Code Structure Difficulty in OOP

an exploration study regarding basic cognitive processes

Mike Barkmin, David Tobinski, Matthias Kramer and Torsten Brinda

UNIVERSITÄT DUISBURG ESSEN

Open-Minded

Background

- On the way to a competency model of OOP the most important code structures have been identified in former studies (see [7])

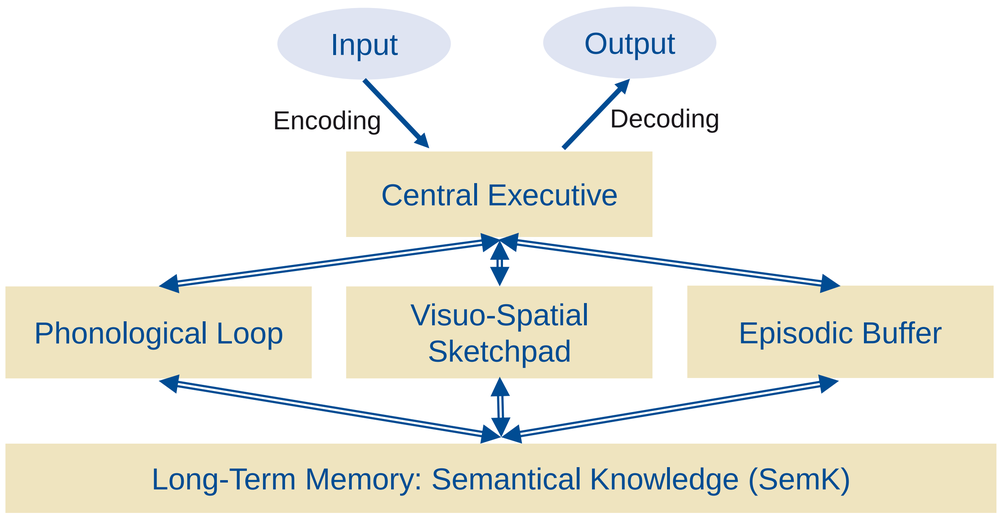
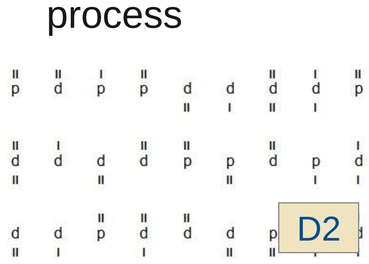


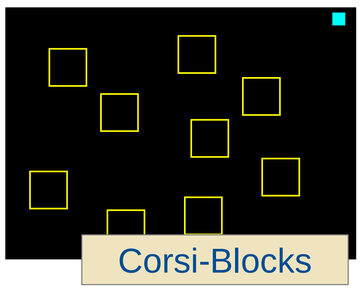
Fig: Working Memory Model with information processing (see [2, 9, 6])

- **RQ**: Is there a difference in encoding or decoding of varied code structures?

Method

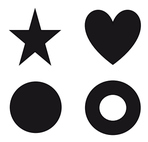
- Assessing cognitive capabilities to control their influence on the process





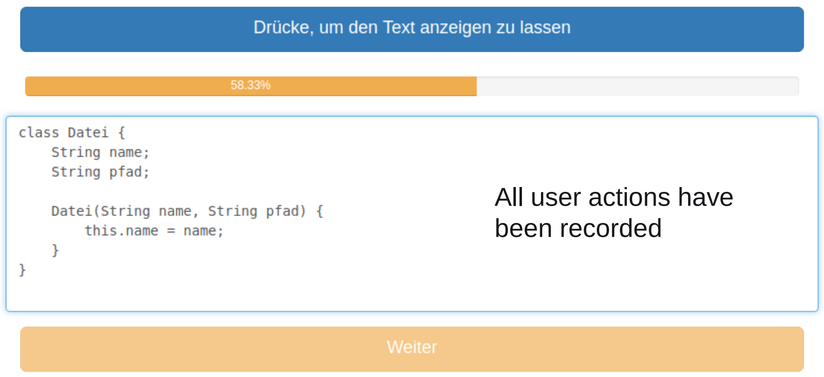
IST 2000-R verbal

Places: Koli, Essen
Buildings: Church, Stall
Fruits: Kiwi, Strawberry



IST 2000-R figural

- Assessing the ability to decode and encode programming language texts (PLT) and natural language texts (NLT)



  - Three levels with three items each were used
  - The items of each level were comparable in length and cognitive complexity metrices (see [1, 3, 5, 8])

- Assessing basic demographic information and the self-estimated programming experience
  - How do you estimate your programming experience?
  - How experienced are you with the programming language Java?
  - How experienced are you with object-oriented programming?
  - How experienced are you compared to an expert with 20 years of experience?
  - How experienced are you compared to a fellow student?

- We divided the computer science students into the SemK-levels novices and experts using the self-estimated programming experience

Results

- 42 students (5 were excluded due to missing data; 10 female; 27 male; mean of age: 25.03; sd: 3.77)
- The covered Levenshtein-Distance [10] during decoding and the time spent on encoding were used to determine differences between the control group (CG), novices (N) and experts (E)

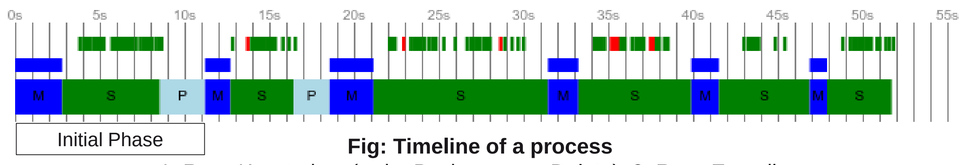


Fig: Timeline of a process
1. Row: Keystrokes (red = Backspace or Delete), 2. Row: Encoding
3. Row: Encoding (M), Decoding (S) und Pause (P)

