An Abstract of the Thesis of

Miguel A. Arredondo-Castro for the degree of Master of Science in Computer Science presented on May 31, 2001.

Title: End-User Programming In Time: Implementation and Empirical Studies

Abstract approved: Redacted for Privacy

Margaret M. Burnett

The temporal behavior in applications involving visual data can be critical for the correctness of some programs. Forms/3 allows the user to specify temporal behaviors in an independent way, without introducing extraneous code in the original spreadsheet, whereas some other languages define new language devices specific to time. In this thesis, we present the implementation of a new user interface for temporal programming in Forms/3 and the results of two empirical studies. The results of the first study show that one of the models for temporal programing in Forms/3 is more suitable for end users than a traditional stream-based approach representative of the approach used by many other languages. The results of our second experiment show that the explicit information provided by the approach can help the users to judge the correctness of their spreadsheets.

End-User Programming In Time: Implementation and Empirical Studies

by

Miguel A. Arredondo-Castro

A Thesis

Submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Completed May 31, 2001
Commencement June 2002

Master of Science thesis of Miguel A. Arredondo-Castro presented on May 31, 2001.

Approved:

Redacted for Privacy

Major Professor, representing Computer Science

Redacted for Privacy

Head of Department of Computer Science

Redacted for Privacy

Dean of Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Redacted for Privacy

Miguel A. Arredondo-Castro, Author

Acknowledgments

I would like to thank my major professor Dr. Burnett for her support and guidance to complete this thesis.

My thanks also go to my minor professor Dr. Rothermel for his advice on literature about the Oracle problem, to Dr. Cook for his advice on statistical analysis, and to Dr. Chen for being on my committee.

Thanks to all the members of the Forms/3 group for his comments and feedback on the design of the empirical studies, specially to Andy Ko for his help on the cognitive walkthrough of the first experiment.

Special thanks to Mojgan Hajebi for allowing us to promote the experiments to her students, and the students in her Fall 2000 and Winter 2001 classes for their participation in the empirical studies.

Finally, I want to thank my parents: Luz Ma. and Miguel Angel, and my brothers: Ivan and David, for always supporting me.

Table of Contents

Table of Contents (Continued)

List of Figures

List of Figures (Continued)

List of Figures (Continued)

List of Figures (Continued)

List of Tables

List of Appendix Figures

Dedication

Para Rosana, gracias por apoyarme e inspirarme cada día.

End-User Programming In Time: Implementation and Empirical Studies

Chapter 1: Introduction

The temporal behavior in applications involving visual data can be critical for the correctness of some programs. Several researchers have incorporated the definition of temporal behavior into Visual Programming Languages either by defining new language devices specific to time [Duisberg 1986, Wolber 1997, Hibino and Rundersteiner 1997] or by treating time as another dimension [Ashcroft and Wadge 1985, Du and Wadge 1990, McDaniel and Myers 1999].

Previous work in Forms/3 showed how adding a time dimension to the spreadsheet paradigm facilitates programming animations [Atwood et al. 1996, Carlson et al. 1996]. Also in previous work, the Forms/3 group developed several time models that showed how algorithm animation can be achieved without modifying the original algorithm and how to program temporal behaviors in a notation closer to how they evolve over time than a one-dimensional syntax used by many other languages [Burnett et al. 2000, Cao 2000].

In this thesis, we continue this work by implementing one of the models, re-implementing the user interface, and starting a series of empirical studies to study whether these models are actually usable by end users.

The first of these experiments investigates:

1. Usability of one new time model in Forms/3 by end users.
2. If one of the new models is more suitable for end users than a stream-based approach representative of the approach used in many other languages.

We also present the results of an empirical study that investigates:

1. If the explicit information provided by the GUI to specify the temporal behaviors could help the users judge the correctness of their spreadsheets (i.e. help with the oracle problem, which we will discuss in Section 2.1).

In Chapter 2, after an introduction of related work about the oracle problem, we give some background about the time models previously developed in Forms/3. In

Chapter 3, we describe our first experiment, which investigates usefulness of one of the time models to end users. In Chapter 4, we describe the second experiment, which investigates if the visual mechanism used to specify temporal behaviors in Forms/3 can help the users to improve their performance as oracles. The implementation of new features and of model k*x (we will describe time models in Forms/3 in Section 2.3) in JavaForms is discussed in Chapter 5. Finally, Chapter 6 presents the threats to the validity of our experiments, future work planned given the results of our experiments and the conclusions of this thesis.

Chapter 2: Background

2.1 Related Work on the Oracle Problem

As software systems have become part of our daily lives (directly or indirectly), the need to assess their reliability has increased. One approach is program testing, where we give some input data to these systems and verify that the output is correct. The mechanism which checks the correctness of the output is known as an oracle.

There have been several examples of software engineering research about automating test oracles [DeMillo 1991, Vogel 1993, Richardson 1994, Memon et al. 2000]. Vogel presents the CONVEX Integrated Test Environment (CITE) [Vogel 1993]. CITE provides a test environment to test software products produced by Convex. In this test environment, the results of a test process are verified against the expected results for the test stored in the database. The approach assumes the existence of an oracle who first judged correct the outcome for a given test case.

Memon et al. describe a test oracle used to determine if a GUI behaves as expected for a given test case [Memon et al. 2000]. The oracle uses a formal model of the GUI that must be developed by the oracle designer from the GUI specifications. The model is composed of the GUI objects and a set of properties for those objects. GUI actions are represented in the model by their preconditions and effects. The oracle automatically derives the expected state using the model and the actions from a test case. Then an execution monitor takes the actual state of the GUI and a verifier in the oracle compares it with the expected state to verify the correctness of the GUI for the test case.

Using design documentation to generate test oracles has been another approach to the design of test oracles [Peters and Parnas 1994, Peters and Parnas 1998, Richardson et al. 1992]. Peters and Parnas [1994, 1998] describe a test oracle generator (TOG) that produces a software oracle from design documentation. The oracle generator takes a relational program specification using tabular expressions and produces a program that acts as an oracle. The oracle takes the input and output of a test case and returns true if the input-output pair satisfies the relation described by the specification, or false if it does

not. The documentation techniques they propose are useful for specifying imperative programs that are required to give output through program variables upon termination; the software oracle generated can be almost as complicated as the program being tested.

These techniques have been developed in the past because of the difficulty humans have in serving as accurate oracles. Instead of aiming at this problem by replacing the humans' participation, our approach aims to supplement information available to the humans performing this task.

2.2 Testing Spreadsheets

Perhaps the most widely used applications are spreadsheets. They are widely used in organizations [Kappelman et al. 1993] and some of them may be non-trivial for the organization. Although many spreadsheets play an influential role in activities such as tax calculations, student grades, and budgets, they often contain errors [Brown and Gould 1987, Panko 1998, Panko 2000]. Despite the wide use of spreadsheets, little research has focused on the development of methodologies to help with the creation and maintenance of spreadsheets.

Rothermel et al. have developed a "What You See Is What You Test" (WYSIWYT) methodology for testing spreadsheets [Rothermel et al. 1998]. Testing coverage is determined by the definition-use associations present in a spreadsheet (du-associations). A du-association links an expression in a cell formula that defines a cell's value with expressions in other cell formulas that use (reference) the defined cell. An empirical evaluation of WYSIWYT in spreadsheets [Rothermel et al. 2000] showed that users applying the methodology tested spreadsheets more effectively and more efficiently than did users not using the methodology. These results motivate us to extend some of the tools of the methodology to help the user to be a better oracle testing spreadsheet applications involving time.

2.3 Time Models in Forms/3

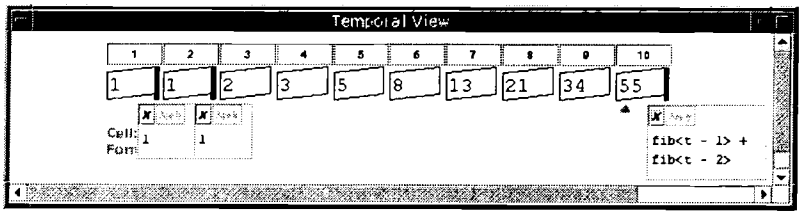
Forms/3 allows end users to specify temporal constraints by reapplying the grid-based way of programming spreadsheet users already know. To specify temporal constraints the end user can choose among four different models. This section describes the four time models developed previously by our group for Forms/3; for a detailed evaluation of the VPL design issues each model introduces see [Cao 2000].

2.3.1 Base Model

This is the default time model in Forms/3. In this model all cells have the same speed.

Figure 2.1 shows how a single cell in Forms/3 can calculate the Fibonacci sequence temporally using the base model. The user defined three temporal regions for the cell `fib`. In the third region the user references `fib`'s value using the format `name<time>`. `t` is a distinguished time meaning "now". For example, in Figure 2.1, `fib<t-1>` references `fib`'s value at the temporal position just before (to the left on the t-axis) it.

(a)



(b)

Figure 2.1: (a) This single Forms/3 cell can compute the Fibonacci sequence temporally under the base model; the computations are specified over time rather than space. (b) The temporal view of the same cell shows its three regions over time.

2.3.2 SNF Model: Slow, Normal, and Fast

This model allows the user to specify one of k speeds for a cell, where k is fixed in the language implementation. In our prototype, k defaults to 3, which allows speeds of Slow, Normal, and Fast. Also fixed in the language are the relationships between Slow, Normal, and Fast (by default the speed difference factor is 2).

Figure 2.2 and Figure 2.3 show how this model might be used by an end-user programmer to produce animated graphics. There are three x-position cells named `slow`, `normal`, and `fast` in this spreadsheet. They have similar formulas (increasing the previous element by 1 each step) but different speeds. The other three cells, `slowCrab`, `normalCrab`, and `fastCrab`, cause the animated crabs to move to the right at different speeds by composing the graphic value of cell `crab` with the appropriate xy-position specification.

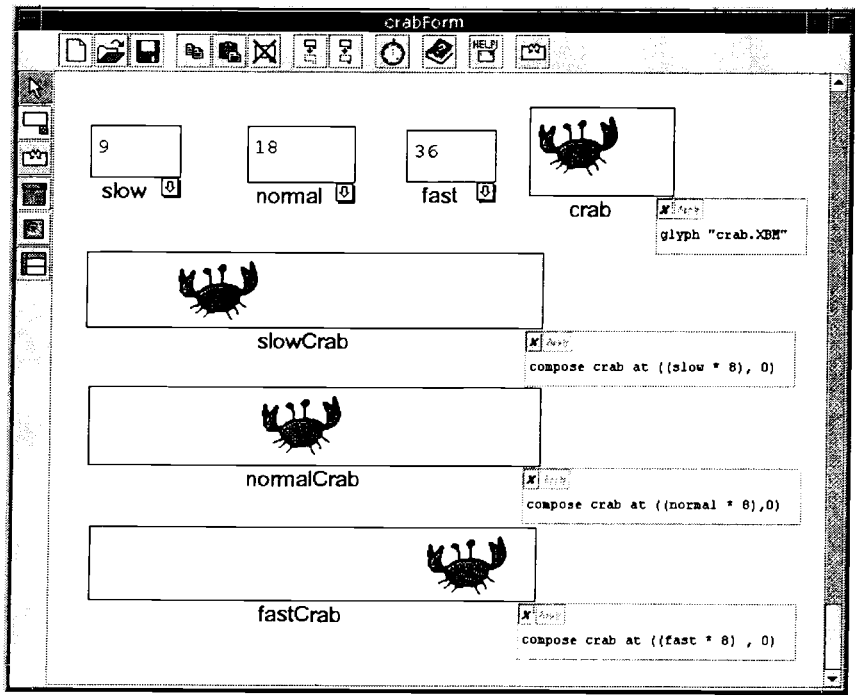


Figure 2.2: The Crabs spreadsheet is an example of how an end-user programmer may use Model SNF. The formulas for cells `slow`, `normal` and `fast` are shown in Figure 2.3.

