Abstract

Information systems of many organizations are processed through system of interrelated ‘C’ programs. In a ‘C’ program each statement contains functional dependencies amongst the attributes. Sometimes these functional dependencies may redundant in different statements. The existing application programs used in the maintenance of the information system are
lengthy, and because of the perennial maintenance of the program, these functional dependencies are unevenly scattered. Thus, some functional dependencies may be implicitly present in other statements or sometimes they are unevenly scattered across the entire program. This situation creates the complication in the reengineering process which creates scuffle in selecting the attributes for a class on the basis of the cohesive property.

While abstracting the object structures, and making the cohesive groups of attributes, the recursive implicit containment of one functional dependency within another creates complication in the granularity of design elements as the implicit dependencies have ripple effect on the dependencies of attribute. This paper attempts to propose the identification of functional dependencies from the realization of program code, and their minimization through the minimal cover process. The correctness and completeness of the abstraction is a straight forward process.

Reference

- Rajkumar N. Kulkarni and Shivanand M. Handigund, “Moulding The Legacy ‘C’ Programs


**Index Terms**

Computer Science

Information Technology

**Key words**

Functional dependencies minimization

abstraction

reengineering

business rules

legacy systems

reverse engineering.