Pearson Institute of Higher Education

Module Guide

Faculty	Information Technology		
Module Code	ITOO211	Module Name	Systems Analysis and Design
NQF Level	6	Credit Value	12
Semester	1/2020	Year Level	2
Module Leader	Dr Kebashnee Moodley	Copy Editor	Mr Kevin Levy
Lecturing Hours	56 (04 hours a week)	Tutorial Hours	N/A
Notional Hours	120	Pre-Requisites	N/A

The module guide must be read in conjunction with the prescribed textbook. This document will be the first port of call to understanding what will be assessed and which assessments form part of the module.

The purpose of the module guide is to highlight:

- The learning outcomes and assessment criteria that need to be met to pass the module
- The assessment required to be completed for the module
- The additional resources required for the module
- The topics that will be focused on for the module

Module Aim

The aim of this module is to provide students with the knowledge and skills to undertake a systems investigation and to conduct process and data modelling techniques during systems analysis and design.

Module Description

Information systems analysis involves focusing on a business problem and the requirements necessary to develop a system that will meet those requirements. Information systems design focuses on the technology that will be applied to implement a solution to the business problem.

You will explore the purpose of conducting a feasibility study and how to determine which systems development option would meet the business need. You will also understand the use of different fact-finding techniques for stakeholder requirements elicitation.

Systems analysts apply the SDLC (Systems Development Life Cycle) to develop information systems. You will examine the use of a traditional methodology (Waterfall model) and some agile models and techniques that could be applied during an information systems development project.

Learning Outcomes

By the end of this module, you will be able to:

Learning Outcomes	Assessment Criteria
1. Compare different systems life cycles.	1.1 Evaluate different systems life-cycle models. 1.2 Discuss the importance of following a procedural/staged life cycle in a systems investigation.
	Blended learning activity: • Flipped classroom: online activities and group discussions during lectures
2. Discuss the importance of a feasibility study.	2.1 Discuss the components of a feasibility report. 2.2 Assess the effect of different feasibility criteria on a systems investigation.
	Blended learning activity: • Flipped classroom: online activities and group discussions during lectures

3. Perform a systems investigation.	3.1 Undertake a systems investigation to meet a business need. 3.2 Use appropriate systems analysis tools and techniques to perform a systems investigation. 3.3 Create documentation to support a systems investigation. 3.4 Evaluate how user and systems requirements have been addressed.	
	Blended learning activity: • Flipped classroom: online activities and group discussions during lectures	

Prescribed Resource(s)

Textbook(s)

Valacich, J. George, J. and Hoffer, J.A. 2015. *Essentials of Systems Analysis and Design*, *Global Edition*, 6th edition. Essex: Pearson Education. ISBN: 9781292076614.

The following resource(s) will be made available on *my*LMS, which you must check regularly:

- Assignment specification
- Blended learning items
- Continuous assessments
- Exam scopes
- Important notifications from your lecturer
- Module guide
- Module announcements

Recommended Resource(s)

Take note that all disciplines and their corresponding textbooks are frequently updated. Therefore, you should use the latest editions, where available. Recommended resources should be used for research purposes. There is a range of resources related to this module, including the following:

Textbook(s)

Dennis, A., Wixom, B.H. and Roth, R.M. 2014. *Systems analysis and design*. 6th edition. New Jersey: John Wiley and Sons.

Lejk, M. and Deeks, D. 2002. *An introduction to system analysis techniques*. 2nd edition. Boston: Addison Wesley.

Whitten, J.L. and Bentley, L.D. 2007. *Systems analysis and design methods*. 7th edition. Boston: McGraw-Hill Higher Education. ISBN: 9780073052335.

Online Document(s)

Whitten, J.L. and Bentley, L.D. 2007. *Systems analysis and design methods.* 7th edition. Boston: McGraw-Hill Higher Education. ISBN: 9780073052335. Student resources: [Online] Available at: http://highered.mcgraw-hill.com/sites/0073052337/student_view0/ [Accessed: 5 November 2018].

Website(s)

Web pages provide access to a further range of Internet information sources. Lecturers may download the web-related material for you to access offline. You must use this resource with care, justifying the use of information gathered.

Freetutes. 2018. *Systems analysis and design*. [Online] Available at: www.freetutes.com/systemanalysis/ [Accessed: 5 November 2018].

Supporting Documents

Geyer, L., Levin, A., Makati, P., Pierce, R., Potter, M., and Wheeler, A. 2019. *PIHE Guide to Referencing (Harvard Referencing Method).* Unpublished document. Pearson Institute of Higher Education

Essential Requirements

- Access to a resource centre or a library with a wide range of relevant resources including textbooks, newspaper articles, journal articles, organisational publications and databases.
- Access to a range of academic journals in electronic format via PROQUEST or other databases.

ICT Requirements

ICT Required	Reason	Lecture Week(s)
PC with internet access	For research, class activities and completing the assignment.	1-13
Microsoft Office 2016		
Microsoft Visio 2016		

Formative Assessment(s)

Continuous Assessments

Continual formative assessment is conducted so that you are given feedback on your progress in the achievement of specific learning outcomes. The formative assessment tasks occur every fortnight and can take the form of one of the following:

- A five-item multiple choice test
- A short-questions test
- Construction of concept maps
- Take home tests with long questions
- Short practical tasks
- Short class presentations

Students could be expected to complete assessments on *my*LMS as well as other digital platforms.

Guidelines for online *my*LMS assessments:

- Time limits should be checked before commencing assessments.
- Ensure that the Internet connection is stable.
- In some cases, assessments are not available indefinitely and will only be available for a day or two.
- Marks may only be available (with a memorandum) after all students have attempted the assessment after the assessment due date.
- Two attempts may be awarded in cases where there is poor Internet connection. Note that no more than two attempts may be awarded in some cases.

Test(s)

There will be one test for this module which will count 20% towards the final mark.

