A K-12 College Partnership[1]

Stephen Cooper
Purdue University
Computer Graphics Technology
West Lafayette, IN 47906 USA
1.765.494.6130
coopers@acm.org

Wanda Dann
Carnegie Mellon University
Computer Science Department
Pittsburgh, PA, 15213 USA
1.412.268.9959
wpdann@andrew.cmu.edu

John Harrison
Princess Anne High School
Virginia Beach, VA, 17837 USA
1.757.648.5600
John.Harrison@vbschools.com

ABSTRACT
To combat the decline of teaching computing courses in high schools, a plan was created to offer professional development (in both content and pedagogy) to high school computing teachers, and to have college faculty partner with those high school teachers to implement innovative curricula. This paper describes a pilot program that was run in support of an NSF Innovative Technology Experiences for Students and Teachers (ITEST) project that funded this plan of action. Professional development activities were offered during summer 2006 and the resulting curriculum was taught in high school the following academic year (2006-2007).

Categories and Subject Descriptors
K.3 [**Computers & Education**]: Computer & Information Science Education – Computer Science Education.

General Terms: Languages

Keywords: Object-oriented, Programming, Alice, K-12, Program Visualization.

1. INTRODUCTION
There has been a significant decrease in the number of Computer Science (CS) students since the peak in 2000-2001. According to the Higher Education Research Institute at the University of California at Los Angeles, "the percentage of incoming undergraduates indicating that they would major in CS declined by more than 60 percent between the fall of 2000 and 2004, and is now lower than … in the early 1980s" [15]. Over the past 3 years, the overall number of baccalaureate students in CS seems to have bottomed out, but at a level nearly 70% lower than the number of students in 2000. The Computer Research Association's Taulbee study reports similar declines [20]. (The Taulbee survey does report a slight uptick in the number of students enrolled in Bachelor's programs this past year, and a decline of only 10% of students receiving BS degrees in computing between 2007 and 2008. However, the Taulbee survey has changed its data collection to include a broader number of degree programs.) Recent data provided by College Board further indicates a significant drop in the number of high-school students that complete advanced placement (AP) tests in CS relative to overall test takers. While the number of AP tests taken increased by 82% between 2000-2006, the number of students taking AP CS A increased by 7% during this time [22]. The situation in CS in high school has deteriorated so much that the College Board recently cancelled offering the CS AB exam, starting this academic year.

As this data suggests, CS is decreasingly viewed as a "hot" career. Attracting students to computing has become an issue for undergraduate CS departments across the US and the situation shows little signs of improvement. Compounding this problem is that many states, at the high school level, are moving computing from the college track to the career and technical track, with a corresponding movement of existing computer teachers back into mathematics and science departments, and the assignment of the teaching of CS increasingly is given to business teachers or any certified teacher, many of whom have no academic preparation in CS [21].

This precipitous drop in interest in CS stands in sharp contrast with job demand data from the US Bureau of Labor Statistics, which predicts that, in the decade 2006-2016, and after accounting for job losses due to outsourcing, there will be more than 700,000 new jobs created just in computer systems and information [9]. This number is much greater than the anticipated number of computing-related degrees granted in the US. Thus, there is a crisis looming on the horizon. Under the current conditions, the US will be unable to satisfy the IT industry's labor demands.

Many colleges have attempted to combat this decline through two main mechanisms. The first has been to improve the quality of the introductory course offerings, in content as well as pedagogic approach. And, there have been many reported successes, both in terms of keeping a higher percentage of declared majors, as well as in convincing students with other majors to take at least a concentration of courses in computing. While it is beyond the scope of this paper to list all of the innovative approaches being tried, a few of the more influential ones are mentioned here. Guzdial [10] has used the approach of students using multimedia to excite them about computing. Both Karel the Robot [2] and some of its offshoots, such as Jeroo [14] use movement of a robot in 2D space as a program visualization to help students to see their programs. Scratch [12], developed at MIT, is a similar tool targeting younger students (and encourages them through animation and storytelling). The Eventdraw library [3] uses the "hook" of GUI building to encourage students in computer

[1] This work was partially supported by NSF grant NSF-0624654. Any views and opinions expressed are those of the authors and do not necessarily represent those of the National Science Foundation.

