama de estados* que representa o estado de despacho de uma encomenda.

Considere que o diagrama é "inspirado"na notação UML para diagramas similares mas não tem de ser completamente coerente com essa notação.

Considere ainda os seguintes conceitos relativos a este tipo de diagramas: estado "normal", estado inicial, estado final, transição, evento, pré-condição.



1. No diagrama apresentado...
   (a) "Partially Shipped" é um estado e para além disso um estado inicial.
   (b) "Invoiced" é um estado e para além disso um estado inicial.
   (c) "Not Shipped" é um estado e para além disso um estado inicial.
   (d) nenhuma das respostas anteriores.
2. No diagrama apresentado...
   (a) O texto dentro dos nodos corresponde a um evento.
   (b) O texto nas setas antes do "/" corresponde a um evento.
   (c) O texto nas setas depois do "/" corresponde a um evento.
   (d) nenhuma das respostas anteriores.
3. Relativamente a diagramas de estados podemos afirmar que...
   (a) Cada estado só pode ser "atingido" através de uma transição.
   (b) Cada estado só pode ter uma "transição" de saída para outro estado.
   (c) Não faz sentido existirem transições com o mesmo estado inicial e final.
   (d) nenhuma das respostas anteriores
4. A partir do diagrama apresentado podemos concluir que...
   (a) "Invoiced" é um estado final.
   (b) "Not Shipped" é um estado final.
   (c) "Partially Shipped"é um estado inicial.
   (d) nenhuma das respostas anteriores

Pergunta 3 (3,5v)

Apresente a resposta em folha anexa a entregar no final do exame.

1. Considere o diagrama de estados apresentado na questão anterior como exemplo de instância de um metamodelo ecore.

   Apresente um diagrama que represente um possível metamodelo ecore que suporte diagramas de estados como o anterior.

   Complemente esse metamodelo com as restrições OCL que considerar necessárias admitindo as seguintes situações:

   - Um diagrama tem de ter apenas 1 estado inicial e 1 estado final. Um diagrama deve ter pelo menos um estado para além do inicial e final.
   - Os eventos "activam" as transições. O mesmo evento pode aparecer em mais do que uma transição. As transições podem ter uma expressão de pré-condição que indica em que condição o evento "activa" a transição.
   - O estado inicial só pode ter transições para "fora" dele e o estado final transições para "dentro" dele.

= Parte 2 =

v A

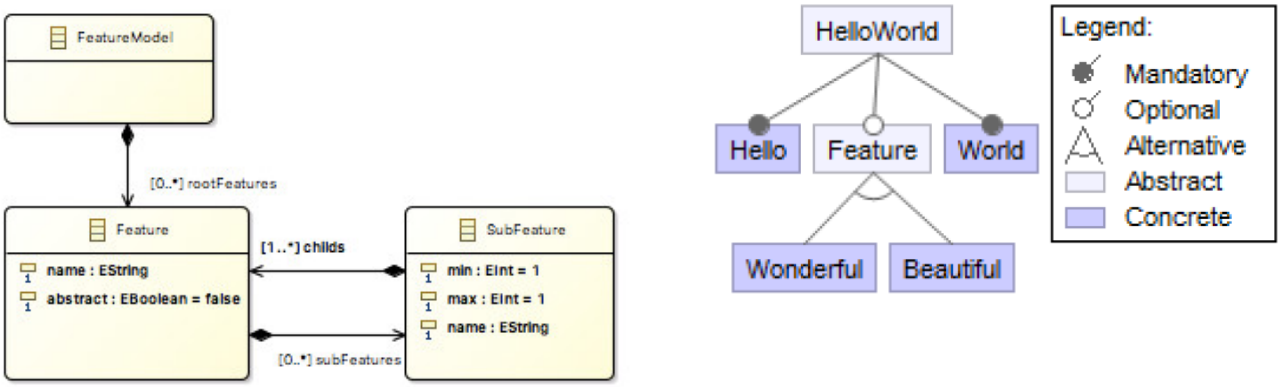
Nome: _____ Número: _____

Atenção:

