A “COMPETENCE ATLAS” AS AN INSTRUMENT IN SOFTWARE ENGINEERING EDUCATION

Lisa Henschel, Kristina M. Schulz

University of Applied Sciences and Arts Coburg (GERMANY)

Abstract

The competence profile required of graduates in software engineering is that complex that it is difficult for the students to access. Individual courses can only address small sections of the overall picture of the competences to be achieved. For this reason, the purpose of individual methods and contents is difficult to classify into the overall curriculum of software engineering education. Even if the meaning of competences and methods to be trained become clear, students usually are not able to locate them into the complex competence profile demanded. Furthermore, it is difficult for students to imagine how the skills required from them will be put to practical use later on. However, it is important for the learning process to develop this overview of the competence goals to be achieved and their practical foundation, in order to be able to derive individual goals for learning and competence development. For this purpose, the “Competence Atlas” - a manual for students in software engineering at the University of Applied Sciences Coburg was developed. This manual is intended to present both the most important contents of software development in an overview, as well as a clear reference of theory and practice for the students and to provide an orientation for their entire study. This facilitates the students’ understanding of the methods used and the level of competence in software engineering to be achieved. In this way, the “Atlas” can help students to deal with the skills required in software engineering and promote the reflection process of the students, what can improve software engineering education directly.

Keywords: Competence Profile, Competence Atlas, Software Engineering.

1 INTRODUCTION

In the course of digitizing, software is becoming increasingly important and is characterized by growing complexity. Without software, the use of digital technologies would be impossible. Software engineering is a knowledge-intensive discipline characterized by rapid development cycles and loss of knowledge [1].

Beside the dynamic of change and the rapid development cycles of the discipline, the competence profile required in software engineering is that complex that it is difficult for students to understand in its entirety. In individual courses, only partial aspects of competences to be trained can be shown and tasks are difficult to place in the overarching context. In order to derive individual learning goals, however, it is important to develop an overview of the goals to be achieved.

Discourses that demonstrate the necessary competence profile in software engineering mainly focus on technical knowledge and often present the results from a scientific perspective, which is hardly accessible to students. This leads to the necessity to translate the scientific results in relation to the required competence profile in software engineering into the language of the students and to adapt the level of abstraction to the state of the students’ understanding. It is also indispensable to present the competence profile holistically and to describe both technical and interdisciplinary competences. In order to meet these requirements and to improve the learning and teaching in software engineering education, a “Competence Atlas” was developed with the aim of providing the students with an overall picture of the competences to be trained in an appropriate language and with a practical reference. For this, existing scientific findings of competence research where validated with the results of interviews with practice partners from various disciplines in software engineering in order to link both, the scientific and the practical perspectives. This contribution presents a novel approach which combines technical content from the field of software engineering with required competences and practical relevance of the field. This should provide a frame of reference through which the competences addressed in a course in software engineering education can be integrated into the general context of the subject.
2 THEORETICAL FRAMEWORK

2.1 Competence Orientation – Definition and Development

In higher education hardly any other concept has been discussed more frequently in recent decades than the concept of competence. Common to these discussions is the description of a rapid economic and social change that is increasingly accompanied by ambiguity and uncertainty [2].

There is no uniform definition of the concept of competence. Preißer & Völzke emphasize that the respective theoretical tradition and the relation to the field of practice influence the definition of the competence [3]. However, there is some consensus about what constitutes and includes competence. Gillen understands competences as action potentials that are related to the subject and the ability and willingness to act independently [4]. Accordingly, competence always includes performance. The understanding of competences in this paper builds on this approach and defines competences as a compound of trainable subjective action potentials with concrete, external conditions of action [5]. Competences are understood as resources that enable subjects to act situationally based on the individual capacity to act in different contexts.

Each individual therefore has specific knowledge, skills, values, attitudes and motivations that can be used in concrete situations (see Fig. 1) [6].

![Diagram](image)

**Figure 1. The connection between competence and action (based on Gnaho 2010).**

Competences develop through different learning paths and modes of appropriation. Gnaho identifies five ways of acquiring skills, where each way emphasizes different components of competences: competence development through socialization, through formal learning, through non-formal learning, through informal learning and through implicit learning. These categories are mutually dependent. The respective ways of competence development depend on the subject and on its personal and structural prerequisites, such as the age or social origin [6].