SIGCSE'10, March 10–13, 2010, Milwaukee, Wisconsin, USA.

programming. The How To Design Programs [8] approach uses functional programming to engage students. The use of programmable clothing [4] may well provide a very strong motivation to encourage women to consider computing. There are numerous approaches to attract students to computing using robots. And while the emphasis of many of these approaches has been in college, it is likely that many of them could be effectively refocused towards high and middle school.

The second approach has been to offer K-12 outreach, most commonly through summer camps or workshops. Here the primary audience is high school and middle school students, though a few efforts have targeted middle and high school teachers. The Computer Science Teachers Association has been very supportive of this approach, both through their Java Engagement for Teacher Training and through Teacher Enrichment in Computer Science workshops [19].

The approach presented here fits into the latter category (focusing on teacher professional development), though with an eye toward building longer term mentoring and support. Rather than limiting the high school teacher professional development to a summer workshop, the approach described in this paper is to build a partnership between a college and nearby school districts, with continued support and interaction throughout the academic year. The remainder of this paper describes a pilot program, funded as part of an ITEST grant, involving a partnership between Saint Joseph's University (Cooper's home institution at the time) and ten of the eleven high schools that make up the Virginia Beach School District.

2. ITEST

NSF's Innovative Technology Experiences for Students and Teachers (ITEST) program "responds to current concerns and projections about the growing demand for professionals and information technology workers in the U.S. and seeks solutions to help ensure the breadth and depth of the STEM [Science, Technology, Engineering and Mathematics] workforce." [18] A focus is on designing programs for K-12 students and/or providing professional development for their teachers.

The project description and project summary from our ITEST proposal are each available from [23]. The plan was to develop partnerships between a college/university and its nearby school districts, and thereby to provide professional development and assistance to middle and high school computer teachers. During the summer, two one-week workshops were to be run, styled on the TECS standard established by the Computer Science Teachers Association [16]. Although details about these workshops are beyond the scope of this paper, we can briefly summarize that the first week focused on Alice [24] content and the second week focused on curriculum development. A third-week workshop would be held in which the K-12 teachers could work with a small group of summer students, to practice using the curricular materials they had just developed. During the following academic year, the college faculty would regularly meet with the middle/high school teachers.

Our ITEST grant proposed six sets of such workshop and mentoring relationships. One of these six was a pilot in Virginia Beach (high schools in partnership with Saint Joseph's University). In following summers, five other workshop and mentoring partnerships were run in Oxford, MS (high and middle schools in partnership with the University of Mississippi), Durham, NC (middle schools in partnership with Duke University), Charleston, SC (high schools in partnership with the College of Charleston and Columbia College), Golden, CO (high schools in partnership with Colorado School of Mines), and San Jose, CA (high schools in partnership with Santa Clara University). Virginia Beach was an ideal location for the pilot program. The nearby naval base helped to ensure significant ethnic and economic diversity among the students. The first two one-week workshops were run during summer 2006. Due to the late awarding of the ITEST grant, the third week workshop was not possible to schedule at the pilot site.

3. VIRGINIA BEACH

The Virginia Beach City Public Schools (VBCPS) serve approximately 70,535 students in grades K-12. The school system includes 57 elementary schools, 14 middle schools, 11 high schools, and a number of secondary/post-secondary specialty centers, including the Advanced Technology Center, Technical and Career Education Center, Center for Effective Learning, Virginia Beach Central Academy, Adult Learning Center, and Open Campus. In addition, the school division presently operates several academy and magnet school programs. In high school, there is a Health Sciences Academy, a Global Studies and World Languages Academy, a Legal Studies Academy, a Technology Academy, a Visual and Performing Arts Academy, a Mathematics and Science Academy, and an International Baccalaureate Middle Years and Diploma Program (Princess Anne High School, PAHS). Harrison, co-author of this paper, is a computing teacher at PAHS.

Demographics: The VBCPS demographics show a culturally and racially diverse population with approximately 56% of the students Caucasian, 28% African-American, 6% Hispanic, 6% Asian, and the remaining 4% divided among Pacific Islanders, and Native American students. Approximately 49% of the students are female. Approximately 27% of the students are considered economically disadvantaged. The 11 high schools reflect the local neighborhoods and vary significantly in cultural makeup, ranging from 80% Caucasian and 11 % African American to 44% Caucasian and 37% African American.