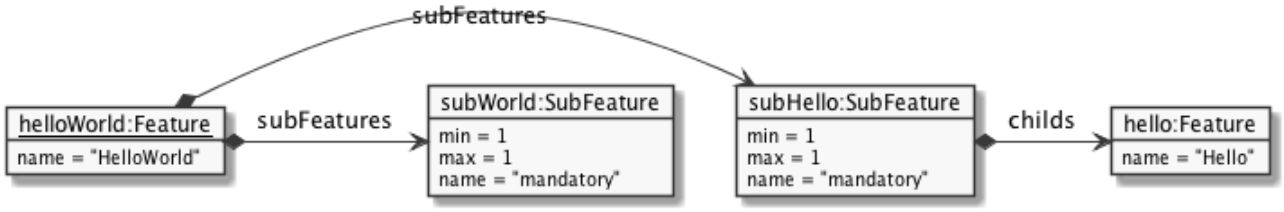
1. Todas as justificações devem ser colocadas numa folha em branco identificada como "**Parte 2**".
2. Nas perguntas com justificação esta vale 50%.
3. Deve colocar o seu número, nome e versão do exame no topo da folha em branco.

Pergunta 4 (3v)

Considere a representação de um metamodelo e uma instância:



1. Complete o seguinte diagrama de objectos do UML de forma a representar a instância apresentada.



Pergunta 5 (3v)

Considere o seguinte conteúdo de um ficheiro `myFeature.feature` ("suportado" por plugins EMFText) que representa parcialmente a instância apresentada na questão anterior

```
1 FeatureModel <rootFeatures:Feature<
2      name = "HelloWorld"
3      subFeatures:SubFeature<
4         min = 1
5         max = 1
6         name ="mandatory"
7         childs:Feature<
8            name ="Hello"
9         >
10     >
11   >
12 >
```

1. Apresente em folha anexa a continuação do código seguinte de um ficheiro `feature.cs` (EMFText) com duas regras adicionais `Feature` e `SubFeature` que permitam ter a representação anterior. Não é importante para a avaliação a representação de `LowerBound` e `UpperBound` dos vários elementos.

```
SYNTAXDEF feature
FOR <http://www.isep.pt/feature>
START FeatureModel

TOKENS {
   DEFINE COMMENT $'//'(~('\n'|'\r'|'\uffff'))*$;
   DEFINE INTEGER $('-')?('1'..'9')('0'..'9')*|'0'$;
   DEFINE FLOAT $('-')?(('1'..'9') ('0'..'9')* | '0') '.' ('0'..'9')+ $;
}

TOKENSTYLES {...}

RULES {
   FeatureModel ::= "FeatureModel" "<" ("rootFeatures" ":" rootFeatures)* ">"; }
```

Pergunta 6 (4v)

Considere o metamodelo e a instância apresentados na questão 4 e o metamodelo mindmap utilizado nas aulas da unidade curricular, cuja representação parcial em OCLinEcore se apresenta:

```
...
class _'Map'
{
   attribute title : String[?] = '';
   property elements : MapElement[*|1] { ordered composes };
   property rootTopics : Topic[*] { ordered derived transient volatile } { ... }
}

abstract class MapElement
{  attribute name : String[?]; }

class Topic extends MapElement
{
   attribute description : String[?];
   property parent#subtopics : Topic[?];
   property subtopics#parent : Topic[*|1] { ordered };
}
```

Considere o seguinte extrato de um ficheiro atl de especificação de transformação M2M com ATL:

```
module feature2mindmap;
create OUT : MM1 from IN : MM;

rule FeatureModel2Map {
   from
      source2: MM!FeatureModel
   to
      target2: MM1!"Map"(
         title <- 'Features'
   )
}
```

Responda às questões considerando que não há dependências entre as mesmas.

1. Altere a transformação de forma a obter um mindmap de tópicos obtidos a partir de elementos `Feature` com o encadeamento de tópicos e subtópicos mas apenas com os elementos referidos. Cada tópico deverá ter um nome.
   Elementos `SubFeature` não devem ser objeto de nenhuma transformação específica, apesar de poderem ser utilizados em alguma regra.

