FAIBLE.nrw

Didaktik der Programmierung

UNIVERSITÄT PADERBORN

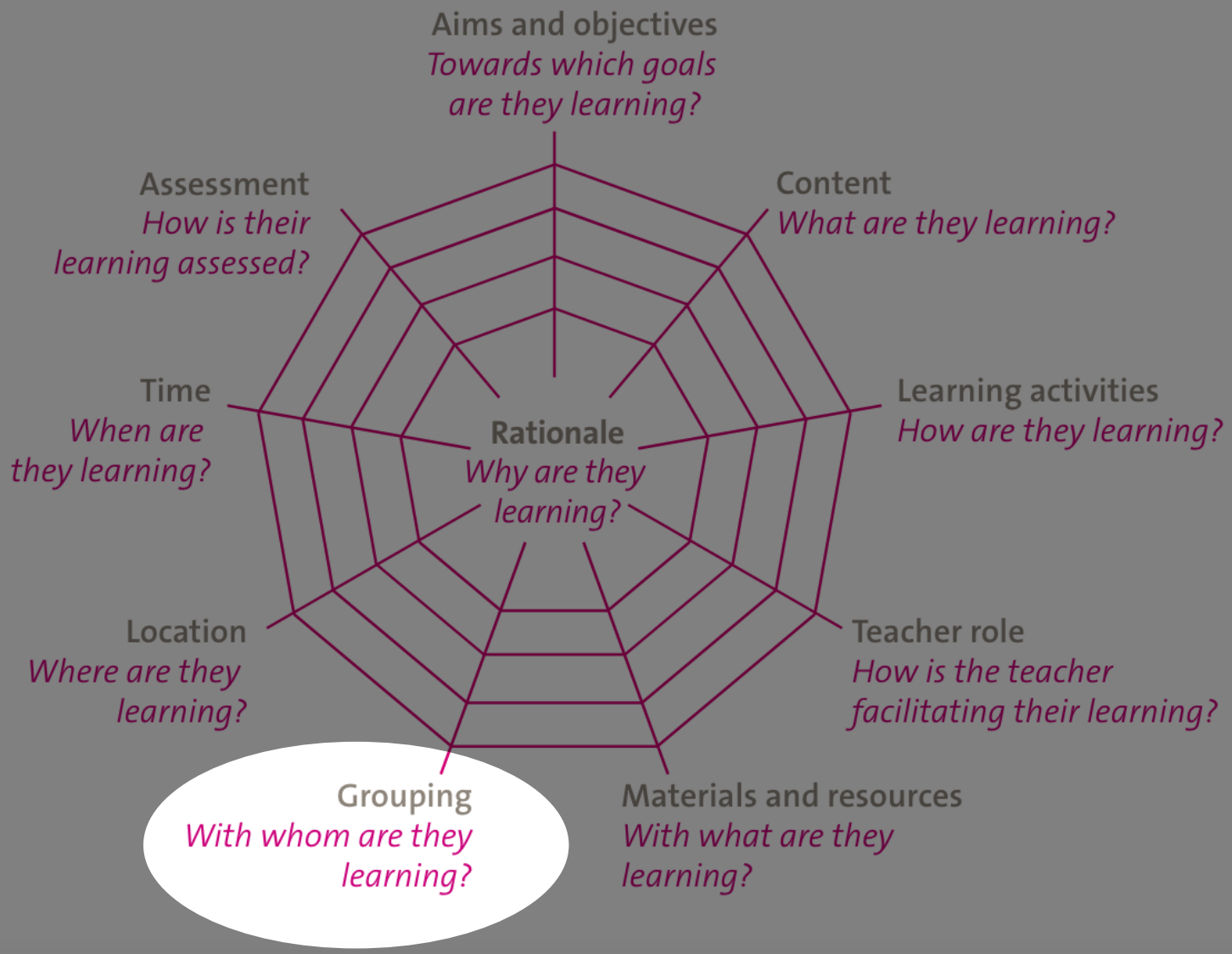

Notional Machine
Modellieren
Projektunterricht
Bildung
Pair Programming
PRIMM
Programmiersprachen
Software Life Cycle
4C/ID
Kompetenzen
Tools
Softwareentwicklung
Problemfelder
Lernroboter
Implementieren
Unterrichtsmethoden
Lernerfolg- und Leistungsbewertung
Worked Example
Inhalte
Lernziele
Programmverstehen
Digitale Welt
Debugging
Cognitive Load
STREAM
Block-basierte Sprachen
Computational Thinking
Programmieren
Use-Modify-Create
Programmierkonzepte
Computational Notebooks
Softwareprojekte
Lehrpläne