Competences evolve especially when learning processes relate to the life of the learning subjects and enable subjective experiences, as this is the only way to recognize the possible ways of using the learning content [5]. Dehnbostel also focuses on the perspective of the subjects and their competence development, since competences are always developed in the immediate life context and work execution and through individual learning processes [7]. Accordingly, the development of competences is a process in which complex causal relationships exist, which require attention to the individual life and work context of the learning subjects.

In addition to the subject reference, motivation is considered to be a necessary condition for the acquisition of competence. Skills, abilities, and learning strategies are initially a collection of possible approaches, but whether or not they are used depends on the motivational circumstances [8].

2.2 Learning with Concrete Application Reference as a Motivational Factor

The process of learning encompasses a transformation of already existent knowledge to expanded knowledge, which requires cognitive abilities that enable the absorption and processing of information.
Learners must be able to reflect their knowledge assets and competences with regard to their learning goals in order to determine the starting point of their necessary learning act [9]. This process, in which new information must be linked to existing knowledge, can be supported by learning strategies.

In addition to learning strategies, motivation should be regarded as a necessary condition for knowledge acquisition and as important as cognitive abilities. Motivational conditions are distinguished in personal factors and in conditions that are located in the learning situation and can be both extrinsic and intrinsic. The intrinsic motivation, that is promoted especially by a strongly felt self-efficacy and combined with volitional strategies, is an essential condition for self-directed learning. In this context, Deci and Ryan define three criteria that influence intrinsic motivation namely perceived competence/effectiveness, perceived autonomy/self-determination and perceived social involvement [10].

Perceived competence/effectiveness depends e.g. on the right level of structure, suitable work material tailored to students and experiences of success, which can be seen in positive feedback. Social involvement of students is achieved if, e.g. they can actively participate in a course and if there is an open learning atmosphere. Transparent communication of the learning goals and a classification of the learning content in the overall context of the course also have positive effects on the perceived self-efficacy of the students [11]. Knowledge must be linked with the characteristics of the field of practice and the interest in exploiting and cognition of the new knowledge to be learned must always be clarified in this context since only then the motivation of the students and the transfer of learning can be promoted.

2.3 The Promotion of the Understanding for Professional Competences in Software Engineering

In order to promote comprehensive competences, which can be applied later in complex everyday and professional situations, it is necessary to establish a practical relevance to the theoretical contents of the study. Because the high degree of abstraction and complexity of theoretical content often makes it difficult for students to recognize the meaningfulness, usefulness and usability of the learned for their future professional life [12], authentic problems can be used as a starting point and point of reference for learning.

In order to be able to better illustrate to students, but also to potential employers, what the knowledge and competences acquired at universities can be used for, it is helpful to regularly highlight the relevance of the content and possible application contexts.

Questions that are helpful to create practical reference in the individual courses can be formulated as follows:

- What role do learning content and related competences play in the subject?
- How do the competences taught in the curriculum relate to other competences in the curriculum?
- In which field of the professions is the learning content relevant?
- What professional opportunities or innovations are associated with the relevant competences?
- What effects can learning content and competences have in the personal life of the students?

The emphasis of the connection between theoretical content and practical relevance is important from a didactic point of view since learning and competence development can be supported by two factors, namely the connection factor and relevance factor [13].

The connection factor refers to the association fields in which new knowledge is integrated. New information is therefore always linked with previous knowledge. The relevance factor refers to the mechanism that new knowledge contents are always checked for their meaningfulness. If learning content is considered meaningless, content will not be linked to existing structures and the new knowledge will remain unused [13].

The development towards this practice-orientation is accompanied by a change in the teaching culture at universities, which places learners at the center of their own learning process and demands an active, self-directed design of the individual learning process from them. This includes a movement from a knowledge transfer didactic towards a didactic of self-directed acquisition of knowledge and competences, which is based on a constructivist didactic. This approach assumes that each person constructs new knowledge by linking it with previous experience [14]. This ability for self-directed learning is very important for the improvement of the individual study ability, the preparation for
professional life, and the orientation in the individualized and ever-changing world. This development and the abovementioned ability are accompanied by a stronger identification with the occupational field, a promotion of self-responsibility, an increase in intrinsic motivation and a promotion of the individual learning competence [15].

