MCAST

Malta College of Arts, Science & Technology

MQF Level 6

IT6-02-15

MCAST Bachelor of Science (Honours) in Software Development

Course Specification

Course Description

This course is intended for those students who are keen on software development with an inclination towards backend systems and solving complex problems. In this course you will learn how to design software applications based on formal design patterns; design and implement the underlying data infrastructure; model and solve complex real problems whilst following formal methodologies. This course covers Secure Development, Enterprise Software Development, Server Side and Client Side scripting, Discrete Mathematics and other similar modules. This course is intended for individuals who have a keen interest in problem solving, have an attention to detail and strive to program efficiently.

Programme Learning Outcomes

At the end of the programme the learner will be able to:

1. Design, implement and document the underlying data infrastructure to support software applications

2. Design, implement and document the back-end of enterprise applications for a given requirement

3. Revise a software design/implementation to optimise its use of resources

4. Test and secure the software application and its content to conform to industry standards

Entry Requirements

MCAST Advanced Diploma in IT
(recommended stream: “Software Development”)
or
MCAST Advanced Diploma in Electronics (Computer Engineering)
or
2 A-Level passes and 2 I-Level passes
Compulsory A-Level: Computing
Compulsory A-Level or I-Level: one subject from Pure Mathematics, Applied Mathematics and Physics

Current Approved Programme Structure

Unit Code	Unit Title	ECTS
ITSFT-506-1605	Object Oriented Programming	6
ITSFT-506-1606	Software Engineering	6
ITMTH-506-1601	Discrete Maths	6
ITDBS-506-1603	Database Programming 1	6
ITSFT-506-1607	Mobile Applications Development	6
ITSFT-506-1608	Data Structures & Algorithms	6
ITSFT-506-1609	Low Level Programming	6
ITDBS-506-1604	Database Programming II	6
ITSFT-506-1610	Interactive Mobile Development	6
ITSFT-506-1611	Client Side Scripting	6
ITSFT-506-1612	Server Side Scripting	6
ITSFT-506-1613	Securing Applications	6
ITSFT-506-1614	Low Level Programming II	6
ITSFT-506-1615	Test Driven Development	6
ITSFT-506-1616	Enterprise Programming	6
ITSFT-506-1617	Distributed Programming	6
ITBSI-506-1601	Business Intelligence & Reporting	6
ITPRJ-506-1605	Project	6
ITIMG-606-1601	Image Processing and Computer Vision	6
ITSFT-606-1618	Applied Computational Intelligence	6
ITSFT-606-1619	Data Structures & Algorithms II	6
ITSTA-606-1601	Statistics for Computer Science	6
ITSFT-606-1620	Programming for the Cloud	6
ITSFT-606-1604	Content Management Systems	6
ITPRJ-606-1606	IT Project	6
CDKSK-506-1605	English	6
CDKSK-606-1615	Entrepreneurship	6
CDKSK-506-1526	Critical Thinking	6
ITDIS-612-1601	Dissertation	12
Total ECTS		**180**

Unit: ITSFT-506-1605 Object Oriented Programming

Unit level (MQF): 5

Credits: 6

Unit description

This unit assumes students are already familiar with OOP principles and concepts, and is as such designed for learners who have completed the level 4 OOP module or who have equivalent experience.

This unit covers advanced object-oriented concepts, allowing the students to design object-oriented solutions from start to finish. Students will learn how to create applications using a markup-driven GUI framework and how to link created applications to a database using object-relational mapping (ORM). It is up to the lecturer to choose the specific technologies to be used in this unit. For example, one can teach markup-driven GUI with WPF/XAML, ANT, GTK or QT, as well as any other markup-driven GUI language. ORM can be delivered using LINQ, Hibernate, TopLink or any other object-relational language.

The unit starts with a revision of object-oriented concepts to ensure that all learners are on track with their knowledge. This revision includes access modifiers, properties, constructors, inheritance, overriding virtual methods, abstract classes, interfaces and polymorphism.

Following this, advanced object-oriented concepts will be covered including aggregation vs inheritance, dependency injection, open-closed principle, single responsibility principle, Liskov substitution principle, interface segregation principle and dependency inversion principle.

Learners will then develop applications using a markup-driven GUI framework. The markup language (such as XML/XAML or similar markup) will be introduced, as well as layout and display control. Learners will learn how to create and style controls as well as how to use WPF/ANT/GTK/QT (or other) user controls. The GUIs created will then be made functional with a focus on event handling and data binding.

Finally, the unit covers persistence. In this part of the unit, learners will integrate with and store data in a database using an object-relational language such as LINQ, Hibernate, or any other ORM language. Students will appreciate the differences between a data-centric approach and an object-centric approach when designing the model layer of an application.

Learning Outcomes

On completion of this unit the student will be able to

1. Design and build object-oriented solutions using both fundamental and advanced object-oriented concepts to be able to address business requirements.
2. Implement persistence in created applications to allow created applications to store and read data from multi-user database management systems.
3. Design applications that leverage object-oriented design principles to ensure best practices are adhered to.

Unit: ITSFT-506-1606 Software Engineering

Unit level (MQF): 5

Credits: 6

Unit description

This unit has been designed to introduce learners to the main concepts behind the science of software engineering. Throughout the course of their studies, students will acquire the skills to understand and support the complete life cycle of a software system – from inception, requirements elicitation and design, through the various stages until release and maintenance. Students will gain an understanding of different software development techniques and will learn how to critically select which technique is best suited to the development of different systems.

