

# Adapting Technology to Teach Koine Greek

Rodney J. Decker, Th.D.

Associate Professor of New Testament

Baptist Bible Seminary, Clarks Summit, PA

Society of Biblical Literature Annual Meeting

Biblical Greek Language and Linguistics Section

November 18, 2001 (S18-5)

Denver, CO\*

## Introduction

In today's cultural vortex of rapid technological change, half-truths and outright falsehoods swirl around us, buffeting our common sense while lifting our imaginations to new heights of revelry. Every day we hear of the promises of cyberspace, the possibilities of the Internet, and breakthroughs in all manner of computer technologies.<sup>1</sup>

Technophiles wax messianic over new devices and bedazzle their auditors with “breathless prophecies of social regeneration” through the latest technology.<sup>2</sup> Cyberspace has been nearly deified.<sup>3</sup> There have been utopians throughout history and contemporary technology has now produced its share of “digitopians.”<sup>4</sup> On the other hand, there is no

---

\* I want to acknowledge my indebtedness to K. C. Cheng, the Distance Education Coordinator at Baptist Bible College and Seminary, for his invaluable expertise in helping me solve some of the technical problems with the presentation that accompanies this paper as well the ongoing support he provides for both our Internet classes and the web-enhanced resources for our resident courses.

<sup>1</sup>Douglas Groothuis, *The Soul in Cyber-Space* (Grand Rapids: Baker, 1997), 9. *Cyberspace* was coined in the 1984 novel *Neuromancer* by William Gibson. It refers to “the ‘space’ in which computer-mediated communication occurs, that is, to the interface between digital bits and human consciousness—or between silicon and the soul” (ibid., 13–14). The first two paragraphs of this paper have been adapted from my earlier article, “Communicating the Text in the Postmodern Ethos of Cyberspace: Cautions Regarding the Technology and the Text,” *Detroit Baptist Seminary Journal* 5 (fall 2000): 45–70. An earlier version of the same article may be found on my NT Resources web site: <[http://faculty.bbc.edu/rdecker/rd\\_rsrc.htm](http://faculty.bbc.edu/rdecker/rd_rsrc.htm)>.

<sup>2</sup>Ibid., 10–11.

<sup>3</sup>Technology has become a god “in the sense that people believe technology works, that they rely on it, that it makes promises, that they are bereft when denied access to it, that they are delighted when they are in its presence, that for most people it works in mysterious ways, that they condemn people who speak against it, that they stand in awe of it, and that in the born-again mode, they will alter their lifestyles, their schedules, their habits, and their relationships to accommodate it. If this be not a form of religious belief, what is?” (Neil Postman, *The End of Education* [NY: Knopf, 1995], 38). Slouka's reaction is similar: “The literature of cyberspace, I now began to see, was all about salvation. The new, electronic millennium. Transcending time and space, the family and the body” (Mark Slouka, *War of the Worlds: Cyberspace and the High-Tech Assault on Reality* [New York: Basic Books, 1995], 29).

<sup>4</sup>One of the most obvious examples of a digitopian is Nicholas Negroponte, *Being Digital* (New York: Random House, 1996). A volume that would be worth reading in this regard is *Responsible Technology*, ed. Stephen Monsma (Grand Rapids: Eerdmans, 1986). Although written when cyberspace was only an infant, and long before the “web” was envisioned, it presents a serious Christian critique of the underlying

end to the nay-sayers (both secular and religious) who are convinced that Armageddon is just around the corner—cyberspace is the kingdom of antichrist. But we need to view all this carefully before we adopt the panaceas that will certainly bring in the kingdom—or before we retreat to non-technological enclaves to await our deliverance.<sup>5</sup>

My purpose in this paper is not to provide a philosophical critique of technological issues, though I certainly realize that they exist. Nor is it a training manual to teach you how to implement the various technologies that are mentioned. My goals are rather to demonstrate some of the potential benefits in adapting current technology for use in teaching Greek on the one hand, and, on the other, to suggest a few of the pitfalls that might be encountered by those who chose to explore such means. This paper is neither a listing of nor a review of all possible avenues, but a representative sampling that will hopefully suggest some creative means to enhance a common end: introducing students to the language of koine Greek, particularly the language of the New Testament.

Let me say at the outset that no one should assume that the use of technology is a pedagogical panacea. It has some unique advantages, but it also comes with inherent complexities, some of which can be counterproductive. It also comes at high cost, both in terms of the time required to implement and maintain such systems, as well as the significant investment in hardware and software up front and the ongoing cost of upgrading the same. Some technologies are not compatible with individual teaching styles and there is no guarantee that even if they are compatible, that they will produce a corresponding increase in the student's mastery of the material.

The use of some technological enhancements in teaching Greek does have the potential in some situations to improve both teaching and learning. For example, the use of (static) graphics, video, and animation in addition to traditional text-based approaches have correspondingly greater potential for effective learning since multiple cognitive pathways are involved.<sup>6</sup> The following multimedia techniques, all of which are adaptable

---

technologies and philosophies involved. The authors argue that “this drive for human autonomy and mastery apart from God and his will manifests itself in what we will call *technicism*. Technicism reduces all things to the technological; it sees technology as the solution to all human problems and needs. Technology is a savior, the means to make progress and gain mastery over modern, secularized cultural desires” (49). This is also the theme of Jacques Ellul's *The Technological Bluff* (Grand Rapids: Eerdmans, 1990); see also David Wells, *Losing Our Virtue* (Grand Rapids: Eerdmans, 1998), 23–25.

<sup>5</sup>“A good long squint and some head-scratching directed at the emerging world of cyberspace may equip us to make some wise choices while we yet have choices to make. From a Christian perspective, such rumination is not merely an academic exercise: It forms the heart of biblical discipleship. Followers of Christ have always lived with the creative tension of being in, but not of, the world system. They are citizens of heaven, yet emissaries of Christ on earth. As such, their pattern of life must resist the corruption and coercion of sinful ways of life in order to honor their Sovereign.” And again, “This does not mean one must regard every new technology as the invincible advance of Antichrist or as another Tower of Babel. We need not be reactionary Luddites, who want to smash new technology simply because they alter our forms of life. Human ingenuity in subduing creation, including technical facility, is part of what it means to be made in God's image.” Later in his helpful book, Groothuis points out that “We should aim to be wise skeptics who realize that something is wrong with everything in a fallen world, that things are rarely as good as they seem initially, and that finite and fallen knowers can never accurately predict all the effects of a new mode of life” (Groothuis, *Cyber-Space*, 19, 20, 53).

<sup>6</sup> Nishikant Sonwalkar, “Changing the Interface of Education with Revolutionary Learning Technologies,” *Syllabus* 15.4 (Nov. 2001): 12.

to use in both the classroom and in an Internet/web environment, suggest some potential examples in which Greek pedagogy might profitably be enhanced.

- Audio annotations to graphics
- Graphical visualization
- Audio annotations to video demonstrations
- Video demonstrations of graphical elements
- Animated graphical frames (animated gifs)
- Audio annotations for animated graphics
- Animation of physical [linguistic?] concepts
- Text annotations to video frames<sup>7</sup>

In all of these techniques there must be a text-based interpretive framework to provide content. Multimedia in itself is of little educational value apart from such a basis, for then it becomes simply entertainment. But such enhancements can be sound pedagogical tools that can help a teacher explain and illustrate that content.

We should also be cognizant of the context in which we attempt to implement such technological endeavors. What may be realistic for use in the setting of North American or European affluence is frequently not realistic in third-world countries. Even the school with the most meager of budgets for technology is remarkably wealthy compared with some of our fellow teachers in other countries. In a discussion this past summer on the b-greek list, a query regarding using technology brought a helpful reminder from Dr. Stephanie Black who teaches Greek in Ethiopia. For her, *chalk* is a helpful technology—and even that is rationed!

The key factor in successfully teaching koine Greek is not in the technology used, but in the teacher's ability to communicate complicated, non-native concepts to students in a manner that is both clear and compelling. Clarity and motivation are far more important than technology. Greek pedagogy will not be improved simply by throwing money at a department saddled with ineffective teachers. A good teacher, by contrast, will make the best of any tools available, whether that be a piece of chalk, a video projector, a PalmPilot, or an electronic whiteboard.

## Background

Although I would normally hesitate to do so in a conference paper of this sort, I think that it would be helpful for you to know something of my experience in attempting to adapt technology to teach Greek. I share the following *vita* with you, not to boast of my own work, but so that you know either how limited my perspective may be (if you happen to have done much more than I have in this area),<sup>8</sup> or so that you can determine

---

<sup>7</sup> Ibid. Sonwalkar's list includes several additional items that do not seem to me to be useful in teaching koine Greek in the usual written/read format, though they would be to language courses that teach an oral/conversational level of the language.