Vocational training is therefore always a combination of potential and motivation to overcome the challenges posed at work. In which expression readiness and ability as competences to act of an individual are combined, can only be verified in the occupational situation [16]. With the aim of establishing a connection of theoretical content in software engineering with professional requirements and competences, a “Competence Atlas” was developed within the project EVELIN (Experimental Improvement of Learning Software Engineering). The focus is on the creation of a connection between theoretical content and workplace-relevant needs of the professional field in software engineering and furthermore on the promotion of students’ motivation and understanding of the professional competence requirements.

The previous chapter outlines how motivational factors are related to competence development of students and to what extent the development of an understanding of skills required in practice in software engineering can be promoted. In order to explain the procedure for the development of the “Competence Atlas”, the methodology is explained in the following chapter.

3 METHODOLOGY

At the beginning of the compilation of the “Competence Atlas”, a substantive structure was developed, which is oriented towards the software development cycle. The atlas was subdivided into five phases of software development (requirements analysis, modeling / design, implementation, test phase and commissioning) and three phase-spanning activities of software development (project management, quality management and configuration management).

This structure was chosen to break down the complexity of the development cycle as well as to explain the interacting elements and tasks in the software development process. For the respective areas subject-related contents, tasks and required competences were described in detail, in order to coherently present the overall picture of the competences to be trained in software engineering.

The contents were described on the basis of a literature analysis in which studies on the necessary competence profile in software engineering education [17–19] as well as specialist texts on the importance of the respective phase for software development [20–22] were considered. The theoretical contents of the respective chapters of the “Competence Atlas” were then linked to courses offered by the University of Coburg, specifically designed to promote the development of the respective addressed competences. This should establish a direct relationship between the theoretical content and the required competences for the students, who should use the atlas as study-accompanying guideline. This should help students understand the context of their studies and help them understand which part of the overall context corresponds to the course they are attending. This can increase learners’ motivation as they can better understand the context of the curriculum and the links to the practicality of learning.

In order to establish the practical relevance of the learning content and to give the students insights into actual professional practice, guided interviews were conducted with practitioners from various areas of software engineering (requirement analysis, application development, business informatics). Subsequently, the interviews were transcribed and evaluated using the Grounded Theory paradigm.

4 RESULTS

The consideration of the collected data with regard to the competence profile required in software engineering has revealed a complex category system. The system of categories was created through bundling, integration and construction, and contains relevant statements for the subsequent interpretation. The following categories were formed during the data analysis:
### 4.1 Role and Task Descriptions in Software Engineering

The tasks and role profile required in the practice of software engineering is complex. Depending on the area of responsibility, differentiated technical and multidisciplinary competences are required. These are not mentioned in detail below, but rather a change in the task profile that occurs in the practice of software engineering in recent years is discussed.

It turned out that the demands on software developers are getting wider. While one specialization in one area of expertise in the past was usually sufficient, software developers today demand a broad base in order to be flexible to use in a variety of development projects. A one-sided specialization in one special field of software engineering leads to the mutual disability in the software development process, since differentiated tasks with differentiated emphases in the development process have to be fulfilled. A distribution of tasks based on the skills of the employees therefore represents a significant factor of efficiency in software engineering.

In addition to specialist competences, working in a team is becoming increasingly important. The development of software requires a high degree of multi-disciplinary thinking and acting, as well as a systematic collaboration of different roles (requirement analysts, software architects, testers etc.) within a software development process. For this reason, the ability to work and communicate in a team is a necessary competence [17]. Because of the heterogeneity in interdisciplinary software development teams, teamwork is absolutely necessary to avoid project-threatening conflicts and to realize a good cooperation between the team members [23].

### 4.2 The Importance of Practical Relevance in Learning

Gaining practical experience is crucial for the future career orientation. After school time, which is considered primarily as a learning time in which the pupils are less encouraged to practice actively and because of the low practical relevance, it is important for students to be able to gain experience in practice while studying. Internships allow to compensate the lack of practice in school education and courses of university study. The data analysis showed that a subdivision into differentiated work modules, each with different tasks, is to be regarded as meaningful, so that a diverse collection of experiences is possible. A connection between the development-oriented promotion of the competences and the action-oriented practical testing of these enables the students to try themselves and to deal with their individual competences in a self-reflexive way.
Here again the reference to the utilization interest of the learning subjects emerges, since the motivation for learning and learning transfer depends to a large extent on this interest [5]. This means that competences develop only when the students recognize a context of utilization of the subject of learning and, on the basis of this, develop their learning and transfer motivation, which is the basic prerequisite for the transfer and application of what has been learned to everyday work and the private life.