The unit places focus on some of the more recent software development processes, making particular emphasis on the Agile philosophy of software development. Students will understand the agile process and its constituent components, its applicability to modern software development and the various actors involved in the process together with their roles and responsibilities. Another core component of this module will be that of introducing students to the Unified Modelling Language, UML, as a tool to facilitate and speed up the software development process. The various constructs of this modelling language will be covered, together with explanations of how they can be utilised to specify and document the software and business processes.

This unit will also present students with a range of advanced software engineering concepts and approaches which will give them the skills required to be able to support new and evolving developments. Students will be introduced to a number of different software architectures and design approaches and they will be encouraged to analyse which setups are most adequate as solutions for diverse scenarios

Learning Outcomes

On completion of this unit the student will be able to

1. Plan and tackle a small software design project as part of a team using an Agile approach.
2. Perform a requirements acquisition exercise in order to identify the main functional and non-functional requirements of a proposed software system.
3. Identify and construct the most applicable UML modelling diagrams to use in particular phases in a software system's development process to achieve a specified goal.
4. Design a solution to a problem by proposing the most suitable architecture and utilising known design patterns.

Unit: ITMTH-506-1601 Discrete Maths

Unit level (MQF): 5

Credits: 6

Unit description

This unit is designed to introduce students to the mathematical foundations of computer science. Writing a computer program goes beyond knowledge of the programming language utilised. Most non-trivial software programs involve the use of complex data structures and mathematical algorithms to function. All of these cannot be created without an understanding of the underlying mathematical concepts. In this unit, focus is placed on encouraging students to think logically and mathematically and on giving them the tools necessary to apply these concepts in their future work as computer scientists and software developers.

Learners will be introduced to logic as the language with which reasoning can be explained. Both propositional and predicate logic will be covered, with students learning how to interpret logical statements and, in turn, how to utilise formal proof systems to validate given propositions. Discrete structures are presented through coverage of set theory, relations and the classification of different types of relations. The numbering system we utilize on a day to day basis will also be placed under scrutiny with learners understanding the foundations behind enumeration and the standard numeric operators applied on natural numbers. Proof techniques such as contraposition, induction, reduction and contradiction will be introduced throughout the course of this unit giving students the necessary tools to handle the problems set.

Learning Outcomes

On completion of this unit the student will be able to

1. Use formal reasoning and notation to explain the correctness of mathematical arguments.
2. Discuss the basic concepts of set and graph theory, relations and functions and reason mathematically about these discrete structures.
3. Demonstrate an understanding of number theory and an ability to produce rigorous proofs centred on the subject
4. Apply proofs using induction techniques to solve problems set

Unit: ITDBS-506-1603 Database Programming 1

Unit level (MQF): 5

Credits: 6

Unit description

This unit provides the basis of advanced database theory and design principles that shall be used in future modules. Following this module, a student should be confident in concepts about relational theory and database design. The theory presented is independent of any specific database management system, however the database design conforms to agreed-upon notation easily adopted to other relational database management systems.

The unit starts with the data manipulation via data manipulation language. In this part of the unit, the learner will learn how to build databases in a relational database management system by creating tables and choosing indices. Attention is given to the enforcement of data integrity rules and their associated delete and update reperussions via foreign keys.

Following this, data query via data query language is discussed. Basic select statements are covered including selecting from multiple tables and using conditions (such as WHERE). This is then extended to include predicates and combining predicates as well as filtering character, date and time data. Finally, data is ordered and paged using appropriate techniques.

Advanced concepts then follow, including performing different types of joins based on the database content as well as using set operations. As part of this topic, different types of joins will also be discussed. At this point, data insertion, updates and deletion will be discussed using the appropriate SQL keywords.

Grouping (via GROUP BY) and windowing are also discussed in this topic, allowing for both single and multiple grouping as well as pivoting. For windowing, aggregation, ranking and offsets will be discussed.

The unit concludes with a discussion and practical use of views, as well as how inline functions can be incorporated into SQL to extract calculated fields.

Learning Outcomes

On completion of this unit the student will be able to

1. Create the structure of a relational database.
2. Retrieve data from a database.
3. Manipulate the data in a database.
4. Prepare advanced reports from data within a database

Unit: ITSFT-506-1607 Mobile Applications Development

Unit level (MQF): 5

Credits: 6

Unit description

Mobile Phones have evolved significantly in technology and application within the short period of their existence. From the original purpose of providing voice calls for travelling office executives these devices have now evolved into handheld, multi-media, smart devices boasting an array of sensors, multi-core processing power and a market penetration across young and old, personal and professional use. A significant factor behind the success of smart phone devices is the ability for the user to personalise with well designed, productive mobile applications. This unit is designed to introduce the learner to key aspects in mobile application development from Operating System and development environment through to application state and data storage.

This is a skills based unit and will allow learners to demonstrate they have the necessary skills to be able to design, program and test a mobile application. The unit will guide learners through the basics of Operating Systems (OS), development environments, device sensors and data storage. Learners will use software programming concepts and as a consequence should able to operate effectively at more than a basic level of competence before commencing this unit.

Outcome 1 introduces the mobile Operating System of choice as well as the development tools available. The learner will familiarise themselves with the development environment and test device/simulator.

Outcome 2 concentrates on implementing the application (Graphical) User Interface (UI). Learners will explore the fundamentals behind building a GUI for the particular OS. They will investigate the various libraries and classes as well as designing and using OS menus.

Outcome 3 focuses on working with differing user states. Learners will understand how to handle application start up, background and resuming as well as considering state changes associated with screen rotation or notifications.

Outcome 4 looks at various options for storing/loading context data internally or externally (through the use of memory cards).

Learning Outcomes

On completion of this unit the student will be able to

1. Examine the fundamental components of mobile application development (OS architecture, development environment).
2. Develop a practical mobile application user interface.
3. Manipulate a mobile application depending upon state change.
4. Demonstrate the ability to store and load data in a mobile application.