- The initial phase is a good indicator for the whole process, because it is not affected by a learning effect
- We analysed the differences between the groups by conducting a Whitney-U-Test

		Novices (n = 16)			Experts (n = 15)		
		PLT-C	NLT	PLT-A	PLT-C	NLT	PLT-A
Control Group (n = 6)	IL1			LVD**	LVD*		LVD*
	IL2	E**LVD*	E*		E*D*LVD*	E*	E**D*LVD*
	IL3	LVD*			LVD*		LVD*

Tab: Differences in initial behaviour of the three groups
Note: * p < .05; ** p <.01; *** p < .001
PLT-C: Class Item, NLT: NLT Item, PLT-A: Algorithm Item, IL: Item-Level
LVD: Levenshtein-Distance, E: Encoding Time, D: Decoding Time

- Items of IL2 are best suitable for the measurement of differences between the groups

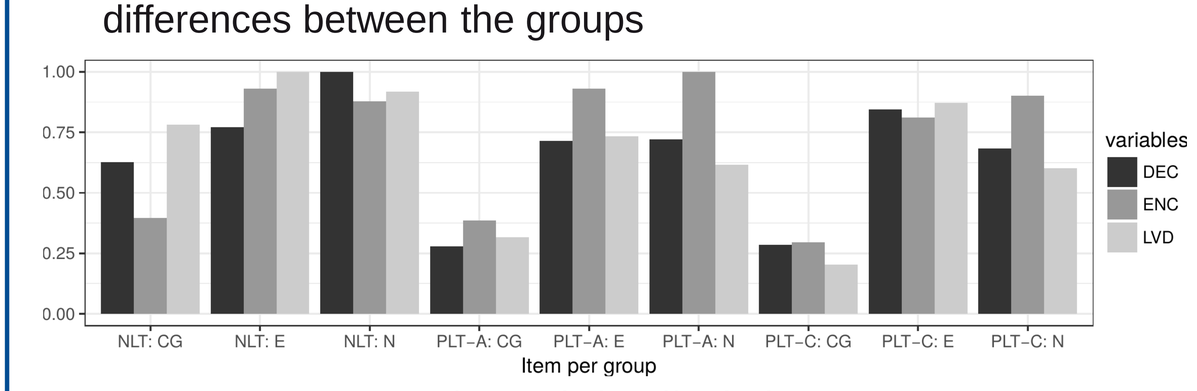


Fig: Normed values of item level 2

```
class Datei {
    String name;
    String pfad;

    Datei(String name, String pfad) {
        this.name = name;
        this.pfad = pfad;
    }

    void umbenennen(String name) {
        this.name = name;
    }
}
```

Meerschweinchen heißen „Schweinchen“, weil sie wie Schweine quieken. „Meer“ kommt daher, dass sie von Südamerika aus über den Atlantik nach Europa gebracht worden sind. In Europa sind sie beliebte Haustiere.

```
public String einruecken(String text, int anzahl, String symbol) {
    if(breite < 0) {
        return null;
    }
    for(int index = anzahl; index > 0; index--) {
        text = symbol + text;
    }
    return text;
}
```

Fig: Items of level 2

Discussion and Outlook

- There seems to be an overall positive effect of SemK on encoding and decoding processes of PLT
- The processing time needs a deeper analysis, because there might be an inference between encoding complex code and SemK
- A follow-up study with new items comparable to those of IL2 seems promising to unravel the interpreted underlying interaction of SemK and code structure complexity
- Further details of the procedure and analysis are in [4]

References

[1] T. Amstad. 1978. Wie verständlich sind unsere Zeitungen? Studenten-Schreib-Service.

[2] A. Baddeley. 2000. The episodic buffer: a new component of working memory? Trends in Cognitive Sciences 4, 11 (2000), 417–423.

[3] R. Bamberger. 1984. Lesen - verstehen - lernen - schreiben : die Schwierigkeitsstufen von Texten in deutscher Sprache. Jugend und Volk [u.a.], Wien.

[4] Mike Barkmin. 2017. Konstruktion und Erprobung eines Bausteins zur Kompetenzmessung im Bereich der objektorientierten Programmierung im den Dimensionen Syntax und Semantik. Unversität Duisburg-Essen.

[5] R. Flesch. 1946. The Art of Plain Talk.

[6] Walter Kintsch. 1974. The representation of meaning in memory. (1974).

[7] M. Kramer, D. A. Tobinski, and T. Brinda. 2016. Modelling Competency in the Field of OOP: From Investigating Computer Science Curricula to Developing Test Items. In Stakeholders and Information Technology in Education. 5.

[8] S. Misra and K. I. Akman. 2008. Weighted Class Complexity: A Measure of Complexity for Object Oriented System. Journal of Information Science and Engineering (2008).

[9] D. A. Tobinski. 2017. Gedächtnis: Informationsverarbeitung in der kognitiven Architektur. Springer Berlin Heidelberg, Berlin, Heidelberg, 23–43.

[10] R. A. Wagner and M. J. Fischer. 1974. The String-to-String Correction Problem. J. ACM 21, 1 (1 1974), 168–173.

CSE

Contact

Mike Barkmin, Matthias Kramer and Torsten Brinda
Didactics of Informatics
University of Duisburg-Essen
mike.barkmin@uni-due.de
matthias.kramer@uni-due.de
torsten.brinda@uni-due.de

David Tobinski
Institute of Psychology
University of Duisburg-Essen
david.tobinski@uni-due.de