Ranking Enterprise Application Integration Platforms from a Performance Perspective: an Experience Report

Preprint *in* Software Practice and Experience · January 2019

DOI: 10.1002/spe.2679

CITATIONS
5

READS
584

3 authors:

Daniela Lopes Freire
University of São Paulo
26 PUBLICATIONS **66** CITATIONS
SEE PROFILE

Rafael Z. Frantz
Universidade Regional do Noroeste do Estado Estado do Rio Grande do Sul (UNIJUÍ)
118 PUBLICATIONS **504** CITATIONS
SEE PROFILE

Fabricia Roos-Frantz
Universidade Regional do Noroeste do Estado Estado do Rio Grande do Sul (UNIJUÍ)
91 PUBLICATIONS **424** CITATIONS
SEE PROFILE

EXPERIENCE REPORT

Ranking Enterprise Application Integration Platforms from a Performance Perspective: an Experience Report

Daniela L. Freire* | Rafael Z. Frantz | Fabricia Roos-Frantz

[1]Department of Exact Sciences and Engineering, Unijuí University, RS, Brazil

Correspondence
*Corresponding Daniela L. Freire.
Email: dsellaro@unijui.edu.br

Summary

Enterprises turn to their software applications to support their business processes. Over time, it is common for a company to end up with a wide range of applications, which are usually developed in-house by its IT department or purchased from third party specialised software companies. The result is a heterogeneous software ecosystem with applications developed in different technologies and frequently using different data models, which brings challenges when two or more applications have to collaborate to support a business process. Integration platforms are specialised software tools that help to design, implement, run, and monitor integration solutions that orchestrate a set of applications. The run-time system is the component of integration platforms responsible for running integration solutions, which makes its performance a critically important issue. In this article, we report our experience in evaluating and comparing four well-known open-source integration platforms in the context of a research project where performance was a central requirement to choose an integration platform. The evaluation was conducted using a decision-making methodology to build a ranking of candidate platforms by means of subjective and objective criteria. The subjective evaluation takes into account experts preferences and compares integration platforms using the Analytic Hierarchy Process, which has been used in many applications related with decision-making. The objective evaluation is build on top of properties distributed on three dimensions: message processing, hotspot detection, and fairness execution, which compose the research methodology we used. The evaluated platforms were ranked to identify the one with best performance amongst them.

KEYWORDS:
enterprise application integration, integration platform, multiple-criteria decision making, integration patterns, integration framework, orchestration engine, message-based middleware, performance evaluation.

1 | INTRODUCTION

Enterprises aim at improving their business processes by increasing the support of software applications. The software ecosystem[1] of a company is built over time and includes applications developed in-house as well as purchased from third-party software development enterprises. Recently, these software ecosystems have also included applications provided as services

[0]**Abbreviations:** AHP, analytic hierarchy process; EAI, enterprise application integration; EIP, enterprise integration patterns

by different providers in the cloud[2]. The result is a heterogeneous software ecosystem, in which applications are expected to work together and collaborate by exchanging data and sharing functionalities to support a business process. However, not all applications are designed and endowed with features to collaborate.

Integration platforms are specialised software tools that help to design, implement, run, and monitor integration solutions that orchestrate a set of applications so as to keep their data synchronised or to develop new functionalities on top of the current ones; ideally, such solutions should not have any impact on the applications[3]. In the last decade, open-source integration platforms[4,5,6,7,8] have emerged, inspired by the pipes-and-filters architectural style[9] and the conceptual integration patterns documented by Hohpe and Woolf[10]. In an integration solution, the filters are implemented by means of tasks each of which supports an integration pattern; and the pipes are implemented as communication channels through which data flows, wrapped in messages. The integration patterns document a set of best-practices to implement tasks which help to solve recurrent application integration problems. Among the elements that make up an integration platform, the run-time system is the one responsible for running integration solutions, which makes its performance crucial[11,12,13]. From now on, we assume that one run-time system is better than another if it consumes less computational resources and is able to process more messages per unit of time. Integration Platform as a Service (iPaaS) represents a solution especially for small and medium enterprises which need to integrate their business processes because such services decrease their concern with the maintenance costs and operations of the integration platforms on-premises[14].

