BINARY OPPOSITIONS:
SHOULD AN INTRODUCTION TO COMPUTING IN ARCHITECTURE BE TAUGHT AS A SEPARATE COURSE?

Column by: Scott Johnson, sven@umich.edu
Contributing author: Glenn Goldman, goldman@njit.edu

For quite some time, the traditional teaching approach taken by most architecture schools has been to separate design studios from other courses. New courses have occasionally been added, to cover subjects not previously included in the curriculum. However, as technologies change and new, computer-based design tools are developed, it is worthwhile to consider whether these new technologies should be introduced into the curriculum in the same way. Should new courses be added to the curriculum, to supplement or replace courses covering traditional tools and media? Or is this unnecessary or even inappropriate for the new technologies?

This Binary Oppositions debate addresses this issue. The question is, “Should an introduction to computing in architecture be taught as a separate course?” I argue to the affirmative. Glenn Goldman of NJIT argues to the negative. These arguments and our respective rebuttals are presented below.

Yes:
A Distinct Subject Deserves a Distinct Course
Scott Johnson, sven@umich.edu

If we are lucky, and if we devote enough effort, we may someday have Computer-Aided Design systems that are so natural that they require no special instruction. We may have CAD systems so integral to design that it is impossible to distinguish between design topics and CAD topics. Unfortunately, that day has not yet arrived.

For the time being, CAD topics form a separate body of knowledge, and as such, are best presented in a separate course, with its own lectures and demonstrations, its own time for lab exercises and projects, and its own grading criteria. These CAD topics include concepts like vector vs. raster graphics, layers, differences between symbol instances (BLOCK INSERTions) and copies of graphic entities, principles of perspective viewing, and principles of solid modeling, as well as mere software commands. (While the primary emphasis of a course needs to be on general concepts instead of on a particular program and its commands, it is difficult to teach general concepts in abstract terms alone—some command syntax must be covered.) There is enough CAD material to fill at least one and perhaps two or three courses, even without delving deeply into topics like non-graphical applications or computer programming.

An argument might be made that some CAD topics—e.g., vector-based graphics, layers, semi-automatic dimensioning, etc.—relate to drafting, and therefore should be taught as part of a course covering drafting in general. I prefer grouping computer drafting topics with other CAD topics, but it is also reasonable to put such topics in a drafting course, provided that some other course is devoted to the remaining CAD topics. Trying to incorporate introductory CAD topics into a course like a design studio is more of a problem, however.

The ultimate purpose of CAD software is to improve the process and product of design, but that doesn’t mean that an introduction to computing should be taught in a design studio. Freehand drawing and architectural history are likewise intended to help students design better, but they are generally taught as separate courses, and rightly so. The same can be said for courses on structures, construction methods, architectural programming, environmental psychology, and many other subjects. Each coherent subject has its own course (or sequence of courses); conceptually, this is the reasonable way to divide up the curriculum.

It is also reasonable pragmatically. Integrating CAD with studio necessitates either hiring two different instructors to teach a single course, or finding a single instructor who excels at teaching both design and computing. Furthermore, it means adding an entire course worth of CAD material to a studio course. Studio courses are already the most time-consuming courses in the curriculum; can a semester’s worth of computing topics really be added without adversely affecting the time and attention given to design topics?

It might be argued that if students are to learn how to use computers in design, the logical place to teach them is in a design course. This is a good approach for later courses, but not for the first one or two courses on computing. There is a need for a course covering topics like strategies for efficient use of computers in design, how to divide projects among several people, how to share data, and so forth. But such a course assumes that students already understand the basics of computer use. Before students can learn how to share and manage symbols and models, they need to understand how to make symbols and models. Because of this, courses combining studio and computing are best taught after students have already learned the basics of computer applications in design.
No:
Computing in Architecture Is Best Learned by Students in Those Courses Where Students Are Using Computing

Glenn Goldman, goldman@njit.edu

To be sure, computing is a discipline unto itself demanding study and research. And the theory of computing and computer programming are, perhaps, best studied as ends unto themselves. However, for a variety of reasons (some pedagogical, some expedient) when learning about computing in architecture, it often is best done on a “need to know” basis within the various courses already existing.

Computing is ubiquitous in our lives and profession. It informs and aids in almost everything we do, from design to analysis to communication. Although the technology is not necessarily simple, the use of computers for many tasks certainly is. Virtually all students at the university level (and many younger) already communicate with one another and with faculty via e-mail. Word processing and presentation software (like Microsoft PowerPoint) are used by students in both public and private elementary schools. In the past, if a student needed to make a linear presentation and had not used a slide projector, the instructor of the class for which the presentation was made showed how. Similarly, the use of presentation software can be explained with a five-minute demonstration before or after class. Analysis software for energy use has been applied to courses for many years without separate courses or programming. The instructions to use any tool that has a specific application (structures, energy use, duct sizing, and so on) are typically provided at the time the tool is needed by whomever requires or recommends the use of the particular tool. Creating a separate course to teach these skills is solving a problem that, by and large, does not exist. Furthermore, given a practical upper limit on the number of credits contained within any program, and the constant pressure to include more and more information in a professional degree, there is simply no room for this additional course.

