TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science Institute of Software and Multimedia Technology, Software Technology Group

WS2020/21 – Design Patterns and Frameworks

Tools and Materials

Professor: Prof. Dr. Uwe Aßmann
Lecturer: Dr.-Ing. Sebastian Götz
Tutor: Dr.-Ing. Karsten Wendt

Task 1 Tools and Materials Fundamentals

This exercise focuses on the *Tools and Materials* metaphor introduced by Züllighoven et al. [2] and discussed in the lecture.

a) Explain the terms *Leitmotif* (Leitbild), *usage model* (Benutzungsmodell), and *design metaphor* (Entwurfsmetapher). What are the relevant instances in the context of the tools and materials approach?

Solution: The definitions can be found in [2]:

- **Usage Model (Benutzungsmodell)** is a domain-oriented model of how the software can be used to perform activities in its usage context. It is sensible to use design metaphors based on a Leitmotif to realise a usage model.
- **Leitmotif (Leitbild)** is (in software engineering) a common frame of orientation for the parties concerned in development and usage of the application. It supports design, usage, and evaluation of software and is based on values and objectives. An example Leitbild is the notion of the "Scientist" as a person who, without prejudice or religion, analyses and explains the world from objective mathematical or empirical findings.
- **Design Metaphor (Entwurfsmetapher)** is a figurative concept used to make a leitmotif more concrete. A design metaphor is always connected to a constructive interpretation in the form of blue prints and design patterns.

The *Tools and Materials* approach employs the leitmotif of the *workplace for independent expert work* (dt. Arbeitsplatz für eigenverantwortliche Expertentätigkeit). The main objective is to create a *supportive* and *flexible* work environment, where the user is an expert in his/her domain and requires to use tools flexibly as he/she sees fit. The two main design metaphors are:

- A **material** is anything that can be manipulated by the expert.
- A **tool** is something that enables such a manipulation. Tools may be materials themselves, for example, they may be monitored or configured themselves.

b) How do *tools* use *materials*? What is the design pattern [1] utilized to couple tools and materials, denoted *connection pattern* [2]?

Solution: Users want to be able to manipulate a material with different tools. Also, the same tool may be used for different kinds of materials. This is achieved by making tools, for instance, a *File Reader* tool work on aspects, such as, something Readable, which are implemented by materials, such as, a `FinancialRecord`. To permit adding new tools to the environment, which might use materials in an unforeseen way, materials can no longer be implemented with a fixed set of `interfaces`. By contrast, the materials interfaces or roles must be extensible. This can be achieved, for instance, by using the `Adapter`, `Decorator`, `Extension Object`, or the `Role Object` pattern.

c) What are the parts a tool consists of? How is the communication between these parts organized? What is the benefit of this separation?

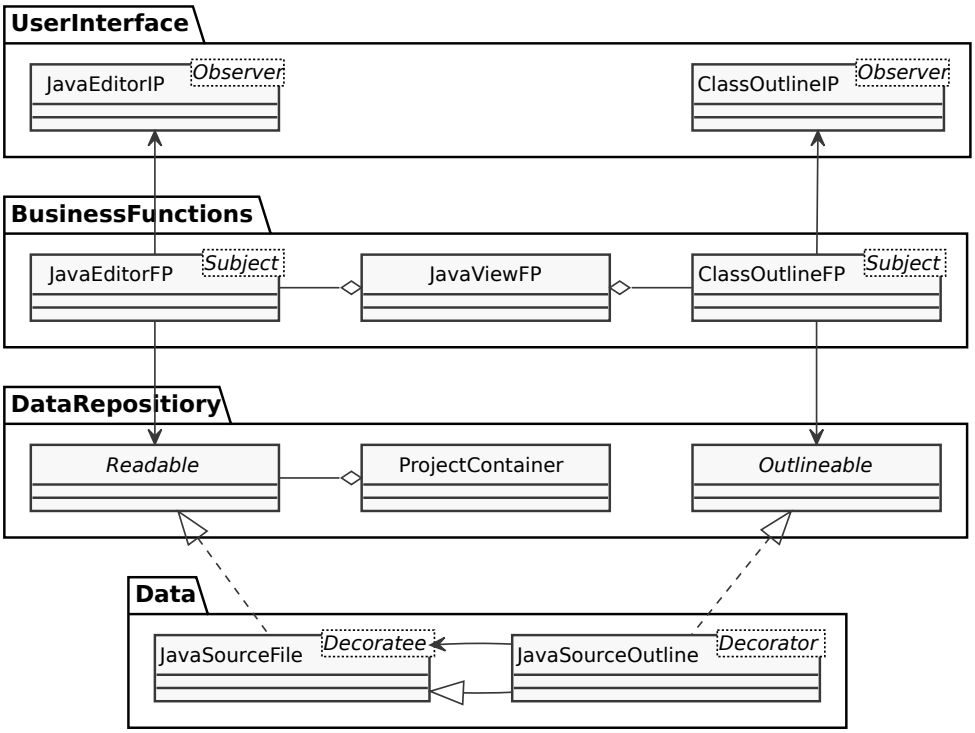
Solution: Tools consist of a **functional part** (FP) and an **interaction part** (IP). This separates the tools functionality from the precise means of accessing this functionality. The IP directly accesses the FP, however, the FP communicates with the IP only via an `Observer` pattern, where the IP observes the FP. The following class diagram illustrates this, by means of a simple *Java class editor*, consisting of two subtools, i.e., `JavaEditor` and `ClassOutline`.

d) Explain the *layers* in an application built following the tools and materials approach. What constraints must be enforced regarding *communication* and *generalization*?

Solution: There are four layers, as depicted below, whereas each layer encompasses the following elements:

1) user interface *(tool interaction part)*,
2) business functions *(tool functional part)*,
3) data repositories *(material containers)*,
4) and data *(materials)*.

Direct communication, including the directed associations for purposes of navigation, always goes from higher layers to lower layers. Communication in the other direction uses *n-T–H*, or *TH* hooks.



Task 2 Homework for Yourself

As homework for yourself, consider the *Integrated Development Environment* (IDE) you are using, e.g., *Eclipse*[1] or *Intellij IDEA*,[2] as an instance of the tools and materials approach. Investigate how an IDE facilitates a useful workbench to programmers.

[1] https://www.eclipse.org/downloads/
[2] https://www.jetbrains.com/idea/

a) Investigate your IDE and enumerate the materials the IDE allows you to work on, for instance, *source code*.

   exercise.

b) Investigate the tools your IDE provides to work on these materials. Are there tools that do not work on a material?

   exercise.

c) Is there a dedicated *Tool Coordinator* and how are new *tools* registered?

   exercise.

d) Finally, classify the *Debugger* as a tool. What are the materials it is working on?

   exercise.

References

[1] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design patterns: Elements of Reusable Object-Oriented Software*. Pearson Education, 1994.

[2] Heinz Züllighoven et al. *Das objektorientierte Konstruktionshandbuch nach dem Werkzeug & Material-Ansatz; dpunkt-Verlag*. dpunkt.lehrbuch. dpunkt, 1998. ISBN 978-3-932588-05-1.