Modelling Competency in the Field of OOP: From Investigating Computer Science Curricula to Developing Test Items

Matthias Kramer[1(✉)], David Tobinski[2], and Torsten Brinda[1]

[1] Computing Education Research Group, University of Duisburg-Essen, Essen, Germany
{matthias.kramer,torsten.brinda}@uni-due.de
[2] Cognitive and Educational Psychology, University of Duisburg-Essen, Essen, Germany
david.tobinski@uni-due.de

Abstract. In this paper, we describe the results of a thorough analysis of 44 K12 computer science curricula and standards documents conducted as part of an ongoing research project aiming at the development of a competency structure model and measurement instruments in the field of object-oriented programming (OOP). The curricula analysis builds upon a first model draft derived theoretically from a literature analysis in prior work. The model draft is 4-dimensional and consists of the four competency dimensions (1) OOP knowledge and skills, (2) Mastering representation, (3) Cognitive processes and (4) Metacognitive processes. We used these dimensions and the belonging sub-dimensions as a coding scheme and coded competency facets concerning OOP contained in the curricula and standards documents using the method of qualitative content analysis according to Mayring. This way, we could firstly successfully prove the curricular validity of our model draft and secondly, after a step of paraphrasing the identified competency facets, use these descriptions to initiate the process of item development to operationalize our competency model draft.

Keywords: Competency modelling · Item development · Competency measurement · Object-oriented programming · K12 education

1 Introduction

One of the current trends in educational research is the defining and measuring of learning outcomes in terms of competencies [1, 2]. Besides various competency modelling approaches in different disciplines in Germany (e.g. [3–5]), on an international level a lot of research has been performed in the context of the Programme for International Student Assessment (PISA) by the Organisation for Economic Co-operation and Development (OECD) which – since the year 2000 – focusses every three years on competencies in the areas of mathematics, science and language. Formerly defined post-hoc (e.g. [6]), the assessed competencies now rely partly on empirically founded competency models [7]. The results of these regular assessments are used to compare the educational levels of different countries or federal states. An expertise which analysed the

T. Brinda et al. (Eds.): SaITE 2016, IFIP AICT 493, pp. 37–46, 2016.
DOI: 10.1007/978-3-319-54687-2_4

"shocking" results (at least from the German perspective) of the first PISA studies under consideration of the different national educational systems came to the key conclusion that those countries performed best which oriented their educational systems at the intended learning outcomes [8] (and not at detailed curricula as before) and recommended (besides other aspects) the development of empirically founded competency models for all educational fields.

Concerning computer science education (CSE), the demand for competency models is still up to date. Although different normative models have been developed as educational standards in the past years (e.g. [9, 10]), empirically founded competency models in the CSE field are still rare (e.g. [4]). So, we have started a project to develop such a competency structure model and appropriate measuring instruments for the field of object-oriented programming (OOP), which is widely included in many K12 computer science curricula worldwide. The long-term goal of this research is to develop instruments to compare the learning outcomes in the CSE field nationally and internationally between different student populations.

2 Background and Related Work

2.1 On the Concept of Competency

In the context of competency modelling and assessment the term competency is often used according to the definition of Weinert. In his expertise for the OECD [11] he defined competencies 2001 as "the cognitive abilities and skills possessed by or able to be learned by individuals that enable them to solve particular problems, as well as the motivational, volitional and social readiness and capacity to use the solutions successfully and responsibly in variable situations". Despite the awareness that metacognitive factors can have major impact on learning results, these aspects of competencies are quite difficult to assess. Hence, Klieme et al. used a more economic definition in a priority programme of the German research foundation on competency modelling by defining competencies as "context-specific, cognitive dispositions of achievement which are functionally related to situations and requirements of certain domains in the sense of specific learning and action fields" [12]. In this context, competency models are used to describe the cognitive dispositions that learning individuals need to solve tasks and problems in a specific content or requirement area. Klieme et al. distinguish between modelling the structure, the level and the development of competencies in corresponding competency models [1]. Schecker and Parchmann [13] distinguish between normative (which describe the behaviour learners should show after a specific period of education, usually found in educational standards) and descriptive competency structure models (which describe the cognitive processes that occur in specific learning situations).