2. Altere a transformação de forma a obter um mindmap de tópicos obtidos a partir de elementos `SubFeature`. Devem ser considerados elementos `SubFeature` cujo atributo `name` tenha um valor único considerados os atributos `name` de todos os elementos `SubFeature`. No modelo gerado devem aparecer apenas esses elementos `Topic` agregados num elemento `Map` sem subtópicos. O atributo `title` do elemento `Map` deverá ter o valor "SubFeatures". Cada tópico deverá ter um nome.
   Elementos `Feature` não devem ser objeto de nenhuma transformação mas podem ser utilizados em alguma regra.

Answer Key for Exam A

= Parte 1 =

Atenção:

1. Todas as justificações devem ser colocadas numa folha em branco identificada como "**Parte 1**".
2. Nas perguntas com justificação esta vale 50%.
3. Deve colocar o seu número, nome e versão do exame no topo da folha em branco.

Pergunta 1 (4v)

Assinale com V as afirmações verdadeiras e com F as falsas. **Cada resposta errada desconta meia certa**.

False Podemos ter referências para uma entidade de um "aggregate" mesmo que esta não seja "root".

True Os Factory podem ser usados para a reconstituição de uma "entity".

True A "Mapping Feature" dos modelos significa que estes devem ser um reflexo completo do sistema que representam podendo inclusive substituir esse mesmo sistema.

False Nas abordagens "software product lines" o termo engenharia de aplicação e engenharia de domínio são sinónimos.

False A abordagem MDSE é conhecida pela "famosa" equação: Models + Transformations = Objects.

False O EMFText é uma ferramenta orientada ao desenvolvimento de sintaxes gráficas para DSLs baseadas em metamodelos ecore.

True Em ATL, quando se configura uma "Run Configuration" é necessário especificar o metamodelo de input e o metamodelo de output.

True Em UML "multiplicity" refere-se ao intervalo possível para a "cardinality" permitida nas instâncias do elemento em questão.

True Pode-se usar transformações ATL para converter instâncias entre versões diferentes de um metamodelo.

False O mecanismo de Profile do UML permite alterar o metamodelo do UML através da incorporação de novos elementos.

False O ecore permite associações unárias, binárias e ternárias entre classes.

False As Entities não devem referenciar Value Objects.

True Numa transformação ATL, numa regra "declarativa", pode-se usar uma expressão para especificar em que condições os elementos do modelo origem são usados na regra.

False Num modelo de domínio os *Value Objects* devem ser considerados mutáveis.

False O elemento EReference do Ecore não têm limites inferior e superior de multiplicidade.

False Um elemento Ecore *EEnum* permite definir uma lista de valores e pode ser utilizada numa das extremidades de uma EReference.

False Não é possível gerar ficheiro ".ecore" com o Acceleo.

False Normalmente existe uma relação de 1 para muitos entre diagramas e modelos.

False SQL é um exemplo de general-purpose language.

True Pode-se considerar que o ATL é uma DSL.

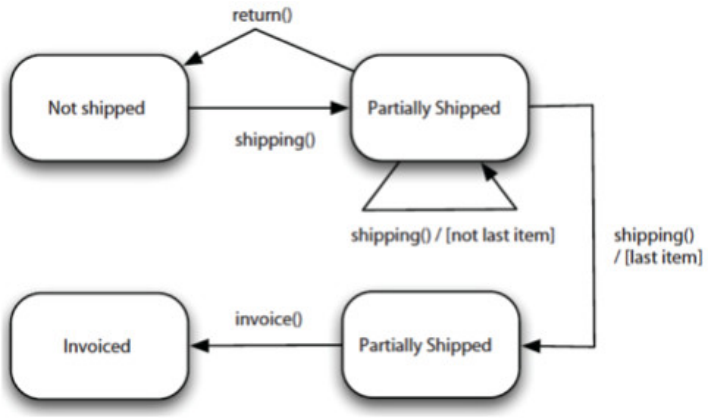
Pergunta 2 (2,5v)

Nas questões seguintes selecione a opção que permite obter uma afirmação correta. **Uma resposta errada desconta meia certa**.

