An Adaptive Web-based Component for Cognitive Ability Estimation

I. Arroyo*; R. Conejo; E.Guzman**; B.P. Woolf***

**** Department of Computer Science - School of Education. University of Massachusetts, Amherst***
*****Departamento de Lenguajes y Ciencias de la Computación Universidad de Málaga, Spain.***
ivon@cs.umass.edu; conejo@lcc.uma.es; guzman@lcc.uma.es; bev@cs.umass.edu

Main contact:

Ivon Arroyo
Department of Computer Science
Computer Science Building
University of Massachusetts, Amherst
MA 01003-4610
Telephone number: +1 413 253 4146
Fax number: +1 413 545 1249

Abstract: This paper describes the development of web-based reusable components for the creation of adaptive pre-tests to be used by on-line ITS. In particular, we have created an adaptive pre-test to measure a student's cognitive ability. We created this component by combining two existing applications: a Piagetian pre-test [Arroyo 99] and SIETTE, an adaptive web-based tutoring system [Rios 99]. The advantages of this architecture are three-fold. Given that the component allows for high interactivity, it is possible to overcome the limitations of traditional multiple-choice evaluations. Thanks to its adaptive nature, the test can be done in minimal time, thus allowing more time to be spent on the actual instruction. Finally, thanks to its modular structure, this component can be used by any tutoring system that needs to establish a student's cognitive ability.

Keywords: cognitive diagnosis, cognitive development, networked learning and teaching systems, model-based reasoning.

An Adaptive Web-based Component for Cognitive Ability Estimation

I. Arroyo*; R. Conejo; E.Guzman**; B.P. Woolf***

**** Department of Computer Science - School of Education. University of Massachusetts, Amherst***
*****Departamento de Lenguajes y Ciencias de la Computación Universidad de Málaga, Spain.***
ivon@cs.umass.edu; conejo@lcc.uma.es; guzman@lcc.uma.es; bev@cs.umass.edu

Abstract: This paper describes the development of web-based reusable components for the creation of adaptive pre-tests to be used by on-line ITS. In particular, we have created an adaptive pre-test to measure a student's cognitive ability. We created this component by combining two existing applications: a Piagetian pre-test [Arroyo 99] and SIETTE, an adaptive web-based tutoring system [Rios 99]. The advantages of this architecture are three-fold. Given that the component allows for high interactivity, it is possible to overcome the limitations of traditional multiple-choice evaluations. Thanks to its adaptive nature, the test can be done in minimal time, thus allowing more time to be spent on the actual instruction. Finally, thanks to its modular structure, this component can be used by any tutoring system that needs to establish a student's cognitive ability.

1. Introduction

There has been a vast amount of research to identify the relationship between different students' learning traits and different teaching strategies [Jonassen, 93]. If a student's learning traits are known, an ITS can adapt its behavior to use the most effective teaching strategy. For example, the SMART tutoring system [Shute, 95] adapts its behavior to different levels of IQ. Another important learning trait that can be identified and used by an ITS is a student's *cognitive ability*, as defined by Piaget's theory of cognitive development [Piaget 53]. The cognitive abilities that Piaget studied are important when teaching young students, as they help determine how much and in what way students will understand the topics being taught. In the realm of ITS, knowing a student's cognitive ability can greately improve the effectiveness of a tutoring system [Arroyo, 00]. Central to the idea of *adaptive teaching* are pre-tests, procedures that allow an ITS to identify which traits a particular student has. [Arroyo 99] describes a computer-based pre-test to measure children's cognitive ability and presents guidelines as to how instruction can be adapted given those differences. Arroyo's pre-test consists of ten computer-based Piagetian tasks to estimate individual differences in cognitive development among students between ages 8-11. One problem with this type of pre-tests is that the time they demand is usually prohibitive. Students must go through 10-15 minutes of testing before they can to actually use the ITS. The immediate question that arises is if it is worth it to use such a long time for pre-testing. What if the student is planning to use the tutoring system for only 20 minutes? Fortunately, techniques for adaptive testing have been widely deployed. Adaptive tests have the advantage of significantly reducing testing time. Thanks to these techniques, pre-tests can be run in minimal time, thus allowing more time to be devoted to the actual interactions with the ITS.

Another important line of research in the ITS community, of particular popularity in recent years, is the construction of reusable components for ITS development. This is a classical aim of software engineering. It has also been a claimed goal in the construction of knowledge-based systems. One of these is the classic basic blocks idea proposed by Chandrasekaran back in the early 80's, for the development of expert and knowledge based systems [Chandrasekaran, 92]. Nowadays, the open architecture of the WWW makes integration of components an easier task than in closed architectures. In the web, system integration can be achieved just by having appropriate hyperlinks. Sometimes these components are also called *agents*, to emphasize their independence. However, there are not many components available for direct use. Researchers usually construct their adaptive systems from scratch and avoid re-using components because the integration does not usually work at the adaptive level, and agents live in their hometowns talking only to other agents in their private societies.

