Computer Standards & Interfaces 47 (2016) 52–61

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Performance assessment and tuning for exchange of clinical documents cross healthcare enterprises

Cheng-Yi Yang, Chien-Tsai Liu *

Graduate Institute of Biomedical Informatics Taipei Medical University, 250 Wu-Xin Street, Taipei 110, Taiwan

ARTICLE INFO

Article history:
Received 22 October 2015
Received in revised form 28 February 2016
Accepted 5 March 2016
Available online 11 March 2016

Keywords:
Electronic health records (EHRs)
System interoperability
Integrating the healthcare enterprise (IHE)
Cross-enterprise document sharing (XDS.b)
Performance testing
OpenXDS

ABSTRACT

Background: To integrate electronic health records (EHRs) from diverse document sources across healthcare providers, facilities, or medical institutions, the IHE XDS.b profile can be considered as one of the solutions. In this research, we have developed an EHR/OpenXDS system which adopted the OpenXDS, an open source software that complied with the IHE XDS.b profile, and which achieved the EHR interoperability.
Objective: We conducted performance testing to investigate the performance and limitations of this EHR/OpenXDS system.
Methodology: The performance testing was conducted for three use cases, EHR submission, query, and retrieval, based on the IHE XDS.b profile for EHR sharing. In addition, we also monitored the depletion of hardware resources (including the CPU usage, memory usage, and network usage) during the test cases execution to detect more details of the EHR/OpenXDS system's limitations.
Results: In this EHR/OpenXDS system, the maximum affordable workload of the EHR submissions were 400 EHR submissions per hour, the DSA CPU usage was 20%, memory usage was 1380 MB, the network usages were 0.286 KB input and 7.58 KB output per minute; the DPA CPU usage was 1%, memory usage was 1770 MB, the network usages were 7.75 KB input and 1.54 KB output per minute; the DGA CPU usage was 24%, memory usage was 2130 MB, the network usages were 1.3 KB input and 0.174 KB output per minute. The maximum affordable workload of the EHR queries were 600 EHR queries per hour, the DCA CPU usage was 66%, the memory usage was 1660 MB, the network usages were 0.230 KB input and 0.251 KB output per minute; the DGA CPU usage was 1%, the memory usage was 1890 MB, the network usages were 0.273 KB input and 0.22 KB output per minute. The maximum affordable workload of the EHR retrievals were 2000 EHR retrievals, the DCA CPU usage was 79%, the memory usage was 1730 MB, the network usages were 19.55 KB input and 1.12 KB output per minute; the DPA CPU usage was 3.75%, the memory usage was 2310 MB, and the network usages were 0.956 KB input and 19.57 KB output per minute.
Discussion and conclusion: From the research results, we suggest that future implementers who deployed the EHR/OpenXDS system should consider the following aspects. First, to ensure how many service volumes would be provided in the environment and then to adjust the hardware resources. Second, the IHE XDS.b profile is adopted by the SOAP (Simple Object Access Protocol) web service, it might then move onto the Restful (representational state transfer) web service which is more efficient than the SOAP web service. Third, the concurrency process ability should be added in the OpenXDS source code to improve the hardware usage more efficiently while processing the ITI-42, ITI-18, and ITI-43 transactions. Four, this research suggests that the work should continue on adjusting the memory usage for the modules of the OpenXDS thereby using the memory resource more efficiently, e.g., the memory configuration of the JVM (Java Virtual Machine), Apache Tomcat, and Apache Axis2. Fifth, to consider if the hardware monitoring would be required in the implementing environment. These research results provided some test figures to refer to, and it also gave some tuning suggestions and future works to continue improving the performance of the OpenXDS.

1. Introduction

Population aging and chronic disease have gradually grown throughout the world [1]. Electronic health records (EHRs) are integrated from different document sources across healthcare providers, hospitals, and/or medical institutions, and can be used to enhance disease management, facilitate communications, and improve the quality of care [1–3]. Naturally, the sharing and exchanging of the EHRs across healthcare facilities and institutions in different geographical regions are now attracting a lot of attention [4]. Integrating the healthcare enterprise (IHE) [5] is promoted by healthcare professionals and

* Corresponding author at: 250 WuXing Street, Taipei 110, Taiwan.
E-mail address: ctliu@tmu.edu.tw (C.-T. Liu).

http://dx.doi.org/10.1016/j.csi.2016.03.001
0920-5489/

industries to improve the way computer systems in healthcare services share information through organizing the established standards, such as the DICOM and HL7, to comprehend specific clinical needs in supporting optimal patient care [5]. One of the IHE profiles, the cross-enterprise document sharing profile (XDS.b) [6], defines the generic architecture on how to share clinical information, provide a standards-based specification for managing the sharing of patient's EHRs between health enterprises, such as hospitals, physician's offices, and clinics. There are many third party open source systems released for implementation of the IHE XDS.b profile. Bruno [7] surveyed nine IHE XDS related open source systems, and found that the OpenXDS system, which is a server side implementation of the IHE XDS.b profile, provides the capacity of good implementation and easy deployment [8]. The OpenXDS has been widely used as a solution in many EHR integrating projects [9]. Several technical and consultant reports also suggested that the OpenXDS could be a component to use in the construction of the Health Information Exchange (HIE) related products [10], and an open source tool to address interoperability framework implementation needs [11]. In this research, we built the EHR system which adopted the OpenXDS as a component to achieve the IHE XDS.b based clinical document sharing. Due to the need for the exchange of the EHRs across health care facilities and/or institutions during their patient encounters, the high performance online transaction processing (OLTP) systems are critical. Therefore, it is very important to know the system performance and service quality of the EHR system based on the OpenXDS framework, or by the abbreviated name of EHR/OpenXDS system. However, there were only a few studies that have investigated the performance issues of the OpenXDS system for exchanges of the EHRs across different systems. Therefore, this research mainly focused on the evaluation of the performance and service quality of the EHR/OpenXDS systems and provided recommendations for system tuning and implementation strategies based on the evaluation.

