Towards a Methodology for the Development of Plug-In

Saket Ravindra Gonte

Department of Computer Engineering, STES's Sinhgad Academy of Engineering, Kondhwa (Bk), Pune, Maharashtra, India.

---***---

Abstract-- *Every software development needs a composed set and clearly defined work phases to deliver a high-quality system. Thus, this work proposes a proper skeleton for the development of a plug-In for MS-Word using Software Development Life Cycle approach. Also, our methodology includes sequence of activities needed to be followed for the system to allow interfacing with scripting language python. In time, we also provide different stages that a document passes through in order to meet the goal of plug-In, provide actual examples, as well as analyze architectural aspects and specialized tool support for the overall development and interfacing of plug-In. We do believe this work contributes and makes plug-in development easier and thus enhance the usability of MS-Word.*

Index Terms--MS-Word, plug-in, python, software development lifecycle.

I. INTRODUCTION

Websites, application or a plug-In are the different ways to extend the usability of a software. In today's world, be it a technical or a non-technical user, one prefers using a system which is efficient and user friendly. It is always much easier for any user to use a plug-In (word/excel/power point) rather than using a completely new system i.e. a website or an Android application as this new system has completely different user interface and functionalities. However, in general, we observed that there are no sufficient methodologies present in order to guide us in the development of plug-In.

This paper contributes for the Development of MS-Word Plug-In(DMP) using Software Development Life Cycle(SDLC) approach. We espouse spectrum of SDLC methodologies along with the tool support. Furthermore, security being an important aspect is also considered in the DMP.

Further the paper will be organized as follows. Here section II gives background and motivation whereas section III analyzes current for cryptography in SDLC. Section IV deeply describes our methodology. Section V concludes the paper.

II. ACKGROUND AND MOTIVATION

A plug-In (also known as an extension) is a software component which enables customization to an existing computer program [1]. These plug-Ins are mainly used to extend the applicability of any application. Audio editors, email client, graphics software, media players, packet sniffers, text editors, web browsers, etc. are some applications where plug-Ins are used in order to enable third party developers to create customized applicably [2]. Plug-Ins being completely dependent on the services provided by the host application, the update and addition of new functionalities are independent of the host.

There are two strategies used for the development of plug-In [3][4]. The first is by using shared libraries installed in a place prescribed by the host application [3] and the second one is by loading a directory of script files written in a scripting language like python [4]. The first approach i.e. using an applet works under a scope of a large program or a plug-In. This reduces the independency which may further lead problems in updating or addition of any new functionality. Thus, in order to increase the efficiency of software, we focused on using the second approach i.e. using a scripting language python for MS-Word.

III.DECONSTRUCTING TRADITIONAL PRACTICE

This section represents all the basic concepts of plug-In which are necessary to know before understanding the development process. Also, this section illustrates how SDLC is related to plug-In. Unfortunately, there is no proper book or a website which provides adequate information for the development in terms of Software Engineering.

A. Plug-In and the SSDL:

MS-Word provides us multiple functionalities irrespective of which there is necessity to include extra feature thus, making the MS-WORD more customized. To make MS-WORD customized, Microsoft has provided a mechanism called plug-In which acts as an interface between MS-WORD functionality and user required functionality. MS-WORD is mainly used for documentation purposes. Various plug-ins are now developed in order to make documentation more efficient.

Software development lifecycle is a systematic process to design a software system in order to obtain hit quality software. [8] Software development lifecycle includes all aspects of software development that is from planning to maintenance of software.

B. Plug-In into the SSDL:

MS Word plug-In development requires various phases which comprises of various conversions. The plug-In development is a combination of various languages which includes Ajax, XML, JavaScript or any other language for UI purpose. Software Development Lifecycle which includes planning, design and architecture, construction, maintenance and deployment. All these phases are explained in detail in section 4.

IV.THE PROPOSED METHODOLOGY

[9] Computer systems are complex and with a pre-planned strategy this complexity can be minimized. For this purpose, SDLC has provided multiple models to synchronize various processes and develop a system in structured, deliberate and methodical way. This is an informal documentation of the methodology for the DMP. Since there is no documented approach which includes the process of development of plugin into SDLC, the following section describes the detail flow of DMP, the important SDLC concepts and their inter-relation.

A. teps for DMP:

SDLC defines the process of development of an application in an organized way [7]. There are multiple phases namely requirements and planning, design and architecture, security, testing and deployment.

1. Plug-In software requirement and planning:

The goal of the project is determined and accordingly to achieve that goal, the tasks are scheduled. For the development of MS-Word plug-in it is necessary to analyze the objective to achieve the goal, go through the requirements of the clients and also check whether the development of this plug-in will not exploit the regulations and policies of MS-Word.