This paper shows how these research ideas, namely pre-tests for student trait diagnosis and adaptive reusable components, can be integrated to create better ITS. It presents a prototype of a reusable component that emerges from the integration of two existing software systems, Arroyo's Piagetian test and the SIETTE system [Conejo 00], a web-based general-purpose tool that implements the IRT and CAT theories. The result is a reusable highly interactive component that measures a student's cognitive ability by taking advantage of SIETTE's adaptive testing techniques. We will first present the Piagetian pre-tests and SIETTE separately, and then describe how we integrated the two systems. Finally we will analyze the advantages of the new component, discuss these ideas and propose future work.

2. Cognitive development levels and the Piagetian tests

The cognitive development test consists of ten highly interactive computer-based tasks that 8-11 year olds are asked to solve. Students of this age are supposed to have abilities that range from concrete to formal thinking. Thus, we designed tasks that measured abilities within these two stages of development. All these tasks are based on those designed by Piaget, adapted from the descriptions in [Voyat, 82]. They were implemented in Java (JDK 1.1) and originally ran as a standalone application. Figure 1 shows screenshots of these tasks already in the SIETTE system.

Seven of the tasks test for *concrete reasoning:*

· ***Seriation***: Students have to sort a group of four pencils, from the shortest to the longest one.

· ***Number conservation***: Students initially observe two identical sets of cookies (each set consists of nine horizontally aligned cookies). When the elements of one set move to form a small circle, students are asked to determine if the amount of cookies has changed in the new state (see Figures 1a and 1b).

· ***Substance conservation***: Students are initially presented with two identical vessels with the same amount of liquid. Each of these containers has an empty vessel next to it: one is narrow and the other one is wide. Students are asked to show where they believe the level of water would be IF the liquid from the two initial vessels gets poured into the wide and narrow vessels next to them (see Figures 1c and 1d).

· ***Area conservation***: Students are asked to compare the space in two parking lots. They both have the same area, but one of them is a big block while the second one is split into separate pieces.

· ***Class inclusion***: Students have to determine whether there are more dogs or animals in a set of animals. The idea is to test their understanding that the "dogs" subset is always smaller or equal to the set of "animals".

· ***Functionality***: Students have to solve the problem of sorting four pencils by length when they can only see the full length of two of them at a time.

· ***Reversibility***: Students see an animation of three colored balls getting inside of a can, one at a time, and they have to predict the order in which the elements will come out of the can (the behavior of a stack).

The other three tasks determine a student's handling of *formal operations*:

· ***Control of independent variables in experimental design, drawing of conclusions***: Students get four simulated plants to experiment with. They have to conclude if it is better to water a plant once a week or once a day, by watching them grow. They not only have to decide whether to water the plant once a day or once a week, but they also have to choose one of two environmental temperature levels for the plant. After this experimentation, students are asked what watering frequency is better for the plant's growth.

· ***Proportionality***: Students see two animals of different heights. They are asked to measure them with two different measurement units (large buttons and small buttons). Students are asked to measure one of the animals with both small and large buttons, and the other animal only with large buttons. Then, buttons are not available for measurement any more, and students are asked to infer the height of the second animal in small buttons.

· ***Combinatorial analysis***: Students have to find the combination of four switches that would open a safe. The goal here is to evaluate if the student can systematically generate combinations of four elements. Because this is hard, we only check the number of combinations they generate: the safe opens when the student generates 15 combinations (pretty good chance they would have obtained them systematically), or after a maximum number of trials.

In the original non-adaptive test version, students go through the 10-item test. After finishing this test, the student has a score, a number ranging from 0 to 10, corresponding to the number of correct answers. A student with a score of 4 or less may be considered to have *low cognitive ability*. A student scoring 5-10 may be considered to have *high cognitive ability*.

The standalone version has been used with hundreds of 8-11 year-old students as a pre-test to Animalwatch, a mathematics tutoring system for children. Although the test has never been validated by paralleling it with human experimentation, the test outcomes make us believe that it is accurately measuring cognitive ability. In particular: 1) there is a strong positive correlation between success in these experiments and students' mathematics performance in Animalwatch [Arroyo, 99]; 2) concrete tasks are easier for students to solve than formal ones (see figure 4); 3) students with a low score in the cognitive ability test specially benefited from more concrete kinds of help in the tutoring system, and less from abstract ones [Arroyo, 00]; 4) students with a high score in the cognitive test benefited more from any kind of help than low cognitive ability students

[Arroyo, 00]; 5) students with a high score in the cognitive test specially benefited from abstract help, much more than low cognitive ability students [Arroyo, 00].

