On “Teaching” “Digital Humanities”

Chris Forster

http://cforster.com/2013/08/teaching-dh

As this academic year warms up, some thoughts on the last one; last year I had the unexpected opportunity to teach a graduate seminar in the Spring.I was not as diligent as a blog coordinator, keeping up with summary posts, as I would have liked, but some summaries and links to student blogs are available here. I wavered between a theories of modernism course (think: Hugh Kenner, Peter Burger, Frederic Jameson, Susan Stanford Friedman) and an “Intro DH” class, settling on the latter simply because I thought it would be more valuable to graduate students (few of whom, in our department, have strong research interests in modernist studies).

The course benefited from a number of sources; one is Scott Weingart’s excellent list of DH syllabi.He says syllabi; I say syllabuses; it is a battle that has raged for centuries. I should also thank a number of people who were kind enough to offer thoughts (and sometimes texts) as I put together the syllabus: Stéfan Sinclair, David Golumbia, and Brian Lennon offered suggestions. They were all more than generous; though, of course, they bear no responsibility for whatsoever for the syllabus. I also was fortunate enough to end up corresponding with a number of other folks during the semester, thanks to all of them.

If you’re interested in seeing the syllabus, you can see it in [PDF], [HTML], or (heaven help you) on GitHub.In theory, the whole

github syllabuses thing sounds promising; and for DH stuff, who knows? But really, just putting the stuff on the web is probably the best way to share teaching materials.

The syllabus includes, at least to some extent, basically all the texts that, a while back, Brian Croxall mentioned as the "usual" DH reading list:

(**digiwonk?**) (**readywriting?**) Graphs, Maps, Trees would be first. Then Debates, two Blackwell books. Ramsay, Jockers, Kirschenbaum. The usual.

— Brian Croxall ((**briancroxall?**)) July 19, 2013

I would have *loved* to use Jockers's *Macroanalysis*, but it was not out in time; I would certainly use it were I to teach the class again. Indeed, I could very easily imagine a class taught around *Macroanalysis* as its central text.

I imagined the course as centered by a basically epistemological perspective: does the "becoming digital of textuality" "becoming digital of textuality" is clumsy; but I think it gets at the thing I'm interested in better than anything else. change the sorts of knowledge about literature and culture that scholars produce. Does it offer, to put it more polemically, a *science* of culture? (I avoided this polemical formulation in class; not least because of the definition of "science" that it assumes). This was the motivation for starting with C.P. Snow's "The Two Cultures," a text I would, in the future, however excerpt Snow; there are a lot more relics of the Cold War in that essay than I recalled, not all of which were directly relevant. which I sometimes see condescendingly referred to as if it were backward, outdated, or self-evidently wrong, when I find its core thesis remains provocative and at least partially compelling.

The course was then organized around questions of digitization and textual representationThe Latour and Lowe essay, from Switching Codes is a real gem, and one I don't see mentioned very frequently., and then a whole slew of things I called distant reading.

I'm not sure that there's anything in my life that I'd call an unmitigated success, and this class is surely no exception. I think though that it did the job of familiarizing folks with at least some of what "DH" is, particularly for folks in English departments.

I'm not sure if I'll ever teach such a course again. But here are some things I think I learned—things I'd change and things I'd do the same way again:

- **More, not less, technical**: I was acutely aware of the worry among students that this class would be "highly technical" and require students to have all sorts of prerequisite knowledge. To avoid that, I think I erred too far on the other side, and set the bar too low. I integrated some tools into class, while leaving the more technical, hands-on stuff for suppelementary (optional) hands-on sessions (we did a little Intro Python & nltk; we did some topic modeling with MALLET, some web-scraping, etc). This was logistically a problem (finding a time that worked for everyone). More fundamentally, my sense is that the class would have been stronger had it been *more* technical. The evaluations seem to confirm this almost unanimously; everyone thought more hands on with software would be a benefit.

  What would that mean in practice? While we spent time talking about encoding; and looking at some examples of TEI encoded documents (including the Folger's), we didn't actually encode anything. But until you've had to complete a teiHeader, you don't really know the burden of metadata. Actually making folks encode a text would be one avenue I might pursue. By selecting texts with a sufficiently interesting textual history and encoding some apparatus, this could be interesting assignment indeed.

  Matt Wilkens reports having success using "The Programming Historian"; I could imagine doing something similar, and centering the class's practical activities around Python

(about which more below); I would do so despite serious reservations about both my own qualification to be teaching such material (I am, after all, an autodidact in these matters) and the utility that a superficial command of a scripting language would give a graduate student in English.

- **One Tool Well**: This is a corollary to the previous point; more technical, but also more focused. As we moved through the semester, the variety of different software packages we looked at increased: basic word frequencies with command line tools, R (particularly for mapping), MALLET, Stanford's Named Entity Recognizer, Python (with a number of libraries—particularly the NLTK), ImageJ, and others. There is a value to examining a diversity of software tools; but its costs, upon reflection, now seem too high. For someone coming to such technologies for the very first time, I think focusing on one, very flexible technology to do all (or at least most) of the things we would be interested in doing might be the best approach. And Python could fit the bill; certain things may be less pleasant in Python, but overall the consistency of a single language and syntax would have been a virtue.

  (I will say that while I think I'd prefer Python over R, *if you were to use R*, running RStudio Server would be a great way to provide a consistent software base for students; as a piece of software, it was pretty stellar. Speaking of which…)

- **Running a Server? Totally Worth It**: One of problems in a class like this is infrastructure; you want folks to be able to play with some of these tools, but trying to get MALLET, or Python plus a handful of libraries, installed on students' machines can be very unpleasant. To create some consistency in the software available to folks and to avoid having to try to install software packages on 6 different OS versions and hardware platforms, I rented some server space with Linode (ap-

proximate cost: $25/month) and set up user accounts for everyone. Then I installed all the software packages we would be interested in using.

Such a setup utterly lacks any GUI (unless you count browser access to something like RStudio), which requires folks to be comfortable at the command line. This was no small thing. Use of the server was essentially optional, and some folks simply never got interested in it. In the spirit of "more, not less, technical" I'd probably require more engagement with the server in a second version. I'll write up a more detailed explanation of the server set up I used soon; but this worked, over all, very well, and with some tweaks could work even better.

- **I'd Probably Require Twitter**: I suggested folks have a twitter account, but didn't require it. The folks who were on twitter, though benefited from it (I think); it provided an additional shared context; when authors we were reading were on twitter or had blogs, I was sure to note it, and I sometimes saw the extra context folks had gleaned from these resources in their contributions to class.

When I planned the syllabus, I was concerned to try to integrate criticism of "DH" into this class, to make it both a class *about* "digital humanities" as well as a digital humanities class. I wanted to leaven ambient excitement or "buzz" with skepticism, to use the tools but to do so with care and reflection, to balance the hacking with yacking (to invoke a short-hand that has produced much hand-wringing). I will say, however, that during class meetings, I generally found myself working harder to overcome the skepticism, rather than to contain the excitement; wanting there to indeed be a little less yack and a little more…