Indian Journal of Science and Technology, Vol 10(34), DOI: 10.17485/ijst/2017/v10i34/115507, September 2017
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Effective Software Installation for Embedded Software by Applying the Reverse Engineering Approach

E. Naresh[1,2] and B. P. Vijaya Kumar[2]

[1]CSE, Jain University, Bangalore - 560069, Karnataka, India; nareshkumar.e@gmail.com
[2]Department of ISE, RIT, Bangalore - 560054, Karnataka, India; vijaykbp@yahoo.co.in

Abstract

Background/Objectives: Reverse Engineering of Electronic Programme Resources (EPR) is the process of reading the binary data from the files having ".bin" extension and presented in a well-structured format. **Method/Analysis:** The resources used with the EP resources are text, pictures, colors and UI controls, each of these are stored in XML format then converted to .bin extension files to be flashed on an embedded device. If the developer wishes to see the contents, it is very difficult to analyze the format as well as the data, as it is in binary format and needs an expert to interpret it. **Findings:** The required concept can be shown visually by which developer will know the format readily, and this work tries to satisfy the requirement of analyzing the binary format for heterogeneous multimedia embedded software systems. **Applications/ Improvements:** The research introduced can be applied to any software binaries that are to be flashed on an embedded device obtaining the necessary binary format.

Keywords: Binary Data, Binary Format, Bin Generator and Widgets, EP Resources, Reverse Engineering

1. Introduction

Electronic Programme Resources (EPR) is an application running on a digital set-top-box. The EPR allows the viewer to view programme's transmitted by the broadcaster, to record programs for viewing later, to pause live programs, to navigate the programs, to buy and order programs, view information and set preferences.

Resources for the Electronic Program (EP) are text, pictures, colors and fonts etc. These resources are converted in the form of binary data, which is to be validated before installing into the Set-Top box.

Reverse engineering of EP Resources is necessary to help the developer to view the contents of the binary files and to validate the binary data that is created by using the EPR Designer before flashing the data into the Set-Top Box (STB). The rest of the paper is discussed as follows; section 2 briefly describes the EPR system. The multimedia embedded systems and their EPRs used with respect to the binary format analysis and description of setting up of EP resources are discussed in section 3. The proposed model is described in section 4. The preliminary works and prototype implementation and its results are explained in section 5. Finally the concluding remarks are depicted in section 6.

Alternative models like component based and aspect-oriented software engineering techniques are used for the said purpose and these models will be helpful in reducing the software development cost in the other perspective.

2. EPR System

The resources used with the EPR's are strings, images and graphics, each of these are stored in XML format then converted to .bin files to be flashed on the set-up box. E.g. an EPR might support strings in Hindi, French and English, so the supported languages are three, and in turn they could have say "fifty strings" supported. So this meta-

*Author for correspondence

data information along with the actual string content is put in strings .bin and the associated binary file is in particular format.

If the developer wishes to see the contents, it's very difficult for user to analyze the format as well as the data, as it is in binary format and needs an expert to count the number of bytes and interpret them. There is a need for a model that can show this information visually by which developer know the format readily. The proposed model meets the above-mentioned requirements and is implemented using java swings.

EPR Designer is a visual drag-and-drop tool that is used to create UI designs for EPRs and interactive applications as shown in Figure 1. The EPR Designer tool provides WYSIWYG environment.

EPR Designer allows a broadcaster to update and re-skin the EPR for transient and persistent changes, that is, it provides an authoring environment that facilitates the broadcasters to create, modify, schedule, and broadcast dynamic updates through graphical user interface[1].

EPR Designer enhances the process of authoring and modifying EPRs[2], since:

- It allows rapid (drag-and-drop) creation and/or customization of EPR UI by a non-programmer.
- It contains a robust set of UI building blocks that make EPR development easy.
- It provides widgets such as Grid, Banner, Marquee, Video, and so on.
- It allows the broadcaster to change or configure widget properties and location.
- It allows the broadcaster to design scenes for both SD (NTSC and PAL) and HD (1280*720) resolution.

EPR Designer internally uses the .net framework to create the .xml files for the designed screens, fonts, images, etc and these xml files are converted into .bin files with the help of a generator (EPR Designer invokes internally). These bin files are then loaded into the Set-Top Box (STB)[3].

The Generator is a command line tool that is used to convert EPR UI XML files into their binary equivalents. The XML files are generated by the EPR designer, which is a visual editor, used for creating UI screens[4]. The Generator is used in conjunction with the EPR designer. i.e., it invokes the Generator when selected through a menu option it provides to the user. It can also be used as a stand-alone tool. If any of the selected input files is not present, then the generation process stops at that point. The prior set of .bin files if any, although usable are recommended not to be used, since the end objective of the user is not met.

