



UNIVERSITÄT
DUISBURG
ESSEN

Open-Minded

Competency Structure Model for Programming for the Transition from School to University

WiPSCE 2020

Mike Barkmin ■ 28. October 2020

Overview

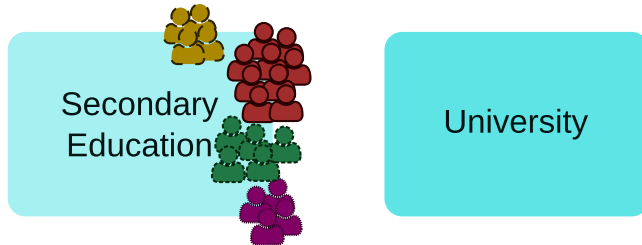
1. Motivation
2. Competencies
3. Related Research
4. COMM_P Model
5. Outlook






Figure: Image by fancycrave1 under Pixabay License via Pixabay

Motivation

Motivation



Previous experience

 None  Private  CS class  Private + CS class

Mathematics

Psychology

Biology

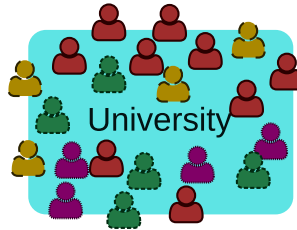
Physics

Computer Science

...

Motivation

Secondary
Education



Mathematics

Psychology

Biology

Physics

Computer Science

...

Previous experience



None



Private



CS class



Private + CS class

Motivation

Secondary
Education

University

Previous experience



None



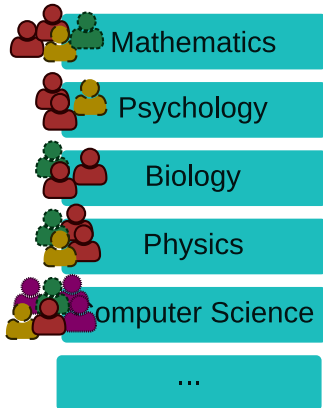
Private



CS class



Private + CS class

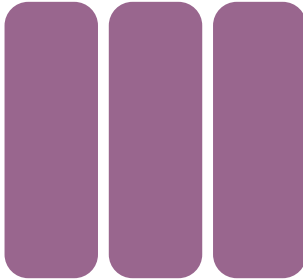


Competencies

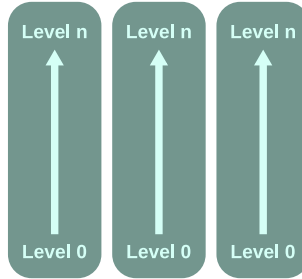
[Competencies are] context-specific cognitive dispositions that are acquired and needed to successfully cope with certain situations or tasks in specific domains. (Koeppen et al., 2008)

Competency Models

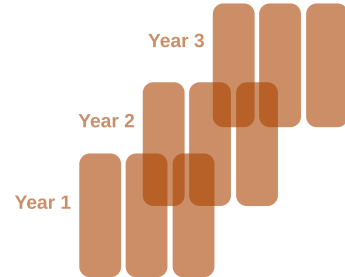
Competency Structure Model



Competency Level Model



Competency Development Model



(Klieme, 2004)

Related Research

Related Research

- In CSE, the development process of competency models is just beginning (Hubwieser and Sentance, 2018)



Figure: Books by LubosHouska under Pixabay License
via Pixabay

Related Research

- In CSE, the development process of competency models is just beginning (Hubwieser and Sentance, 2018)
- Specialized on Programming



Figure: Books by LubosHouska under Pixabay License
via Pixabay

Related Research

- In CSE, the development process of competency models is just beginning (Hubwieser and Sentance, 2018)
- Specialized on Programming
 - Competency Modeling and Measurement for Object-Oriented Programming (COMMOOP) (Kramer, Hubwieser, et al., 2016)
 - Object Interaction Competence Model (Bennedsen and Schulte, 2013)
 - Competence Model for the Novice Programmer (Kiesler, 2020)



Figure: Books by LubosHouska under Pixabay License
via Pixabay

Related Research

- In CSE, the development process of competency models is just beginning (Hubwieser and Sentance, 2018)
- Specialized on Programming
 - Competency Modeling and Measurement for Object-Oriented Programming (COMMOOP) (Kramer, Hubwieser, et al., 2016)
 - Object Interaction Competence Model (Bennedsen and Schulte, 2013)
 - Competence Model for the Novice Programmer (Kiesler, 2020)
- None of the present competency models is suitable for our research goal