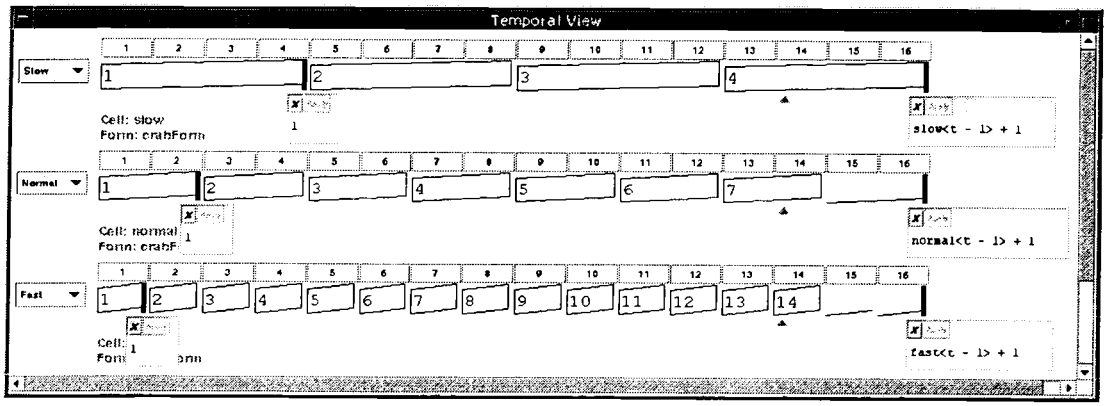


Figure 2.3: Temporal view of the three x-position cells in the Crabs spreadsheet. To select a cell's speed the user selects from a pull-down menu (located on the left part of the screen).

2.3.3 k*N Model: k Times Faster/Slower than Normal

In the k*N model the user can specify the speed of a cell to be an arbitrary constant times faster or slower than Normal speed. Figure 2.4 shows how an end user may define the crabs example using the k*N Model.

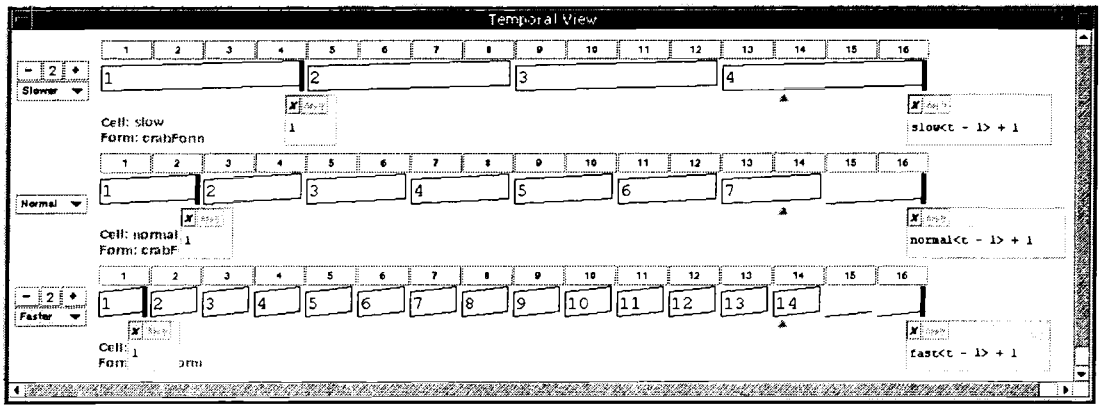


Figure 2.4: Temporal view of the three x-position cells in the Crabs spreadsheet using the Model k*N.

2.3.4 k*x Model: k Times Faster/Slower than x

Model k*x is a straightforward extension of Model k*N. The only difference from Model k*N is that, instead of specifying a cell's speed to be k times faster than Normal, the user specifies the cell's speed to be k times faster than some other cell x. Since there is no "normal" speed in this model, when a cell is not faster or slower than other cell its speed will be shown as "Don't care". Figure 2.5 shows how the crabs example may be defined using this model.

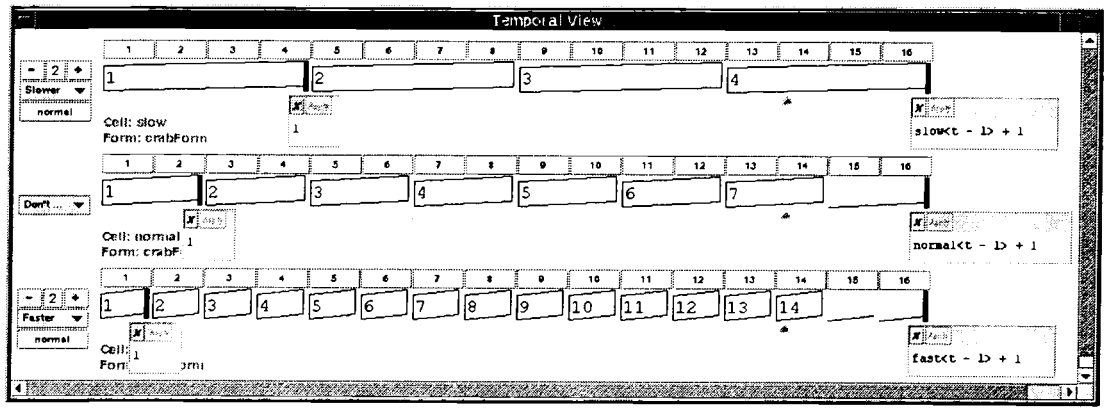


Figure 2.5: Temporal view of the three x-position cells in the Crabs spreadsheet using the Model k*x. Notice how the cells `slow` and `fast` reference the cell `normal` to calculate their speeds.

Although the development of model k*x was done previously, its actual implementation in the prototype is one of the contributions of this thesis.

Chapter 3: An Empirical Study

In the design of the time models in Forms/3, previous members of the Forms/3 group looked to the Cognitive Dimensions[1] to find potential problems related to end users' use of these language devices [Cao 2000]. However, Cognitive Dimensions can help find problems but cannot actually "prove" usability.

To find out if any of these models are actually usable by end-user programmers, we conducted an empirical study. We chose Model SNF as the grid-based one to study because its initial learning time was expected to be small, an important consideration given the limited time possible for initial learning in a lab setting. We chose the traditional stream-based approach with temporal operators such as `fby` to compare it with because it is representative of the approach used in many other languages, and in this sense represents a "standard" in temporal programming.

The objectives of our study were to investigate the following research questions:

RQ1: Do end users who are given Model SNF to express temporal relationships modify a spreadsheet more correctly than end users who are given the traditional approach?

RQ2: Do end users who are given Model SNF to express temporal relationships modify a spreadsheet faster than end users who are given the traditional approach?

RQ3: Do end users have better understanding of a spreadsheet that uses Model SNF than of a spreadsheet that uses the traditional approach?

To answer these research questions we first asked the participants to modify temporal relationships in a spreadsheet using a PC. Then we asked them to fill in values

[1] A small vocabulary of about 12 terms which describes aspects that are cognitively-relevant to all kinds of information-handling devices, meant to be comprehensible to non-specialists [Green and Petre 1996].

on a second spreadsheet without using the computer. Half of the participants completed the experiment using the Model SNF approach and the other half used the traditional approach.

The study was conducted one participant at a time. First, the lecturer led a hands-on tutorial and practice session in Forms/3, introducing the participants to the concept of temporal relationships and how they can be expressed in the Forms/3 environment. Half of the participants were given this instruction only for Model SNF, and the other half were given this instruction only for the traditional approach. Participants in both groups received the same time for training and practice. The kinds of modifications to the spreadsheet taught during the tutorial were also identical, regardless of which time model participants were using. To minimize short-term memorization skill differences as a determining factor, participants were also given a Forms/3 reference card to refer to as needed throughout the experiment.

Following the tutorial, the participants were asked to modify a spreadsheet using the computer and then to answer a written test designed to measure the understanding of a second spreadsheet without using the computer. These tasks will be described in Section 3.2. The data produced by the study included pre-task and post-task questionnaires, electronic transcripts of participants' on-line activities, their final modified spreadsheets, and their written comprehension tests.

3.1 The Participants

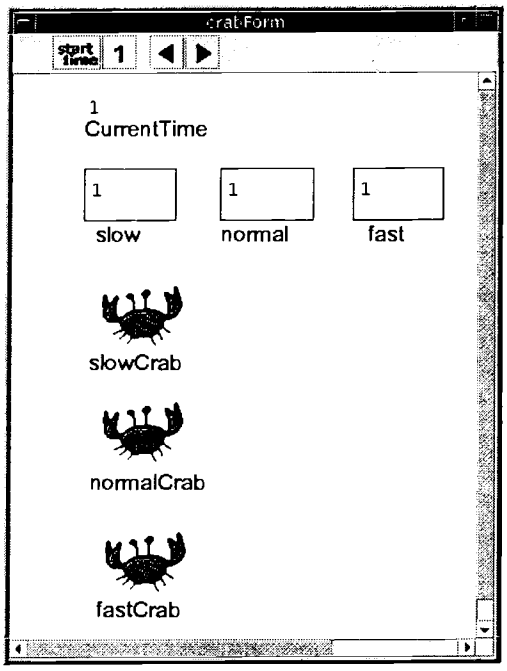
Our participants were drawn from a computer literacy course. The participants were randomly divided into the two groups. Two weeks before the experiment they had completed the spreadsheet module in their class. Thus, all of the participants had previous experience using the Excel spreadsheet system; two also had previous experience with other spreadsheets. None of the participants had used Forms/3 before. In our background questionnaires, none of the participants claimed to be an expert or a heavy user of spreadsheets and none of the participants had programming experience. Table 3.1 shows the previous spreadsheet experience of the participants.

Group	Excel	Lotus	Other	Forms/3
Traditional	16	0	1	0
SNF	16	1	0	0

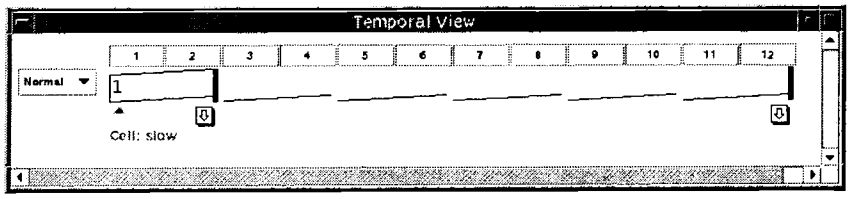
Table 3.1: Previous spreadsheet experience of the end users participating in the empirical study.

3.2 The Problems

The Crabs spreadsheet. This spreadsheet, based on the one described earlier in Section 2.3.2, produces animated graphical output. Recall that the three crabs move along the x-axis. In the task involving this spreadsheet, the participants were to modify the speeds for the cells `slow`, `normal`, and `fast` (Figure 3.1); these cells define the position on the x-axis of the crabs in cells `slowCrab`, `normalCrab`, and `fastCrab` respectively. As given to the participants, the crabs were set to move at the same pace. The participants' goal was to make `normalCrab` move twice as fast as `slowCrab`, and `fastCrab` move twice as fast as `normalCrab`. We chose this problem because a primary motivation behind our work is to allow spreadsheet users to program animated graphics.



(a)



(b)

Figure 3.1: (a) The Crabs spreadsheet as initially presented to participants. See Appendix B for a picture of this spreadsheet with the formulas used to create it.

(b) An example of the temporal view window that participants in the SNF group could use to modify the speed of cells. Initially all cells had speeds set to Normal. See Appendix B for a picture of the temporal view with all the formulas the users had access to.

The Drug Company Deliveries problem. This spreadsheet was used in the written comprehension test. It is based on the spreadsheet used in [Saariluoma and Sajaniemi 1994]. In that paper the participants were asked to act as bookkeepers for a fictitious medical company. Their task was to find information for executives about possible errors in deliveries by using a single formula. In our experiment, we added a time factor. The spreadsheet represents the deliveries of three different pharmaceutical products. Participants were given a picture of the spreadsheet and were asked to figure out how much the company will deliver for each of the three products in the 5th and 8th months. Then they were given the same spreadsheet with modified formulas/speeds and asked to answer the same questions again. See Figure 3.2.