2.2 The COMMOOP Project

Our project COMMOOP aims to develop a normative competency structure model [1, 13] and appropriate measurement instruments for beginners, such as secondary school or undergraduate students, in the field of object-oriented programming (OOP).

We started by reviewing existing literature on competency modelling in other subject areas (such as mathematics, science, humanities, arts, computer science and problem solving) regarding the development methodology as well as the model structures (for a detailed description of this process, see [14]). We identified the areas of knowledge structures, content representation, cognitive processes as well as metacognitive influences as common, recurring structural elements, and verified, expanded and refined these based on an extensive literature analysis on theoretical and empirical studies on teaching and learning as well as on psychological aspects in the field of OOP. As theoretically derived candidates for potential competency dimensions we identified:

1. *OOP knowledge and skills*
   1.1. *Data structure (graph, tree, array)*
   1.2. *Class & object structure (object, attribute, association)*
   1.3. *Algorithmic structure (loops, conditional statement)*
   1.4. *Notional machine (data, working memory, processor, statement, program, automaton)*
2. *Mastering representation (language, syntax, semantics)*
3. *Cognitive process*
   3.1. *Problem solving stage (understanding the problem, determine how to solve the problem, translating the problem into a computer language program, testing and debugging the program)*
   3.2. *Cognitive process type*
      3.2.1. *Interpreting (Remember, Understand, Analyze, Evaluate)*
      3.2.2. *Producing (Apply, Create)*
4. *Metacognitive processes*

The proposed dimensions span a multidimensional competency space. Each competency candidate that is contained therein covers a certain subspace that is limited by fixed values on some dimensions that can be assigned to this competency. For example, the competency candidate "*be able to program arrays in Java*" might cover a subspace that is limited by the following fixed values: 1.1 Array, 2 Java, 3.1 Translating the solution in a computer program and 3.2. Understand and Apply. The remaining dimensions might be covered in total by this competency. It has to be noted that, due to national regulations of Java being one of the predominant languages in schools, our research will first focus on this language.

The theoretically derived model draft was validated based on various competency descriptions in terms of applicability and completeness. For this purpose, we identified competency descriptions related to OOP in 44 computer science curricula and standards from several countries and compared these with our model (see Sect. 3). In 2015, an international working group at the ACM ITiCSE conference analysed 14 case studies on K12 computer science education in 12 different countries, which resulted in 247 extracted competency descriptions. Out of these, 119 were related to programming and could be fully integrated in our model. Finally, the structure model was aligned with the results of a survey among 59 computer science teachers and teacher students on learning difficulties. Their views on which skills and abilities someone needs, to become a

competent programmer as well as their point of view on typical problems during that learning process could be also fully integrated in our model.

During this process, we had to adjust our theoretically derived model only in minor subsections (see Sect. 3).

2.3 Learning Metrics, Psychometric Models and Item Development

The discussion around coding to be the new literacy is vivid and the way to a proficiency scale in computer literacy is in its first stages. In PISA the development of scales followed six steps: (1) Identifying possible scales, (2) Assigning items to scales, (3) Skills audit, (4) Analysing field trial data, (5) Defining the dimensions and (6) Revising and refining with main survey data.

In a first step, we identified OOP as an important subscale [14]. Currently we're dealing with the construction of items as well as bringing them on their way to first field trials. For this stage of development, a more functional concept of competency is followed. This perspective is more interested in abilities to cope with challenges in specific situations rather than in the generative and cognitive systems behind [15]. From this background items have a single achievement threshold; the derived information only distinguishes dichotomously between correct and incorrect answers. Thus, the Rasch model can be applied to the raw data, in PISA a generalized form of the Rasch model has been used [16].