<sup>8</sup> I well remember the experience of a noted NT scholar at an SBL textual criticism section meeting a few years ago. He demonstrated how he had adapted technology for use in NT textual criticism. He was using, at that time, the AppleWorks database running, if I remember correctly, on an Apple IIe. The

how closely my experimentation and institutional resources might parallel your own situation.

I have taught Greek for the past eleven years at two different institutions, both at the undergraduate and the graduate level. My present position involves teaching Greek at an unaffiliated, regionally accredited, Baptist seminary in which two and a half years of Greek are required in a three-year Master of Divinity program. I teach the first three semesters, my colleague the fourth in that sequence. We divide exegetical electives and I also teach several other Greek electives on alternate years, including an introduction to NT textual criticism. Our school is not large—the seminary program accounts for approximately 250 students in a total undergraduate and graduate enrollment of 1,100. I thus speak from a context quite different from those of you who may work in a university setting.

My first venture into adapting technology was in the early 90s as I designed and installed a new biblical languages lab built around Accordance on a Macintosh network, though we also grandfathered some older DOS equipment that ran GRAMCORD on dual-floppy 80286s. In the classroom I installed a Macintosh-controlled grayscale LCD panel on an overhead projector.

When I moved to Pennsylvania to teach six years ago I inherited a similar system, though the equipment consisted of a color LCD active-matrix panel running from a Windows box. In the past few years we have gradually moved to video projection systems, initially a single portable unit, now to multiple classrooms with ceiling-mounted projectors, electronic white boards, and live Internet connections, all controlled from permanent teaching stations.

I have also ventured into teaching Greek as an Internet course. The first such class was offered four years ago and was implemented with standard web pages. More recently we have moved to commercial course delivery software and have twice taught the first three semesters of Greek in this format. I have also experimented extensively with using similar technology to supplement the traditional residential course.

My current semester resident classes consist of approximately twenty students in each of first semester Elements of Greek and a third semester Greek Reading course. There are an additional half dozen students enrolled in the Internet-only version of Greek Reading. These Internet students are physically located in New York, Pennsylvania, Maryland, and Florida.

## Hardware

With that background, consider the potential options, both hardware and software. The most common delivery method for teaching Greek is the traditional resident classroom. More recently Internet courses have become feasible and popular.<sup>9</sup> A third

---

following speaker then demonstrated a sophisticated program dedicated solely to textual criticism—much to the chagrin of the first speaker! I realize that I run a similar risk in this forum.

<sup>9</sup> Contrary to common belief, many Internet students are not strictly distance education students. They are more likely to be simultaneously enrolled in resident courses. They often take Internet courses for scheduling convenience rather than to overcome limitations of geographical location. This has been the experience of our Seminary's Internet course offerings and is substantiated by Steven W. Gilbert, "The Hybrids Are in Bloom," Technology Implementation column, *Syllabus* 14.6 (Jan. 2001), 16.

option is the hybrid “super course” which supplements a resident course format with a web site that adds technology more commonly associated with Internet courses.<sup>10</sup> The technologies discussed and illustrated below will have varying application in these different settings.

The most versatile piece of hardware for the classroom is probably the video projector. Anything that can be displayed on a computer monitor, whether class content or actual linguistic software, can be projected onto a screen of almost any size. If the projector is properly sized to the room, this can be done at normal room lighting levels—something that was very difficult (and expensive) to accomplish only a few years ago.<sup>11</sup> Ideally these projectors would be permanently mounted on the ceiling of the classroom, although portable solutions are feasible when necessary.

Another technical tool that has become more affordable of late is the electronic whiteboard.<sup>12</sup> Although smaller, portable models are available, to be practical in the classroom, these must be large and permanently mounted.<sup>13</sup> They look like a standard whiteboard, but are actually a large touch screen that works in tandem with a computer and video projector. The resistive touch screen transmits contact information from a plastic tipped pen to the computer which translates the location to a colored mark or line on the computer monitor as well as projects the same colored mark onto the surface of the white board itself using the video projector. The board thus serves two purposes: to record the location of the pen and to serve as the projection screen for the video projector.

This is a remarkably versatile tool for teaching Greek. The teacher can project prepared course material that is far more legible than his or her handwriting on a conventional chalkboard or white board. It is also much faster than writing by hand.

<sup>10</sup> Such hybrid courses are sometimes referred to as web-enhanced or web-enabled courses. The web component of these courses is accessed by the student outside of the regular classroom context.

<sup>11</sup> Projectors with a minimum of 1,000–1,500 lumens are necessary in classrooms of average size (1,000 lumens is the absolute minimum for a 20-student room; 1,500 will handle rooms up to about 50 students in capacity). Brighter (and more expensive) projectors are necessary for lecture halls or auditoriums. For the typical classroom, our school has found the Panasonic PT-L701U (1,000 lumens) and PT-L711U (1,400 lumens, about \$3,800.00) models to be well suited to our classroom needs <[http://www.panasonic.com/pbds/subcat/Products/mnu\\_presentations.html](http://www.panasonic.com/pbds/subcat/Products/mnu_presentations.html)>. Less powerful projectors will require that the lights be dimmed in the classroom. One solution for this is to install both regular fluorescent lighting and a separately-switched bank of down spots that put light on the desks where students need it but do not wash out the screen. For comparison, for a lecture hall, our undergraduate division uses a Panasonic PT-L6500U which is 3,600 lumens with a very high contrast—but it costs \$8,000.00. Also factor in the cost of mounting brackets and cabling for permanent installations: about \$365.00 and \$135.00 respectively.

<sup>12</sup> There are several manufacturers; our school uses the SmartBoard <[www.smarttech.com](http://www.smarttech.com)>.

<sup>13</sup> I use boards of approximately 4' x 5' in size (72" diagonal). This is adequate for my classes of twenty students, and I think this size could adequately serve up to 30 or 40 students. Screens much larger than this would be unusable along the top simply because most people can't reach that high. In early October 2001 there was a 20% cut in the list price of the SmartBoard 500 series. The board that we use (model 580) lists at \$2,000.00 <<http://www.smarttech.com/company/mediacenter/press/releases/price.asp>>. This is a front projection unit; rear projection models are available as well, but I suspect they are considerably higher in price. Note that SmartTech has “grants” of several hundred dollars for educational institutions to help offset the cost of their SmartBoards: <<http://www.smarterkids.org/>>.

Although a video projector can do that on any surface, the advantage of the white board is that it is interactive and instantly reusable. The teacher can write on the screen in a “layer” that is separate from the projected text, can then save those annotations, print them, or instantly erase them. It serves as an infinite series of blank whiteboards with the added advantage of being able to return to any previous board if desired. I use this constantly in my classes, whether by marking up paradigm charts to show similarities or differences with previous material, walking students through translation exercises, dissecting the morphology of a verbal form, showing them the discourse structure of a passage, emphasizing key points of new material, or highlighting features of manuscripts being studied. If there is both a resident and an Internet section of the class, selected screens can be saved to a graphic file and posted on the course web site for the benefit of the Internet students.<sup>14</sup>

## Software

There have been few software products designed explicitly for teaching Greek. Due to the nature of our discipline, it is frequently necessary to find creative ways to adapt general purpose software to meet the specialized needs of our craft.<sup>15</sup> Although the hardware is unavoidably expensive, it is not necessary to expend large sums of money on specialized software. There are a number of general software tools that can be used to good effect in the Greek classroom.

### Word Processors

Although many people think immediately in terms of presentation programs such as PowerPoint for classroom materials, I have found that a word processor offers much greater flexibility and effectiveness, at least given my own style of teaching.<sup>16</sup> Since most institutions will have site licenses for such software, no additional cost is involved. Our school licenses the Microsoft Office suite, so I work primarily with Word 2000, but most word processors can do similar things. My basic approach to using Word is to prepare a single document for each textbook chapter. This document contains the information from the chapter that I intend to highlight in class, complete with any appropriate charts, graphics, translation exercises, etc. Since this is projected and not printed, I can use color

---

<sup>14</sup> Other options for making this a progressive, step-by-step movie are considered below.

<sup>15</sup> Since my topic is *adapting* technology, I will not be considering the use of software designed explicitly for the biblical languages such as Accordance on the Mac or BibleWorks on Windows, either of which can be used in the classroom to good effect. Also worth noting are programs such as Mounce’s textbook CD, Parson’s GreekTutor, and PocketGreek (now called MiniFlash since as of v. 2 it also supports Hebrew; this is a vocabulary drill program that runs on the PalmOS).

<sup>16</sup> Those who spend more time working with, e.g., PowerPoint, may think that my view is jaundiced at this point! Someone who is skilled in PowerPoint can accomplish much of what I describe here, though to use this teaching style one would often have to revert to running PowerPoint in edit mode rather than in presentation mode—which defeats the purpose of using a presentation program in the first place. Once one does that, Word is explicitly designed for efficient editing and, I would argue, is the better tool for such purposes. But each to his own! Not everyone teaches the same way and not everyone uses a computer the same way, despite the similarities that are certainly present.

freely—something that enables me to highlight, e.g., each morphological piece of a verb form, the various functional parts of a sentence, etc.