CS Course Offerings – Background: VBCPS offers Computer Programming, Advanced Placement CS, International Baccalaureate Standard Level (IBSL) CS, and International Baccalaureate Higher Level (IBHL) CS. All courses satisfy graduation requirements in Mathematics with the AP and IB CS Courses satisfying graduation requirements for the Virginia Advanced Diploma.

CS courses have been offered by VBCPS at the various high schools since the early 1990s. IBSL and IBHL CS were introduced in 1999 and 2003, respectively, with AP CS following the transition from Pascal to C++ in 1998 and from C++ to Java in 2003.

The computer programming course has varied greatly in content and purpose. In the late 1990s the course consisted of one semester of Qbasic followed by one semester of C++. For the 2000 - 2001 school year, the course transitioned to a full year of C++, emphasizing procedural programming and numerical solutions.

Computer Programming Course – a Ten Year History at Princess Anne High School, PAHS: When computer

programming was first taught in the 1999 – 2000 school year, approximately 25 students in two sections were enrolled in the course. Of these 25 students, only about one-half chose to continue the study of C++ during the second semester. Discussions with other computer programming teachers in VBCPS indicated a similar trend.

With the transition of the AP CS Course from Pascal to Java in 1998, VBCPS decided to change the computer programming curriculum to a full year course in C++. The course was taught at a much slower pace than the AP Course, covered one-dimensional arrays as the data structure, and emphasized numerical computation using real world problems. During the first three years, the number of students enrolled in computer programming rose dramatically, peaking at nearly 100 students at PAHS in the 2002-2003 school year. During this time, AP CS numbers remained fairly constant also, with about 20 students studying some combination of AP and IBSL CS.

Following the 2002-2003 school year, computer programming and AP CS enrollment dropped significantly, not only at PAHS, but across the VBCPS division. The decrease in enrollment was attributed to three main factors: transition to Java in the AP Curriculum, the number of students attempting to earn the Virginia Advanced Diploma by taking Algebra II (computer programming does not satisfy the Advanced Diploma requirement until the student has completed Algebra II), and the perceptions of students and parents that (1) CS is hard and (2) there are no jobs in CS. The 2004-2005 school year saw barely 30 students enrolled in computer programming, 10 in AP CS, and 5 in IBHL CS at PAHS. Numbers were similar at other high schools across the division. VBCPS CS teachers recognized the need to make significant changes in the curriculum to save CS in Virginia Beach.

Fortunately, 2006 marked the beginning of the textbook adoption cycle for Computer Programming. The textbook adoption committee, supported by the Mathematics Instructional Specialists, recognized that there was a unique opportunity to change the structure of the Computer Programming course so as to address the needs of students who desire to pursue the entire CS curriculum as well as those who have completed Geometry and need an additional mathematics credit to satisfy their graduation requirements. The committee decided that C++ was not the answer to the problem.

First, the committee looked at using Java in the computer programming course, perhaps with the BlueJ development environment, [1] focusing on more interesting but less mathematically challenging problems. Karel J Robot [2], among other approaches, was examined and rejected for the course. GreenFoot [11], at that time, was considered too immature, and there was little textbook support. Finally, the committee came across Alice, described at [5, 6, 7] with the results of a college-level study available from [13]. Alice seemed to be a potential answer to this dilemma. With Alice's drag-and-drop interface, the syntax errors that plague and frustrate beginning programmers were eliminated. Interesting problems could be explored from the beginning of the course. Complex animation was possible with the built in Alice commands. Interactive programming using the mouse and keyboard was easy, allowing students to create interactive programs without learning about listeners, events and other complex constructs.

The four members of the VBCPS textbook adoption committee attended an Alice workshop, which confirmed opinions concerning using Alice at the high school level as the first programming course. Upon consultation with Cooper and Dann, the VBCPS teachers were invited to be the pilot testing ground for the proposed ITEST grant. The goal was to provide faculty professional development in the form of Alice workshops and curriculum development workshops for high schools in Virginia Beach, and to use this as a model for the remaining five sites.

