1st June

Today is the first day of working on my dissertation project. I’ve been looking forward to this for a while, largely because I’ve been wanting to do something more practical than the endless reading I’ve been doing while revising for my exams.

My job is to build an emulator that emulates the properties of BASE transactions over an ACID database. Just before the end of last term I submitted an annotated bibliography on the project. My first plan of action for at least today and tomorrow is to begin reading through the resources that I’ve gathered, reminding myself of the different approaches that I considered, and choosing where to go next. While working on the bibliography I decided that the first BASE property that I should try to emulate would be that of eventual consistency. I still plan to stay with this original objective. I also picked out two potential real cloud databases on which to base this model, and I now need to look at the eventual consistency algorithms for both of these and decide which one I will actually try to implement. My supervisor suggested the possibility of maybe using ideas from both of the approaches, and I would certainly like to do this if it were possible, because then it might make the second half of the project, where I have to compare my solution against real cloud systems, more interesting.

I’d also like to take the opportunity to point out how relevant this project is to the rest of the modules I studied as part of my degree. The language I plan to use for my implementation is the Scala programming language, which is based upon Java and runs on the Java virtual machine, and so I will be putting the skills acquired in CO871 to good use. Because BASE transactions are heavily used in cloud-based databases, my project has obvious links to CO846. I also plan to make use of a large number of the tools introduced in CO894, as I believe that this will ensure a product of better quality.

Scala is a functional programming language, so I will be able to put the recursion skills learnt in CO884 to good use. This project lends itself well to concurrency, and therefore my skills learnt in CO890 will also be highly valuable.

2nd June

I spent most of yesterday going through the two main approaches that I found during my initial project research, which were the two eventual consistency models supported by CouchDB and DynamoDB. I have set myself a small target that by the end of today, I would like to have written up both a small description of the project and a technology review for the eventual consistency section of the project.

So far, both approaches seem to be very interesting and I can see how each one of them is effective in what it is doing. After reading through the two approaches more carefully, however, I think that both of them have various tradeoffs and I’d need to consider which one I’d want to implement very carefully.

With the approach offered by DynamoDB, all pieces of data are stored under version control. In implementation terms, this would be relatively straight forward to implement, as every time a piece of data is changed, one can just record the change and the time it happened in the database. While this would be straightforward to implement algorithmically, it wouldn’t scale well. This is because if we had to track every time something was changed in the database, this would take a lot of space relatively quickly.

The other approach is that of using vector clocks, which would be more complex to implement algorithmically, as it involves comparing these lists to determine which changes come before another. On the other hand, though, it would be easy to store a vector clock in the database for each object, as this is just a list of timestamps and other metadata, so maybe I can compress that into a more serialisable form. This would, however, make more of a challenge in implementing the actual algorithms.

4th June

Yesterday I had a meeting with my supervisor, we talked through the feedback of my annotated bibliography and I was able to get a clearer idea of what I needed to do and what then might be needed to get there.

I am now working on my project plan. I have decided that I will use an agile approach to my software development. This is partly because this kind of project is unfamiliar to me, and so by making it incremental, I will be able to assess my progress based upon my targets each week and check to see whether I am on target or not. I am currently in the middle of writing some broad goals. Once this has been done I will write more specific user stories and try to establish a plan from week to week. This will also be accompanied by the creation of UML use case diagrams to illustrate the requirements in a more visual form.

8th June

Today I had a really informative meeting with my project supervisor, which reassured me of two things. Firstly, I now know that I was going down the right track with regards to the project. Secondly, it also gave me some valuable pointers about where to go next. I also started implementing the project today by writing a very simple web service that takes in strings and runs them on the database. I’m really surprised how easy the play framework makes this. I think that although it took me a while at the tail end of last week to learn the technologies, this was really worth it, as it they are well suited to the project I’m doing and will make my job so much easier.

Now I need to think more carefully about the design for my project. Design work doesn’t normally come naturally to me. This is because of my visual impairment, I find it difficult to see the big picture, and sometimes I need to get this part of the work right, however, as a good design will mean that the work is far easier to implement and test, and a bad design will not only lead to a project of lesser quality but could also limit the scope of the project.

Another thing I need to investigate in the next few days are the testing features. Web applications are notoriously difficult to test. On the other hand, however, some of the things that I’ve had to do that were meant to be challenging I found to be straight forward, so I would not be surprised if in fact this task was also relatively straight forward.