After preparing this document and formatting it with an appropriately sized font,<sup>17</sup> I then select the entire document and change the font size to 8 point. This makes it essentially unreadable on screen, but it allows me to control the flow of information as I teach. As I introduce each section in class, I select the appropriate line or paragraph and change the font size to 24 point. I have assigned several function keys on the keyboard to apply specific fonts and font sizes so that I do not have to reach for the mouse and access the font menu as I teach.

I can also change font colors or sizes interactively as I proceed. This makes it easy to review or administer a spontaneous self-test as I teach: columns and/or rows of charts can be made invisible by changing the text to white, then filled in as students supply the correct information, cell by cell. It is also very helpful in showing students how to analyze a sentence, applying either colored text or highlighting to the successive parts of a sentence, and to the specific morphological markers that identify each syntactical part. Any formatting that can be applied to text in a word processor thus becomes a potential tool in an appropriate setting. Newer versions of most word processors can also embed video clips or contain live links to web pages, all accessible within the document during the class session.

An additional advantage of using a word processor instead of a presentation program is the flexibility in moving backwards or forwards through the material in random sequence as questions arise regarding material that was previously covered, or forward if the class discussion raises a point not yet covered. Such flexibility and adaptability is much more difficult with a presentation program which assumes that the sequence be planned in advance. Although there are means of moving to different parts of a presentation out of sequence, this is much more difficult than navigating a word processing document in this fashion.<sup>18</sup>

It is likewise not possible to edit text live with a presentation program which assumes that specific content, wording, and sequence is known in advance. The spontaneity of the classroom, however, constantly suggests the value of improvisation. If, e.g., a question arises regarding word order in a Greek sentence, a word processor enables the teacher to select and drag text to a different position in the sentence to illustrate the difference such a change makes or (more likely) does not make in the meaning of the sentence. This is not possible in the midst of a slide presentation unless one cancels the presentation and returns to editing mode—but that defeats the point of a presentation, to say nothing of being a very clumsy way of doing what is much easier in a word processor.

---

<sup>17</sup> A 24 point font is minimum for good legibility in projected text; 36 point may be necessary for larger classes.

<sup>18</sup> PowerPoint does have the means of hiding a certain slide or series of slides and optionally accessing them at a particular point in the presentation if they are needed, but even then, the link is from a particular slide—which may not be where the presenter is when the question arises in class! It is also possible to right click in presentation mode and pop up a list of all the slides in a presentation file and go directly to a specific slide—if one knows the title or slide number one needs. And, of course, the user can always drop back into edit mode to find the right slide. Some may prefer these avenues, but I have found Word to be the more efficient tool at this task.

The list of possibilities could go on, but perhaps these suggestions are adequate to stimulate your own creativity in using a tool that you may not have considered. I would advise that you should know your software very well before attempting to use it before a live audience in this way. The last thing you want to do is become frustrated with the software while teaching. You should be able to perform the operations described above without stopping to figure out each step in the midst of your class session. The focus should not be on the technology, but on the content of your teaching session. The technology should be as nearly transparent as possible. Never draw attention to the “gee-whiz” factor of what you can do on the screen. Use it naturally so that the students focus on what you are communicating to them, not on the means that you use to do so. A related caution: be careful that you do not go too fast. Since the technology allows you to present very large quantities of information very rapidly, do not assume that your students can keep pace. They still need the same amount of time to absorb the points you are making and to record what is necessary in their notes.

### Adobe Acrobat

An ideal supplement to a word processor when material is to be presented not only live in the classroom but also on the web is Adobe Acrobat.<sup>19</sup> This program creates a pdf file (portable document format) that can display and print a document originally created in almost any other software program without the user having that same software program or fonts installed. It is also a cross-platform solution that allows the interchange of documents between a wide variety of platforms.<sup>20</sup> This is a major advantage when students are likely to represent at least the Windows and Macintosh worlds. The Acrobat Reader is a free program that can be downloaded from Adobe’s web site. It enables anyone to view and print files created with the full version of Acrobat. I have developed all of my Internet courses around this core format for all the course content materials. Since I developed my course materials over nearly 10 years in a different font that my present students do not use,<sup>21</sup> this enables them to view the materials without the font installed on their computer. (See additional notes regarding the font issues below.)

Acrobat can also be used with other programs. (Anything that can be printed can be captured in a pdf file.) I use it, for example, with Accordance (a Mac-only program), creating Acrobat documents from search results or from the diagramming module. This enables me to distribute files from a Mac-specific program to my students, most of whom are Windows users.

---

<sup>19</sup> Information is available at <<http://www.adobe.com/products/acrobat/main.html>>.

<sup>20</sup> Platforms supported include: Macintosh, Windows, Sun Solaris, SunOS, IBM AIX, HP-UX, Silicon Graphics, IRIX, Digital UNIX, Linux, and OS/2. Not all of these may be available in the most recent version of Adobe Acrobat (v. 5).

<sup>21</sup> Since I use Bill Mounce’s excellent textbook, *Basics of Biblical Greek* (Zondervan, 1993), which now comes with a CD containing, among other software, the Greek font “Mounce,” that is the font that students use and which I use for Greek text entered on regular web pages.

## Presentation Programs

My preference for a word processor does not preclude using a presentation program, however. I do use it in my first year classes, though more selectively. I recommend that you use it for what it is good at: straightforward presentation and illustration of a tightly-knit sequence of material. I use it, e.g., to introduce verb morphology. In this instance I am trying to communicate the concept of how verbs are formed and the significance of each morph rather than expecting them to understand each particular verb form. My presentation sequence utilizes the basic animation features of PowerPoint to illustrate the concept. I then return to my word processing document to take them step-by-step through their first verb form. I do something similar when I am introducing them to the concept of contract verbs. I also tend to use PowerPoint more in later classes where I am not concentrating on teaching the elemental mechanics of first year grammar and syntax.

## Screen video recording

A somewhat more specialized piece of software (or at least less standard) is a program such as Windows Media Recorder/Player or Camtasia<sup>22</sup> which captures everything that happens on the computer monitor with optional simultaneous voice recording from an external microphone to a video file. Especially when used in conjunction with an electronic white board, this can be very effective in recording a classroom segment step-by-step, complete with progressive board annotations and instructor explanation. Although this will not have direct, in-class use as a teaching tool, it is invaluable in providing material for two other formats: first as a review tool for students who can study the clip from a class and listen to the teacher's explanation as they watch the material being illustrated, and second as a means of providing some of the traditional classroom flavor to Internet courses. I frequently record such segments in my regular class and then use them as part of the Internet class in which the students never sit in the classroom.

## Animation Tools

There are a number of programs designed explicitly for creating computer animations. These are not usually part of a standard installation or site license and they are much more complicated to learn and use than a word processor. If, however, you have the support staff to help (or a technically savvy student willing to contribute his skills), some very helpful animations can be constructed. These can be used either in the classroom or as web-based study/review tools. This past year I have had the good fortune of having, for the first time, a technical support person whose job description is dedicated to assisting instructors in educational media, particularly Internet-based materials. He uses Macromedia Director (v.8.5)<sup>23</sup> to create interactive study tools for my students, often

---

<sup>22</sup> Camtasia is produced by the TechSmith Corporation <<http://www.techsmith.com/>>. We currently use Camtasia Recorder and Camtasia Producer (both v. 2.2.1).

