A Methodology to Improve Dependability in Spreadsheets

Margaret Burnett
Oregon State University
burnett@eecs.oregonstate.edu

Marc Fisher II, Gregg Rothermel
University of Nebraska - Lincoln
{mfisher,grother}@cse.unl.edu

1 Introduction

Spreadsheets are one of the most commonly used end-user programming environments. As such, there has been significant effort on the part of researchers and practitioners to develop methodologies and tools to improve the dependability of spreadsheets.

Our work has focused on the development of the "What You See Is What You Test" (WYSIWYT) family of techniques. WYSIWYT is designed to be seamlessly integrated into a spreadsheet environment and the user's development processes. It uses visual devices that are integrated into the user's spreadsheet to guide the process of finding and fixing problems with the spreadsheet.

There are three major components to the WYSIWYT methodology: a testing and debugging methodology, an assertions mechanism, and the "Surprise-Explain-Reward" strategy.

2 Testing and Debugging Methodology

WYSIWYT provides a testing and debugging methodology [3]. As the user edits their spreadsheet they are provided with visual devices indicating the "testedness" (coverage relative to an underlying data-flow adequacy criterion) of the cells and the spreadsheet and the ability to mark the values in cells as correct or incorrect. If the user marks a cell's value as correct, the testedness of the contributing cells is updated as is the testedness of the spreadsheet. If, instead, the user marks a cell's value as incorrect, additional information is displayed to the user about the "fault likelihood" of cells based on the number of correct and incorrect values to which they contribute[4]. Figure 1 shows our visual devices in the Excel spreadsheet language.

In addition to tracking testedness and fault likelihood, WYSIWYT includes a "Help-Me-Test" feature that generates new test cases to cover unexercised portions of the spreadsheet [2] or replays test cases to re-validate changed portions of the spreadsheet.

3 Assertions Mechanism

WYSIWYT also includes an assertions mechanism where users can supply a valid range of values for a cell [1]. These ranges are then propagated through dependent cells using interval arithmetic techniques, and conflicts between user-supplied ranges, propagated ranges and cell values are displayed to the user.

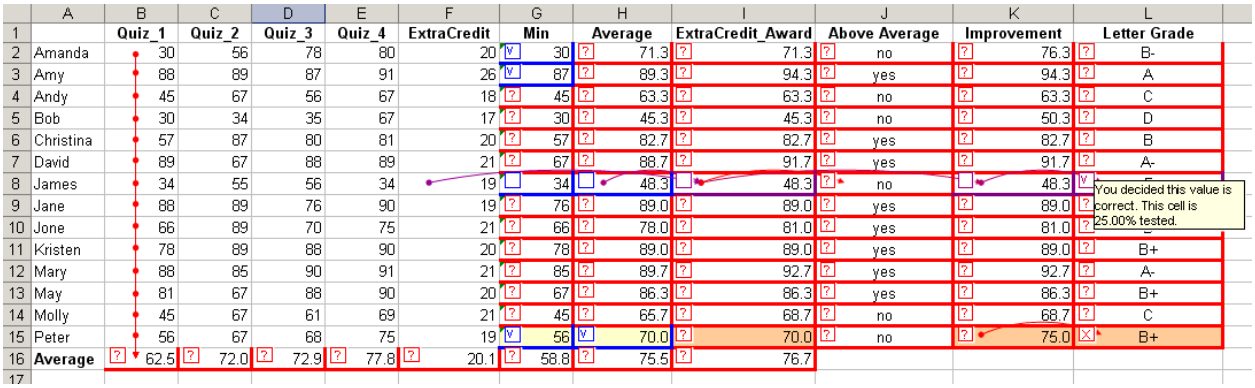
The assertions mechanism interacts with Help-Me-Test in two ways. User-supplied ranges on cells whose formulas are simple data values are used to limit the inputs used when generating new test cases. Help-Me-Test then attempts to generate test cases that violate the ranges on formula cells as they indicate that the user has an error either in a formula or a range.

4 Surprise-Explain-Reward

A key strategy in getting end users to effectively utilize the WYSIWYT methodology is Surprise-Explain-Reward [5]. Surprise-Explain-Reward relies on a user's curiosity about features in the environment. According to research about curiosity, if the user is surprised by something, such as the checkboxes in the spreadsheet, the surprise can arouse the user's curiosity, potentially causing the user to seek an explanation.

