ABSTRACT

Software evolution of Information System (IS) most often implies the integration of legacy components with newly developed ones leading to mixed architectures. The evolution process focuses on the increased productivity and quality of such IS. It is also considered an effective methodology when the functionalities and requirements are satisfied for the legacy systems. Due to the invention and availability of the new technologies, there is a certain need to reuse the existing application which was developed using legacy code and its components. Hence, the components can be integrated into the new applications seamlessly.

A legacy system is an existing computer system or an application program which continues to be used because the user does not want to replace or redesign it. These systems are often hard to maintain, improve and expand. Legacy migration is about retaining and extending the value of the legacy investment through migration to new platforms. Migration also provides an opportunity to align applications with current and future business needs through the addition of business functionalities and through application restructuring.

From the survey of related works and literature, it is inferred that (i) The techniques for system reengineering and migration for legacy systems need to be improved to reduce the total interferences (ii) The
attributes in the legacy system should be correctly retrieved without exceeding the designer specified time constraint (iii) The efficiency of the new system need to be optimized.

In this research work, Semantic Role Abstract Syntax Graph (SRASG), Automatic Specification Evaluator (ASE) and Reengineering Legacy to Modern (RL2M) with One Time Checker (OTC) are proposed to obtain the above mentioned objectives tested with the relevant experiments and the results are discussed.

There are many issues and difficulties in automated source code conversion such as poor or no support, scarce man power, no integration with new hardware and software, inability to handle the increased transactions etc., To overcome these limitations, a novel concept has been developed based on the logic to build a syntax structure based on the semantics of the source program which is named as SRASG code conversion kit which converts a CPP program to Java.

Software systems evolve over time, as a result of requirement changes. The resulting systems tend to have a rich and complex structure which is highly coupled. Due to long term maintenance and evolution, the documentations may not be able to reflect the actual structure of legacy systems. In many cases, effective partitioning or re-partitioning is needed to decompose the legacy systems with a high cohesion and low coupling principle. The evaluation of the current legacy system is essential for further reuse, because the legacy system normally has been used and maintained for a
quite long time. This evaluation reveals the current status of a legacy system and specifies what phase it is in its lifecycle.

ASE method evaluates the legacy system specification in the low level by introducing some granules in the new system and gives some breakpoints in the new system. In general, a system has many attributes and peculiarities. The ASE method easily obtains these data without any complications. Further, the categorization as positive and negative during conversion enhances this method quite easily, as also produces an error free target system. The ASE process removes the interference from the new system but still there are some redundancy remains in the target system. This situation arises because of overlapping and inter-dependencies of the system components. When the proposed ASE method is applied again to the system, this interference is further reduced as much as possible. In many cases, the ASE gives the redundant free new modern system.

If the legacy system components are too complex, then it is very difficult to modify and extract the information to the new system. The cost, platform, and time are the main features that decide whether the older system is easy to maintain or to redevelop/redesign the available system to modern one. Usually a manual process is carried out by the system inspection and re-organisation. To improve the efficiency and decrease the complexity, a new system has been proposed based on a reengineering model which is named as RL2M system.
The RL2M system involves the functionalities: (1) Program Analysis (2) Constructs Identification (3) Slicing (4) Template Creation (5) Wrapper Generation (6) Target System Execution (7) Integration for VLSI Application.

The concept of the program slicing is used in this RL2M model which finds all the statements in a program that directly or indirectly affect the value of the variable occurrence. It is used for simplifying the programs by focusing on the selected aspects of semantics. Dynamic slicing is one of the program slicing technique tends to produce small slices of the original system.

A dynamic slice consists of the statements that influence the value of the variable occurrence for a specific program input. Wrapping is an interface technique based on the program slicing which presents around an existing piece of software, providing a new view of the software to the external systems/views.

Each system component has its own exclusive function. All the components of the system follow their respective unique patterns. To define and identify the pattern before the implementation of the converted target system, we also made an attempt in RL2M which is normally automated i.e. an provision for manual conversion is also available as optional at the stage of execution of the target system. To overcome the drawbacks like RL2M approach, a new method has been introduced namely One Time Checker (OTC) where the bug detection and rectification is done on the target system.