Considere o seguinte *diagrama de estados* que representa o estado de despacho de uma encomenda.

Considere que o diagrama é "inspirado"na notação UML para diagramas similares mas não tem de ser completamente coerente com essa notação.

Considere ainda os seguintes conceitos relativos a este tipo de diagramas: estado "normal", estado inicial, estado final, transição, evento, pré-condição.



1. No diagrama apresentado...
   (a) "Partially Shipped" é um estado e para além disso um estado inicial.
   (b) "Invoiced" é um estado e para além disso um estado inicial.
   [(c)] "Not Shipped" é um estado e para além disso um estado inicial.
   (d) nenhuma das respostas anteriores.
2. No diagrama apresentado...
   (a) O texto dentro dos nodos corresponde a um evento.
   [(b)] O texto nas setas antes do "/" corresponde a um evento.
   (c) O texto nas setas depois do "/" corresponde a um evento.
   (d) nenhuma das respostas anteriores.
3. Relativamente a diagramas de estados podemos afirmar que...
   (a) Cada estado só pode ser "atingido" através de uma transição.
   (b) Cada estado só pode ter uma "transição" de saída para outro estado.
   (c) Não faz sentido existirem transições com o mesmo estado inicial e final.
   [(d)] nenhuma das respostas anteriores
4. A partir do diagrama apresentado podemos concluir que...
   [(a)] "Invoiced" é um estado final.
   (b) "Not Shipped" é um estado final.
   (c) "Partially Shipped"é um estado inicial.
   (d) nenhuma das respostas anteriores

Pergunta 3 (3,5v)

Apresente a resposta em folha anexa a entregar no final do exame.

1. Considere o diagrama de estados apresentado na questão anterior como exemplo de instância de um metamodelo ecore.

   Apresente um diagrama que represente um possível metamodelo ecore que suporte diagramas de estados como o anterior.

   Complemente esse metamodelo com as restrições OCL que considerar necessárias admitindo as seguintes situações:

   - Um diagrama tem de ter apenas 1 estado inicial e 1 estado final. Um diagrama deve ter pelo menos um estado para além do inicial e final.
   - Os eventos "activam" as transições. O mesmo evento pode aparecer em mais do que uma transição. As transições podem ter uma expressão de pré-condição que indica em que condição o evento "activa" a transição.
   - O estado inicial só pode ter transições para "fora" dele e o estado final transições para "dentro" dele.

```
package state : state = 'http://isep.ipp.pt/state'
{
   abstract class State
   {
      attribute name : String[1];
   }
   class InitialState extends State;
   class FinalState extends State;
   class RegularState extends State;
   class StateDiagram
   {
      attribute name : String[1];
      property regularStates : RegularState[+|1] { ordered composes };
      property initialState : InitialState[1] { composes };
      property finalState : FinalState[1] { composes };
      property transitions : Transition[+|1] { ordered composes };
      property events : Event[+|1] { ordered composes };
   }
   class Event
   {
      attribute name : String[1];
   }
   class Transition
   {
      attribute preCondition : String[?];
      property event : Event[1];
      property incomming : State[1];
      property outgoing : State[1];
      invariant notIncommingFromFinalState : not self.incomming->oclIsTypeOf(Set(FinalState));
      invariant notOutgoingToInitialState : not self.outgoing->oclIsTypeOf(Set(InitialState));
   }
}
```

