LIFT Revisited: Enhancing the Understanding of NATURAL/ADABAS Legacy Systems

Kellyton dos Santos Brito
Exact Sciences Department
Feira de Santana State University
Feira de Santana, Brazil
kellyton.brito@gmail.com

Vinicius Cardoso Garcia
FACITEC - Pernambuco State University, and
Informatics Center – Pernambuco Federal University
Recife, Brazil
vcg@cin.ufpe.br

Silvio Romero de Lemos Meira
Informatics Center - Pernambuco Federal University, and
Recife Center for Advanced Studies and Systems (C.E.S.A.R)
Recife, Brazil
silvio@cesar.org.br

Abstract— LIFT (Legacy InFormation retrieval Tool) is a tool for reverse engineering and understanding of legacy systems, in particular NATURAL/ADABAS systems. Its provides several capabilities, such as call graphs, identification and visual representation of application tiers, the presentation of flows from screen to database entities, cluster analysis and documentation generation, among others. In this paper, we present two new functionalities of LIFT tool: the reconstruction and visualization of screen layouts, and the graphical visualization and automatic migration of ADABAS database structure.

Keywords – Legacy systems, system understanding, reverse engineering

I. INTRODUCTION

Legacy systems are a main concern for organizations, which must manage these systems while keeping up with new technologies and business requirements. In addition, it is consensus that software maintenance is one of the most expensive tasks of the software life cycle. Although software migration is presented as one solution to reduce this cost, and recent studies show that the top priority for IT in the next year for most companies is application modernization [1], the large scale adoption of tools to increase the quality and productivity of these tasks is still limited [2].

In this context, this work presents the new functionalities of LIFT – Legacy InFormation retrieval Tool. A previous version of the tool was already used in the first phase of a large reengineering project involving almost 2 million lines of code [3]. Based on the lessons learned in this project, the tool was enhanced with new functionalities.

In Section 2 we briefly present the tool’s architecture and functionalities, and a description of the new capabilities. Section 3 concludes the paper.

II. TOOL’S ARCHITECTURE AND FUNCTIONALITIES

LIFT is a three tier application written in JAVA, with an Understanding Environment on top tier, business components and modules responsible for the main functionalities in the middle tier, and a layer for persisting all manipulated information in a database. The architecture follows the common reverse engineering process, as shown in Figure 1.

The parser component has modules for organizing the available system data. It receives the source code, parses it and inserts all statements in a structured database. Next, the data is manipulated and prepared to be used by the application in a higher abstraction level, for the identification of modules and relationships, in addition to code statements analysis, program slicing and identification of source code comments.

The analyzer component is responsible for analyzing the pre-processed code and generating representations. This component generates the application’s call graph, identifies application paths from screens to database access, and detects clusters, aiming to help the user to identify related modules and define components.

The visualizer component is responsible for managing the data generated by other modules and presenting them to the user in an understandable way, in special the call graphs, clusters and paths.
On top, the Understanding Environment is responsible for integrating the other components, containing graphical interfaces for the tool’s functionalities, including reports generation and application visualization and manipulation.

The first version of the tool was used in the first phase of a large reengineering project [3]. With the experience acquired in this phase, two new functionalities were identified and implemented:

(i) **Visualization of application screens layout:**
NATURAL legacy applications run on mainframe environments and, in most cases, the only way to visualize these screens is by running the application. It is a strong limitation for system understanding, because in this context several data input and validations are needed to advance from one screen to another. In order to solve this limitation, we enhanced the analyzer component with the Screen Visualization functionality, which allows the user to see the application screens in the understanding environment. LIFT analyzes the legacy code, searching for screen layout information, and then builds its own version of the original screen, which can be then visualized without the need to run the actual legacy system, as illustrated in Figure 2.

(ii) **Visualization and migration of database structure:**
NATURA/ADABAS databases contain structures that are not present in relational databases, such as multivalored and periodic attributes. Multivalored attributes are composed of items that are stored in the same attribute, such as an address composed by a street, number and postal code. The concept of periodic attributes is the same as that of the first normal form. For instance, a person can have several phone numbers (home, mobile and business number), which are stored in the same attribute of the same relation. In order to aid the user to perform the migration of these database structures, we developed the Database Reconstruction, Visualization and Migration modules. The Reconstruction module rebuilds the original database structure. The Migration module generates a new relational structure, with SQL DDM (Data Definition Language) constructions for all tables, SQL and HQL (Hybernate Query Language) commands for basic data manipulation (create, update, find and delete) that can be reused in new applications, and JAVA classes representing all tables, which can also be used in new applications. Finally, the Visualization module shows a visual representation of these data. It is important to highlight that these modules automatically deal with periodic and multivalored attributes, by generating new tables with foreign keys to the original table. The visual interface of these modules is shown in Figure 3, with the Reconstruction module at left, the Migration module at right and Visualization module in the center.

![Figure 2. LIFT screen visualization](image)

![Figure 3. Database visualization](image)

### III. CONCLUDING REMARKS

In this work we focus on the new features of LIFT tool, for screen visualization and database structure migration. Although database migration is presented in several works in the literature, migration of ADABAS databases is not well covered. Moreover, the reconstruction and presentation of application screens, mainly NATURAL applications, is not yet well covered by reverse engineering literature and practice.


### REFERENCES

