ABSTRACT

Software engineering describes the group of methods that construct and support software products by employing an engineering approach. Methods based on models and theories are employed by engineering disciplines. Specifying a hypothesis, designing and performing an experiment to verify its truth, and interpreting the results involved in scientific methods. Measuring the variables differentiates cases, measuring the changes in behavior, and measuring the causes and effects are the supportive scientific method measurements. After the validity of a model or the truth of a theory is confirmed by scientific method, the theory is applied to practice by continuously using measurements. More visible characteristics and relationships in estimating the enormity of problems and in shaping a solution to problems can be obtained by means of effective measurements. Coupling analysis is one of the diverse methods used in software system for modeling and measuring the relationships between components. Two components having any type of connection or relationship between them are coupled by coupling analysis. Generally, the coupling nature has been categorized into diverse levels or types.

Coupling analysis attempts to capture all the attributes of the relationships between components of a given software program, by defining a theoretical model. By defining a set of measures, the coupling levels are also quantified by it, a major role is played by software metrics in the planning and control of software development projects. Software development and maintenance has important applications for coupling measures. The extent to which each program module depends on each one of the other modules is termed as coupling or dependency. “Static” couplings are only taken into account by conventional coupling measures. They may considerably underestimate the complexity of software leading to underestimation of code inspection, testing and debugging needs because dynamic coupling due to polymorphism are not taken into account. Therefore, inferior
predictive accuracy is likely in quality models that utilize static coupling measurement.

Works available in the literature for software metrics have mainly concentrated on centralized systems and only very few of them have focused on distributed systems and more specifically on service-oriented systems. Conventional non-distributed systems differ from systems with distributed components in several ways including communication type, latency, concurrency, partial, versus total failure and referencing parameters—passing strategies. Normally, distributed systems with service-oriented components are more complex because they accomplish efficiency and other quality characteristics in a more heterogeneous networking and implementation environment. In order to overcome these issues,

We propose a hybrid model to measure the dynamic coupling present in distributed object-oriented software. The proposed method has three steps; they are instrumentation process, post process and coupling measurement. First, the instrumentation process is performed. In this process, to trace method calls, a modified instrumented JVM has been used. During this process, three trace files, .prf, .clp and .svp are created. In the second step, the information present in these files, are merged. At the end of this step, the merged detailed trace of each JVM contains pointers to the merged trace files of the other JVM’s such that the path of each remote call from the client to the server can be uniquely identified. Finally, the coupling metrics are measured dynamically.

We have also conducted the experiments for more iteration in the continuous test runs as well as discrete test runs and the results support the claim that dynamic coupling is reduced. The Proposed system was implemented by Java. The implementation results show that the proposed system effectively measures the dynamic coupling.