PERGAMON

Computers & Education 39 (2002) 1–18

COMPUTERS & EDUCATION

www.elsevier.com/locate/compedu

# A visual programming approach for teaching cognitive modelling

Trevor D. Collins[a,*], Pat Fung[b]

[a]*Knowledge Media Institute, The Open University, Milton Keynes MK7 6AA, UK*
[b]*Institute of Educational Technology, The Open University, Milton Keynes MK7 6AA, UK*

Received 29 January 2001; accepted 4 December 2001

**Abstract**

This paper describes an investigation into the use of a visual programming language to teach computer-based modelling to undergraduate cognitive psychology students. Four sets of evaluation studies were carried out. The findings of these theoretical and empirical evaluations are related to the design principles that informed the language and the context in which it was examined. The educational benefits of gaining some practical experience of cognitive modelling where highlighted in these studies, as was the importance of introducing the visual language within a sound teaching framework. The comments of the students and tutors regarding the use of the Hank visual programming language to teach cognitive modelling indicate that Hank avoids some of the syntactic problems associated with textual programming languages, it can be used to illustrate the flow of control during a program's execution, and it is intuitive and easy to use 

*Keywords:* Applications in subject areas; Programming and programming languages; Teaching/learning strategies

## 1. Introduction

This paper provides an overview of an 18-month project that set out to explore the use of a visual programming language for teaching cognitive modelling to undergraduate psychology students. Hank is a visual programming language designed to support the development of computer-based models of cognitive theories by students with little or no previous computer programming experience (Mulholland & Watt, 1998). Computer modelling is an investigative approach used in a variety of fields including sociology, psychology, chemistry and physics. By

* Corresponding author. Tel.: +44-1908-655731; fax: +44-1908-653169.
*E-mail address:* t.d.collins@open.ac.uk (T.D. Collins).

0360-1315/02/$ - see front matter 
PII: S0360-1315(01)00087-2

successive iterations through the processes of model design, development and testing, an accurate physical model of an abstract theory can be produced. The behaviour of the completed model can then be used to reason about the validity of the theory it embodies. In this way, computer modelling provides a means for exploring, analysing and critiquing theories.

Within cognitive psychology, computer modelling is an important methodology used to explore theories, validate experimental findings and inform further research (Eysenck & Keane, 1995). However, providing undergraduate psychology students with a practical training in cognitive modelling is a non-trivial task. Not only must the student learn how to represent knowledge and the processes involved in creating, retrieving and applying knowledge, they must also learn how to program a computer. Computer programming with textual languages, such as Lisp or Prolog, has a steep learning curve associated with it that can deter students from taking an active role in modelling. In order to alleviate this problem, a visual programming language called Hank was developed at The Open University's Knowledge Media Institute (Mulholland & Watt, 1998).

Visual programming languages, like Hank, use visual elements, such as graphics or icons, in addition to text, to support program development. The use of Hank to support psychology students building cognitive models was explored in a series of programming walkthrough evaluations and empirical studies. The findings of the studies, and their implications for both the development of visual programming languages and the use of visual programming languages within an educational context are discussed in this paper. The remainder of this section introduces the research objectives, the context of the studies undertaken and the related work. The following section provides an overview of the Hank visual programming language. Subsequently, the paper describes the methodology used and the findings of the research. The paper concludes with a discussion of the implications of this work.

## 2. Project objectives

The design of the Hank language was informed by existing research undertaken in the areas of end-user programming and software visualization. The objectives of the project were to examine the use of Hank in order to support or refute the design principles that informed it, and, given that Hank was proposed as an easy to learn alternative to traditional text-based programming languages, investigate the suitability of using visual languages in this educational context.

### *2.1. Context of these studies*

All of the empirical studies described in this paper were carried out in 1999 with students from The Open University's third level course on cognitive psychology. The Open University is a distance learning university based in the United Kingdom. The cognitive psychology course is studied part-time, over a 9-month period, for approximately 8-h each week and includes a 1-week full-time residential school.

The students on the course have two opportunities to study cognitive modelling. For their third assignment the students are required to complete an introductory (paper-based) cognitive modelling project. The educational objectives of this project are to introduce the students to the idea of developing models of cognition, and to highlight the problems and benefits of this approach to

studying psychology. Later in the course, as part of a 1-week residential school, the students are given the option to undertake a further advanced (computer-based) cognitive modelling project. This project is intended to further the students' practical experience and extend their existing understanding of cognitive modelling visualization.

### 2.2. Related research

Related research in the areas of end-user programming and software was used to inform the design of the Hank language. The primary sources of direction in this regard are briefly considered here, but a more complete explanation of the design rationale behind Hank is available in Mulholland and Watt (1998).

In his previous work Mulholland (1998) concluded that in order to support people's understanding of computer programs, software visualization has to provide adequate support for the user. Specifically, the user should be able to map between the program code and the visualizations used, review previous events in the execution from any particular point, and be able to predict and test for future events within a program's execution.

Nardi and Miller (1991) argued that spreadsheets offer strong support for the co-operative development of programs. In their study of 11 users from a range of different companies, nearly all of the spreadsheets used were the result of collaborative work by people with different levels of programming and domain expertise. They noted two key characteristics of spreadsheets that facilitated collaboration: layered functionality and tabular structures.

Firstly, spreadsheet functionality can be separated into two distinct programming layers, referred to as the fundamental layer and the advanced layer. The fundamental layer includes the fundamental facilities that the user requires to produce a spreadsheet (i.e. a formula language for computations and the spreadsheet table that provides a means for structuring and presenting data). The advanced layer provides additional functionality that is unnecessary for constructing a basic spreadsheet model but nevertheless is very useful. Examples of facilities from the advanced layer include the following: conditional and iterative control structures, date-time, error trapping functions, graphs and charts, and a user interface toolkit.