4. THE COURSE

The course and curricular materials developed at the Virginia Beach pilot session are based on the entire Learning to Program with Alice [7] textbook and approach. Chapter 8, Recursion, was specified as an optional unit in the curriculum. Approximately 27 weeks are allotted to cover the 11 chapters of [7]. This was planned to allow for 9 weeks to explore Java using BlueJ. The Java exploration focuses on the BlueJ Shapes and Picture project, introducing students to the Java programming constructs of selection and repetition using a familiar but significantly different interface. (It was decided to make the students' Java experiences as visual as possible, after having worked with the visual Alice for the major portion of the school year.) Most students quickly grasped the similarities, though they became frustrated with the same issues (syntax, lack of realism) that affect many beginning programming students.

The course is broken into five major sections consisting of 14 Units of instruction. Each unit covers a specific set of programming concepts or constructs, with spiraling built-in to refresh student understanding of previous topics. Programming concepts are presented using PowerPoint slides and illustrated with Alice program segments. Students then try these concepts using the examples developed in class before moving on to additional programming exercises. Each unit concludes with a programming project of greater scope and complexity than the programming exercises. Students construct either visual or textual storyboards depicting their program before actually writing the program and present their project to the class. This reinforces problem solving and algorithmic thinking, which are two major concepts of the course. VBCPS high school courses are taught on the alternating day block schedule. Each class meets alternate days for 95 minutes for the entire school year. Students are provided ample time to work on their programs in class. A brief course description follows:

Section 1 is an Introduction to Alice. *Unit 0 – Introduction to Computer Programming, History of Computing*: In this unit, students are introduced to the discipline of CS. They explore the history of the subject by constructing timelines of hardware and software development, discuss computer ethics, and research famous computer scientists. *Unit 1 – Getting Started with Alice*: Students explore the Alice programming environment by completing the four tutorials and appendix exercises. They are exposed to a number of computer programming terms, storyboarding and modify existing Alice Programs. *Unit 2 – Program Design and Implementation:* Students create storyboards to solve a variety of problems. They are introduced to simple Alice statements and the concept of a method. Working in teams of two, they develop a storyboard for a more complex scenario and present their solution to the class. Students then use their storyboards to create their first Alice program. *Unit 3 – Putting the*

Pieces Together: Students are introduced to control structures such as If/Else and Loop statements and functions. They use these structures and built-in functions to create storyboards and Alice programs. Student teams then complete and present a storyboard design and program as the culminating exercise for this unit.

Section 2 introduces object-oriented concepts. *Unit 4 – Classes, Objects, Methods and Parameters*: Building on the concepts developed in units 2 and 3, students more fully explore the differences between a class and an object. The concept of passing information to a method using parameters is developed using the built-in methods for each object. Students create their own methods and functions as part of the problem solution. They create a Halloween themed project and present their storyboard and project to the class. *Unit 5 – Interaction: Events and Event-Handling*: Students are introduced to the concepts of interactive programming using mouse and keyboard events. They create interactive programs and develop unit projects that require user interaction.

Section 3 focuses on functions and control statements. *Unit 6 – Functions and If/Else*: Students use functions to control decision making in If/Else statements. Incorporating previous concepts, the programs and projects become increasingly complex. Storyboards start to become more textual and algorithmic in nature due to the non-linear program flow as interactivity and decision making are added. *Unit 7 – Repetition: Definite and Indefinite Loops*: Students use both for and while loops to create more interesting programs. The concept of using functions to control the repetition either as a counter in the for loop or condition in the while loop is explored. Student projects continue to increase in complexity.

Section 4 covers advanced topics. *Unit 8 – Recursion (Optional Unit)*: Students are introduced to recursion using classic problems such as the Towers of Hanoi or game play where the number of moves is not known at the beginning. Students explore the concepts of base case and recursive case and discuss the pros and cons of recursive solutions. *Unit 9 – List Processing*: Students are introduced the concept of list data structures by creating lists of objects. Using the built-in Alice programming statements for-all-in-order and for-all-together, students process lists of objects. The advantages of using a list instead of multiple individual objects are discussed and illustrated. *Unit 10 – Variables and Revisiting Inheritance*: Students are introduced to mutable variables and use these variables as scorekeepers or timers. The concept of inheritance is revisited from earlier chapters where student created methods are added to an object to create a more capable object. This new object may be saved and used in later projects.