The Rasch model is "the only model to date that provides the tools for approximating objective reproducible additive measures in the human sciences." [17, p. 8]. It follows the concept that data conforms a reasonable hierarchy on a single continuum of interest. The most important fact is the measuring of a single variable with a map of persons (ξ_v) and items (σ_i) on the same scale, therefore the relationship between the participant and the item is probabilistic. The resulting model can be regarded as a model of competency levels. While those levels have already been regarded in the step of item construction, they will be precised by the step of skills audit and the analysis of the field trial data. At this stage numerical scores and terms of content, which describe what persons who achieve a specific level know and can do, can be reported. The final assessment of COMMOOP will be a computerized adaptive test (CAT), wherefore the Rasch model is essential [18].

Generative and structure models are more interested in the components behind the specific ability in a concrete item [1]. Modelling latent variables, like working memory or intelligence, comes into the focus, which is mostly driven by cognitivism. These models are rather the result of small experimental designs than of large-scale assessments. Modelling the cognition behind competencies is nowadays a challenge. One of its fruits will be the improvement of intelligent tutorial systems (ITS) and the rise of cognitive tutors. A pioneer of this type of modelling is John R. Anderson and his ACT-R Theory [19, 20]. Anderson and his colleagues have realized different ITS considering the cognitive limitations of the learner, one specific ITS is about learning to code in LISP. At a later stage COMMOOP will regard cognitive models for code trainings. The prior testing will combine the most promising items with different working memory, intelligence and planning scales.

3 Analysis of K12 Computer Science Curricula and Standards

A key step in our model development process (see Sect. 2.2) was a thorough analysis of K12 computer science curricula and educational standards. Altogether, we analysed 44 documents (German and international ones) and extracted competency facet descriptions concerning OOP from them. We performed this step because of two main reasons: first, we wanted to check the applicability of our theoretically derived normative competency model draft by trying to assign the competency facet descriptions to the proposed model categories to check and to ensure "curricular validity". This was an important step also in other competency modelling projects (cf. e.g. [3, 4]). Second, we wanted to prepare the operationalization of our model draft by starting to develop test items suitable to measure the competency facets found in the curricula.

Since we plan to measure OOP competency first in German schools, we included all publically available computer science curricula of Germany's 16 federal states as well as national standards documents of the Board of Federal Ministers of Education and Cultural Affairs and the German Informatics Association in the analysis. If a federal state had different curricula e.g. for lower and upper secondary computer science education, we included both documents in the analysis. To ensure the applicability of our model also in the international context, we also investigated important documents published in English language by other nations, such as the "Computing at school" curriculum (UK), as well as recommendations of international organizations, such as the ACM/IEEE curriculum and the CSTA K12 standards. We included documents from secondary and higher education, because we aim to model and to measure competencies of programming novices regardless of the educational level.

Curricula and standards documents are usually highly structured documents, in which intended learning outcomes are ordered e.g. by age level or by fields of the discipline (such as "algorithms" or "impact on society"). We used scientific definitions of the term "programming" taken from the literature (e.g. [21–23]), extracted possible search terms from them, such as "writing" and "implementing", and searched each of the documents for the sections, in which relevant OOP competency facets were located. Within these sections we performed a qualitative content analysis (QCA) according to Mayring [24] using the tool MAXQDA. We used the dimensions and subdimensions of our theoretically derived normative competency model draft as our category system and coded each OOP competency facet found in the documents with its belonging category. The QCA was performed deductively and inductively (cf. [25]), so the theoretically derived category system was extended inductively whenever a new category candidate was found in the documents.

Since we cannot present the whole coding process and its results in detail here, we give two examples. The text fragment "*By the end of this course, students will […] use constants and variables, including integers, floating points, strings, and Boolean values, correctly in computer programs*" (taken from the Ontario curriculum for Computer Studies in grades 10 to 12, Canada) was coded with the category *OOP knowledge & skills*, respectively with the subcategory *data structure*. The expected outcome that students should "*know the basics of object-oriented programming, e.g. inheritance,*

polymorphism and encapsulation" (taken from the computer science curriculum of Saxony, Germany) was assigned to the subdimension *class and object structure*.