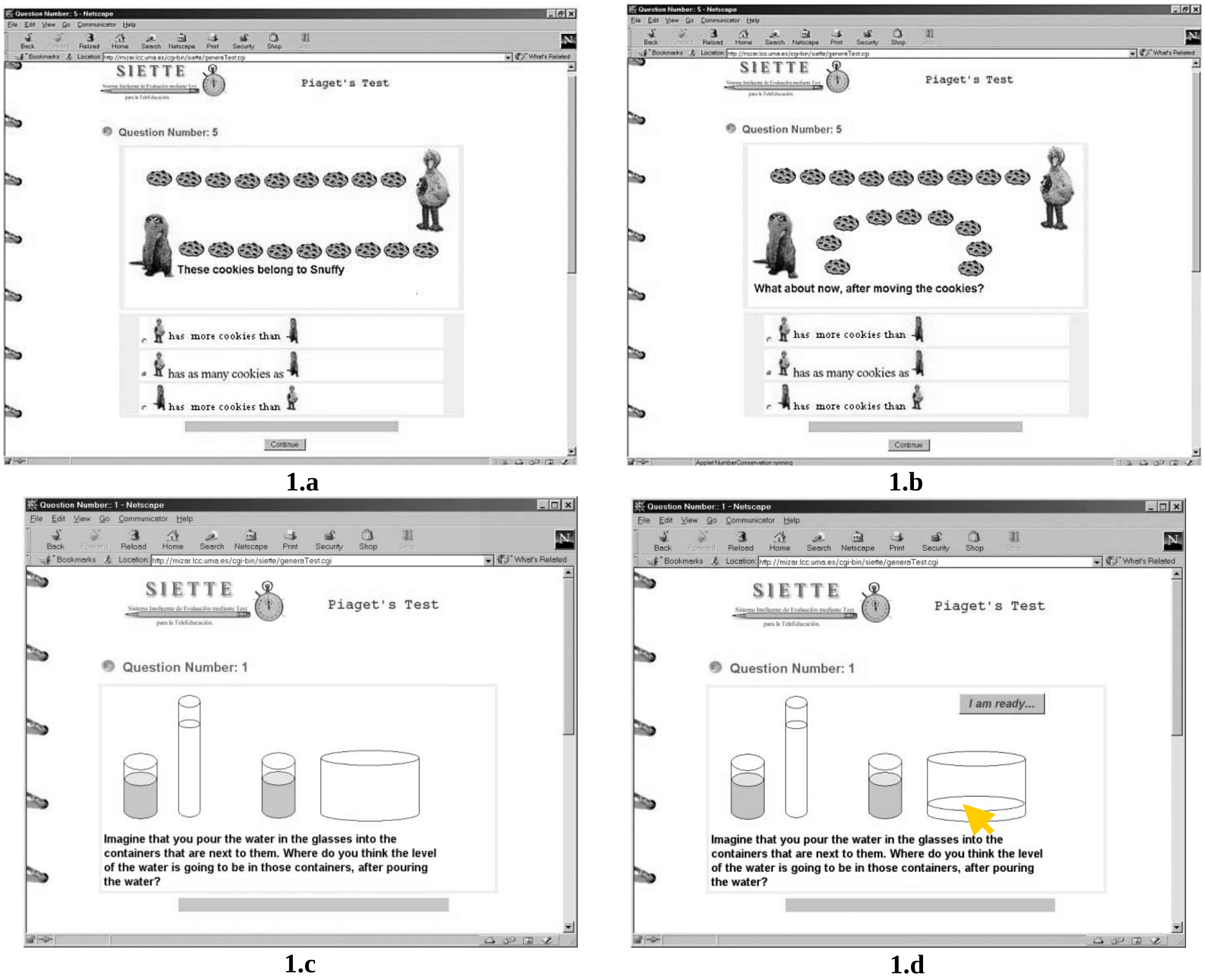

Figure 1: Screenshots of the tasks that test for *Number Conservation* and *Substance Conservation*

Given results 3, 4 and 5, we can devise a policy for adapting help provision in Animalwatch. Given that Animalwatch has access to the student's cognitive ability score, it should provide different help to low vs. high cognitive ability students: it should give concrete explanations[1] to students of low cognitive ability, and more formal explanations[2] to students with high cognitive ability. We plan to run a new experiment for testing the validity of this hypothesis in the near future.

Adapting the ITS's behavior to other student traits [Jonassen, 93] is a promising idea, which requires the use of more diagnostic tests. It is important that the total pre-testing time is shorter than the actual time the student spends actually using the tutoring system. Adaptive testing thus becomes an essential component in such student-trait-sensitive tutoring systems. The next sections introduce SIETTE, a system with adaptive testing facilities, and they show how we have used those facilities to create an adaptive Piagetian cognitive ability test.