Figure: Books by LubosHouska under Pixabay License
via Pixabay

COMM_P Model

- COMMOOP model as base, because it got evaluated in empirical studies (e.g. Kramer, Barkmin, et al., 2019; Kramer, Tobinski, et al., 2016)
- Problem: Specific for object-oriented programming
- Analyze literature in the field of programming languages and paradigms to expand the model
 - e.g.: Ambler et al. (1992), Armstrong (2006), Gabbrielli and Martini (2010), Horowitz (1984), Janeček and Pergl (2017), Sebesta (2016), and Wirth (2000)

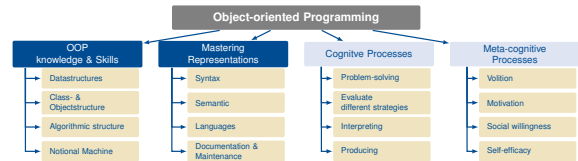


Figure: COMMOOP Model (Kramer, Hubwieser, et al., 2016)

Programming

Knowledge

High-Level Paradigm

Elements

Language

Cognitive Process

Interpreting

Creating

Programming

Knowledge

High-Level Paradigm

Elements

Language

Cognitive Process

Interpreting

Creating

Competencies = **Knowledge** + **Cognitive Process**

can only be observed in context-specific tasks

Programming

Knowledge

High-Level Paradigm

Elements

Language

Cognitive Process

Interpreting

Producing

- **Low-Level Paradigms:** for example, copying versus sharing data structures
- **Algorithmic Paradigms:** for example, divide and conquer and dynamic programming
- **High-Level Paradigms:** for example, object-oriented and functional

(Floyd, 1979)

Object-Oriented (Armstrong, 2006)

- Structure
 - Abstraction
 - Class
 - Encapsulation
 - Inheritance
 - Object
- Behavior
 - Message Passing
 - Method

Functional (Janeček and Pergl, 2017)

- First-class functions
- Referential transparency
- Immutability of variables and values
- Closure
- Recursion

Programming

Knowledge

High-Level Paradigm

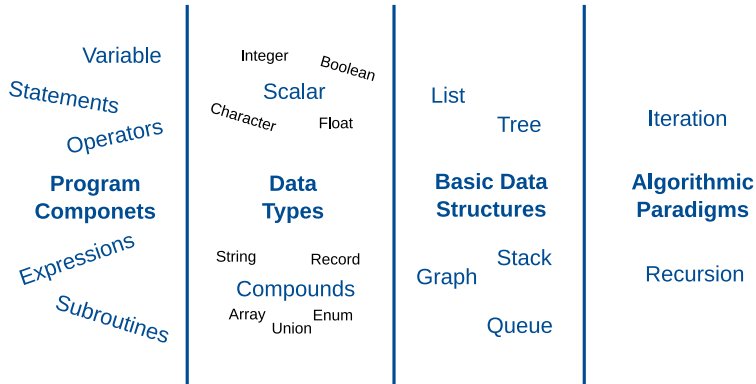
Elements

Language

Cognitive Process

Interpreting

Producing



Object-Oriented	Generalized
Object	Record, accessed via pointer
Class	Record type
Method	Subroutine, bound to a record
Message	Call of bound subroutine
Subclass	Record type extension

Table: (adapted from Wirth, 2000, p. 6)

- **Primitive:** A data type whose values is stored directly in memory (Sebesta, 2016)
- **Reference:** A data type whose value is an address in memory (Sebesta, 2016)
- **Scalar:** A data type which only consists of a single value (Horowitz, 1984)
- **Compound:** A data type which is composed of a set of types (Horowitz, 1984)

Object-Oriented	Generalized
Object	Record, accessed via pointer
Class	Record type
Method	Subroutine, bound to a record
Message	Call of bound subroutine
Subclass	Record type extension

Table: (adapted from Wirth, 2000, p. 6)

- **Primitive:** A data type whose values is stored directly in memory (Sebesta, 2016) Implementation
- **Reference:** A data type whose value is an address in memory (Sebesta, 2016)
- **Scalar:** A data type which only consists of a single value (Horowitz, 1984) Concept
- **Compound:** A data type which is composed of a set of types (Horowitz, 1984)