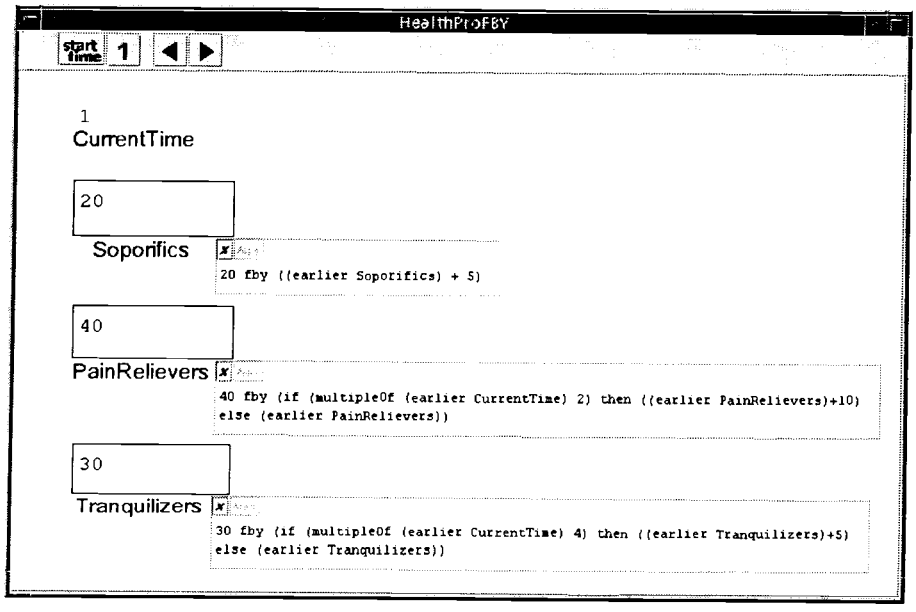


Figure 3.2: Spreadsheet seen by participants in the traditional group.

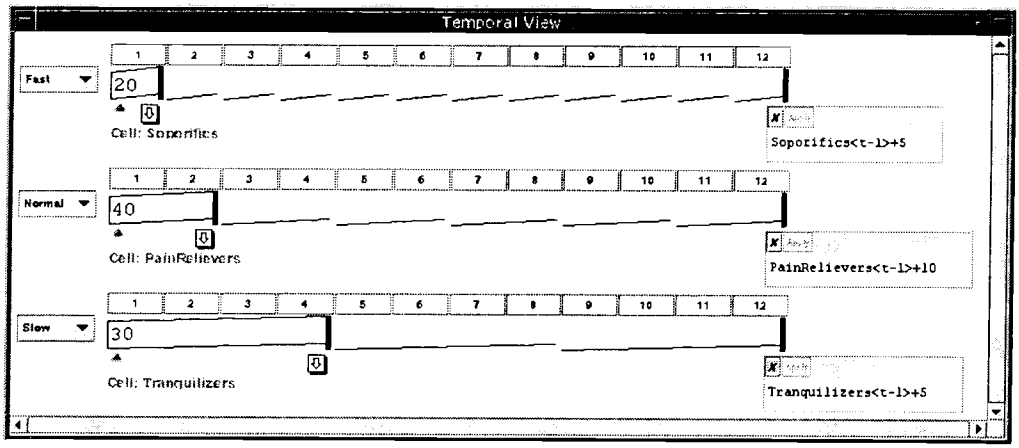


Figure 3.3: Spreadsheet seen by participants in the SNF group.

3.3 RQ1: Correctness Results

Our first research question considers whether using Model SNF to express temporal relationships improved the correctness with which the participants could complete their programming tasks. A summary of the results is shown in Figure 3.4.

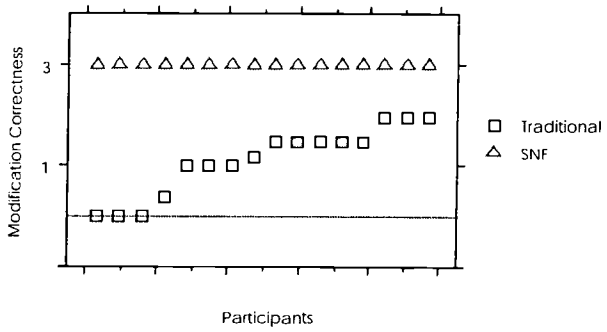


Figure 3.4: Participants in the SNF group modified the Crabs spreadsheet more correctly than participants in the traditional group (SNF mean = 3.0, traditional mean = 1.13).

To measure the correctness of the spreadsheets modified by the participants we scored the correctness of the final formulas using the following criteria: if the position of a crab was the same as we asked, we gave one point; if the crab was not at the specified position but was closer to the desired position than it had been initially, we gave one half of a point. If the crab was closer to the desired position than it had been initially but the participant altered the initial position for the crabs, we gave one fourth of a point. The maximum possible score was 3 (one point for each crab). We defined this grading scheme because the whole modification needs changes in the three crabs, and we awarded fractions of point to avoid a binary grading scheme that would ignore solutions that were almost correct.

As another way of detecting difficulties participants might be experiencing, we also measured how many syntactically invalid formulas the user entered along the way, regardless of whether they were ultimately corrected. See Figure 3.5. While SNF participants did not have opportunities to make syntax errors on the speeds per se, some of them also modified the other parts of the formulas, and there it was possible to make syntax errors. Since dealing with syntax errors can turn a user's attention away from the logic of a problem's solution, the differences in activity spent on syntax may at least partially explain the differences in the correctness results.

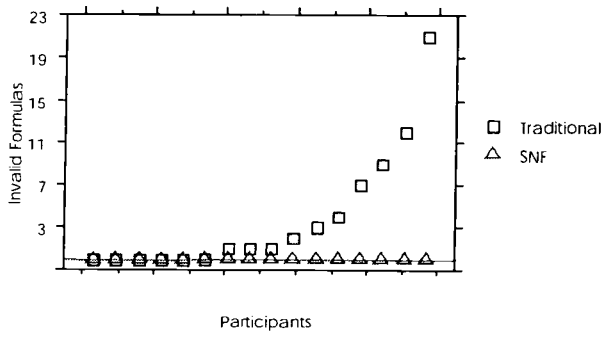


Figure 3.5: No participants in the SNF group entered any syntactically invalid formulas even temporarily while modifying the Crabs spreadsheet, whereas participants in the traditional group entered an average of 3.82 invalid formulas (some of which were eventually corrected).

As the figures show, the differences were very large. Our analysis of the data showed that these differences were statistically significant. Significantly more participants in the traditional group entered invalid formulas along the way (Fisher's Exact test, $p = .0002$) and formulas of participants in the SNF group were significantly more correct (Mann-Whitney test, $p < .0001$).

3.4 RQ2: Speed Results

Our second research question considers whether participants using Model SNF would modify temporal relationships faster than those using the traditional approach. Using the electronic transcripts of the on-line activity of our participants, we measured the elapsed time between the first interaction with the spreadsheet (showing a formula, editing a formula, changing the speed of a cell, etc.) and the last action. See Figure 3.6. The participants had 15 minutes to modify the crabs spreadsheet, but some of the participants in the traditional group were allowed to use a couple more minutes if they felt they were close to the solution. However, in our statistics scores in excess of 15 minutes were scored as 15 minutes. Thus, since this allowance favored the traditional group's correctness, and did not affect the statistics on speed, it does not invalidate either the correctness or the speed results.

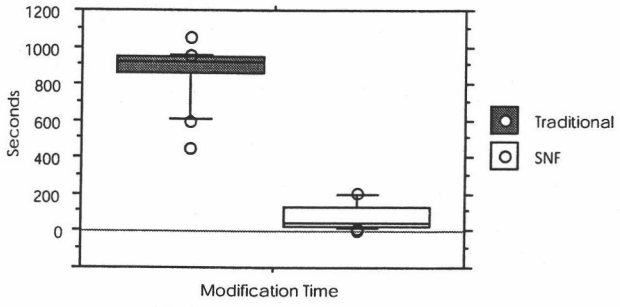


Figure 3.6: Time to perform modification of Crabs spreadsheet. The box plots are composed of 5 horizontal lines at the 10^{th}, 25^{th}, 50^{th}, 75^{th} and 90^{th} percentiles; values above the 90^{th} or below the 10^{th} percentiles are plotted separately. Note that all the SNF participants were faster than *any* participant in the traditional group.

Participants using Model SNF modified the Crabs spreadsheet significantly faster than participants using the traditional approach (Mann-Whitney test, $p < .0001$). In fact, the *slowest* participant using the SNF approach spent only three minutes and forty three seconds getting all the modifications right, whereas the *fastest* participant using the traditional approach spent seven minutes and twenty eight seconds and did not get any modifications right.

As another measure of speed, we also counted the number of edits the participants performed. As Figure 3.7 shows, participants in the SNF group performed fewer edits than participants using the traditional approach (Mann-Whitney test, $p<.0001$).

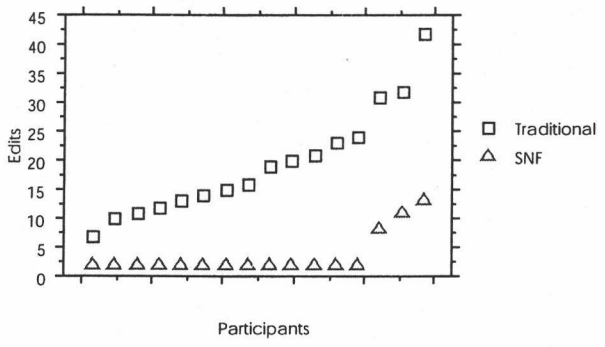


Figure 3.7: Number of edits performed to modify the Crabs spreadsheet (SNF mean=3.62, traditional mean=19.37).

3.5 RQ3: Understandability Results

Our last research question focuses on ability to understand programs written by someone else. To measure understandability we used the Drug Company Deliveries problem described in Section 3.2. The test consisted of 12 questions and we awarded 1 point for each correct answer. Thus, the maximum score possible in the test was 12. Participants in the SNF group scored significantly higher in this task (Mann-Whitney test, p=.0003). See Figure 3.8.

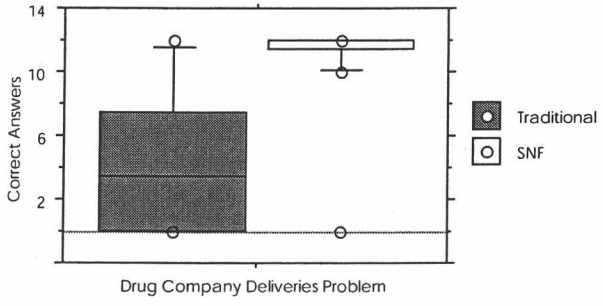
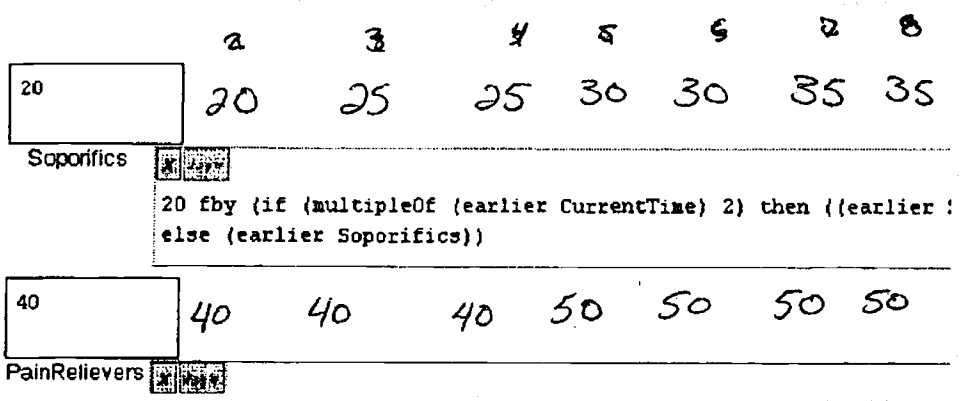
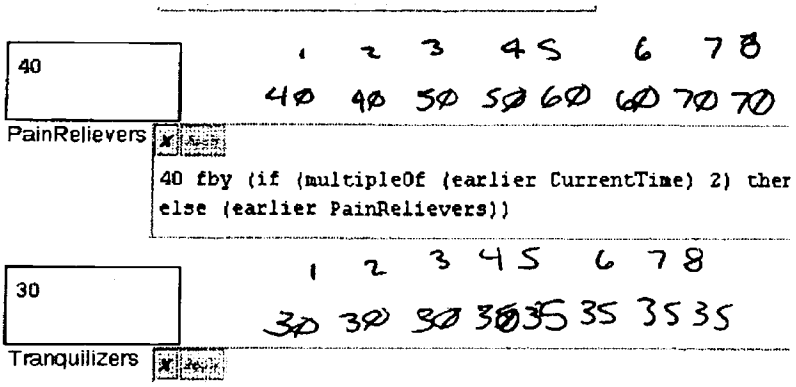


Figure 3.8: Number of correct answers in the Drug Company Deliveries problem (SNF mean = 11, traditional mean = 4.31).