Section 5 transitions to Java. *Unit 11 – Transition to Java*: Students use the Java Style of the Alice programs to begin to explore Java syntax. They are introduced to the BlueJ Java programming environment and explore simple graphical Java programs. Students add constructors and create new programs using existing classes. *Unit 12 – Understanding Class Definitions*: Students explore existing Java classes and related Java concepts to Alice concepts. Object state, methods and functions are explored as are the programming constructs of decision-making and repetition. Real world examples are used to illustrate these concepts. *Unit 13 – Object Interaction*: Students explore more complex situations that require multiple objects in the solution. They explore how these objects interact by passing information to solve the problem.

The interested reader is invited to visit [17] to see the entire course curriculum. The zip files contain a course plan (including a schedule), unit plans, lesson plans, power points, laboratory exercises, project assignments, exams, and grading rubrics. Individual teachers at different high schools taught slight variations of the course, available from [17], but all covered nearly identical content.

5. RESULTS

The new Computer Programming Course is now in its fourth year, confirming the enduring effects of this project at Virginia Beach. PAHS is maintaining a solid 40 to 45 student enrollment in two sections. Many of these students are seniors who may have only completed Geometry. Classes are typically 35 – 50 percent female, 10 – 15 percent African American with a few high achieving ninth and tenth grade students. In addition, a small number of students with various learning challenges, such as those with Asperger's syndrome or other high-performing types of Autism, have participated in the course. All ten high schools are showing similar gains. All but one school has multiple sections of Alice, with marked increases in female and minority student enrollment. Two high schools, Kellam and Kempsville, offer four or five Alice sections, attracting those students who have struggled in mathematics courses by offering them the opportunity to fulfill their mathematics requirements in an interesting but challenging manner.

The new computer programming course is also showing some positive impact on the AP and IB CS courses. At PAHS, enrollment in AP CS is up for the second year, increasing from nine students in 2006 – 2007 to 18 in 2007 – 2008 and now 33 in 2008 – 2009. Four Computer Programming students were among the 18 AP Students in 2007 – 2008 and five are among the 33 in 2008 – 2009. While not large numbers, the success of the Alice project seems to be carrying over to attract students to AP CS.

6. CONCLUSION

The ITEST pilot project has been remarkably successful. In the third year after introducing Alice across the school district, all ten high schools are continuing to use Alice, offering significantly larger numbers of sections to significantly larger numbers of students. While the number of students taking the AP CS exam A is flat nationwide [22], there is a significant increase in the Virginia Beach schools.

Several lessons have been learned. (1) The first is that it would be extremely helpful if the secondary schools were close (geographically) to the university partner. While there were several advantages to having one of the original Alice researchers running the workshops, the 300 mile distance between Virginia Beach and Philadelphia meant that follow-up mentoring contacts were limited to once every 3 to 4 weeks, and it was not possible for students from Saint Joseph's University to work closely with the Virginia Beach teachers, to help them to build virtual worlds for use in the high school classes. (2) To disseminate the results of the workshops and of the course and of the teacher experiences to other Virginia high school teachers, it would have been helpful to have Virginia State Department of Education officials involved in the workshops or in the implementation of the project. (3) One of the side highlights of this project has been the community building within the Virginia Beach computing teachers. Often, there is only one computer teacher for an entire high school. That teacher is

typically responsible for a different subject (in Virginia Beach, most of the computer teachers were also mathematics teachers), and would teach four preparations (introductory computing, AP computing, and two mathematics preparations) among the five classes. This higher number of preparations and being the only computer teacher often leads to an increased sense of isolation. Getting together with other teachers to discuss Alice every few weeks was beneficial to the teachers' sense of community.

Despite the successes of the Virginia Beach pilot program, several challenges for the future are anticipated. (1) All of the Virginia Beach teachers were also either mathematics or science teachers, and most had previous programming experience. Workshops that are held after this project concludes must be cognizant that many of the high school teachers will likely be business teachers, with no previous programming experience and, indeed, with very little prior STEM exposure. The movement of computing (as taught within high school) from the college-track to the career and technical track practically guarantees that this will be the case. Thus it is likely that the pace of the presentation of the Alice content within the workshops will need to be slower. (3) Future workshops will include teachers from multiple school districts (where all of the teachers at the Virginia Beach workshops taught at schools within the Virginia Beach School District). Thus, it will likely be more difficult for the teachers to develop the same sense of community that has been developed as part of the pilot. (4) There will likely be challenges moving to school districts that are more urban and rural. For example, computer access (both inside and outside of school) will likely be a challenge in many rural schools. (5) It will be essential to involve state Department of Education officials involved in the projects, as they have an important role to play in the dissemination of results throughout their states. (6) There is also a need to build a community of presenters, teachers who will be able to present Alice workshops throughout their state, and assist other teachers with the teaching of Alice. There are over 30,000 high schools in the United States, and offering workshops that "teach the teachers" will be essential to successfully disseminate project results.