Kapitelübersicht

1. Einführung
2. Ziele und Inhalte
3. Lerntheorien
4. Unterrichtsplanung
5. Programmiersprachen und Umgebungen
6. **Programmieren im Team**
7. Lernerfolgskontrolle und Leistungsbewertung

6. Programmieren im Team

Pair Programming, Softwareprojekte


Aims and objectives
Towards which goals are they learning?
Content
What are they learning?
Learning activities
How are they learning?
Teacher role
How is the teacher facilitating their learning?
Materials and resources
With what are they learning?
Grouping
With whom are they learning?
Location
Where are they learning?
Time
When are they learning?
Assessment
How is their learning assessed?
Rationale
Why are they learning?

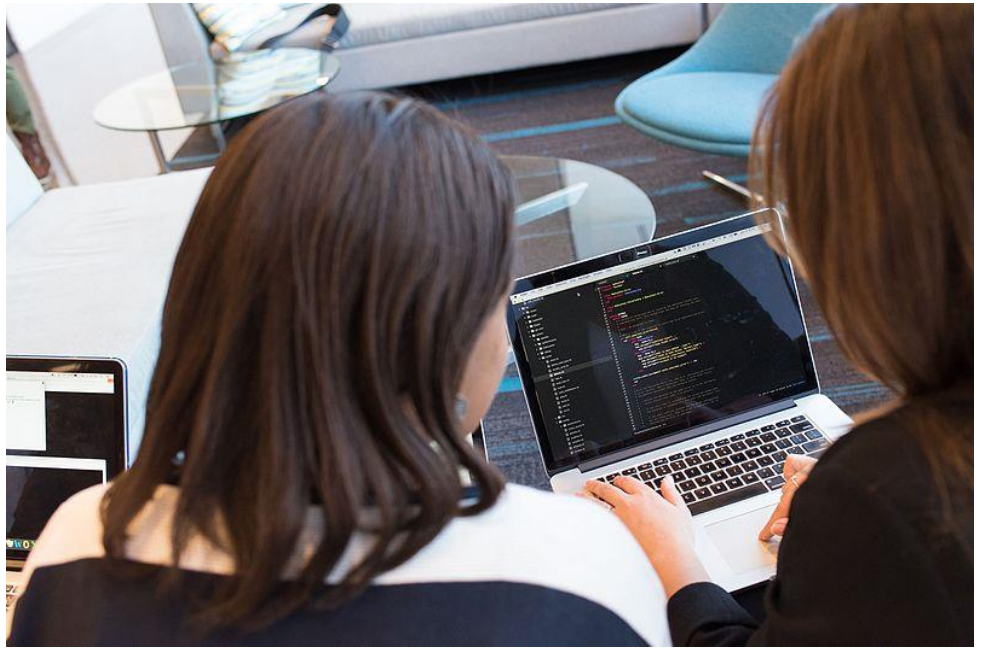
Pair Programming

Pair Programming ist eine Methode aus der Softwareentwicklung, bei der sich zwei Programmierer:innen einen Arbeitsplatz teilen und gemeinsam programmieren.

Rollenverteilung beim Pair Programming

(Williams, 2001)

- Driver (Fahrer)
  - Der Driver ist die ausführende Kraft
  - Er ist für das Schreiben des Codes und die Bedienung der Maus und Tastatur zuständig
- Navigator / Observer (Beifahrer)
  - Der Navigator unterstützt den Driver mit Tipps und Ideen und achtet auf Fehler
  - Er ist für die „strategische Planung“ zuständig
- Die Rollen werden regelmäßig getauscht



Williams, L. (2001). Integrating pair programming into a software development process. [DOI]

Effekte von Pair Programming

- Die Qualität des Codes nimmt zu / Anzahl der Fehler nimmt ab *(McDowell et al., 2002)*
- Gesamteffizienz nimmt ab *(Lui und Chan, 2006)*
  - Insbesondere bei einfachen Aufgaben
- Kollaboration und Diskussionen führen zu gesteigertem Lernerfolg *(Porter et al., 2013)*
- Pair Programming stärkt das Selbstvertrauen und die Freude am Programmieren *(McDowell, 2003; Williams et al., 2000)*
- In einer Untersuchung von McDowell et al. (2002) schnitten die Schüler:innen, die nach dem Pair Programming Prinzip unterrichtet wurden signifikant besser ab als die Kontrollgruppe

