CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

Due to the rapid growth of computer and information technology, increasing attention is given on the development of computer software. Software is embedded in many modern systems, including expensive scientific systems, financial banking systems, industrial applications and home personal computers. According to the IEEE, Software is: Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. The world of software development has evolved rapidly over the past two decades. With an escalating sophistication of computer applications, the demands for complex and large scale software are increasing. To build large software system, the need for systematic approaches to software development and software maintenance becomes increasingly apparent (Fairley 1997).

Many people have attempted to improve software development practices by improving design techniques, developing more expressive notations for capturing a system’s intended functionality and promoting reuse of predeveloped system pieces rather than building from scratch. In many engineering applications, the idea of aggregating standardized components to create a software system has allowed the creation of better systems more easily. This new paradigm is said to be Component Based Software Development (CBSD).
CBSD can be used to reduce software development time by bringing the system to market as early as possible. Furthermore software reuse is recognized as one of the efficient means for reducing the number of bugs, if the development costs have to be kept at a comparatively low level. Software development management and software quality goals are necessary but not sufficient for the needs of today’s marketplace. This research proposes a new direction in selection and assembly of components and also the quality improvement of component based software systems.

1.2 COMPONENT BASED SOFTWARE DEVELOPMENT

The component based software development approach is one of the most promising solutions for the emerging high development cost, low productivity, unmanageable software quality and high risk, and move to new technology in software development today. This approach is based on the idea that software systems can be developed by selecting appropriate Commercial-Off-The-Shelf (COTS) components and then assembling them with well-defined software architecture. This new software development approach differs from the traditional approach in which software systems can only be implemented from scratch.

The term COTS is very generic; it can refer to many different types and levels of software. For the purpose of this work the term COTS means a software product, supplied by a vendor that has specific functionality as part of a system. It is a piece of prebuilt software that is integrated into the system and must be delivered with the system to provide operational functionality or to sustain maintenance effort.

The COTS components can be developed by different developers using different languages and different platforms. This is shown in Figure 1.1,
where COTS components are retrieved from a component repository, and assembled into a target software system.

In general, a component has three main features:

i) A component is an independent and replaceable part of a system that fulfills a clear function

ii) A component works in the context of a well-defined architecture

iii) A component communicates with other components by its interface

![Diagram of Component-based software development]

**Figure 1.1 Component based software development**

### 1.3 SOFTWARE QUALITY

Software Quality activities are conducted throughout the project life cycle to provide objective insight into the maturity and quality of the software processes and associated work products. International Standard Organization
(ISO) formally defines quality as “the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs”. According to the IEEE, Software quality is:

i) The degree to which a system, component, or process meets specified requirements.

ii) The degree to which a system, component, or process meets customer or user needs or expectations.

According to Pressman (2005), Software quality is: Conformance to explicitly stated functional and performance requirements, explicitly documented development standards and implicit characteristics that are expected of all professionally developed software. The relationship between requirements and characteristics in conjunction with quality is shown in Figure 1.2.

![Figure 1.2](image)

**Figure 1.2  Relationship between requirements and characteristics in conjunction with software quality**

Software developer must react quickly and aggressively to meet ever-changing market demands. Maintaining software quality hinders fast-paced software development, as many testing cycles are necessary to ensure quality products. Software testing is a process used to identify the correctness, completeness and quality of developed computer software. Many activities are required in software quality maintenance and improvements. So, a
systematic planned set of actions are necessary for the software development and maintenance process. This leads to software system confirming to established functional and technical requirements along with the managerial requirements to keep the schedule within the budgetary confines. Component based software development is being proposed as a means of improving software quality and reducing development costs.

1.4 BACKGROUND OF THE RESEARCH

Normally software components are purchased or developed and an application system is built by assembling them together. Achieving a reliable system is a difficult task, even when high quality software components are composed together. The following points are forced to take the research work in this area.

i) There is a growing need in designing and developing a component based software with increased quality.

ii) There is a need for solutions that help to reduce the risk associated with component based application development. Appropriate mechanisms can significantly reduce the cost and increase the reliability of component based application.

iii) For any component based software, component selection and assembly are the two main phases in system development. So, an optimal component selection and a proper assembly sequence are needed in the early stage of software design.

iv) After developing the component based software system, an appropriate system reliability assessment method based on component reliability should be developed to build sufficient confidence about the system.
v) The customer requirements for a software product should be incorporated from the design, planning and development stages. This upstream process of increasing the quality is needed to satisfy the customer and at the same time improving the key factors for software quality assurance.