<sup>23</sup> Macromedia produces a number of web-related tools: <http://www.macromedia.com/software/>, including a simpler animation tool, Flash. Another of their products, Authorware, can create several media instruments, including quizzes that can be run within a regular web site apart from utilizing course delivery software.

adapting my previously static illustrations in a captivating way. This program generates Shockwave files for use on the web (requires the Shockwave web browser plugin on the user's computer) or executable (exe) files for stand-alone use.<sup>24</sup>

Another animation technique that my OT colleague, Dr. Alan Ingalls, has pioneered in his Internet Hebrew course with the help of our support staff is the use of animated gif files.<sup>25</sup> He has used this technique to develop a set of graphics which teach beginning Hebrew students to write the characters of the Hebrew alphabet. A similar application to the Greek alphabet would be helpful, and I plan to add these to my course next year. These graphic files are included on appropriate web pages for use by either resident students or by Internet students. They are particularly helpful to the Internet student who otherwise never gets to see the instructor write the alphabet on the board. Creating an animated gif is much simpler than the complex process of creating an animation in Macromedia Director—though obviously the options are much more limited. There are a number of programs that can be used to create these animated gif files; we have used Microsoft GIF Animator, which comes with FrontPage.<sup>26</sup>

### Web Browsers/Pages

With the explosion of the World Wide Web over the past decade there are a great many options available for teaching almost anything and Greek is no exception. This is not the place to discuss the ins and outs of the web, but only to suggest how it may be adapted to the needs of the Greek classroom. The web can be used to considerable advantage as a supplement to the regular classroom. A course web site can contain the usual syllabi, reading assignments, etc., but it can also be adapted to the specialized needs of teaching Greek. Many of the other items discussed in this paper can be organized easily in a web site since the framework is a very inclusive one that allows a myriad of other file formats and programs to run within the confines of a web browser. Some of the better candidates for use in this fashion are PowerPoint presentations, static or animated gifs, Acrobat documents, Shockwave animations, straight html content, and audio/video files. The one format that is not as flexible here is the word processing document due to formatting and font issues (see below), though this can be accommodated if a standard platform, program and fonts can be defined and required of students so that the teacher can be confident that the student is seeing exactly what the teacher intends.<sup>27</sup>

The same approach can be incorporated in a course that is taught totally over the Internet. Here a more comprehensive approach is necessary since *everything* necessary for the course must be included. This can be done in a normal web site, though specialized course delivery software can expedite this to a considerable extent. The first

---

<sup>24</sup> Macromedia's Flash is a simpler animation tool with similar, though less extensive, capabilities.

<sup>25</sup> Animated gifs are the computer equivalent of the old fashioned "flip books" in which a series of small pictures are rapidly displayed to create the appearance of motion.

<sup>26</sup> It can be accessed from as a menu item in Image Composer, the image editor included with FrontPage, or as a stand-alone program.

<sup>27</sup> Converting word processing documents to Acrobat pdf files is usually the best option for posting this sort of material on the web. Otherwise using an interchange format such as rtf allows most word processors to view a document with most of its formatting intact.

year that I taught elemental Greek as an Internet course I used a standard web site constructed with FrontPage. I have since been able to move to the WebCT course delivery system (see below).

There are many different programs available for designing web pages. I have used several on both Mac and Windows, but my preference is the Windows version of Microsoft FrontPage.<sup>28</sup> There are probably other programs that are better for graphic intensive pages or other specialized uses, but FrontPage handles everything that I need for page creation very capably and also manages the entire course web site. (Since our campus runs their entire web structure with Microsoft's Internet Information Server (IIS) with FrontPage extensions, the integration at this level is ideal.) Incorporating Greek font material is as easy in FrontPage as it is in a word processor since it uses a standard font menu.

In web page editors that do not incorporate a font menu for designating a font face (probably a rarity these days, but it is necessary still in some programs), Greek text can still be used by including the following coding: `<font face = "Galilee"> type Greek text here </font>`.<sup>29</sup> Depending on the Greek font used, it may also be helpful to format the Greek text as one size larger and as bold.<sup>30</sup> Where this is manually necessary, it can be done with this string: `<font face = "Galilee"><big><b> type Greek text here </b></big></font>`.

When designing web pages, especially those that incorporate Greek fonts, it is always important to double check the resulting page in multiple versions of different browsers on different platforms since each will handle such things slightly differently. As of early fall 2001, the key browsers to check include, on Windows: Netscape v. 4.7 and 6.1, Internet Explorer v. 5.5 and 6.0; and on Macintosh: Netscape v. 4.7 and 6.1, Internet Explorer, v. 5.<sup>31</sup>

## Audio and Video

Adding audio and video to a course, either resident or Internet, requires greater expertise, but can be very valuable. The greatest need for audio-video material is in an Internet course since the only spoken Greek the student hears will be the audio files the teacher puts online. This is most crucial at the beginning of the course as the student learns to first pronounce the alphabet, then individual words, then entire sentences. I have recorded the beginning steps as very small wav files that consist of only an individual letter or word. Students can play them over as many times as necessary as they become proficient. I have also recorded the vocabulary lists for each chapter, as well as a number

---

<sup>28</sup> Unfortunately, the Mac version of FrontPage has never been updated past v. 1. The Windows version is far more capable.

<sup>29</sup> Any font name can be substituted for "Galilee" in this example.

<sup>30</sup> The Mounce font appears much better on a web page if handled this way.

<sup>31</sup> These version numbers are a moving target, and newer versions of these programs may be available. The ones listed here, however, are, at the time of writing, the most commonly used versions. Due to the technical differences, these are also the minimum versions that should be required for Internet course use or for web resources for regular courses. Older versions will not work well with the newer software discussed in this paper. WebCT, in particular, will not run properly on older browsers.

of the early reading assignments so that the student can hear the proper pronunciation (or at least the one that I use!).

Video recording is now feasible at a much lower cost and skill level than previously. Although they will not replace a professional video studio for some purposes, the inexpensive video cams that attach directly to a computer can record video that is of acceptable quality for most Internet course purposes. In our Internet courses we have used both video shot in our campus studio as well as video recorded directly to the instructor's computer, either in his study or in the classroom. The later option is not only less expensive, but much more efficient in terms of a teacher's time since it is not necessary to reserve time in the studio, arrange for the camera technicians, and walk across campus to the studio.<sup>32</sup>

Longer video clips are converted to RealMedia format and streamed from our RealMedia Streaming Server. Using a streaming format makes both audio and video usable even on a 56K modem connection. Shorter clips (letters and words) are simply posted in wav format. An alternate format that appears to have equal or better quality with better compression (thus smaller file size) is Microsoft's Windows Media Recorder/Player. A streaming server is also available for this format.

One of the things that I have attempted to do in an Internet class to compensate for the lack of my ability to provide "hands on" motivation for learning Greek is to record short videos consisting of the comments of a number of pastor friends regarding their use of Greek. I sprinkle these throughout the year, all streamed in RealMedia format.

Although it is not as crucial for a resident class, it can still be useful to incorporate recorded audio. I have, e.g., used an audio track that attempts to reproduce first century koine pronunciation.<sup>33</sup> Although I still use Erasmian pronunciation, it helps the student conceptualize Greek as a living language. I have also incorporated a number of the video clips in the regular class.

### Font Issues

One of the major problems with using modern technology to teach Greek is with fonts and various technical issues related to how computers handle fonts—much of which is arcane or incomprehensible to most of us in the classroom.<sup>34</sup> We are in a transitional time

---

<sup>32</sup> This is a more specialized topic than can be treated here and one with which I am not as familiar.

<sup>33</sup> Randall Buth has recorded the results of his work in this area on a CD. I have only a sample CD with a few tracks, but it is helpful for my limited purposes.

<sup>34</sup> Please note that this simplified summary is intended for the non-technical user, not for the font designer or programmer. Thus keyboard and "keys" is the way the non-technical user (perhaps most Greek professors) views this situation. There are technical distinctions that should be made between encodings, codepoints, codespaces, fonts, characters, glyphs, etc. that are not defined here in technical terms, though I have tried to reflect the basic notions in a reasonably accurate way. Also please note that in the first part of this section I have tried to describe the situation as non-technical users typically view it; in the second part I have attempted to provide some correctives and more technically accurate information to move us towards some real solutions. Even these, however, are not complete. More detail is necessary, but I have already extended this section beyond proportionate limits within the overall purpose of the paper. I want to express my appreciation to Peter Kirk, Patrick Rourke, and especially to Peter Constable of the Biblical-Languages List for their assistance and patience in helping me understand some of the technicalities that are included here. Misrepresentations that remain are, of course, my own.

technologically and standards continue to develop. The following section is more extensive than might at first seem appropriate in proportion to the rest of the paper, but this is a crucial area and deserves far more attention than has typically been given it by our particular academic community.

Since computers have traditionally used a standard typewriter-based keyboard for text entry, the non-technical user has assumed either that there have been only approximately 128 characters (i.e., “lower ASCII,” those that can be directly accessed from the keyboard, shifted and unshifted positions) available, or, for those who managed to figure out how to access the upper ASCII” characters (not intuitive for the non-technical user), 256 characters total.<sup>35</sup> While this is not really true, it is how many of us have viewed things.

Such a pragmatic and limited conception has worked adequately for standard English text entry since most users have, in the past, been concerned only with entering text and printing it out in an appropriate format, whether class notes, journal articles, or books. In this “standalone” format and for this limited purpose, it has worked. Even when we have tried to adapt this system to include Greek, we have managed to cobble together an approach which works (sometimes well, sometimes only marginally)—but again, primarily for standalone use in printing material. The complications have arisen when we have tried to share data between computers, in which case the cobbled solutions have proven inadequate. The rise of the web as a medium of information exchange and as a pedagogical environment has also highlighted the shortcomings of our system for “doing Greek” in biblical studies.