Overall, our model draft proved to be well applicable for the classification of most of the OOP competency facets found in the documents, which contributes to the curricular validity of our approach. Still there are three places, in which we modified it during the coding process: First, we found competency descriptions relating to the ability of being able to evaluate a different solution approach for a problem, so we added a subdimension "*Evaluating different strategies*" in the proposed competency dimension *Problem solving stage*. Second, it became obvious that syntax and semantics are almost inseparable and that third, there was a need to code competency facets relating to "*Documenting and maintenance*" (both: top-level dimension *Mastering representation*) to cover text fragments relating to the use of comments in the code as well as to writing clear and maintainable code. Especially because of the inseparability of syntax and semantics we decided to combine the subcategories belonging to *Mastering representation* in the top-level category and to analyse its inner structure in future work. Table 1 shows the distribution of codings at the end of the coding process.

Table 1. Coding results of the curricula analysis.

Competency dimensions	Subdimension	Number of codings
OOP knowledge & skills	Data structure	104
	Class & object structure	74
	Algorithmic structure	164
	Notional machine	15
Mastering representation	Semantics	36
	Syntax	43
	Programming language	37
	Documenting & maintenance	27
Cognitive processes	Understanding the problem	67
	Determining a plan	103
	Translating the problem solution into a program	116
	Testing and debugging	108
	Evaluating different strategies	11
Metacognitive processes		4

The next important step of the QCA according to Mayring [24] is the paraphrasing and summarizing of specific categories. The aim of this step is to combine similar competency facet formulations in fewer distinct joint formulations. Table 2 shows the result of this process within the dimension "*OOP knowledge and skills*".

It must be noted here that composite competency facet formulations like "*Students are able to assign values to variables of primitive and compound data types as well as more complex data structures*" were coded with more than one category.

Table 2. Categories formed in the subdimensions of the dimension OOP knowledge & skills by paraphrasing and summarizing the competency facet formulations

Subdimensions	Categories	Number of codings
Data structure	Data representation/structure in general	52
	Primitive data types	29
	Composite data types	36
	Dynamic data structures	27
Class & object structure	OO in general	20
	Fundamental OO-concepts (class, object, attributes, methods)	48
	Higher OO-concepts (polymorphism, encapsulation, inheritance)	19
	Using predefined structures	14
Algorithmic structure	Algorithms in general	45
	Fundamental algorithmic structures (sequence, selection, repetition)	91
	Subprograms (methods, procedures, functions)	21
	Recursion	24
	Searching/Sorting	22
	Algorithmic optimization	19
Notional machine		15

It is not surprising, that fundamental algorithmic structures can be found most often in the documents, since neither does every country or state follow an object-oriented paradigm nor, in case they do, is the objects-first approach the most dominant one.

4 Developing Test Items

The next step on the way to an empirically validated competency structure model for OOP (see Sect. 2.2) is the operationalization of the competency facets of our theoretically derived model by describing specific situations that require a certain behaviour that persons show who are competent in this area.

While competency formulations describe a subspace of skills and abilities limited by certain parameters (see Sect. 2.2), the items should only test for a distinct competency dimension in the model to prove that the assumed multidimensionality of the model can be shown empirically. Moreover, the items have to be constructed in a way that solving them correctly will indicate the underlying competency. Hence, many test items of varying difficulty are required. Furthermore, it is necessary to take into account the probands' previous knowledge to ensure the validity of the results. In Germany, students at the end of grade 12 should have the necessary competencies when they have gone through upper secondary computer science education or when they completed an "Introduction to OOP" course at university level.

As a result of the curricula analysis we have collected numerous formulations describing OOP competency facets and abstracted those in the paraphrasing step (see Sect. 3). These descriptions are now used to construct test items, respectively to gain access to programming competencies and hence to validate the assumed theoretical competency structure on an empirical level. In the following, we give two examples of test items to be used at the end of an introduction to OOP. Item 1 is a considerably easy item addressing the understanding of classes as templates, objects as instances, attributes as characteristics and methods as modifiers of attribute values:

Item 1: *You are given the following source code:*

```
class House{
 Triangle roof;
 Square apartment;
 public House(){
  roof = new Triangle();
  apartment = new Square();
  roof.setLength(12);
 }
 public void adjust(){
  apartment.setLength() = roof.getLength();
 }
}
```

Mark all appearing class identifiers in red.

