ISSN-E: 2542-3401, ISSN-P: 1316-4821

Universidad, Ciencia y Tecnología,
Vol. 26, Núm. 117, (pp. 24-32)

https://doi.org/10.47460/uct.v26i117.657

Comparative analysis of the efficiency of Data Persistence Frameworks. Case study Hibernate 5.4 and MyBatis 3.4

Yolvis Luis Mendoza-Palomino
https://orcid.org/0000-0002-6948-5558
yolvis.mendoza.27@unsch.edu.pe
Qualifacts
Lima, Perú

Manuel Avelino Lagos-Barzola
https://orcid.org/0000-0001-8078-755X
manuel.lagos@unsch.edu.pe
Universidad Nacional de
San Cristóbal de Huamanga
Ayacucho, Perú

Efraín Elías Porras Flores
https://orcid.org/0000-0002-5560-0308
efrain.porras@unsch.edu.pe
Universidad Nacional de San Cristóbal de Huamanga
Ayacucho, Perú

Recibido (29/07/2022), Aceptado (01/10/2022)

Abstract. - The objective of the study was to determine whether Hibernate 5.4 or MyBatis 3.4 is the most efficient in terms of time, RAM, and CPU usage. Two REST microservices were implemented to perform CRUD operations; which were accessed by REST services developed with Hibernate and MyBatis invoked from a client. Our results were that Hibernate was the most efficient with respect to processing time at 509.1% and 47.7% for select and delete queries, while Mybatis was at 12.7% and 14.2% for update and insert queries, correspondingly. Regarding the use of RAM and CPU, MyBatis was determined to be the most efficient at 55.2% and 20% respectively.

Keywords: Data persistence framework, comparative analysis, Hibernate, MyBatis.

Análisis comparativo de la eficiencia de Frameworks de persistencia de datos, caso de estudio Hibernate 5.4 y MyBatis 3.4

Resumen. - El objetivo del estudio fue determinar si Hibernate 5.4 o MyBatis 3.4 es el más eficiente en referencia al uso del tiempo, la RAM y la CPU. Se implementó dos microservicios REST para realizar operaciones CRUD; los cuales fueron accedidos por servicios REST desarrollados con Hibernate y MyBatis invocados desde un cliente. Nuestros resultados fueron que Hibernate fue el más eficiente respecto al tiempo de procesamiento en un 509.1% y 47.7% para consultas de selección y eliminación, mientras que Mybatis en un 12.7% y 14.2% para consultas de actualización e inserción, correspondientemente. Con respecto al uso de la RAM y la CPU se determinó que MyBatis fue el más eficiente en un 55.2% y 20% respectivamente.

Palabras clave: Framework de persistencia de datos, análisis comparativo, Hibernate, MyBatis.

I. INTRODUCTION

Given the wide variety of data persistence frameworks, usually software developers faced the dilemma of which framework to use in order to achieve their objectives in terms of software implementation, the shortest time, and the highest quality. The study in [1] highlights and describes that the object-relational mapping (ORM) solution is available in many popular platforms, that offer useful tools allowing developers to be concerned about other pending issues.

The work in [2] indicates that among of the most relevant aspects when choosing a framework are speed and efficiency, which may have many issues due to the diversity of metrics to measure the performance of a framework. However, there are other aspects that could help to decide, such as the response speed, the integration with the database, the use of the CPU, and the use of RAM.

The study [3], concerned about data processing time, reported that the Hibernate framework slightly increases the processing times when compared to the MyBatis framework, in an experimentation query automatically generated and variation of caching algorithms. Similarly, the work in [4] compared Hibernate against MyBatis, concluding that Hibernate outperforms MyBatis in terms of available documentation, quality, robustness, and easy learning curve, which makes it ideal for developing software projects.

Because the Hibernate and MyBatis frameworks are widely used by software developers in Java language programming; it is desirable to carry out an evaluation of the efficiency of both frameworks aiming to establish which could be selected as a framework for a software development project. Although previous studies tried to establish which is the best one, there is a lack of a deep study in terms of time, RAM, and CPU usage. In this work, we provide a study to determine whether the Hibernate 5.4 and the MyBatis 3.4 frameworks is the best one for software development.