Three dimensions should be considered when making a decision on the most appropriate integration platform to be used: message processing, hotspot detection, and fairness execution. In message processing, the concern is with improving the efficiency of the run-time system to process a message; in hotspot detection, the focus is on the detection of tasks that may represent a bottleneck within the integration solution; and, fairness execution deals with the minimisation of the average time messages take to be processed by a fair assignment of computational resources to tasks. In our review of the literature, we have identified few proposals that evaluate and compare integration platforms. The majority of these proposals focus their evaluation and comparison on market aspects like company maturity, customer relationships, or innovation, and technology aspects like the availability of connectors, security, monitoring, and governance of integration processes.

In this article, we report on our experience in evaluating and comparing four well-known open-source integration platforms that represent the state-of-the-art in the development of message-based integration solutions, namely: Mule[5], Camel[4], WSO2[6] and Guaraná[8,7]. These platforms follow the pipes-and-filters architectural style and support several integration patterns documented by Hohpe and Woolf[10]. The evaluation of the integration platforms is based on objective and subjective criteria. The objective evaluation is build on top of message processing, hotspot detection, and fairness execution-related properties that integrate the research methodology we used. The subjective evaluation takes into account experts preferences. The subjective evaluation of the integration platforms is based on the Analytic Hierarchy Process (AHP)[15], which has been used in many applications related with decision-making[16]. AHP has a strong mathematical and psychological background to organise and analyse complex decisions and help to perform human judgements and choice[17]. The experience proved that it is possible to evaluate platforms based on objective criteria through the inference of property values, and subjective criteria through the knowledge and experience of using such platforms. At end of the evaluation, amongst the compared integration platforms, Camel demonstrates to be the best of them regarding performance, because it is the most endowed with appropriated values for properties, which support the objective criteria and also it is the one that received the best rating in the subjective judgement.

This article is the result of several years of experience on the development of integration projects in real-world software ecosystems, in which performance is a key feature. Although we have ranked these four integration platforms, the same research methodology can be applied to evaluate and compare any other integration platform. Please refer to a recent article (see Freire et. al[18]) in which we provide a survey with a complete list of integration platforms. The rest of this article is organised as follows: Section 2 discusses the related work regarding to evaluation and comparison of integration platforms; Section 3 provides background information on multiple criteria decision making and on the AHP method, in which the research methodology used in this article is reasoned; Section 4 introduces the research methodology that we use to evaluate and compare the integration platforms; Section 5 reports our experience in evaluation and ranking of integration platforms; and, Section 6 presents our conclusions.

2 | RELATED WORK

In this section, we discuss related works that report experiences or approach subjects, which intersect the research about performance analysis, comparison or ranking of integration platforms. Corchuelo et al.[19] analysed five integration platforms, namely: Camel, Mule, ServiceMix, Spring Integration, and BizTalk, approaching properties regarding platform independence, usability, easiness of programming, and maintainability. They divided the properties in three groups: scope of the tool, modelling capabilities, and technical features. Scope of the tool deals with properties that the authors consider as essential. Modelling capabilities deal with important but not essential properties; according to the authors, the absence of such properties makes integration modelling more complex and less intuitive. Technical features address properties that affect the ease of programming, performance, or management of integration solutions. Their work differs from ours, mainly because it does not focus on performance, does not provide a ranking for the integration platforms nor a decision-making method to compare them. Tan et al.[20] applied the environment-based design methodology to deal with the design of integration solutions. In their methodology, a design problem is implied in a product system and is composed of three parts: the environment of the product project, the requirements on product structure, and the requirements on the performance of the product project. Their methodology aims to generate and refine the design specifications and design solution. It is composed of steps, which are realised progressively and simultaneously: environment analysis, conflict identification, and solution generation. Environment analysis identifies the key environment components and the relationships amongst them. Conflict identification identifies conflicts within the environment components relationships. Solution generation resolves environment conflicts proposing a design solution. This methodology is a recursive process, which is repeated until the conflicts are eliminated. Whereas their work addresses the design of integration solutions, our work addresses the performance of execution of integration solutions. Vishal and Bartere[21] reported on their experiences using Amazon SQS and Boomi integration platforms, along with pros and cons of each of them, noticing features that impact productivity, scalability, elasticity, reduced cycle time and baked in quality of IT teams. The authors suggest using Boomi, in cases in which lowering costs of projects is targeted and when the software engineers are not seasoned. Whereas their work analysed and compared two integration platforms without to present a research methodology, our work analysed and compared four integration platforms, describing the steps that were followed to realise this comparison so that it is possible to repeat or apply the same process to other platforms. Schlauderer and Overhage[22] presented a framework for the assessment of software service providers, which contains 39 requirements in order to evaluate and compare characteristics of cloud service providers for enterprises. Such requirements were grouped into four criteria: service contract trustworthiness, technology, IT security, and service management. Amongst the requirements of the technology, the criterion is the interoperability, which deals with a seamless integration of services from different cloud providers, however, the authors do not provide many details on this criterion. Regarding academia, the authors presented their proposed framework as a starting point to formulate more deep approaches to the cloud procurement process and to improve existing approaches that support the assessment and selection of software services to enterprises, that needs to be better explored in order to unleash the full potential of this new trend. Whereas their work address evaluation of integration in more general terms and leave the application of their framework to future work, we evaluate a group of platforms focusing on performance attributes. Ebert and Weber[23] proposed a taxonomy for the analysis and evaluation of integration platforms and applied it to four of them, namely: Boomi, Informatica, Mule, and SAP. They described two groups of criteria: functional and non-functional. The functional criteria studied were: processed execution, number of operators, connectivity, administration, and development. The non-functional criteria studied were: price, service contract, trustworthiness, technology, safety, and service management. Their work distinguishes from ours since they provided a generic approach and did not provide a ranking for the integration platforms using a formal decision-making method to evaluate them.