If a test is missed because of illness, a doctor's note must be presented within 48 hours of the missed test to the Academic Manager/Administrator/Coordinator.

To make up for this missing assessment, you may be able to write a deferred test. However, in order to gain entry to this test, you will have to follow various procedures and meet certain criteria. You must complete a *Deferred Test Application Form* available on *my*LMS. You will be required to pay a non-refundable fee per application. Each test missed requires a separate application. This will be your only opportunity to make up for a missed test.

It is the students' responsibility to collect their tests and verify their marks on the day they are handed out. No adjustment of marks will be entertained beyond the date scripts are returned to students after marking.

Assignment(s)

There is only assignment for this module. This assignment will be completed individually. The assignment will test various theoretical concepts and application of theory to scenarios. In order for students to achieve a 50% (pass) on the assignment they should spend approximately 10 – 15 hours on the whole assignment. This assignment will count 20% towards the final mark.

Assignments must be submitted on or before the due date to the lecturer in class or as per arrangement. Five percent (5%) will be deducted for every day that the assignment is late. Assignments that are more than a week late will be awarded a zero. Late submissions must be accompanied by a medical certificate.

Summative Assessment

Summative assessment is concerned with the judgement of learning in relation to the exit-level outcomes of the qualification. Such judgement includes integrated assessment(s), which test your ability to integrate the larger body of Information Systems knowledge, skills and attitudes that are represented by the exit-level outcomes as a whole.

Special Project

The special project is a take home project with a submission time of 72 hours and this will count 50% towards the final mark.

Plagiarism

All assignments and reports must be submitted to the online similarity checker (Turnitin) available on *my*LMS prior to being submitted for marking. When submitting your assignment/report, it is compulsory to submit the entire Turnitin report. Marks will be deducted in accordance with the institutional policy.

Also, when submitting assessments, you should include the completion and signing of the applicable Assessment Coversheet as an acknowledgement that the work submitted is your own original work, except for source material explicitly acknowledged. This declaration will serve as proof that you are aware of the Institution's policies and regulations on academic integrity.

Final Mark

In order to pass the module, a sub-minimum mark of 40% or higher is required for the examination and a final average of 50% or higher is required for the entire module.

The final mark is calculated as follows:

Coursework Mark [(Continuous assessment percentage × 0.10) + (Take-Home Test percentage × 0.20) + (Assignment percentage × 0.20)] + **Special Project Mark** [(Special Project percentage × 0.50)]

Details of Assessments

Methods of Assessment	Weighting[1]	Dates
Semester 1		
Assignment	20%	11/05/2020 – 15/05/2020 Scope of coverage: Weeks 1 – 7
Take-Home Test	20%	04/05/2020 – 08/05/2020 Scope of coverage: Weeks 1 – 4
Continuous Assessment	10%	Lecturer will stipulate the date(s) of these assessments and scope of coverage.
All formative marks submitted		Semester 1: 22/06/2020
Special Project	50%	08/06/2020 – 12/06/2020

Putting Together a Portfolio of Evidence

You must demonstrate, through the presentation of evidence, that you have met all module requirements within the qualification being undertaken. To do this, you must organise your evidence into what is known as a 'portfolio'.

A portfolio will take time and effort to complete. It is a means of focusing and demonstrating to others your strengths and achievements. A portfolio is an important resource that you may find useful to retain once you have achieved your qualification, particularly when applying for future positions.

You are encouraged to read more about building a portfolio and to begin populating your evidence to illustrate your full skill-set to future employers.

Consultations

Consultation times will be pinned onto the lecturer's office door/notice board. You must give lecturers 24 hours' notice for appointments. Meetings can be requested in-class or via email. It is important that you detail the requirements (chapter, section, etc.) for your consultation.

Module Content

You are required to attend all classes. In addition, exercises and activities, which are supplied by lecturers, are compulsory.

[1] Refer to the **Conditions of Enrolment**, available on *my*LMS.

Continuous assessments may run throughout the semester.

Semester 1: Schedule

Lecture Weeks	Topics and Assessment Criteria Covered	Assessments	Textbook References
1 1: 03/02/2020 – 07/02/2020	The Systems Development Environment AC: 3.1		• Chapter 1
2 1: 10/02/2020 – 14/02/2020	The sources of software Managing the Information Systems Project AC: 1.1, 1.2		• Chapter 2 • Chapter 3
3 1: 17/02/2020 – 21/02/2020	Systems Planning and Selection AC: 2.1, 2.2		• Chapter 4
4 1: 24/02/2020 – 28/02/2020	Determining System Requirements AC: 3.1, 3.2, 3.3, 3.4		• Chapter 5
5 1: 02/03/2020 – 06/03/2020	Process modelling AC: AC: 3.1, 3.2, 3.3, 3.4		• Chapter 6 • p. 180 - 185 • Appendix A
6 1: 09/03/2020 – 13/03/2020	Process modelling AC: AC: 3.1, 3.2, 3.3, 3.4		• Chapter 6 • p. 186 - 191 • Appendix A
1: 16/03/2020 – 17/04/2020	**ACADEMIC SHUTDOWN/RECESS**		