Secondly, Nardi and Miller (1991) argued that the spreadsheet table is a strong visual representation that enables users to understand and interpret each other's work. The tabular structure of spreadsheets has also been used in other end-user programming languages in an educational context. Agentsheets (Repenning & Citrin, 1993), and KIDSIM (Smith, Cypher, & Spohrer, 1994) both use a two-dimensional grid layout and graphical rewrite rules to help children write programs. However, although spreadsheet tables are particularly effective at showing large amounts of data, they are less effective at illustrating the formulas underlying the cell values in the table (see Hendry & Green, 1994; Nardi & Miller, 1991).

Spreadsheets are effective at illustrating what a spreadsheet model does, but are poor at illustrating how the model works. Conversely, flowcharts have been found to be good at illustrating how things work but not what they do (Cunniff & Taylor, 1987; Scanlan, 1989; Vessey & Weber, 1986). Comic strip notations offer another means for illustrating the flow of control through a program. In an approach to programming known as programming by demonstration, comic strip notations have been developed to show how a series of actions fit together by presenting them as a set of before and after states (e.g. Lieberman, 1992; Traynor & Williams, 1997). Traynor and

Williams (1997) found that although comic strip notations are valuable for illustrating the linear execution of a program, they are less effective at representing the execution of more complex control flow constructs such as branching and looping.

In summary, spreadsheets offer strong support for collaborative work, tabular structures are a strong visual representation that are commonly used and easily interpreted, flowcharts and comic strip notations explicitly illustrate the flow of control through a program, and software visualization should be used to help the user to map, review and test their programs.

Having introduced the objectives of this work, the context of the studies, and the related research, this section gives a brief overview of the Hank language, its model of execution, and development environment.

## 3. Overview of Hank

Having introduced the objectives of this work, the context of the studies, and the related research, this section gives a brief overview of the Hank language, its model of execution, and development environment. Hank is made up of three basic constructs: fact cards, instruction cards and questions. Fact cards adopt a tabular structure for representing factual knowledge, instruction cards represent procedural knowledge, and questions are used to query fact cards and instruction cards. *Fact cards* are made up of a title on the top row, column label(s) referring to components on the second row, and one or more data rows representing individual facts (see Fig. 1, left).

*Questions* are used to query information by looking for matching fact cards, or instruction cards, in the Hank program. Questions are made up of a title on the top row, column label(s) on the second row, and a single data row on the third row (see Fig. 1, right). Variables, delimited by question marks (e.g. ?Day?), can be used within the data row of a question. When a match is found for a query the result is reported as the Status of the system- successful matches report a status of OK, unsuccessful matches report a status of Fail. In addition to the system's status, the values of any instantiated variables are also reported. The example question shown on the right in Fig. 1 would return the following message: ?Day? = Monday, Status = OK.

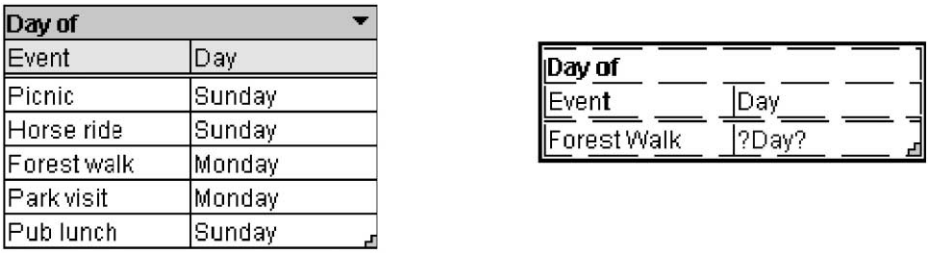


Fig. 1. An example of a Hank fact card (left) and a Hank question (right). The title of the fact card's table (Day of) identifies the relationship being modelled, the column labels (Event and Day) identify the concepts being described, and each data row defines a fact, such as there is a Picnic on Sunday, a Horse ride on Sunday, and a Forest walk on Monday. In the example of a Hank question, shown on the right, the title and column labels (i.e. Day of, Event and Day) are used to identify the card being queried. The value Forest walk and variable ?Day? match with the third row of the Day of fact card shown on the left. The response ?Day? = Monday, Status = OK would be given to this query.

*Instruction cards* are used in Hank to represent procedures (Fig. 2). The top section of an instruction card is referred to as the instruction card's "wish box." The wish box is made up of three rows; the title on the first row, the component column label(s) on the second row, and the variables or values used to pass information into and out of the procedure on the third row. The lower section of the instruction card is referred to as the "process box." The process box describes the steps involved in carrying out the procedure using a set of questions connected together using OK and Fail links. According to the status of each question, the system follows either an OK or a Fail link. If no applicable link is found after a question has been answered, the system (i.e. question processor) halts and returns the value of any variables used in the original question as well as the final status.

The following explicit set of house rules are used in Hank to answer any queries:

1. Ask the question: look to see if there is a match with a fact card or instruction card in the database. This involves matching the title of the card, the column labels, and one of the data rows. The data rows are tried in order from top to bottom.
2. Match the variables: if a match was found and the question contained one or more variables, make a note of the values that they have taken on.
3. Give a reply: if a match was found, then the reply (i.e. the Status) is "Status = OK." The answer should also state any variables that have matched with values. If a match was not found then the reply is "Status = Fail" and the variables need not be reported.

The three programming constructs described above are used to write program on paper, the above set of house rules are used to follow how those programs would execute.