Market trend and advisory enterprises such as Gartner[24] and Ovum[25] frequently publish reports in which they review and compare integration platforms. Their reports focus on properties related with market aspects, such as provider company maturity, customer relationship, or innovation, and, technology aspects such as the availability of connectors, security, monitoring, and governance of integration processes. Although these reports provide an interesting view of the integration platforms, they do not focus on properties that have an impact on the performance of the run-time system and even do not indicate a decision-making method that allows to compare and choose an integration platform focusing on performance and thus achieve a ranking for them .

It is noticed that existing proposals do not address properties to help software engineers to compare integration platforms in terms of performance, they also do not provide a decision-making method to guide enterprises in this important activity. Unfortunately, choosing an adequate integration platform is not a trivial task and this decision may lead to a cost overrun if an integration platform that has performance problems is chosen.

3 | BACKGROUND

In this section, we provide information on two key subjects to help in the understanding of the methodology used to evaluate the integration platforms, namely: Multiple Criteria Decision-Making (MCDM) and Analytic Hierarchy Process (AHP). First, we define MCDM and describes the main methods for decision-making. Then, we present AHP, which is the foundation behind the research methodology we have used.

3.1 | Multiple Criteria Decision Making

The making of decisions has been characterised by the rigorous scientific approaches in the literature, and they increase continuously. There are many decision methods that use numerical techniques to help decision makers to choose amongst a discrete set of alternative decisions, on the basis of their impact on certain criteria[26]. Any of the decision-making techniques involving numerical analysis of alternatives has the following three steps: *(i)* assign the relevant criteria and alternatives; *(ii)* assign numerical measures to the relative importance of the criteria and to the impacts of the alternatives on these criteria; and *(iii)* compute the numerical values to determine a ranking of each alternative.

Multi-criteria decision making is one of the most well-known branches of decision making and can be divided into multi-objective decision making and multi-attribute decision making[27]. The former approaches decision problems in which the decision space is continuous. The latter addresses problems with discrete decision spaces. Some of the methods widely used are the weighted sum model (WSM)[28], weighted product model (WPM)[29], AHP[15], and some variants of them.

In the WSM method, if there are m alternatives and n criteria, then the best alternative is the one that satisfies Equation 1:

$$A^*_{WSM-score} = \max_i \sum_{j=1}^{n} a_{ij} \cdot w_j, \tag{1}$$

for $i = 1, 2, 3, ..m$ and where $A^*_{WSM-score}$ is the WSM score of the best alternative, n is the number of decision criteria, a_{ij} is the actual value of the i alternative in terms of the j criterion, and w_j is the weight of importance of the j criterion. WSM can be used in single-dimensional cases, in which it uses the same units, however, presents difficulty when applied to multi-dimensional MCDM problems, in which it uses different units, because of the violation of the the additive utility assumption.

In the WPM method, each alternative is compared with the others by multiplying a number of ratios, one for each criterion. Each ratio is raised to a power, equivalent to the relative weight of the corresponding criterion. In general, it uses Equation 2 to compare two alternatives A_K and A_L.

$$R(\frac{A_K}{A_L}) = \prod_{j=1}^{n} (\frac{a_{Kj}}{a_{Lj}})^{w_j}, \tag{2}$$

for $i = 1, 2, 3, ..., m$ and where n is the number of criteria, $a_i j$ is the actual value of the i alternative in terms of the j criterion, and w_j is the weight of importance of the j criterion. If $R(\frac{A_K}{A_L})$ is greater than or equal to one, then it indicates that alternative A_K is better than alternative A_L. WPM eliminates any units of measure, thus, it can be used in single- and multi-dimensional MCDM.