McDowell, C.; Werner, L.; Bullock, H. & Fernald, J. (2002). The Effects of Pair-Programming on Performance in an Introductory Programming course.
Lui, K. M. & Chan, K. C. C. (2006). Pair programming productivity: Novice–novice vs. expert–expert.
Williams, L., Kessler, R. R., Cunningham, W., & Jeffries, R. (2000). Strengthening the case for pair programming.
Porter, L., Guzdial, M., McDowell, C., & Simon, B. (2013). Success in Introductory Programming: What Works?

Reflexion

- Untersuchungen haben gezeigt, dass sowohl leistungsschwächere als auch leistungsstärkere Schüler:innen von Pair Programming profitieren können *(McDowell, 2002)*

Überlegen Sie, woran das liegen könnte! Wie unterscheiden sich die Lernaktivitäten der beiden Gruppen?

McDowell, C.; Werner, L.; Bullock, H. & Fernald, J. (2002). The Effects of Pair-Programming on Performance in an Introductory Programming course.

Vergleich verschiedener Paarungen

(Lui und Chan, 2006)

	Experte	**Novize**
Experte	• Hohe Produktivität • Kaum neue Einsichten • Etablierte Lösungen	• Experte kann Novizen anleiten • Novize bringt neue Ideen mit • Experte lernt durch das Erklären • Novize möglicherweise eingeschüchtert (passiv) • Experte möglicherweise ungeduldig und bestimmend
Novize		• Kann zu besseren Ergebnissen im Vergleich zu Einzelleistungen führen • Ohne Experten ist es schwierig, gute Praktiken zu entwickeln

Lui, K. M., & Chan, K. C. C. (2006). Pair programming productivity: Novice–novice vs. expert–expert. *International Journal of Human-Computer Studies*, *64*(9), 915–925. [DOI]

Softwareprojekte

(Schubert und Schwill, 2011)

Schubert, S., & Schwill, A. (2011). *Didaktik der Informatik*

Projektunterricht

(Schubert und Schwill, 2011)

- Ein Projekt ist eine „**längere**, **fächerübergreifende** Unterrichtseinheit, die durch **Selbstorganisation** der Lerngruppe gekennzeichnet ist und bei der der **Arbeits- und Lernprozess** ebenso wichtig ist wie das **Ergebnis oder Produkt**, das am Ende des Projekts steht“ *(Schubert und Schwill, 2011)*
- Der Informatikunterricht ist für Projektunterricht besonders geeignet, da Projekte fester Bestandteil der professionellen Softwareentwicklung sind
  - Große Möglichkeiten für Wechselwirkungen der Schüler:innen untereinander
  - Viel Freiraum für eigene Aktivitäten

Schubert, S., & Schwill, A. (2011). *Didaktik der Informatik*

Merkmale von Projektunterricht I

(Gudjons, 1986 zitiert nach Schubert und Schwill, 2011)

- Situationsbezug und Lebensweltorientierung
  - Projektthemen entstammen der Lebenswelt der Schüler:innen und sind inhaltlich nicht an Fachwissenschaften und somit auch nicht an Schulfächer gebunden
- Orientierung an Interessen der Beteiligten
  - Wünsche, Bedürfnisse und Abneigungen der Projektbeteiligten (Schüler:innen) werden berücksichtigt
- Selbstorganisation und Selbstverantwortung
  - Beteiligte bestimmen gleichberechtigt Ziele und Vorgehen im Projekt
  - Regelmäßig eingeschobene Reflexionsphasen
- Gesellschaftliche Praxisrelevanz
  - In Projekten soll die Wirklichkeit nicht nur – wie im traditionellen Unterrichtsalltag – beobachtet, gespeichert, analysiert oder simuliert, sondern auch verändert werden, und seien die Veränderungen auch noch so klein.

