ABSTRACT

The object oriented paradigm has been developed as the most effective passage to develop flexible software systems in the last decade, and is promptly supported by software communities even in the recent years. Though, the benefits of object based technology are realized by many, but its usage does not necessarily result in general, modifiable families of systems. These huge systems are often suffering from improper use of object oriented techniques, like inheritance and building of families of systems instead of developing single application. So, recent demands for re-engineering of object oriented systems has originated, because re-implementing these business critical systems from scratch within limited time and resources is a difficult task. Therefore, software re-engineering has proved itself as an effective design and architecture recovery, increasing reusability and maintainability of legacy systems. Recently, software communities paid more attention towards the component based software development to update the existing legacy systems and enhance the reusability aspects of the software systems. The component based programming provides easier way to update the existing software systems and to change these further. The metrics based estimation, system interoperability and usability become an easy process for component based systems. To address the gaps in object oriented systems this study focuses on transformation of these systems to component based systems along with component reusability aspects of components.

To better understand the step by step research flow, work is divided into four main sections. First section discusses the re-engineering and reverse engineering techniques for legacy systems. The literature shows that majority of the suggested re-engineering techniques are for object oriented systems and orientation of migration is preferably for component based system. The main drivers for re-engineering are evolving technology and high maintenance costs. Major challenges faced during process of re-engineering are lack of expertise, adaptability and lack of documentation. In the second section, legacy object oriented systems are reverse engineered using RSA tool, and hierarchical clustering is used to generate their components using the design model of classes. Once, the clusters of strongly connected classes are available, developers can easily identify bugs and introduce changes that will help in maintaining the software systems. The third section proposes a three stage framework with a new automated reverse engineering tool (ODRET) support for migrating an object oriented system to a component based system. Design model generated using the developed tool are concrete in nature. Ontology produced as an intermediate stage reduces the efforts for
understanding the inner structure and elements of the entire software system. The fourth section introduces a suite of four metrics for evaluating the degree and significance of reusability at component level, integration level and system level. This metrics based quantification of reusability provides important guidelines to judge the component quality before deploying it in final developed system and ensuring the benefits of use of component based paradigm. The obtained results show clear identification of critical components in the whole software system and the improvement in software system quality of service (QoS) by applying the reusability in an integrated form.

**Keywords:**
Component Based Systems (CBS), Framework, Hierarchical Clustering, Legacy System, Model Driven Engineering (MDE), Object Oriented Systems (OOS), Ontology Driven Reverse Engineering Tool (ODRET), Ontology Extraction, Rational Software Architecture (RSA), Re-engineering, Reverse Engineering, Reusability, Software Metric, Software Quality, Software Reuse, Unified Modelling Language (UML), Use Case Patterns (UCPs).