DIAL
Digital access to libraries

"Création d'exemples résolus avec objectifs étiquetés pour l'apprentissage de la programmation avec Python"

Goletti, Olivier ; De Pierpont, Florian ; Mens, Kim

ABSTRACT

L'enseignement de la programmation peut profiter de l'utilisation de stratégies d'instruction explicites pour diminuer la charge cognitive et favoriser le transfert des apprentissages. Une de ces stratégies est l'utilisation d'exemples résolus avec objectifs étiquetés. Nous avons utilisé une méthodologie d'analyse de tâche pour extraire à des experts en programmation les connaissances nécessaires à la création de tels exemples résolus. Cet article présente la méthodologie utilisée et propose un document de formation réutilisable par d'autres membres de la communauté enseignante d'informatique.

CITE THIS VERSION

Goletti, Olivier ; De Pierpont, Florian ; Mens, Kim ; et. al. *Création d'exemples résolus avec objectifs étiquetés pour l'apprentissage de la programmation avec Python.*Didapro 9 – DidaSTIC (Le Mans, du 18/05/2022 au 20/05/2022). http://hdl.handle.net/2078.1/260190

Le dépôt institutionnel DIAL est destiné au dépôt et à la diffusion de documents scientifiques émanant des membres de l'UCLouvain. Toute utilisation de ce document à des fins lucratives ou commerciales est strictement interdite. L'utilisateur s'engage à respecter les droits d'auteur liés à ce document, principalement le droit à l'intégrité de l'œuvre et le droit à la paternité. La politique complète de copyright est disponible sur la page Copyright policy

DIAL is an institutional repository for the deposit and dissemination of scientific documents from UCLouvain members. Usage of this document for profit or commercial purposes is stricly prohibited. User agrees to respect copyright about this document, mainly text integrity and source mention. Full content of copyright policy is available at Copyright policy

Available at: http://hdl.handle.net/2078.1/260190

[Downloaded 2022/10/19 at 22:04:28]

Création d'exemples résolus avec objectifs étiquetés pour l'apprentissage de la programmation avec Python

Olivier Goletti, UCLouvain, INGI
olivier.goletti@uclouvain.be
@OGoletti

Avec :
Florian De Pierpont, UCLouvain
Prof. Kim Mens, UCLouvain, INGI

UCLouvain

En un slide

- Cadre : Thèse de doctorat
- Présentation d’une **stratégie** d’enseignement **explicite**
- Création et **publication** de matériel de formation pour **tuteurs**
- **Étape intermédiaire** avant formation, observations, analyse

Contexte

Précédentes publications

What's the plan?

Promoting Learning Transfer in Computer Science Education by Training Teachers to use Explicit Programming Strategies

Olivier Goletti*
ICTEAM/INGI, UCLouvain
Louvain-la-Neuve, Belgium

ABSTRACT

Some Computer Science concepts, and programming in particular, are hard to learn. As CS is (re-)entering national school curricula throughout the world, qualified CS teachers need to be trained. In this PhD work we will propose a training that will help teachers teach those concepts effectively. Based on the educational framework of learning transfer and cognitive load theory, we will do this through evidence-based instructional strategies. These explicit programming strategies aim to decrease cognitive load and foster learning transfer. My PhD will advance the topic of CS teacher training by understanding how CS teachers apply those programming strategies in their teaching through qualitative studies and by designing a validated training that can be used with tutors and teacher trainers.

CCS CONCEPTS

the ICER'21 Doctoral Consortium will allow me to discuss my methodological setup for my next experiment.

2 CONTEXT AND MOTIVATION

Basic knowledge of computing is more and more regarded as a fundamental skill nowadays. Computing is re-entering school curricula throughout the world, creating a lot of research opportunities. In French-speaking Belgium, an educational reform is ongoing and will bring some basic computing knowledge in K-9[2]. This will lead to a demand in trained teachers, since no similar topic is present as a mandatory subject right now [7]. However, teachers recognise a need for training if they would have to teach this subject [8]. Training teachers is indeed critical to teach the hard topic of programming [3]. Considering the above, it is important to investigate and invest in teacher training.

Où en est ma thèse ?

☑ **Analysis**: Analyse a CS1 course through the lens of learning transfer

☑ **Exploration**: Explore how tutors with CS knowledge can integrate explicit programming strategies in their instructional practice.

