16.03.2012	**Advanced Software Engineering** **University of Vienna** **Forschungsgruppe Software Architecture**	
Kennzahl	*Matrikelnummer*	*FAMILIENNAME* *Vorname*

14	15	6	15	7	3	∑ 60
1	2	3	4	5	6	

Bitte lesen Sie diesen Abschnitt. Sie haben **60** Minuten Zeit die Fragen zu beantworten.

Hinweise:

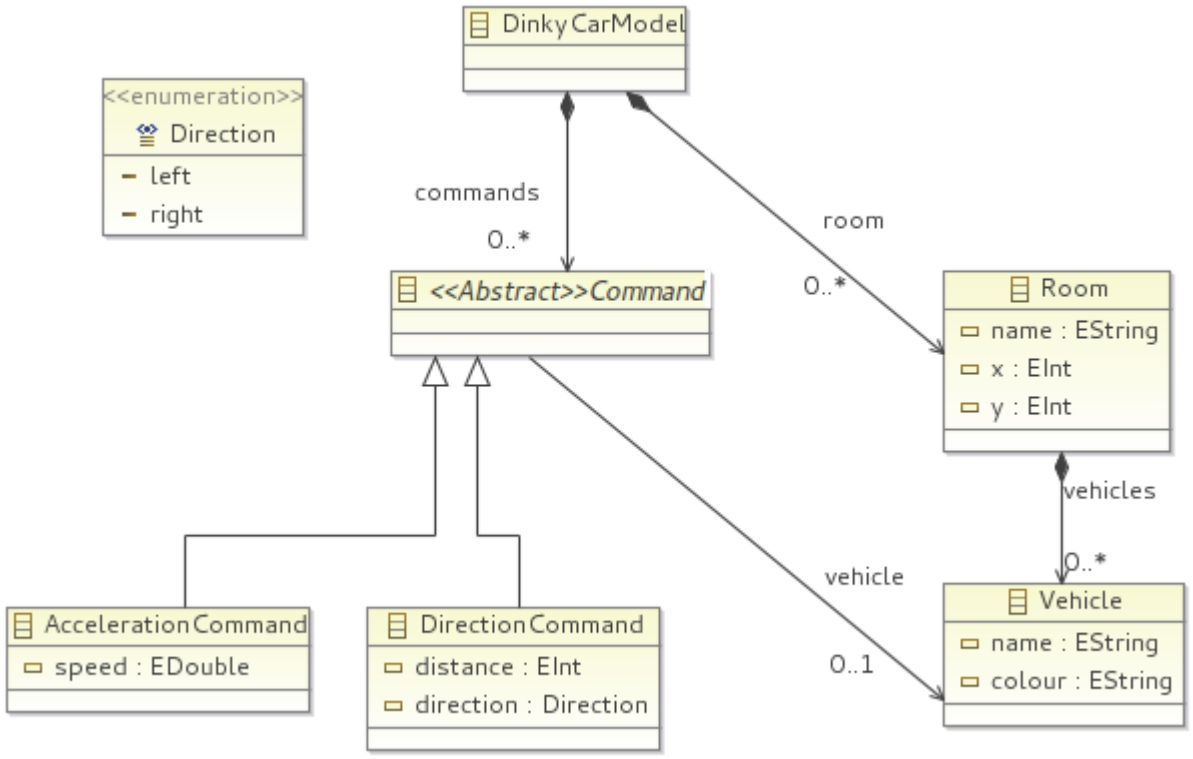
- Füllen Sie Kennzahl, Matrikelnummer, Familienname und Vorname zuerst aus.
- Während der Prüfung sind keinerlei Unterlagen erlaubt!
- Technische Hilfsmittel wie Übersetzungscomputer, Taschenrechner, Mobiltelefone, etc. sind nicht erlaubt!
- Wenn Sie Probleme beim Verstehen einer Frage haben, fragen Sie.
- Sie können die Fragen sowohl auf Deutsch als auch auf Englisch beantworten.
- Schreiben Sie mit dokumentenechten Schreibutensilien (kein Bleistift).
- Sie können den Appendix (letzte Seite) von der Prüfung abtrennen.