Unit: ITSFT-506-1608 Data Structures & Algorithms

Unit level (MQF): 5

Credits: 6

Unit description

The basis of solving a problem requires an understanding of how to break it down into a series of much more manageable small parts. In order to do this, students need to be able to assess the complexity of a problem. Once the algorithm has been broken down into smaller sections, a student should start to write logical instructions using pseudocode. Each instruction in turn will manipulate data, which may be for instance array structures, which are similar to vectors in mathematics or abstract data types such as pointers, these are similar to how machine code uses memory addresses to access data.

In this unit students will learn about writing algorithms for common problems such as Queues and thereby choosing the most appropriate data structure.

Students need to implement a series of algorithms which are well known in Computer Science. For a given algorithm a student will need to analyse the complexity and make a decision on how this may affect the efficiency of the algorithm in terms of run time. Although computers now have very powerful processors, students still need to estimate the time it would take for their algorithm to process a given amount of data. In particular as the amount of data becomes larger the amount of time it takes to process the data can grow exponentially.

Students will learn and appreciate that algorithms can be translated into programming code. This in turn will give them an insight into solving problems on paper before typing their code into a text editor for a given programming language. This experience will allow them to see how their programs run as originally indented in the specification. Also it has been written in such a way that it runs efficiently, avoiding complexity in their solution as well as making best use of the processing power of their computer system.

Data structures such as pointers, which allow a programmer to use memory addresses to access data, give the student a much more flexible method to manipulate data. For each algorithm a student needs to select the most appropriate data structure, in

order to produce a solution which will carry out the required tasks as set out in the specification for a computer program

Learning Outcomes

On completion of this unit the student will be able to

1. Construct programs using Abstract Data Types and Structured Data types.
2. Design efficient algorithms for commonly encountered problems using existing examples.
3. Make use of algorithm analysis to determine the efficiency of an algorithm.
4. Compare algorithms in terms of their correctness, proof and intractability.

Unit: ITSFT-506-1609 Low Level Programming

Unit level (MQF): 5

Credits: 6

Unit description

A comprehensive understanding of Low Level Programming allows engineers to implement efficient and resilient software within computer based solutions as well as providing a strong grounding in the various microprocessor/microcontroller architectures available. The 'C' low level programming language is a widely used and versatile general purpose computer programming language found in many engineering applications such as Embedded Computing. It has many derivatives within areas such as GUI development, gaming and control.

This is a skills based unit and will allow learners to demonstrate they have the necessary skills to be able to design, program and test 'C' based low level programs for various applications. The unit will guide learners through the principles of constructing programs using libraries, header files and pre-processor directives, working with data types and data management as well as dealing with memory management, interrupts, Networks and Files. This unit is a continuation from Programming Concepts 1 (Level 4) and it is expected that Learners will be familiar with this unit's content.

Outcome 1 concentrates on the typical requirements for embedded low level programs. The learner will familiarise themselves with the process of compiling and linking 'C' programs including libraries as well as using header files and pre-processor directives. The learner will also use an Integrated Development Environment tool set to test and debug code.

Outcome 2 focuses on dealing with the application structure and code logic of a 'C' program. Learners will use a range of data types and manipulate them using common 'C' commands including bitwise logic and arithmetic operations as well as string manipulation.

Outcome 3 emphasises flow control, arrays and pointers. Learners will build an understanding of various flow control techniques including, if-else, switch-case, while and for loops as well as investigating types of arrays and pointer, their construction and manipulation.

Outcome 4 highlights the implementation of functions or subroutines. Learners will also demonstrate their mastery of the Unit by developing a 'C' language program implementing all of the features they have studied in order to solve a specified engineering problem.

Learning Outcomes

On completion of this unit the student will be able to

1. Construct a viable Low Level Language development environment ('C').
2. Demonstrate the ability to manipulate various data types using arithmetic means.
3. Produce 'C' programs with the ability to change flow and search lists.
4. Construct 'C' programs using subroutine structures (functions).

Unit: ITDBS-506-1604 Database Programming II

Unit level (MQF): 5

Credits: 6

Unit description

This unit is a continuation of Database Programming I and furthers the student's knowledge in advance database programming concepts. The unit is based on four main concepts commonly needed in large scale database systems: advanced database objects such as triggers, stored procedures, views and sequences; query optimization; error handling and concurrency support; integration of advanced techniques such as XML support and integration with external applications. This unit empowers the student with the necessary information to design a scalable database and integrate it with external applications, which is a very common and realistic industry need.

It is important to note that this unit is a continuation of Database Programming I and so the latter is a pre-requisite. This unit delves into advanced techniques commonly faced by backend software developers and systems architects.

Business Intelligence topics have not been included in this unit since these are considered as more advanced topics which merit a unit of their own. It is also strongly recommended that such a unit would follow this unit, since this unit offers a solid basis for the advanced BI topics.

Learning Outcomes

On completion of this unit the student will be able to

1. Improve a database design using advanced techniques needed for support of big systems.
2. Optimise database and query performance.
3. Design a resilient database system via error handling and concurrency support.
4. Integrate support for external data and applications with an existing database.

Unit: ITSFT-506-1610 Interactive Mobile Development

Unit level (MQF): 5

Credits: 6

Unit description

Mobile Phones have evolved significantly in technology and application within the short period of their existence. From the original purpose of providing voice calls on the move these devices have now evolved into handheld, multi-media, smart devices boasting multi-core processing power, a wide range of sensors, and a market penetration across young and old, personal and professional use. A significant factor behind the success of smart phone devices is the ability for developers to innovate upon the device hardware and Operating System in order to provide beneficial and lucrative mobile applications (Apps). This unit is designed to afford the learner with vital skills and experience in mobile application development from the use of OS multi-tasking features and network capabilities through to sensor utilisation.