= Parte 2 =

v A

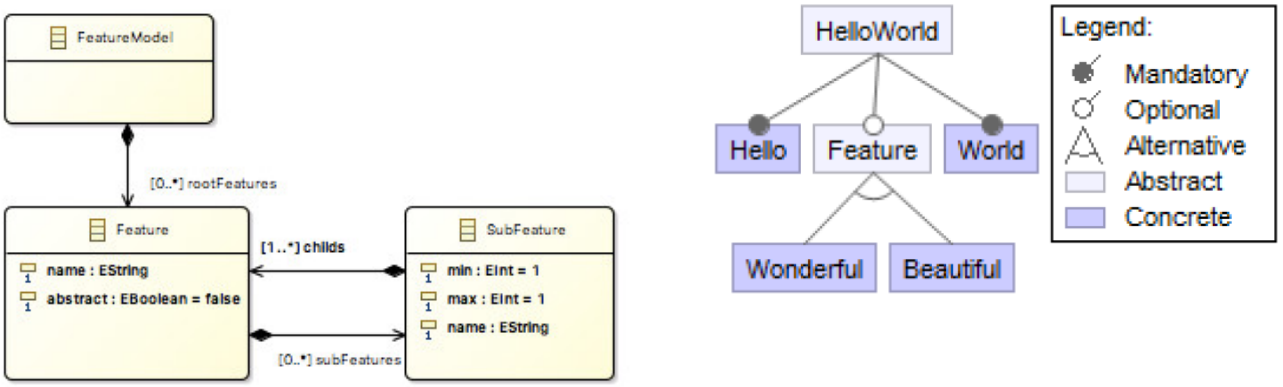
Nome: _____ Número: _____

Atenção:

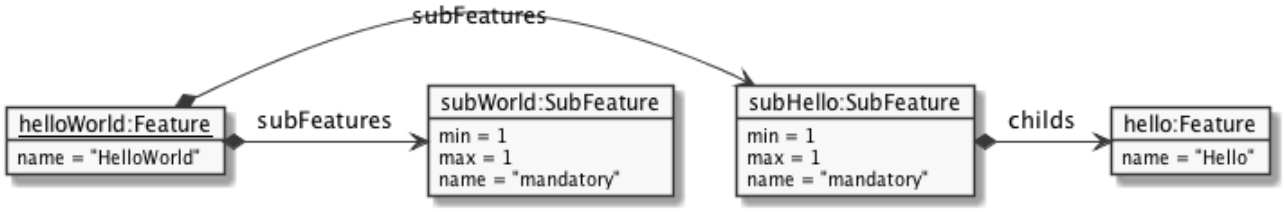
1. Todas as justificações devem ser colocadas numa folha em branco identificada como "**Parte 2**".
2. Nas perguntas com justificação esta vale 50%.
3. Deve colocar o seu número, nome e versão do exame no topo da folha em branco.