3.2 | Analytic Hierarchy Process

AHP method[15] structures the problem in successive levels, allowing to model complex decisions in a hierarchical structure in the form of an inverted tree. At the highest level of the structure is the main purpose of the decision; below are the decision criteria and sub-criteria, when there are; and at the last level are the alternatives. AHP applies the technique of pairwise comparisons for obtaining numerical evaluations of qualitative aspects from experts and decision makers. The experts and decision makers, which judge the alternatives, must first establish priorities for their main criteria, by evaluating them in pairs considering their relative importance, thus generating a pairwise comparison matrix. Judgements are made based on a scale proposed by Saaty[15], in which values vary from a minimum value of zero to a maximum value of nine, cf. Figure 1 . The judgements are represented by the numbers on the scale and used to make the comparisons. This scale is widely used in decision-making methods and is based on psychological observations to realise judgements[30]. A consensus must be achieved, but if there are differences of opinions, the final value assigned must be the average of the values assigned in the individual judgement.

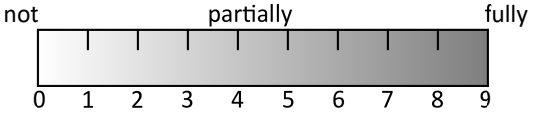

FIGURE 1 Saaty competence scale.

The number of judgements needed for a given matrix of order n is $n(n-1)/2$ because it is reciprocal and the diagonal elements are equal to unity. The results of pairwise comparison for every alternative is organised into positive reciprocal nxn matrix $\mathrm{P} = \left(a_{ij}\right)$, shown in Equation 3.

$$\mathrm{P} = \begin{pmatrix} 1 & a_{12} & \cdots & a_{1i} & \cdots & a_{1n} \\ \vdots & \ddots & \cdots & \vdots & \cdots & \vdots \\ a_{i1} & a_{i2} & \ddots & 1 & \cdots & a_{in} \\ \vdots & \vdots & \cdots & \ddots & \cdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{ni} & \cdots & 1 \end{pmatrix} \tag{3}$$

In the P matrix, a_{ij} represents the relative value of alternative A_i regarding to criterion C_j, in which the sum $\sum_{i=1}^{n} a_{ij}$ is equal to one. The best alternative in AHP is found in a similar way than in WSM, the difference is that AHP uses relative values instead of actual ones, thus, it can be used in single or multi-dimensional decision-making problems, as shown in Equation 4.

$$A^{*}_{AHP-score} = \max_{i} \sum_{j=1}^{n} a_{ij} \cdot w_j, \quad for\ i = 1, 2, 3, ..., m. \tag{4}$$

4 | RESEARCH METHODOLOGY

This section presents the research methodology for decision-making that was applied to compare and rank enterprise application integration platforms focusing on the performance of their run-time systems. This methodology is composed of the following steps: components selection, evaluating properties, rank construction, and results analysis, cf. Figure 2 . Components selection chooses some stakeholders and tools for the evaluation; evaluating properties find values for performance attributes; ranking construction deals with the production of the ranking for the integration platforms; and, results analysis examines and discusses the results. This methodology structures the problem in successive levels that allow modelling complex decision in a hierarchical structure in the form of an inverted tree. The highest level is the goal of choosing the integration platform with the best performance, at the second level are the criteria for decision-making and in the lowest level are the integration platforms that represent the possible alternatives. The steps of the research methodology and their activities are detailed below.

4.1 | Components Selection

This step aims to select key components for the evaluation and ranking of the integration platforms; these components are people and tools. People are stakeholders who analyse the platforms or make the pairwise comparison of them. Tools are platforms that will be evaluated or analysis tools that will be used to evaluate them. There may be inputs for this step, i.e., enterprise requirements such as previous lists of people and tools to be filtered, budget limit, etc. The outputs are people, tools, and attributes. It is not within of the scope of this article to present a methodology for selecting tools and people, but it is recommended this selection is made carefully and by people who know and represent the interests of the enterprise to ensure the success of evaluation. This step is divided into the following activities: expert selection, platform selection, analysis tool selection, and decision-maker selection. Some of these activities can be dispensed in case the enterprise has already selected tools or people.