The computer environment for Hank includes: a graphical environment in which the students can write Hank programs, an automated question processor that follows the house rules in order to execute programs, and a bi-directional tracer for illustrating the steps of a program's execution. Three windows are available under the "Window" menu for switching between each of these

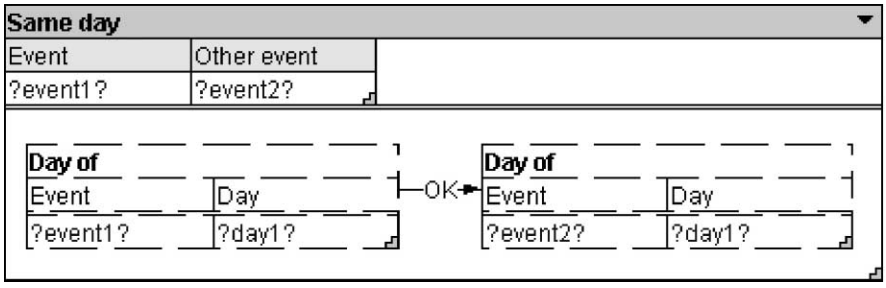


Fig. 2. An example of a Hank instruction card. An instruction card is made up of two parts—a "wish box" (at the top) and a "process box" (at the bottom). In the *wish box* the title of the table (Same day) identifies the relationship being modelled, the column labels (Event and Other event) identify the concepts being described, and the variables under the column labels (?event1? and ?event2?) are used to pass information in and out of the instruction card. The *process box* contains a set of instructions i.e. linked queries. Starting at the top left hand corner the questions are asked in sequence, each time a question is answered the out-going link with the same status value as the current question (i.e. OK or Fail) is followed to the next question.

three tasks; the ''Program Window,'' ''Control Panel'' and ''Workspace.'' Fig. 3 shows a screen shot containing an example of each window.

By drawing on the use of simple visual representations, such as tables to represent data in fact cards and directed links to illustrate the control flow of the program within instruction cards, the language itself is intended to be easier for psychology students to learn and use than more formal textual languages, such as Prolog or Lisp. The Hank modelling environment can be downloaded from the following WWW address: http://kmi.open.ac.uk/projects/hank

## 4. Methodology

Four separate studies were undertaken. The first study used the programming walkthrough methodology as described in (Lewis & Rieman, 1993) to examine the course materials prior to the course starting in February 1999. The second study used a postal questionnaire to survey the students' opinions of the introductory modelling project undertaken in April. The third study

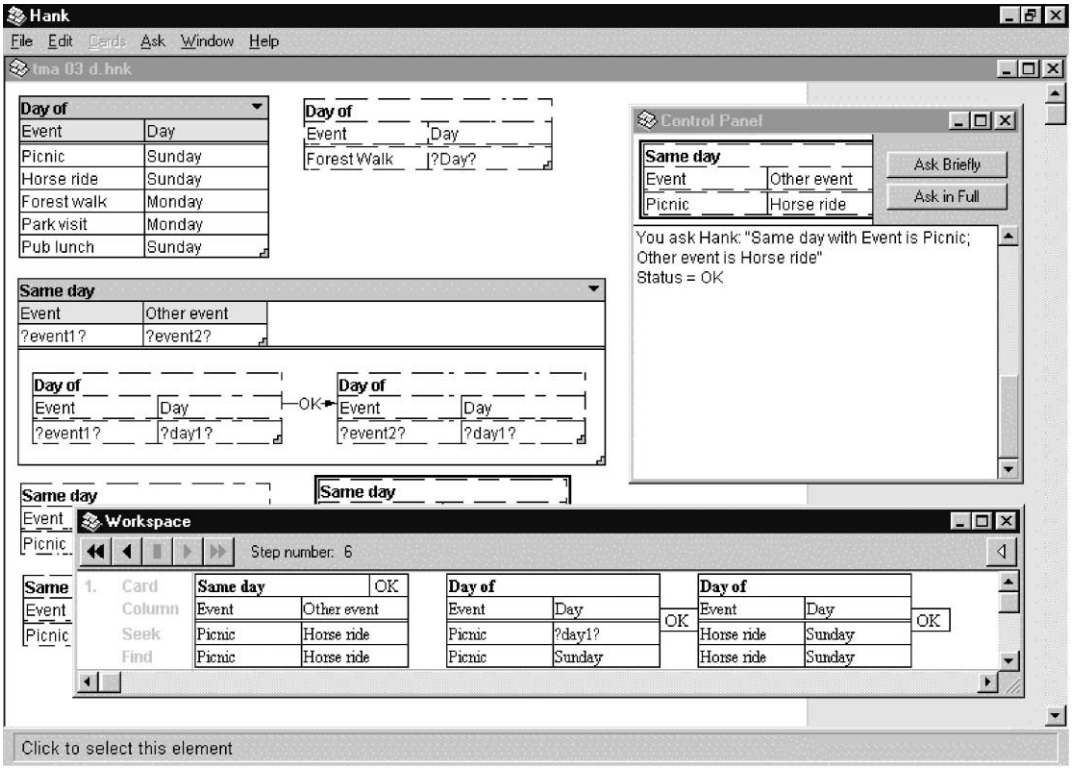


Fig. 3. The Hank modelling environment showing an example program. Three windows are displayed in this view: the program window (background left), control panel (foreground right) and workspace (foreground bottom). The program window is used to draw the fact cards, questions and instruction cards that make up a program, the control panel is used to ask a selected question in the program window, and the workspace window is used to show step by step traces of the matching process used by Hank to answer a question.

involved a second student questionnaire survey to gather the students' opinions on the advanced modelling project, that was carried out at residential school during the months of July and August. The fourth and final study, that was carried out after the course had ended, surveyed the opinions of the course tutors regarding the cognitive modelling aspects of the course and the suitability of the Hank language. Details of the methodology used in these four studies are given in the following four subsections.

### 4.1. Programming walkthroughs

In order to understand the context of use for the Hank language, a theoretical evaluation of the support materials was carried out prior to the course starting. This used the programming walkthrough methodology to evaluate the course materials, practice exercises and assignment tasks. The programming walkthrough evaluation method is a task centred approach used to predict user difficulties (Lewis & Rieman, 1993). The complete task to be carried out by the user is analysed step by step, in an attempt to identify any possible sources of error in the interface and the supporting materials, such as ambiguities, inconsistencies, and insufficient or poorly structured guidance.