This is a skills based unit and will allow learners to demonstrate they have the necessary skills to be able to design, program and test a mobile application. The unit will guide learners through the basics of multi-tasking, Internet services, device sensors and telephony features. This unit is a continuation from Mobile Programming (Level 4) and it is expected that Learners will be familiar with this unit's content.

Outcome 1 introduces the concept of multi-tasking/threading typically found on multi-core processor enabled mobile devices. The learner will familiarise themselves with the software development requirements when working with multi-tasking Operating Systems.

Outcome 2 concentrates on the common libraries provided for delivering World Wide Web content in a mobile device use case. Learners will investigate various development libraries and classes as well as implementing a web service application.

Outcome 3 focuses on working with differing device sensors. Learners will understand how to handle sensor initialisation as well as considering state changes associated and notifications.

Outcome 4 looks at information provided by the mobile devices phone Application Programming Interface. The learner will experience the types of information provided including connection status, access to the address book and use of text messaging.

Learning Outcomes

On completion of this unit the student will be able to

1. Utilise Multi-threading within a mobile Operating System.
2. Develop a mobile application to utilise Web API's.
3. Manipulate a mobile application depending upon on-board sensor readings.
4. Demonstrate the ability to work with a mobile telephony Application Programming Interface.

Unit: ITSFT-506-1611 Client Side Scripting

Unit level (MQF): 5

Credits: 6

Unit description

This unit covers advanced client side scripting. In particular, an emphasis will be placed on using a design pattern that allows the developer to separate the business logic of a web application from the UI design. This unit also covers some aspects of HTML5.

The unit starts with a brief revision of client side scripting, whereby the learner is shown how HTML, CSS and JavaScript can be used together to build the presentation layer of a web application. This includes basic tasks covered in pure JavaScript.

Following this, learners will be introduced to JavaScript libraries, which simplify client-side scripting by offering a functional layer above pure JavaScript. Learners will learn how to bind to DOM elements via the JavaScript library and hence how to manipulate DOM objects. They will also learn how to listen for changes in the DOM and react accordingly with JavaScript functions. Learners will also be introduced to asynchronous JavaScript, and how this can improve the performance of a web application.

Using these libraries and JavaScript, the learners are then shown how to create and consume data, and also how to implement object-oriented concepts in JavaScript.

At this point, the learners are introduced to an MV* architecture. Using this methodology, learners will learn to separate business logic from presentation. The learners will present data using this architecture, including how data can be presented in this model. Learners will also be exposed to HTML5 (and to a lesser degree, CSS3). They will learn about the impetus for a new version of HTML and the new tags and functionality that it brings.
Learners will have practical sessions where they will apply HTML5 elements to web applications and manipulate them via JavaScript libraries.

For this unit, the lecturer is free to use any JavaScript library of their choice. However, importance should be given to the most popular, ‘defacto’ JavaScript libraries. The lecturer is also free to choose a development environment

Learning Outcomes

On completion of this unit the student will be able to

1. Build the presentation and controller layers of a web application using HTML, CSS and JavaScript to meet system requirements.
2. Use JavaScript libraries in web applications to simply client side scripting and build interactive elements for the system being developed.
3. Describe a design pattern and implement it to separate the concerns of business logic and presentation in web applications.
4. Explain the motivation for the development of a new version of HTML5 and use the new elements introduced in web applications.

Unit: ITSFT-506-1612 Server Side Scripting

Unit level (MQF): 5

Credits: 6

Unit description

Server side scripting is a powerful and customizable technology for creating dynamic web pages. It has several benefits over client side scripting. For example it allows controllable environment for executing server side scripts whereas user environment cannot be controlled when using client side scripting. Server side scripts are able to change the HTML output according to the web browsers. Server- side technologies are installed on the web servers to process the scripts and HTML stream is then returned to the client's web browser. This unit does not require the use of any particular server side scripting language and the user can choose depending upon their knowledge and skills.

This is a skills based unit and will allow learners to demonstrate that they have the necessary skills to be able to design, program and test server side scripts. The unit will guide learners through the process of constructing dynamic web pages using relevant server side technology.

Outcome 1 concentrates on the typical requirements for running server side scripts and understanding the difference between static and dynamic web pages. The learner will familiarise themselves with the requirements and installation process. The learner will also understand the process of configuring and setting up the web server to test web pages.

Outcome 2 focuses on dealing with the anatomy of a dynamic web page using server side scripting and how this technology works. Learners will understand different data types supported by their chosen programming language through the use of Server side controls/web controls while creating simple web pages.

Outcome 3 focuses on the use of control structures in the chosen programming language. Learners will understand the use of operators, branching and looping structures. They will also understand the practical aspects of the event driven programming. They will also understand the use of functions by passing parameters and returning values. They will learn how to identify problems when loading the web page and testing its functionality.

Outcome 4 emphasises on theoretical and practical aspects related to the use of databases in dynamic web pages. They will understand the use of data handling controls to fetch data for web pages. Learners will also be given the opportunity to demonstrate their mastery of the unit by developing a program using their chosen programming language implementing all of the features they have studied in order to design a dynamic data handling web application.

Learning Outcomes

On completion of this unit the student will be able to

1. Construct a viable environment to create dynamic web pages with reference to client-server architecture.
2. Develop asynchronous dynamic web pages using server side scripting language.
3. Demonstrate the ability to implement OOP event driven programming and handling errors.
4. Demonstrate the ability to integrate server side scripting with databases to manage and manipulate data.