My Rebuttal

Computer use in service of design (and the design studio) is one of the applications most frequently mentioned as needing a separate prerequisite course. Even if a program has room for a separate course teaching skills, it is not necessarily the best solution to the problem. Architecture students generally put a large portion of their efforts into studio. (At many schools, courses are either officially or unofficially divided into studio and “support courses.”) It takes TIME to become graphically proficient with any medium and time and effort are most readily available in the design studio—or a “graphics lab” that is equivalent to a studio. Students must be afforded sufficient time to learn and practice and explore; and a simple lecture course does not afford that kind of time to draw well with digital—or traditional—media. Like learning to play a musical instrument (which is generally not accomplished in a lecture course) students learn best by doing.

Beyond expediency, there are pedagogically sound reasons to integrate architectural computing into the design studio. The computer can change the design process itself as it offers new types of abstractions. Students can more readily visualize a proposal in context—throughout the entire schematic design phase. The computer allows students (and critics) to visualize a project serially, as a user might experience it. Designers and critics can iteratively test time-dependent phenomena like time of day or season while designing. Color and materiality can be explored. The computer is not only a tool for representation; it is a tool for designing. Image processing, painting, modeling, rendering and animation can all be integrated into the presentation and design processes. And, unless one integrates the computers into the design process itself—and this is reinforced by the design studio critic—students may miss some opportunities to understand the capabilities of digital design processes. With care in the preparation of a design studio program, one can design a studio sequence (that) will afford students the opportunity to learn about the use of different applications while designing. All projects can have design components that simultaneously further the goals of the design studio as well as teach beginning students “how to” accomplish a particular task. Each project (and there are many available) can be designed to address a particular design issue and require students to learn a particular application or portion of an application. Students then will learn what they need to know. And like any design studio situation, they will learn from, and help, one another. By integrating CAD/graphics instruction into a studio, students become comfortable with the computer as a design and presentation tool, and will use their creativity to push the medium to its limits.

**Freshman design studio at NJIT (Spring 2001). Photo by Glenn Goldman.**
Image processing, drafting, modeling, rendering, and animation software can all be integrated to some degree into the design and presentation processes, but until students gain a certain level of proficiency with these tools, the tools will be more of a hindrance than an aid. However, the converse is not really true: students can learn how to use visualization tools even before they are able to design well.

Most visualization tools take much more time to learn than a slide projector or even the most feature-laden word processor. To learn them well, students need instruction. They need demonstrations of the software, and they need faculty, TAs, and/or staff available to help them overcome problems they encounter. Furthermore, students need an explanation of the concepts underlying the software, so that the software makes sense and behaves predictably. This lets students understand and correct problems that occur, and better understand later versions of the software and competing products.

Such services cannot typically be provided in a studio environment. Even if the entire faculty for a given level of studio was proficient at teaching software meant for design, they could not devote time to demonstrating software, explaining concepts, or helping students overcome technical problems, without taking time away from teaching design.

It does indeed take a lot of time for a beginning student to learn to use software for design, but taking that time from studio can harm both subjects.

Glenn Goldman's Rebuttal
Life is getting more complex. The courses we teach now must contain more material than ever before. While I have sympathy for the idea of trying to get more information to students in more courses, the realities of curricular and institutional requirements make this difficult. Students must still have opportunities to broaden their educational experiences with elective courses and concentrations. The fact is that adding another required course to what is already a packed program is rarely possible—and is not desirable.

There is room in design studio to teach architectural computing when students have six to ten semesters and are spending upwards of 13 contact hours (plus innumerable work/study/design hours) each week in studio. Simply put, there is more time for teachers to teach, and for students to learn and practice.

Schools should not have to hire two faculty members to teach studio with computers. We generally don’t hire one teacher who can draw (with traditional media) and another for design. The studio critic is expected to be able to assist students in those areas where assistance is needed in graphic communication/presentation or design. The lack of qualified instructors at some institutions should be a short-term transitional phenomenon.

Finally, the intellectual content of some of the skills and tasks that students need to learn simply don’t warrant a separate course. “How to” skills—especially those that are software specific (not to mention version specific) should be placed at the point of need, quickly and expediently, so as not to make such a big deal of relatively simple topics. The CAD/graphics knowledge and skills needed to design should be integrated at the very beginning of a student’s design career, where he/she can learn to use them as an integral part of the process and not as a separate “add-on.”

Final Comments
Should an introduction to computing in architecture be taught as a separate course? Glenn Goldman and I take opposing positions on the question. We start with a mutual goal of providing architecture students with the highest quality, most relevant education possible. We both wish to ensure that students and teachers can devote adequate time to CAD instruction. Yet, we diverge in our opinions on how to best achieve these goals.

I argue that design visualization tools and the concepts underlying these tools constitute a distinct subject. I contend that the best way to devote sufficient time and attention to this subject is through a separate course or series of courses. Glenn Goldman, on the other hand, argues that design studio is the logical place to teach tools for design, especially given the lack of room for additional courses in most architecture degree programs. He contends that the best way to allow students sufficient time to practice computing in architecture is to integrate the lessons into design studio.

The task of considering these arguments and weighing their merit is left to the reader. It is hoped that this column will prompt further thought on the issue and generate discussion among colleagues and students.

The “Binary Oppositions” column is intended as a forum for the informed debate of controversial CAAD topics like this one. Feedback and ideas for future columns are actively encouraged. If you have comments or ideas, or if you would like to volunteer for future debates, please contact me via e-mail at sven@umich.edu.

Scott Johnson is a Candidate in the Doctoral Program in Architecture, Taubman College of Architecture and Urban Planning, University of Michigan.

Glenn Goldman is a Professor of Architecture and Director of the Imaging Laboratory at the New Jersey Institute of Technology.