All features and feedback must therefore be able to explain themselves. These explanations must do two things: first, make the user aware of why the item is worthy of further attention (i.e., make the rewards clear), and second, help the user make an informed judgment as to whether the reward is worthwhile. Users explore a feature by viewing its explanation, on demand, via tool tips and other low-cost mechanisms.

Dagstuhl Seminar Proceedings 07081
End-User Software Engineering
http://drops.dagstuhl.de/opus/volltexte/2007/1088



	A	B	C	D	E	F	G	H	I	J	K	L
1		Quiz_1	Quiz_2	Quiz_3	Quiz_4	ExtraCredit	Min	Average	ExtraCredit_Award	Above Average	Improvement	Letter Grade
2	Amanda	30	56	78	80	20	30	71.3	71.3	no	76.3	B-
3	Amy	88	89	87	91	26	87	89.3	94.3	yes	94.3	A
4	Andy	45	67	56	67	18	45	63.3	63.3	no	63.3	C
5	Bob	30	34	35	67	17	30	45.3	45.3	no	50.3	D
6	Christina	57	87	80	81	20	57	82.7	82.7	yes	82.7	B
7	David	89	67	88	89	21	67	88.7	91.7	yes	91.7	A-
8	James	34	55	56	34	19	34	48.3	48.3	no	48.3	
9	Jane	88	89	76	90	19	76	89.0	89.0	yes	89.0	
10	Jone	66	89	70	75	21	66	78.0	81.0	yes	81.0	
11	Kristen	78	89	88	90	20	78	89.0	89.0	yes	89.0	B+
12	Mary	88	85	90	91	21	85	89.7	92.7	yes	92.7	A-
13	May	81	67	88	90	20	67	86.3	86.3	yes	86.3	B+
14	Molly	45	67	61	69	21	45	65.7	68.7	no	68.7	C
15	Peter	56	67	68	75	19	56	70.0	70.0	no	75.0	B+
16	Average	62.5	72.0	72.9	77.8	20.1	58.8	75.5	76.7			
17												

Figure 1: WYSIWYT Devices in the Excel Spreadsheet Environment

The reward in the explanation informs the user when weighing costs, benefits, and risks in deciding how to complete a task. By providing users a projection of future benefits, they can better assess if the cost of using the feature is worth their time. If all goes well, if the user follows up as advised in the explanation, rewards will ensue, such as an increase in testing coverage or the discovery of an error.

5 Future Work

Our most recent research continues to investigate Surprise-Explain-Reward, focusing primarily on the explanations aspect. We are doing significant experimental prototyping and empirical work to understand what end-user programmers actually want to know when debugging, how explanations provided by the system can help them, and how they might be able to help each other. We are thus looking into strategies end-user debuggers follow, whether the system helps to support these strategies, and whether the explanations are helpful in improving their debugging strategies.

We are also interested in aspects of the end-user software engineering lifecycle beyond testing and debugging, and how to support them.

References

[1] M. Burnett, C. Cook, and G. Rothermel. End-user software engineering. *Communications of the ACM*, 47(9):53–58, September 2004.

[2] M. Fisher II, G. Rothermel, D. Brown, M. Cao, C. Cook, and M. Burnett. Integrating automated test generation into the wysiwyt spreadsheet testing methodology. *ACM Transactions on Software Engineering and Maintenance*, 15(2):150–194, April 2006.

[3] G. Rothermel, M. Burnett, L. Li, C. DuPuis, and A. Sheretov. A methodology for testing spreadsheets. *ACM Transactions on Software Engineering and Maintenance*, 27(1):110–147, January 2001.

[4] J. Ruthruff, S. Prabhakararao, J. Reichwein, C. Cook, E. Creswick, and M. Burnett. Interactive, visual fault localization support for end-user programmers. *Journal of Visual Languages and Computing*, 16(1-2):3–40, February/April 2005.

[5] A. Wilson, M. Burnett, L. Beckwith, O. Granatir, L. Casburn, C. Cook, M. Durham, and G. Rothermel. Harnessing curiosity to increase correctness in end-user programming. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 305–312, April 2003.