Two participants using the traditional approach got all of the answers correct in this task. We were very interested to see that both of these participants supported their answers with a drawing very similar to the temporal window used by the participants in the SNF group. Recall that the traditional group participants were never shown temporal windows. This suggests that the grid-based temporal window is a close match to the way some end users choose to think about this kind of problem. See Figure 3.9.



(a)



(b)

Figure 3.9: The two participants in the traditional group who correctly answered all the questions in the second task supported their answers with drawings that closely resemble the window used by our visual models. (a) and (b) are scanned-in drawings these two participants wrote on their test forms.

3.6 Discussion

In the design of the time models the Cognitive Dimensions of premature commitment[2] and viscosity[3] were revealed as two drawbacks of Model SNF [Cao 2000]. This was not evidenced by our empirical results; rather, these issues generally do not have time to arise in a short controlled experiment. In particular, in the crab spreadsheet problem, the participants will not have to deal with future modifications, and thus are not faced with premature commitment or with viscosity. This seeming difference in usability results simply demonstrates the fact that no single evaluation technique can alone reveal complete usability information.

As mentioned earlier, the traditional approach is used in several languages. It is representative of the one-dimensional, stream-based approaches and therefore presents a standard for comparison. Still, we tried to simplify the knowledge needed to use the traditional approach so that results would depend on the approach itself, not extraneous factors. Toward this end, we added an `isMultipleOf` operator, so that the participants did not need to use the `mod` operator, which our pilot study revealed to be unfamiliar to some non-programmers.

In the traditional approach it is necessary to use conditions (`if` operators) to decide if a given cell should be calculated at the current time. For example, in the Drug Company Deliveries problem (Figure 3.2), the deliveries of `PainRelievers` increase every two months. We need a condition to determine if the value should increase at the current time (in this case every step of time represents a month). In the electronic transcripts we observed that these conditional expressions were a particular source for erroneous formulas. Since the participants had no programming experience, it is not surprising that they had trouble with `if` expressions, even though they were given tutorial instructions and practice, and even though they had a reference card throughout the experiment explaining the spreadsheet operators.

---

[2] Constraints on the order of doing things force the user to make a decision before the proper information is available [Green and Petre 1996].

[3] Resistance to change: the cost of making small changes [Green and Petre 1996].

Given the strong understandability results for the SNF approach, a natural question that arises is whether some particular portion of the approach was key to the results. Although we do not have a rigorous answer to this question, we have the opinions of the participants. In the post-task questionnaires we asked the SNF participants "Did the temporal window help you check that your modifications were correct?" Thirteen out of the sixteen SNF participants answered "Yes" to this question, and the other three participants answered "No." Of the participants who answered affirmatively, eleven said in the unstructured "Why?" follow-up question that they liked the fact that they were able to visualize how the value of the cells changed over time.

Chapter 4: The Second Empirical Study

The temporal window provides an explicit visual display of progression patterns and dependencies of values over time. Our previous empirical study demonstrated how the time devices are actually usable by end users modifying programs, and help them in comprehending programs. We became interested in learning whether this explicit information could also aid with the "oracle problem". Recall that an oracle is the mechanism which checks the correctness of the output of a program. The oracle problem is that, although much of the testing literature describes methodologies which are predicated on both the theoretical and practical availability of an oracle, in many cases such an oracle is pragmatically unattainable [Weyuker 1982].

It occurred to us that arrows pointing out patterns of values over time might help users ascertain correctness in some spreadsheets. In Forms/3, even predating the WYSIWYT methodology, the user has been able to display arrows that show the dataflow relationships among cells [Yang et al. 1997]. For example, the arrows in Figure 4.1 show that the cell price is used to calculate the cells Tax and Total.

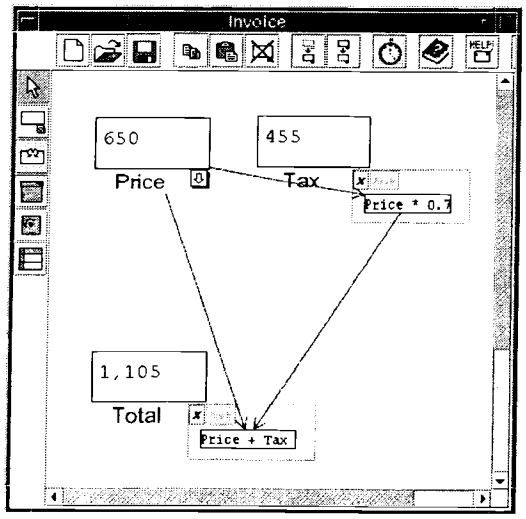


Figure 4.1: Arrows in the spreadsheet show the dataflow relationships.

In a "regular" Forms/3 spreadsheet the arrows show which cells (and therefore which values) are used to calculate the value of a cell. However, this may not be the case on a spreadsheet involving time. For example, in the Fibonacci example (Figure 2.1) the cell fib references itself one and two steps before the current time. Even a self-pointing arrow would not be very useful, since the user could not see the actual values used for calculation at a given time. The user would need to go back two steps in time and then return to the original time to verify the result (we will refer to this action as traveling in time).

To solve this problem, we extended the use of arrows to the temporal window. Since several values are displayed at the same time, the user can show the arrows for a cell at a given time and see which values are used without traveling in time. Figure 4.2 shows the arrows for the cell fib at time 6; the user can see that the cell is taking the two previous values to calculate the current value. The question is whether the explicit display of these temporal relationships patterns and values as demonstrated by the example, can be helpful to the user's oracle task.

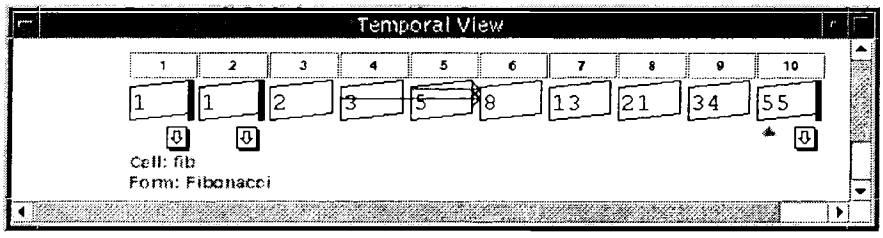


Figure 4.2: The arrows show that cell `fib` takes the two previous values to calculate the current one.

To answer this question we designed and performed another empirical study.

The objectives of this study were to investigate the following research questions:

RQ1: Can end users judge the correctness of a value more accurately using the visual temporal display than without using the visual temporal display?

RQ2: Can end users judge the correctness of a value faster using the visual temporal display than without using the visual temporal display?

RQ3: Do end users prefer to use the temporal window to judge the correctness of a value rather than travel in time?

To answer these research questions we asked the participants to answer "Yes/No" questions about the correctness of values on three different spreadsheets, using dialogs like the one in Figure 4.3.

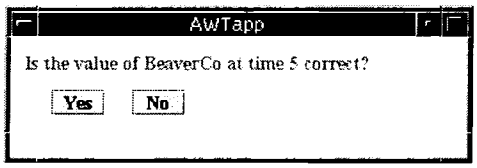


Figure 4.3: Users answered Yes/No questions about the correctness of different values on time.

We asked the participants to judge the correctness of values on each of three different spreadsheets. On one spreadsheet, they used the temporal window, on another they did not, and on the third we left the choice to them.

The study was conducted one participant at a time at a workstation. The lecturer led a hands-on tutorial and practice session that included an introduction to the Forms/3 environment, how to travel in time and how to interact with the question dialogs. All of the participants received the same time for instruction and practice.

Following the tutorial, the participants read the description of a first spreadsheet and answered six questions about the correctness of the same number of values. Next, the participants read the description of a second spreadsheet and answered another six questions about the correctness of six values. Finally, the participants read the description of the Crabs spreadsheet and answered four questions about the correctness of four

values. In this spreadsheet, the participants had the option to use or not use the temporal window. We will describe these spreadsheets in Section 4.2. Since we wanted the participants to focus only on judging the output of the spreadsheets, we did not allow the participants to access the formulas on the spreadsheets.

The study was counter-balanced with regard to the use of the temporal window; that is, half of the participants used the temporal window to answer the questions for the Stocks spreadsheet, and the other half used the temporal window to answer the questions for the HealthPro spreadsheet. To eliminate a learning advantage, half of the participants answered the questions of the Stocks spreadsheet first, and the other half answered the questions of the HealthPro spreadsheet first. All the participants answered the questions about the Crabs spreadsheet last; since the only measure we were interested in from that spreadsheet was the preference of using or not the temporal window, the learning advantage was not an issue.

The data produced by the study included background and post-task questionnaires, and electronic transcripts of participants' on-line activities.

4.1 The Participants

Thirty-two students participated in the experiment, drawn from a computer literacy course. During their literacy course they covered basic use of spreadsheets. Thus, all of the participants had previous experience using the Excel spreadsheet system; two also had previous experience using Quattro. None of them had used Forms/3 before. In our background questionnaires, three of the participants claimed to be frequent users of spreadsheets; however, since all participants used both techniques this do not invalidate the results. Table 4.1 shows the previous spreadsheet experience of the participants.

Excel	Quattro	Forms/3
32	2	0

Table 4.1: Previous spreadsheet experience of the participants.

4.2 The Spreadsheets

The Stocks Spreadsheet. This spreadsheet was used as a tutorial in the experiment described in Section 3.2. The spreadsheet models how the price of three different stocks (`Micron`, `BeaverCo` and `DuckInc`) increases over time. See Figure 4.4. Participants answered six questions about the correctness of the value of a different stock at six steps in time. A characteristic of this spreadsheet is that references to other cells are to one previous step in time.

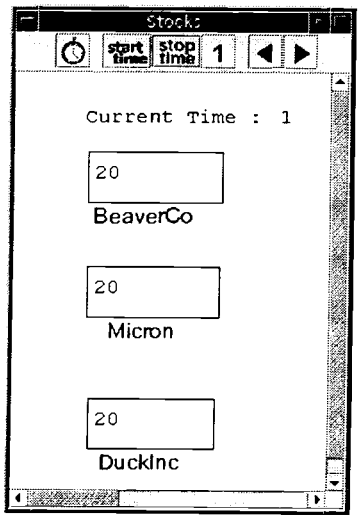


Figure 4.4: The Stocks spreadsheet as seen by the participants. See Appendix A for a picture of this spreadsheet with the formulas used to create it.

The Drug Company Deliveries Problem. This spreadsheet is similar to the one used in the previous experiment. The spreadsheet represents the deliveries of three different pharmaceutical products (`Tranquilizers`, `PainRelievers` and `Soporifics` in Figure 4.5). In this spreadsheet, cell references are made to values four steps before the current time or five times ahead of the current time. Participants answered six questions about the correctness of the value of the delivery of a different product at six steps in time.

