Designing complex softwares.

PLAN

[Introduction: 1](#_Toc14786076)

[I. Context and Research problem 2](#_Toc14786077)

[A) Context 2](#_Toc14786078)

[i) AXA Bank Information System Department (ISD) 2](#_Toc14786079)

[ii) The missions within the DSI Laboratory *(DSI Lab’)* 2](#_Toc14786080)

[i) What is Design, develop, complex software? 3](#_Toc14786081)

[ii) Fined grained view and sub categories of the problem 3](#_Toc14786082)

[II. State of the art 4](#_Toc14786083)

[A) Model Driven Architecture (MDA) 4](#_Toc14786084)

[B) Behavior Driven Design (BDD) 4](#_Toc14786085)

[C) Domain Driven Design (DDD) 4](#_Toc14786086)

[III. Research Methodology 4](#_Toc14786087)

[IV. Study 4](#_Toc14786088)

[V. Validation 4](#_Toc14786089)

[VI. Conclusion and further work 4](#_Toc14786090)

[Table of acronyms 4](#_Toc14786091)

# Abstract:

# Introduction:

According to the IT context, a complex software is considered to be related to critical business areas (health, insurance, banking), where business knowledge is held by a small group of industry experts. In addition, the software is designed with a distributed architecture, allowing the deployment and execution of the latter on several platforms and on different environments. Such software is also able to handle traffic of millions of users and run 24/7.

We use complex software platforms on a daily basis (Airbnb, Leboncoin, platforms and mobile applications for banking services) and we cannot ignore how vital they are to us. Complex software is undeniably important, and so are the challenges to developing it. Addressing the issues implied by these challenges is therefore inevitable. Over the past 30 years, we have seen a series of project management methodologies and software design models emerge to solve these issues that are of several types: from project management to software design patterns and technical coding best practices. However, to get to know these practices, you have to read tons of documents, sometimes not clear enough and not very explicit, which can lead to misunderstandings, abandonment of the learning process, misuse and finally facing the same issues that the methodologies where supposed to solve.  
This paper did focus on software designs patterns and technical development best practices with the aim to produce guidelines with concrete step by step methodology.  
Methodologies have been elicited and categorized by the kind of issues they are addressing:  
Design issues, Software implementation related issues, and Methodology concreteness related issues, then thanks to qualitative and quantitative research through semi-structured interviews and questionnaires, feedbacks from stakeholders of complex software projects have been extracted with the most important methodologies and practices in order to provide the guidelines, that have been assessed through a bank loan project at AXA Bank.

# Context and Research problem

## Context

### AXA Bank Information System Department (ISD)

The organization of AXA Bank Information System Department

…

### The missions within the DSI Laboratory *(DSI Lab’)*

Missions:

Conducting, in a scrum based way, applications development for diverse ISD projects.

My missions:

Participating to the designing of java based applicative systems providing customer services.

Achieve experimental tasks about technologies that may be used for future projects.

…

The problems:

AXA needs to build customer loyalty, supply ever better products and services on an ongoing basis. Hence they face the challenge of creating software for critical domains such as loans and mobile banking.

Yet, they are facing poor collaboration between domain experts and development teams, since frequent exchanges happen only on early project steps, there are no specific software architecture, the *big ball of mud* pattern is applied.

Sluggish pace of feature enhancement, buggy releases/Frequent production incidents and Developers firefighting continuously.

Last but not the least, there are a bunch of designing principles out there that should be applied to the good context to prove its usefulness, unfortunately, this decision is not easy for developers, since deciding of a designing principle with respect to the context requires reading and mastering the concepts through years of experiences and tons of documentation.

…

The Research Problem:

How to Design and Develop complex Software in order to tackle the previous problems?

### What is Design, develop, complex software?

Designing is deciding what to build And How to build.

Developing is implementing what have been decided.

A complex Software is characterized with the following characteristics:  
- Related to critical business areas (health, insurance, banking),  
- Business knowledge held by a small group of domain experts.   
- Distributed architecture, allowing the deployment and execution of the software on several platforms and on different environments,  
-Handle traffic of millions of users and run 24/7.

According to these definitions, the problem can be reformulated as following:

How to decide what to build and how to build to provide critical services to thousands of users?

How to build and correct quickly what have been decided?

Following clearly defined and concrete steps?

Still, these statements represent the problem in a coarse grained view that we need to zoom in to get more insights about resolution axis.

### Fined grained view and sub categories of the problem

…

How to emphasize collaboration between domain experts and development teams and fill the gap between both teams?

…

How to help the development teams to focus on domain issues as described by domain experts?

How to separate technical implementation concerns from domain logic issues?

How to apply an architectural style facilitating scalability and features enhancements?

…

How to define concrete steps easy to apply?

With fine grained statements comes out that we are facing three kinds of issues:

* Designing issues
* Technical implementation issues
* Approaches concreteness issues

Getting to know the diverse solutions suggested before now requires analyzing what have been done so far in companies and projects.

# State of the art

## Model Driven Architecture (MDA)

## Behavior Driven Design (BDD)

## Domain Driven Design (DDD)

## 

# Research Methodology

# Study

# Validation

# Conclusion and further work

# Table of acronyms

BDD: Behavior Driven Design

DDD: Domain Driven Design

ISD: Information System Department

MDA: Model Driven Architecture

TDD: Test Driven Design