7 1: 20/04/2020 – 24/04/2020	Process modelling exercises AC: 3.1, 3.2, 3.3, 3.4		• Valacich et al. extra notes (Chapter 6, 2015) • Appendix A
8 1: 28/04/2020 – 30/04/2020	Data modelling AC: 3.1, 3.2, 3.3, 3.4		• Chapter 7 • p. 220 - 227 • Appendix B
9 1: 04/05/2020 – 08/05/2020	Data modelling AC: 3.1, 3.2, 3.3, 3.4	Take-Home Test	• Chapter 7 • p. 228 - 233 • Appendix B
10 1: 11/05/2020 – 15/05/2020	Data modelling AC: 3.1, 3.2, 3.3, 3.4	Assignment due	• Chapter 7 • p. 234 - 237 • Appendix B
11 1: 18/05/2020 – 22/05/2020	Data modelling exercises AC: 3.1, 3.2, 3.3, 3.4		• Valacich et al. extra notes (Chapter 7, 2015) • Appendix B
12 1: 25/05/2020 – 29/05/2020	Process and data modelling exercises AC: 3.1, 3.2, 3.3, 3.4		• Valacich et al. extra notes (Chapter 6, 2015) • Valacich et al. extra notes (Chapter 7, 2015)
13 1: 01/06/2020 – 05/06/2020	**Revision**		
14 1: 08/06/2020 – 12/06/2020	**Special Project**		
15 1: 15/06/2020 – 19/06/2020	**Revision: Mastering Concepts**		
06/07/2020	**Semester 2 Teaching Period Continues**		

Appendix A

This appendix should be referenced when studying the content and the assessment criteria for the week(s) listed below.

Week	5 - 8
Learning Outcome	LO3
Assessment Criteria	3.1, 3.2, 3.3, 3.4

Process Modelling

Valacich, J.S.; George, J.F. & Hoffer, J. 2015. *Essentials of systems analysis and design, global edition.* 6th edition. Pearson Education. Chapter 6.

Introduction

In many industries, professionals working on new projects have to plan and analyse a situation carefully before developing a solution for it. In the medical profession, doctors have to carefully plan an operation before performing actual surgery on a patient. In the industrial architecture industry, architects plan the project and then analyse and design the project by drawing models (graphical representations) of the building before moving into the construction phase. This trend of planning, analysis and design can thus be found in many industries.

In the IT industry, systems analysts also draw logical and physical models (various diagrams) of the information system before going into the construction phase, where the programmers then build the system according to the specifications completed by the analysts. Drawing diagrams supports analysts to analyse the new system from various perspectives and focus on different aspects. Analysts use modelling software tools (CASE tools) to assist them in constructing complete and consistent diagrams. Various types of diagrams are useful, each focusing on a specific aspect of a system.

Process Modelling

A **DFD** is a key tool for modelling the business and system processes. It shows the processes in a system, with its inputs, outputs and data (Whitten & Bentley, 2007:162). A DFD is also called a "process model", as this type of diagram models processes. A systems analyst usually draws different levels of DFDs. The analyst will break the system down into sub-processes until the lowest level is reached wherein the most detail is captured. This is usually done by constructing a decomposition diagram and then using the decomposition diagram to create DFDs and other diagrams.

A context diagram is the highest level DFD. Lower level DFDs are drawn from the context diagram to analyse the processes in more detail. A systems level DFD is the next level, and the processes can be broken down into sub-processes until the lowest level DFD is drawn, which is referred to as the primitive level DFD.

The following terms are used during process modelling:
Data flow: Depicts data that are in motion and moving as a unit from one place to another in the system (single piece of data or logical collection of information).

Data store: Depicts data at rest and may represent data in a file folder, computer-based file or notebook. Four actions are usually carried out on data stores. This is referred to as CRUD (Create; Read; Update; Delete). When using Create, Update and Delete, the direction of the data flow will move towards the data store. When using Read (to view data), the direction of the data flow will move towards the process.

Process: Depicts work or actions performed on data so that they are transformed, stored or distributed. Each process should perform only one activity. A process should always start with a verb.

Source/sink/external entity: Depicts the origin and/or destination of the data. It can be a person, organisation or another system that is external to the system but interacts with it.

Decomposition Diagram
Decomposition was discussed in Unit 1: Section 1.3. Systems analysts find it useful to first draw a decomposition diagram to understand the system with its related sub-systems and processes. By breaking the system down into its component parts, the analyst increases their understanding of the system being analysed. Figure 1 is an example of a decomposition diagram for an online cell phone apps store.

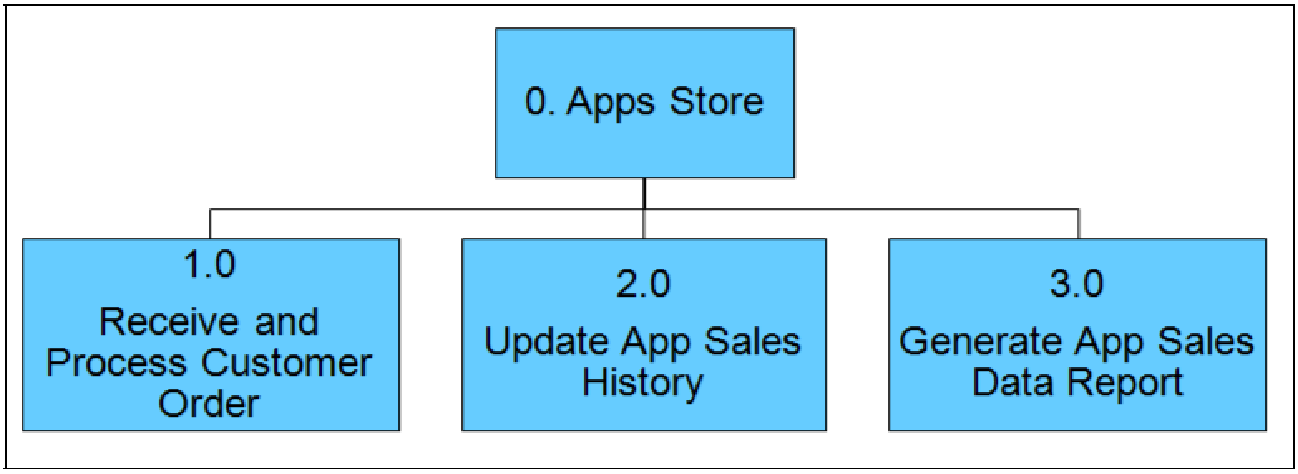

Figure 1 – Decomposition diagram example

Source: Visagie (2018) & Valacich *et al.* extra notes (2015: Chapter 6:6-15)

The decomposition diagram helps the analyst to plan the breakdown of sub-systems and processes which can then be used to start drawing the context diagram and the rest of the DFDs. Figure 2 shows the usefulness of the decomposition diagram in identifying sub-systems and processes which can then be used to construct the different levels of DFDs.

