INTERNATIONAL JOURNAL OF MEDICAL INFORMATICS 83 (2014) 849–859

journal homepage: www.ijmijournal.com

Evaluation of software maintainability with *open*EHR – a comparison of architectures

Koray Atalag[a,*], *Hong Yul Yang*[b], *Ewan Tempero*[c], *James R. Warren*[a,c]

[a] *National Institute for Health Innovation (NIHI), The University of Auckland, Auckland, New Zealand*
[b] *Ocean Informatics Pty. Ltd., Brisbane, Australia*
[c] *Department of Computer Science, The University of Auckland, Auckland, New Zealand*

ARTICLE INFO

Article history:
Received in revised form
15 November 2013
Accepted 29 July 2014

Keywords:
Electronic health records
Software design
Standards
Software maintainability
*open*EHR
Archetypes

ABSTRACT

Purpose: To assess whether it is easier to maintain a clinical information system developed using *open*EHR model driven development versus mainstream methods.
Methods: A new open source application (GastrOS) has been developed following *open*EHR's multi-level modelling approach using .Net/C# based on the same requirements of an existing clinically used application developed using Microsoft Visual Basic and Access database. Almost all the domain knowledge was embedded into the software code and data model in the latter. The same domain knowledge has been expressed as a set of *open*EHR Archetypes in GastrOS. We then introduced eight real-world change requests that had accumulated during live clinical usage, and implemented these in both systems while measuring time for various development tasks and change in software size for each change request.
Results: Overall it took half the time to implement changes in GastrOS. However it was the more difficult application to modify for one change request, suggesting the nature of change is also important. It was not possible to implement changes by modelling only. Comparison of relative measures of time and software size change within each application highlights how architectural differences affected maintainability across change requests.
Conclusions: The use of *open*EHR model driven development can result in better software maintainability. The degree to which *open*EHR affects software maintainability depends on the extent and nature of domain knowledge involved in changes. Although we used relative measures for time and software size, confounding factors could not be totally excluded as a controlled study design was not feasible.

1. Introduction

Software continues to change along the course of its life-cycle, either due to day to day maintenance issues such as bug fixes, or in a rather planned fashion to evolve into a more capable product. Software maintainability ('maintainability' from here on), the ease with which these changes can be made, constitutes the lion's share of software development costs [1,2]. Poor maintainability may also negatively affect the reliability of changed products and delay time to market limiting potential business opportunities [3]. Therefore, even small

* Corresponding author at: National Institute for Health Innovation (NIHI), The University of Auckland, Private Bag 92019, Auckland, New Zealand. Tel.: +64 9 923 7199.
E-mail address: k.atalag@auckland.ac.nz (K. Atalag).
http://dx.doi.org/10.1016/j.ijmedinf.2014.07.006
1386-5056/

improvements in maintainability will imply significant cost savings.

The size and complexity of the biomedical domain and the variability of practice pose great challenges for building and maintaining health information systems (HIS) [4]. Furthermore, rapid advances in the body of knowledge and biomedical technology create a need to modify HIS [5–7]. Girosi et al. report that most long term costs associated with electronic medical record systems are due to maintenance [8].

The mainstream software development approach today is to hardcode domain knowledge into program code and data model. Inevitably, the resulting software has to be altered by developers triggering the costly redevelopment cycle during the maintenance phase [9]. In spite of notable advances in software development technology (such as conceptual modelling, rapid prototyping, integrated development environments) maintenance is still a big issue, and hard-coding domain knowledge into software seems to be a part of essence of the problem [10].

This study leverages the existence of a deployed endoscopic reporting application (called GST) against which we benchmark the maintainability as compared to a newly-developed *open*EHR based application (GastrOS) developed to the same requirements. GST had been used in a real clinical setting between circa 1999 and 2003 at a university hospital endoscopy unit [11]. It was developed using Microsoft Visual Basic version 6 (VB6) and Microsoft Access 2000. The Minimal Standard Terminology for Digestive Endoscopy (MST) provided most of the domain knowledge [12]. This knowledge was embedded into the program code and relational data model. During the course of its lifetime many modifications were needed to meet the changing clinical and business requirements at the facility. However implementing these modifications has not been easy and the cost of maintenance became a significant barrier to its further evolution. Over an extended period of time, and consistently, we observed that the main source of change was related to the domain knowledge.

We have selected the *open*EHR model driven development approach to build a more maintainable application. The main premise of this approach is the ability to separate clinical and technical aspects by way of modelling. Moreover there was outstanding community support and strong direction towards global standardisation [13]. The modelling work using *open*EHR started circa 2002 when the formalism was still in its early stages. A few contributions were made to the formalism in due course to faithfully represent the MST [14].

While we expected improved maintainability at the outset by the use of the model-driven approach there was lack of empirical evidence to support this. The aim of this paper is to present the comparative quantitative maintainability assessment results and related qualitative findings including our implementation methodology.

2. Background and motivation

Gastrointestinal endoscopy is a niche clinical domain with excellent terminology standardisation. MST contains a minimal list of terms and structure to record the results of 99% of routine endoscopic examinations [12]. It depicts a uniform hierarchy to express endoscopic findings and procedures which constitutes the bulk of its content.

Software maintenance comprises activities needed to carry out bug fixes, improvements or adaptation to changes in requirements and technology. Maintainability, on the other hand, is a quality characteristic and defined as the capability of the software product to be modified [15]. A distinction is made about quality from *internal* (design artefacts and code), *external* (characteristics after building software) and *in-use* views (user's view of the quality of software) [16]. In this study we investigated maintainability from an external view by directly observing maintenance activities. *open*EHR defines a model driven approach, called Multi Level Modelling (MLM), where domain knowledge can be expressed in the form of *Archetypes* using the Archetype Definition Language (ADL) [10]. ADL is a compact and declarative programming language with high expressive power focused on health information representation which can easily be understood and modified by domain users. Considered as a formal domain specific language (DSL), Archetypes harness the proven benefits in terms of increased productivity and ease of maintenance [17–21]. They are used to define healthcare concepts together with clinical context, associated meta-data and terminology in a technology agnostic way. Archetypes define all possible data items for a given concept; hence they are *maximal* datasets. Reuse of the same archetype, albeit with a different set of data items at a time, provides consistency which is crucial for interoperability. One level up is the *open*EHR *Templates* which can aggregate, further constrain and annotate a set of archetypes for specific use (e.g. a clinical document such as discharge notes, health summary, referrals or an endoscopy report).

In MLM all data conform to simple and generic technical buildings blocks defined in the *Reference Model* (RM). These are fairly generic and stable technical artefacts used to depict certain characteristics of health information (e.g. data structures and types) and the means to define clinical context to meet ethical, medico-legal and provenance requirements. The RM also defines a uniform health record structure, namely the EHR RM, which depicts where each type of clinical information will be added in a longitudinal record. The RM forms the first level. In the second level archetypes bring together and configure RM building blocks (e.g. defining hierarchy, optionality, repeatability, providing default values, linking to biomedical terminologies). Additional levels exist at terminology, *open*EHR templates and presentation levels. This layering approach helps tackle complexity and also separates clinical and technical concerns.

The motivation of our study was the existence of GST at our disposal and the real-world maintenance experience with which we could benchmark *open*EHR based software development.

The research questions in the study are: 1) is there a difference in maintainability between GastrOS and GST? Rather than a binary answer we are mostly interested in individual differences across different changes if any; 2) did hard-coding domain knowledge into the software render GST more difficult to maintain? The challenge to answering this question is to rule out any difference due to programming language, programmer productivity and the way software is designed; 3) to