3. Siette and on-line adaptive tests

Introduction to Siette and its adaptive testing facilities

SIETTE [Conejo 00] is a web-based tool to assist instructors in the evaluation process. Teachers can define their tests in SIETTE through a WWW-based interface, and then their students can then take the tests on-line.

[1] Concrete explanations have a low symbolism level. They are related to real-life objects, and they are highly conceptual. They involve high manipulation and interactivity (dragging, partitioning, etc.). They are highly visual. While these concrete procedures seem simpler and more intuitive (i.e. adding by grouping together base-10 blocks –cuisenaire rods–), they are hard to execute with large numbers.

[2] Formal explanations are highly symbolic, abstract, focussing on numeric procedures. They have the advantage of being more generalizable. They are generally faster to execute than the concrete ones, though it is harder to make sense of how each step relates to a real life situation (i.e. adding multi-digit numbers in columns, and writing the carried number on top of the column to the left).

SIETTE can be used in two different ways: as an independent evaluation tool, or as a component of the diagnostic module of an ITS with a curriculum structured knowledge base [Trella,00].

SIETTE is based on the Computer Adaptive Testing theory and the Item Response Theory [cite]. The tests are *adaptive*, as the questions are selected intelligently to fit the student's level of knowledge. A *Computer Adaptive Test* is a computer-based test where the decision of presenting a test item and the decision to finish the test are dynamically made depending on the examinee's performance in previous answers. If two examinees are compared, they will usually receive different sequences of questions, and even different questions.

In SIETTE, each question ***i*** in a test is assigned an Item Characteristic Curve (ICC), which is a discrete function representing the probability of a right answer to that question given the student's knowledge level θ (defined as an integer between **0** and **N**). The probability of succeeding in a test item ***i*** can be represented by the expression: $P(u_i=1|\theta)$ or just P_i. The probability of failing the question is $P(u_i=0|\theta) = 1-P(U_i=1|\theta)$, or simply Q_i. If the test is composed by ***n*** questions, knowing the ICCs, and assuming local independence of items, a likelihood function can be constructed:

$$L(u_1, u_2 \ldots u_n | \theta) = \prod_{i=1}^{n} P_i^{u_i} Q_i^{1-u_i} \qquad \textbf{(I)}$$

The maximum of this function gives an estimation of the most likely value of θ. A distribution of the probability of θ can be obtained applying Bayes' rule ***n*** times. SIETTE performs the bayesian classification of the examinee in one of the **N+1** categories of knowledge levels according to his answers to the ***n*** questions proposed.

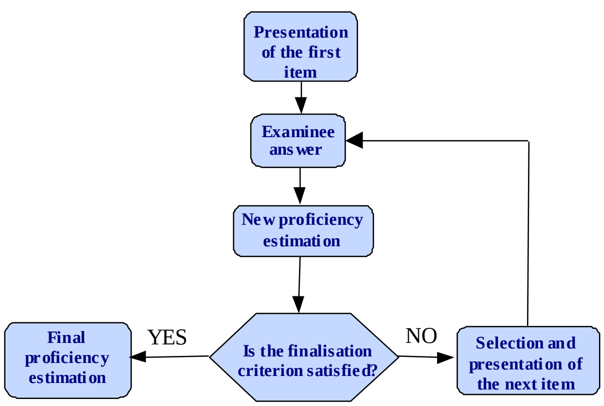

Figure 2. Flow diagram of an adaptive test. Adapted from (Olea & Ponsoda, 1996).

A *CAT* test uses the ICC in an iterative algorithm, which starts with an initial estimation of the examinee's proficiency level and then performs the following steps:

1. All the questions in the database (those not administered yet) are examined to determine the best one to be given next, depending on the current estimation of the examinee's proficiency level.
2. The question is asked. The examinee responds.
3. A new proficiency level estimation is computed, depending on the examinee's answer.

Steps 1 to 3 are repeated until the defined stopping criterion is met (see figure 2)

The item selection and finalization steps consist of a bayesian procedure whose accuracy is controlled by certain parameters, which need to be estimated based of data points corresponding to people taking the full test. While the student is taking the test, the system creates (and updates) a temporary student model.

The main advantage of adaptive testing is that it reduces the number of questions needed to estimate the knowledge level of the student (and the accuracy of that estimation is of course much higher than the estimation achieved by randomly picking the same number of questions). In addition, large question databases can be stored and selection algorithms can be performed efficiently [Rios, 99].

Interactive tests with question templates and applets

SIETTE was initially designed to deal only with multiple-choice questions. In the original version, the body of the question was a chunk of HTML code with radio buttons for the multiple choices. Once the student had selected the answer, it was sent to the server where a CGI program would process it. One of the earliest improvements of SIETTE was the generation of dynamic HTML code (using the PHP programming language[3]), both for the question and for the answers. Instead of having static HTML text for the question and alternative answers, it is possible to include programming code that dynamically generates a different stem each time it is called, and the corresponding correct and distracting answers. So instead of including a single fixed question, designers could now include templates of questions that are considered in SIETTE as a single item, and are instantiated with different values each time they are used in a test.

