**Module Reflection**

Unit 1-2: Exploring Key Information System Concepts

The content covered in the first two units presented the fundamentals of an Information System (IS), and the ways ISs might fail.

Bourgeois (2014) explained that an IS exists as a combination of 5 main components: hardware, software, data, people, and processes. The rest of Bourgeois’ book, “Information Systems for Business and Beyond” was covered, where he performed a cursory overview of the various components mentioned above. I found the book helpful in explaining the holistic process of using information and its representive data to produce value. The required reading from Sommerville’s (2016) “Software Engineering” introduced the software component mentioned above as it pertains to its development within the field of Software Engineering.

The content covered guided me in learning the major components of an IS, how they interrelate, how they can fail, and their importance to businesses and society in general. Furthermore, I found it crucial to take note of the fact that for an IS to be effective, the business processes that undergird and govern the use of them need to be efficient and all-encompassing respectively. This point and the rest of the learning helped me to see the various IS inefficiencies within my own organisation, where disparate systems produce duplication of activities. Furthermore, the automation and business workflows that should be provided by the ISs have been broken by various manual activities not linked to the IS.

Unit 3-4: The Object-Oriented Paradigm

In unit 3 and 4 the Object-Oriented (OO) model of creating and implementing software was covered. Many principles and their related constructs which I had previously covered in a Python certification course were here reiterated. For instance, class and instance variables, methods, inheritance, encapsulation, and polymorphism. However, I experienced further development by now being required to understand design principles for an OO system, where relationships of the envisaged modules or objects had to be formed to produce specific abstractions and functional flows. It was challenging for me to understand the types of relationships: dependency, association, aggregation, and composition; without first seeing these abstract concepts through code examples. Therefore, I had to extensively browse the Internet for appropriate and well explained illustrations.

Being required to understand the referencing of variables within Python as the interaction of objects significantly improved my grasp of OO programming concepts. Furthermore, I can now plan my code a lot better having learnt the modularity and relational principles indicative of OO design and programming. However, the design element can seem very intuitive and producing the right classes with their respective relationships will certainly require further practice.

Unit 5-6: Universal Modelling Language (UML)

Unit 5 and 6 covered the UML and its various notations, specifically as it pertains to the Use Case, Class, Object, Sequence, Activity, and State Chart diagrams.

UML provides a standardised method to model the previously learnt concepts of OO programming when required to design a OO system (Ambler, 2003). The various diagrams help by explaining certain aspects through focussing on different elements, and provide a means whereby non-technical and technical personnel may engage when creating a system (Ambler, 2003).

I had to apply the concepts learnt from unit 1 up to this point to be able to produce the design for my online store IS required as the mid-module assignment. It was particularly difficult to produce the design, not because of a lack of understanding of the concepts and notations covered, but due to not having sufficient experience with using best practices as it pertains to creating objects and interrelations, especially in the abstract way of a UML representation. Therefore, more exposure to implemented designs as well as experience in creating my own will be required to master this skill. However, I find UML a very useful tool that will aid me in future to model a system during the design phase of the Software Development Lifecycle (SDLC).

Unit 7-8: Relational Database Design

Unit 7 and 8 covered various general concepts pertaining to databases but focused on the Relational Model and the techniques required to design a database of this type.

Databases provide an information system with various functionality not easily available through normal file-based systems (Connolly & Begg, 2014). For instance, moving data to a database for persistence, where a Database Management System (DBMS) is the key piece of software for controlling various aspects, provides better data integrity and consistency, easier sharing of data, and better security (Connolly & Begg, 2014). The relational model has been the dominant database paradigm for more than 20 years, and uses the idea of setting table structures to house different data fields (Connolly & Begg, 2014). These structures are then linked through different fields representing the same values (Connolly & Begg, 2014).

I learnt the importance of normalising a relational database to at least the third normal form (3NF), and the various ways one might want to link tables by setting up primary and foreign keys. Additionally, I now feel comfortable with designing a relational database. Furthermore, it was very interesting to relate the concepts learnt back to the work management IS as well as the SAP IS used in the company I work for, and to think how the underlying database structures could reside.

Unit 9-10: Database Implementation and Manipulation

The concepts learnt in the previous two units were here engaged with in a more pragmatic approach through exposure to the Structured Query Language (SQL), and the open-source MySQL Relational Database Management System (RDBMS).

SQL provides the two important facets of a database query language: data definition and data manipulation, where the former provides the means whereby one can create the tables structures, and define the type of data that will reside in them, as well as the relationships that will exist between them (Connolly & Begg, 2014).The latter enables the insertion and manipulation of information whether using another high-level coded program with SQL statements, or connecting directly to the database through a local command line client (Connolly & Begg, 2014).

I found creating and manipulating a database using SQL very interesting and straightforward, however applying the last four units’ concepts in the development of my Online Store required reengineering, as the database design was not originally accounted for in my first thought processes. When incorporating a database into my system certain functions were a lot easier to perform, but the OO modelling previously used seemed to breakdown slightly. This due to the facilities now available through a separate system where data could be fetched and manipulated at any point in the code, which does not align the with the principle of private attributes in OO programming.

Unit 11: Web Development with Python (92)

Unit 12: Emerging Trends of Information Systems (92)

References

Ambler, S. (2003) The Elements of UML Style. Cambridge: Cambridge University

Press.

Bourgeois, D. (2014) Information Systems for Business and Beyond. Washington: The Saylor Academy.

Connolly, T., Begg, C. (2014) *Database Systems: A Practical Approach to Design, Implementation, and Management*. Pearson Education available

Sommerville, I. (2016) Software Engineering. 10th ed. Essex: Pearson Education

Limited.