The application relevance and the practical transmission possibilities of the learning content are significant as condition factors for the learning transfer and the competence development. In the learning field, the content-related connection of already existing knowledge structures with the subject matter of learning and the method-based training of what has been learned are in the foreground [5, 24].

The features of the functional field, the field in which the learning contents are used, are also determining factors for the learning transfer. Here, the central question is whether the scope for the application of knowledge is given [5].

4.3 The Role of the Customer in the Development Process

When it comes to the role of the customer in the development process, the interviewees emphasize various aspects that relate both to the communication between the developer and the customer and to the technical level of the requirements analysis. It was emphasized that there is generally a polarity between the developers of the software and the customers, which has to be bridged through good communication. In order to promote a successful communication process, which is particularly important for the collection of requirements, an understanding of the requirements of the customers must be developed so that requirements can be implemented to the full satisfaction of the customers. This means that customers must be involved in the development process at all times in order to reach the agreement between customers and developers [21].

Challenges in this part of software development can lead to additional work and conflicts due to misunderstandings during the requirements analysis and documentation. Another important challenge lies in the communication of the planned development solution, which has to be presented in an understandable way to the customers.

4.4 Challenges and Recommendations for Young Professionals

When asked what challenges young professionals would have to deal with in software engineering, the interviewees mentioned several aspects. Career entrants often have difficulties with the fact that they are not assigned to a fixed area of responsibility, what can be accompanied by uncertainty at the beginning of their work life. A major challenge is also the high complexity with which software engineers have to deal, with complexity relying on both software systems and the complex processes surrounding software development. According to the interviewees, a proficiency in handling these types of complexity usually can only be acquired in working life, since complexity cannot be mirrored realistically in university education.

The interviewees also gave tips on how to deal with these challenges described above. Thus, an examination of the basics of software engineering is helpful, which means not only programming skills, but also an examination of process models. In addition, junior professionals should strive to develop their own specialization, but they should focus not only on technical issues, but also on soft skills. According to the interviewees, these include competences of communication, self-organization, the awareness of the importance of holistic competences and openness to further development opportunities. In addition, students in the software engineering should gain practical experience as early as possible, e.g. through internships and occupation as working students or writing a practical thesis. These practical experiences can change the view of the theoretical content in the study [13], help for a later positioning in the professional field and enable a reflection of one’s own strengths and weaknesses. As part of a practical activity, students and young professionals should gain new experience as quickly as possible and be introduced to a professional activity through light tasks. The software experts also emphasized the quality of proactivity, which is extremely important in everyday working life, especially for junior professionals.

4.5 Skills that Help Dealing with Practical Challenges

One part of the guided interviews was to inquire about the skills necessary in dealing with the challenges in the field of software engineering. The interviewees identified many important competences which are needed to work successfully in their field of profession. The necessary skills mentioned could be
grouped into specialist, cross-disciplinary and personal competence areas. Specific skills include specialist expertise relevant to tasks, which include a wide range of technical know-how and necessary knowledge in relation to the respective contract domain. In addition, every software engineer should develop a technical focus with in-depth expertise, which plays an important role in the course of professional practice. This is highly relevant because detailed software development requires specific expertise and knowledge of necessary programming steps to ensure the functionality of the software to be developed.

The ability to work in a purposeful and structured manner, which also includes the ability to prioritize, were cited as interdisciplinary and inter-task competences. Also the ability to reflect one's own abilities and needs, the examination of new content, a general willingness to learn, especially with regard to the handling of new specialist domains and the interest in further training opportunities have been named as important for a successful professional life as a software engineer. In addition, software engineers also need communicative skills, which include not only foreign language skills but also presentation skills and the ability to explain facts in an understandable and clear way to customers and colleagues. Among the skills that are personally necessary, cross-phase and cross-disciplinary skills such as openness, resilience, teamwork skills and self-reliance were named. Openness and flexibility are particularly important in view of the low level of planning security and the constant engagement with new knowledge domains, depending on the client [22]. Teamwork skills furthermore play a major role throughout the software development process, as all employees from the different development stations must be constantly in contact with each other to ensure a satisfactory software development.

After these insights into the results of the data collection in the form of conduct-based interviews, it is important to stress that these results have been integrated directly into the “Competence Atlas”. In this way, they underline the theoretical basis and enable a direct connection between the curricular courses of software engineering education and the professional practice.

