The Software–System Architecture of YEROTH–ERP–3.0

DR. XAVIER NOUMBISSI NOUNDOU

This document describes the thick–client software–system architecture of YEROTH–ERP–3.0. This document also explains the reasons for which we chose to design and implement YEROTH–ERP–3.0 as a thick–client software–system, as opposed to currently more popular web–browser–based software–system.

– December 5, 2020 –

Contents

List of Figures

List of Tables

Chapter 1

Developer Biography

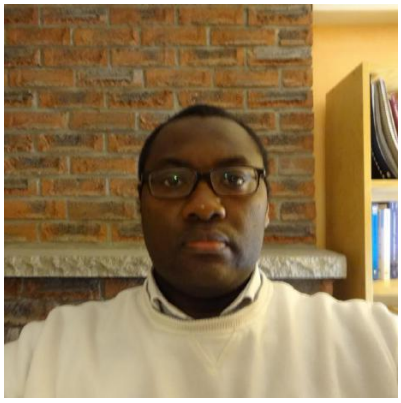

Figure 1.1: Portrait of Xavier.

DR. XAVIER NOUMBISSI NOUNDOU is a Cameroonian born on September 16 1983 in DOUALA (LITTORAL region, CAMEROON).

Xavier is a "Diplom–Informatiker (*Dipl.-Inf.*)" of the **University of Bremen, Bremen, Bremen, GERMANY** (May 25, 2007).

Chapter 2

Introduction

2.1 Motivation

YEROTH–ERP–3.0 is an **Enterprise Resource Planing (ERP)** software–system that aims 'effectiveness' and 'simplicity', compared to other high ranked ERP software–systems (e.g.: 'Sage Gescom i7', 'SAP Business One', etc.).

We chose to design and implement YEROTH–ERP–3.0 as a thick–client software–system because of the following reasons:

1) the implementation language C++ offers much flexibility (use of macro, multiple inheritance, etc.)
2) the availability of 'WHAT YOU SEE IS WHAT YOU GET' (WYSIWYG) tools for fast and useful user interface design (e.g.: `Qt designer` [Com20], `miniStudio (vxWorks)` [WEI20], etc.)
3) the low number of logical software architecture layer (2) involved with the operation of a thick–client software–system, as opposed to a web–browser–based software–system (with at least a 4 layers in its logical software architecture).

2.2 Definitions

2.2.1 Logical software–system architecture

2.2.2 Physical software–system architecture

Chapter 3

Thick–Client VS Web–Browser–based Software–System Architecture

	Thick–client application ✓	Web–browser–based application
business code	all computers	application server
co–related software–systems	1 (DBMS)	at least 3 (DBMS, web / application server)
user interface	all computers (thick–client gui)	all computers (web–browser)
number of logical layers	2 (client and data)	4 (client, presentation, logic, and data)
rapid prototyping (WYSIWYG tools)	yes	very limited
software security vulnerability	low (1 programming language)	high (*several* programming languages)

Table 3.1: Thick–client application VS Web–browser–based application.

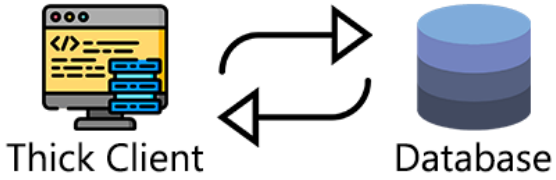

Figure 3.1: 2–layers logical architecture of thick–client software–system (Image copied from [sec20]).

Figure 3.1 illustrates an example of a thick–client software–system with a 2–layers logical architecture.

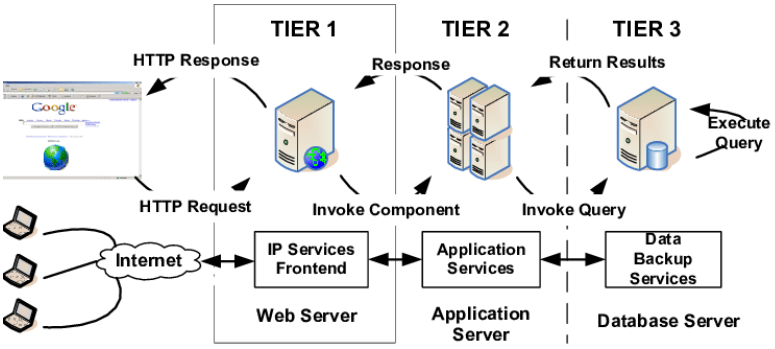

Figure 3.2: 4–layers logical architecture of web–browser–based software–system (Image copied from [KM06]).

Figure 3.2 illustrates an example of a web–browser–based software–system with a 3–layers logical architecture.

Table 3.1 compares thick–client software–systems against web–browser–based software–systems.

Chapter 4

The Thick–Client Software–System Architecture of YEROTH–ERP–3.0

4.1 Business and user interface code deployment

Table 3.1 depicts the issue of business and user interface code deployment on all computers participating in the functioning of YEROTH–ERP–3.0, as a software–system for a user.

We tackle the problem of automatic deployment of business and user interface code on all user computers by using the '`apt upgrade`' software–system on '`Debian-Linux`'.

4.2 Co–related software–systems

4.3 User interface

4.4 Number of logical layers

4.5 Software security vulnerabilities

4.5.1 Vulnerability detection

4.5.2 Vulnerability prevention

4.5.3 Vulnerability protection

Chapter 5

Related Software–System Architectures

5.1 Fat–client software–system architecture

5.2 Thin–client software–system architecture

Chapter 6

Conclusion

YEROTH–ERP–3.0 has a thick–client software–system architecture because we found thick–client software–system architectures simpler than web–browser–based software–system architectures.

Thick–client software–system architecture because it requires less layers in its logical software–syste architecture.

Table 3.1 illustrates a thick–client software–system requires less layers in its logical software–system architecture than a web–browser–based software–system.

a web–browser–based software–system potentially entails more present software security vulnerabilities because its implementation requires to use at least 2 different programming languages, and frameworks in combination.

A web–browser–based software–system architecture has more drawbacks as follows:

1) it requires at least 3 other software applications to fully operate (e.g.: DBMS, web server, application server.).
2) A web–browser–based software–system requires at least 4 layers in its logical system architecture (e.g.: client, presentation, logic, and data layers).
3) A web–browser–based software–system potentially possesses more software security vulnerabilities because its implementation requires to use at least 2 different programming languages, and frameworks in combination.

Chapter 7

Bibliography

[Com20] The Qt Company. Qt Designer Manual. http://doc.qt.io/qt-5/qtdesigner-manual.html, 2020. Last accessed on September 4, 2020 at 15:21.

[KM06] Taeho Kgil and Trevor Mudge. Flashcache: A nand flash memory file cache for low power web servers. In *Proceedings of the 2006 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, CASES '06, page 103–112, New York, NY, USA, 2006. Association for Computing Machinery.

[sec20] securityboulevard.com. Thick Client Penetration Testing Methodology. http://securityboulevard.com/2020/02/thick-client-penetration-testing-methodology/, 2020. Last accessed on September 4, 2020 at 15:21.

[WEI20] Yongming WEI. miniStudio User's Guide. http://www.minigui.net/en/ministudio, 2020. Last accessed on September 4, 2020 at 15:21.