Validating a SOLO framework on HtDP-based Program-Design

Francisco Castro
fgcastro@cs.wpi.edu

Advisor: Kathi Fisler
kfisler@cs.brown.edu

Dissertation Goal

Develop a conceptual framework of how novice programmers use HtDP to design programs.

To this end, we have developed a SOLO-based framework that details the skills that students from a single HtDP course use, as well as the variations in the way students applied each skill.

– we coded for these skills and skill-level variations from think-aloud and interview data collected from students as they designed solutions for programming problems.

Current questions

- What does it mean for the framework to be valid?
- How do I validate this framework?

How to Design Programs

HtDP teaches a **multi-step process** called the **design recipe**. The curriculum is used in several higher education institutions and some K-12 programs.

1: Data Definition

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

Think about the shape of the data

2: Examples of Data

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

3: Contract and Purpose

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

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

4: Examples / Test Cases

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

5: Datatype Template

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

Does the code structure match the data?

6: Function Details

```
(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))])]))
```

Multi-strand Skill Framework	Prestructural	Unistructural	Multistructural	Relational
Methodical choice of tests and examples	Does not know how to write tests; misses the input/output structure of tests	Able to write tests; descriptions of tests do not explain the purpose of the test(s); does not express the idea of varying test scenarios	Able to write multiple tests; articulates the purpose of individual tests but does not articulate any relationship between or collective purpose for the tests	Able to write tests; identifies a collective purpose for the tests, i.e. boundaries, edge cases, test space coverage, but limited within the context of the problem
Composing expressions within function bodies	Does not know how to define a function	Able to define functions in a simple context – uses primitive operations on primitive types in a function body	Able to define functions whose bodies contain nested non-primitive expressions or function calls, but does not articulate the semantics of how the results of calling a function return to the calling context	Able to define functions whose bodies contain nested non-primitive expressions or function calls and is able to articulate the semantics of how the results of calling a function return to the calling context
Decomposing tasks and composing solutions	Does not identify relevant tasks for a problem	Able to identify relevant tasks but no reflections of separate tasks when talking about the code	Able to identify relevant tasks; articulates the delegation of tasks into separate functions but fails to articulate how to effectively compose the tasks in a way that solves the problem	Able to identify relevant tasks; articulates the delegation of tasks into separate functions and can articulate how to effectively compose the tasks in a way that solves the problem
Leverage multiple representations of functions	Just dives in and writes code; uses only a single representation	Blindly follows the design recipe; sees each function representation as independent of others	Articulates a sense of the function representations talking about or referring to the same computation	Articulates a mechanism through which function representations are related, e.g. template uses types to drive the code structure, execution of a program connects to a test space, etc.

Hypothesis

If the framework is valid –

- HtDP instructors would agree that the framework captures (1) the skills they expect students to demonstrate and (2) the nuances/details with which they differentiate one skill level from another
- The quality of students' skill performance would reliably be reflected in the framework – for example, when instructors use the framework to grade students' skills performance, each student would get the same rating per skill from multiple instructors
- The previous two items hold across instructors and across institutions

Method 1: Validation with instructors

HtDP instructors will rate students on the skills identified in the framework, based on students' think-aloud/interview transcripts and code solutions.

1. Instructors will be given categories of skills on which to rate students, but not the breakdown into levels (prestructural, etc.) – they will rate students' demonstration of the skills on a numeric scale (e.g. 0 – 4) and explain their ratings.
2. Instructors will also report skills/factors they think weren't covered in the set of skills provided.
3. Our analysis of the data will check for the following:
   a. If the instructors' explanations of their ratings correlate with the skill levels of the framework
   b. If the set of skills in the framework is consistent with what instructors expect students to demonstrate or what they grade students on

Method 2: Validation with new student cohorts

1. We will replicate our previous study on HtDP-based CS1 student cohorts from 1-2 other schools – students will design solutions to programming problems as they think aloud.
2. We will code for the skills displayed by these students and contrast them to those from our original dataset. We will also code for the levels within the skills to check if the same levels arise in students' demonstration of the skills.

WPI

Acknowledgments This work is supported by US National Science Foundation grant no.s 1116539 and 1500039.