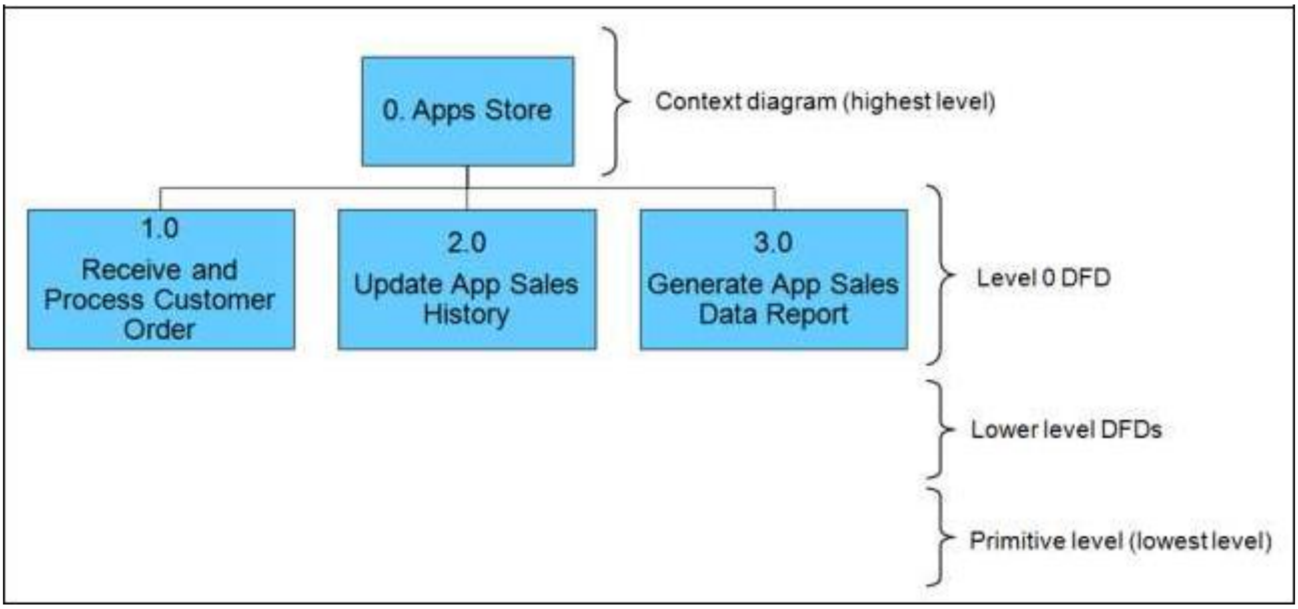

Figure 2 – Decomposition diagram and DFD diagram levels

Source: Visagie (2018) & Valacich *et al.* extra notes (2015: Chapter 6:6-15)

Context diagrams (Context DFDs)

A context diagram is the highest level DFD, and it shows the system boundary, overall scope, by including one overall process (usually the system name; numbered with zero), external entities, input data flows and output data flows. No data stores are shown on a context diagram. This diagram gives an overview of the entire system scope and thus paints a bigger picture of the system.

DFDs

A DFD shows the system broken down into processes and sub-processes – the data flowing in and from the processes, where the data is stored, as well as the origin (external entities/sources/sinks) of the input and output data flows.

Systems analysts use DFDs to analyse and understand the system being investigated. Analysts can also use DFDs to explain the system and their understanding of it to the customer and users for feedback and validation, as the people who work in the domain usually understand the processes very well.

Balancing DFDs

When the systems analyst decomposes a DFD, the inputs to and outputs from a process should be conserved at the next level of decomposition, as this would ensure the balancing of a DFD. Balancing ensures that DFDs are consistent with other diagrams in the information systems DFD set and that all data flows, data definitions and process descriptions are included. All the sources/sinks (external entities) and data flows on the context diagram should be included in the lower level DFDs drawn. If a data flow is changed, for example – because of an error on the lower level DFD, ensure that the corresponding context diagram is also updated. Consistency is very important as part of balancing all the DFDs drawn on different levels of detail.

DFD mistakes: Black hole, miracle and grey hole

The following three mistakes are commonly made when constructing DFDs, and should be avoided:

Black hole: A process has input flows but no output flows. The example in Figure 3 shows that process 1.0 does not have any outputs.

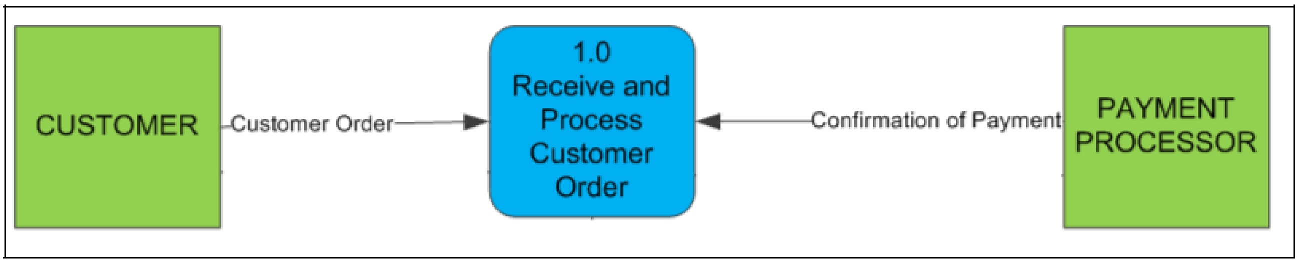

Figure 3 – Black hole example

Source: adapted from Valacich *et al.* extra notes (2015: Chapter 6:6-15)

Miracle: A process has output flows but no input flows. Figure 4 shows that process 1.0 does not have any inputs.

