R. Hinze, S. Franck Summer Semester 2005

# Software Engineering

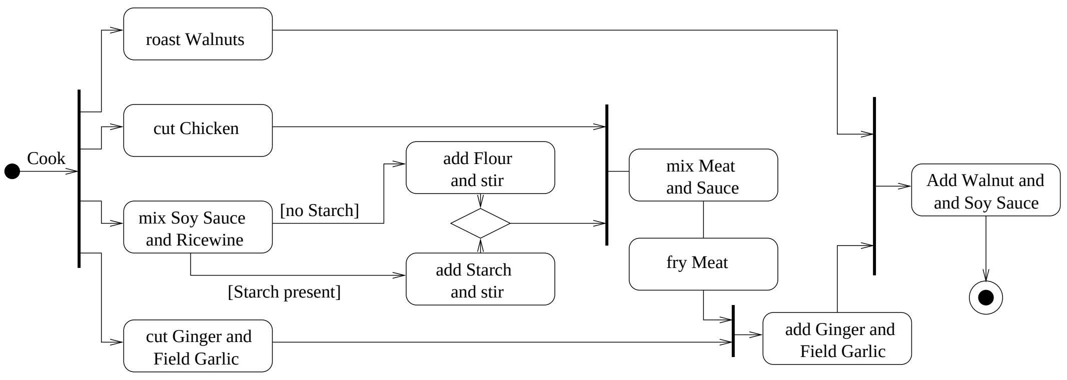
http://proglang.informatik.uni-freiburg.de/teaching/swt/2005/

Exercise Sheet 5 & 6

Deadline: May 31st, 2005

**Exercise 1 – Activity Diagramm:** (4 points)
Discribe the following recipe with an Activity Diagram:
Cut 400g chicken into small dices. Mix 30ml rice wine with 10ml soy sauce. Mix this sauce with starch. If no starch is available, use flour. Mix the chicken with the sauce. Cut 10g ginger and 50g field garlic. Sear 150g walnuts. Sear the meat, add ginger and field garlic and fry the meat. Add the walnuts and 10ml soy sauce.

**Solution:**



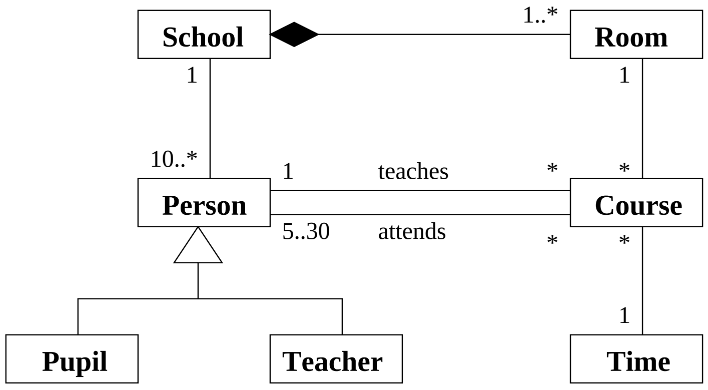
**Exercise 2 – Data Dictionnary:** (3 points)
Write a Data Dictionnary for german adresses.
Regard the different possibilities, like P.O. box or street, additional roomnumbers, etc.

**Solution:**

$$\begin{array}{rcl}
\langle \textit{adress} \rangle & ::= & \{(\text{c/o}\langle \textit{name} \rangle) + \langle \textit{name} \rangle + [\langle \textit{street} \rangle \mid \langle \textit{postbox} \rangle] + \langle \textit{town} \rangle + \langle \textit{country} \rangle\} \\
\langle \textit{name} \rangle & ::= & [(\langle \textit{forename} \rangle) + \langle \textit{lastname} \rangle \mid \langle \textit{company name} \rangle] \\
\langle \textit{street} \rangle & ::= & \langle \textit{streetname} \rangle + \langle \textit{housenumber} \rangle + (\langle \textit{roomnumber} \rangle) \\
\langle \textit{postbox} \rangle & ::= & \text{P.O.box} + \langle \textit{number} \rangle \\
\langle \textit{town} \rangle & ::= & \langle \textit{zipcode} \rangle + \langle \textit{town name} \rangle
\end{array}$$

**Exercise 3 – Class Diagram:** (5 points)

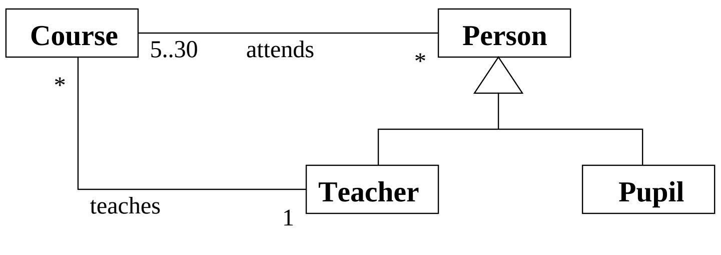
Analyse the following class diagram, explaining in detail what it models. Are there any flaws in this design?