▶ **Case study: Understand how tutors apply the strategies and whether they indeed foster learning transfer**

☐ **Validation**: Assess how effectively tutors can be trained to use strategies in classroom and the impact on student learning

☐ **Different audience**: Understand how teacher trainers apply the strategies

Analyse d'un cours d’informatique

- Basée sur le **Transfert des apprentissages**
- Améliorations potentielles identifiées :
  - **Mettre en évidence** les opportunités de **transfert** en **amont**
  - Proposer **plus de stratégies explicites de rappel** dans le cours

Besoin de stratégies explicites pour programmer

- "human-executable procedure for accomplishing a programming task"
- **Basées sur la littérature**
- Pour réduire **charge cognitive** des étudiants
- Favoriser le **transfert**
- Plus **d'étayage** le cadre de méthodologies d'enseignement moins guidées
- Appliquées par des **tuteurs**

Exploration

Tutors' Experiences in Using Explicit Strategies in a Problem-Based Learning Introductory Programming Course

Olivier Goletti*
ICTEAM/INGI, UCLouvain
Louvain-la-Neuve, Belgium
olivier.goletti@uclouvain.be

Kim Mens
ICTEAM/INGI, UCLouvain
Louvain-la-Neuve, Belgium
kim.mens@uclouvain.be

Felienne Hermans
LIACS, Leiden University
Leiden, The Netherlands
f.f.j.hermans@liacs.leidenuniv.nl

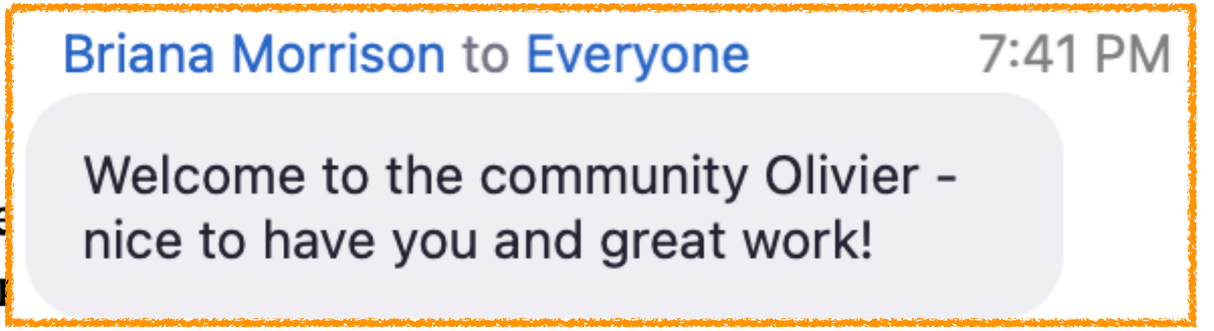
ABSTRACT

In programming education, explicit strategies are gaining traction. The reason for this study was to improve an introductory programming course based on a problem-based methodology, by using more explicit programming strategies. After analysing a previous run of this course for first year undergraduate students, we concluded that such strategies could improve learning transfer for students across

1 INTRODUCTION

Introductory programming courses are increasingly taught using problem-based and project-based learning methodologies (PBL) [23]. At UCLouvain, an introductory bachelor computer science (CS1) course given to the future civil engineers and bachelors in CS, is taught following a methodology inspired by PBL. In this CS1 course, tutors facilitate students' work during lab sessions. Tutors are more

Exploration



Tutors' Experience
Problem-Based Learni

Olivier Goletti*
ICTEAM/INGI, UCLouvain
Louvain-la-Neuve, Belgium
olivier.goletti@uclouvain.be

Kim Mens
ICTEAM/INGI, UCLouvain
Louvain-la-Neuve, Belgium
kim.mens@uclouvain.be

Felienne Hermans
LIACS, Leiden University
Leiden, The Netherlands
f.f.j.hermans@liacs.leidenuniv.nl

ABSTRACT

In programming education, explicit strategies are gaining traction. The reason for this study was to improve an introductory programming course based on a problem-based methodology, by using more explicit programming strategies. After analysing a previous run of this course for first year undergraduate students, we concluded that such strategies could improve learning transfer for students across

1 INTRODUCTION

Introductory programming courses are increasingly taught using problem-based and project-based learning methodologies (PBL) [23]. At UCLouvain, an introductory bachelor computer science (CS1) course given to the future civil engineers and bachelors in CS, is taught following a methodology inspired by PBL. In this CS1 course, tutors facilitate students' work during lab sessions. Tutors are more

Quatre stratégies

- Explicit Tracing
- Subgoal-labeled worked examples
- Parsons' problems
- Explicit problem solving

Focus

De cette publication

Cognitive Load Theory

- CLT : Théorie basée sur l’architecture cognitive humaine
- **Mémoire de travail** limitée
- Impact sur l’instruction : utiliser des **stratégies** pour réduire la charge mentale

Learning Transfer

- **Ré-utiliser** des **connaissances** acquises (d'une tâche source) dans un nouveau contexte (une tâche **cible**)

- Un processus **actif**, dépendant du **contexte** qui implique l'apprenant et qui mène à de **nouveaux apprentissages**

Focus sur une stratégie

- Exemples résolus avec objectifs étiquetés
- Subgoal-labeled Worked-examples
- SLWEs

SLWEs

- Worked examples: "worked examples focus the learners' attention on problem states and associated operators (i.e. solution steps), enabling them to induce generalised solutions"
- Subgoal-labels: "Novices solve programming problems better when they explicitly learn the subgoals of a procedure because they often do not recognise these functional pieces on their own "
- Worked examples need a cover story though => combination of the two

Existant

- Publication protocole, SLWEs, efficacité pour Java
- Principal site web de B. Morrison et L. Margulieux
- Ebook avec des SLWEs pour Java

=> adaptation

=> création de nouveaux SLWEs

Existant

Figure 1. Subgoals identified through TAPS protocol.