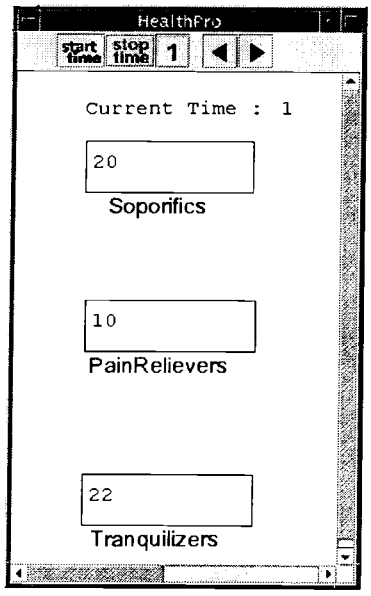


Figure 4.5: The HealthPro spreadsheet as seen by the participants. See Appendix A for a picture of this spreadsheet with the formulas used to create it.

The Crabs Spreadsheet. This spreadsheet is also similar to the Crabs spreadsheet used in our previous experiment. We slightly changed the description, since the problem was formulated without using different speeds, but the idea of the spreadsheet is the same: three crabs (`slowCrab`, `normalCrab` and `fastCrab` in Figure 4.6) should move at three different paces. Although we asked the participants to answer four questions about the correctness of the position of the crabs at four different steps, our

main interest in this stage of the experiment was to investigate if the participants preferred to use the temporal window or to travel in time to verify the correctness of a value.

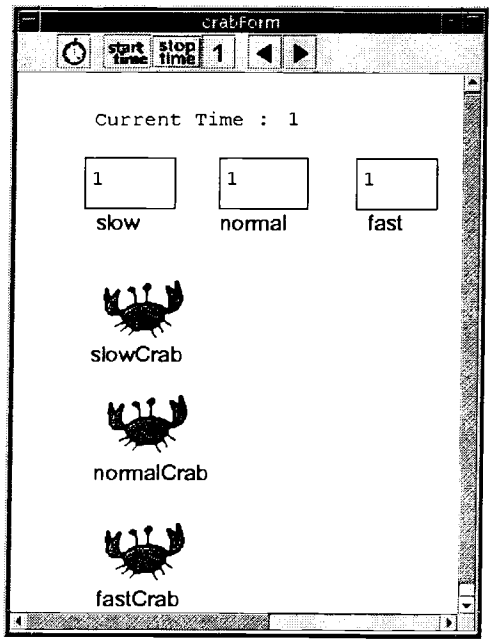


Figure 4.6: The Crabs spreadsheet as seen by the participants. See Appendix A for a picture of this spreadsheet with the formulas used to create it.

4.3 RQ1: Accuracy Results

Our first research question considers whether using the temporal window to judge the correctness of a value improved the accuracy with which the participants judged the values. To measure the accuracy of the judgements we scored how many questions about the correctness of values the participants answered right on a given spreadsheet. Thus, the maximum possible score for each spreadsheet was six.

Our analysis of the data using non-parametric tests (Mann-Whitney) showed that there was no statistical difference in the accuracy with which participants judged the

correctness of the values on the Stocks spreadsheet (Mann-Whitney test, p = 0.0935). Figure 4.7 summarizes the participants' performance.

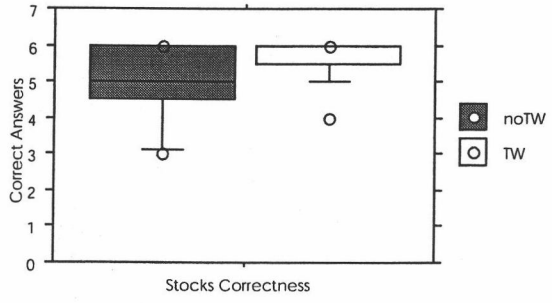


Figure 4.7: There was no statistical difference in the accuracy between the two groups (no temporal window mean = 5.06, temporal window mean = 5.69).

For the HealthPro spreadsheet, our analysis showed that there was a significant difference in the accuracy with which the participants judged the correctness of the values (Mann-Whitney test, p = 0.0021). Participants who used the temporal window judged the correctness of the values more accurately than participants who did not use the temporal window did. See Figure 4.8. In fact, none of the participants who did not use the temporal window answered all six questions right, whereas six participants using the temporal window answered all questions correctly.

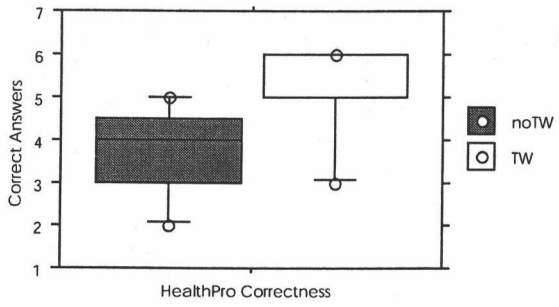


Figure 4.8: Participants who used the temporal window judged more accurately the correctness of values on the HealthPro spreadsheet than participants who did not use the temporal window did (no temporal window mean = 3.81, temporal window mean = 5.06).

4.4 RQ2: Speed Results

Our second research question considers whether participants using the temporal window to judge the correctness of values judge faster than participants not using the temporal window. To measure the speed of a judgement we measured the time elapsed between the moment that the question appears on the screen and the user clicks on the Yes or No buttons.

Similar to the correctness results, our analysis of the data using non-parametric tests showed no statistical difference between participants who used the temporal window to judge the correctness of values on the Stocks spreadsheet and participants who did not use it (Mann-Whitney test, $p = 0.7774$). See Figure 4.9.

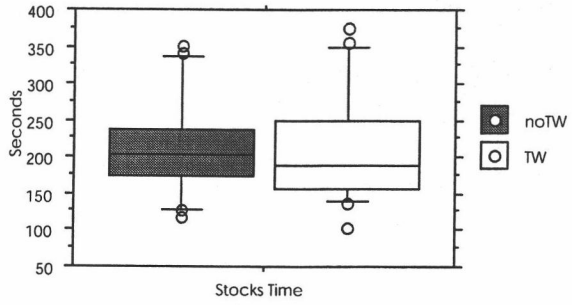


Figure 4.9: There was no statistical difference in the speed of the judgements between the two groups (no temporal window mean = 212.69, temporal window mean = 210.75).

Also similar to the correctness results, our analysis showed a significant difference in the participants' performance on the HealthPro spreadsheet. Participants who used the temporal window judged the values significantly faster than participants who did not use it (Mann-Whitney test, $p = 0.0018$). See Figure 4.10.

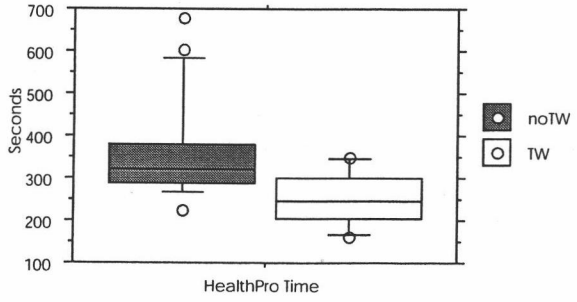


Figure 4.10: Participants who used the temporal window judged the correctness of values faster than participants who did not use the temporal window (no temporal window mean = 357.31, temporal window mean = 251.62).

4.5 RQ3: Preference Results

Our last research question considers whether participants prefer to use the temporal window to judge the correctness of values rather than travel in time.

To measure their preference, at the beginning of this stage of the experiment the temporal window was not visible. If participants wanted to use the temporal window they explicitly had to make it visible (we taught the participants how to make the temporal window visible during the tutorial).

The electronic transcripts showed that 23 participants preferred to use the temporal window, whereas only 9 participants preferred to travel in time. That is, 72% of the participants used the temporal window and only 28% did not use it. See Figure 4.11. Our statistical analysis showed that there is a significant difference on the preference shown by the users ($\chi^2 = 6.125$, $p = 0.0133$).

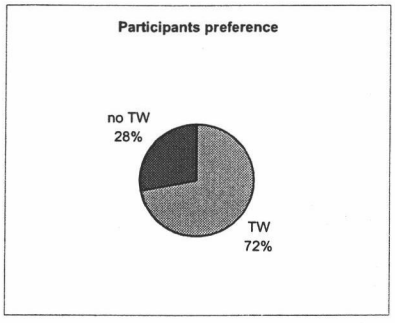


Figure 4.11: Participants preferred to use the temporal window to judge the correctness of a value (72%) rather than travel in time (28%)

4.6 Discussion

Given the results showing a significant difference on the performance of the participants only in one of the spreadsheets using the temporal window, a question that may arise is "why a significant difference only in one of the spreadsheets and not both?".

In the Stocks spreadsheet the references to other cells are made to only one step before in time, whereas in the HealthPro spreadsheet references to other cells are made to four steps before in time or five steps in the future, as shown in Figure 4.12. Thus, users who did not use the temporal window on the HealthPro spreadsheet spent some time going back and forth in time. In addition, users who did not use the temporal window could not visualize all of the values at the same time; perhaps this introduces some degree of complexity for the users, since they have to remember the values referenced. In the post-task questionnaire, 31 participants (97%) mentioned that the temporal window was useful because they could see all the values at the same time, and 23 participants (74%) mentioned that the arrows in the temporal window were useful.

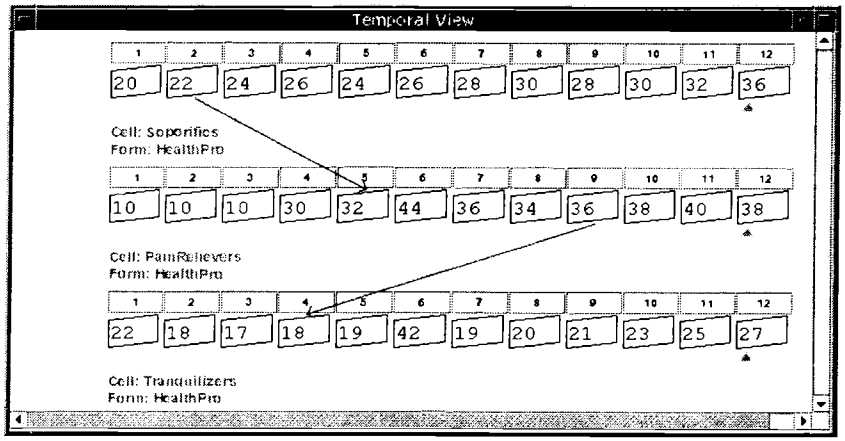


Figure 4.12: An example of the temporal window that participants used. The arrows show how cells reference values five steps in the future (Tranquilizers) or four steps before (PainRelievers).

In both spreadsheets we noticed a savings of time tied with the use of the temporal window to answer the correctness questions. In Figure 4.13 we can see how the participants' efficiency improvement increased with the span of temporal references in formulas. Recall that formulas in the Stocks spreadsheet refer back only one step, but formulas in the HealthPro spreadsheet refer four steps back and five steps ahead. We

wonder whether the time savings will keep increasing in such a drastic way as the span of temporal references in formulas increase.

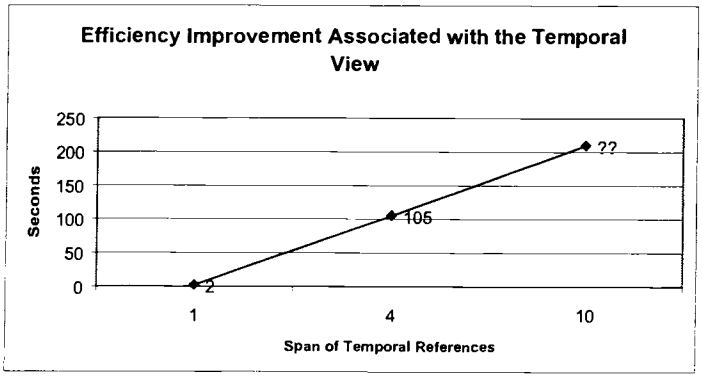


Figure 4.13: The time saved to answer the questions by using the temporal window increased with the span of temporal references.