Item 1 can also be extended by various other tasks, such as "*If objects are created, please write down the respective line number*" or "*Please write down all the attributes and methods a House-object can access*".

Item 2: *You are given the following source code:*

```
public class A {
 void doThis(){
  System.out.println("A");
 }
}

public class B extends A {
 void doThis(){
  System.out.println("B");
 }
}

public class C extends B {

}
```

```
public class Z{
 public static void main
 (String[] args){
  A[] z = new A[7];
  z[0] = new A();
  z[1] = new B();
  z[2] = new C();
  z[3] = new B();
  z[4] = new A();
  z[5] = new C();
  for(A e : z){
   e.doThis();
  }
 }
}
```

Predict the output, which is generated by the main method of class Z.

To raise complexity, it is reasonable to choose more advanced concepts, to differ in the respective task, to use more abstraction, etc. Item 2 is a more complex task addressing the understanding of inheritance, polymorphism and encapsulation. The correct solution of item 2 can indicate a correct understanding of inheritance along with a corresponding

concept for polymorphism. As well as item 1 item 2 can be varied, e.g. by adding one *doThis* method in class C and asking to predict the correct output, by writing a new subclass *D*, using *super* to refer to attributes and methods in super classes and so on.

The items will finally be evaluated by means of Item Response Theory. Hartig and Frey [26] already showed that for a multidimensional competency modelling this is a valid approach.

5 Summary and Outlook

In this paper, we gave an overview of the current state of our project COMMOOP, in which we aim to develop an empirically founded competency structure model for the field of object-oriented programming. We focussed on a curricula analysis, which we conducted to identify competency descriptions concerning OOP. We used these descriptions in a first step to check the applicability of our theoretically derived competency model draft for the classification of these descriptions. Second, we used the competency descriptions assigned to the subdimension "OOP knowledge and skills" to start the operationalization of our model by developing first test items to measure the proposed competencies. The process of item development is continuing under consideration of aspects, such as item difficulty and variety, suitability for adaptive testing and test length restrictions. Several iterations of pretests with the items will be conducted next on the way to develop a test instrument to validate the proposed competency structure.

References

1. Klieme, E., Hartig, J., Rauch, D.: The concept of competence in educational contexts. In: Hartig, J., Klieme, E., Leutner, D. (eds.) Assessment of Competencies in Educational Contexts. Hogrefe & Huber Publishers, Toronto (2008)
2. Hartig, J., Klieme, E., Leutner, D. (eds.): Assessment of Competencies in Educational Contexts. Hogrefe & Huber Publishers, Toronto (2008)
3. Jordan, A.-K., Knigge, J.: The development of competency models: an IRT-based approach to competency assessment in general music education. In: Brophy, T.S. (ed.) The Practice of Assessment in Music Education: Frameworks, Models and Designs, pp. 67–86. GIA Publications, Chicago (2010)
4. Linck, B., Ohrndorf, L., Schubert, S., Stechert, P., Magenheim, J., Nelles, W., Neugebauer, J., Schaper, N.: Competence model for informatics modelling and system comprehension. In: Proceedings of the Global Engineering Education Conference 2013, pp. 85–93. IEEE (2013)
5. Schecker, H., Parchmann, I.: Modellierung naturwissenschaftlicher Kompetenz. Zeitschrift für Didaktik der Naturwissenschaften **12**, 45–66 (2006)
6. Neumann, K., Kauertz, A., Lau, A., Notarp, H., Fischer, H.E.: Die Modellierung physikalischer Kompetenz und ihrer Entwicklung. Zeitschrift für Didaktik der Naturwissenschaften. **13**, 101–121 (2007)
7. Neumann, K., Fischer, H.E., Kauertz, A.: From PISA to educational standards: the impact of large-scale assessments on science education in Germany. Int. J. Sci. Math. Educ. **8**(3), 545–563 (2010)

