**Technology Review**

When I started this project, I didn’t have a clear picture in my heads of where to go to find more information. My project supervisor pointed me in the direction of Joe Celko’s[1] book on cloud databases. This allowed me to gain a basic grasp of the relevant background information for my project.

Finding information on the relevant project itself was difficult. This is because no one has tried to build an emulator like this before. This means that there did not appear to be a lot of material available to me initially. Once I clarified what the word ‘Models’ meant in the original project brief, I began to find more useful information.

I found The Cloud Handbook{2] to be particularly useful. The main reason for this is that it provided descriptions of real cloud systems and the extent to which they implement the BASE propertied as well as brief descriptions of the algorithms themselves. This allowed me to narrow my search for information considerably.

One of the most useful resources was Amazons cloud database. The reason for this is that the paper describes the approach to ensuring consistency is achieved well and in great detail. While reading the appropriate chapter of 7 Databases in 7 weeks, I also was able to download Risk, a database that is built around the same consistency model that Amazon uses.

Another highly useful approach was that provided by CouchDB. I found here that the CouchDB handbook, freely available online, provided thorough and in depth descriptions of the technical algorithms used, and I was able to start coming up with some ideas on how I might implement this.

Another approach that I considered was Google’s model of achieving eventual consistency. While I find the algorithms employed here to be particularly clever, I haven’t decided to implement them for this project. This is due to several reasons. Firstly, the Google ecosystem that supports GFS is highly complex, not only encapsulating the database, but also a lock server and various other components as well. This means that it would be too complex to emulate within the time available to me and also require the system to have a much closer access to the database. This might not be possible due to the fact that ultimately, the data will be stored on a standard ACID database.

Another key component of the literature review for me was finding out about the programming languages and tools that I would be using. Scala for the Impatient, and Akka Concurrency were found to be essential in getting a solid grasp of the Scala programming language down. I also found it useful to refer to Ian Somerville’s Software Engineering for ideas and advice on how to plan such a large piece of work.

**Anderson, C. J., Lehnardt, J. and Slater, N. (2010) *CouchDB: The Definitive Guide*. Sebastopol, CA: O’Reilly Media, pp. 11–20.**

This book chapter gave a great deal of information on the consistency model for CouchDB. I found the information here to be clear and concise. I believe that it gives me enough information to implement at least the consistency model for CouchDB as well as the availability model for my project, if I wanted to do that. I particularly liked the fact that the consistency model is based on version control, meaning it should be relatively straightforward to implement should I choose to do so. This was one of the earlier resources I looked at, and I found it good to contrast it with Amazon’s Dynamo database, especially with regards to consistency. Judging on what I have read, I wil; most likely choose either the consistency model implemented by Dynamo or CouchDB in the actual emulator.

**Burckhardt, S., Leijen, D., Fähndrich, M. and Sagiv, M. (2012) ‘Eventually Consistent Transactions’, *Programming Languages and Systems*. Springer, pp. 67–86. doi: 10.1007/978-3-642-28869-2\_4.**

This paper gave a good overview of what eventually consistent transactions were and also went into great detail as to why they are so useful.It was one of the resources found early in my research. The eventual consistency model described here is similar to the one used by CouchDB. While the paper took a while to digest because of the large amount of mathematics involved, I found the examples at the end to be particularly useful.

**Celko, J. (2013) *Joe Celko’s Complete Guide to Nosql: What Every SQL Professional Needs to Know about Nonrelational Databases*. United States: Morgan Kaufmann. pp 3-14.**

This was the second text I read in my project research. It helped me solidify my understanding on what BASE and ACID transactions were, and why BASE transactions were preferred over ACID transactions in certain situations. It also introduced me to several important pieces for background information, such as the CAP theorem.

**Chang, F., Dean, J., Ghemawat, S., Hsieh, W., Wallach, D., Burrows, M., Chandra, T., Likes, A. and Gruber, R. E., Google (2006) ‘Bigtable: A Distributed Storage System for Structured Data’, in *OSDI*.**

After reading about the Google file system in the Cloud Handbook, I then read the Google research paper on Bigtable to get more information. Whilst I found the algorithms here extremely clever I think that it might be a bit too challenging for me to implement within my given timescale.

**DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P. and Vogels, W. (2007) ‘Dynamo: Amazon’s Highly Available Key-value Store’, *ACM SIGOPS Operating Systems Review*, 41(6).**

Amazon’s Dynamo database was the first cloud system that I looked at to see how eventual consistency was implemented after I had done the preliminary background reading. The thing that struck me about the Amazon Dynamo database is the consistency model. This was one of the texts that got me enthusiastic about this project, as I began to realize just how difficult it is to keep a database in a valid state. I also found the algorithm that is used regarding vector clocks both clever and interesting.