[3] See http://www.devshed.com/Server_Side/PHP/Introduction for how PHP can dynamically generate web pages

The latest version of SIETTE can also present JAVA applets as questions. There are two ways of incorporating applets in SIETTE:

Adding the applet to the question or the answer section. It is possible to define test items where the question or the answers include an applet (for example, a simulation of physical phenomena). The student is asked to select the correct option after having seen the applet. This mechanism allows SIETTE to measure abilities that will be difficult to measure with traditional pencil and paper tests. This way, visual or auditory abilities can be tested, even perception and attention. This kind of questions can be defined and used in a SIETTE test by including the applet in the section corresponding to the coding of the question or the answers (see Figures 1a and 1b).

Having the applet perform the evaluation. The applet itself can determine whether the answer is correct (see Figures 1c and 1d for an example). In this case, the system will pose a question that consists of a small program, which is executed and its output shown to the student. The student submits an answer by interacting with the applet, instead of choosing a multiple-choice option. It is the applet who determines the correctness of the answer, and passes along this information to the SIETTE inference mechanism, as illustrated in Figure 3.
It is very easy to define these items for the test designer. The applet programmer must inherit from a JAVA abstract class and overwrite two methods: a method called `evaluation()`, which evaluates if the answer is correct, and another called `solve()` which, if implemented, will show the correct solution to the student. The applet can decide when to call the evaluation method depending on user events on the applet (clicking, button pressing, etc.), or other events (a time limit, etc.). Moreover, this applet-evaluated question type can be combined with the traditional multiple-choice ones in one same test. Using applets in the questions opens new possibilities: it is possible to minimise the effect of guessing factors, to control valuable information like answering time, or to measure abilities which are difficult to evaluate using multiple-choice format.

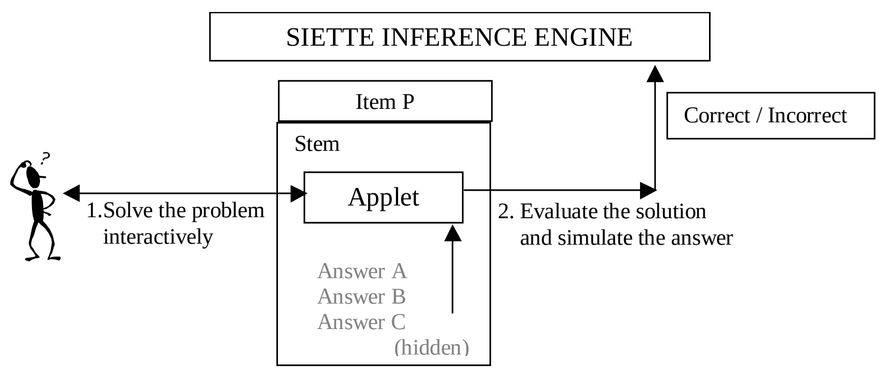

Figure 3. Including applets in the evaluation process

4. Integrating SIETTE and the Piagetian tests: A reusable component

In order to take advantage of the two applications, we modified the Java-based Piagetian test and turned it into a Java applet that could be included in SIETTE. To do it, we altered the original application and split it into a set of 10 independent applets, one for each Piagetian task. Some of the items were multiple-choice based questions, and they were rearranged as standard SIETTE items. For the non-multiple-choice tasks, the `evaluation()` and `solve()` methods were implemented (see Figure 3). The result is a SIETTE module that diagnoses a student's cognitive ability by presenting the tests it considers necessary to the students and then classifies them into one of 11 categories.

To construct the adaptive test in SIETTE, we also need to estimate the value of the conditional probabilities of each single test item given some student knowledge level θ. This determines the ICC(θ) function. Many methods have been proposed for this task in IRT (see [Wainer, 90]). The main difference between SIETTE and classical IRT method is that IRT deals with the ICC function in the domain of real numbers, while SIETTE uses a discrete function. So, the estimation achieved by SIETTE is much easier and can be done by adapting the IRT method for discrete values, or just by measuring the frequencies of correct answers to each question given a certain knowledge level. We used data from previous uses of the standalone Piagetian test to estimate the parameters that need to be used in the bayesian procedure for finalization and item selection, as explained in the previous section. The data was obtained from a study with three groups of 150 8-11 year olds attending rural and suburban schools in Massachusetts. The total population was randomly divided into two groups, one of 100 cases (group A) and another one of 50 cases (group B). We applied SIETTE's estimation technique to the data from Group A, and considered that the actual cognitive ability of each child is given by the total number of correct answers. This hypothesis is not necessarily true, because the test has not yet been contrasted with any other psychometric measures of cognitive ability. That goes beyond the objective of this paper, which is to show we can mimic the behavior of the original test getting a similar performance with less questions.

