**Code and Poetry lecture: Drafts, Versions, Versioning 11-10-14**

* Working on core factors: textuality, digitality, vectors
* Wednesday: doing versioning
  + Hopefully moving on to intertextuality, networking, servers, cryptography, security
* Writing document H: H1.doc (parts 1,2,3), H2.doc (just parts 1,3), H3, H4, HF (which you send to your publisher), HF1.doc (you keep revising while publisher considers manuscript), publisher gets back to you on HF draft, but you’ve moved on from that version, so you take HF and the publisher’s version and try to integrate them
  + Initially a linear process, but when you get another person involved the process becomes asynchronous
  + Published as book by publisher, then they reprint (which you make some edits to, so the versions differ slightly), translations are printed (which obviously differ)
  + Notion of what the text is changes drastically in different versions – there isn’t an ur-text, there’s multiplicity (individual instantiations of different versions/drafts)
* Problem of duplication (Goodman, Benjamin, Genet (not assigned)): about identity of duplicates – are different prints the same thing or different things?
  + Same: fall under the uber concept of Hamlet (or whatever)
  + Different: each text is a separate material thing
    - Benjamin – trace of the person in the material text itself (earmarking, underlining)
    - Genetic criticism – trying to create critical editions where you preserve each change the author made (preserve the genetic mutation of the text)
    - End goal: reconstruct the mental process from initial idea to final product, trace intellectual development (but are we interested in the text or mental states?)
  + Bible: desire for it to be the idea of the Bible in the mind of God = the Bible’s text itself (one to one correspondence, so you always know what it means because the material = the idea; when you have different versions you can question who decides which is the right one/which one gets printed)
    - Sending vectors along different trajectories is a political thing
      * Documentary hypothesis: we pretend that the Bible is the word of God, but we’re going to analyze the language of each sentence – sees that syntax, dialect is different at different parts of the document, puts them into different sources (five of them), not a cohesive text (stylistically or logically). It’s a collection of different documents (genetic idea)
      * Problems: what point in the process of writing are we trying to reconstruct?
  + Years later, the professor decides that you have to buy a particular version (which they consider the authoritative version), but others will read a different version (paperback version, e-book, photocopy, online version) – left with a bunch of different material versions. Metaphysical problem.
  + Publishers have financial concern: the fact that you can make copies is lost revenue. If you can control each copy you can monetize it (Amazon kindle: you cannot print the book, share it (it’s connected to who you are/what else you have on your kindle) – strong control over duplication). Financial problem.
  + Unsolved problem for literary scholars and startups
    - Censorship: you cannot have any further duplication of a text (it’s just in your head and computer – the only way it can go is into the trash) – this is the radical control of vectors. Political problem.
      * How should we legislate to what extent you can duplicate/pass on a document? Should they be able to peek into what’s being sent on a vector and ensure that it’s not illegally sending copyrighted material?
      * One of US’s top exports: networking technology. A big part of this is servers. One major type is packet-sniffing servers (opens up the packet, sees what’s inside, closes it): censorship, security, copyright protection. Trying to reconstruct and control what’s being sent on vectors.
* Problem of collaborative annotation
  + Rap Genius: people annotate the lyrics (collaborative). Semantic web: you have all these texts (which are accessible via Google), but on top of that you have people churning the data (tagging, sharing, annotating) - the smart social level on top of it. Who gets to make this annotation/social layer? Rap Genius stepping in to do that (big business).
    - Kindle: locked platform (the document lives in our box – canonical version of text, not concerned with versions). Once you have the idea of the canonical version you can attach annotations to it.
    - Problem of collaborative annotation isn’t so big if you just look at one canonical version (lines will be same across translations, editions)
    - Concordance: addresses for each line are universal, so you can compile everything that’s been written about a particular line (history of comments about that line). Can see the history of interpretation
    - Rap Genius is a concordance machine, but idea of concordance only works if you know what the canonical version of the text is (the ur-text), otherwise you can’t annotate the same thing (have to work to find out what coincides in each version).
    - In real world, you don’t have a canonical version. Rap Genius wants you to take the text, put it on the website so we can go through the text together. Appealing vision, but how do they decide which version to use on their site? Which version do they own the rights to? They don’t have the rights to the stuff they’re annotating. Also, what rights do they assert over the contributions made? They’ve just copyrighted the whole thing (Genius) – haven’t worked out the particulars, so Rap Genius just owns everything that’s happening on the site (platform lock-in). Lots of intellectual property issues.
    - Decentralized annotation: possible solution, but if you don’t put it under the same platform, how do you create an annotation layer over it?
      * Instead of forcing people to read in a particular way, you aim something (a phone) at a page and have the annotation layer there.
* Problem of collaboration (working and writing together)
  + Single document model
  + Lots of business writing is collaborative (technical specs, letters to clients, advertisements)
  + V1 doc (which you wrote), three colleagues look at it and edit it, colleague 2 gets copy and edits it (V2), c3 gets it and makes v3, c4 gets it and makes v4. Then you have the problem of trying to make sense of all the different versions, reconcile them.
    - Optimize the model by picking someone to organize (editor), send to one person, have them send it to next person, then to one more, eventually back to you
      * Not asynchronous, instead step by step (comes back to you as completed document).
      * Single document model works best if you avoid the asynchronous
  + Central (google docs) model
    - Only one document – no versions. We all look at the same doc, write on the same page.
  + Third solution: introduced on Wednesday. Another problem is that you did all the work, one person didn’t, but do you both get the credit? Who gets ownership? This third solution tries to resolve the problem of clear ownership/visibility of labor.
  + NOTE: will have to install something (keep eye out for that)