7. REFERENCES

[1] Barnes, D. and Kolling, M (2009). *Objects first with Java: A Practical Introduction Using BlueJ*. Prentice Hall.

[2] Bergin, J., Stehlik, M., Roberts, J., and Pattis, R. (2005). *A Gentle Introduction to the Art of Object-Oriented Programming in Java*. cafepress.com.

[3] Bruce, K., Danyluk, A., and Murtagh, T. (2005). *Java: An Eventful Approach*. Prentice Hall.

[4] Buechley, L., Eisenberg, M., Catchen, J., and Crockett, A. (2008). The LilyPad Arduino: using computational textiles to investigate engagement, aesthetics, and diversity in computer science education. In *Proceeding of the Twenty-Sixth Annual SIGCHI Conference on Human Factors in Computing Systems* (Florence, Italy, April 05 - 10, 2008). CHI '08. ACM, New York, NY, 423-432.

[5] Cooper, S., Dann, W., and Pausch, R. (2003). Teaching objects-first in introductory computer science. In *Proceedings of the Thirty-fourth SIGCSE Technical Symposium on Computer Science Education*, Reno, NV, February.

[6] Cooper, S., Dann, W., and Pausch, R. (2003). Using animated 3D graphics to prepare novices for CS1. *Computer Science Education*, 13(1).

[7] Dann, W., Cooper, S., and Pausch, R. (2005) *Learning to Program with Alice*. Prentice Hall.

[8] Felleisen, M., Findler, R., Flatt, M., and Krishnamurthi, S. (2001). *How to Design Programs: An Introduction to Computing and Programming*. MIT Press.

[9] Figueroa, E. and Woods, R. (2007). Employment Outlook: 2006-2016. Industry output and employment projections to 2016. *Monthly Labor Review*, 130(11):53–85.

[10] Guzdial, M. (2005). *Introduction to Computing and Programming in Python: A Multimedia Approach*. Prentice Hall.

[11] Kölling, M. (2008). Greenfoot: a highly graphical ide for learning object-oriented programming. In *Proceedings of the 13th Annual Conference on innovation and Technology in Computer Science Education* (Madrid, Spain, June 30 - July 02, 2008). ITiCSE '08. ACM, New York, NY, 327-327.

[12] Malan, D. J. and Leitner, H. H. 2007. Scratch for budding computer scientists. *SIGCSE Bull.* 39, 1 (Mar. 2007), 223-227.

[13] Moskal, B., Lurie, D., and Cooper, S. (2004). Evaluating the Effectiveness of a New Instructional Approach. In *Proceedings of the Thirty-fifth SIGCSE Technical Symposium on Computer Science Education,* pages 75-79. ACM, Mar.

[14] Sanders, D. and Dorn, B. 2003. Jeroo: a tool for introducing object-oriented programming. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education* (Reno, Navada, USA, February 19 - 23, 2003). SIGCSE '03. ACM, New York, NY, 201-204.

[15] Vegso, J. (2005). Interest in CS as a major drops among incoming freshmen. *Computing Research News*, 17, 3, 6-1.

[16] http://tecs.acm.org/ Accessed 8/27/2009

[17] https://www.aliceprogramming.net/ITEST/vb/index.html Accessed 8/27/2009.

[18] http://www.nsf.gov/pubs/2009/nsf09506/nsf09506.htm Accessed 8/28/2009.

[19]http://csta.acm.org/ProfessionalDevelopment/sub/TeacherWorkshops.html Accessed 8/26/2009.

[20] http://www.cra.org/CRN/issues/0905.pdf Accessed 8/29/2009.

[21]http://www.csta.acm.org/ComputerScienceTeacherCertification/sub/State_Certification_Files/requirementsbystate.html?CFID=42722&CFTOKEN=78079580 Accessed 8/26/2009.

[22]http://professionals.collegeboard.com/data-reports-research/ap Accessed 8/28/2009

[23] http://visualization.sju.edu/grants.html Accessed 8/27/2009

[24] http://www.alice.org