II. DEVELOPMENT

A. Object-relational mapping

The study [5] established that object-relational mapping (ORM) is a technique that allows converting system data used in an object-oriented programming language, into data used by a relational database. Thereby, ORM creates a virtual object-oriented database over a relational one. Thus, allowing the use of object-oriented features such as inheritance and polymorphism.

The work [6] indicated that ORM refers to the automatic persistence of objects from a Java application to tables in a relational database and that the mapping between the objects and the database is described in metadata, which is used to reverse effects. Despite this advantage, a considerable aspect to take into account about the use of this technique is the compromise of the performance of applications.

Another study from a developer perspective [7], concluded that ORM encompasses solutions for mapping business objects to relational data, delegating to the persistence layer the details of its persistence. However, ORM is not fully automatic, and that automation is indescribable.

B. Data persistence

Data persistence can be understood as the maintenance of the state of the object as well as its class, not reducing it to any particular program [8]. In a more specific sense, data persistence is the ability of a computer to retain information even after shutting down or closing the program that uses that information [8],[9].

C. Framework

The term framework is used in many contexts [11], such as applications in medicine, computer vision, games, and among others, in which it is used to describe a software structure consisting of adaptable and interchangeable components for developing applications. In other words, a framework can be thought of as something incomplete, but configurable generic application by means of adding complements.

D.Hibernate

Hibernate is a framework that provides a complete solution to the problem of data persistence in Java. By interacting between the application and the relational database, Hibernate allows developers to focus on the business problem. Hibernate also enables us to follow a lot of design rules and specific patterns when creating persistence classes and business logic. Moreover, Hibernate furnishes high integration with most new and existing applications without the need to make major changes to other parts of the application [6]. Finally, the framework is an open-source Data Persistence Framework, easily combined with HQL and SQL software [12].

E. MyBatis

Mybatis is another persistence framework that supports SQL, stored procedures, and, advanced mappings. MyBatis does not require JDBC code, manual parameter setting, and retrieval of results. MyBatis can be configured with XML, and annotations, and allows maps and POJOs (Plain Old Java Objects) to be mapped to database records [13],[14].

F. Efficiency

Efficiency is defined as the characteristic that would allow evaluation of the relationship between the level of performance of software and the number of resources used. The aspects to be evaluated would be the time behavior related to adequate response and processing times, that is to say, the performance when executing its function under specific conditions and, on the other hand, the resource behavior, related to the capacity of the software to use suitable amounts and types of resources, also operating under specific conditions [15].

G. Aspect-Oriented Programming

The Aspect Oriented Programming (AOP) is a methodology that provides separation of crosscutting concerns by introducing an Aspect, a new unit of modularization. Each aspect focuses on a particular crosscutting function. Main classes are no longer plagued with crosscutting concerns. The aspect weaver assembles the final system by combining main classes and crosscutting aspects through a process called weaving. Therefore, with the use of the methodology, applications that are easy to design, implement and maintain would be created.

III. METHODOLOGY

A. About the data used

In this work, we use the Employee data table of the Oracle HR scheme. It consists of 10,000 records for tests. Four basic queries: selection, insertion, update, and elimination, are performed.

B. Employees table DDL

```
CREATE TABLE "HR"."EMPLOYEES"
  (
     "EMPLOYEE_ID" NUMBER(6,0),
     "FIRST_NAME" VARCHAR2(20 BYTE),
     "LAST_NAME" VARCHAR2(25 BYTE) NOT NULL ENABLE,
     "EMAIL" VARCHAR2(25 BYTE) NOT NULL ENABLE,
     "PHONE_NUMBER" VARCHAR2(20 BYTE),
     "HIRE_DATE" DATE NOT NULL ENABLE,
     "JOB_ID" VARCHAR2(10 BYTE) NOT NULL ENABLE,
     "SALARY" NUMBER(8,2),
     "COMMISSION_PCT" NUMBER(2,2),
     "MANAGER_ID" NUMBER(6,0),
     "DEPARTMENT_ID" NUMBER(4,0),
      CONSTRAINT "EMP_EMAIL_UK" UNIQUE ("EMAIL")
   )
```

C. About the architecture of the developed system

First, we developed three applications to perform basic CRUD queries. Fig 1. Illustrates the proposed system architecture of the application. The modules are outlined as follows:

- The rest-client: Implements a REST client and through threads, we send a hundred requests simultaneously; fifty of which are directed towards a REST service of hr-rest-MyBatis and the other fifty towards a REST of hr-rest-hibernate, in this way we subject our applications to a stressful situation since each REST service will have to process fifty requests simultaneously.
- The hr-rest-mybatis: microservice implemented with MyBatis. This microservice exposes four REST services.