To date, most foreign language fonts in common use in English-speaking academic settings (in our case, classical and koine Greek) have functioned by using English character positions (technically, the Western Latin character set) but foreign language (i.e., Greek) character shapes (technically, glyphs) instead of English. Thus the obvious equivalents (such as alpha = a, beta = b) have been used along with some less obvious ones based on some similarity of appearance (e.g., omega = w; theta = q, etc.). Diacritical marks have been a constant problem in Greek. Some Greek fonts commonly used in our circles implement this by creating overstrike accents and breathing marks (i.e., the diacritical character appears over the preceding character). This overstrike (“combining characters”) approach works well in some software, but displays very poorly in others. A few other such fonts have tried using precomposed characters (i.e., providing a series of, say, alphas with different diacritics over each one). This severely presses the limits of easily accessible character positions and also results in the need to remember all the possible character combinations and their respective (physical) keyboard equivalent.

Either approach leaves open the problem of entering such characters since they cannot be accessed directly from the keyboard. Two possible solutions are available. The

---

<sup>35</sup> It is possible to access upper ASCII characters from both Macintosh (through use of the option key) and Windows applications (and, I am told, even from DOS), but a standard installation of Windows does not make this at all easy or intuitive. In MS Word running under Windows 95/98/ME, e.g., one accesses the upper ASCII characters by pressing LeftAlt-zero (on the keypad) and then typing (with the Alt key still down), the 3-digit numerical ASCII character code (Num Lock must be on). This may work in other programs also. In any event, this requires a four-keystroke, arcane, non-mnemonic sequence to enter a single character.

first, is a smart input system. This might be through the use of a keyboard utility such as Tavultesoft's Keyman <<http://www.tavultesoft.com/>>. I have not used this Windows-only utility, so I cannot comment on its usefulness.<sup>36</sup> The second possible solution is to use some form of complex-script rendering technology which converts sequences of keystrokes into the appropriate, precomposed character or into the properly positioned sequence of each component element (character and diacritic/s).<sup>37</sup>

OpenType,<sup>38</sup> for example, if I understand its potential correctly, will allow a font designer to specify tables of characters that automatically convert a sequence of keystrokes into a precomposed character. For example, an alpha followed by an acute accent would be automatically mapped by the font to a character position (often above the standard ASCII range) that is a precomposed character consisting of an alpha with an integral, properly-positioned acute accent.<sup>39</sup> This allows the user easy access to such characters from a standard keyboard (input) and also allows the font designer to correctly place the accent in relation to the underlying glyph.<sup>40</sup> If this materializes in a useful form and tools<sup>41</sup> become available for the creation of good quality Greek OpenType fonts by

---

<sup>36</sup> Those who have are quite pleased with the results, but given the context of this paper, it should be pointed out that it is unrealistic to expect students to master the use of such a utility for purposes of learning Greek. It may serve a helpful purpose in terms of the instructor preparing materials that the student only views (e.g., on screen). It is important to remember that although many more current students use computers and would, perhaps, call themselves "computer literate," that seldom describes the proficiency necessary to install and use such utilities at Keyman. I have enough difficulties getting some of my students to install the Greek font necessary to read the materials and take the online quizzes!

<sup>37</sup> The current list of candidates include OpenType (Microsoft and Adobe), Apple's AAT (Apple Advanced Typography), and Graphite (SIL International). See further information in Peter Constable's information below. Either conversion option described above would work for koine Greek; there are probably advantages to one or the other in other languages such as Arabic which has a large number of characters that have multiple forms depending on their position in a word. Using precomposed characters requires a larger number of glyphs in the font. Specifying the proper positioning of each of the separate, decomposed characters requires fewer glyphs but involves a greater complexity in specifying the relative positions of each possible combination.

<sup>38</sup> The OpenType font format is an extension of the TrueType font format with added support for PostScript font data. These fonts are also referred to as TrueType Open v.2.0 fonts. For more information, see Adobe's and Microsoft's typography pages at: <<http://partners.adobe.com/asn/developer/opentype/>> and <<http://www.microsoft.com/typography/default.asp>>. (OpenType was developed jointly by Adobe and Microsoft). More technical OpenType information: <<http://www.microsoft.com/typography/creators.htm>>.

<sup>39</sup> As Peter Constable explained it to me, OpenType "transforms character sequences into glyph sequences. It might take a character sequence < alpha, acute > and output a single glyph; in another font (implemented in a different way), it might take that same sequence and output a sequence of glyphs < glyf\_alpha, glyf\_acute > with a set of positioning coordinates" (biblical-languages list post, 11/08/2001).

<sup>40</sup> This is not easier for the font designer—it is a more complex design process with many more variables to define, but it is surely easier for the end user. Other possible features of such smart font technology include such things as automatic final sigmas whenever a sigma is followed by a space or punctuation mark—though this could be counterproductive in some instances for designing first year teaching materials where this is not a desirable behavior in some instances. Of course, the same font technology could also define a non-transforming sigma for such instances.

<sup>41</sup> Tools for working with OpenType are not yet easily available; the next major release of FontLab (v. 4) which is due "imminently" will provide the first such "friendly" tool for this technology of which I am

Greek scholars for free or inexpensive use by the scholarly community (rather than professional type designers and commercial developers at commercial prices), this could be a major boon for those of us who work with Greek text, particularly in a web environment.<sup>42</sup>

Assuming this scenario for the moment, some of us have concluded that such existing fonts pose a variety of problems, particularly in a Greek class environment. Many of these problems we see as cosmetic in the sense that they relate to readability on screen. For example, the current Mounce font (αβγδεζηθικλμνξοπρστυφξψω) is serviceable, but is not well polished at this point (something that Bill Mounce is working hard to rectify) and is also a serif design best suited for printed output (though it can be used in other contexts). A similar verdict is given to SPionic (αβγδεζηθικλμνξοπρστυφχψω), plus its standard character assignments using a Windows-standard installation varies more radically from what has become a fairly standard character assignment (at least in our academic circles) for basic characters other than alphabetic ones.<sup>43</sup> The character shapes are also rather awkward and uneven. SIL Galatia (αβγδεζηθικλμνξοπρστυφχψω) is quite nice (if a bit “rotund”) and has many precomposed characters (ά έ ι̂ ό ύ η̂ ω̂), but it has major differences in even the alphabetic character assignments (e.g., ξ and χ are reversed; η is on the j key, etc.). The most typographically polished fonts for koine Greek are the expensive commercial fonts produced by Linguist’s Software (Graeca, et al), but there are major problems with these fonts when moving documents using these fonts from one version to another of major software programs (most notably Microsoft Word), rendering a long term investment in time and data questionable.<sup>44</sup> Not only are these fonts

---

aware. I have been working with FontLab v. 3 during the past year and have found it to be relatively easy to learn and use. FontLab appears to be the premier font creation tool in the “under \$1,000.00 range,” surpassing the former dominance of Fontographer (which has not been updated in many years). There are, of course, much more sophisticated tools used by commercial type houses, but these programs, if commercially available, cost thousands of dollars.

<sup>42</sup> I have developed a basic Greek font (“Galilee”) that I plan to move to the OpenType format as soon as that is feasible. It will be freely available for use by the scholarly community. Information on the current beta version in TrueType format may be found at <<http://faculty.bbc.edu/rdecker/galilee.htm>>. The beta version is available for download from the same page. One of the goals for this project is to design a font whose glyphs are optimized for screen display (either projected or on the web) rather than primarily for printed output.

I also note the Hampton Keathley of Galaxie Software (their specialty is publishing theological journals on CD) has also been working on developing Unicode Greek and Hebrew fonts for Windows with biblical studies in mind: <<http://www.galaxie.com/html/unicode.htm>>. The brief summary at this URL is worth reading. I have not seen these fonts (they require Windows 2000 or XP), so I cannot comment further.

<sup>43</sup> Such problems as character assignments can, of course, be corrected by use of a keyboard utility, etc., but that is not a helpful solution in the context of teaching Greek due to the lack of technical sophistication of the average student. The goal is to keep technology from being any more intrusive than necessary. If a student can’t use is “straight out of the box,” then he or she isn’t likely to be highly motivated to learn how to modify it. There are always the exceptions—students who love to tinker with such things, but we can’t design courses, particularly online courses where the student isn’t physically present to help sort out such complexities, with that sort of student as our standard.

<sup>44</sup> I have had to rework countless documents when upgrading to newer versions of Word since previously entered Greek text becomes a series of square boxes that cannot be salvaged by any easy method

too expensive to require of students, but licensing has also become unduly restrictive, making them unusable for Internet course development using Adobe Acrobat (the license now forbids even subset embedding in a pdf document!). For these and other reasons I have ceased using Linguist Software fonts.