**Edlich, P. D. S. (*NOSQL Databases*. Available at: http://nosql-database.org (Accessed: 1 April 2015).**

This website was an incredibly invaluable tool for my research. The reason for this is because it provides a list of most of the NoSQL databases currently out there. From this list, I was able to pick out the databases that supported some form of BASE transactions and figure out if they had models that were worth implementing.

**Furht, B., Escalante, A. and editors. (2010) *Handbook of Cloud Computing*. Edited by B. Furht and A. Escalante. Boston, MA: Springer Science+Business Media.pp 138-158**

Once I understood what ACID and BASE transactions were, this book proved invaluable in allowing me to gain key information on some of the biggest cloud databases. It was useful because it provided a simple overview of each database, and often said whether it implemented any of the properties of BASE transactions or not. This meant that I got a brief overview to each of the major cloud systems, which I could then research in more detail using other resources.

**Horstmann, C. S. (2012) *Scala for the impatient*. 1st edn. United States: Addison-Wesley Educational Publishers Inc.**

I have decided to use the programming language Scala for this project. The reasons are that it has a clean syntax, and it has excellent support for concurrency. In particular, it has the Akka framework, which I could easily use to model different nodes of a replicated database. This book has helped me get up and running quickly on the basics of the language.

**Kalin, M. (2009) *Java Web Services: Up and Running*. United States: O’Reilly Media, Incorporated.**

One of the eventual aims of my project is to implement a web service. Therefore, I have done some light reading on how to do this as part of my background research. The reason why I have chosen a Java book here is because Scala is fully interoperable with all standard java classes.

**Kuznetsov, S. D. and Poskonin, A. V. (2014) ‘NoSQL data management systems’, *Programming and Computer Software*. Springer, 40(6), pp. 323–332.**

This paper is here because it contains a round up of some of the databases that I have not been able to find much information on, such as Risk and MongoDB. It provides an overview of how these systems work, which might be useful if I wanted to compare the performance of my solution to one of these further down the line.

**Pritchett, D. with eBay (2008) *BASE: An Acid Alternative - ACM Queue*, *ACM Queue*. Available at: http://queue.acm.org/detail.cfm?id=1394128 (Accessed: 30 March 2015).**

This was the first document given to me by my supervisor. It provided my first look at what BASE transactions are. A lot of it I found difficult to get through at first, but other documents listed helped things become clearer over time.

**Redmond, E. and Wilson, J. R. (2012) *Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement*. Lewisville, TX: The Pragmatic Programmers.**

I found this book particularly useful once I had looked at all of the major cloud databases. The reason for this is because the majority of the databases this book chooses to study are NoSQL databases that import some form or part of BASE transactions. Therefore, this book was useful in getting more information on certain algorithms or techniques that might not be gained in other publications. An example is one of the databases studied is the risk database which is based on Amazon’s Dynamo database. As a result, it gives a very detailed description of the vector clocks algorithm for consistency used by Dynamo, whereas the official paper by Amazon only gives more of an overview. It also shows how these databses are used in practice, which makes the theoretical concepts easier to digest.

***Sharding and replication* Available at: http://rethinkdb.com/docs/sharding-and-replication/ (Accessed: 1 April 2015).**

This webpage provides a detailed look at Rethinks replication strategy. This is a possible candidate for a model of basic availability that I could implement if I get the time.

**Sommerville, I. (2011) *Software Engineering*. 9th edn. Harlow: Pearson Education (US).**

The main technical outcome of my project is to develop a software system. I have therefore found this book useful in thinking about what quality controls and processes need to be followed. It has also been useful in finding out ways in which developing web based systems is different from traditional software development.

**Stirling, S., JavaWorld, S. S. and PT, 2004 1:00 AM (2004) ‘Testing J2EE applications’. JavaWorld. Available at: http://www.javaworld.com/article/2072923/testing-debugging/testing-j2ee-applications.html (Accessed: 1 April 2015).**

This article has been useful in figuring out how I should go about testing the software system that I am planning to develop. This is important, as I have not had much previous experience of testing a large complex web application.

**Wyatt, D. K. (2013) *AKKA Concurrency*. Canada: Artima Inc**

I have been looking at the Akka concurrency framework as a way of modeling my application as I wanted an easy way of building the system. As an example, it is not hard to imagine different actors on different hosts, or different actors communicating with each other to ensure consistency.