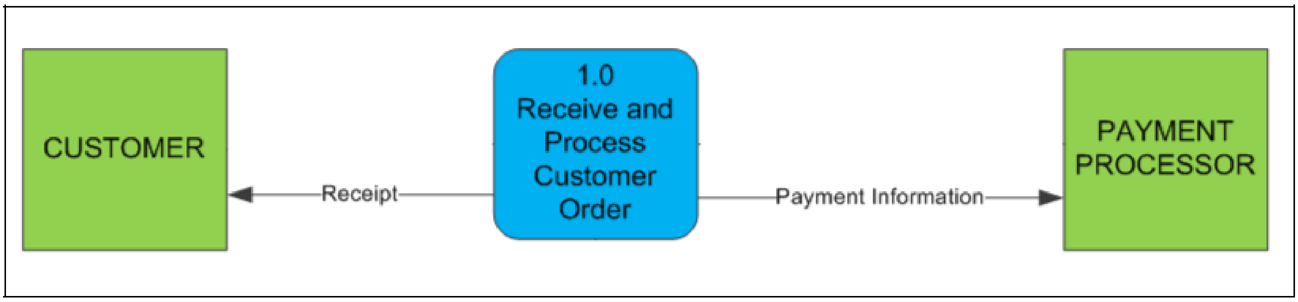

Figure 4 – Miracle example

Source: adapted from Valacich *et al.* extra notes (2015: Chapter 6:6-15)

Grey hole: A process has output which is greater than the sum of its input, or the output does not make sense if one studies the input. The example in Figure 5 shows that the input does not make sense for the output produced.

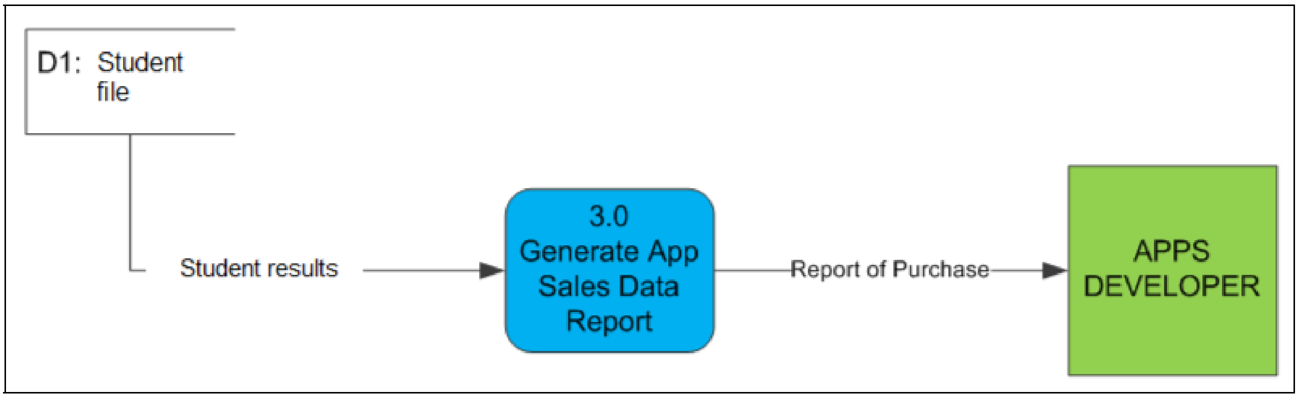

Figure 5 – Grey hole example

Source: adapted from Valacich *et al.* extra notes (2015: Chapter 6:6-15)

Process Modelling Exercises

DFD exercise 1

Group activity

Create a context diagram and Level-0 DFD by studying the case study and completing the steps.

The lecturer will discuss the suggested solution.

Starting with a context diagram, draw as many nested DFDs as you consider necessary to represent all of the details of the patient flow management system described in the following narrative. You must draw at least a context diagram and a Level-0 diagram. In drawing these diagrams, if you discover that the narrative is incomplete, make up reasonable explanations to complete the story. Provide these extra explanations along with the diagrams.

Case Study

Dr Frank's walk-in clinic has decided to go paperless and will use an information system to help move patients through the clinic as efficiently as possible. Patients enter the system at the front desk by providing demographic information to the personnel. If this is the first time the patient has been seen, insurance and basic demographic information is collected from the patient. If they have been seen previously, they are asked to verify the information pulled from the patient registry.

The front desk person then updates the patient registry and ensures that the patient has a chart in the electronic medical records system; if not, a new medical record is started by placing formatted demographics into a blank medical record. The front desk person then enters the medical record ID into the system. Next, a medical technician collects the patient's health history, weight, height, temperature, blood pressure and other medical information, and combines this information with any information from the patient's medical record, summarising the information into a health trend. A doctor then sees the patient, prescribes medication or treatment, where appropriate, based on the medical trend, and sends the patient to checkout. The employee at checkout updates the patient's electronic medical record and provides prescriptions for medications or treatments and a printed record of the health services received.

Step 1: Identify the system name and number it with a zero.

Step 2: Identify all processes and sub-processes and create a decomposition diagram.

Step 3: Identify the external entities (sources/sinks).

Step 4: Identify all data flowing in and data flowing out.

Step 5: Draw a context diagram.

Step 6: Draw a Level-0 DFD.

DFD exercise 2

Group activity

Create a context diagram and Level-0 DFD by studying the case study and completing the steps.

The lecturer will discuss the suggested solution.

Starting with a context diagram, draw as many nested DFDs as you consider necessary to represent all of the details of the patient flow management system described in the following narrative. You must draw at least a context diagram and a Level-0 diagram. In drawing these diagrams, if you discover that the narrative is incomplete, make up reasonable explanations to complete the story. Provide these extra explanations along with the diagrams.