At this point I have settled on using Mounce as the standard student font since it comes with their textbook and is serviceable. I have been gradually moving all my course materials for classroom use to my own Galilee font (αβγδεζηθικλμνξοπρστυφξψω) since these materials are for video projection display purposes in the classroom or for reading on screen by Internet students. I have deliberately designed Galilee as a sans serif, thick-stem-width font to make it more legible in those settings. I am watching developments in Unicode and OpenType and will incorporate them when it is practical to do so.<sup>45</sup>

What I have just described in the preceding paragraphs is probably the way that most of my fellow Greek teachers might evaluate the situation. It contains some truth and reflects several valid concerns. But it does not represent an accurate assessment of the situation. We non-technical users do not, at least in most cases, understand the underlying font technology and often assume that surface problems (e.g., typographical design and input complications) are the real problems. These two examples are problems, but relatively minor ones that have relatively straightforward solutions.<sup>46</sup>

The real issues lie elsewhere and the solution to them requires a more technical discussion. The currently emerging standard is Unicode which provides a very large number of character assignments, including the glyphs from multiple languages, allowing a single encoding to represent and distinguish hundreds of different scripts in the same document.<sup>47</sup> In preparing this paper I have explored these font issues and the Unicode-related solutions. Several members of the Biblical-Languages list have been very helpful in my understanding these issues. The following explanation of Unicode has been

that I have been able to discover. The workarounds that have been suggested by the developers have been clumsy and time consuming at best and sometimes results in loss of the formatting of the document—which takes nearly as long to redo as simply reentering the Greek text in a different font.

<sup>45</sup> At this point my font (see n. 42) still uses the traditional tactic of using Greek glyphs for Latin character positions; it is not encoded for Unicode. Assuming that becomes feasible in the near future, I would still face the challenge of converting substantial quantities of my existing teaching materials material (in excess of 100 meg) to use a Unicode font for Greek. This is perhaps the greatest challenge.

<sup>46</sup> The typographical design issues are solved with a font editor such a Fontographer or FontLab (though remember that only the original font designer/owner can legally distribute a modified font) and input issues can be addressed with a keyboard utility such as Keyman (see a description of this utility later in the paper). Both of these solution, while “straightforward,” are neither easy or intuitive for the typical user (professor or student). Some are also expensive (i.e., the cost of font design software).

<sup>47</sup> More technically, Unicode allows large codeset fonts to use the same encoding standard for multilingual data. There have been systems devised previously to enable more than 256 character positions in a font both on Mac and Windows (e.g., Far East Windows used double-byte encodings). Initial Unicode standards allotted two bytes per character rather than one byte, allowing  $256^2$  possible characters: 65,536. The current Unicode specification apparently allows more than a million possible characters using multi-byte encodings. For more information on the technical issues surrounding Unicode Greek fonts, see “Unicode Polytonic Greek for the Web,” by Patrick Rourke at < <http://www.methymna.com/unicode/>>. Please note that this document will be receiving a major update shortly (the draft posted there as of this writing is v. 0.93), so check back later if the revision is not yet posted. (Rourke’s specialty is classical Greek.)

provided by Peter Constable of SIL International, one of the relatively few people who are conversant with the biblical languages and with Unicode and is included here with his permission. It is very important that anyone planning to adapt technology for use in Greek pedagogy to come to grips with this information, especially if the intended delivery medium includes the web.

---

### *Excursus by Peter Constable, SIL*

Unicode provides a key building block that is solving these font problems and making it possible to have software that knows how to handle both English (and other Latin text), and Greek text. So, what does it take to take advantage of these benefits offered by Unicode? There are four things that all need to fall into place: data, keyboards, fonts and applications. We'll look briefly at each one in turn.

First, we need to create data that follows Unicode's definitions for how characters match up with the numerical codes used to store them. Because Unicode is far more comprehensive than the old 8-bit character sets we have been used to working with, it's also a bit more complicated. One problem related to switching to Unicode is that we have lots of data that isn't in Unicode. That's one of the challenges that we'll have to overcome, and there are some tools for this that are starting to appear.

The next thing we need is keyboards.<sup>48</sup> There is good news here at least for Windows users: Windows XP comes with keyboards that can be used for entering polytonic Greek data encoded in Unicode. Not everyone will necessarily like the default keyboard layout, but programs like Tavultesoft Keyman 5 <<http://www.tavultesoft.com>> can be used to design customized keyboard layouts.

The third thing we need is fonts. The fonts that we have been using will not work with Unicode because they have the Greek character shapes on the character codes for Latin characters. We will need fonts that have been designed to conform to Unicode encoding.

There are more font-related issues, but before we can talk about them, we need to explain a complication with Unicode that has to do with pre-existing standards. Because Unicode needed to be backward compatible with earlier standards, it supports two ways of dealing with Greek diacritics: either using separate "combining" characters for the diacritics, or using characters for precomposed combinations. So, for example, an alpha-acute can be represented in Unicode as an alpha (character code U+03B1) followed by a combining acute (U+0301), or it can be represented by a precomposed character alpha-acute (U+03AC). Of course, this has implications for our data and keyboards. Keyboards need to be designed to work one way or the other. As for the data, we need ways to deal with the possibility that the same thing can get represented in more than one way. This all sounds rather complex, and it is, but fortunately Unicode not only inherited this problem but has devised ways for software developers to solve them. Again, the technical details go beyond the scope of our discussion here.

This matter of having either non-composed or precomposed ways of representing Greek base + diacritic combinations has implications for the fonts that we need. If data is encoded using the precomposed way of doing things, then the fonts just need to have shapes ("glyphs" in technical terms) for each of those precomposed combinations. That is not all that difficult. If data is using the non-composed way of doing things, though, then we are faced with the problem of overstriking diacritics: some combinations will not look good, and in some applications none of the combinations will look good.

---

<sup>48</sup> The reference here is to logical keyboards, not the physical keyboards on which we type.

Unicode normally assumes that scripts that have diacritics are encoded using the non-composed approach. It is intended to be used with software that automatically handles issues of making the diacritic combinations look good. Some software builds this support directly into the application, but most new software programs use standard font technologies known as “complex-rendering” or “smart-font” technologies. There are three of these technologies being introduced by different vendors.

The most commonly used of these is OpenType, developed jointly by Microsoft and Adobe. OpenType requires special software support. In Microsoft products, this is provided in a component known as Uniscribe. One of the aspects of Uniscribe is that support for particular scripts needs to be specifically written into the software. As a result, OpenType support with Uniscribe has been included with Microsoft applications starting with Internet Explorer 5.0 and Office 2000, but this did not include support for polytonic Greek. The Uniscribe software is being updated by Microsoft, however, and newer products will be able to support polytonic Greek quite well.

Apple computer has a different font technology known as Apple Advanced Typography (AAT), which has been part of the Mac OS since OS 8.6. Unlike OpenType, AAT places all of the script-related support inside the font. Thus, if you’ve got the font, you don’t have to wonder whether the particular version of the Mac OS will support that script or not.

The third font technology is known as Graphite, and has been developed by SIL International. The initial version has been written to run on Windows (it is now open source, and there has been talk of a port to Linux). The main difference between Graphite and OpenType is that Graphite follows AAT in putting all of the script-specific support into the font. (This was done since SIL needs to be able to work with minority scripts on Windows without being dependent on Microsoft having added support for that script to the Uniscribe component.)

So, we need fonts that conform to Unicode, and for the non-composed way of storing data, we need fonts that use one of the “smart” font technologies to deal with getting all the diacritics in just the right place. This takes a greater amount of development effort than the earlier fonts did, but once it is all working our lives will be a lot easier.

The last thing we need is applications. We need applications that support Unicode-encoded data, but we also need applications that can work with the new “smart” font technologies. For AAT and Graphite, the fonts will not work at all unless applications have been specifically written to support those technologies. This has a significant consequence in that, so far, there are very few applications that support these technologies, and none of our existing applications can benefit from them. This will be of particular concern to Mac users, since AAT is the main solution on that platform but hardly any applications exist so far that support it.

For Windows users, the situation is much better. First, Uniscribe and OpenType support have been incorporated into Windows 2000 and Windows XP so that most existing applications will immediately gain at least some benefit from whatever level of script support is provided in Uniscribe and the OpenType fonts the user has on their system. Moreover, all of Microsoft’s major applications—Internet Explorer, the entire Office Suite, Publisher and FrontPage—all provide full support for Uniscribe. For people wanting to work with polytonic Greek in those applications, they only need to wait for the right version of Uniscribe that supports Greek, and if they are working with data that uses the precomposed way of doing things, they can start working with these applications right away.

An additional note on working with Unicode-encoded Greek on web pages: the reader must have a browser that supports Unicode. Recent versions of browsers (at least IE and Netscape) do support Unicode. The reader also must have appropriate fonts, and appropriate “smart” font rendering support (if combining diacritic characters are used). Unicode-conformant Greek fonts are starting to appear, and Microsoft may well bundle some adequate fonts as they update IE and Windows (they are tending

to do this as their developers add support for more and more scripts). Also, it's important to note that the World Wide Web Consortium recommends that text data on the Web use precomposed characters where they exist in Unicode (technically, Unicode normalisation form C). What this means is that "smart" font rendering actually isn't needed for polytonic Greek. (That's good news for many people using browsers other than IE on Windows.)