9th June

Today I managed to get the vast majority of the simple web service completed that I was planning to implement yesterday. This currently means that I am a day ahead of schedule. I am very surprised, but also very pleased at how things are turning out. Now that I have started development work, I also plan to write a chapter for my dissertation about the development technologies used and why I have chosen them. Tomorrow I will begin the testing work that I mentioned in yesterdays entry.

10th June

Today I managed to complete the code for iteration One. This took longer than expected due to the complications with the Java JDBC library. I might use the axiom library that is provided with the Play framework for the main bulk of the development work. I’ve also begun to design the main structure of the application.

11th June 2015

Today I completed my first experience of unit tests for the play framework. The experience was fairly similar to other unit test frameworks I‘ve used so it didn’t take me long to figure out how to use it effectively. I really like the http features that it has though. This allows me to run mock versions of my application and check that the http responses are the ones that I actually want to have back. This greatly alleviates some of my worries for this project with regards to unit testing, as I was worried that it might not be that effective due to the web based nature of my project.

12th June 2015

Today I wrote a chapter for my dissertation. Not much more to say here. I also wrote a design document for my project. The aim of this was to convert the ideas that I’d been having on the rest of the project into a more formal document. I now think I have an overall design sorted for the project, but I still think I need to do some more work on parts of the algorithm, in order to make sure that everything is clearly planned, particularly the part about resolving any inconsistencies.

15th June 2015

Today the design document I produced on Friday got a substantial rewrite after I had some more thoughts on the problem over the weekend and decided that what I had wasn’t appropriate. The improved document is now much clearer and easier to understand. I also began coding this iteration today. I think that there’s quite a lot of work to do here, so I’m worried that I might not complete the whole iteration within a week, but I’ll assess that closer to the time.

17th June 2015

Today and yesterday I ‘ve written large chunks of the functionality for the second iteration. I am very thankful I chose to use the Akka framework to handle the concurrency that I’ve introduced into this project, as its made reasoning about that concurrency so much easier. There’s still a lot of code to write though. Currently, I’m not sure if I’ll hit the deadline that I imposed upon myself, although my study support tutor thinks that it wouldn’t be the end of the world as I’ll catch that up later in the term.

18th June 2015

Today, I finally overcame the difficulties that I was having with eventual consistency. There’s still a lot of work for me to do, but I’m confident that I’ll be able to get through it now with relatively little difficulty. I now need to start thinking about how the sample programs are going to work in order to prove that the application is working as effectively as it should be.

19th June 2015

I got through most of the work that I had set myself for this iteration. I need to look into testing Akka actors again, as the tools I have don’t seem to be very effective for this piece of work. Overall, I’m relatively pleased with how this week has gone.

21st June 2015

Today I’ve been testing the work I did last week, My supervisor seems pleased with my progress, and wants a demo of the system by the following Monday. I think I can deliver this, but I’ve had nightmares getting the testing equipment set up for testing Actors again. Over the weekend a new version came out, which meant I needed to reconfigure everything. I have now discovered TestActors though, which allow me to bring all the standard unit testing tools to bear on Actor based systems. I also think that after I submit the early deliverable, I could do with having a good look through the project and doing a code review. There are some pieces of code that I think could be designed better.

22nd June 2015

Today I’ve been fine tuning my design document as I prepare for submitting my early deliverable. I’m quite surprised that, for the most part, I implemented what I said I was going to implement. Maybe everything isn’t as fully tested as I would have liked at this stage, but everything seems to work and there aren’t that many bugs in the system. The main changes I’ve been doing are fixing spelling and grammatical things, and clarifying a few points. I’m also in the middle of putting a dissertation draft together for the early deliverable. I’m quite surprised that there was so much to write about with only a few weeks of working on the project.

24th June

Today I finally submitted my early submission, I now need to do the evaluation of my project that I mentioned a couple of entries ago and then I need to start thinking about basic availability. I have now started to really beef up the testing of my application. I know most things work but I want to be able to verify the components on an individual level and ensure that they work, as they should. This should also give me more evidence that my system does what it should be doing.

Monday 28th June

Today I showed my system to my project supervisor and attempted a live demo. I found a race hazard bug in the code during the demo that meant things didn’t go according to plan, I am now determined to track down that bug. I finished all of the testing, and I am now convinced that the system works, as it should. I now need to evaluate the system and improve it so that it works the best way it can. This includes doing a thorough code review of the system as I mentioned earlier.