The hr-rest-hibernate: microservice implemented with Hibernate. This microservice also exposes four REST services.

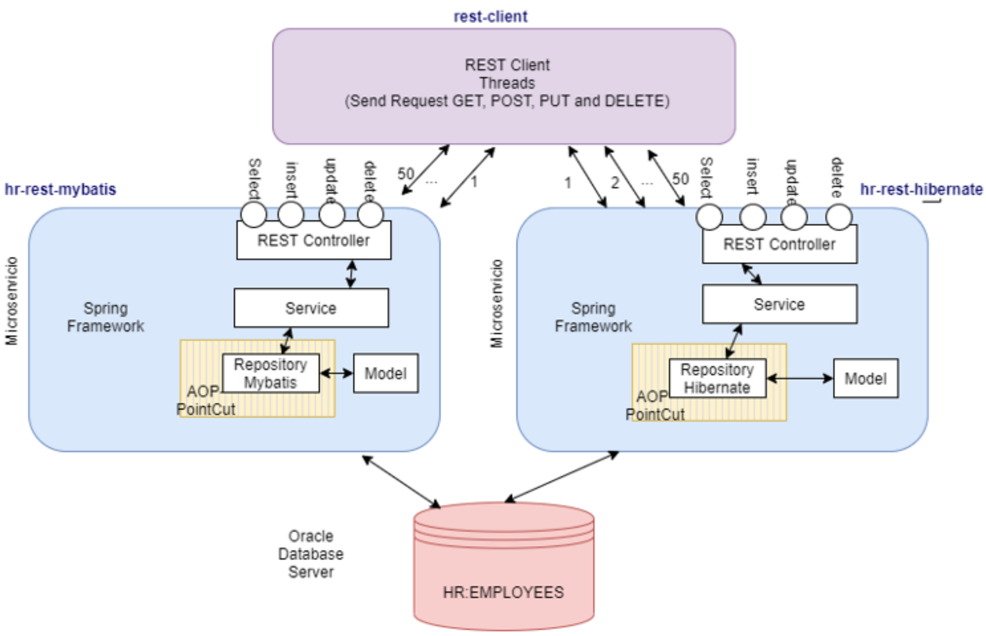

Fig. 1. System architecture for our experiments

Next, Table 1, Table 2, Table 3, and Table 4, summarize the details of the REST services implemented in hr-rest-mybatis and hr-rest-hibernate.

Table 1. Detail of the rest service for the selection of a record from the Employee table.

Name	Select
URL	http://{domain}:{port}/employee/{id}
HTTP method	GET
Parameters	[None]
Description	Obtains the data of a record from the EMPLOYEES table by the identifier of the record that we obtain from the value of {id} in the URL

Table 2. Detail of the REST service for the update of a record in the Employee table.

Name	Update
URL	http://{domain}:{port}/employee/{id}
HTTP method	PUT

Parameters	{
	"employeeId": <anyLongValue>,
	"departmentId": <anyLongValue>,
	"managerId": <anyLongValue>,
	"jobId": <anyStringValue>,
	"firstName": <anyStringValue>,
	"lastName": <anyStringValue>,
	"email": <anyStringValue>,
	"phoneNumber": <anyStringValue>,
	"hireDate": <anyDateValue>,
	"salary": <anyDecimalValue>,
	"commissionPct": <anyDecimalValue>
	}

Description	Updates a record in the EMPLOYEES table with the submitted data.

Table 3. Detail of the REST service for inserting a record in the Employee table.

Name	Insert
URL	http://{domain}:{port}/employee
HTTP method	POST

Parameters	{
	"departmentId": <anyLongValue>,
	"managerId": <anyLongValue>,
	"jobId": <anyStringValue>,
	"firstName": <anyStringValue>,
	"lastName": <anyStringValue>,
	"email": <anyStringValue>,
	"phoneNumber":<anyStringValue>,
	"hireDate": <anyDateValue>,
	"salary": <anyDecimalValue>,
	"commissionPct":<anyDecimalValue>
	}

Description	Inserts a record in the EMPLOYEES table with the submitted data.