Unit: ITSFT-506-1613 Securing Applications

Unit level (MQF): 5

Credits: 6

Unit description

Securing applications is becoming an important concern for most development companies, mainly, due to the fact that in the last decade, many companies are utilizing applications over the network for their business, thus they are more vulnerable to a wider variety of external threats. Examples are Web applications such as an online banking system, an auction site and so on. Whilst the new capabilities of these web applications offers a variety of benefits to the business, if not developed with security in mind, they can cause vulnerabilities in the system that can expose the business to various risks, such as for example a simple configuration error might leave a door open for the hackers to access the business database.

Security should be a design requirement for a project even though there are time and cost constraints to the project. In order to develop secure applications, important security considerations should be adapted throughout the software development life cycle.

The majority of the application attacks are based on common vulnerabilities. OWASP which stands for Open Web Application Security Project publishes a list of most ten most popular programming mistakes whilst developing an application. These vulnerabilities are common by the hackers and are exploited frequently. Through this unit, students will be exposed to possible categories of vulnerabilities. Each category contains a set of threats, such as SQL Injection, Cross Site Scripting and so on.

The majority of these threats can be mitigated by applying some well-known security standards, and throughout this unit, the student will be provided with practical examples on how these vulnerabilities can be avoided.

This unit will provide students with a level 5 theoretical and practical knowledge of application security aspects.

Learning Outcomes

On completion of this unit the student will be able to

1. Discuss the importance of securing an application.
2. Apply security throughout the development life cycle.
3. Identify vulnerabilities and Implement practical countermeasures for the threats.
4. Apply a variety of cryptographic algorithms to encrypt data.

Unit: ITSFT-506-1614 Low Level Programming II

Unit level (MQF): 5

Credits: 6

Unit description

Smart devices embedded with powerful microprocessors running sophisticated low level programs have now become part of the mainstream. These devices are finding their way into more and more sophisticated but non-intrusive use-cases such as machine-to-machine Internet of Things. As such, the modern software engineer requires a broad yet in depth understanding of the requirements for Low Level programming of these systems, associated hardware and networking.

This skills based unit provides learners with a platform to learn and demonstrate the skills required to engineer involved, 'C' based low level programs for a range of relevant use case scenarios. The unit will develop learner's abilities within structure and compilation of 'C' programs, the effective design and use of system memory, provision of real-time functionality with the use of interrupts, culminating with practical investigations into 'C' based networking. This unit is a continuation from Low Level Programming 1 (Level 5) and it is expected that Learners will be familiar with this unit's content.

Outcome 1 concentrates on the typical requirements for embedded low level programs. The learner will familiarise themselves with the process of compiling and linking 'C' programs using multiple source files, cover duplicate inclusion and the use of 'Make' or 'Build' processes. The learner will also use typical pre-processor directives.

Outcome 2 focuses on dealing with the application structure of a 'C' program as well as memory management implementations. Compile and run time memory management will be investigated through the use of Malloc commands, complex pointer operations and linked lists.

Outcome 3 emphasises application features such as concurrency, multi-tasking and File Input/output using 'C'.

Outcome 4 investigates the implementation of network programming using sockets. Learners will also be given the opportunity to develop a 'C' language program implementing all of the features they have studied in order to solve a specified engineering problem.

Learning Outcomes

On completion of this unit the student will be able to

1. Construct a multiple-source ‘C’ program for a specific hardware platform.
2. Develop a memory managed Low Level application.
3. Demonstrate the ability to implement a multi-tasking operating application in ‘C’.
4. Demonstrate the ability to implement a networked application in ‘C’ using Sockets.

Unit: ITSFT-506-1615 Test Driven Development

Unit level (MQF): 5

Credits: 6

Unit description

Computer programs are essentially a product which is supplied to a customer for a given fee. Initially a client would agree on a specification provided by analysts and once approved the client will enter into a legally binding contract to receive a software program which will carry out specific tasks, for example it would lead to an improvement in the day-to-day operations of the client's organisation. Similar to other products which we purchase in the shops, it would be essential that it works correctly and carries out the specific tasks which were promised by the manufacturer. There is same expectation for computer software to meet the similar standards. For this precise reason, it is important to thoroughly test programs using well known techniques in Computer Science. Often the reputation of the persons developing the computer software would be at risk if testing is overlooked or inadequate.

Test Driven Development is a development approach, where a test written before writing the code. This approach will make sure that the source code is tested well, this has the effect of having code which is in modules, it is also flexible so it can be changed more easily and it is extensible so it can be adapted without much further coding. The stages involved in Test Driven Development Cycle include: adding a test, running all tests and checking if the new code fails, writing code, running automated tests, re-factoring code and finally repeating the cycle as in previous steps with a new test. The initial testing process would start with receiving messages from Compliers or Interpreters telling the programmer that there is a syntax error in a given line of code. Secondly, test data would need to be used check whether the program has any logic errors and to see that the program carries out the tasks which were stated in the specification. If the wrong output is being shown, then the programmer needs to find and correct the errors, often by using special tool known as debugger software, this is piece of software is built into the compiler.

Taking the testing process further, the computer program needs to ensure the integrity of the data is maintained by means of a verification process of checking data. Secondly, data validation is the processor of getting the computer software to check data is valid in terms of the context in which it is being used. These tests will ensure the correctness of the data being processed and will involve writing complex code. Nevertheless, if computer programs are to gain the respect of business and industry, then this important aspect should not be overlooked.

This unit is geared towards enabling students to learn about the current well known methods of testing software. These will include White Box and Black Box testing. Also students will learn about the different levels of testing which Include Unit Testing (using white box testing), Integration Testing (using black box testing) and System Testing.