Getting 150 cases was already extremely expensive given that each case corresponds to a different child. The number of cases we need to make an accurate estimation is unfortunately higher than what we have. If students want to be classified into one of 11 categories and we have 100 cases to estimate the ICC, there is an average of 9.1 cases per value. The statistical error in the estimation (according to binomial distribution formulas) is considerably high in this case (±0.166 in the neighborhood of a probability of 0.5). We could improve the estimation by forcing the ICCs to behave according to a best fitting curve, but still the accuracy of the estimation can not be guaranteed. On the other hand, we will later show that this amount of cases is good enough when aiming for less than five target categories. Moreover, this initial parameter estimation is just a starting point, as one of the advantages of a web-based system is that the results of the adaptive test can be used as a source for new parameter estimation. This kind of on-line learning procedures have been studied [Conejo 00] and are currently under implementation in SIETTE.

The results of this initial calibration are shown table 1. The rows correspond to each task and the columns correspond to the three parameters needed for a logistic function used for test finalization and item selection. As a general conclusion of the analysis of these data, there are some tasks that are better than others for finding cognitive differences in this age range. For instance, the *Seriation* task is too easy, as it has been completed by almost all children, and so it doesn't provide much information. The discrimination factor is usually low, which means a single task does not provide enough information to classify into 11 categories. Although important, the adaptive behavior would not be expected to be impressive in this case because of high statistical uncertainty.

Item	Difficulty	Guessing	Discrimination
SERIATION	1	0.95	0.25
NUMB CONS	2	0.05	0.20
SUBS CONS	3	0.10	0.40
AREA CONS	4	0.15	0.45
FUNCTION	5	0.15	0.40
REVERSIBIL	5	0.05	0.35
HYPOTHES	5	0.10	0.30
INCLUSION	6	0.10	0.60
PROPORT	7	0.00	0.55
COMBINAT	9	0.00	0.40

Table 1. Results of parameter estimation

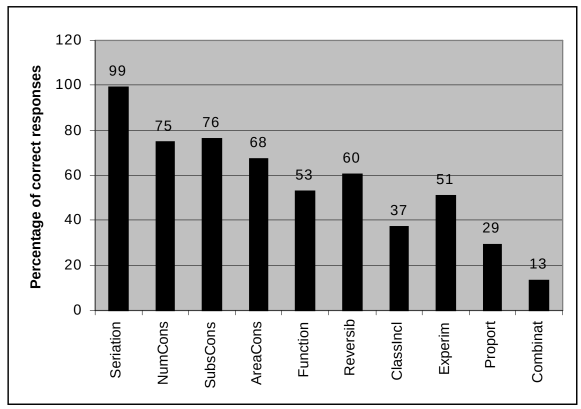

Figure 4. Test item difficulty (data from 150 students)

The Piagetian test as a reusable SIETTE component

With its integration into the SIETTE system, the Piagetian test becomes a module that any web-based tutoring system can use to estimate its students' cognitive development. All an interested on-line ITS has to do is redirect the student's browser to the URL for the Piagetian test in SIETTE. SIETTE will present the test to the student and, when it has collected enough information, it will redirect the student back to the original site, passing along the diagnosed cognitive ability of this student.

Furthermore, a calling ITS can specify a collection of different URLs to which SIETTE should redirect the student according to his cognitive level (for example, one URL if the student has low cognitive ability, another for medium cognitive ability, and a third for high cognitive ability, as shown in Figure 5). Each of those different URLs could provide the student with a starting point for different kinds of tutoring (different student model, different pedagogical actions, etc.). If the desired number of outcome categories is less than 11 (3 in the last example) a mapping function is constructed to estimate the probability of the student belonging to each category. The number of desired outcome categories is determined dynamically by SIETTE depending on the number of call-back URL references in the first call to SIETTE. If the call to the SIETTE Piagetian test includes three call back URLs, SIETTE will automatically construct a mapping function between the eleven internal levels and the three desired categories (for low, medium and high). With this mapping function, SIETTE estimates the probability of the student belonging to each of the desired categories and picks the category that first reaches statistical certainty. SIETTE finishes the diagnosis at that time. The maximum error tolerance in the classification is also passed in the first call, and dynamically set in SIETTE. Of course, the lower the number of categories the lower the number of questions the adaptive testing system requires to achieve a diagnosis, and the greater the accuracy in the classification.