Consider a MS-Word plug-in is to be developed for content improvisation (which includes spell check, tone check, grammar check). Now for this system it is necessary to plan how the working of the system will be, which components to be used for the development like software components namely Visual Studio, for the backend purpose scripting languages like python and for the UI purpose HTML, JavaScript can be used.

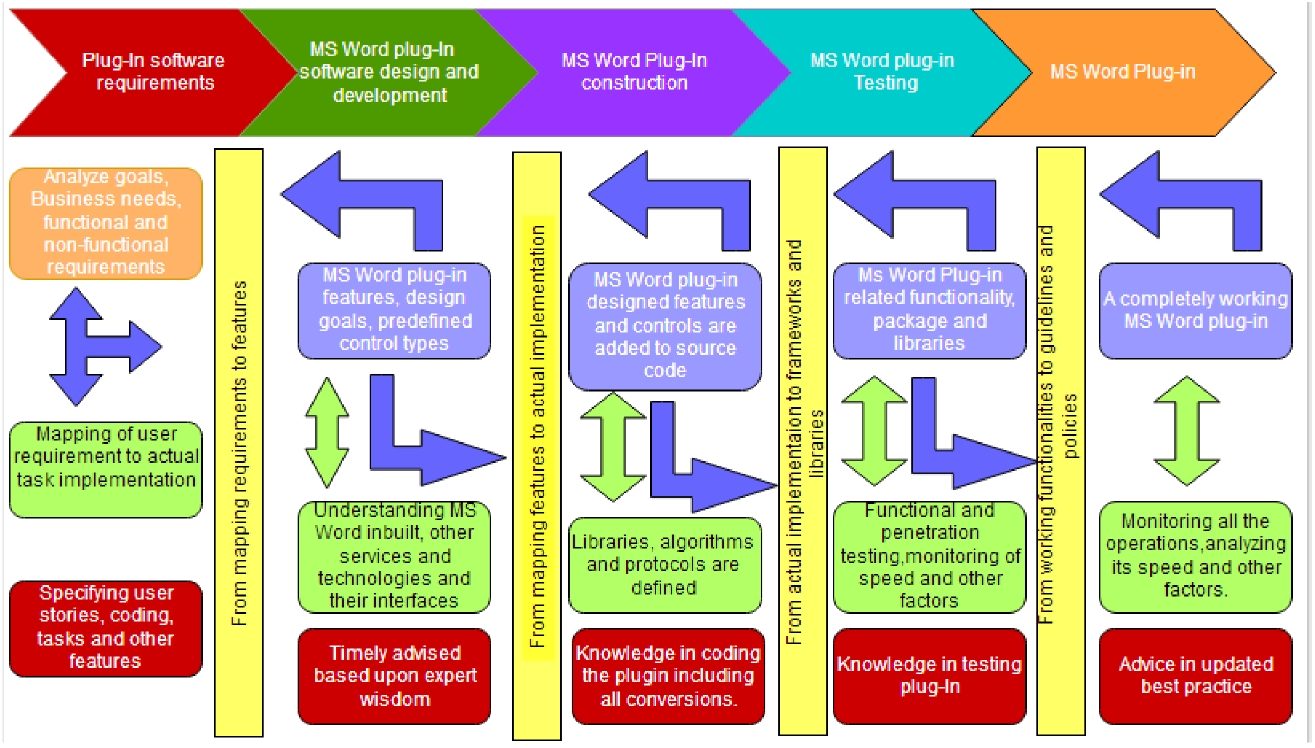
Plug-In software requirements
MS Word plug-In software design and development
MS Word Plug-In construction
MS Word plug-in Testing
MS Word Plug-in
Analyze goals, Business needs, functional and non-functional requirements
Mapping of user requirement to actual task implementation
Specifying user stories, coding, tasks and other features
From mapping requirements to features
MS Word plug-in features, design goals, predefined control types
Understanding MS Word inbuilt, other services and technologies and their interfaces
Timely advised based upon expert wisdom
From mapping features to actual implementation
MS Word plug-in designed features and controls are added to source code
Libraries, algorithms and protocols are defined
Knowledge in coding the plugin including all conversions.
From actual implementaion to frameworks and libraries
Ms Word Plug-in related functionality, package and libraries
Functional and penetration testing,monitoring of speed and other factors
Knowledge in testing plug-In
From working functionalities to guidelines and policies
A completely working MS Word plug-in
Monitoring all the operations,analyzing its speed and other factors.
Advice in updated best practice

Fig 1. Steps for DMP with SSDL phases

2. Plug-In software design and architecture:

[9] The design and architecture phase describes the layout of the system being developed. It also describes the desired features and the flow of operations in detail. Design decides how the system will be developed and also decides the User Interface of that system. A design should be such that it helps users to get one job done without affecting the other.