Businesses that produce computer software often have a dedicated team or department who take on the job of testing programs written by developers, once it is fully functioning. Alpha testing is the initial testing phase carried out by a dedicated team of testers, in order to find bugs that were not found originally through previous tests. Alpha testing is also known as acceptance testing, once the software is approved at this stage we move to Beta testing. For example, Beta testing involves the distribution of pre-release Mobile application (app) software to a select group of people so that they can test it in their own homes. The beta version of the software is as close to perfect as the company can make it.

As per a product in the phases of production, in for instance a manufacturing environment, each product has a product life cycle where it is initially designed, developed, tested and sold to customers in shops. In the same way software programs have systems life cycle. There are various models for the system life cycle available to students which can be applied in developing computer software. We need to select the model which best suits the organisation which requires the computer system.

Learning Outcomes

On completion of this unit the student will be able to

1. Compose tests using Test Driven Development methodology.
2. Explain the process involved in the different levels of testing.
3. Use the Systems Life Cycle in relation to developing a computer system.
4. Develop code which will verify and validate data utilised in the program.

Unit: ITSFT-506-1616 Enterprise Programming

Unit level (MQF): 5

Credits: 6

Unit description

This unit will provide students with a level 5 theoretical knowledge on how enterprise applications are designed and built and expose them to create well defined business logic that can meet the clients' requirements.

Students will learn the about the various software architecture styles available and the role that software architecture plays in the development of larger scale software applications. These will be strongly coupled with software patterns and the capabilities that will be offered within the enterprise to adhere to common entities or policies set up by the client. Furthermore students will gain the necessary knowledge on the provision of common interfaces that may be applied by other applications. They will also identify common frameworks to keep a common standard of design and development of software applications.

Through this units students will have a deep insight business logic and acumen to understand and meet with the clients requirements and moving along in grasp a better understanding of the what is involved in the actual configuration management and the scalability needs for such large scale applications. This will also prepare the students to appreciate the specific security requirements needs to safeguard enterprise knowledge and share the common capabilities amongst the various applications that may developed.

This unit will also allow students to appreciate the complexity of cloud services, the notion of cloud services and how it is utilised in such large scale enterprise as well as have a taste at uploading applications. Students will be able practice uploading test applications to a specific cloud services community.

Learning Outcomes

On completion of this unit the student will be able to

1. *Identify the importance of software architecture and its role within enterprise applications*
2. *Describe and illustrate the different software patterns used to design large scale software*
3. *Discuss the capabilities, configuration and management tactics involved in designing and delivering enterprise software*
4. *Outline the use of cloud services and the method of uploading content*

Unit: ITSFT-506-1617 Distributed Programming

Unit level (MQF): 5

Credits: 6

Unit description

The purpose of this unit is to provide the learners an understanding on Service Oriented Architecture (SOA). It will focus on how a set of components can be set to be invoked, and how their descriptions can be published and discovered through the internet. Additionally it will also cover the benefits of developing in such a software design pattern. This unit will also give a hands-on approach and will guide learners to create APIs which enable communication between different applications by utilizing some of the modern architectures such as REST, SOAP and Web Services. Learners will also be able to apply security techniques to enforce security when developing APIs.

Apart from setting up the architecture on the Server, and undergoing security considerations, this Unit will outline how Services modelled using a service oritned approach such as can REST, SOAP and Web Services can be consumed from the Client by making use of pre-structured data such as JSON and XML. Learners will be taught the differences and similarities between JSON and XML together with the technical detail to be able to construct and parse both JSON and XML data.

In this Unit, learners will also be taught how to makes use of widely used APIs to get information such as weather data, social networking data and transactional information. Learners will also be exposed to the concept of cloud storage and how cloud solutions can be used when designing a distributed application.

Learning Outcomes

On completion of this unit the student will be able to

1. Describe the main concepts behind a Service Oriented Architecture.

2. Use modern architectures such as REST, SOAP and Web Services to create APIs which enable the communications between different applications.

3. Explain and demonstrate how data is sent from the Server and consumed from a Client by utilizing standard data exchange formats such as XML and JSON.

4. Use and consume third party APIs and Cloud Storage solutions to retrieve data.

Unit: ITBSI-506-1601 Business Intelligence & Reporting

Unit level (MQF): 5

Credits: 6

Unit description

This unit is designed to provide students with the skills necessary to understand and participate in data warehousing projects and to support the analytical reporting tasks that would successively be carried out on these data stores. Technological advances made in recent years have led to an explosion in the amount of data being collected and stored within organisations. Most businesses now recognize the fact that they can harness the power of big data to increase their competitiveness and/or improve their processes, and this is not limited only to commercial scenarios. The ability to analyse voluminous data sets has been an important step forward for the most disparate of fields spanning government institutions, medicine and health, astronomy and biology and many more. During the course of this unit, students will be introduced to Business Intelligence as a collection of tools and techniques that allow for the extraction of knowledge from large sources of data.

Due to its central role in any BI solution, focus will be placed on the data source itself with the first part of the unit dedicated to the data warehouse. The architecture and logical and physical design of the data warehouse will be covered both theoretically as well as through practical exercises. Students will learn to identify data sources and analyse and transform data sets in an ETL procedure to populate the data warehouse

OLAP analysis will be covered as a second main topic of this unit. The concept of multidimensional data structures and the various operations that can be carried out on them to analyse data from different viewpoints and at varying level of detail will be highlighted in order to give students an understanding of the benefits brought about by BI technology.

