exploring evolution of program design skills

Francisco Castro
fgcastro@cs.wpi.edu

Kathi Fisler
kfisler@cs.brown.edu

Introduction

- Learning effective **program design** remains a nontrivial goal for novice programmers in introductory computing courses
- We report on the first phase of a project exploring the **evolution of students' design skills** in a two-course introductory sequence that uses the **How to Design Programs** (HTDP) curriculum
- The program design skills fostered by HTDP and how students learn with HTDP remains largely unexplored in CSEd research
- We collected **code output**, **interview**, and **think-aloud data** from students taking CS1 through to CS2
- Initial analysis of the CS1 data yielded a **multi-strand framework** for multiple, interrelated program design skills and their progressions

How to Design Programs

HTDP is an introductory computing curriculum used in higher education institutions and some K-12 programs that teaches a **multi-step process** of program design called the **design recipe**

DESIGN RECIPE STEPS

1: The Data Definition
2: Examples of Data
3: Contract and Purpose
4: Test Cases / Function Examples
5: Datatype Template
6: Function Details

```
; A list-of-string is
; - empty or
; - (cons string list-of-string)

(define names (cons "barry" (cons "barry" (cons "jesse" empty))))
(define one-name (cons "jay" empty))

; find-name : list-of-string string -> boolean
; Produces true if the given string appears in a given list,
; false otherwise

(check-expect (find-name empty "barry") false)
(check-expect (find-name names "barry") true)
(check-expect (find-name names "wally") false)

; List Template
#|
(define (list-function list-input)
  (cond [(empty? list-input) ... ]
        [(cons? list-input) ... (first list-input)
                            ... (list-function (rest list-input)) ... ]))
|#

(define (find-name names-list name)
  (cond [(empty? names-list) false]
        [(cons? names-list) (cond [(string=? (first names-list) name) true]
                                  [else (find-name (rest names-list))])]))
```

The recipe promotes abstract and concrete thinking. Each step builds on another, scaffolding the process of program design, while also serving as a diagnostic for educators.

Think about the shape of the data

Think about the problem space: input/output space, boundary cases

Does the code structure match the data?

Methods

- Participants: 13 volunteer students taking CS1 through to CS2

First Exam Grade: A (90-100) B (80-89) C (65-79)

We open coded interview and think-aloud data to identify emergent themes

Some themes suggest **progression of sophistication** within a core **skill**

Some themes suggest **factors** that contribute to design decisions/practice

- Quality attributes
- Value judgments
- Knowledge recall
- Metacognitive reflections

Multi-strand Skill Framework	Prestructural	Unistructural	Multistructural	Relational
Methodical Choice of Tests and Examples	Cannot write tests	Cannot explain purpose of written tests	Multiple but unrelated tests	Tests cover a problem space
Writing and Evaluating Function Bodies	Cannot write functions	Primitive operations on primitive types	Complex expressions, no function semantics	Semantics of function calls & return contexts
Decomposing Tasks and Composing Solutions	Does not identify relevant tasks	Identifies tasks, but no logical separation	Decomposed tasks, no semantic composition	Semantic decomposition & composition of tasks
Leverage Multiple Function Representations	Just dives into writing code	Mechanical use of design recipe	Representations seen within problem context	Mechanisms of how representations relate
		Syntactic		Semantic

Observations and Findings

- The multi-strand framework provides a more nuanced understanding of the evolution of students' design skills: it captures (1) different skills targeted/promoted by the curriculum and (2) variances in the ways students develop in these skills

Non-uniform and non-linear skill progressions
Student progress across skills were not simultaneous. There were also instances of regression within skills.

Skills vary in abstractness
Skills vary in their mechanical application and the degree of abstract thinking they require.

Need to consider the interaction of problems and activities with the framework
The nature of problems can push students towards particular levels within the framework. The types of activity may affect insights drawn about students' skill levels.

Ongoing Work and Open Questions

Understanding relationships across skill progressions and between skills and identified factors of design practice

- Are students' performance in one skill indicative of their performance in other skills?
- Do factors in design practice influence students' skill levels? – e.g. do they promote jumps between skill levels?

Aligning the framework with expert assessments of HTDP-based work

- Does the framework capture the key nuances that expert instructors use to assess students?

Usability of the framework across programming language shifts and other courses

- Which skills do students transfer when they shift programming languages?
- Do students in other courses (HTDP and non-HTDP based) in other institutions manifest similar skills and skill progressions?

References and Acknowledgments

1. Francisco Castro and Kathi Fisler. 2016. On the Interplay Between Bottom-Up and Datatype-Driven Program Design. In Proceedings of SIGCSE '16. New York, NY, USA: ACM, 205–210.
2. Matthias Felleisen, Robert Findler, Matthew Flatt, and Shriram Krishnamurthi. How to Design Programs. MIT Press, 2001. http://www.htdp.org/.
3. Judy Sheard, Angela Carbone, Raymond Lister, Beth Simon, Errol Thompson, and Jacqueline Whalley. 2008. Going SOLO to Assess Novice Programmers. In Proceedings of ITiCSE '08. New York, NY, USA: ACM, 209–213.
4. Elliot Soloway. Learning to Program = Learning to Construct Mechanisms and Explanations. Commun. ACM, 29(9):850–858, Sept. 1986.

Several CSEd researchers offered valuable advice on the methods used in this work. This work is supported by the National Science Foundation grant 1500039.

WPI BROWN