3. Analysis of Binary Files

The size of the related fields in the format may vary from requirements of the set-top box. E.g., Lang Type may require 1 byte of data and the respective offset value may occupy 4 bytes of data to read as shown in the Figure 2.

Manual Analysis of binary file like strings .bin using Hex Editor is shown in Figure 3. It gives the hexadecimal view of the binary data in the strings .bin file[5]. For exam-

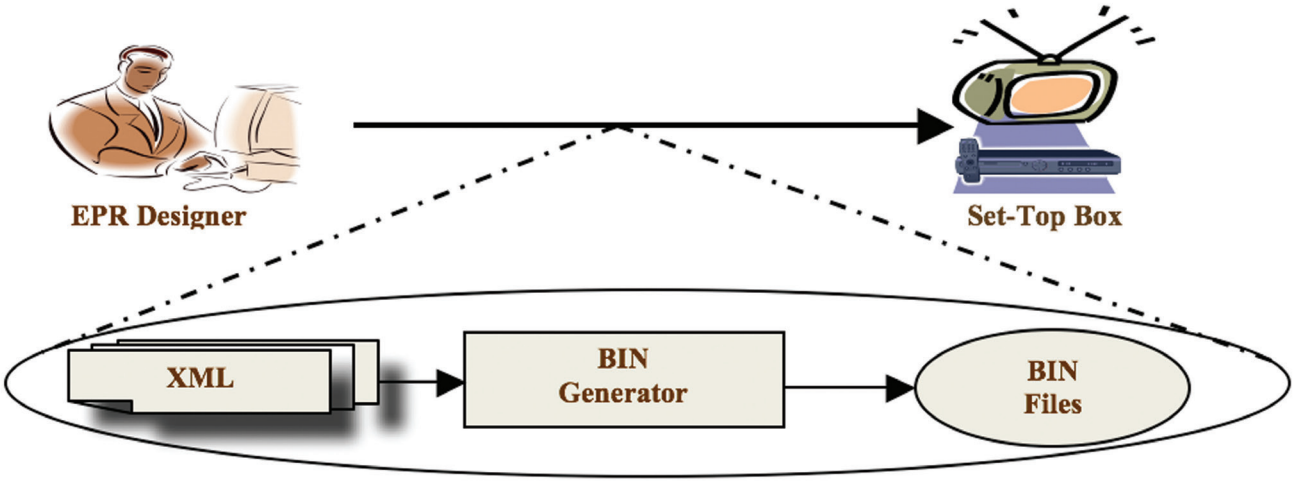

Figure 1. Installing EP Resources into Set-Top Box.

Ver	B/L	Link_Ct	Link_Type	Link_FileName	Link_Type	Link_FileName
Lang_Ct	String_Ct	Lang_Type	Offset	Lang_Type	Offset	
RID	RID	Size	Value	Size	Value	Size
Value	Size	Value				

Figure 2. Strings Binary Format.

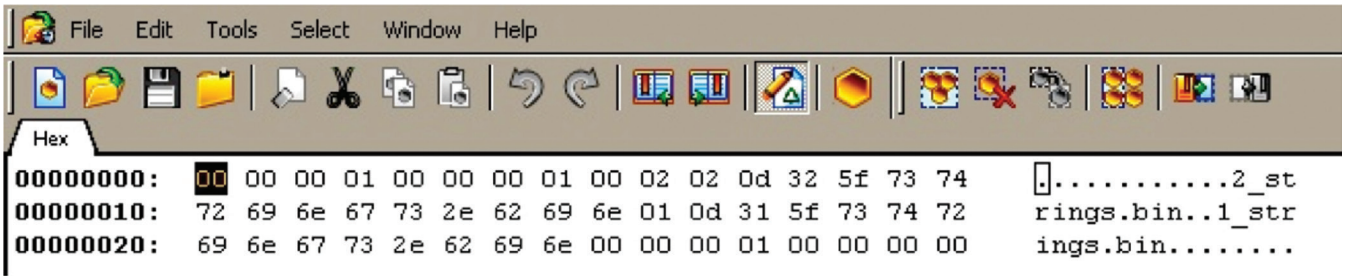

Figure 3. View and analysis of binary data using Hex Editor.

ple, the representation of one byte binary data like 1111 1111 is represented as ff in the hexadecimal value in the Hex Editor. The process of reading the binary file in the hex editor is based on the format of binary file and its field size value (in bytes). For example, In the Strings, Images and Graphics binary format, there is a version field, which contains two more fields called serial version and file version, which is 4 bytes of each. Read the 4 bytes i.e., 00 00 00 01 from the hex editor shown in the figure 3 and assign the value to first field i.e., serial version = 1. In the similar way, read the whole binary file till the format completes.