Table 4. Detail of the REST service for deleting a record in the Employee table.

Name	Delete
URL	http://{domain}:{port}/employee/{id}
HTTP method	DELETE
Parameters	[None]
Description	Deletes a record from the EMPLOYEES table for the record identifier that we get from the {id} value in the URL

D. Determining the processing time of different types of queries

To obtain this information, once the microservices developed with Hibernate and MyBatis were deployed, we sent fifty Requests to each of them simultaneously, and once the Requests were received through the REST services, using AOP, we obtained the time difference captured before and after the execution of the method defined in EmployeeDAO.

E. Determining the RAM usage

The built-in Task Manager tool in the Windows operating system was used; We proceeded to open the "Resource Monitor" window and went to the "General Information" tab to obtain the information from the "Workspace (KB)" column of the "Memory" section, which shows the amount of physical memory used in real-time by a process.

F. Determining the CPU Usage

In the same way, the Task Manager of the Windows operating system was used; The "Resource Monitor" section was accessed, the "General Information" section and the information were obtained from the "CPU" column of the "CPU" section, which shows the percentage of CPU usage by a process.

IV. RESULTS

A. Average processing time using Hibernate

Table 5 shows the achieved processing time with Hibernate.

Table 5. Average processing time of the Hibernate framework.

Processing time for each type of query (MS)			
Select	Update	Insert	Delete
97.0	539.4	650.3	348.9

B. Average processing time using MyBatis

Table 6 outlines the achieved processing time with MyBatis.

Table 6. Table 1 .Average processing times of the MyBatis framework.

Processing time for each type of query (MS)			
Select	Update	Insert	Delete
591.1	470.6	558.2	515.4

C. Average processing time obtained with Hibernate and MyBatis
Table 7 presents the average time of both frameworks for comparison.

Table 7. Hibernate and MyBatis Average Processing Time Summary

Framework	Average processing time for each type of query (MS)			
	Select	Update	Insert	Delete
Hibernate	97.0	539.4	650.3	348.9
MyBatis	591,1	470,6	558,2	515,4
Gap	-509,1 %	12,7%	14,2%	-47,7%

D. RAM usage using Hibernate and MyBatis
Table 8 summarizes the RAM usage of both frameworks.

Table 8. Table 1 .RAM usage using Hibernate and MyBatis for each type of queries.

Framework	RAM usage* for each type of query (KB)			
	Select	Update	Insert	Delete
Hibernate	70556	56056	57560	69876
MyBatis	26328	19112	39620	27252
Gap	62.7%	65.9%	31.2%	61.0%

Note: * Results were obtained from the difference between maximum and minimum RAM used by a process

E. CPU usage using Hibernate and MyBatis
Table 9 sums up the CPU usageof frameworks.

Table 9. CPU usage when running Hibernate and Mybatis rest services for each type of query.

Framework	CPU usage for each type of query (%)			
	Select	Update	Insert	Delete
Hibernate	36%	35%	34%	41%
MyBatis	13%	10%	25%	18%
Gap	23%	25%	9%	23%

CONCLUSIONS

Based on the performed experiments, we can conclude that Hibernate is more efficient against MyBatis by 509.1% and 47.7% for select and delete queries, respectively. However, we found that MyBatis is more efficient by 12.7% and 14.2% for update and insert queries.

Taking the RAM on the other hand, it was determined that MyBatis was more efficient regarding the use of RAM in general by 55.2% compared to Hibernate, specifically by 62.7%, 65.9%, 31.2%, and 61.0% for selection queries update, insertion, and deletion of data respectively.

We found that MyBatis is more efficient with respect to the use of the CPU by 20% compared to Hibernate, specifically by 23%, 25%, 9%, and 23%, for select, update, insert, and delete queries, respectively.

REFERENCES

[1] A. Torres, R. Galante, M. S. Pimenta, and A. J. B. Martins, "Twenty years of object-relational mapping: A survey on patterns, solutions, and their implications on application design," Inf. Softw. Technol., vol. 82, pp. 1–18, 2017, doi: 10.1016/j.infsof.2016.09.009.