Of course, you also need a way to create those web pages, and know how to deal with the character encoding issues in the HTML editor. There are some important technical details (HTML numeric character references, charmap settings, Unicode character encoding forms and UTF-8) that would be too involved to cover in a brief overview. What is worth noting is that some ways of creating the HTML pages will work better than others.

For example, let's assume that you have a keyboard that can generate Unicode Greek characters. You could use this to enter text into Word 97 and later, and from there export HTML pages that will work. On the other hand, if you try keying text directly into FrontPage 2000, you'll probably find that it won't accept them.<sup>49</sup> If you can create the text, FrontPage 2000 will view them without a problem; it just isn't set up to receive polytonic Greek from the keyboard. This is fixed, however, in FrontPage 2002.

There are additional complications you'll run into with keyboarding: Windows 95/98/Me have significant limitations in this regard. There are ways to enter Unicode polytonic Greek on those platforms, but few apps support them. The same situation occurs on the Mac. If you create a keyboard layout using Keyman 5, you can key Unicode polytonic Greek into Word 97 and later on Win9x/Me. There are not many other options for those platforms. I don't know of any options at this time for creating Unicode polytonic Greek in HTML on the Mac. There may be something out there, but it would probably have to be something that was developed for the NextStep platform and ported to the Mac. The best platform for working with polytonic Greek at this point is probably Windows XP or Windows 2000.

So, there are at least some ways to create the pages. Many users will have adequate browsers, but they may not yet have the fonts. Indeed, there isn't yet much of a selection of Unicode-conformant fonts for polytonic Greek. It's not all in place yet, but we're getting there.<sup>50</sup>

---

## Specialized Course Delivery Software

With the growing popularity of Internet courses, there have been significant developments in specialized course delivery software. A number of such software packages are available including BlackBoard, WebCT, eCollege, etc.<sup>51</sup> I have experience only with WebCT, so my comments here will be focused on that product. Please bear in

---

<sup>49</sup> There is a way you could get it to accept Greek characters *other than* the diacritics, i.e. what would be needed for Modern Greek, but since that's not what this audience wants, there's not much point describing it.

<sup>50</sup> Peter Constable, Non-Roman Script Initiative, SIL International, Dallas, TX, personal correspondence, 11/8/01. The SIL website is at <<http://www.sil.org>>. I have added one note and made a few minor revisions in wording for stylistic purposes.

<sup>51</sup> A comparative feature review of about twenty such software packages may be found at the Marshall University site: <<http://www.marshall.edu/it/cit/webct/compare/comparison.html>>. Individual URLs for the three mentioned are: <<http://www.blackboard.com/>>, <<http://www.webct.com/>>, and <<http://www.ecollege.com/>>.

mind that I have no basis of comparison with the other products available. Our staff that was responsible for the choice of WebCT did so on the basis of both features and affordability. Some of these products are very expensive and can only be justified by large institutions. Though WebCT is not inexpensive, our staff concluded that it had the best balance of features versus price for our budget range.<sup>52</sup>

Although much of what course delivery software does can be implemented using regular web pages, there are five major advantages to using course delivery software.

- All courses delivered with such software will have a consistent look and user interface. This makes it much easier for students to learn how to take a course.
- Unless there is professional development help available, such a course will also have a more professional appearance. A sloppy, amateurish appearance will diminish the credibility of the course and its content.
- Automated quiz/test capabilities are built into many course delivery packages. This is very difficult to implement in a regular web page.
- Threaded discussion forums are an integral part of many such programs.
- The publisher provides technical support.

If a teacher has adequate skilled support available through his institution, particularly if there are html and Java programmers and network engineers on staff, then such advantages may be less significant, especially if there are only a few courses being taught or supplemented with online resources. But if Internet capabilities are being utilized very extensively, then it is far more resource efficient to work within the framework of existing course delivery software solutions, even if these are supplemented with other web-based materials.

Speaking only of WebCT, I would caution you that, in contrast to the student's interaction with the program, the user interface for the course designer<sup>53</sup> is not particularly user friendly. At times it is downright clumsy. The learning curve is a steep one. The most recent version of WebCT (v. 3.6, released late summer 2001) contains some improvements in this area (esp. in comparison with the v. 2.x program that we used in previous years), though you will discover that it is not as well designed as the typical word processor or presentation program. Part of the reason for that is due to the software architecture: the software runs totally on a server and both designers and students access all features of the program through a web browser. It is thus limited to the kind of controls and interface that can be implemented in a browser. All processing is done on the server; all the user sees is a screen display. Each change or command must be sent to the server and a new screen drawn for making even small changes. This has the disadvantage of slower response, although that is somewhat offset by the versatility of the design: the software can be accessed from any computer with an Internet connection and a web browser. The teacher can work on the course from home or out of town just as easily as on campus, and that without installing any additional software on the other computer.

---

<sup>52</sup> I was not part of the evaluation or selection process, so I cannot comment further on these factors.

<sup>53</sup> By course designer I refer to the teacher—unless there is support staff to handle all the mechanics of an Internet course.

I cannot possibly introduce you to all aspects of the software in this presentation, so I will try to highlight some of the key features and show you how I have adapted it to teaching Greek. The designers never envisioned using the software for teaching a foreign language that uses a script different from English. Consequently, it takes some creative tactics to coax Greek out of a web browser running WebCT, but it can be done effectively.

First, an overview of the WebCT program.<sup>54</sup> MyWebCT. Home Page. Course Materials. Ch 16 materials. Quiz system: student view (Quiz list; Individual quiz; Parsing example). Quiz system: designer view (Index page; Edit quizzes; Quiz editor; Quiz settings; Question editor). Bulletin Board: Forum list; Threaded display.

There are no font menus anywhere in WebCT. The user's web browser default font is used for all text. That means that, as in any web page, the designer has no way of knowing exactly what any given student's screen will look like. The content will be unchanged, but it may be formatted and spaced differently on each student's computer. As a result, spacing cannot be assumed; formatting must use tables or graphics if spacing is crucial. Fonts can be specified manually, however. In almost every WebCT screen for which the designer inserts text, html coding can be included with the content information and will be properly interpreted and displayed on the user's computer. I have made it a habit to always specify a standardized font for all my courses so that there is some consistency in appearance as students may access the course from different computers.<sup>55</sup> To do this requires manually typing the html code: `<font face = "font name"> type text here </font>`.

This same technique allows the use of Greek characters, even within those sections of text that have a particular English font specified. As an example, here is the text that is entered in a WebCT dialog to produce a text block on the introductory quiz page in my first year Greek course:

```
<font face = "Trebuchet MS">If the next phrase in brackets is <b>NOT</b>
in Greek characters, then you haven't installed the Greek font yet:
<big><big><b>[<font face = "Mounce">a b g d e z h q i k </font>]</big>
</big></b><p>If you need to install the font, go to the "File Download"
area (icon on the course home page), click the link to download the
correct font (= Windows TT font for most of you), then... </font>
```

This produces text that the student sees as:

If the next phrase in brackets is **NOT** in Greek characters, then you haven't installed the Greek font yet: [α β γ δ ε ζ η θ ι κ]  
If you need to install the font, go to the "File Download" area (icon on the course home

---

<sup>54</sup> If convention center facilities allow (as they are supposed to), a live connection to the Baptist Bible Seminary WebCT site will be demonstrated at this point. If that is not possible, or if connection speed proves to be unacceptably slow, then static screen shots of several key portions of WebCT will be shown instead.

<sup>55</sup> If you do this, I recommend using one of the standard cross-platform fonts that work the same on both Mac and Windows. The best choices are to be found among the "web core fonts" available from Microsoft: <http://www.microsoft.com/typography/>. They are not only cross-platform, but are optimized for use on screen.

page), click the link to download the correct font (= Windows TT font for most of you), then...

This technique is used most frequently in the quiz system in WebCT. The quiz generator is quite capable, though it will take some experimentation to figure out some of the options. I use this, not only for the Internet students, but also for the regular resident classes. Doing so saves me a significant amount of class time each week and also allows my students to take the quiz when they are ready to do so. They can study as much as necessary to master the material and then take the quiz anytime before the deadline. I am experimenting this year with allowing the students to take each quiz twice and receiving the average grade of the two attempts. WebCT administers this automatically, so it does not add an administrative load to my schedule and (hopefully!) encourages the students to master the material.<sup>56</sup>

A variety of question types are possible, most of which can be graded automatically by WebCT. In order to ask a question about Greek, most questions will require Greek text. All but one question type accepts the html coding necessary to include Greek characters. Multiple choice (also used for True/False), short answer, and paragraph answer questions are straight forward in this way.<sup>57</sup> The only type that is problematic is the matching type. WebCT will accept the html for the question “preview,” but will not display Greek in the drop-down menus the student must use to enter the answers. Instead the menu items show uninterpreted html—which is generally worthless and confusing to the student. Since they *can* see the Greek text in the preview panel just above these menus, I have adopted the strategy of placing random alphanumeric characters at the beginning of each answer. The student can then match the first letter in the menu with the preview answer and select the intended response.