Subgoals for evaluating and writing expression (assignment) statements	
A. Evaluate expression statement	B. Write expression statement
1. Determine whether data type of expression is compatible with data type of variable 2. Update variable for pre based on side effect 3. Solve arithmetic equation 4. Check data type of copied value against data type of variable 5. Update variable for post based on side effect	1. Determine expression that will yield variable 2. Determine data type and name of variable and data type of expression 3. Determine arithmetic equation with operators 4. Determine expression components 5. Operators and operands must be compatible
Subgoals for evaluating and writing selection statements	
A. Evaluate selection statement	B. Write selection statement
1. Diagram which statements go together 2. For if statement, determine whether expression is true or false 3. If true – follow true branch, if false –follow else branch or do nothing if no else branch	1. Define how many mutually exclusive paths are needed 2. Order from most restrictive/selective group to least restrictive 3. Write if statement with Boolean expression 4. Follow with true bracket including action 5. Follow with else bracket 6. Repeat until all groups and actions are accounted for
Subgoals for evaluating and writing loops.	
A. Evaluate loops	B. Write loops
1. Identify loop parts a. Determine start condition b. Determine update condition	1. Determine purpose of loop a. Pick a loop structure (while, for, do_while) 2. Define and initialize variables

Comment extraire les labels ?

- Analyse de tâche par résolution de problème (TAPS)
- Un ou plusieurs (4) Experts du sujet (Subject-Matter Expert - SME)
- Un Analyste (Knowedlge Extractor Expert - KEE)

Résultats

De cette publication

Concepts pour lesquels des SLWEs ont été créés

Concept	Lecture du code	Écriture du code	Adaptation de l'existant en Java
Affectation	x	x	x
Condition	x	x	x
Boucle	x	x	x
Fonction	x	x	x
Parcours de chaîne/liste		x	
Lecture de fichier		x	
Écriture de fichier		x	
Création/MàJ dictionnaire		x	
Création de classe		x	
Ajout noeud liste chaînée		x	
Retrait noeud liste chaînée		x	