In the same way, the participants' accuracy improvement increased with the span of temporal references in formulas. See Figure 4.14.

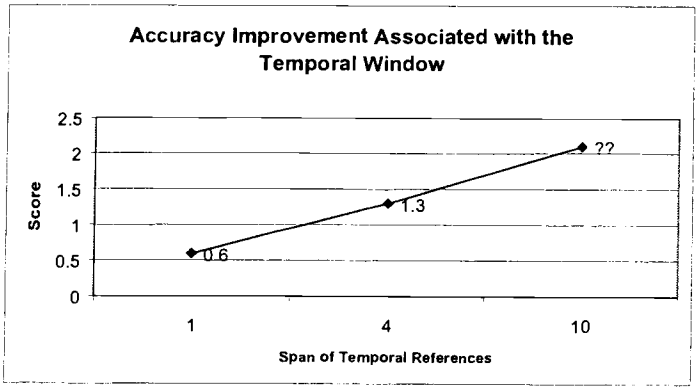


Figure 4.14: The accuracy improvement increased with the span of temporal references.

In summary, the results showed that the temporal window did help end users to improve their performance as oracles, in terms of both accuracy and efficiency, and that the more complex the temporal relationships, the more benefit the temporal window delivered. Further empirical studies are needed to explore how much the temporal window can impact the end users' performance as oracles as the span of temporal references in formulas increase.

Chapter 5: Implementation

Besides the implementation of the current prototype of model k*x, another contribution of this thesis is the implementation of a new graphical user interface for the temporal view in JavaForms.

To help future students in the Forms/3 group to understand the new temporal view interface, this chapter explains its structure and the changes performed to implement model k*x and the arrows in the temporal window.

5.1 Structure of the Temporal View

Before this thesis, the user interface for the temporal view was implemented using Garnet. Since the Garnet parts of the Forms/3 implementation were then discontinued, a new user interface for the temporal view in Java was created.

In JavaForms the temporal view is composed of a GuiTVWindow, TempViewObj, GuiTVNode, GuiTVFmlaWindow, GuiTVTimePointer and RegionButton. See Figure 5.1.

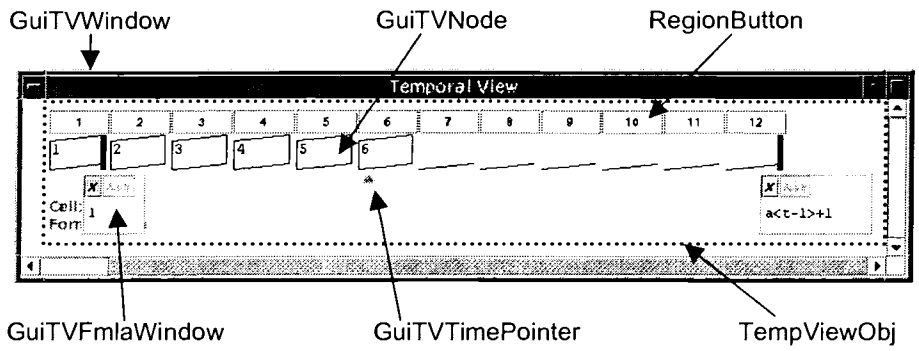


Figure 5.1: The objects that compose the temporal view of a cell.

The GuiTVWindow represents the window of the temporal view. The TempViewObj is a view of the GuiRO (it implements the interface GuiROView), and represents the temporal vector of one cell in the temporal window. The TempViewObj has a reference to the DataRO of the cell it represents, which is the only way the temporal view knows about the data structure in the Engine side. The TempViewObj also has references to the GuiTVNodes that are used to display the different values that a cell has over time, to the GuiTVFmlaWindows (which display the formula of a cell at a given region), and to the GuiTVTimePointer (this is the small arrow that points to the GuiTVNode at the current time). The RegionButtons are used to create a new region in the formula of a cell. Figure 5.2 shows the structure of the temporal window.

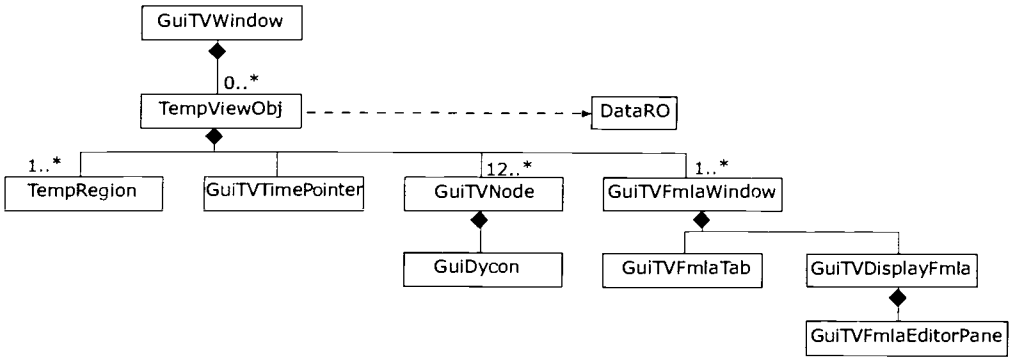


Figure 5.2: This UML Class Diagram shows the static structure of the Temporal Window. The diamonds denote the composition of the objects and the dashed arrow shows how the TempViewObj depends on the DataRO of the cell it represents.

Notice in the Class Diagram and Figure 5.1 how the temporal view is initially composed of 12 GuiTVNodes (unless the user shows the temporal window when the current time is greater than 12). As the user travels forward in time, more GuiTVNodes will be added to the temporal window to allow the display of more values.

5.2 Implementation of Model k*x

Before this thesis, model k*x was not supported in Forms/3. Recall from Section 2.3.3 that model k*x allows the user to specify the speed of a cell to be k times faster/slower than some other cell x. To support this model we introduced a new kind of dependency in the WAWTable in the engine side: a temporal speed dependency. We also added two data members to an RO: TVSpeed-reference-cell and TVSpeed-reference-times. The first is a pointer to the RO that the cell will use as a reference to calculate its speed, the second is an integer that indicates how many times faster or slower the speed of the cell will be (a negative integer means that the cell is slower than the referenced cell and a positive integer means that the cell is faster). For example, if cell a is two times faster than cell b, then TVSpeed-reference-cell points to cell b, and TVSpeed-reference times has a value of 2.

We added some code to the function gui-cell-set-speed in gui.lisp to accommodate model k*x. First, we add the new speed relationship to the WAWTable, then we communicate this new relationship and the new TVInterval to the GUI. Finally, the function update-timeDependentCells in time3.lisp will be called to collect the cells affected by this speed change from the WAWTable and update them if necessary.

Figure 5.3 shows the sequence diagram of the action sequence when the user changes the speed of a cell in the model k*x.

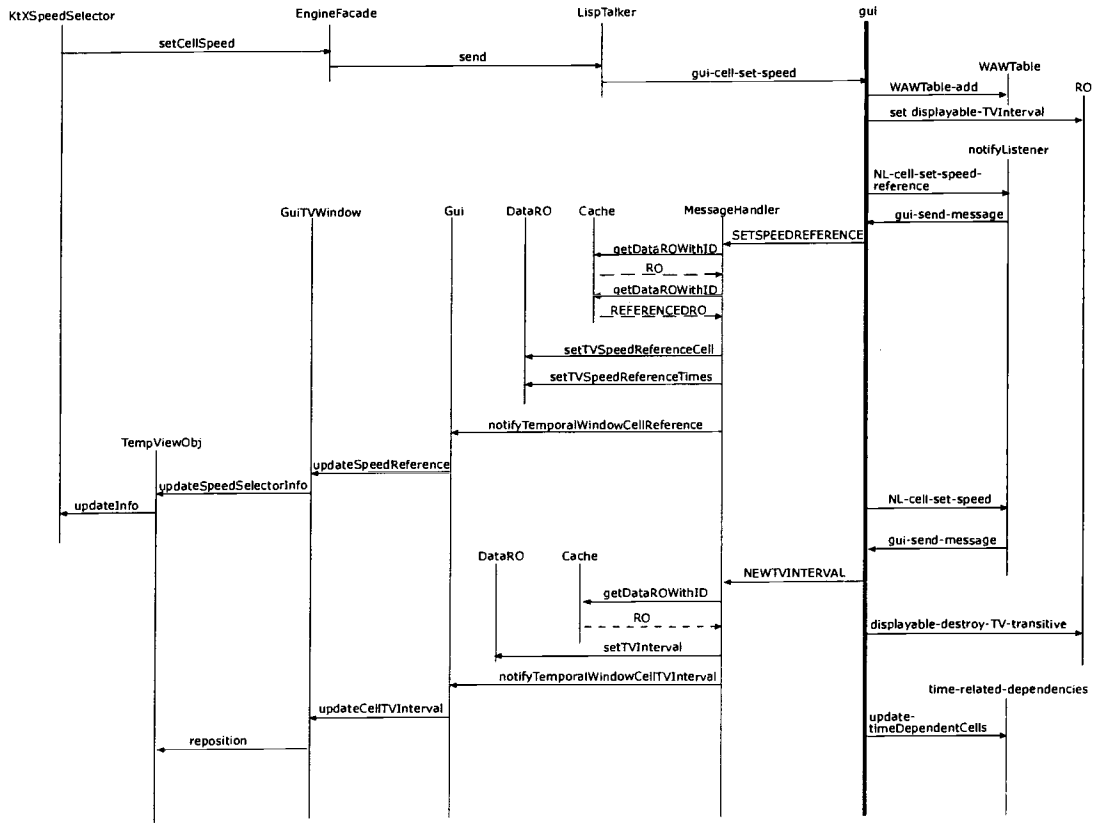


Figure 5.3: This UML sequence diagram shows the action sequence when a user changes the speed on model k*x. The dashed arrows show the cells returned by the Cache. The thicker line in the gui shows the division between objects in the Engine side and the GUI side.

5.3 The Arrows in the Temporal Window.

As we mentioned earlier, we implemented arrows to display dataflow relationships in the temporal window. To display these arrows we created a new class named GuiTVArrow which inherits from class Arrow that is used to paint the arrows in the spreadsheet. We also implemented a new function named gui-time-arrows-info in gui.lisp. This function returns a list of the cellID's that contribute to calculate the value of a cell at a given time and the time at which their values become defined. Notice that the

function returns the time at which the values become defined, not the time at which the cell is referenced. We chose this option to avoid drawing arrows from steps in time not explicitly populated. For example, if cell b references cell a on a previous step of time and cell a is a constant, since the value of a does not need to be recalculated every step of time it will only be displayed a time 1. See Figure 5.4. Drawing an arrow from a GuiTVNode on a time other than time one would come from an empty space and this may cause confusion to the user.

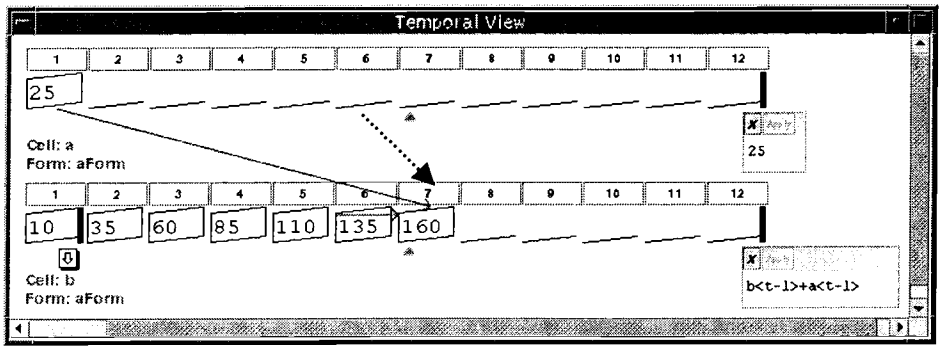


Figure 5.4: The arrows are drawn from the time the value is defined, not the referenced time. This figure is annotated with the dotted arrow to show how an arrow from an empty space may confuse the user.