Since MS Word is being used by all i.e. technical as well as non-technical people, it is necessary that the design of the plug-in should be user-friendly. Also, as per the rules and regulations provided by the Microsoft, the plug-in developed should not exploit the basic layout of the MS Word.

3. Plug-In software construction:

This step includes the actual coding that is implementation of algorithms, protocols and libraries. The source code of various modules integrated thus producing complete software functionality.

To develop a MS Word plug-In, the input i.e. the data provided by the user requires various conversions. Following diagram shows how the document processing is done.

For example, consider the above example of content improvisation. Suppose, if the user is interested in correcting all the spell errors in the document, the MS Word plug-in will scan the entire document which will later be converted into XML format (since any document in MS Word is accepted in XML only). This converted text matter is further sent to backend for processing, but the backend requires the input in text format hence, an XML-to-Text conversion is mandatory. This document is processed and accordingly the output is displayed to the user.

4. Plug-In software testing:

[9] This mainly brings all the modules being developed together and accordingly the entire functionality is checked. Testing is done on some specific modules and on the software functionality as a whole. The implementation of the libraries, packages and the coordination between various modules are tested. Various kind of tests are performed and analysis on the basis of how the code is executed, which module in the code requires more time and why, all these factors are ensured in this phase.

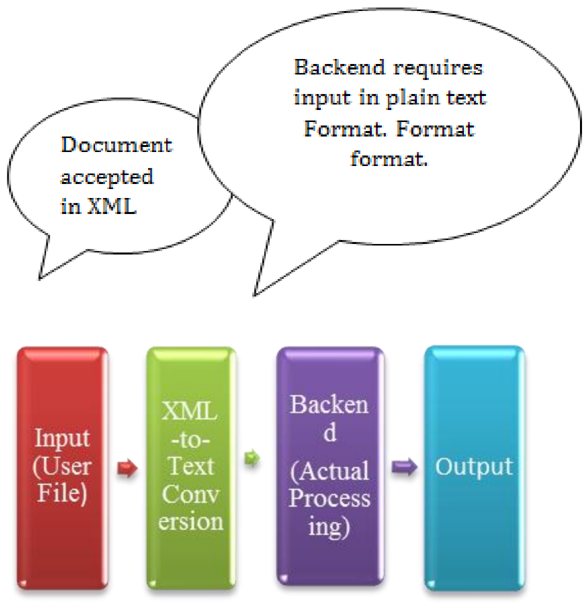
Document accepted in XML
Backend requires input in plain text Format. Format format.
Input (User File)
XML -to-Text Conversion
Backend (Actual Processing)
Output

Fig 2. Text Processing phases in MS-Word Plug-In.

For example, as per the given example of a MS-Word plug-In for document checking, testing will be performed on each module i.e. spell check, grammar check and tone checking. Finally, the entire software as a whole will be tested. Time complexity of each module will be ensured. The maximum limit of a document that the software can deal with, type of document, MS-Word features and its coordination with developed plug-In.

5. MS Word Plug-In Deployment:

This phase comprises of the complete functionality which is ready to run out of the production environment that is in real world. The entire software is continuously monitored and periodically tested in order to ensure if there is any kind of violation of policies and guidelines. User manuals are provided for the user to know the features and other details including the terms and conditions.

V.CONCLUSION

Plug-In is a provision which lets you to work in an existing environment. Development of a plug-In without a proper software approach is not possible. Understanding all the approaches given above, MS Word Plug-In development will be efficient. We do believe that DMT, in its current stage of maturity, is a step forward, in providing better ways in development of a plug-In.

REFRENCES

[1] Beginner's Guide for Wordpress "http://www.wpbeginner.com/glossary/plugin/"

[2]https://en.m.wikipedia.org/wiki/Plug-in_(computing)

[3]"https://en.m.wikipedia.org/wiki/Shared_library"

[4] "https://docs.moodle.org/dev/Plugin_types"

[5] TechEase: What is a Plug-In and how do I installit http://etc.usf.edu/techease/win/internet/what-is-a-plugin-how-do-i-install-it/

[6]"http://www.google.co.in/amp/s/www.cigital.com/blog/what-is-the-secure-software-development-lifecycle/amp/"

[7] Roger S Pressman "Software Engineering: A Practitioner's

[8] Pfleejer " Software Engineering- Theory and Practice" 4th edition