Case Study

Projects, Inc. is an engineering firm with approximately 500 engineers who provide mechanical engineering assistance to organisations, which requires managing many documents. Projects, Inc. is known for its strong emphasis on change management and quality assurance procedures. The customer provides detailed information when requesting a document through a Web portal.

An engineer is assigned to write the first draft of the requested document. Upon completion, two peer engineers review the document to ensure that it is correct and meets all requirements. These reviewers may require changes or may approve the document as is. The document is updated until the reviewers are satisfied with its quality.

It is then sent to the customer for approval. The customer can require changes or accept the document. When the customer requires changes, an engineer is assigned to make the changes to the document. When those changes are made, two other engineers must review them. When those reviewers are satisfied with the changes, the document is sent back to the customer. This may happen through several iterations until the customer is satisfied with the document.

Step 1: Identify the system name and number it with a zero.

Step 2: Identify all processes and sub-processes and create a decomposition diagram.

Step 3: Identify the external entities (sources/sinks).

Step 4: Identify all data flowing in and data flowing out.

Step 5: Draw a context diagram.

Step 6: Draw a Level-0 DFD.

Other exercises

The lecturer will distribute more exercises and discuss the suggested solutions in class.

Appendix B

This appendix should be referenced when studying the content and the assessment criteria for the week(s) listed below.

Week	9 - 12
Learning Outcome	LO3
Assessment Criteria	3.1, 3.2, 3.3, 3.4

Data modelling

Valacich, J.S.; George, J.F. & Hoffer, J. 2015. *Essentials of systems analysis and design, global edition.* 6th edition. Pearson Education. Chapter 7.

Introduction

Conceptual data modelling focuses on representing an organisation's data, and the aim is to highlight the rules related to the meaning and interrelationships among an organisation's data. The main aim of conceptual data modelling is to create an Entity-Relationship Diagram (E-R diagram or ERD).

The following are key data-modelling terms which will be discussed in the sections that follow:

- Conceptual data model ERD (or E-R diagram) Entity type
- Entity instance Attribute
- Candidate key
- Multivalued attributes Relationship
- Degree
- Cardinality
- Associative entity

Entities

Entity types could be grouped into strong, weak and associative entities. Strong and weak entity types will be explained with the following examples:

Example

A STUDENT entity is seen as a strong entity type, because it will exist independently of entities such as COURSE or DEGREE.

An ORDER entity could be seen as a weak entity type, because without a CUSTOMER or a PRODUCT, an order will not exist.

An associative entity type links the instances of one or more entity types and has attributes that are specific to the relationship between these entity instances. Thus, new attributes could be formed for the associative entity. Associative entities are created when two entities have a many-to-many relationship which needs to be resolved. This will be discussed further under the relationship section that follows.

The following example distinguishes between the terms “entity”, “entity type” and “entity instance”:

Example

Entity: STUDENT
Entity type: Strong entity
Entity instance 1: “Jack Wilson”
Entity instance 2: “Neo Mokoena”

For the purpose of systems analysis, entities are objects of importance in organisations. There are the same entities which are found in many organisations, like EMPLOYEE, but each organisation will have its own unique entities. For example, a higher-education institution will find entities like “student” and “course” important. The following are examples of entities:

Student Course Customer Order
Product Stock

Attributes

The following are some examples of attributes (take note of the use of capital letters in the words used after the first word listed for the attribute):

studentNumber employeeName productColour

departmentName productWeight
bookAuthor
studentSurname

Example

A higher-education institution might be interested in only some characteristics of students. These might include a student's first name, last name, date of birth, contact number, and so on. The institution is not interested in all characteristics associated with the student entity – only in the attributes that are relevant to the type of information they need to maintain in the information system for the student entity. The institution might want to store attributes like studentNumber (PK), studentFirstName, studentLastName, studentContactNumber, etc.

It is important to name attributes in such a way that it is easy to interpret what the attribute means. Attribute names should not be too long, but should clearly describe the characteristic of the entity.

Example

If the analyst identifies a "Date" attribute for an ORDER entity, it is necessary to be more specific as to what date this attribute refers to. The word "Date" can be linked to many attributes, like an order date, a submission date, a date of birth, a resignation date, an appointment date, a salary payment date, etc. For this reason, it is very important to clearly and accurately specify attributes, for example in this case, "orderDate".

It is important to identify an attribute or set of attributes for an entity which uniquely identifies each entity instance of an entity type. This is why organisations create membership numbers or customer identification numbers, or why higher-education institutions use student numbers, as no two students will have the same student number.

This unique identifier is unique for each entity instance and is referred to as a **PK** (**Primary Key)** or a **candidate key**. It is important to identify a candidate key that will not change over its lifetime. Choosing "age" as a candidate key is not a good choice, as a person's age will change over time. If organisations struggle to identify candidate keys from the attributes identified, they create a new attribute as a PK, for example – a customer membership number captured at a fitness club. Another important key is an **FK** (**Foreign Key)**. A FK is normally the PK from the main entity that is added to another entity. The FK is explained with the following example.

Example

If a customer places an order, it is important to know which order the customer is linked to. Thus, by displaying the customer number on the order slip, the customer will be linked to that order. The customerNumber (PK) appearing on the order slip is then referred to as an FK (Foreign Key) as part of the ORDER entity.

Organisations have to pay for database storage, and the more attributes that are created for data storage, the costlier the system will be. The aim is to not take up unnecessary space in the database. For this reason, it is important to not include attributes which can be automatically calculated by the system. If the analyst needs to include an “Age” attribute for an EMPLOYEE entity, the system can be programmed to automatically calculate an employee’s age by using the “dateOfBirth” attribute and the current date.

