In order to develop high quality software it is imperative that Information Systems Professionals have an understanding of software development as an engineering discipline. This understanding should be focused on both the practical and theoretical aspects of the discipline.

**Aims**
To gain knowledge of:
- The principles of software engineering
- Software process models and process improvement
- Development methods
- Requirements and specification
- Software design
- Software implementation
- Verification and validation
- Software project management
- Software quality

**Objectives**
- An understanding of a wide variety of software engineering principles, process models and development methods
- A knowledge of the different phases in software engineering; requirements, specification, design, implementation, verification and validation
- Project management capability
- A knowledge of how to produce high quality software
- An awareness of the special requirements for specific types of system such as safety critical systems
- An understanding of the practical problems in developing software systems

**Prior Knowledge Expected**
Candidates are expected to be familiar with the material covered in the Certificate syllabuses and Software Engineering 1 Diploma syllabus and should have knowledge of at least one structured programming language and understand the principles of software design and implementation.

**Content**
**Principles of Software Engineering**
Candidates must be aware of the fundamental principles underlying Software Engineering and understand why an ‘engineering discipline’ is required to develop high quality software systems.

**Engineering vs Programming**
Programming in the Large vs Programming in the Small

**High Quality Software Systems vs Software Programs**

**Process Models**
Candidates should be aware that there are many process models that can be used as the basis for a software development life-cycle:
- Overview of Process Models and their importance
- V-model
- V-model with prototyping
- Evolutionary Development
- Incremental Development
- Spiral Model
- Cleanroom Process Model
- DSDM (a process model not a method)
- Comparison of Models
Development Methods and Techniques
Candidates should show a theoretical and practical understanding of the following:
- Structured Methods and Techniques (e.g. SSADM)
- DataFlow Diagrams
- Entity-Relationship Diagrams
- State Transition Diagrams
- Statecharts
- Object Oriented Methods and Techniques (e.g. UML)
- Use Cases
- Class Diagrams
- Behaviour Diagrams (e.g. collaboration diagram)
- Implementation Diagrams (e.g. component diagram)
- Formal Methods
- VDM, Z, B
- CSP, Mascot, Petri-Nets

Software Development Life Cycle
Candidates should show a practical understanding of developing software products through a development life-cycle:
- Requirements
- Requirements Analysis and Capture
- Requirements Engineering, e.g. CORE
- Requirements Tracking
- Functional Requirements
- Non-Functional Requirements Specification
- Refinement of Requirements
- Natural Language Specification
- Diagrammatic
- Formal, Mathematical Specification
- Use of Abstraction
- Rapid Prototyping
- Design
- Refinement of Specifications
- System Architectures

Different Design Approaches
- Design Trade-Offs (e.g. modular vs performance)
- Performance Modelling/ Simulation
- Structured Design vs OO Design
- Implementation
- Encapsulation and Modularisation
- Information Hiding (ADTs/ Interfaces)
- Coupling and Cohesion
- Component Reuse
- Debugging
- Performance Measurement
- Validation and Verification
- Testing; black-box and white box, statistical
- Formal Proof
- Correctness Arguments
- Inspections and Reviews
- Static and Dynamic Analysis Tools
- Using the Compiler
- Maintenance

Project Management
- Risk Management
- Team Management (Personnel and Technical)
- Project Planning (Resource and Technical)
- Education and Training
- Cost Estimation
- Project Scheduling

Software Quality
- Software Quality Assurance
- Configuration Management and Change Control
- Software Tools
- The ‘ilities’
- Standards
- Documentation
- Metrics
Primary Texts
One of many books that give an overview of Software Engineering.
An excellent book on development of software development processes through the management of risk and quality. Provides a good introduction to different software development processes.
A good introduction to object oriented principles with a view to developing software. Also introduces Eiffel.

Other Texts
As indicated, some of the books mentioned here are out of print. They are included because they are still relevant and useful and can be found in many libraries.
A good text on the use of SSADM (out of print).
A good overview of the Unified Modelling Language for those with a knowledge of object-oriented analysis and design.
A classic text on software engineering management that provides a good description of evolutionary software development.
An excellent text for those wishing to develop their knowledge of requirements engineering process.

A great introduction to formal specification using the Vienna Development Method. Look out for the deliberate mistake in the first few pages. Best to ignore the sections on proof the first time you read it.
A good introduction to the specification language Z.

Other Reading
An alternative introduction to OO principles.
A classic text on the economics behind software engineering. (Out of print).