1.5 SEQUENCE OF THE RESEARCH WORK

As the goal of component based software development is to increase the productivity, quality and time-to-market in software development, the research work is carried out in the following manner to satisfy these aspects.

i) Initially, component selection is made based on the user requirements.

ii) Then the component assembly process is analyzed and simplified.

iii) After that, reliability of component based software system is analyzed.

iv) Finally quality improvement is done by emphasizing on the user requirements.

1.6 PROPOSED METHODOLOGIES

The component based software system and its development process is analyzed. Quality improvement on the component based software is obtained by the following methods in this research work.

i) Software component selection process improvement is proposed using genetic algorithm.
ii) A new algorithm using Component dependency rating is proposed to assemble the complicated components in component based software assembly.

iii) Reliability of component based software system is analyzed using Dolbec’s model which is explained in section 6.2.

iv) Software quality assurance is done by applying quality function deployment involving voice of customer.

1.7 RESEARCH CONTRIBUTION

The following new initiations have been proposed and successfully applied in the component based software systems in this research.

i) Application of genetic algorithm for component software to select the optimal components is a new initiation in the field of CBSD.

ii) A novel technique in software component assembly has been developed and applied with a case study.

iii) A modified reliability model for component based software system has been developed and applied with a case study.

iv) Application of quality function deployment for component based software development is proposed to improve the quality of software product and its development process.

1.8 ASSUMPTIONS AND LIMITATIONS

i) Expert opinion used in this thesis is obtained from academicians and industrial experts. It is refined according to our applications to suit the research problems.
ii) The case study of Enterprise Resource Planning is chosen to go with the component based software system. General components involved in ERP are alone considered for this research work.

iii) Reliability of all the components is calculated during testing or inspection process and it is assumed that it is known before applying the reliability assessment process begins.

iv) This research work is highly suitable and may extend for all component based software systems.

1.9 CONCLUSION

Building software product is a profoundly intellectual activity. This provides a fundamental distinction between the activity of creating software product and the activity of manufacturing goods. In both cases, a common vocabulary is used to express concepts that pertain to the quality of the respective products. One way of increasing the quality of large software system is component software development. So, there is a need for solutions that help reduce the risk associated with component based software development. Appropriate tools can significantly reduce the cost and increase the reliability and quality of component based applications. Moreover they can increase the productivity of component based application developers and reduce the level and range of expertise required to build software based application.

This chapter gives an overall idea of the research work and its objectives. An overview and step by step procedure of the research work is also presented in this chapter.
Chapter 2 provides the basic definitions, background concepts and principles involved in the component based software system and also presents a brief description about methods and techniques to carryout the research work.

Chapter 3 presents a literature review of component based software engineering, component selection, component assembly, software quality, software reliability, component reliability and quality function deployment. This chapter analyses the existing related works in the literature leading to the selection of most appropriate method.

In Chapter 4, component selection is automated using genetic algorithm. The above attempt helps to take a decision about the selection of right software component. This finds an optimized component for a specification in a limited amount of time. Usage of genetic algorithm leads to the selection of best possible components among the repository of available components.

After selecting the best components from different repositories of components as explained in the previous chapter, chapter 5 describes a new algorithm to assemble the above components. Further the above algorithm finds the sequence in which the above components are assembled based on the interactions among them. This systematic procedure leads to correct component assembly.

In Chapter 6, usage ratio based software reliability estimation model is analyzed for reliability improvement on component based software system and a case study is taken. Then quality function deployment is applied for further improvement of quality of component based software system by focusing on the user requirements.

Chapter 7 provides the discussion and conclusions of all the previous chapters and also the scope for further research in this field.