Wednesday 1st July

Today was a very productive day. First of all, I figured out the race hazard bug that I was having. The issue was that the algorithm I was using to wait for queries to be applied played havoc with the internal workings of the Akka framework. It turns out that there’s a built in way to fix this problem, although I had to learn about futures in Scala to implement it., also, ’ve begun comparing the different models for basic availability that I could use. There doesn’t seem to be an obvious solution like there was for eventual consistency, so the end result is that I’ll pick and choose elements of several implementations in my eventual system.

Thursday 2nd July 2015

Today, I’ve made massive improvements to my system. I’ve now changed the web services so that they run asynchronously using futures rather than hogging up the web server. This means that my application can now process several requests at once rather than just one a time. Another major benefit that has come out of today is the code to make everything consistent has become much simpler and I am also more confident it works effectively. This was to fix a bug where the database was not letting deletes go through if the data was already deleted, so I made some changes to fix this. Both of these issues could have been avoided had I learnt the tools better before I started the project, but then I felt that there is a balance between reading and actually doing the practical work that needs to be struck.

Monday 6th July 2015

Over the last few days, I’ve been doing some research into basic availability and how I’m going to implement this in my application. I’m now thinking that this should be much easier to implement than eventual consistency was. This is partly because I am taking elements from several of the best models of basic availability from several real database examples. therefore, I have been able to tailor my model of basic availability so that it plays nicely with the rest of the program. I’m not sure, however, if that’s because I’ve already designed the basic architecture of the application, or because the functionality is easier to implement. A key thing that I’ve learnt though is that maybe I’m not as bad at system design work as I thought I was at the beginning of the project. The reason this is because I only need to modify one component of the system, with minimal changes everywhere else. This process has also enforced the importance to me of cohesion and coupling as taught in CO871.

Wednesday 15th July

I returned from a week’s holiday in Crete yesterday. Therefore, most of my time today has been spent going through all the sections of my project to bring myself back up to speed from where I left off. I found this process extremely useful, because it allowed me to look at the quality of my code and pinpoint the bits that were complicated or needed refactoring. My theory is that if I can’t understand a section of code after a week on holiday, then it is very unlikely that the examiners will be able to understand it without having seen it before. This means that while I might not have fully completed updating the project plan based upon my supervisor’s feedback, I learnt a great deal about code quality and software development in general.

Thursday 16th July

Today I started implementing the features of basic availability that I designed before I went on holiday. After reviewing my designs, I needed to make two small adjustments. I firstly needed to consider what it means for a component to fail. This is because my application runs in one JVM, so taking the whole JVM down would crash the whole system. I therefore designed a system whereby upon receipt of a database update, my system runs an algorithm to decide whether it should be accepted or not. If not, then it is sent to retry and it is recorded. I don’t think this work will take as long as I originally thought it would, partly because I am mainly extending existing components of the system and partly because the loose coupling within my system means that the new component I designed can be slotted into the system without too much difficulty.

Friday 17th July

Today I implemented the general infrastructure for basic availability. Some of this work went fairly smoothly. The settings manager that I implemented to manage the greater degree of customization that my system now supports particularly springs to mind. I did have some problems testing my system. This is because, when I tested the new component I had written, everything worked fine in the unit tests I wrote but deadlocked in the live system. The bug was easy enough to fix, but It reminded me that I need to think carefully about how any new components that I implement fit into the overall package.

Tuesday 21st July 2015

Today I carried on with my implementation of basic availability, Over the weekend, I thought long and hard about technical motivations for my system as I was instructed to do by my supervisor. The biggest thing that I found that would motivate the development of this is the setting of targets. As an example, if data requests to the various servers are likely to be lost 75% of the time, can we hit our basic availability targets? I could then evaluate the performance of my system against the real ones provided by Amazon’s cloud platforms to see if there would be any major difference. In other news, I’ve finished the main implementation of basic availability, but still need to write the unit tests that show it working effectively. I’ve seen individual components working based purely on log outputs, but I haven’t seen the whole thing work as it should. I think I need to fine tune the system parameters a little more to get a more accurate impression of whether the system works or not, and from there start my evaluation. I also did a run through with the static analysis tool lately and found that it really reinforced the points I learnt in CO894. There are a lot of false positives generated by the system at the minute. That’s not to say that the system isn’t useful as it’s pointed out several bugs to me that I hadn’t spotted but it does mean that I need to be vigilant when going through the reports to ensure I don’t overlook anything.