5 IMPLEMENTATION AND STRUCTURE OF THE “COMPETENCE ATLAS”

5.1 Structure of the “Competence Atlas”

The “Atlas” is divided into two major substantive chapters, of which one chapter describes the five phases of software development and the other chapter delineates the three cross-phase activities during the software development process. This structure was developed in accordance with the development cycle in software engineering in order to illustrate the complexity of these processes without neglecting important theoretical details.

When describing the respective phases of software development, each subchapter is structured according to the same composition: First, an identification figure, which is explained in more detail below, describes its current working life during a several month’s practical training at a software development company in Coburg. There the persona gets to know different departments according to the different phases of software development. The protagonist describes challenges of the respective phase and explains what is particularly important in the phase (see example excerpt 1). At the beginning of each subchapter there are also references to courses in software engineering training at the Coburg University, which are relevant for the respective phase and the necessary competence development (see example excerpt 2). This introduction to the respective phase is followed by a more detailed description of the individual tasks, requirements and needed competences of the development phase. The relevant facts were explained scientifically sound, but adapted to the language and level of abstraction of the students (see example excerpt 3). Graphic models visually set out the content in order to present the complex processes in a simplified way and thus to promote an understanding of them. The end of each chapter an "At a glance" page, conclusively presents the most important information and challenges of the individual phases of software development in a compact version.
5.2 The Introduction of a Prototypical Persona as an Identification Figure

In software engineering, especially in the field of requirements engineering, it is common to visualize future users for the software product to be developed in the form of a prototypical user figure. These personas are user archetypes that describe the different goals and observed behavioral patterns of potential users and customers. Personas are therefore not real people, but ideal types, but based on the behaviors and goals of real people [25].

The “Competence Atlas” uses this approach and introduces “Max Scheuer” as a prototypical student in software engineering at the University of Coburg right at the beginning. Max is currently completing an internship at a software development company in Coburg. In doing so, he goes through all areas of work, relevant to software development, in line with the development phases described in the “Atlas”. In the course of his internship, Max learns about the challenges of software development that can occur in each phase. By actively addressing these challenges, he sets out ideal-typical how these challenges...
can be dealt with. On the one hand, the person “Max Scheuer” serves as an identification figure, since Max also completes his studies at Coburg University of Applied Sciences and so the students can deal with him, the course of his studies and his experiences. By linking theoretical content of the study program with practical tasks in the internship, Max makes it easier for students to establish a link between theory and practice. Complex problems are broken down by the simplified presentation and explanation by Max and his simplified language and are thus made comprehensible to the students. This allows students right from the beginning of their studies to put themselves in the situations described in the “Competence Atlas”.

6 CONCLUSIONS

The consideration of the theoretical foundations has shown, that competences develop through different learning paths and modes of appropriation. It became clear, that competences evolve especially when learning processes relate to the life of the learning subjects and have a clear practical relevance, what in addition promotes the motivation of the learners. It was emphasized, that learning and competence development can be supported by the connection factor and relevance factor, which both refer to the relevant and meaningful connectivity of new learning contents. To cope with the requirements in today’s knowledge society, self-directed learning abilities have been stated to have a great relevance. These abilities are very important for the improvement of the individual study ability and the preparation for professional life. These theoretical assumptions are supported by the evaluation of the interviews, as the interviewees also repeatedly stated how important and necessary the ability to independently become familiar with new domains of knowledge in later professional life is and that it is necessary to have a general interest in continued learning processes. The “Competence Atlas” for software engineering attempts to meet the requirements for learning and competence development outlined above and addresses the demands for the practical applicability of learning content, transparent communication of the planned courses of study, and the promotion of the understanding of professional competences in software engineering. The goal is to use the “Competence Atlas” as a study-accompanying instrument in software engineering education. It is intended to show students the full range of required competences for their field of study at an early stage and to establish a connection to the practical work as a software engineer. Learning contents should thus be linked with existing knowledge structures directly from the beginning of the study, which can lead to an increase in learning motivation and can help orient oneself in the individual learning process in the context of the study.

The “Competence Atlas” can thus make a valuable contribution to the reflection and further development of students’ competences and support competence-oriented teaching in software engineering.

ACKNOWLEDGEMENT

This work is part of the EVELIN project and funded by the German Ministry of Education and Research (Bundesministerium für Bildung und Forschung) under grants 01PL12022A and 01PL17022A.

REFERENCES