### 4.2. Survey of students undertaking an introductory project

To collect the students' opinions regarding the introductory cognitive modelling project a questionnaire was posted to a third of the 1500 students registered for the course (184 completed questionnaires were returned). The questionnaire asked a combination of closed and open-ended questions regarding the following:

- the student's perceived improvement in their understanding of the theory they modelled;
- the student's normal usage of computers;
- the level of support provided by the project booklets and tutorial sessions;
- how easy the students found it to build a model using Hank; and
- whether or not the students found the house rules, describing how to execute a program, helpful.
- In this way it was intended that an overall quantitative summary of the students' experiences could be made, along with detailed qualitative analysis of the good and bad features they encountered.

### 4.3. Survey of students undertaking an advanced project

To survey the opinions of the students, concerning their use of the Hank modelling environment in groups as part of their residential school project, a questionnaire was made available to the students at the end of each cognitive modelling project session. During the 1999 presentation of the course, a total of 22 cognitive modelling sessions were run as part of the residential school programme. Ninety-one completed questionnaires were received from an approximate course maximum of 550 students. The questionnaire itself asked a combination of closed and open-ended questions regarding the following:

- the student's self-perceived improvement in understanding cognitive theories;
- the extent to which their understanding had changed of how cognitive modelling is used in psychology;
- whether the tutorial sessions supporting the project had been helpful;
- how easy the student had found it to build a model;
- whether the student had found the programming environment helpful; and
- how easy the student had found it to work in a group to program their model.

### *4.4. Course tutor survey*

Open University students are assigned an associate lecturer whom they can contact for advice and support. Associate lecturers, usually referred to as tutors, are responsible for providing monthly (face-to-face) tutorials for their students and for marking their students' assignments (referred to as 'tutor marked assessments', TMAs). During the 1999 presentation, there were 71 tutors employed for the course. This study was designed to capture the opinions of the tutors regarding the use of Hank on the course. A copy of a questionnaire was posted to all 71 tutors and 25 completed questionnaires were returned. The questionnaire itself was structured around the following:

- the tutor's opinions on the effectiveness of the introductory support materials;
- how easy the tutor found it to assess their students' understanding from the introductory assignment;
- the tutor's opinions on the effectiveness of the advanced cognitive modelling project carried out at residential school; and
- the effectiveness of the Hank visual programming language for tutoring cognitive modelling.

## 5. Findings of the research

The findings of the above four studies is given here. A more detailed description of the findings from the programming walkthrough evaluation can be found in (Collins & Fung, 1999). Further details of the three questionnaire surveys are available in the following survey reports: (Collins & Fung, 2000a, 2000b, 2000c).

### *5.1. Programming walkthroughs*

For the introductory cognitive modelling project booklet (a written text document made available to all students as support material), several recommendations were made regarding the layout and terminology used in the examples. Some concerns were raised over the sufficiency of the explanations given for computer programming concepts. Specifically, the explanation of variables and the matching process used. The explanation of variables was deliberately made at a non-technical beginners level, however, it was difficult to predict whether the explanation given would be sufficient for the requirements of the project. The matching process in the introductory

exercises used only the first match found for a query, the consequences of multiple matches within a fact card were not addressed in the project booklet.

Regarding the students' exercises and assignments, a concern was raised in relation to the level of support the students should receive for producing their paper-based models. Could the students be expected to draw out their own fact cards, instruction cards and questions, or should a blank set of cards be supplied for the students to complete? Also, it was recommended that in addition to completing the modelling exercises and assignment questions, the students should be required to provide a written explanation of their model. Otherwise, in some of the questions used, the students could guess the answer without fully understanding the programming involved.

With respect to the advanced modelling project carried out at residential school, three potential difficulties were noted. These referred to the additional Hank primitives required for some of the models, the level of computer feedback provided by the Hank modelling environment, and the degree of guidance given to the students at residential school. Firstly, in the introductory project the students use the "Ask" primitive to ask a question, this returns the first match to a given query. However, there are a number of additional primitives, which the students are not introduced to, for example the "Ask All" primitive that asks for all the solutions to a query, not just the first match. Some of these primitives may be used in the residential school projects. The residential school tutors must ensure that a sufficient set of additional Hank primitives is introduced to the students at an appropriate time.

Secondly, at the time of evaluation few error messages were available and the cause of failure was not reported in the computer-based modelling environment. Hank may fail to find a matching fact card or instruction card to a query, or it may fail to find a specific data row within a fact card. For the purposes of debugging a program it was felt that feedback on the cause of failure would be beneficial to the user. Thirdly, it was proposed that the students should be provided with a detailed description of the stages involved in building their model. The anticipated stages should be ordered such that the students are faced with evenly paced steps through the modelling process, that build on the students' previous knowledge, steep or uneven jumps in complexity for which the students would be unprepared should be avoided.

As a result of the feedback given from the programming walkthrough evaluations, several changes were made to the course materials. The layout and terminology used in the project booklet were improved. No changes were made as a result of the concerns raised regarding the sufficiency of the description of variables or the matching process. The authors of the course materials were cautious to ensure that the introductory material remained sufficiently straightforward for students learning at a distance and suggested that a more complete description of variables and the matching process could be provided for the students at residential school.

The decision was also made to ask the students to draw their own cards rather than providing a set of blank templates. Although drawing the cards on paper was seen as a repetitive task for the students, there are few alternatives. By providing the students with a blank set of cards to complete, a fair proportion of the modelling task is already given. For example: the number of columns in a Hank fact card or question card would indicate the number of items being represented and the number of questions inside a Hank instruction card would indicate the number of steps in a process. The students were asked to explain their models in the introductory assignment.

It was decided that the introduction of the additional Hank primitives used in some of the residential school projects should be left to the discretion of the residential school tutors, to gauge the progress of their students and introduce only the required primitives. The level of feedback given in the Hank modelling environment was extended to include detailed messages describing the cause of failure.