Programming

Knowledge

High-Level Paradigm

Elements

Language

Cognitive Process

Interpreting

Producing

Syntax | Semantic | Standard Library | Build and Run

Competencies = **Knowledge** + **Cognitive Process**

can only be observed in context-specific tasks

Programming

Knowledge

High-Level Paradigm

Elements

Language

Cognitive Process

Interpreting

Producing

Competencies = **Knowledge** + **Cognitive Process**

can only be observed in context-specific tasks

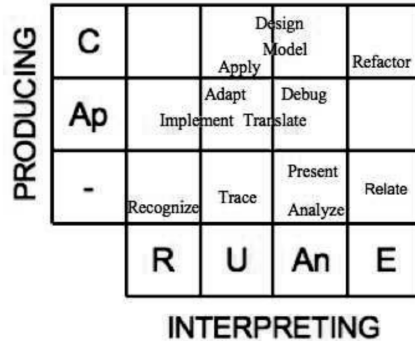


Figure 7. Mapping programming activities to the Matrix

(Fuller et al., 2007)

Application

Please determine for every statement below whether it is correct. (adapted from Kramer, Barkmin, et al., 2019)

- A class is a building plan for objects.
- For every class there needs to exist at least one object.
- ... +3

high-level paradigm → principles

interpreting → remember

Which problem will only need one if-statement in its solution if conditional operators are not allowed? (adapted from Parker et al., 2016)

- Returning different sounds for a pig and a cow.
- Printing a statement if a number is between 1 and 10 and even.
- ... +3

elements → *Program Components*

interpreting → *evaluate*

Which Java-keyword is missing that would make the file compile without errors?

```
■ public _____ Car {}
```

language → *Syntax*

producing → *apply*

Please determine for every statement below whether it is correct. (adapted from Kramer, Barkmin, et al., 2019)

- A class is a building plan for objects.
- For every class there needs to exist at least one object.
- ... +3

high-level paradigm → principles

interpreting → remember

Which problem will only need one if-statement in its solution if conditional operators are not allowed? (adapted from Parker et al., 2016)

- Returning different sounds for a pig and a cow.
- Printing a statement if a number is between 1 and 10 and even.
- ... +3

elements → Program Components

interpreting → evaluate

Which Java-keyword is missing that would make the file compile without errors?

■ `public _____ Car {}`

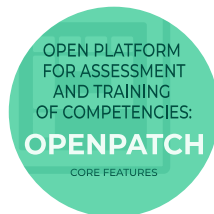
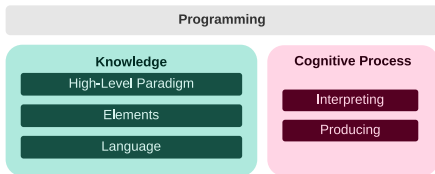
language → Syntax

producing → apply



Outlook

Outlook



Outlook

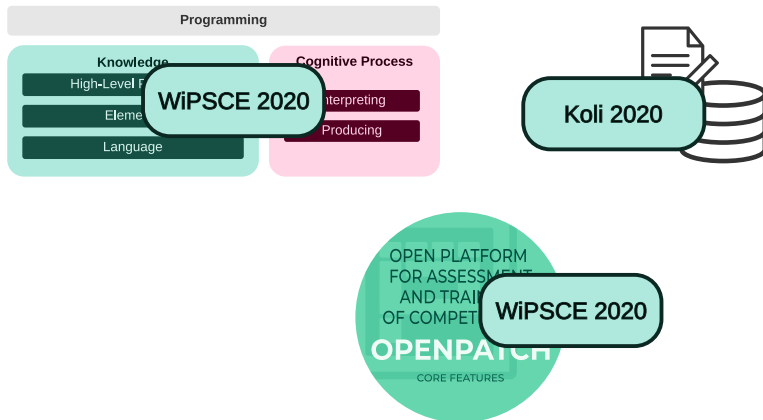


Figure: Publications (Barkmin, 2020a,b; Barkmin and Brinda, 2020)

28. October 2020

Thanks for Listening

Contact

Mike Barkmin
Didaktik der Informatik
Universität Duisburg-Essen
Schützenbahn 70, 45127 Essen
mike.barkmin@uni-due.de
<http://udue.de/mba>

References I

- Ambler, A. L. et al. (Sept. 1992). “Operational versus Definitional: A Perspective on Programming Paradigms”. In: *Computer* 25.9, pp. 28–43.
- Armstrong, Deborah J. (Feb. 2006). “The Quarks of Object-Oriented Development”. In: *Commun. ACM* 49.2, pp. 123–128.
- Barkmin, Mike (2020a). “An Open Platform for Assessment and Training of Competencies”. In: *Proceedings of the 15th Workshop on Primary and Secondary Computing Education. WiPSCE '20*. Virtual Event, Germany: Association for Computing Machinery. ISBN: 9781450387590.
- (2020b). “Competency Structure Model for Programming for the Transition from School to University”. In: *Proceedings of the 15th Workshop on Primary and Secondary Computing Education. WiPSCE '20*. Virtual Event, Germany: Association for Computing Machinery. ISBN: 9781450387590.