4. Proposed Framework

The Figure 4 gives a general description of the Reverse Engineering of EP Resources. The input data to the Reverse Engineering of EP Resources is in the form of binary files, generated by the EPR Designer. It processes the binary file and displays the source data in a user interface, called JTree, by using the property file where the binary format structured is defined[6].

The main purpose of using property file in this project is used to update the field values of binary format dynamically and hence reduces the future maintainability cost. i.e., defining the field name and its size in the property file and by accessing this property file at run time, we can update the values at any time in the project cycle[7,8].

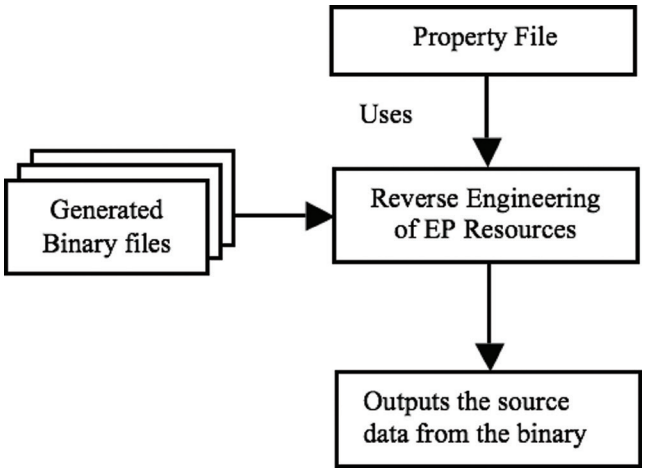

Figure 4. Framework for Reverse Engineering the EP Resources.

User/Developer is the main part of the system that involves in the execution of the binary files, where user checks that the selected binary file is in its defined structure and is in valid format. Application manager helps in executing the binary files and displaying the structure and contents of the selected binary file. It also acts as the Front End application for the Testing of the given Binary Files using the component based software engineering approach[9].

5. Experimental Results

Below are the paraphrasing results of binary file representing the time taken by different users in Figure 5, 6 and 7.

From the above results, we can say that the automation process is more efficient for reading the binary data, as automation process takes much less time compare to the manual interpretation of binary data.

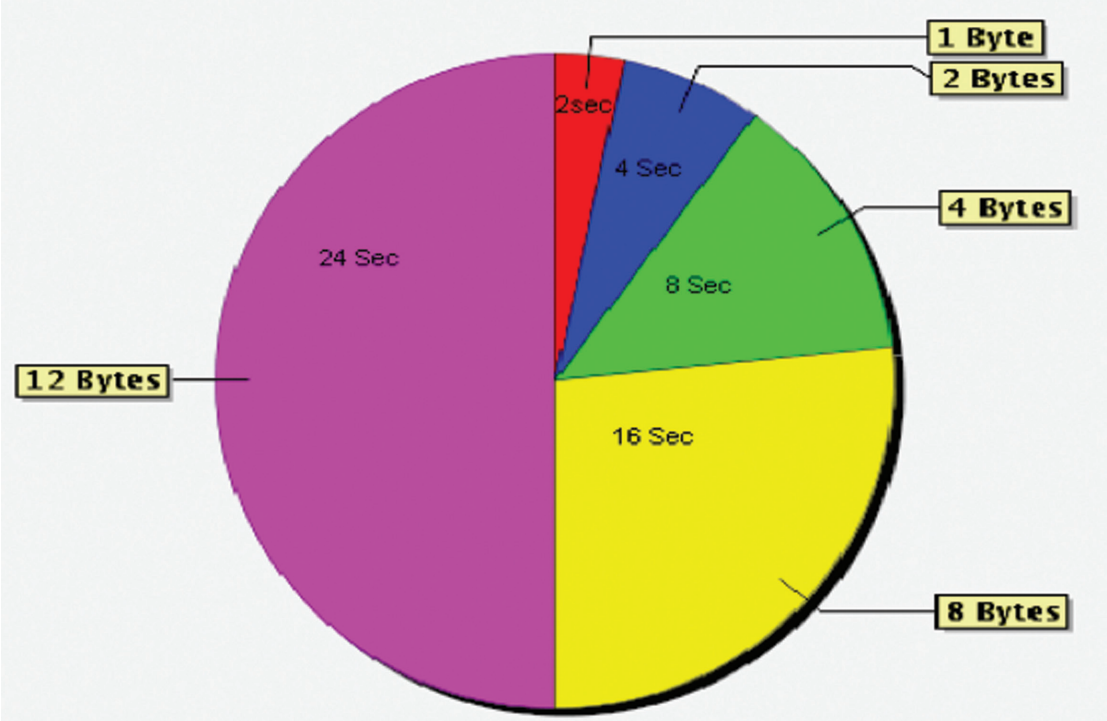

Figure 5. Time taken by the No-Voice user to read binary data.

