Chapter 11
Software Development Techniques for Constructive Information Systems

Runa Jesmin
Imperial College London, UK & Harefield Hospital, UK

ABSTRACT

This chapter discusses the software engineering lifecycle, history, and software architecture as well as the foundation of Information Engineering and Information Systems. The first part of this chapter discusses the software lifecycle phases and how to make effective use of various technical methods by applying effective technical and other efficient methods at the right time. This chapter also shows the technical similarities between software database design and Information System's database design. In the second part of this chapter, the author introduces the information engineering life cycle and discusses the key phrases for information engineering as well as Information System. In fact, this part is a good dictionary of information and software engineering. This chapter provides guidance for decision-makers in selecting an appropriate Information System strategy that contributes to the achievement of information engineering sustainability targets and leverages competitiveness.

1. INTRODUCTION

The term 'life cycle' means the changes that happen in the life of an animal or plant. In software engineering, this term is usually applied to artificial software systems to mean the changes that happen in the 'life' of a software product. Various identifiable phases between the product's 'birth' and its eventual 'death' are known as lifecycle phases.

Typical software lifecycle phases are (Leszek, 2005):

1. Requirement Analysis
2. System Design
3. Implementation
4. Integration and Development
5. Operation and Maintenance

DOI: 10.4018/978-1-4666-3679-8.ch011

There are a number of life cycle models – each for a different way of organising software development activities.

2. BACKGROUND

In software engineering, life cycle models representing different software engineering methodologies, aim to centralise developers efforts around critical issues such as usability, efficiency, reliability and customer satisfaction.

There are numerous books in software modelling but comparatively Information Engineering fields could improve with more research in this area (Ivar et al,1999). On the contrary, Information Engineering is certainly well developed in providing successful system for overall profit and success (Sidney et al. review of 1974).

There are many Information Engineering dictionaries available in the market (e.g. businessdictionary.com) but this chapter will provide the reader a good simple groundwork of Information Engineering.

3. SOFTWARE DEVELOPMENT

John Tukey has used the term "software" the first time in the American Mathematical Monthly journal in January 1958. However, the purpose of software remains the same today as it was in the 1940s – at the very beginning of computing.

Today software development is often used to enhance a company's enterprise information capabilities by building new functions into an already existing infrastructure, or by gluing together existing systems to carry out new services (Kevin et al. 2005).

In general, the emphasis in software development is increasingly on systems as components in larger systems, components, which interact and combine with each other, perhaps in ways that were not envisaged by their original developers.

The true process of software development is usually based on an irrational process (Pauline, 2004). A rational approach to software development is believed to lead to better software development. In addition, measurements of project progress are made simpler if an initial agreement on how a project is to be carried out exists, based on such guidance.

(Parnas & Clements, 86) Faking the design process allows ease of readability and understanding of the development process by outsiders.

3.1 Software Architecture

Software architecture is a coherent set of abstract patterns guiding the design of each aspect of a larger software system.

A Dutch computer scientist first used the concept 'Software architecture' in 1960s but has increased in popularity since the early 1990s, largely due to activity within Rational Software Corporation and within Microsoft.

Mary Shaw and David Garlan's book on Software Architecture perspective on an emerging discipline in 1996 brought forward the concepts in Software Architecture, such as components, connectors, styles and so on.

The software architect develops concepts and plans for software modularity, module interaction methods, user interface dialog style, interface methods with external systems, innovative design features, and high-level business object operations, logic, and flow.

Software architecture describes the coarse grain components (usually describes the computation) of the system. The connectors between these components describe the communication, which are explicit and pictured in a relatively detailed way. In the implementation phase, the coarse components are refined into "actual components,"