Learning Outcomes

On completion of this unit the student will be able to

1. Discuss the ways in which data warehousing coupled with Business Intelligence technologies help to meet the data requirements of strategic decision makers within organisations.
2. Design the schema for a data warehouse to meet a given set of requirements and implement the said model as a relational data warehouse.
3. Outline and discuss the main steps involved in an ETL process and support the theoretical knowledge with the design and implementation of such a process to populate a Data Warehouse.
4. Explain multi-dimensional data structures and discuss and demonstrate the capabilities of OLAP tools through the practical application of dedicated OLAP analysis software on a data warehouse store.

Unit: ITPRJ-506-1605 Project

Unit level (MQF): 5

Credits: 6

Unit description

The rise and evolution of network technology and the internet has made significant changes and improvements to the world around us. Individuals, businesses and countries have benefited greatly through global dissemination of information, growth of electronic commerce, and business innovation through collaboration.

Recent technology trends and the rapid and significant developments in mobility and cloud computing have all played a major role in promoting and enhancing growth in global economies. Whilst this growth and the benefits derived from it have been welcomed they have compounded to a growing shortage of highly skilled employees with the necessary skillsets to work within the industry.

The primary aim of the MCAST Higher Diploma in Network Design and Implementation is to provide participants with a comprehensive range of knowledge and practical skills in designing, building and maintaining computer network infrastructures.

The structure of the award and the inclusion of units particularly in the areas of Virtualisation, cloud computing, network security and intrusion prevention will ensure the award is well placed to address the growing demand for the aforementioned skillsets.

The purpose of this unit is to allow students to demonstrate the practical and technical skills they have acquired whilst studying on the diploma.

The unit should help students to consolidate their learning of network design, implementation and maintenance and should help students considering a career in IT with a particular emphasis on networks gain successful employment or progression to studies at a higher level. Students should be encouraged to choose a project in an area of their own specific interest derived from the course. The chosen project should however allow the students to demonstrate their competence in the field of network design and should reflect the other units they have learnt on the course namely, installation and configuration of computer platforms, management of databases and Content Management Systems, Virtualisation/Cloud computing and Wider Area Networks, and network security.

Learning Outcomes

On completion of this unit the student will be able to

1. Analyse a networking project assignment brief and develop a solution to meet a given specification
2. Plan and organize a technical solution that meets the given specification
3. Implement the chosen solution in phases through to project completion
4. Reflect and evaluate the success of the implemented solution against the given specification

Unit: ITIMG-606-1601 Image Processing and Computer Vision

Unit level (MQF): 6

Credits: 6

Unit description

This unit is designed to give the students a basic yet solid understanding of Image Processing techniques, as well as the application of such techniques in the rapidly-developing area of Computer Vision. The latter has been known to be a difficult problem, due to the complexities involved in the human vision system. Nonetheless, it is finding its way through many sectors such as safety and security, health, and entertainment. Moreover, methods of acquiring such data (digital images and videos) have become even more available and affordable.

The unit first introduces the theory behind Image Processing and Computer Vision, and moves on to the application of fundamental operations in Image Processing, such as quantisation and removal of noise. These fundamental operations are then used in more complex problems, such as finding edges, corners and other interest points or shapes in the image.

Finally, some typical problems of Computer Vision will be examined. These include object detection, recognition and tracking. These problems will serve as challenges as well as motivation to students, as they highlight the relevance and practicality of the material covered. As much as possible the unit will adopt a learn-by-doing approach, in that for most of the topics the underlying theory will be explained followed by a practical example or implementation. This will ensure comprehension and engagement of the student.

Learning Outcomes

On completion of this unit the student will be able to

1. Understand the basic concepts and relevance of Image Processing and Computer Vision
2. Apply fundamental techniques in Image Processing
3. Apply techniques to identify shapes and features in images
4. Demonstrate Object Detection, Recognition and Tracking Techniques

Unit: ITSFT-606-1618 Applied Computational Intelligence

Unit level (MQF): 6

Credits: 6

Unit description

This unit builds on the previous data related modules, namely Database Programming I, Database Programming II, and Business Intelligence & Reporting. The purpose of this unit is to focus on the data analysis aspect of artificial intelligence.

The aim of this unit is to provide the learner with the data analytical skills needed to identify patterns from data reports and to solve real world problems commonly encountered by professionals. This unit will highlight the different kind of solutions and approaches that can be taken in specific scenarios.

The first aspect of this module will be to expose the learner to NP-complete kind of problems and explain when to use exact solution algorithms and when to use heuristic algorithms. Then the student should be able to do exploratory analysis on a dataset to determine an appropriate approach to address the problem at hand.

After an exploratory analysis, the student should be able to clean the data and normalise it in preparation for use within an algorithm. Such algorithm will vary from exact algorithms to heuristic algorithms. Such algorithms will be used to proof or disproof a hypothesis.

Finally, a proper analysis of the gathered data will be done in an academic report following a specific conference template. This report will follow a similar structure as that used for the dissertation which will serve as further practice and preparation for their final year project.

For this unit, a statistical analysis software such as Microsoft Excel and R-Programming will be used. Development of a small prototype can be done using any other programming language. Documentation of the final report will be done using a conference template such as IEEE, ideally with a LaTeX editor.

Learning Outcomes

On completion of this unit the student will be able to

1. Understand different solution algorithm types and when to use each.
2. Analyse a dataset and identify an appropriate approach to solve the problem.
3. Implement several solution algorithms for a given data set and problem.
4. Document findings in a technical report.

Unit: ITSFT-606-1619 Data Structures & Algorithms II

Unit level (MQF): 6

Credits: 6

Unit description

This unit covers data structures and algorithms and builds upon previous data structures and algorithms units. An understanding of basic data structures, O and Θ notation is expected.