References II

- Barkmin, Mike and Torsten Brinda (2020). “Analysis of Programming Assessments — Building an Open Repository for Measuring Competencies”. In: *Koli Calling '20: Proceedings of the 20th Koli Calling International Conference on Computing Education Research*. Koli Calling '20. Koli, Finland: Association for Computing Machinery.
- Bennedsen, J. and C. Schulte (Mar. 2013). “Object Interaction Competence Model v. 2.0”. In: *2013 Learning and Teaching in Computing and Engineering*, pp. 9–16.
- Floyd, Robert W. (Aug. 1979). “The Paradigms of Programming”. In: *Commun. ACM* 22.8, pp. 455–460.
- Fuller, Ursula et al. (2007). “Developing a Computer Science-Specific Learning Taxonomy”. In: *Working Group Reports on ITiCSE on Innovation and Technology in Computer Science Education*. ITiCSE-WGR '07. New York, NY, USA: ACM, pp. 152–170.
- Gabbrielli, Maurizio and Simone Martini (2010). *Programming Languages: Principles and Paradigms*. en. Undergraduate Topics in Computer Science. London: Springer-Verlag. ISBN: 978-1-84882-913-8.

References III

- Horowitz, E. (1984). *Fundamentals of Programming Languages*. en. Second. Berlin Heidelberg: Springer-Verlag. ISBN: 978-3-642-96729-0.
- Hubwieser, Peter and Sue Sentance (2018). “Taxonomies and Competency Models”. In: *Computer Science Education*. First. London: Bloomsbury Academic, pp. 221–240.
- Janeček, Lukáš and Robert Pergl (2017). “Analysing Functional Paradigm Concepts”. en. In: *Recent Advances in Information Systems and Technologies*. Ed. by Álvaro Rocha et al. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, pp. 882–891. ISBN: 978-3-319-56535-4.
- Kiesler, Natalie (June 2020). “Towards a Competence Model for the Novice Programmer Using Bloom’s Revised Taxonomy - An Empirical Approach”. In: *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*. ITiCSE '20. Trondheim, Norway: International Foundation for Autonomous Agents and Multiagent Systems, pp. 459–465. ISBN: 978-1-4503-6874-2.

References IV

- Klieme, Eckhard (2004). *The Development of National Educational Standards : An Expertise*. BMBF : Education Reform.
- Koeppen, Karoline et al. (Jan. 2008). "Current Issues in Competence Modeling and Assessment". In: *Zeitschrift für Psychologie / Journal of Psychology* 216.2, pp. 61–73.
- Kramer, Matthias, Mike Barkmin, et al. (July 2019). "Identifying Predictors for Code Highlighting Skills: A Regression Analysis of Knowledge, Syntax Abilities and Highlighting Skills". en. In: *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*. Aberdeen Scotland Uk: ACM, pp. 367–373. ISBN: 978-1-4503-6895-7.
- Kramer, Matthias, Peter Hubwieser, et al. (Mar. 2016). "A Competency Structure Model of Object-Oriented Programming". In: *2016 International Conference on Learning and Teaching in Computing and Engineering (LaTICE)*. Mumbai, India: IEEE, pp. 1–8.

References V

- Kramer, Matthias, David Tobinski, et al. (2016). "On the Way to a Test Instrument for Object-Oriented Programming Competencies". In: *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*. Koli Calling '16. New York, NY, USA: ACM, pp. 145–149. ISBN: 978-1-4503-4770-9.
- Parker, Miranda C. et al. (2016). "Replication, Validation, and Use of a Language Independent CS1 Knowledge Assessment". In: *Proceedings of the 2016 ACM Conference on International Computing Education Research*. ICER '16. New York, NY, USA: ACM, pp. 93–101. ISBN: 978-1-4503-4449-4.
- Sebesta, Robert W. (2016). *Concepts of Programming Languages*. Englisch. Eleventh. Boston: Pearson Addison Wesley. ISBN: 978-0-13-139531-2.
- Wirth, Niklaus (2000). "The Development of Procedural Programming Languages Personal Contributions and Perspectives". en. In: *Modular Programming Languages*. Ed. by Wolfgang Weck and Jürg Gutknecht. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 1–10. ISBN: 978-3-540-44519-7.