**Solution:**

The diagram models a school. A school consists of at least 1 room. All rooms depend on the school. A school has also at least 10 persons, who can be teachers or students. Every person is in exactly one school. A person can teach or attend arbitrary many courses. Every course is taught by exactly one person and can be attended by 5 to 30 persons. A course takes place at a specified time. At any time arbitrary many courses may be held. Every course is taught in exactly one room, but there may be arbitrary many courses situated in the room.

This design has some flaws. First of all it is possible for a student to hold courses. This flaw can be resolved by connecting the teaches-association directly to the teacher:



A second flaw is, that it is possible to have several students, teachers or rooms involved in several courses at the same time. This can be resolved by using additional constraints, defined for example in OCL.

**Exercise 4 – Understanding Z:** (4 points)

What is defined by the following (generic) axiomatic definitions and what are the missing types?

$$
\begin{array}{|l}
walter : ??? \\
\hline
\forall M : \mathbb{P}\,\mathbb{N}, x : \mathbb{N} \bullet \\
\quad walter(\varnothing) = 0 \\
\quad x \in M \Rightarrow walter(M) = \\
\qquad\qquad x + walter(M \setminus \{x\})
\end{array}
$$

$$
\begin{array}{|l}
[X] \\
\hline
xaver : ??? \\
\hline
\forall s : \mathbb{F}\, X, n : \mathbb{N} \bullet \\
\quad n = xaver \Leftrightarrow \exists f : (1..n) \rightarrowtail\!\!\!\!\!\twoheadrightarrow s \bullet true \\
\hline
\end{array}
$$

$$
\begin{array}{|l}
yolanda : ??? \\
\hline
\forall m, n : \mathbb{N} \bullet \\
\quad yolanda(m, n) = \{i : \mathbb{N} \mid m \leq i \leq n\}
\end{array}
$$

$$
\begin{array}{|l}
[X] \\
\hline
zoe : ??? \\
\hline
\forall R : X \leftrightarrow X \bullet \\
\quad zoe(R) = \bigcap\{T : X \leftrightarrow X \mid \\
\qquad\qquad (R \cup \mathsf{id}\, X) \subseteq T \wedge \\
\qquad\qquad T \mathbin{⨾} T \subseteq T\} \\
\hline
\end{array}
$$

**Solution:**

| | **type** | **function** |
|---|---|---|
| walter | $\mathbb{P}\,\mathbb{N} \rightarrow \mathbb{N}$ | calculates the sum of all numbers in the argument set |
| xaver | $\mathbb{F}\, X \rightarrow \mathbb{N}$ | calculates the number of elements in a set |
| yolanda | $(\mathbb{N} \times \mathbb{N}) \rightarrow \mathbb{P}\,\mathbb{N}$ | calculates the set of all numbers between its parameters (inclusive) |
| zoe | $(X \leftrightarrow X) \rightarrow (X \leftrightarrow X)$ | calculates the reflexive transitive closure |

**Exercise 5 – Queue in Z:** (4 points)

Specify a queue with Z-Schemata. The queue stores numbers. The functions *Init*, *Add* and *Next* have to be implemented on the queue. Every function returns a variable *result*! of type *REPORT*, which may be *OK*, an error message or *Return* $i$ where $i$ is a number.

The functions work as follows:

*Init* instantiates an empty queue.

*Add* appends a given number to the end of the queue.

*Next* returns the first element of the queue and removes it.

**Solution:**

$$REPORT ::= Return\langle\!\langle \mathbb{Z} \rangle\!\rangle \mid OK \mid QueueUnderflow$$

*Queue*

$content : \mathrm{seq}\, \mathbb{Z}$

---

*Add*

$\Delta Queue$
$input? : \mathbb{Z}$
$result! : REPORT$

$content' = content \cup \{(\#content + 1 \mapsto input?)\}$
$result! = OK$

---

*NextFail*

$\Xi Queue$
$result! : REPORT$

$content \notin \mathrm{seq}_1\, \mathbb{Z}$
$result! = QueueUnderflow$

---

*Init*

$Queue$
$result! : Report$

$content = \{\}$
$result! = OK$

---

*NextOK*

$\Delta Queue$
$result! : REPORT$

$content \in \mathrm{seq}_1\, \mathbb{Z} \land$
$result! = Return(head\ content)$
$content' = tail\ content$

---

$Next = NextOK \lor NextFail$