An attribute that can have more than one value for an entity instance is referred to as a **multivalued attribute**. If a higher-education institution wants to store their employees’ qualifications, an attribute called “lecturerDegree” will be created. It is possible for a LECTURER entity to have obtained more than one degree; thus, “lecturerDegree” will then be called a multivalued attribute, as it may contain more than one value for each entity instance. A multivalued attribute is shown in these brackets: {}, for example {lecturerDegree}.

ERD relationships: degree, cardinality and naming

ERD relationship

Relationships indicate the significant real-world associations between various entities. In other words, relationships show how one entity is associated with another. Relationships between entities are usually seen when evaluating an organisation’s business rules. An example of a business rule in a library could be that a member is not allowed to borrow a book if outstanding fees are not settled.

Relationships always flow in two directions and are, therefore, bi-directional. The following explains this concept:

An employee must work for a department.

A department may consist of one or more employees.

In such a case, there is a relationship between the EMPLOYEE and DEPARTMENT entities stating that an employee must work for a certain department and that a department could consist of one or more employees.

Relationship degree, cardinality and naming

A relationship degree shows the number of entity types that participate in the relationship. There are three common relationships used in data modelling, as shown in Figure 6: unary, binary and ternary.

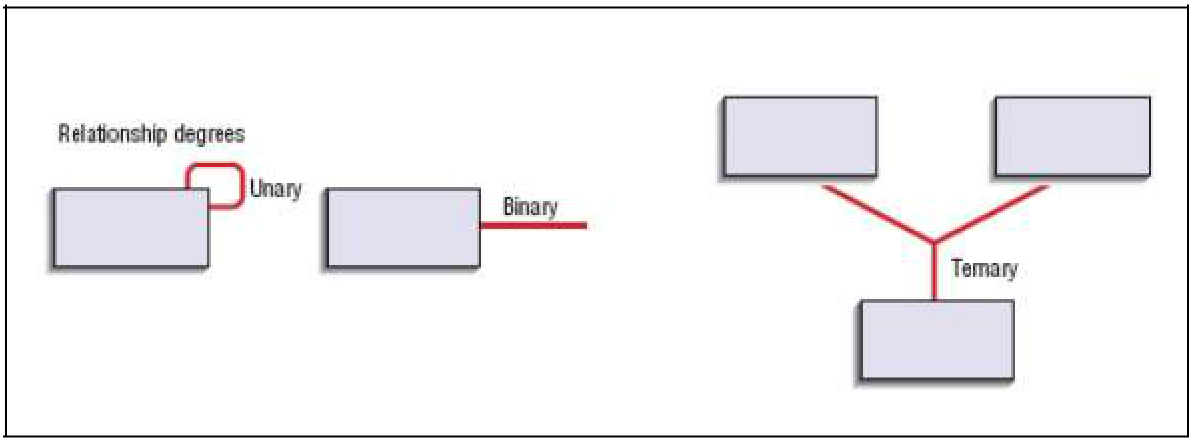

Figure 6 – Three common relationship degrees

Source: Valacich *et al.* (2015:229)

Example

Unary: An EMPLOYEE manages an EMPLOYEE (because a manager is also an employee).
Binary: An EMPLOYEE works for a DEPARTMENT.
Ternary: A CUSTOMER places an ORDER with a SALESCLERK.

Cardinality refers to the amount of instances of one entity that can be associated with another entity. A relationship can be identified between a CUSTOMER and an ORDER, as shown in Figure 7.

If the minimum cardinality of ORDER is one, then ORDER is a mandatory participant in the relationship. However, if the minimum cardinality for CUSTOMER can be zero, then CUSTOMER is an optional participant in the relationship. A customer cannot be forced to place an order (“may”).

The degree is thus optional or zero for the customer. A customer may place zero or many orders. An order must be placed by one customer for the order to exist. By studying the notation used to show “many”, one can recognise why this is known as the “Crow’s Foot Notation”.

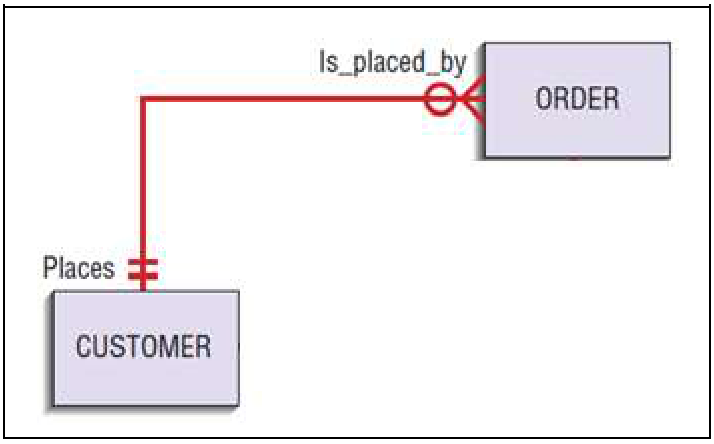

Figure 7 –Example of cardinality

Source: adapted from Valacich *et al.* (2015:244)

Cardinality can be included in more detail as part of the relationship if such information is provided by the organisation. For example, if the organisation has a business rule in place that only a maximum of ten orders per customer is allowed, the cardinality will be noted as shown as [0;10] in Figure 8:

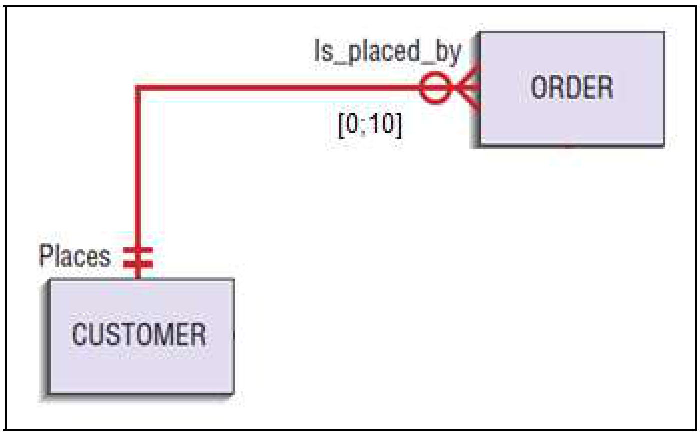

Figure 8 – Example of cardinality (minimum and maximum specified)

Source: Visagie (2018) & adapted from Valacich *et al.* (2015:244)

It helps to understand the ERD when relationships are named. The name of a relationship should not be too long and should capture the essence of the relationship as shown in Figure 15: A customer “places” an order and an order “is placed by” a customer. Some verbs fit the situation better than others, for example – it will not make sense to state “A customer ‘asks’ for an order”.

