

# 2XW05 Mid-Term Examination – Software Specification

Faculteit Wiskunde en Informatica  
Technische Universiteit Eindhoven (TU/e)

Model Examination - October 2009

**Exercise 1 (25 points)** Consider the following informal specification of a tool for handling Software Specification documents.

“A software specification document has a name, a date of creation and bears the date of its latest change. It also comprises one or more models; models can be added to or removed from a specification. Each model has a name and may reference (import) some models. There are three types of models: structural models, behavioral models and logical correctness specifications. Structural models by themselves have two types: Z Specifications and Alloy Specifications. A Z Specification has a list of schemas and has methods to add schemas, remove them, and to calculate their pre-conditions. Details of an Alloy specification need not further be specified.”

Design a class diagram which captures the above-given specification.

**Exercise 2 (60 points)** Consider the following state and operation schemas which (partially) specify the state of a car dealer and the operations to buy and sell a car. Give the definition of the sequential composition  $Sell \ ; \ Buy$  and simplify it as much as possible using the rules of logic.

$[Car, Price]$

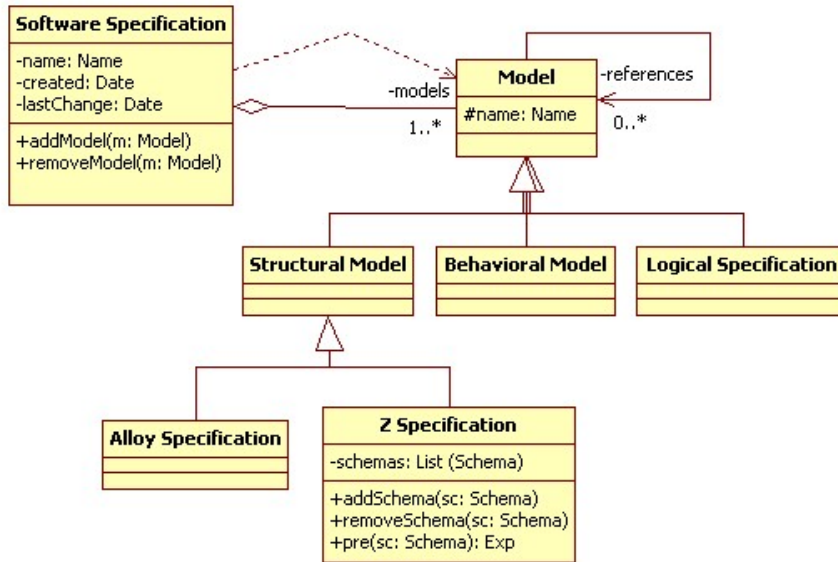
$ZDealer$
$showroom : \mathbb{P} Car$ $priceList : Car \mapsto Price$
$dom\ priceList = showroom$

$Sell$
$\Delta ZDealer$ $c? : Car$
$c? \in showroom$ $priceList' = priceList \setminus \{c? \mapsto priceList(c?)\}$

$Buy$
$\Delta ZDealer$ $c? : Car$ $p? : Price$
$c? \notin showroom$ $priceList' = priceList \cup \{c? \mapsto p?\}$

**Exercise 3 (35 points)** Formalize the specification of  $ZDealer$  and  $Sell$  in Alloy.

Answer 1



Answer 2

$Sell \ ; \ Buy$

= Def. of  $\ ;$

$\exists showroom'' : \mathbb{P} Car; priceList'' : Car \leftrightarrow Price \bullet Sell[''/] \wedge Buy[''/]$

= Def. of  $\wedge$

$\exists showroom'' : \mathbb{P} Car; priceList'' : Car \leftrightarrow Price \bullet$

$Sell[''/] \wedge Buy[''/]$
$showroom, showroom', showroom'' : \mathbb{P} Car$ $priceList, priceList', priceList'' : Car \leftrightarrow Price$ $c? : Car$ $p? : Price$
$dom priceList = showroom$ $dom priceList' = showroom'$ $dom priceList'' = showroom''$ $c? \in showroom$ $priceList'' = priceList \setminus \{c? \mapsto priceList(c?)\} \cup \{c? \notin showroom''\}$ $priceList' = priceList'' \cup \{c? \mapsto p?\}$