The unit covers several topics including: analysis of array growth in array based vectors, graphs and spanning trees and their algorithms, weighted graphs, strings and string searching algorithms, hash tables and other tree algorithms.

The core aim of this unit is to give the learners additional algorithm analysis skills and tools that they can use in their work. In addition to this, the unit aims to introduce the students to a variety of commonly used data structures along with algorithms based on these structures. These can be applied to solve particular problems at the work place or as part of research, such as the research being undertaken in the student's thesis.

At the end of the unit, the student is expected to be able to carry out analysis of new data structures and algorithms, as well as being able to apply new data structures and algorithms to problem domains that the student is working in. The student should be able to independently analyse similar problems and select the ideal structure and/or algorithm to solve a particular problem as well as demonstrate their skills through application.

The lecturer can choose any programming language. A programming language familiar to the students is recommended. Pseudo-code and mathematical notation can also be used where applicable.

Learning Outcomes

On completion of this unit the student will be able to

1. Understand and implement a variety of data structures and algorithms.
2. Apply data structures and algorithms to correctly and efficiently solve real-world problems.
3. Analyse and compare different data structures and algorithms.
4. Synthesise solutions to real-world problems through the application of one or more data structures and algorithms or variants, correctly and efficiently.

Unit: ITSTA-606-1601 Statistics for Computer Science

Unit level (MQF): 6

Credits: 6

Unit description

This unit covers a variety of statistical concepts and applications to algorithms and computer science.

The unit covers several topics including: probability, randomised algorithms, statistical inference, clustering, high dimensional data analysis, data mining, and artificial intelligence and Monte Carlo algorithms.

The core aim of this unit is to give the learners additional tools related to statistics and its application in computer science. These can be applied to solve particular problems at the work place or as part of research, such as the research being undertaken in the student's thesis.

At the end of the unit, the student is expected to have applied these tools and be able to identify the correct tool and apply it independently.

For this unit, the lecturer can choose any programming language however a programming language for which the students are familiar is recommended. The lecturer may also use pseudo-code and mathematical notation to describe concepts and algorithms within the unit. Mathematical notation is essential for presenting statistical concepts and probability.

Learning Outcomes

On completion of this unit the student will be able to

1. Understand how statistics and randomised algorithms are used to make inferences based on the data and are applied to algorithms.
2. Apply and evaluate statistics and randomised algorithms in contexts related to computer science.
3. Use statistical inference to extract information from data.
4. Implement randomised algorithms.

Unit: ITSFT-606-1620 Programming for the Cloud

Unit level (MQF): 6

Credits: 6

Unit description

This subject will add on to the concept of cloud computing. The unit will introduce the motivating factors, benefits, challenges and the service models i.e., software-as-a-service, platform-as-a-service and infrastructure-as-a-service. Moreover, the unit will provide the learner hands-on experience with the commonly found tools within the cloud infrastructure industry such as storage technologies, security measures, highly elastic scalability in delivery of enterprise applications and software as a service (SaaS), caching techniques, and different hosting options.

Practical sessions will be the basis for this unit where a number of technologies will be explored, compared, analysed and then selected to be used within a much larger project to make use of the discussed advantages they will bring about in today's applications.

Moreover, the students taking this unit will be provided with a cloud account were they will undertake the task to configure the necessary settings to make use such mentioned technologies. In the end their work should be deployed on this cloud account.

Learning Outcomes

On completion of this unit the student will be able to

1. Describe the main concepts and benefits behind Cloud Computing.
2. Use Cloud Storage solutions to store structured and large data.
3. Use other services provided by the Cloud services provider.
4. Use Cloud services available to host and consume web applications, APIs or other services.

Unit: ITSFT-606-1604 Content Management Systems

Unit level (MQF): 6

Credits: 6

Unit description

When learning how to program students typically write short programs all by themselves. This contrasts with the reality of most workplaces which require employees to work in teams on larger projects. A content management system supports a number of users working in an environment that promotes synergies for people to achieve their goals. Enterprise-level software is more complex and typically requires a future-proof architecture that keeps maintainability and extendibility manageable.

The unit starts by exploring the anatomy of a Content Management system; a system that has stood the test of time. Furthermore it considers how its design maintains the delicate balance between simplifying extendibility and keeping conflicts and bugs in check.

Next, learners can explore how this architecture can be extended through the creation of themes, plugins, and REST API applications.

Themes allow the programmer to modify the look and feel of the software. While it focuses on design, other development considerations such as responsiveness, and support for different layouts come into play.

Plugins, on the other hand, allow the developer to extend the functionality of the application via self-contained modules that can easily be activated or deactivated by a website administrator.

Finally REST API application development allows one to write and integrate external applications that can have their own independent architecture and that can be written using an entirely different programming language. Although the unit is software intensive, it also covers administration topics such as authentication and authorization, configuration, and module management.

Learning Outcomes

On completion of this unit the student will be able to

1. Identify the features and functionality of an extendible content management system.
2. Activate and customise a versatile and responsive theme that supports multiple layouts
3. Install and set up several plugins including ones that incorporate settings and short-codes.
4. Integrate an external application capable of communicating with the content management system through a REST API.

Unit: ITPRJ-606-1606 IT Project

Unit level (MQF): 6

Credits: 6

Unit description

In this unit, students will be studying the research methods which may be applied to the results and discussions section in their thesis. They will be guided in the development of conclusions and inferring patterns from their results, and will learn how best to interpret and explain data gathered as part of their research. This module is intended to help students working on their final year projects and the practical assessments will be based in part on their project work.

Learning Outcomes

On completion of this unit the student will be able to

1. Define research scope.
2. Identify the correct approach for a given research problem.
3. Explain different research methodologies.
4. Present information and conclusions effectively.