The known term used with an order is to “place” an order. For this reason, it helps to read the relationship names aloud in both directions to ensure that it makes sense within the context being analysed.

Constructing an ERD

The notation used to construct an ERD involves three constructs: entities, attributes and relationships. These constructs were discussed in the previous sections. Read the following case study, and take note how the process works for constructing an ERD (Valacich *et al.* extra notes (Chapter 7:7-18), 2015).

Case study A performance venue hosts many concert series a year. Performers have a name and perform several times in a concert series (each constituting a performance with a different date). Concert series have one or more performers and have a name and a specified seating arrangement. A concert series is held in one (and only one) of several concert halls, each of which has a room number. Represent this situation of concerts and performers with an ERD.

Step 1: Identify the entities

Performer

Concert_Series

Concert_Hall

Step 2: Identify the attributes for each entity

Although it is possible to identify more attributes for each entity, in this example only some attributes are identified.

Performer

- Performer_ID o Name
- Concert_Series o Series_ID
- Name

Concert_Hall

- Concert_Hall_ID o Room#

Step 3: Identify the relationships, including the degrees and cardinality

A performer may be involved in one or many concert series.

A concert series may have one or many performers associated with it. A concert hall may host zero or many concert series.

A concert series is associated with one concert hall.

Take note that there is a many-to-many relationship between "Performer" and "Concert_Series" which needs to be resolved with an associative entity called "Performance". This new entity could have an attribute such as "Date" of performance. The two primary keys of the linked entities will also be included for the "Performance" entity.

Step 4: Name the relationships

It is important to name the relationships, although relationship naming is not included in this activity. It helps to read the relationships aloud in both directions to confirm whether the relationship names make sense in the business domain being studied.

Step 5: Draw the ERD

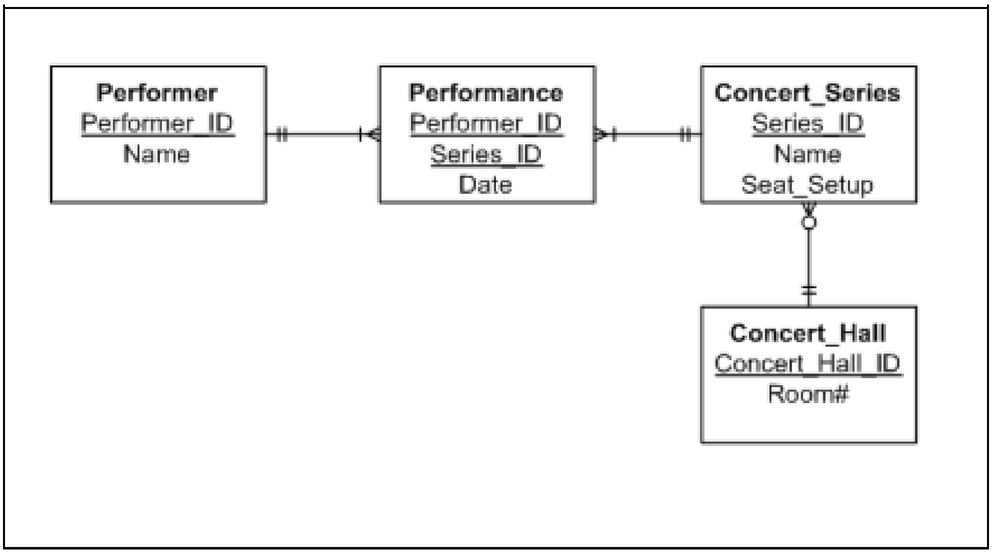

Figure 9 – Example ERD

Source: notes from Valacich *et al.* (2015: Chapter 7:7-18)

Data modelling exercises

ERD exercise 1

Group activity

Create an ERD by studying the case study and completing the steps. The lecturer will discuss the suggested solution.

Case study
Draw an ERD for a patient appointment system. Consider the entities patient, patient appointment, doctor, prescription and pharmacy. Identify and depict the cardinality of each relationship. Also identify the attributes and candidate key (primary key) of each entity. Step 1: Identify the entities (already done in case study). Step 2: Identify the attributes for each entity. Step 3: Identify the relationships, including the degrees and cardinality. Step 4: Name the relationships. Step 5: Draw the ERD.

ERD exercise 2

Group activity

Create an ERD by studying the case study and completing the steps. The lecturer will discuss the suggested solution.

Case study
A restaurant chain has several store locations in a city (with a name and ZIP code stored for each), and each is managed by one manager. Managers manage only one store. Each restaurant location has its own unique set of menus. Most have more than one menu (e.g. lunch and dinner menus). Each menu has many menu items, and items can appear on multiple menus and with different prices on different menus. Represent this situation of restaurants with an E-R diagram.

Step 1: Identify the entities. Step 2: Identify the attributes for each entity. Step 3: Identify the relationships, including the degrees and cardinality. Step 4: Name the relationships. Step 5: Draw the ERD.

Other exercises

The lecturer will distribute more exercises and discuss the suggested solutions in class.

Concluding remarks

This addendum highlighted logical data modelling by explaining ERDs. Data modelling terms used when constructing ERDs were explained. Case studies were applied to practise the construction of ERDs.