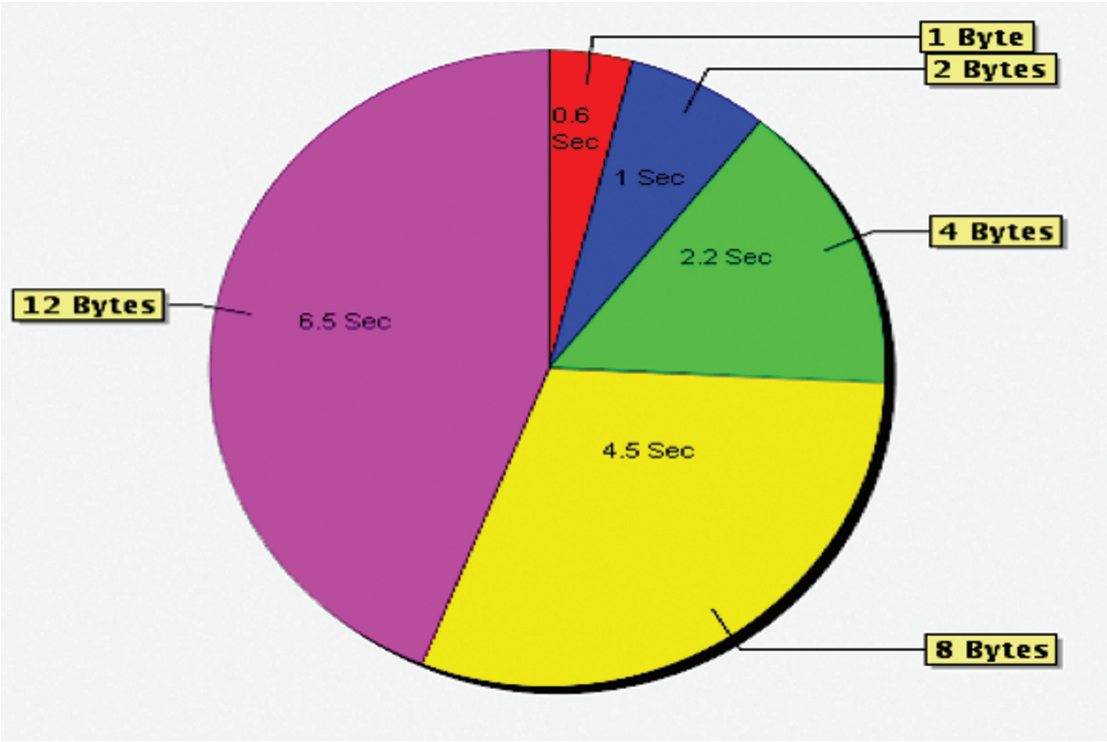

Figure 6. Time taken by the Expert user to read binary data.

