08.05.2015	**Advanced Software Engineering** **University of Vienna** **Forschungsgruppe Software Architecture**		
Kennzahl	*Matrikelnummer*	*FAMILIENNAME*	*Vorname*

8	12	8	12	4	14		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!

Please read this section. You have **60** minutes, in which to answer the questions.

Tips:

- Before you start, please fill in your name, surname and student ID number.
- No teaching material is permitted during the test!
- Technical aids such as translators, pocket calculators, mobile telephones, etc., are not allowed!
- If you do not understand a question, please ask the supervisors for explanations.
- You can answer the questions in German as well as in English.
- Please write using indelible writing utensils (no pencil) .
- You can rip off the appendix (last page) from the test document for easier use.

- Good luck!

Task 1 / Aufgabe 1 (8 Points / Punkte)

a) Explain the difference between statically and dynamically typed languages, and the advantages and disadvantages of each.
b) What are the main characteristics of dynamically typed languages?

a) Erklären Sie den Unterschied zwischen Statically-Typed- und Dynamically-Typed-Sprachen, und die Vorteile bzw. Nachteile jeder Sprachgruppe.
b) Welche sind die Hauptmerkmale der Dynamically-Typed-Sprachen?

Task 2 / Aufgabe 2 (12 Points / Punkte)

Moodle is a web application used in educational settings.
Document the architectural design decision that is described in the text below using the (simplified) AD template from Tyree / Ackermann.

Moodle ist eine Web-Applikation sie für Bildungszwecke verwendet wird.
Dokumentieren Sie die Architectural Decision, die im Text unterhalb beschrieben wurde, mittels der (vereinfachten) Vorlage zur Dokumentation von Ads von Tyree / Ackermann.

Like many successful open source projects, Moodle is built out of many plugins, working together with the core of the system. This is a good approach because at allows people to change and enhance Moodle in defined ways. An important advantage of an open source system is that you can tailor it to your particular needs. Making extensive customisations to the code can, however, lead to big problems when the time comes to upgrade, even when using a good version control system. By allowing as many customisations and new features as possible to be implemented as self-contained plugins that interact with the Moodle core through a defined API, it is easier for people to customise Moodle to their needs, and to share customisations, while still being able to upgrade the core Moodle system.

There are various ways a system can be built as a core surrounded by plugins. Moodle has a relatively fat core, and the plugins are strongly-typed. When I say a fat core, I mean that there is a lot of functionality in the core. This contrasts with the kind of architecture where just about everything, except for a small plugin-loader stub, is a plugin.

When I say plugins are strongly typed, I mean that depending on which type of functionality you want to implement, you have to write a different type of plugin, and implement a different API. For example, a new Activity module plugin would be very different from a new Authentication plugin or a new Question type. At the last count there are about 35 different types of plugin. This contrasts with the kind of architecture where all plugins use basically the same API and then, perhaps, subscribe to the subset of hooks or events they are interested in.

Generally, the trend in Moodle has been to try to shrink the core, by moving more functionality into plugins. This effort has only been somewhat successful, however, because an increasing feature-set tends to expand the core. The other trend has been to try to standardise the different types of plugin as much as possible, so that in areas of common functionality, like install and upgrade, all types of plugins work the same way.

A plugin in Moodle takes the form of a folder containing files. The plugin has a type and a name, which together make up the "Frankenstyle" component name of the plugin. The plugin type and name determine the path to the plugin folder. The plugin type gives a prefix, and the foldername is the plugin name. Here are some examples:

Plugin type	Plugin name	Frankenstyle	Folder
mod (Activity module)	`forum`	`mod_forum`	`mod/forum`
mod (Activity module)	`quiz`	`mod_quiz`	`mod/quiz`
block (Side-block)	`navigation`	`block_navigation`	`blocks/navigation`
qtype (Question type)	`shortanswer`	`qtype_shortanswer`	`question/type` /shortanswer
quiz (Quiz report)	`statistics`	`quiz_statistics`	`mod/quiz/report/statistics`

The last example shows that each activity module is allowed to declare sub-plugin types. At the moment only activity modules can do this, for two reasons. If all plugins could have sub-plugins that might cause performance problems. Activity modules are the main educational activities in Moodle, and so are the most important type of plugin, thus they get special privileges.