Exemple

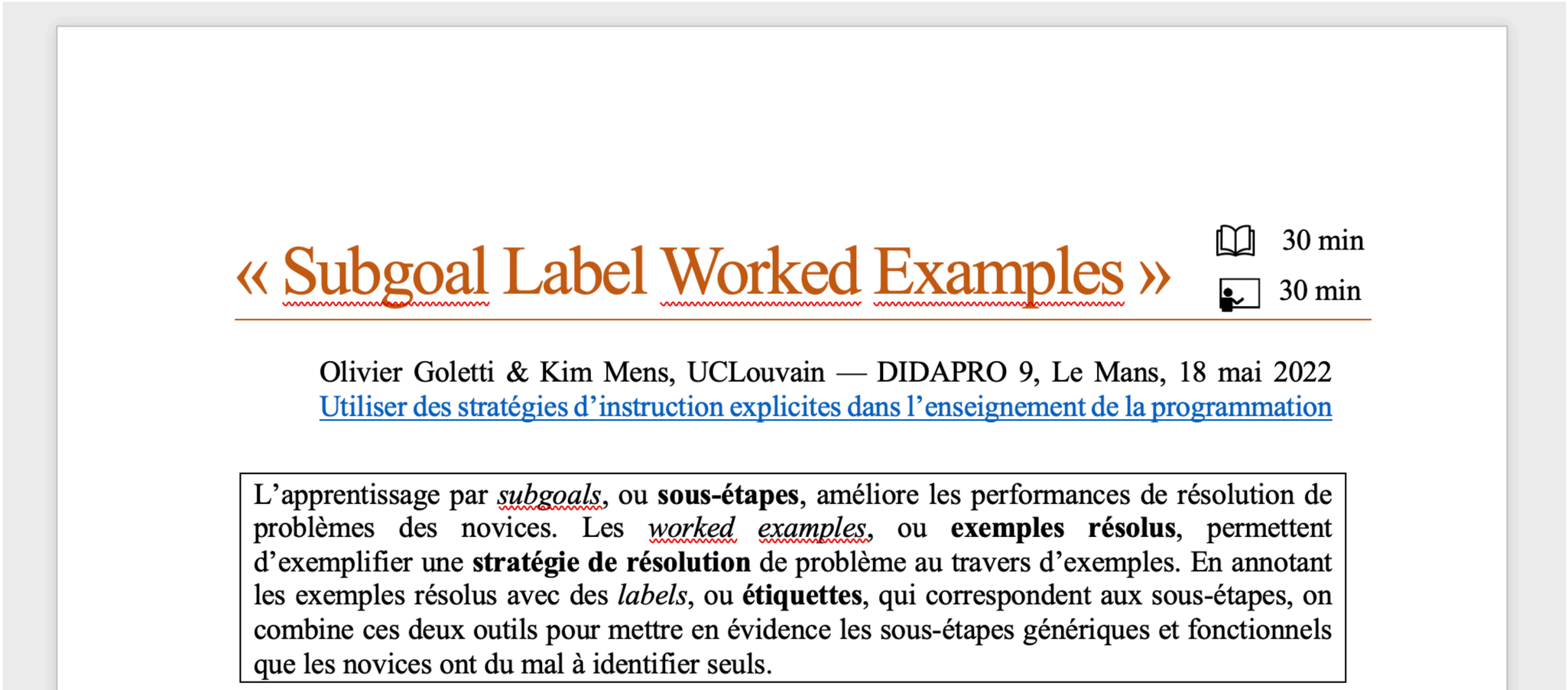
1. **Ouvrir** le fichier
   (a) Identifier le nom et le chemin du fichier (typiquement dans `filename`)
   (b) Choisir le mode `"r"`
   (c) Ouvrir le fichier avec :
      — Soit : `with open(filename , mode) as f :`
      — Soit : `f = open(filename , mode)`
2. **Traitement** du fichier en fonction du format
   (a) Parcours des lignes
      — ligne par ligne avec `f.readline()`
      — en itérant sur les lignes avec `for line in f:`
   (b) Traitement des lignes
      — Retrait des blancs en début et fin de ligne avec `line.strip()`
      — Séparer les éléments en fonction du format avec `line.split()`
      — Convertir les éléments en fonction du type attendu
   (c) Traiter les erreurs de formatage du fichier (ignorer ligne, `raise ValueErr`
3. **Fermeture** du fichier
   — Avec un `with`, il n'y a rien à faire
   — Sinon, avec un `f.close()`
4. Gérer les **exceptions** susceptibles de se produire durant le traitement du fichier (typiquement `IOError`)
   (a) Mettre le code dans un `try : ... except :`
   (b) Traiter les exceptions par le suite avec des `except error_type: ...`

Ecrire une fonction read_coordinates(filename) qui lit les coordonnées du fichier nommé filename dont chaque ligne est au format x,y et retourne une liste de tuples (x, y)

```
def read_coordinates(filename):
    l = []
    try:                                              # ③ fermeture
        with open(filename, 'r') as f:                # 1a, 1b ; ① ouverture
            for line in f:                            # 2a ; ② traitement
                tokens = line.strip().split(',')      # 2b-c
                if len(tokens) != 2:                  # 2c
                    raise ValueError(
                        "il faut deux valeurs par ligne séparées par une virgule")
                l.append((float(tokens[0]), float(token[1])))   # 2b
    except IOError:                                   # ④ Exceptions
        return []
    return l
```

Matériel de formation

- http://hdl.handle.net/2078.1/260190



« Subgoal Label Worked Examples »

30 min
30 min

Olivier Goletti & Kim Mens, UCLouvain — DIDAPRO 9, Le Mans, 18 mai 2022
Utiliser des stratégies d'instruction explicites dans l'enseignement de la programmation

L'apprentissage par *subgoals*, ou **sous-étapes**, améliore les performances de résolution de problèmes des novices. Les *worked examples*, ou **exemples résolus**, permettent d'exemplifier une **stratégie de résolution** de problème au travers d'exemples. En annotant les exemples résolus avec des *labels*, ou **étiquettes**, qui correspondent aux sous-étapes, on combine ces deux outils pour mettre en évidence les sous-étapes génériques et fonctionnels que les novices ont du mal à identifier seuls.

La suite

Formation et suivi des tuteurs

en cours…

- Formation via le document
- Suivi via observations, focus groups, interview (voir vidéo)
- Analyse qualitative de l’utilisation en classe de la stratégie
- Vers un modèle d’adoption de stratégies explicites

Intégration dans le cours

- Intégration dans le matériel du cours
- Mémorant

Conclusion

- Un **effort non-négligeable** à faire pour appliquer les résultats de la littérature
- **Mise à disposition** de la communauté des SLWEs pour Python
- **Analyse qualitative** de **l’utilisation** de stratégies à venir