In Figure 5.5, the sequence diagram shows the action sequence when the user clicks to show the arrows on a given node on the temporal window.

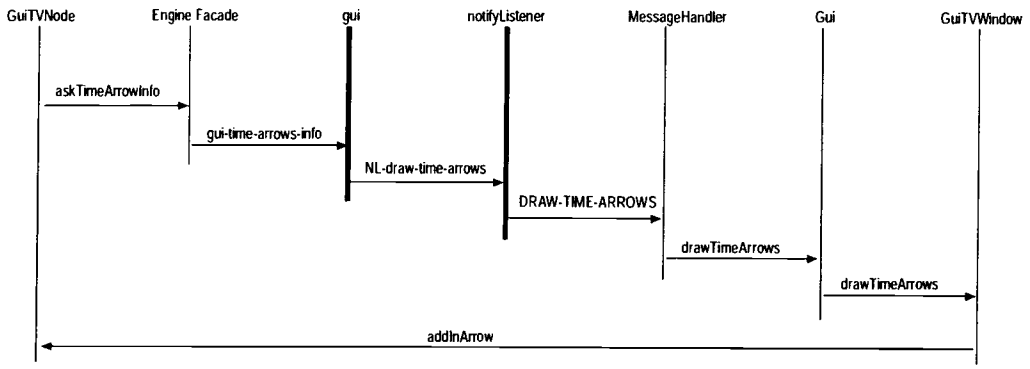


Figure 5.5: This UML diagram shows the action sequence when the user draws the arrows in the temporal window. The two objects with thicker lines (gui and notifyListeners) are objects in the Engine side, and communicate with the Java side via socket messages.

Chapter 6: Threats to Validity and Conclusions

In this chapter we consider some of the threats to the validity of our empirical studies, present the conclusion of this thesis and future work on temporal programming in Forms/3.

6.1 Threats to Validity.

In our experiments we addressed threats to internal validity by choosing a group of participants with similar experience using spreadsheets (novices), by equalizing training time and material covered during the training, and by selecting problems from familiar domains. However, threats to external validity are more difficult to address given the need to control all other factors. For example, novice spreadsheet users may not be representative of the population of spreadsheet users. Similarly, the spreadsheets used in the experiments may not be representative of the population of real-world spreadsheets.

Because the focus of our second experiment was on accuracy and speed of the judgement of values, the users did not have access to the formulas. This may be unrealistic. However, we were interested in the users' oracle ability, and we wanted them to focus on judging the correctness of the values, not analyzing or judging the correctness of the formulas.

In our first experiment, to measure understandability, we used a questionnaire that required the participants to calculate the values of three cells at two different times. We could also have asked the participants to describe the spreadsheet or the functioning of the cells, but then the grading of this questions could have been too subjective. Asking the participants to calculate values by looking at the formulas on a spreadsheet requires an understanding of these, and although the arithmetic ability of the participants may affect their answers, we feel it is a good measure.

The results of the empirical studies may not generalize to the other time models mentioned in Section 2.3, other problems, other populations of users, or longer-term tasks. Further empirical studies are needed to determine these issues.

6.2 Future Work Based on Current Results

In the empirical results of our first experiment, end users with light to moderate spreadsheet experience and no programming experience were able to succeed at both programming and at understanding temporal behaviors in one of the time models of Forms/3 (Model SNF). Further, the results showed that their ability to succeed was significantly greater than in the traditional approach, which most were not able to use successfully. These strong results suggest that the traditional approach is not a reasonable language feature for this population, but that a grid-based model is a viable approach. However, these results were specific to Model SNF. Further empirical investigation is required to determine the level of spreadsheet expertise required for mastery of the k*N and k*x models.

The empirical results of our second experiment showed that end users' performance as oracles, in terms of both accuracy and efficiency, increased with the use of the temporal view. Thus, the use of the temporal view and the arrows to point out patterns of values over time helped users to ascertain the correctness of spreadsheets which define temporal relationships. Further, the results suggest that this improvement in oracle performance may increase with the span of temporal references in formulas. Perhaps these benefits gained from the temporal window will keep increasing with the complexity of the spreadsheet. However, the benefits shown by our results are specific to the base model and to spreadsheets of moderate complexity. Further empirical studies are required to determine the benefits of the temporal view to aid end users to increase their ability as oracles using other time models, like k*N or k*x, and on more complex spreadsheets.

A question that arises from the results of the second experiment is whether the oracle benefits would happen even without formulas explicitly based on time, such as if the temporal relationships shown were completely derived from data dependencies. An example of this could be showing recursive definitions over time, so that the user could see the patterns of how values calculated first affect those calculated later.

Our results show that some time models were actually usable by end users, and that the temporal view can aid them in verifying the correctness of values in a

spreadsheet. However, as we mentioned earlier, issues like viscosity and premature commitment do not arise in a controlled environment. Further empirical studies should include longer term tasks to analyze the extent in which the aspects of the temporal models revealed as flaws by the Cognitive Dimensions could affect the effectivenes and usefulness of the models.

6.3 Conclusions

In summary, our results show that temporal programming can be seamlessly integrated in spreadsheets, allowing the user to reapply the spatial way of programming she already knows. This consistency flattens the learning curve of the language and allows the user to express temporal relationships in an independent way, without introducing extraneous code into the original spreadsheet. Furthermore, our results show that the temporal view used with the time models can aid the user in tasks such as effectively and efficiently judging the correctness of values.

Bibliography

[Ashcroft and Wadge 1985] E. A. Ashcrof and W. Wadge, *Lucid, The Data-Flow Programming Language*. Academic Press, 1985.

[Atwood et al. 1996] J. Atwood, M. Burnett, R. Walpole, E. Wilcox, S. Yang, Steering programs via time travel, *IEEE Symposium of Visual Languages*, Boulder, CO, Pages 4-11, September 1996.

[Brown and Gould 1987] Polly S. Brown and John D. Gould, An experimental study of people creating spreadsheets, *ACM Transactions on Information Systems*, Vol 5, No 3, Pages 258-27, July 1987.

[Burnett et al. 2000] M. Burnett, N. Cao, J. Atwood. Time in Grid-Oriented VPLs: Just Another Dimension?, *IEEE Symposium on Visual Languages*, Seattle, WA, Pages 137-144, September 2000.

[Carlson et al. 1996] P. Carlson, M. Burnett, and J. Cadiz, A seamless integration of algorithm animation into a visual programming language. *ACM Proceedings of International Workshop on Advanced Visual Interfaces*, Gubbio, Italy, Pages 194-202, May 1996.

[Cao 2000] Nanyu Cao. Temporal programming in grid-oriented visual programming languages. Thesis (M.S.), Oregon State University, 2000.

[DeMillo 1991] Richard A. DeMillo, Progress toward automated software testing; *Proceedings of the 13th International Conference on Software Engineering*, Pages 180-183, 1991.

[Du and Wadge 1990] W. Du and W. Wadge, A 3d spreadsheet based on intensional logic, *IEEE Software*, Pages 78-89, May 1990.

[Duisberg 1996] Robert A. Duisberg. Animated graphical interfaces using temporal constraints; *Proceedings of the ACM SIGCHI '86 Conference on Human Factors in Computing Systems*, Boston, MA, Pages 131-136, April 1986.

[Green and Petre 1996] T. Green, M. Petre, Usability analysis of visual programming environments: a 'cognitive dimensions' framework, *Journal of Visual Languages and Computing*, 131-174, June 1996.

[Hibino and Rundersteiner 1997] S. Hibino and E. Rundersteiner, User interface evaluation of a direct manipulation temporal visual query language, *ACM International Multimedia Conference*, Seattle, WA, Pages 99-107, November 1997.

[Kappelman et al. 1993] Leon A. Kappelman, John P. Thompson and Ephraim R. McLean, Converging end-user and corporate computing; *Communications of the ACM*, Vol 36, No 12, Pages 79-92, December 1993.

[McDaniel and Myers 1999] R. McDaniel and B. Myers, Getting more out of programming-by-demonstration, *ACM Conf. Human Factors in Computing Systems*, Pittsburgh, PA, Pages 442-449, May 1999.

[Memon et al. 2000] A. Memon, M. E. Pollack, and M. L. Soffa, Automated Test Oracles for GUIs, *8th InternationalSymposium on the Foundations of Software Engineering*, November 2000.

[Panko 1998] R. R. Panko, What we know about spreadsheet errors. *Journal of End User Computing's*, Pages 15-21, Spring 1998.

[Panko 2000] R. R. Panko, Spreadsheet errors: What We Know. What We Think We Can Do. *Proceedings of the Spreadsheet Risk Symposium*. Greenwich, England, July 2000.

[Peters and Parnas 1994] D. Peters and D.L. Parnas, Generating a Test Oracle from Program Documentation, *Proceedings of the 1994 International Symposium on Software Testing and Analysis (ISSTA)*, Pages 58-65, August 1994.

[Peters and Parnas 1998] D. Peters and D.L. Parnas, Using Test Oracles Generated from Program Documentation, *IEEE Transactions on Software Engineering*, Vol. 24, No. 3, March 1998.

[Richardson et al. 1992] Debra J. Richardson, Stephanie Leif Aha and T. Owen O'Malley; Specification-based test oracles for reactive systems; *Proceedings of the 14th International Conference on Software Engineering*, Pages 105-118, 1992.

[Richardson 1994] Debra J. Richardson; TAOS: Testing with Analysis and Oracle Support, *Proceedings of the 1994 International Symposium on Software Testing and Analysis*, Pages 138-153, 1994.

[Rothermel et. al 1998] G. Rothermel, L. Li, C. DuPuis, and M. Burnett, What You See Is What You Test: A Methodology for Testing Form-Based Visual Programs, *Proceedings of the 20th International Conference on Software Engineering*, Pages 198-207, April 1998.

[Rothermel et. al 2000] K. J. Rothermel, C. R. Cook, M. M. Burnett, J. Schonfeld, T. R. G. Green, and G. Rothermel, WYSIWYT Testing in the Spreadsheet Paradigm: An Empirical Evaluation, *Proceedings of the 22nd International Conference on Software Engineering*, Pages 230-239, June 2000.

[Vogel 1993] Peter A. Vogel, An integrated General Purpose Automated Test Environment. *Proceedings of the International Symposium on Software Testing and Analysis*, New York, NY, Pages 61-69, June 1993.

[Weyuker 1982] E. J. Weyuker, On Testing Non-Testable Programs, *The Computer Journal*, Vol. 25, No. 4, Pages 465-470, 1982

[Wolber 1997] D. Wolber, Pavlov: an interface builder for designing animated interfaces, *ACM Transactions on Computer-Human Interaction*, Vol 4, No 4, Pages 347-386, December 1997

Appendices

Appendix A

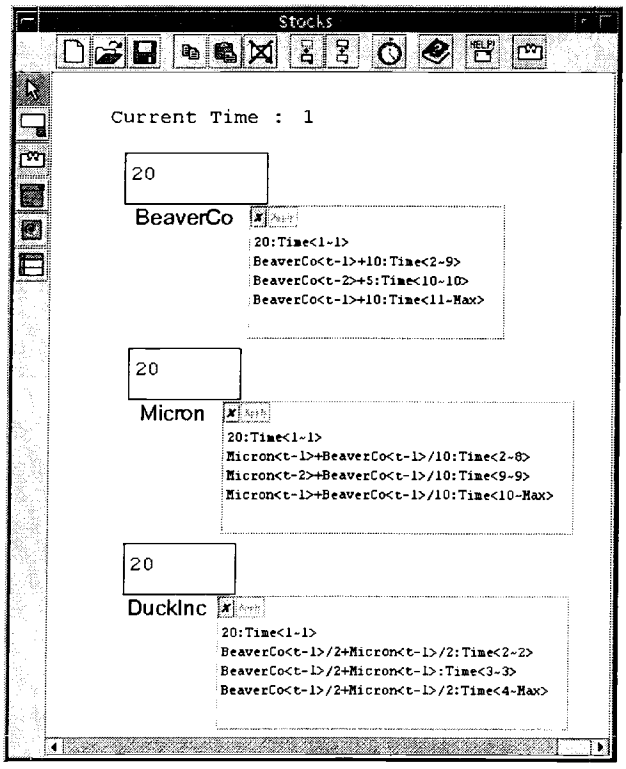


Figure A.1: The Stocks spreadsheet used in the second experiment with the formulas shown.