Another problem area in the quiz system is that diacriticals are not displayed clearly. This is partly font related, part web browser related, and probably partly WebCT. The Mounce font that I use in the quizzes is a small x-height font. To compensate, I normally add a <big> tag to increase the size for Greek text. But if a question hinges on a diacritical mark (e.g., a quiz on accents and breathing marks the first week of the semester), even this is not adequate. I compensate in two ways. First, I use *two* size tags, <big><big> (which increases the font by two sizes), and second, I include a graphic that clearly distinguishes the diacriticals. (WebCT includes an option with each quiz question to display a designated graphic file in gif format.)<sup>58</sup>

Quiz answers must be manually reviewed even when WebCT provides an automated scoring (only paragraph questions are not scored automatically). This is an ideal task for

---

<sup>56</sup> I specify in the quiz settings that there must be a one hour waiting period between the two attempts to encourage study and to make it more difficult to remember the precise questions. There is no way to police cheating in such a scenario and it may not work well in some settings. In my seminary context, this has not proved to be a problem. The “proof” of their mastery comes on the exams, not the chapter quizzes. I therefore reduce the value of the total quiz points earned in the semester to 80% so that quizzes count less and exams more.

<sup>57</sup> There is also a calculated question type that is not relevant to Greek.

<sup>58</sup> I am hoping that when my Galilee font is ready to use in this context that it will help ameliorate this problem. I have not revised the quizzes to use that font yet, however.

a teaching assistant or grader if you have that luxury, but it does not take a long time to review the submitted quizzes. Each quiz is displayed for review and incorrect answers (per WebCT) are flagged in red. All that is necessary is to scroll rapidly down the quiz until a red answer appears. Then a quick check can determine if it is a simple English misspelling, legitimate alternate translation, etc. (WebCT grades “literally,” letter-for-letter, so one extra space, incorrect spelling, or variant punctuation mark will be scored as incorrect; “case insensitive” can be specified.) Multiple correct answers can be included and I always try to do this with vocabulary. Translation is tough to grade. I include a simple translation and sometimes an additional variation or two. This sometimes scores correctly, but most of the time you have to do the translation questions manually.

Parsing-related quiz questions stumped me for some time and initially I defaulted to making them short answer questions. If students used my “single letter parsing abbreviations”<sup>59</sup> in the correct sequence, this would grade correctly, but I could never determine easily if they had the correct lexical form. (It is asking too much to require students to learn to enter html coding to include Greek text in their answers!) I have finally devised a multiple choice method of expediting a parsing question. Instead of the usual four or five choices, I list about twenty. These answers, for which multiple responses are allowed, list the major parsing categories in sequence. For example,

Parse the following form; mark all that apply: ἐβόλομεν.

- |                  |                             |
|------------------|-----------------------------|
| 1. First person  | 10. Active voice            |
| 2. Second person | 11. Middle voice            |
| 3. Third person  | 12. Middle or passive voice |
| 4. Singular      | 13. Lexical form βολω       |
| 5. Plural        | 14. Lexical form βλαω       |
|                  | 15. Lexical form βολλω      |
| 6. Present       | 16. Lexical form ἐβολον     |
| 7. Future        | 17. Lexical form ἐβολω      |
| 8. Imperfect     | 18. Lexical form ἐβολλω     |
| 9. Aorist        |                             |

In this format, WebCT can correctly score the parsing. Be aware of one thing, however. If you only indicate the *correct* answers, WebCT will score a 100% if the student marks *all* the options! A savvy student will figure this out after a few quizzes. To avoid this problem, it is possible to define a *negative* score for *incorrect* answers. In this example, since there are five correct answers, each one can be scored as 20%, but the incorrect options can be marked in the WebCT question editor as worth -10% (or -20%, etc.). If students are told in advance how this parsing is scored, it becomes an effective way to prevent either guessing or cheating on such questions.

### Course Development Suggestions

I conclude with six suggestions regarding developing technologically-aware koine Greek courses. 1) Start small and build over several years. Do not try to incorporate everything possible, or even everything that I have demonstrated today, in one year—unless you have abundant resources, both financial and support staff. Learn as you go. Become comfortable with one or a few technologies per semester. To develop a complete

---

<sup>59</sup> I use the single letter abbreviations that originated in the DOS version of GRAMCORD.

Internet course takes hundreds of man hours. Even though I already had all my course materials on computer, most of them already in use with a video projector in the classroom, it took me an entire summer to prepare my first Internet course offering of first year Greek, and then I completed many of the pieces as the semester progressed.<sup>60</sup>

2) *Do* work toward developing web-based support materials for your resident classes even if you never attempt to teach Greek in a totally web-based class. Providing these resources to your students *is* worth the effort. It can save you a considerable amount of class time if you do nothing other than develop a quiz system that can be administered automatically out of class. Remember that everything you develop is a long term investment in your pedagogical tool box. You only have to do it once. (At least until you upgrade to different software or your old software no longer functions on new hardware, etc.—but that’s life in the world of technology!) Future years can add additional pieces or enhance earlier ones.

3) In materials designed for student access (web enhancements to a course or especially Internet courses), use mainstream, “least common denominator” technology whenever possible—technology that does not require the student to implement non-standard configurations or specialized technologies with complex installation procedures.<sup>61</sup> Also keep cross-platform issues in mind. Since you will frequently have students using at least Windows and Macintosh, do not use materials that can only be accessed from one or the other of those platforms. It is reasonable to expect students to install a particular font, to use an up-to-date browser,<sup>62</sup> to have a current version of Adobe Acrobat, and to be able to access word processing files through rtf interchange format. Beyond these basics, I would advise only experimenting with more complex technologies for optional course resources.<sup>63</sup> In the classroom or a campus computer lab setting, you can, of course, assume more robust options. Where you need to be careful is in assuming what a student has in the dorm or at home, especially if he or she is a remote, Internet student.

4) Be sure to provide explicit instructions for students, especially for technologies (such as online quizzes) that are a required part of a course. Provide the necessary information, step-by-step, in printed form, on the web, and in email. Tell them multiple

---

<sup>60</sup> This was using a FrontPage web with no support staff to assist. I am not on a 12 month contract, so I did not always invest a 40 hour week all summer as I had other things to do as well. But there were hundreds of hours expended that summer. Using WebCT might have been different, but if I was just learning WebCT, I doubt it would have shortened the time. When I did convert to the WebCT platform, I would estimate that it took me nearly as long as the original FrontPage creation process. Perhaps longer. If someone was already familiar with WebCT, development time would have been significantly shorter. Assistance from support staff (which I now have, but did not then) would also have been of considerable help. My moral is that you should not underestimate the time required to develop a credible course of this sort.

<sup>61</sup> The continued growth and development of the web is a positive step in the direction of open access and cross-platform standards, but it is not a perfect world in this regard and probably will never be in every area.

<sup>62</sup> See note 31.

<sup>63</sup> At this stage in the technology cycle, I would advise assuming that most students on Windows will be using Windows 98 or ME, many still on 95, with a much smaller group using Windows 2000 or XP. On the Mac platform, assume an average OS of 8.x, with somewhat fewer running 9.x, and only a very few using the new Unix-based OSX.

times what to do and how to do it. The first two weeks can be somewhat hectic and even initially disheartening as they learn a new system, but they'll get the hang of it and they will appreciate the flexibility and additional help that it provides. It *is* worth it.

5) Never experiment with a new piece of hardware or software in an actual class setting. Always test it without an audience, using the same computer, projector, etc. as will be used in class. One can lose far too much teaching time by debugging problems in front of a class. Even on a fast computer, rebooting wastes a lot of time, to say nothing of the disruption of the lesson when problems occur at an inopportune time. Always have a backup plan in mind. If you plan to incorporate network dependent resources or a web page from another site, remember that network outages are seldom anticipated. Internet congestion can hamper access to other sites—and all too often that other site just happens to be down when you need it. Such is life on the net!

6) Technology is not a magic wand for all your teaching woes. It won't transform your teaching or make your work easier. By itself, it can't motivate students. It won't raise scores overnight. But it can help. It provides enough promise to be explored by any Greek teacher who has access to the necessary resources so long as unrealistic expectations are avoided and whatever technological avenues that are attempted are compatible with an individual instructor's teaching style.