2. Related work

Many studies have indicated that one of the most important factors of patients' satisfaction to the healthcare service is the waiting time [12–14]. Therefore, an inefficient healthcare information system prolongs the waiting period and thereby decreases the patients' satisfaction of the healthcare service. Moreover, an efficient healthcare information system cannot only improve the physicians' work efficiency, but also increases the physicians' acceptance of the healthcare information system. To understand the performance of the healthcare information system before being adopted into the healthcare environment is an important and required work. Thus, in this section, we review the performance testing of the web application for a healthcare information system, and the performance is then used to verify if the target system is fulfilling the specific performance goals from the users [15,16]. Generally speaking, at the beginning of the performance testing, the researchers will set up a benchmark for performance objectives to reflect the users' expectations and needs.

Performance failures happened due to the lack of performance considerations in the early stages of the system's development, and thereby instigated the system to be redesigned, and caused the cost overruns [16]. There are a lot of performance testing methods, such as implementing the Software Performance Engineering (SPE) methodology to develop the system, and then run the testing to achieve the performance goals [16]. To assess the diverse software systems, such as the web application system, the cloud based application, etc., some studies have developed different performance testing methods for these specific software systems [17–23]. However, this is mostly based on the load and stress testing. The goal of load testing is usually conducted to validate a systems reaction under both normal and anticipated peak load conditions. It is also used to distinguish the system service quality under certain workloads. The web application load testing helps to identify the maximum operating capacity of an application as well as any bottlenecks that might interfere when operating at capacity [23]. Stress testing is able to determine or validate the performance characteristics, or the behavior of the system, when subjected to either conditions beyond normal or peak loading conditions. Stress testing is also able to reveal the application bugs that surface only under high load conditions, identify the application's weak points, evaluate an application's behavior when it is pushed beyond normal or peak load conditions, and shows how the application behaves under extreme load conditions [22,23].

Many studies have used several performance objectives to describe the system performance characteristics, that include 'request–response time', 'throughput', 'hardware resource utilization', 'workload', and 'scalability' [20,24,25]. For example, a number of systematic performance testing approaches focus on the observation of the 'request–response time', 'hardware resources utilization', and 'bottleneck' under certain workloads. Therefore, the researchers can use these designed performance testing models to collect the testing results and then compare them with the performance goals. The study [16] states that the 'request–response time' estimation is a basic factor to validate the performance objective. For the web application system performance, many studies believe that the end-user web system quality awareness, also referred to as the web system QoE (Quality of Experience), is able to assess through the 'request–response time' [26,27]. If the request–response time of a web application system is inadequate, users may lose interest even if the function of the web application is correct [20]. Many scholars believe that the longer system waiting time caused dissatisfaction of the web application system, and the increased time the users have to wait for the web page to arrive (or transactions to be completed), the more dissatisfied they tended to become with the service [26,28]. As indicated in the previous studies [26,28,29], the request–response time of a web application system commonly accepted by web users is up to 15 s. While the web application system request–response time that exceeds 15 s can impede productivity, it may account for the longer waiting time to not only affect the satisfaction of the user to the web application system, but also decrease the staff's work efficiency and thereby affect the company's operations. Nonetheless, other studies have also proposed a different ideal request–response time, such as the three kinds of response-time limits [30], two seconds [26], etc. However, whilst we considered that the mechanism of the cross enterprise EHR exchange is complex, the performance goal of the request–response time of the EHR/OpenXDS system being set to 15 s will be more appropriate.

3. The framework of EHR system based on the IHE XDS.b profile

3.1. IHE Cross-enterprise document sharing profile (IHE XDS.b)

The IHE XDS.b profile constrains the OASIS ebXML 3.0 standard in order to build a system to exchange the EHRs through the web services [31]. Referring to Fig. 1, there are five actors and five transactions in the XDS.b profile. The Document Source Actor (DSA) submits a document set (with an episode of the EHRs) to the Document Repository Actor (DPA) through an ITI-41 transaction [32]. Once the DPA receives and processes the ITI-41 transaction, it registers the metadata of the episode of the EHRs to the Document Registry Actor (DGA) through an ITI-42 transaction [32]. The ITI-41 and ITI-42 transactions define how to embed the EHRs into the IHE XDS.b metadata message format and then transfer through the web service [6]. The Document Consumer Actor (DCA) supports two activities, one is the query on the metadata of the EHRs in the DGA through the ITI-18 transaction, and the other is the retrieval of the EHRs from the DPA through the ITI-43 transaction. When retrieving an episode of a patient's EHRs stored in the DPA, the DCA must query the DGA to find where the EHR is stored (the metadata) by initiating the ITI-18 first, and then by using the returned link to retrieve the EHR through the ITI-43. Finally, both the ITI-8 and the ITI-44 transactions are used to communicate the patients' identification