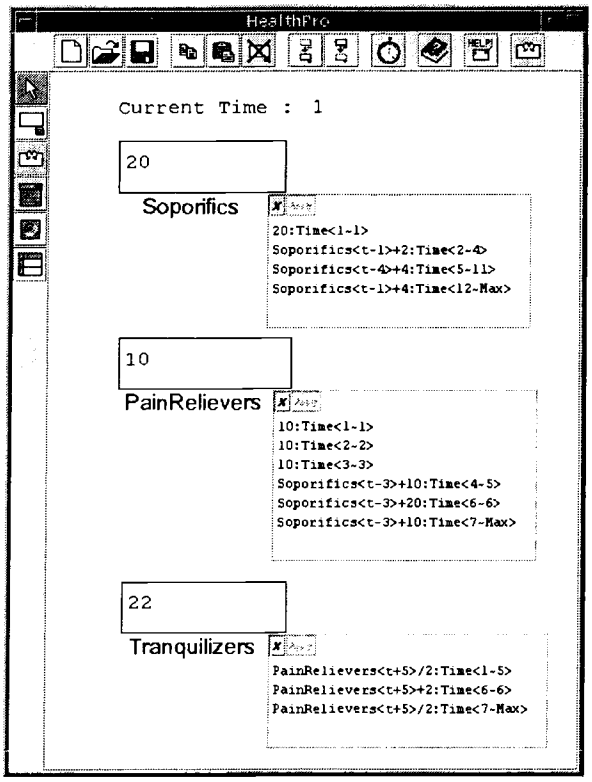


Figure A.2: The HealthPro spreadsheet used in the second experiment with the formulas shown.

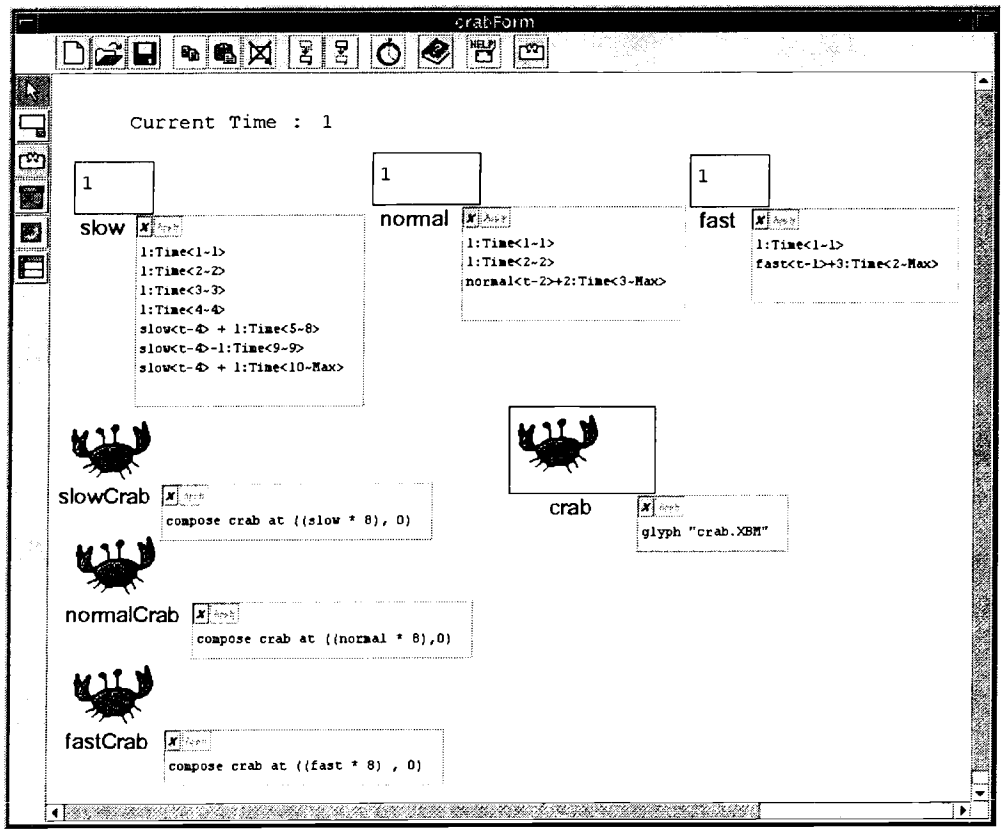


Figure A.3: The Crabs spreadsheet used in the second experiment with the formulas shown.

Appendix B

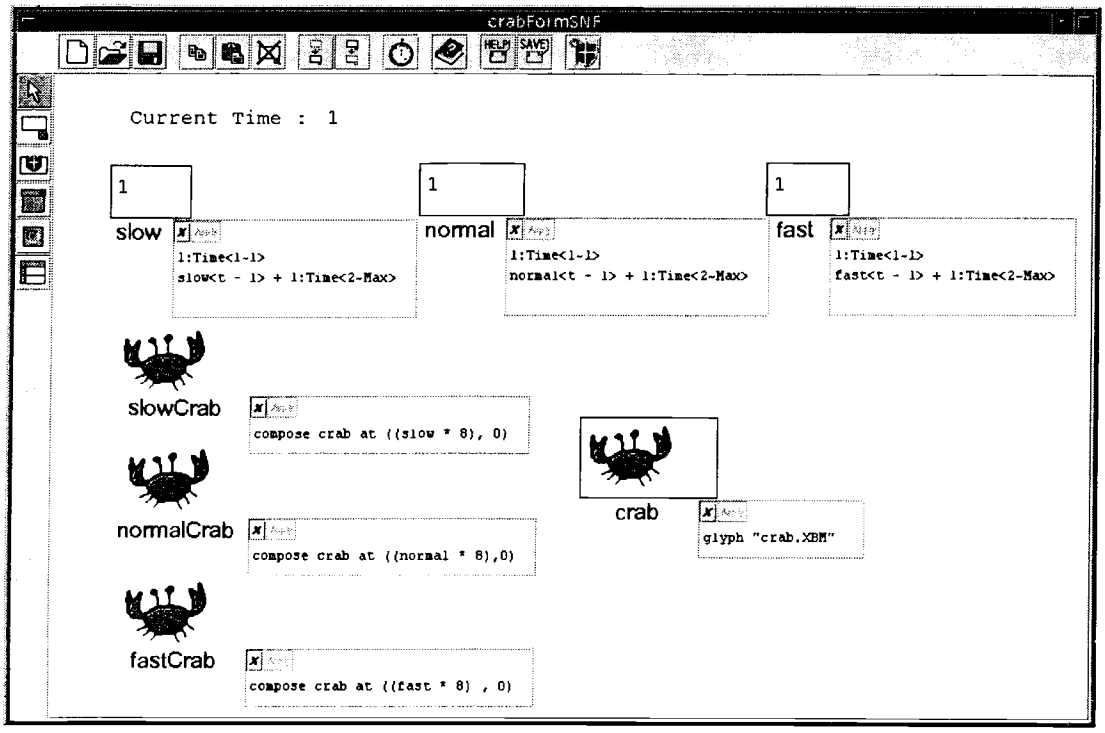


Figure B.1: The Crabs spreadsheet used in the first experiment with the formulas shown.

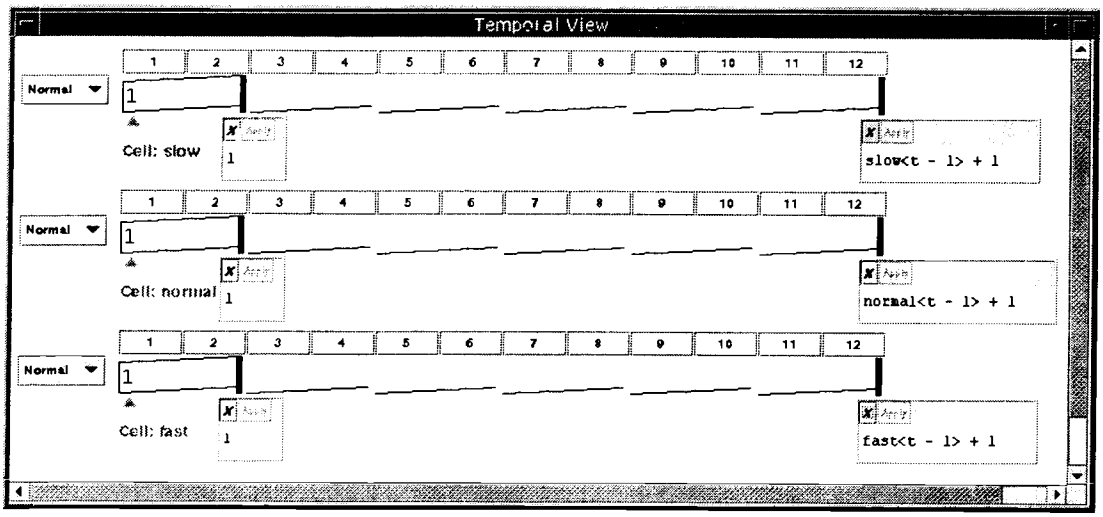


Figure B.2: An example of the temporal view window that participants in the SNF group could use in the first experiment. It shows all the formulas the participants had access to. All cells originally had a Normal speed.

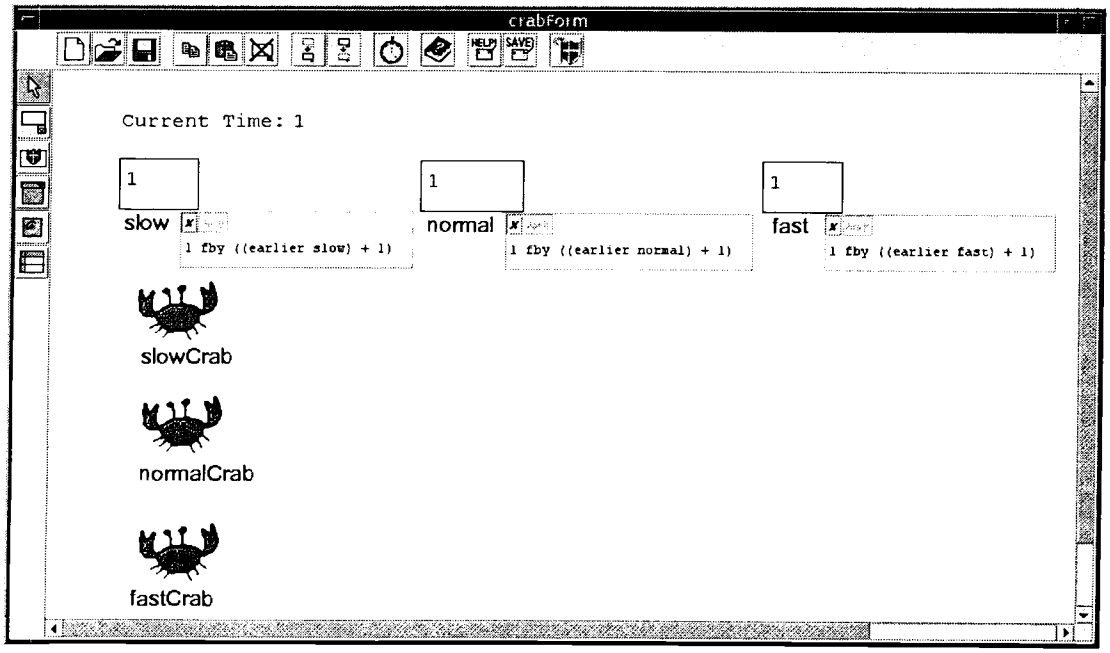


Figure B.3: The Crabs spreadsheet as seen by the participants in the traditional group in the first experiment. It shows all the formulas the participants had access to.