Wednesday 22nd July 2015

Today I had a very productive meeting with my supervisor. I was able to show her some of the user stories that I had created in response to her feedback, and she was able to point me in the right direction as to what still needed changing. I’ve found this part of the project extremely useful. During my industrial placement, these sorts of things were not always taken as seriously as they should be, so it’s been really useful getting some feedback on my project planning skills.

Friday 24th July

Today I’ve finished my implementation of basic availability. However, I’ve found that there are several problems with what I have. I’m currently working through several bugs related to my use of futures within the application. A future is basically a computational device provided by the Akka framework that lets me send a message to an actor within my system and then wait for a reply. While waiting, the program can be getting on with other work. For the most part, they seem to work fairly well. The problems I’m having are that there seem to be several ways of doing the same operation which can make reading through the documentation confusing. The API for futures also uses several advanced features of the Scala language, and misuse of these can lead to strange results.

Tuesday 28th July

Today I finally cleared out all the bugs for basic availability. It turns out that the bugs I was experiencing had nothing to do with the usage of futures at all. The problems were more to do with my understanding of the way Akka handles messages compared to Erlang. In Erlang messages are matched for the first pattern that is reached in the list of receive clauses. In my cluster overseer actor, I had a bottom clause that sent any message directly on to all of the child actors. I thought the Akka framework would use the same protocols as Erlang, as it does in many other aspects, so I was surprised when all my communications back to the master actor failed. As it turns out the messages were being picked up, they were just being resent back to all of the child actors again. I solved the problem by adding handlers for all the messages that I wanted to pass on to the children, but I intend to post on Stack Overflow to find out if there’s an easier way of doing this.

Wednesday 29th July

Today I’ve been doing a lot of research online to figure out how to use Amazon’s web service. The API seems simple enough to use, but at the minute I’m getting an error that I haven’t provided a security key. These sorts of things are to be expected though, as the Amazon API is designed for real world use, and as such has many more pieces of functionality than my own small system.

Thursday 30th July

Today I’ve managed to solve the security key issues that I was having yesterday, but now I’ve hit a new problem. I created some dummy data for the application to use manually in the console that comes with Dynamo. Then when I try to query this data with the application it is not there. The downside of this is that it is taking me a lot longer than I thought to get to grips with Amazon Web Services (AWS). On the plus side though, I should have a great deal of understanding of the technology should I need to use it again in a future project.

Friday 31st July

Today was a really useful day, I found a new book in the library called ‘Effective Akka’, that has allowed me to improve upon a great deal of my code. In particular, I finally figured out how actors normally choose which messages to process next. I was also able to refactor the code I was using in the backend class so that I could be more efficient. This newfound knowledge helped me fix a bug that had developed in my eventual consistency code.

Monday 3rd August

Today I’ve managed to create a Scala trait for my sample programs. An implementation of this trait will be passed into my sample programs for database access, so that I don’t need to keep rewriting them each time they need to run. I had a really interesting meeting with my supervisor today. She suggested I write a program that translates into code a formal specification that came up in the original background review. We could then see if my system meets that description of consistency.

Wednesday 5th Aughust

I sent most of today and yesterday running my examples and wondering why they didn’t work. It tuns out the problem isn’t in my logic, but in the order I do things. Originally, when I retrieved data from the database, I ran the SQL query and then evaluated all the updates to get the eventualy consistent results. This worked most of the time, but if you select only a subset of the columns, but then include other columns in the where clause, too much data is returned because the where clause is never evaluated. What I therefore did is to get the database to pull down all the data, and then filtered out all the columns I didn’t need. I do wish that I had thought about this in my design more, as this would have saved me a lot of hassle in the long run.

Thursday 6th August

Today I ran the first of my experiments. This is a simple program that counts mthe number of attempts it takes before an inconsistent result is found. This worked surprisingly well, and the ruslts matchec my own hypotheis. I need to find a way to automate this exprirment more effectively though. As currently the timings are a bit all over the place.

Friday 7th Aughuts

Today I an the second of the two exprriments that I have devised. This produced more interesting results as the results I got from Amazon were much slower than those I got from my own system. Amazon took a lot longer to find inconsistent results. I think this s because eof scale and the fact that Amazon makes things consistent more quickly than my system does. I believe that’s to do with scale, amazon has thousands fo servers to run against, wheras my system only currently uses three.