[2] U.Hernández, "Cómo elegir un framework para el backend," códigofacilito, 2018. https://codigofacilito.com/articulos/elegir-framework-backend#! (accessed Apr. 15, 2022).

[3] S. M. Thampi, "Performance Comparison of Persistence Frameworks," CoRR, 2007, [Online]. Available: http://arxiv.org/abs/0710.1404

[4] J. Gavilanes, "Estudio comparativo de la productividad entre los Frameworks de persistencia en Java Hibernate y Mybatis, aplicado al Sistema de Evaluación Docente del IPEC," 2016.

[5] O. Y. Enriquez and H. Gracia, "Mapeo Objeto / Relacional (ORM) .," Rev. Telem@tica, vol. 10, no. 3, pp. 1–7, 2011.

[6] C. Bauer and G. King, Hibernate in Action, vol. 4, no. Part I. 2005. [Online]. Available: http://books.google.com/books?id=WCmSQgAACAAJ&pgis=1.

[7] J. W. Yoder and R. E. Johnson, "Connecting business objects to relational databases," Urbana, 2005, [Online]. Available: http://test.joeyoder.com/Research/objectmappings/Persista.pdf%5Cnpapers2://publication/uuid/F7960186-CF74-482F-ADF4-86EE560C47D9.

[8] E. Reinosa, C. Maldonado, R. Muñoz, L. Damiano, and M. Abrutsky, Base de datos, 1st ed. Buenos Aires: Alfaomega, 2012.

[9] J. Maldonado, "Persistencia de Datos," Universidad de Cuenca. http://docplayer.es/111524351-Capitulo-8-persistencia-de-datos.html (accessed Apr. 20, 2022).

[10] L. You et al., "JDap: Supporting in-memory data persistence in javascript using Intel's PMDK," J. Syst. Archit., vol. 101, no. June, p. 101662, 2019, doi: 10.1016/j.sysarc.2019.101662.

[11] J.J. Gutiérrez, "¿Qué es un framework?," Available http//www. lsi. us. es/~ javierj/investigacion_ficheros/Framework. pdf Accessed May, vol. 12, pp. 1–4, 2014, [Online]. Available: http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

[12] E. N. Navarrete and R. Navarro, "Desarrollo de un marco de trabajo (framework) para el desarrollo de aplicaciones web en la Universidad Nacional de Costa Rica," p. 12, 2017, [Online]. Available: http://documentas.redclara.net/bitstream/10786/1287/1/131-17-4Desarrollo de un marco de trabajo %28framework%29 para el desarrollo de aplicaciones web.pdf.

[13] MyBatis, "MyBatis – MyBatis 3 | Introducción," 2021. https://mybatis.org/mybatis-3/es/ (accessed Apr. 15, 2022).

[14] Y. Z. Li, S. Gao, J. Pan, B. F. Guo, and P. F. Xie, "Research and application of template engine for web back-end based on Mybatis-PLUS," Procedia Comput. Sci., vol. 166, pp. 206–212, 2020, doi: 10.1016/j.procs.2020.02.052.

[15] M. A. Abud Figueroa, "Calidad en la Industria del Software . La Norma ISO-25000," Calid. en la Ind. del Software. La Norma ISO-9126, p. 255, 2012, [Online]. Available: javier8a.com/itc/bd1/Normas iso 9126.pdf.

LOS AUTORES

Yolvis L. Mendoza Palomino is a Bachelor of Systems Engineering from the National University of San Cristóbal de Huamanga. He is also a Systems Engineer graduate from the same University. He currently works on large-scale software development projects. His research interests basically include software engineering.

Manuel A. Lagos Barzola has a Bachelor's degree in Computer Engineering from the National University of San Cristóbal de Huamanga, Peru, 2006. He also graduated as a Computer Engineer from the same university in 2010. Master in Computer Engineering in 2017 from the Ricardo Palma University, Peru. Doctor in Educational Sciences from the National University Enrique Guzmán y Valle, Peru, 2020.

Efraín E. Porras Flores has a Bachelor of Science in Mechanical Engineering from the National University of Engineering, Peru, in 1981. He also graduated as a Mechanical Engineer from the same university in 1991. Master of Science in Systems Engineering from the same university in 2010. Doctor in Systems Engineering from the Universidad Nacional Federico Villarreal, Peru, 2019.