Gudjons, 1986 zitiert nach Schubert und Schwill, 2011

Merkmale von Projektunterricht II

(Gudjons, 1986 zitiert nach Schubert und Schwill, 2011)

- Zielgerichtete Projektplanung
  - Fortlaufende Planung und Korrektur bisheriger Aktivitäten
- Produktorientierung
  - Vorzeigbares Produkt (z.B. ein Film, ein Bericht, ein Modell, ein Programm mit Dokumentation), das der Öffentlichkeit zugänglich gemacht wird
- Einbeziehen vieler Sinne
  - Lernen als „ganzer Mensch“
- Soziales Lernen.
  - Gemeinsames Lernen und Handeln in Gruppen
  - Konflikte lösen und kooperativ arbeiten.
- Interdisziplinarität
  - Fächerübergreifend (i.d.R. kann ein Schwerpunkfach zugeordnet werden)

Gudjons, 1986 zitiert nach Schubert und Schwill, 2011

Differenzierung des Projektbegriffs

(Baumann, 1996)

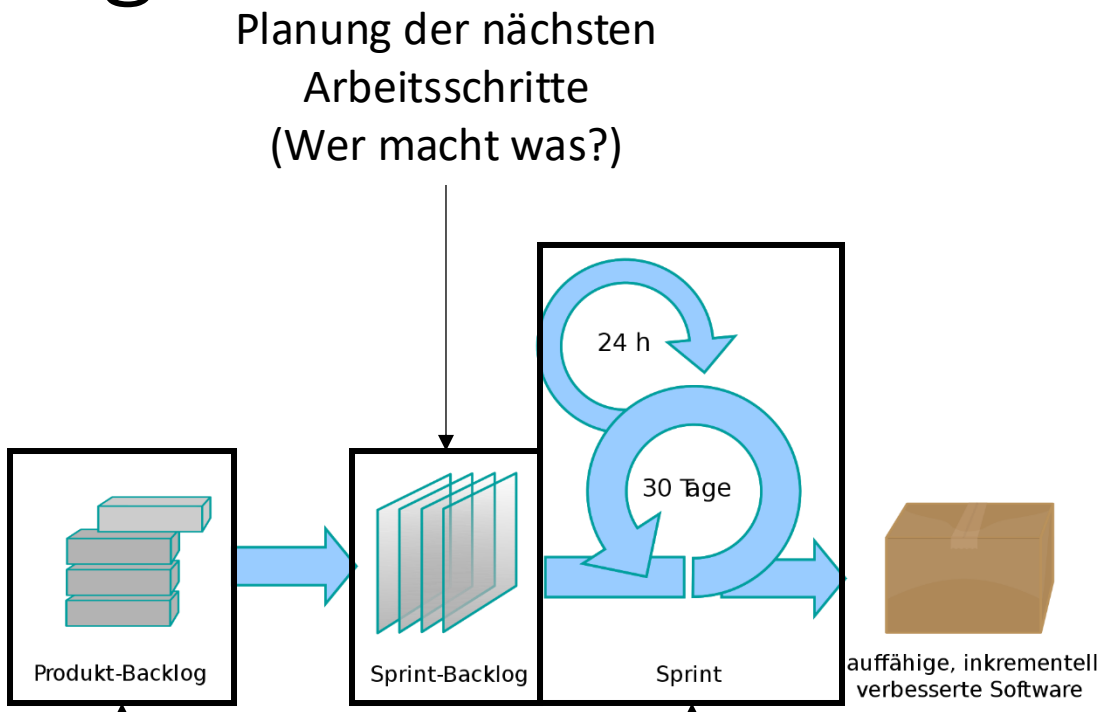
Informatische Projekte	Pädagogische Projekte
Effizienz und Qualität Leistungsorientierung	Lernerfolg steht im Vordergrund Soziales Lernen
Größtenteils unabhängiges Arbeiten in Einzelarbeit	Konsequente Kooperation
Spezialisten • Fokus liegt auf einem Teilbereich • Klare Verantwortlichkeiten	Generalisten • Überblick über das Gesamtprojekt • Wechselnde Aufgabenbereiche
Hierarchische Organisation	Gleichgestellte Teammitglieder • Gleichberechtigt • Fortlaufende Berührungspunkte

Baumann, R. (1996). Didaktik der Informatik

Agile Softwareentwicklung

am Beispiel SCRUM

- Leichtgewichtiges, iteratives und inkrementelles Framework für die Entwicklung, Bereitstellung und Wartung komplexer (Software-)Produkte
- Rollen
  - Scrum Master (Lehrkraft)
  - Product Owner (Lehrkraft oder Dritter)
  - Entwicklerteam (Schüler:innen)