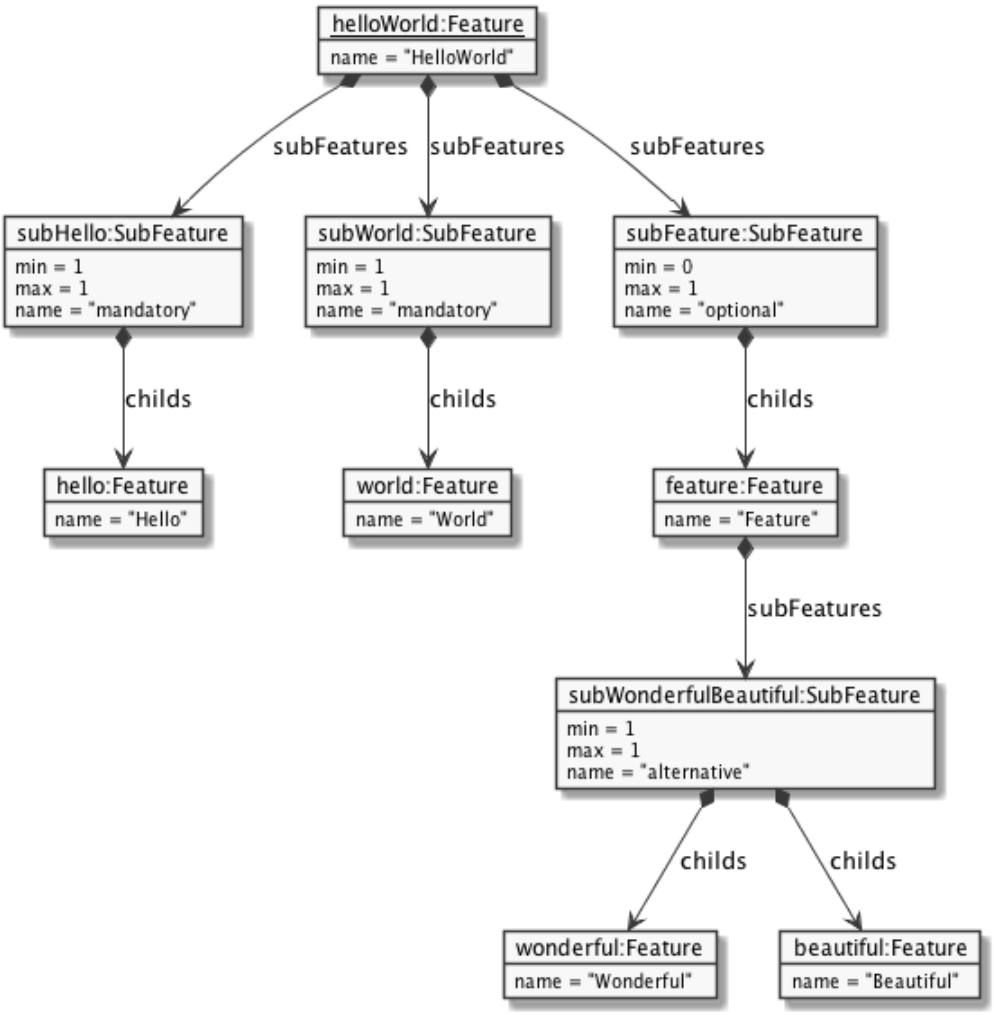
Pergunta 4 (3v)

Considere a representação de um metamodelo e uma instância:



1. Complete o seguinte diagrama de objectos do UML de forma a representar a instância apresentada.





Pergunta 5 (3v)

Considere o seguinte conteúdo de um ficheiro `myFeature.feature` ("suportado" por plugins EMFText) que representa parcialmente a instância apresentada na questão anterior

```
FeatureModel <rootFeatures:Feature<
     name = "HelloWorld"
     subFeatures:SubFeature<
        min = 1
        max = 1
        name ="mandatory"
        childs:Feature<
           name ="Hello"
        >
     >
  >
>
```

1. Apresente em folha anexa a continuação do código seguinte de um ficheiro `feature.cs` (EMFText) com duas regras adicionais `Feature` e `SubFeature` que permitam ter a representação anterior. Não é importante para a avaliação a representação de `LowerBound` e `UpperBound` dos vários elementos.

```
SYNTAXDEF feature
FOR <http://www.isep.pt/feature>
START FeatureModel

TOKENS {
   DEFINE COMMENT $'//'(~('\n'|'\r'|'\uffff'))*$;
   DEFINE INTEGER $('-')?('1'..'9')('0'..'9')*|'0'$;
   DEFINE FLOAT $('-')?(('1'..'9') ('0'..'9')* | '0') '.' ('0'..'9')+ $;
}

TOKENSTYLES {...}
```

```
12
13 RULES {
14    FeatureModel ::= "FeatureModel" "<" ("rootFeatures" ":" rootFeatures)* ">"; }
```

```
1 SYNTAXDEF feature
2 FOR <http://www.isep.pt/feature>
3 START FeatureModel
4
5 TOKENS {
6    DEFINE COMMENT $'//'(~('\n'|'\r'|'\ufffd'))*$;
7    DEFINE INTEGER $('-')?('1'..'9')('0'..'9')*|'0'$;
8    DEFINE FLOAT $('-')?(('1'..'9') ('0'..'9')* | '0') '.' ('0'..'9')+ $;
9 }
10
11 TOKENSTYLES {...}
12
13 RULES {
14    FeatureModel ::= "FeatureModel" "<" ("rootFeatures" ":" rootFeatures)* ">";
15    Feature ::= abstract["abstract" : ""] "Feature" "<" ("name" "=" name['"','"'] | "subFeatures" ":"
          subFeatures)* ">";
16    SubFeature ::= "SubFeature" "<" ("min" "=" min[INTEGER] | "max" "=" max[INTEGER] | "childs" ":" childs
          | "name" "=" name['"','"'])* ">";
17 }
```