### 5.2. Survey of students undertaking an introductory project

The introductory project involved the development of a simple model of schema theory. The majority of respondents (126 out of 184, 68.5%) felt that the introductory cognitive modelling assignment had helped their understanding of schema theory. Furthermore, 80% of the students who commented were able to identify specific aspects of schema theory about which they felt their understanding had improved as a result of completing the project.

It was a matter of interest as to whether or not the students' prior computer usage or programming experience would make it easier for them to understand the principles of cognitive modelling and/or use the Hank visual programming language. Eighty five percent of the respondents (156 out of 184) said they used a computer. One hundred and fifty-five of the students said they had used some form of word processing package, 97 students reported using a spreadsheet package, 81 students had used a database package, and 19 students had used some form of programming package.

The students reported frequency of use for word processing and programming packages were both found to correlate with their self-perceived ability to create instruction cards ($r_s = 0.182$, $P = 0.025$, and $r_s = 0.193$, $P = 0.024$, respectively) and interpret Hank questions ($r_s = 0.164$, $P = 0.042$ and $r_s = 0.227$, $P = 0.007$). Interestingly, those students with little or no programming experience were more likely to say they found the assignment improved their understanding of schema theory than those that were familiar with programming ($r_s = -0.184$, $P = 0.03$). This may be the result of the students' prior expectations of programming gained though the use of traditional text languages, such as Prolog. Whether or not the students had previously used a computer did not affect their ability to use Hank on paper.

The cognitive modelling project and assignment booklets were used to introduce the topic of cognitive modelling. The survey data indicated that both of the booklets were considered, in the majority of cases, to be helpful, well written and easy to follow. However, some problems were experienced on the final modelling questions in the introductory assignment.

Less than half of the students attended a tutorial on cognitive modelling, which is made available to students before completing their written assignment (90 of the 184 respondents, 48.9%). Of those that did attend the tutorial, 61 students (67.8% of tutorial attendees) said it had helped their understanding of cognitive modelling and 52 (57.8%) said it had helped their understanding of the use of the Hank language. There were no significant differences in the responses from the students who had attended a tutorial and those who had not, in the following four areas: how easy the students found it to use fact cards, how easy they found it to create Hank question cards, how easy they found it to interpret Hank question cards, and in how helpful they found the house rules. However, there was a significant difference between the responses for the students' reported ease of using instruction cards (Mann Whitney, $z = 2.073$, $P = 0.038$). In other words, those that had attended a tutorial on cognitive modelling were more likely to say they found it easier to use instruction cards than those who had not.

As illustrated in Fig. 4, the majority of students found fact cards and instruction cards easy to use. Similarly, as illustrated in Fig. 5, creating Hank questions and following the steps involved in answering Hank questions, were also considered to be fairly easy tasks by the majority of the students. Unsurprisingly, those respondents who said they found it easy to create Hank questions also tended to say that they found it easy to follow the steps involved in answering Hank questions ($r_s = 0.784$, $P < 0.001$). Of the concerns noted, the most common difficulty experienced was that of choosing the best knowledge representation. Other frequently noted concerns related to the time and effort involved in drawing the cards, and the students' confidence in their ability to develop their own models.

The house rules (i.e. the instructions used to process Hank questions) were widely used. Seventy-seven percent of the respondents (141 out of 184) said they did refer back to the house rules when building their own models. It was of interest that this group of students were significantly more likely than those students that had not used the house rules to say that their understanding of schema theory had improved as a result of the cognitive modelling project (Chi-squared, $X^2 = 8.306$, df = 1, $P = 0.004$). There were also significant differences in how easy these

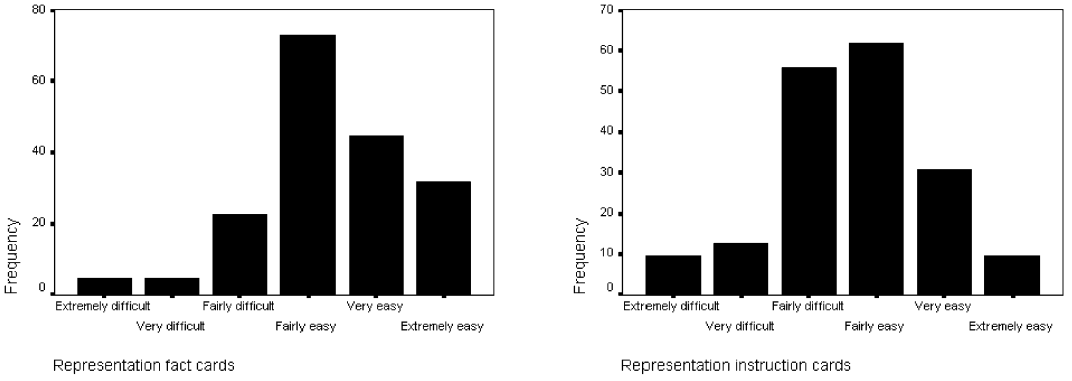


Fig. 4. Two bar charts showing the students' ratings of how easy they found it to represent knowledge using fact cards (left) and instruction cards (right).

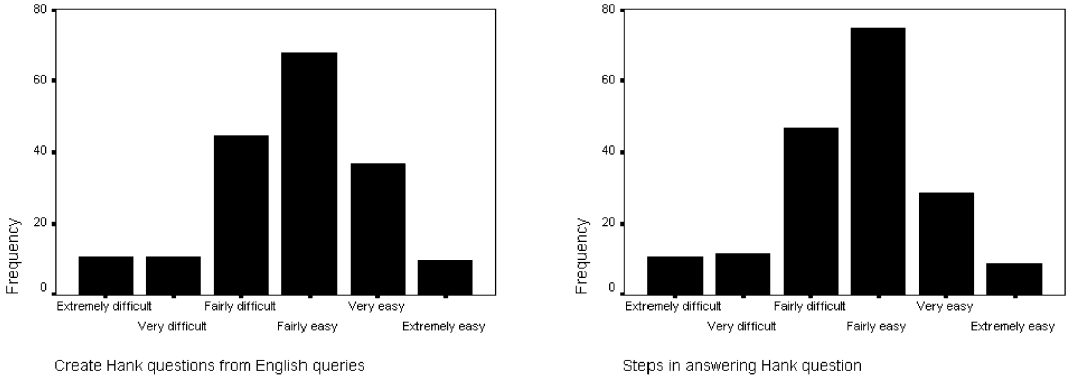


Fig. 5. Two bar charts showing the students' ratings of how easy they found it to created a Hank question for a given query in English (left) and to understand the steps involved in answering a Hank question for a given model (right).

two groups of students found the following operations: to represent knowledge using fact cards (Mann Whitney, $z = 2.958$, $P = 0.003$), to create Hank questions for a given query in English ($z = 2.302$, $P = 0.021$), and to understand the steps involved in answering a Hank question for a given model ($z = 2.706$, $P = 0.007$).