Anforderungen in Form von User Stories:
„Ich als <Rolle>, möchte <Feature>, um <Wert>“

Lern- und Leistungsbeurteilung

im Projekt

- Klausuren als alleiniges Mittel nicht ausreichend, um erworbene Fähigkeiten und Wissen innerhalb eines Projekts zu überprüfen
- Stattdessen: Fortlaufende Beobachtung der Schüler:innen während des laufenden Prozesses
- Mögliche Beurteilungsfaktoren
  - Beitrag zur Gruppenarbeit
  - Schwierigkeitsgrad der übertragenen Aufgabe
  - Einhaltung von Absprachen, Terminen
  - Kommunikation, Hilfsbereitschaft, Beteiligung an Diskussionen
  - Dokumentation und Präsentation der Arbeitsergebnisse
  - Sachgemäßer Umgang mit den Tools
  - …

Übung

(Gruppenarbeit, 30 Minuten)

Skizzieren Sie den Ablauf eines Softwareprojekts, das in der Schule durchgeführt werden kann! Berücksichtigen Sie dabei die Merkmale von Projektunterricht und beantworten Sie die folgenden Fragen:

- Was ist das Thema des Projekts?
- Wie viel Zeit planen Sie für die Durchführung des Projekts?
- Wie teilen Sie die Aufgaben auf?
- Welche Hilfestellungen bieten Sie den Schüler:innen?
- Wie sieht das Endprodukt des Projekts aus?
- Was lernen die Schüler:innen in dem Projekt?

Referenzen I

- Dewey, J., & Kilpatrick, W. H. (1935). *Der Projekt-Plan: Grundlegung und Praxis*. Böhlau. https://doi.org/10.25656/01:13277.
- McDowell, C.; Werner, L.; Bullock, H. & Fernald, J. (2002). The Effects of Pair-Programming on Performance in an Introductory Programming course. In SIGCSE '02: Proceedings of the 33rd SIGCSE technical symposium on Computer science education (S. 38 – 42). Cincinnati: ACM. https://doi.org/10.1145/563340.563353.
- Mcdowell, C.; Werner, L., Bullock, H. E. & Fernald, J. (2003). The impact of pair programming on student performance, perception and persistence. In ICSE '03: Proceedings of the 25th International Conference on Software Engineering (S. 602 – 607). Portland: IEEE Computer Society. http://dx.doi.org/10.1109/ICSE.2003.1201243.
- Porter, L., Guzdial, M., McDowell, C., & Simon, B. (2013). Success in Introductory Programming: What Works? In Communications of the ACM 56 (8) (S. 24 – 36). https://doi.org/10.1145/2492007.2492020.
- Lui, K. M. & Chan, K. C. C. (2006). Pair programming productivity: Novice–novice vs. expert–expert. In International Journal of Human-Computer Studies 64 (9) (S. 915 – 925). USA: Academic Press. https://doi.org/10.1016/j.ijhcs.2006.04.010.

Referenzen II

- Schubert, S., & Schwill, A. (2011). *Didaktik der Informatik* (2. Aufl). Spektrum, Akad. Verl. https://doi.org/10.1007/978-3-8274-2653-6
- Williams, L., Kessler, R. R., Cunningham, W., & Jeffries, R. (2000). Strengthening the case for pair programming. In IEEE Software 17 (4) (S. 19 – 25). https://doi.org/10.1109/52.854064.
- Williams, L. (2001). Integrating pair programming into a software development process. In Proceedings 14th Conference on Software Engineering Education and Training. 'In search of a software engineering profession' (Cat. No.PR01059). https://doi.org/10.1109/CSEE.2001.913816.

Illustrationen

- Illustrationen von Limpitsouni, K. unter freier Lizenz via https://undraw.co

FAIBLE.nrw

Beteiligte Hochschulen:

- RWTH-Aachen
- Westfälische Wilhelms-Universität Münster
- Universität Duisburg-Essen
- Universität Bonn
- Universität Paderborn
- Technische Universität Dresden
- Carl von Ossietzky Universität Oldenburg

ORCA.nrw
Das Landesportal für Studium und Lehre.

Ein Kooperationsvorhaben empfohlen durch die:
DIGITALE HOCHSCHULE NRW
INNOVATION DURCH KOOPERATION

gefördert durch:
Ministerium für Kultur und Wissenschaft des Landes Nordrhein-Westfalen