Pergunta 6 (4v)

Considere o metamodelo e a instância apresentados na questão 4 e o metamodelo mindmap utilizado nas aulas da unidade curricular, cuja representação parcial em OCLinEcore se apresenta:

```
1 ...
2 class _'Map'
3 {
4    attribute title : String[?] = '';
5    property elements : MapElement[*|1] { ordered composes };
6    property rootTopics : Topic[*] { ordered derived transient volatile } { ... }
7 }
8
9 abstract class MapElement
10 {  attribute name : String[?]; }
11
12 class Topic extends MapElement
13 {
14    attribute description : String[?];
15    property parent#subtopics : Topic[?];
16    property subtopics#parent : Topic[*|1] { ordered };
17 }
```

Considere o seguinte extrato de um ficheiro atl de especificação de transformação M2M com ATL:

```
1 module feature2mindmap;
2 create OUT : MM1 from IN : MM;
3
4 rule FeatureModel2Map {
5    from
6       source2: MM!FeatureModel
7    to
8       target2: MM1!"Map"(
9          title <- 'Features'
10    )
11 }
```

Responda às questões considerando que não há dependências entre as mesmas.

1. Altere a transformação de forma a obter um mindmap de tópicos obtidos a partir de elementos `Feature` com o encadeamento de tópicos e subtópicos mas apenas com os elementos referidos. Cada tópico deverá ter um nome. Elementos `SubFeature` não devem ser objeto de nenhuma transformação específica, apesar de poderem ser utilizados em alguma regra.

```
-- @path MM=/feature2mindmap/metamodels/Feature.ecore
-- @path MM1=/feature2mindmap/metamodels/mindmap.ecore
module feature2mindmap2;
create OUT : MM1 from IN : MM;

rule FeatureModel2Map {
   from
      source2: MM!FeatureModel
   to
      target2: MM1!"Map"(
         title <- 'Features',
         elements<-MM!Feature.allInstances()
   )
}

rule Feature2Topic {
   from
      source: MM!Feature
   to
      target: MM1!Topic(
         name <- source.name,
         subtopics <- source.subFeatures->collect(e | e.childs)
   )
}
```

2. Altere a transformação de forma a obter um mindmap de tópicos obtidos a partir de elementos `SubFeature`. Devem ser considerados elementos `SubFeature` cujo atributo `name` tenha um valor único considerados os atributos `name` de todos os elementos `SubFeature`. No modelo gerado devem aparecer apenas esses elementos `Topic` agregados num elemento `Map` sem subtópicos. O atributo `title` do elemento `Map` deverá ter o valor "SubFeatures". Cada tópico deverá ter um nome.
   Elementos `Feature` não devem ser objeto de nenhuma transformação mas podem ser utilizados em alguma regra.

```
...

module feature2mindmap;
create OUT : MM1 from IN : MM;

helper def: allTopics: OrderedSet(MM1!Topic) =
OrderedSet{};

helper def: AllNames():Bag(String) =
MM!SubFeature.allInstances()->collect(x|x.name);

rule SubFeature2Topic {
   from
      source: MM!SubFeature (thisModule.AllNames()->select(x|x=source.name)->size()=1)
   to
      target: MM1!Topic(
         name <- source.name
   )
   do {
      if (not thisModule.allTopics->exists (x|x.name = target.name) )
         thisModule.allTopics <- thisModule.allTopics->append(target);
   }
}

rule FeatureModel2Map {
   from
      source2: MM!FeatureModel
   to
      target2: MM1!"Map"(
         title <- 'SubFeatures',
         elements <- thisModule.allTopics
   )
}
```