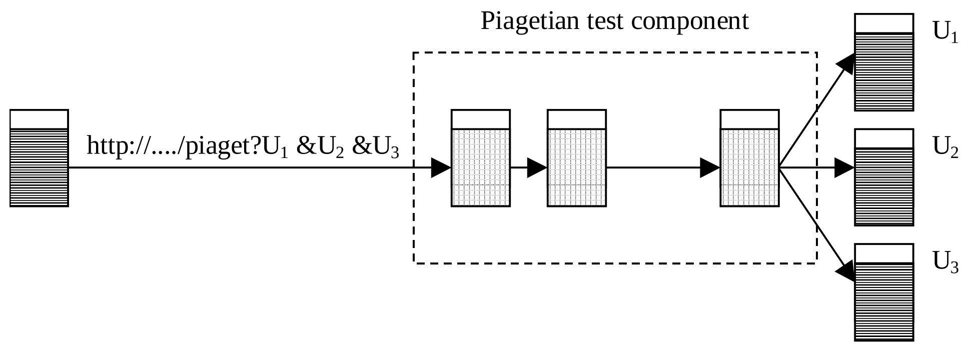

Figure 5. Integration of the component with another system

The next section analyses the behavior of the component for the Piagetian test and presents some experimental results.

5. Experimental tests and results

The first thing to study is the validity of the adaptive implementation of the Piagetian test. As we have mentioned in the last section, our aim is to mimic the behavior of the stand-alone version. The key question is: *are we obtaining the same results in the adaptive as in the non-adaptive test?* Let's define *accuracy* as the percentage of children who are classified into the same category as the non-adaptive test. As expected, the correlation between the non-adaptive and adaptive test increases when the desired number of categories decreases (figure 6a). However, 2/3 of the children was correctly classified when using 11 categories. If less than four cognitive development levels are considered, which is the typical case in the macroadaptive ITS we referred to in previous sections, then accuracy is very high (98% for 2 categories). Figure 6b shows the cross frequencies between both types of tests. As can be seen, the diagonal contains most of the cases.

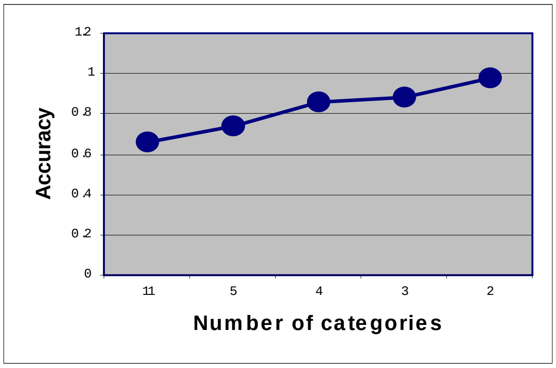

Figure 6.a Accuracy of the non-adaptive test

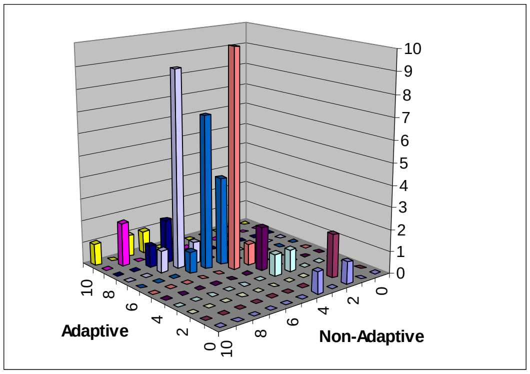

Figure 6b. Cross frequencies

The second interesting aspect is the *number of questions posed*. In the non-adaptive test each child should take exactly 10 questions. In the adaptive test, the average number of questions posed depends on several factors. There is a trade-off between *accuracy* and number of questions posed. The highest possible *accuracy* has been chosen for this study. Fig 6a shows the relationship between the number of questions posed and the person's cognitive level, assuming that there are 11 possible cognitive levels. The reader may notice that for higher levels the number of questions needed is less. The explanation of this effect relays on two factors. First, the statistical variance of the population is higher for central values, and so the number of questions needed to achieve the same accuracy is higher. Second, there is an important influence of the guessing factor, which is greater for easier questions that are posed at lower levels, as pointed in the previous section.

Fig 7 shows the average number of questions posed to each child depending on the number of cognitive development levels desired for the final classification. The reader may notice that the rougher the classification the less the number of questions needed to finish the test. Because the discrimination factor of the items is low, we can not expect to correctly classify a subject with less than 3-4 questions even for only two levels.