### 5.3. Survey of students undertaking an advanced project

In the second student survey, all 87 students that reported using a computer had some experience of word processing, 61 students had used a spreadsheet package, 49 students had used a database package, and 14 students had used some form of programming package. There were no significant correlations between the students' frequency of computer use and their perceived improvement in their understanding of the theory they modelled, the change in their understanding of the use of cognitive modelling in psychology, or how easy they found it to build their models using Hank. However, as detailed in Table 1, there were significant positive correlations between the students' reported usage of different types of computer package and their reported ease of building models with Hank and how helpful they said they found the Workspace view.

These findings indicate that the students' frequency of use, particularly of spreadsheet and database packages, is related to their perception of the Hank programming language and how easy they find it to build models using the Hank modelling environment.

Four existing theories taken from cognitive psychology were suggested at residential school for the students to model: conceptual hierarchies (Collins & Quillian, 1969), production systems (Young & O'Shea, 1983; Brown & Burton, 1986), means ends problem solving (Simon, 1986), and semantic primitives (Shank, 1972). The majority of respondents chose to model the conceptual hierarchies theory modelled by 42 students, production systems were modelled by 34 students, means ends problem solving by 12 students, and semantic primitives by two students. Only one student modelled an alternative theory, namely a theory of face recognition.

Fifty-three students felt that their understanding of the theory they had modelled improved a lot, 25 students felt there had been some improvement in their understanding, and 12 students felt that their understanding had improved a little (one respondent gave no response to this question). The comments made by respondents, regarding which aspects of cognitive theory modelling had helped them understand, broadly fell into three categories: those identifying aspects of the theory (35 comments), those regarding the use of cognitive modelling (28 comments), and those regarding

Table 1
The positive correlations found between the students' reported frequency of using word processing, spreadsheet, database and programming packages, and their ease of building models using Hank and how helpful they found the Workspace view

| Frequency of computer package use | Ease of building model using Hank | How helpful found Workspace view |
|---|---|---|
| Word Processing packages | $r_s = 0.017$, $P = 0.881$ | $r_s = 0.266$, $P = 0.041$ |
| Spreadsheet packages | $r_s = 0.299$, $P = 0.008$ | $r_s = 0.326$, $P = 0.004$ |
| Database packages | $r_s = 0.319$, $P = 0.004$ | $r_s = 0.213$, $P = 0.062$ |
| Programming packages | $r_s = 0.276$, $P = 0.015$ | $r_s = 0.029$, $P = 0.803$ |

the Hank programming language (17 comments). Only six comments fell outside of these three categories. An example from each comment category is as follows:

"Helped me gain an insight into the search process also started thinking about processes involved in false sentences" [7]
"It helps to explore in more depth the theory and to critically analyse it." [61]
"I understand queries/instruction cards now." [65]

On a four-point scale, 34 of the 91 questionnaire respondents felt that their understanding of how cognitive modelling is used in psychology had changed a lot as a result of their computer based project. Thirty-six students felt that their understanding had changed to some extent and 15 felt that their understanding had changed a little. Five of the students reported no change in their understanding.

When asked to describe the ways in which their understanding of how cognitive modelling is used in psychology had changed, 46 of the 74 comments made were with regard to an improved appreciation of the use of cognitive modelling through personal experience. Some illustrative examples are as follows:

"More flexible approach to CM to try out ideas/more creativity. There are many ways to model the same process—just as humans use different strategies, etc." [3]
"It has improved the distinction between theory and programming, also shown how to focus on the theory and using modelling as a tool." [13]
"It helped see how it is applied rather than simply an abstract area of psychology." [61]
"I can see its relevance where before I couldn't." [91]

Other categories formed from the comments concerned an improved appreciation of the distinction between cognitive modelling and artificial intelligence (three comments). Three also commented on an improved understanding of the theory and three on an appreciation of cognitive modelling as a means for analysing a theory. Four students commented on feeling that there had been a general improvement in their appreciation of cognitive modelling.

The students were asked to rate how easy overall they had found using the computer version of Hank when building their models. Three students rated it as extremely difficult, four rated it as very difficult, 48 as fairly difficult, 26 as fairly easy, and four as very easy (six respondents gave no response). Although the majority of respondents rated building models in Hank as a fairly difficult task, the specific difficulties reported using fact cards, instruction cards and questions were relatively few (six comments, 38 comments and 15 comments, respectively). Several common categories were evident in the comments received, such as generally OK (i.e. no problems), problems with the Hank language (i.e. non-specific difficulties), problems with unfamiliar Hank primitives, problems with the interface, representation problems, and problems with the terminology used.

Over half of the respondents when asked how helpful they found the control panel and workspace windows rated them as very helpful. Regarding the control panel, 42 respondents said they had found it to be very helpful, 28 felt it was quite helpful, eight felt that it had helped a little, and three respondents considered it to be of no help. When asked about the workspace window, 51 students reported it to be very helpful, 25 felt it was quite helpful, five felt it had helped a little, and four of the students considered it to be of no help.