- Viel Erfolg!

Task 1 / Aufgabe 1 (14 Points / Punkte)

Two important concepts in domain-driven development are Entities and Value Objects. Identify all Entities and Value Objects in the following domain model and explain why you identified a model element as Entity or Value Object.
Mark each Entity with an “E” and each Value Object with a “V”

Zwei wichtige Konzepte im Domain-driven Development sind Entitäten und Value Objects. Identifizieren Sie im folgenden Domänenmodell alle Entitäten und Value Objects und begründen Sie Ihre Entscheidungen.
Markieren Sie jede Entity mit einem „E“ und jedes Value Object mit einem „V“



Task 2 / Aufgabe 2 (15 Points / Punkte)

Depict the metamodel of the grammar for a domain specific language that is described below.

Erstellen Sie ein Diagramm des Metamodells der folgenden Grammatik einer domänenspezifischen Sprache.

```
grammar at.ac.univie.swa.ase.xtext.example.CarDsl with org.eclipse.xtext.common.Terminals
generate carDsl "http://www.ac.at/univie/swa/ase/xtext/example/CarDsl"
import "http://www.eclipse.org/emf/2002/Ecore" as ecore

Model:
        availableVehicles+=Vehicle*
        rentals+=Rental*;
Vehicle:
        Car | Motorcycle | Truck;

Rental:
"Rental"
        "vehicle:" rentedVehicle=[Vehicle]
        rentedBy=Customer
        "pickup:" pickupDate=Date
        "dropoff:" dropoffDate=Date;

Customer:
        "customer:"
                "name:" name=STRING
                "driverslicense:" driverslicense=STRING
                "paymenttype:" paymenttype=PaymentType;

Car returns Vehicle:
{Car} "Car"
        name=ID
        "manufacturer:" manufacturer=STRING
        "type:" type=STRING
        "licenseplate:" plate=STRING
        "max persons:" persons=INT;

Motorcycle returns Vehicle:
{MotorCycle} "MotorCycle"
        name=ID
        "manufacturer:" manufacturer=STRING
        "type:" type=STRING
        "licenseplate:" plate=STRING;

Truck returns Vehicle:
{Truck} "Truck"
        name=ID
        "manufacturer:" manufacturer=STRING
        "type:" type=STRING
        "licenseplate:" plate=STRING
        "max payload:" payload=INT
        "requiredlicensetype" requiredLicense=LicenseType;
Date:   (day=Day"."month=Month"."year=Year);
enum PaymentType: Cash = 'cash' | CreditCard = 'credit card';
enum LicenseType: TypeB = 'B' | TypeC = 'C' | TypeD='D' | TypeE="E" | TypeF="F";
terminal Month returns ecore::EInt: ('0'..'1')?('0'..'9');
terminal Day returns  ecore::EInt: ('0'..'3')?('0'..'9');
terminal Year returns ecore::EInt: (('0'..'9')('0'..'9'))?('0'..'9')('0'..'9');
```

Task 3 / Aufgabe 3 (6 Points / Punkte)

What are advantages and disadvantages of Static Typing? Name and explain at least 3 advantages or disadvantages.

Nennen und beschreiben Sie mindestens 3 Vorteile / Nachteile von statischer Typisierung.

Task 4 / Aufgabe4 (15 Points / Punkte)

Fill the gaps 2-6 in the code on thenext pageby inserting the correct letters (as you can see for gap number1) that are given to you in the table below. Be careful, some lines of code may not be used.The expected output of the script is printed below.
For your convenience, you will find the code fragments at the Appendix as well, which can be separated from this exam.

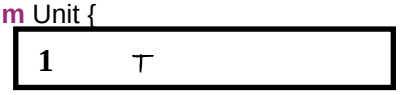
Füllen Sie die leeren Felder 2-6 im Code auf der nächsten Seitemit den richtigen Buchstaben (wie im Feld Nr. 1 zu sehen ist), welche in der Tabelle unten zu finden sind. Beachten Sie: einige Codezeilen werden nicht genutzt. Die erwartete Ausgabe des Scriptes finden Sie unten.
Zur leichteren Handhabung finden Sie die Code-Fragment im Anhang, den Sie von dieser Prüfung abtrennen können.