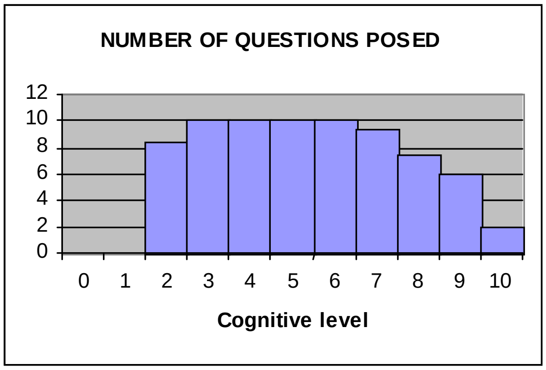

Figure 7a. Num. questions posed / cognitive level

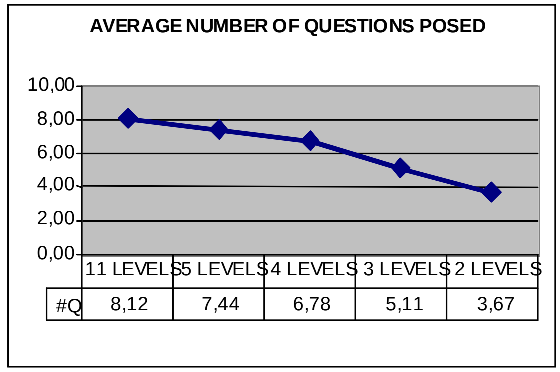

Figure 7b. Num. questions posed / Num. categories

6. Conclusions and future work

In this paper we have presented an independent, adaptive, easy to integrate web-based component to evaluate a student's cognitive development. This component can be used as the pre-test of an adaptive tutoring system. It has been constructed by including an existing test's items into the SIETTE adaptive web-based testing system. The result is a flexible and reusable adaptive component that works over the web. It can be easily plugged-in as a front end to any adaptive web based system that needs a quick cognitive diagnosis of the user. We have also introduced technical modifications in SIETTE to allow the inclusion of applets in the item pool. The possibilities that open because of this technique go far beyond the possibilities of the classical multiple choice items that have been traditionally used in Computer Adaptive Testing. We have empirically analyzed the benefits of using the adaptive capabilities of this component compared to the results obtained in a sequential non-adaptive implementation of the cognitive test. We showed that a substantial reduction in the number of questions can be achieved for a low number of target categories. If trying to classify the student into five possible cognitive levels this adaptive test would save half of the time in testing compared to the full version.

We also presented the idea of adaptive testing as a tool to help intelligent tutoring systems make a quick diagnosis of student's traits, to then adapt its pedagogical and student models to these differences. We intend to create other computer-based tests with the final goal of making ITS sensitive to student individual differences, not only with respect to cognitive abilities, but also for the diagnosis of learning styles and emotional traits [Jonassen, 93]. We also intend to study the potential integration of this adaptive cognitive test with other mathematics ITS.

This component can be tried at **http://www.lcc.uma.es/siette/piaget**

Acknowledgements. We acknowledge Agustin Schapira for his insights, reviews and comments.

7. References

[Arroyo, 99] I. Arroyo, J. E. Beck, K. Schultz, and B. P. Woolf. Piagetian Psychology in Intelligent Tutoring Systems. In: *Proceedings of AIED '99*. pp. 600-602. 1999.

[Arroyo, 00] I. Arroyo, J. E. Beck, B. P. Woolf, C. R. Beal, K. Schultz. Macroadapting Animalwatch to gender and cognitive differences with respect to hint interactivity and symbolism.In: *Proceedings of ITS'2000*, Montreal (2000) 604-614.

[Conejo, 00] R.Conejo, E.Millán, J.Perez-de-la-Cruz, M.Trella, An empirical approach to on-line learning in SIETTE, en: *Proceedings of ITS'2000*, Montreal (2000) 604-614.

[Chandrasekaran, 92] B. Chandrasekaran. Generic Task: Evolution of an idea. (1992). *Technical Report. Laboratory for AI Research. Ohio University. Columbus OH.*

[Jonassen, 93]D. Jonassen; B. Grabowski (1993) *Handbook of Individual Differences, learning and Instruction*. L. Erlbawm.

[Olea, 96] J. Olea & V. Ponsoda. Tests adaptativos informatizados. In Muñiz, J.(ed) *Psicometría*. 1996. Madrid: Universitas.

[Piaget, 53] J. Piaget. How Children Form Mathematical Concepts. 1953. *In Scientific American*.

[Rios, 99] Ríos, A., Millán, E., Trella, M., Pérez-de-la-Cruz, J. L., & Conejo, R. (1999). Internet Based Evaluation System. In *Proceedings of the 9th World Conference of Artificial Intelligence and Education AIED'99* (pp. 387-394).

[Shute, 95] SMART: Student Modeling Approach for Responsive Tutoring. (1995). In *User Modeling and User-Adapted Interaction*. 5:1-44.

[Trella, 00] M.Trella, R.Conejo, ITS Web based Architecture for Hierarchical Declarative Domains, en: *Young Researchers Track Proceedings, ITS'2000*, Montreal (2000) 57-60.

[Voyat, 82] Gilbert E. Voyat. *Piaget Systematized*. Hillsdale, N.J. : L. Erlbaum Associates, c1982.

[Wainer, 90] Wainer, H. (ed.). *Computerized adaptive testing: a primer*. Hillsdale, NJ: Lawrence Erlbaum Associates. (1990).