Sixty-five percent of the respondents said they had found it easy to work with Hank in a group. Six students said that it had been extremely easy, 25 said it had been very easy, 28 students had found it fairly easy, 22 found it fairly difficult, five felt it had been very difficult, and five felt group working had been extremely difficult. When asked to describe in their own words what, if anything, they had found difficult about working in a group, out of the 66 students who commented, 20 said they had no difficulties working in a group.

The majority of the difficulties reported regarding working in a group did not refer to the Hank visual programming language or environment. Rather, the difficulties commented upon were primarily to do with the social dynamics of the students' groups. Of the 46 respondents that did describe difficulties that they experienced with working in a group, 15 said they had experienced problems due to the variation in the abilities of their group's members. Five said only one person was able to use the computer at a time, four said they had been frustrated with other members of their group, three reported problems with group dynamics, and three had had difficulties because one individual hogged the computer. Three reported that one individual in their group tended to lag behind, three said they found it difficult to make group decisions, and three said they had problems because their group was too large. Two respondents said they had problems seeing the screen and five gave other individual comments.

### 5.4. Course tutor survey

Overall, the majority of the 25 course tutors that responded felt that the project booklet provided a good introduction to cognitive modelling, even though several of them said they had difficulties explaining the card format of the Hank language and the matching process employed in their tutorial sessions. With regard to improving the booklet, three tutors said they felt the document was fine as it was, four tutors felt more examples and exercises would improve the booklets, and two tutors requested blank modelling cards. Two tutors felt the booklet should be revised to reflect the learning objectives of the project, three felt that the course should return to using Prolog.

The majority of the tutors felt that the assignment questions got progressively more difficult. They preferred the report assignment format over the essay format used in previous years, and believed the marking guidelines provided were sufficient. However, about half of the respondents said they had difficulties marking the introductory assignment and/or providing the students with appropriate feedback. Attributing marks between the multiple components of each question was the most commonly noted difficulty with marking, and identifying and explaining the students' programming errors was the most commonly noted problem regarding student feedback. The tutors' suggestions for improving the assignment included clarifying the written instructions, changing the assignment questions, and preventing the students from spending too much of their time drawing Hank cards.

The residential school projects were in general considered to provide a good extension to the students' understanding of cognitive modelling. Although the comments made regarding the residential school projects were quite diverse, several tutors suggested providing a broader range of projects as a potential improvement for the residential school. Almost half of the respondents said they found marking the cognitive modelling reports more difficult than other residential school project reports. The two most common reasons put forward for these difficulties were that

the tutors found it difficult to understand the students' models and that the students did not explain their models sufficiently.

The Hank language was generally perceived as being good for enabling the students to build their own models. Although nearly half of the respondents identified misconceptions within their student group, only two misconceptions were noted by more than one tutor. These indicated a misunderstanding of variables and a belief that the language can only be used to model schema theory. Twenty-two of the 25 respondents had previously tutored the course using the language Prolog. Of those 22, two rated Hank as extremely easy compared to Prolog, two rated it as very easy compared to Prolog, nine as easy, four as difficult, one as very difficult, and one as extremely difficult (three gave no response). The most commonly noted advantages of using Hank were that, through its visual representation it avoids some of the syntactic problems of Prolog, it is more user-friendly and intuitive, and allows the students to see the flow of control. The main disadvantages of using Hank were with regard to understanding the students' models, the effort involved in drawing the cards in the introductory assignment and learning the terminology used to describe the language. When asked to suggest improvements to the Hank language, four tutors suggested improvements should be made to the explanation of the language rather than the language itself.

## 6. Discussion

The objectives of the research were to examine the use of Hank in order to support or refute the design principles that informed it, and investigate the suitability of using visual languages in an educational context. The implications for visual language design and the use of visual languages are discussed here.

As noted in the course tutor survey the advantages of using Hank on a cognitive modelling course for undergraduate psychology students are that it shows the control flow of the program, it avoids the syntactic problems associated with text languages, and it is intuitive and easy to use. Both of the cognitive modelling projects were considered by the students to improve their understanding of the theories they modelled, and the practical experience gained on the students' second project strengthened their understanding of cognitive modelling and its role in psychology. The card format of the language was picked up quickly by the students and used effectively to produce accurate models of cognitive theories. The majority of the problems the students encountered while developing their models were to do with their representation of the theory, rather than the programming of the computer.

The house rules, which explicitly state the execution process of the language, provided a significant support for modelling on paper and enabled the students to predict and verify the behaviour of their model when using the computer environment. Interpreting the behaviour of their model was further supported by the workspace window, which provided an explicit comic strip illustration of the program's execution. The majority of students rated Hank as easy to work with in a group, where difficulties did arise they were generally due to the social dynamics of the group rather than the language being used.

The effect of the face-to-face tutorial support could not go unnoticed in these studies, support and encouragement were provided by tutors and welcomed by the students. Programming was

not an activity that the vast majority of students had any experience of and although the topic was seen as a challenging part of the course, the students' appreciation of its worth and sense of achievement from completing the cognitive modelling projects were considerable. The students' usage of computer packages particularly spreadsheets and databases, did appear to have an affect on the students' perception of their ability to use Hank. However, this may be a reflection of the students' confidence in their abilities rather than an accurate indication of their actual ability. The students' and tutors' comments regarding the advantages of using Hank indicate that prior computing experience does not influence the students' ability to use it.

The introduction of the fundamental computing concepts used in Hank is another aspect of the course that requires careful attention. It is clearly important, given the comments of the students and tutors, to introduce the existence of the additional Hank primitives to the students at the beginning of their advanced modelling project, even if a fuller explanation of their function is delayed until the additional primitives are required. The concerns raised in the programming walkthrough evaluations regarding the sufficiency of the explanation of programming variables and the matching process used in Hank were echoed by some of the tutors' comments. These concerns may be alleviated by clarifying or extending the explanation of the computing concepts used in Hank either at a face-to-face tutorial or in the support materials provided.