Expected output / ErwarteteAusgabe:
Car moves 252 METERS to LEFT
Truck stops Car!

Code fragments

A	`Number.metaClass.getMeters`
B	`NumberCategory`
C	`DistanceFactory.create`
D	`Def distanceClosure`
E	`new Car()`
F	`"Car"`
G	`Person: "Car"`
H	`[name: "Car"]`
I	`new Vehicle(name: "Truck")`
J	`new Vehicle(*truck)`
K	`[new Vehicle("Truck")]`
L	`"Truck".new`
M	`// a newscope`
N	`use(NumberCategory)`
O	`NumberCategory.each`
P	`truck.doesSomethingTo car`
Q	`truck."$doesSomethingTo" car`
R	`truck.doesSomethingTo { car }`
S	`"$doesSomethingTo"(truck, car)`
T	`MILLIMETERS, CENTIMETERS, METERS, DECIMETERS, KILOMETERS`
U	`GRAM, KILOGRAM, TON`
V	`(left)`
W	`.“down“`
X	`.left`
Y	`UP`

```
final class Distance {
   def Number number
        def Unit unit
        def Direction direction

   String toString() { "${number} ${unit}" }

        def propertyMissing(String name, value) {
                direction = Direction.valueOf(name.toUpperCase());
                if(direction == null)
                        print "Given direction ${name} cannot be interpreted!"
                return this;
        }
}

enum Direction {
        LEFT,RIGHT
}

enum Unit {
```

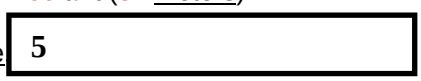



```
}

left = Direction.LEFT
right = Direction.RIGHT
```


```
{ -> new Distance(number: delegate, unit: Unit.METERS)  }

class NumberCategory {
        static Distance and(Number lhs, Distance rhs) {
                new Distance(number: lhs+rhs.number, unit: rhs.unit)
        }
}

class Vehicle {
        def name
        def move(distance) {
                println "${this} moves ${distance} to ${distance.direction}"
        }
        def stops(someone) {
                println "$this stops ${someone}!"
        }
        String toString() { "${name}" }
}

def doesSomething = "stops"
Vehicle car, truck
```


```
{

        car =
```


```
        truck =  new Vehicle(name: "Truck")

        distance = 200.and(52.meters)
}
car.move distance
```



Task 5 / Aufgabe 5 (7 Points / Punkte)

Describe a model-driven tool chain in software development and explain each of its items in a short sentence.

Beschreiben Sie eine „Model-driven Tool Chain“ in der Softwareentwicklung. Beschreiben Sie jedes Element der Tool Chain mit einem kurzen Satz.

Task 6 / Aufgabe6 (3 Points / Punkte)

What are typical application areas for scripting languages? Name 3 examples!

Nennen Sie 3 typische Anwendungsgebiete von Skriptsprachen.

Appendix

Task 4

Code fragments

A	`Number.metaClass.getMeters`
B	`NumberCategory`
C	`DistanceFactory.create`
D	`Def distanceClosure`
E	`new Car()`
F	`"Car"`
G	`Person: "Car"`
H	`[name: "Car"]`
I	`new Vehicle(name: "Truck")`
J	`new Vehicle(*truck)`
K	`[new Vehicle("Truck")]`
L	`"Truck".new`
M	`// a newscope`
N	`use(NumberCategory)`
O	`NumberCategory.each`
P	`truck.doesSomethingTo car`
Q	`truck."$doesSomethingTo" car`
R	`truck.doesSomethingTo { car }`
S	`"$doesSomethingTo"(truck, car)`
T	`MILLIMETERS, CENTIMETERS, METERS, DECIMETERS, KILOMETERS`
U	`GRAM, KILOGRAM, TON`
V	`(left)`
W	`.“down“`
X	`.left`
Y	` UP`

Expected output:
Car moves 252 METERS to LEFT
Truck stops Car!