chris & sean’s brainstorming spectacular

1) how should we let code get checked in?

-we need QA of some sort

-I submit a code change, it goes to QA

-First, code is run through automated test script that verifies inputs/outputs

-Developers responsible for updating automatic tests

-QA runs a black-box test against the “new” components that are affected (devs must inform them of all affected components)

-There is an internal, updated list of all components touched if one in the chain gets changed

2) what data do services need to be aware of?

-general health report

-incoming and outgoing requests

-maximum capacity for requests in current state

-project to warn company officials when projections look risky (e.g. if we can’t deal with the network traffic)

-also keep track of costs per request, and report when costs get too high (we set this value)

-In general we must track:

-Every instance of a service running with a version number

3) how lenient are we about errors/warnings at compile time

-in general, errors and warnings are not acceptable and should be dealt with immediately

-if an error is a compiler bug/issue, and percieved by upper level developers/designers/architects to not be potentially damaging, this rule can be overridden on a case by case basis

4) How do we deal with incompatibilities between third-party libraries?

-If the third party library doesn’t affect other third party libraries, just swap it in as per usual release procedure

-If the third party library does interact with others we have, test in a simulation outside of release, and release if all tests succeed.

-when pushing code out to release, keep a copy of the old system to let users still using the old system finish their current work

5) How do we deal with moving to new versions of our software?

-keep copy of any system that a user is still doing work on, switch them over to the new system when their currently-running process is done

-provide converters for data in case their data is no longer compatible – these will make their data compatible with the new system

-after 6 months, force the “upgrade” (unless a user talks to us, maybe for instance they’re doing a really big batch job)

6) Are there issues where we have to upgrade services in multiples? How do we do that?

Use the general approach above, but test more – this only possibly increases the old system instance copies we have to keep

7) Password Management issues?