= Def. of  $\exists$

<p><i>Sell</i> ; <i>Buy</i></p> <p><math>showroom, showroom' : \mathbb{P} Car</math>  <math>priceList, priceList' : Car \leftrightarrow Price</math>  <math>c? : Car</math>  <math>p? : Price</math></p>
<p><math>\exists showroom'' : \mathbb{P} Car; priceList'' : Car \leftrightarrow Price \bullet</math>  <math>dom priceList = showroom</math>  <math>dom priceList' = showroom'</math>  <math>dom priceList'' = showroom''</math>  <math>c? \in showroom</math>  <math>priceList'' = priceList \setminus \{c? \mapsto priceList(c?)\} c? \notin showroom''</math>  <math>priceList' = priceList' \cup \{c? \mapsto p?\}</math></p>

= One point rule on  $priceList''$

<p><i>Sell</i> ; <i>Buy</i></p> <p><math>showroom, showroom' : \mathbb{P} Car</math>  <math>priceList, priceList' : Car \leftrightarrow Price</math>  <math>c? : Car</math>  <math>p? : Price</math></p>
<p><math>\exists showroom'' : \mathbb{P} Car \bullet</math>  <math>priceList \setminus \{c? \mapsto priceList(c?)\} \in Car \leftrightarrow Price</math>  <math>dom priceList = showroom</math>  <math>dom priceList' = showroom'</math>  <math>dom(priceList \setminus \{c? \mapsto priceList(c?)\}) = showroom''</math>  <math>c? \in showroom</math>  <math>c? \notin showroom''</math>  <math>priceList' = (priceList \setminus \{c? \mapsto priceList(c?)\}) \cup \{c? \mapsto p?\}</math></p>

=  $priceList$  is a partial function, hence  $priceList \setminus \{c? \mapsto priceList(c?)\}$  is a partial function, as well.  
 $f \setminus \{x \mapsto y\} \cup \{x \mapsto y'\} = priceList \oplus \{x \mapsto y'\}$   
 $dom(f \setminus \{x \mapsto y\}) = dom f \setminus x$

<p><i>Sell</i> ; <i>Buy</i></p> <p><math>showroom, showroom' : \mathbb{P} Car</math>  <math>priceList, priceList' : Car \leftrightarrow Price</math>  <math>c? : Car</math>  <math>p? : Price</math></p>
<p><math>\exists showroom'' : \mathbb{P} Car \bullet</math>  <math>dom priceList = showroom</math>  <math>dom priceList' = showroom'</math>  <math>dom priceList \setminus c? = showroom''</math>  <math>c? \in showroom</math>  <math>c? \notin showroom''</math>  <math>priceList' = priceList \oplus \{c? \mapsto p?\}</math></p>

= One point rule on  $showroom''$

$\text{Sell} \ ; \ \text{Buy}$
$\begin{aligned} & \text{showroom}, \text{showroom}' : \mathbb{P} \text{ Car} \\ & \text{priceList}, \text{priceList}' : \text{Car} \leftrightarrow \text{Price} \\ & c? : \text{Car} \\ & p? : \text{Price} \end{aligned}$
$\begin{aligned} & \text{showroom} \setminus c? \in \mathbb{P} \text{ Car} \\ & \text{dom priceList} = \text{showroom} \\ & \text{dom priceList}' = \text{showroom}' \\ & c? \in \text{showroom} \\ & c? \notin \text{showroom} \setminus c? \\ & \text{priceList}' = \text{priceList} \oplus \{c? \mapsto p?\} \end{aligned}$

= Def. of  $\Delta ZDealer$

$\text{Sell} \ ; \ \text{Buy}$
$\begin{aligned} & \Delta ZDealer \\ & c? : \text{Car} \\ & p? : \text{Price} \end{aligned}$
$\begin{aligned} & c? \in \text{showroom} \\ & \text{priceList}' = \text{priceList} \oplus \{c? \mapsto p?\} \end{aligned}$

### Answer 3 module ZDealerModule

```
sig Car, Price {}
```

```
sig ZDealer {
  showroom : some Car,
  priceList : Car -> lone Price
}
```

```
fact ZDealerPred {
  all zd: ZDealer | zd.priceList.Price = zd.showroom
}
```

```
pred Sell ( zd, zd' : ZDealer, c : Car ) {
  c in zd.showroom
  zd'.priceList = zd.priceList - c -> zd.priceList[c]
}
```