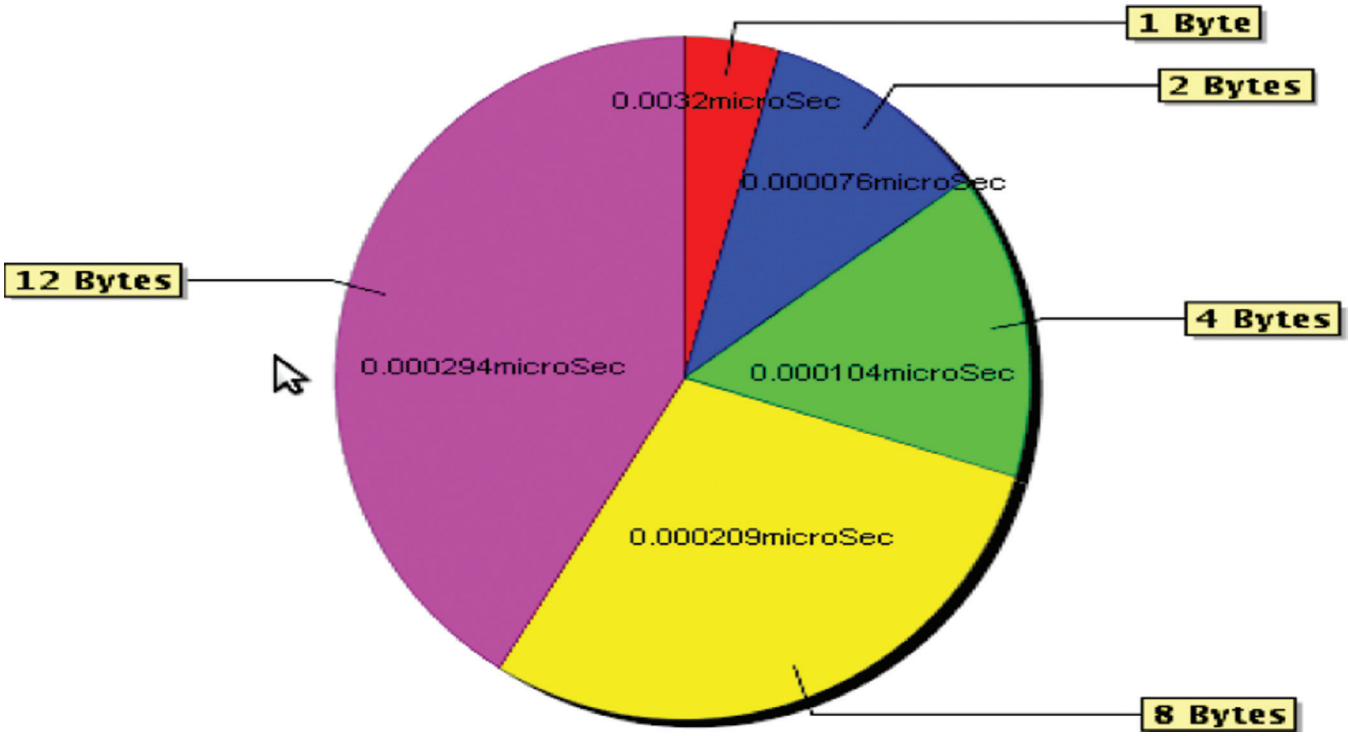

Figure 7. Time taken by automated process to read binary data.

6. Conclusions

Reverse Engineering of EP Resources is very helpful for all the developers who work on the Set-Top boxes, as they should validate their binary files before setting up into the Set-Top boxes. This model is used as a reverse engineering process tool, as it decrypts the binary data into meaningful information and for the replacement of manual interpretation of the binary code; hence increases in efficiency of the developers and curtails the time and cost of binary file validation. This tool can be integrated with EPR designer, instead of using from the software repository. So, this tool will be used as an automated tool for validating the binary files from the EPR designer.

Creating the log files, gives the information about the binary files after validating from the EPR designer. For example, if there is any string id is missing, then it should be recorded the same in the log file. The future work to the Reverse Engineering of EP Resources is to provide the Drag-and-Drop feature, so that we can directly execute the binary files without using the file chooser dialogue box.

7. Acknowledgement

We indebted to management of Ramaiah Institute of Technology, Bangalore for excellent support in completing this work at right time, and also for providing the academic and research atmosphere at the institute. A special thanks to the authors mentioned in the references.

8. References

1. Foster E, Endicott NY. Design of a Set-Top Box System on a Chip. IBM Research, EEE Computer Society. 1999; p. 608.
2. Fowler M, Scott K. MA: Addison-Wesley: UML Distilled: Applying the Standard Object Modeling Language. 1997.
3. HHD software products. Available from: http://www.hhdsoftware.com/doc/hex-editor/getting-started-with-hex-editor-neo-documentation-road-map.html. Date accessed: 05/05/2014.
4. Lee H, Ferguson P, Gurrin C, Smeaton AF, O'Connor NE. Silicon Valley, California, USA: Balancing the Power of Multimedia Information Retrieval and Usability in Designing Interactive TV. 2008 Oct; p. 1-10.

5. Janet G. Luxembourg: Infomedia SA: Data Preparation for Interactive Electronic Program Guides. 1996; p. 12-16, Conference Publication No. 428.
6. Han J, Han I, Park HR. Daejeon, Korea: IEEE Publication: User-Configurable Personalized Mosaic Electronic Program Guide, Home Network Group, ETRI. 2008; p. 1-2.
7. Rutledge L, Bailey B, Ossenbruggen JV, Hardman L, Geurts J. Generating Presentation Constraints from Rhetorical Structure. San Antonio, TX: Hypertext. 2000; p. 19-28. Crossref.
8. Sahaj Computer Solutions. Object-oriented-systems-development-life-cycle. 2011; p. 1-51.
9. Jain P. Towards The Adoption of Modern Software Development Approach: Component Based Software Engineering. Indian Journal of Science and Engineering. 2016 Aug; 9(32):1-5.