AD shortname		AD name	
Problem Statement			
Decision drivers			
Alternatives			
Recommendation			
Decision Outcomes			
Decision Outcomes			
	Status		
	Chosen alternative		
	Justification		
	Consequences		
	Assumptions		

Task 3 / Aufgabe 3 (8 Points / Punkte)

What are the criteria that have to be considered when choosing between a scripting language and a system programming language for a task at hand? Explain for each criterion why it will cause the use of a scripting language or a system programming language.

Welche Kriterien sind zu beachten, wenn Sie sich zur Entwicklung eines Programmes zwischen einer Skriptsprache und einer Systemprogrammiersprache entscheiden müssen? Erklären Sie für jedes Kriterium, warum dies zum Einsatz einer Skriptsprache oder einer Systemprogrammiersprache führt.

Task 4 / Aufgabe 4 (12 Points / Punkte)

a) Explain the term Architectural / Design Analysis! Name and explain the goals of architectural analysis!
b) Name and explain the techniques for performing architectural analysis!

a) Erklären Sie den Begriff “Architekur- / Designanalyse”. Nennen und erklären Sie die Ziele der Architektur-/ Designanalyse.
b) Nennen und erklären Sie die Techniken, welche für Designanalyse eingesetzt werden können.

Task 5 / Aufgabe 5 (4 Points / Punkte)

Explain the principles of Loose Coupling and High Cohesion in your own words.

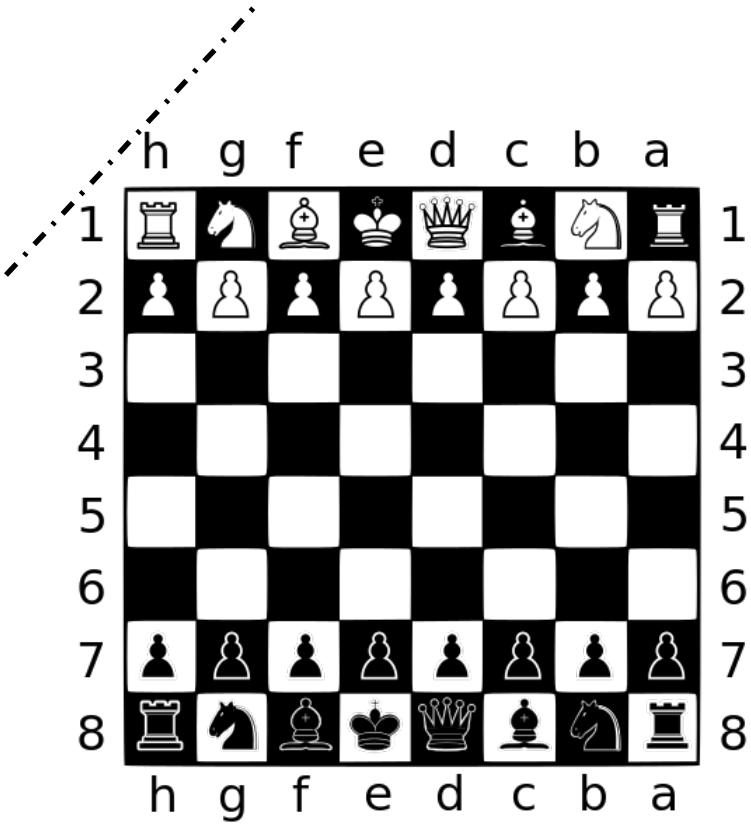
Erklären Sie in eigenen Worten die Prinzipe von Loose Coupling und High Cohesion.

Task 6 / Aufgabe 6 (14 Points / Punkte)

Create a valid Xtext grammar that is able to describe the following chess game in the appendix (next page). Ignore the constraints on the pieces' moves.

Erstellen Sie eine gültige Xtext-Grammatik für das Schachspiel im Anhang (auf der nächsten Seite). Die Einschränkungen bei den Bewegungen der Schachfiguren sind nicht zu beachten.

Appendix



Game 4 (white player: John Smith, black player: Jack Johnson)
....
Round 5:
White knight at g1 moves to f3
Black rook at h8 captures white pawn at h5
....
Round 17:
White queen at d4 captures black bishop at c5
Black king at d8 moves to e8
....