8. Klieme, E., Avenarius, H., Blum, W., Döbrich, P., Gruber, H., Prenzel, M., Reiss, K., Riquarts, K., Rost, J., Vollmer, H.J.: The Development of National Educational Standards. An expertise. Federal Ministry of Education and Research, Berlin (2003)
9. Brinda, T., Puhlmann, H., Schulte, C.: Bridging ICT and CS: educational standards for computer science in lower secondary education. ACM SIGCSE Bull. **41**(3), 288–292 (2009)
10. Seehorn, D., Carey, S., Fuschetto, B., Lee, I., Moix, D., O'Grady-Cunniff, D., Boucher Owens, B., Stephenson, C., Verno, A.: CSTA K-12 Computer Science Standards: Revised 2011. ACM Technical report, New York (2011)
11. Weinert, F.E.: Concept of competence: A conceptual clarification. In: Rychen, D.S., Salganik, L. (eds.) Defining and Selecting Key Competencies. Hogrefe & Huber, Seattle (2001)
12. Klieme, E., Maag-Merki, K., Hartig, J.: Kompetenzbegriff und Bedeutung von Kompetenzen im Bildungswesen. In: Hartig, J., Klieme, E. (eds.) Möglichkeiten und Voraussetzungen technologiebasierter Kompetenzdiagnostik, Bonn (2007)
13. Schecker, H., Parchmann, I.: Modellierung naturwissenschaftlicher Kompetenz. Zeitschrift für Didaktik der Naturwissenschaften **12**, 45–66 (2006)
14. Kramer, M., Hubwieser, P., Brinda, T.: A competency structure model of object-oriented programming. In: Proceedings of the Fourth International Conference on Learning and Teaching in Computing and Engineering (LaTiCE), pp. 1–8. IEEE (2016). (in press)
15. McClelland, D.C.: Testing for competence rather than for intelligence. Am. Psychol. **28**, 1–14 (1973)
16. OECD: PISA 2012 Technical report, PISA. OECD, Paris (2014)
17. Bond, T., Fox, C.: Applying the Rasch Model. Lawrence Erlbaum Ass., London (2001)
18. Frey, A., Kroehne, U., Seitz, N.-N., Born, S.: Multidimensional adaptive measurement of competences. In: Leutner, D., Fleischer, J., Grünkorn, J., Klieme, E. (eds.) Competence Assessment in Education: Research, Models and Instruments. Springer, Heidelberg (in press)
19. Anderson, J.R., Bothell, D., Byrne, M.D., Douglass, S., Lebiere, C., Qin, Y.: An integrated theory of the mind. Psychol. Rev. **111**(4), 1036–1060 (2004)
20. Anderson, J.R., Corbett, A.T., Koedinger, K., Pelletier, R.: Cognitive tutors: lessons learned. J. Learn. Sci. **4**, 167–207 (1995)
21. Blackwell, A.F.: What is programming? In: Kuljis, J., Baldwin, L., Scobl, R. (eds). Proceedings of the 14th PPIG, pp. 204–218 (2002). http://www.ppig.org
22. Böszörményi, L., Weich, C.: What is programming? In: Böszörményi, L., Weich, C. (eds.) Programming in Modula-3. Springer, Heidelberg (1996)
23. Williamson, I., Dale, R.: What is programming? In: Williamson, I., Dale, R. (eds.) Understanding Microprocessors with the Science of Cambridge Mk14. Macmillan Education UK, pp. 13–17 (1980)
24. Mayring, P.: Qualitative content analysis–research instrument or mode of interpretation. Role Res. Qual. Psychol. **2**, 139–148 (2002)
25. Bortz, J., Döring, N.: Forschungsmethoden und Evaluation für Human- und Sozialwissenschaftler. Springer, Heidelberg (2006)
26. Hartig, J., Frey, A.: Sind Modelle der Item-Response-Theorie (IRT) das "Mittel der Wahl" für die Modellierung von Kompetenzen? Zeitschrift für Erziehungswissenschaft, 16 (Sonderheft 18-2013), pp. 47–51 (2013)