The tutors' comments regarding the assessment of the students' assignments indicated that the cognitive modelling assignments were relatively difficult to mark compared to other course assignments. Changing the modelling language used on the course has implications for those tutoring the course as well as those studying it.

## 7. Conclusions

The research has considered what advantages were to be gained from using visual programming in this educational context and has sought to identify the source of those advantages. The data has also indicated, however, that the setting of the visual language within a sound teaching framework appears to be an important factor in its success in meeting its design goals. The studies described here support the following conclusions:

- Visual languages by their design can provide a simple syntax that makes them easier to use than textual languages with a more complex syntax.
- The comic strip notation used in the Workspace view of the Hank modelling environment does offer effective support for showing how a program works. However, like the comic strip notations studied by Traynor and Williams (1997), it may be less effective when showing complex control flow, such as branching and looping.
- While the visual programming language being investigated appeared to facilitate programming for those with little experience, it is nevertheless important that support materials and teaching dialogue uses consistent and clear descriptions throughout.
- The importance of practical experience in cognitive modelling is as important when using a visual programming language as it would be when using a conventional text-based programming language. In the same way as reading about other people's experiments is not as interesting or informative as carrying out those experiments for oneself, reading about

cognitive modelling is no substitute for practical experience. Hank did appear, however, to make that experience more accessible.

## Acknowledgements

The authors would like to thank the members of the Open University's Psychology Department for their help and assistance in completing this research, particularly the members of the cognitive psychology course team, namely: Sandy Aikenhead, Nick Braisby, Judy Green, Peter Naish, Ingrid Slack and Stuart Watt. The programming walkthrough evaluations described in this paper were undertaken in collaboration with Clayton Lewis from the University of Colorado at Boulder and Paul Mulholland from the Open University's Knowledge Media Institute. The authors would like to thank Clayton and Paul for their help and support during this project. This work was funded by a research committee grant from The Open University's Institute of Educational Technology.

## References

Brown, J. S., & Burton, R. R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science*, *2*, 155–192.

Collins, T. D., & Fung, P. (1999). Cognitive modelling for psychology students: the evaluation of a pragmatic approach to computer programming for non-programmers. In *The Proceedings of the 7th International Conference on Computers in Education*, pages 216–223. Osaka, Japan. November 1999.

Collins, T. D., & Fung, P. (2000a). Hank- a visual programming language: Its use as an introduction to cognitive modelling. CITE Technical Report Number 259. Institute of Educational Technology, The Open University, UK. June 2000.

Collins, T. D., & Fung, P. (2000b). Novice cognitive modellers using a visual programming language. CALRG Technical Report Number 193. Institute of Educational Technology, The Open University, UK. October 2000.

Collins, T. D., & Fung, P. (2000c). Visual programming for cognitive modelling: A teaching perspective. CALRG Technical Report Number 194. Institute of Educational Technology, The Open University, UK. October 2000.

Collins, A., & Quillian, M. (1969). Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behaviour*, *9*, 432–438.

Cunniff, N., & Taylor, R. P. (1987). Graphical vs. textual representation: an empirical study of novices' program comprehension. In *The Proceedings of the Empirical Studies of Programmers Second Workshop*. G. M., Olson, S. Sheppard, and E. Soloway, Eds.

Eysenck, M., & Keane, M. (1995). *Cognitive psychology: a student's handbook* (3rd ed.). Hove, UK: Lawrence Erlbaum Associates.

Hendry, D. G., & Green, T. R. G. (1994). Creating, comprehending and explaining spreadsheets: a cognitive interpretation of what discretionary users think of the spreadsheet model. *International Journal of Human-Computer Studies*, *40*, 1033–1065.

Lewis, C., & Rieman, J. (1993). *Task-centered user interface design*. Shareware book available from ftp://ftp.cs.colorado.edu/pub/distribs/clewis/HCI-Design-Book/.

Lieberman, H. (1992). Dominos and storyboards: beyond icons on strings. In *The Proceedings of the IEEE Conference on Visual Languages*. Seattle, 1992.

Mulholland, P. (1998). A principled approach to the evaluation of Software Visualization: a case-study in Prolog. In J. Stasko, J. Domingue, M. Brown, & B. Price (Eds.), *Software visualization: programming as a multimedia experience*. Cambridge, MA: MIT Press.

Mulholland, P., & Watt, S. (1998). Hank: A friendly cognitive modelling language for psychology students. In *The Proceedings of the IEEE Symposium on Visual Languages, VL'98*. Nova Scotia, Canada.

Nardi, B. A., & Miller, J. R. (1991). Twinkling lights and nested loops: problem solving and spreadsheet development. *International Journal of Man-Machine Studies*, *34*, 161–184.

Repenning, A., & Citrin, W. (1993). Agentsheets: applying grid-based spatial reasoning to human-computer interaction. In *The Proceedings of the 1993 IEEE Workshop on Visual Languages, VL'93*.

Scanlan, D. A. (1989). Structured flowcharts outperform pseudocode: an experimental comparison. *IEEE Software*, *6*, 28–36.

Schank, R. C. (1972). Conceptual dependency: a theory of natural language understanding. *Cognitive Psychology*, *3*, 552–631.

Smith, D. C., Cypher, A., & Spohrer, J. (1994). KIDSIM: Programming agents without a programming language. *Communications of the ACM*, *37*, 13–30.

Simon, H. A. (1986). Information processing theories of human problem solving. In W. K. Estes, (Ed.) *Handbook of Learning and Cognitive Processes*.

Traynor, C., & Williams, M. G. (1997). A study of end-user programming for graphic information systems. In S. Wiedenbeck, & J. Scholtz (Eds.), *The Proceedings of the Empirical Studies of Programmers Seventh Workshop*. New York: ACM Press.

Vessey, I., & Weber, R. (1986). Structured tools and conditional logic: an empirical investigation. *Communications of the ACM*, *29*, 48–57.

Young, R., & O'Shea, T. (1983). Errors in Children's Subtraction. *Cognitive Science*, *5*, 153–177.