4.1.1 Expert selection

This activity determines the experts that will choose and study integration platforms and analysis tools. Then, these experts will define the performance attributes which will be analysed in the run-time system of the platforms. It is recommended that the selected experts be software engineers or professionals experienced in enterprise application integration or familiar with some integration platform, as this makes the process faster and more consistent.

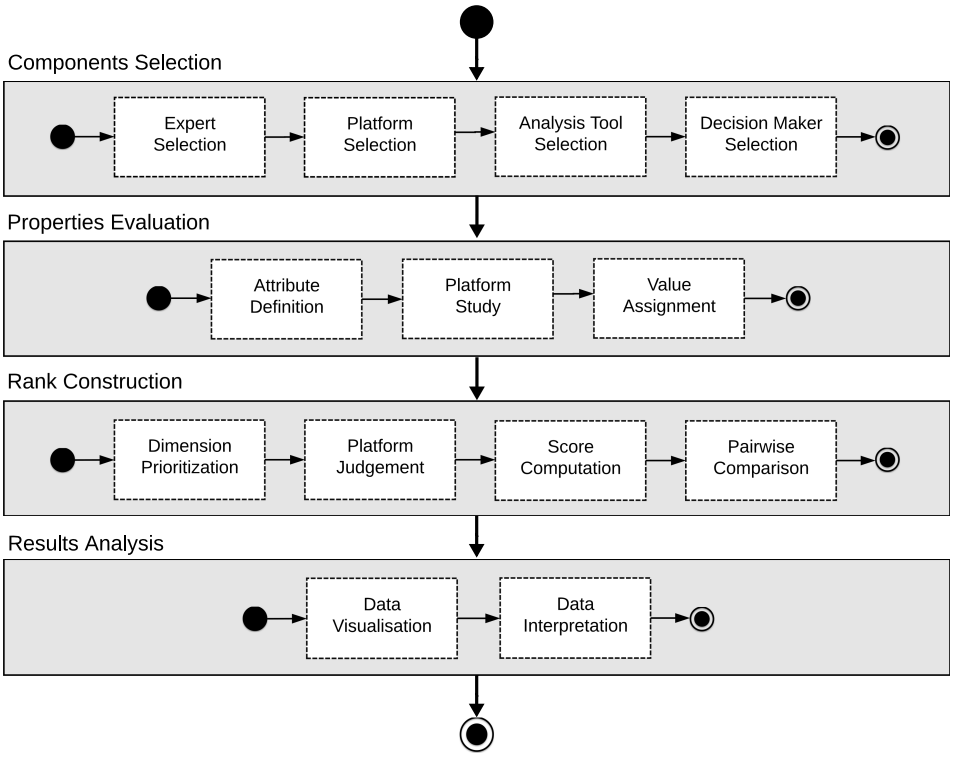

FIGURE 2 Overview of the research methodology we used.

4.1.2 Platform selection

This activity determines the integration platforms that will be compared. This choice can be made according to the interests of the enterprise and must be realised by experts. There are no restrictions on the choice of the platforms, but it is recommended that their source codes are available and accessible, or there is some project with documentation reporting about platforms' usage, from which it is possible to infer information regarding the performance of their run-time systems.

4.1.3 Analysis tool selection

This activity involves choosing the analysis tools that will serve to abstract the mathematical foundation behind the AHP method. The tools must provide a friendly graphical interface for experts and decision-makers, and the selection must be realised by an expert.

4.1.4 Decision-maker selection

This activity determines the decision-makers that will judge the integration platforms. Hereafter, they will define levels of relevance for dimensions and points for platforms. It is not necessary that these people are professionals in enterprise application integration, but it is recommended that they are more aligned with the interests of the enterprise.

4.2 | Properties Evaluation

This step aims to produce the data that supports the evaluation of the integration platforms, and every activity of this step must be realised by experts. This step is divided into the following activities: attribute definition, platform study, and value assignment.

4.2.1 Attribute definition

This activity determines properties of platforms that can impact their performance grouping them in dimensions. Attribute definition also determines qualitative candidate values for the properties and their order of preference such that the first value in the order is the one that provides the best performance to the run-time system, and the last is the one that provides the worst. Furthermore, each qualitative value must have a corresponding quantitative value, such that the first value in

the order of preference corresponds to the quantitative value equal to 10^{10}, the following corresponds to 10^9, and so on, until the last value in the order of preference that corresponds to the lower potency of zero.

4.2.2 Platform study

This activity consists of a deep study of the selected platforms with focus on the performance of their run-time systems. It is necessary to gather information from books, reports, official websites of providers, but when it comes to the performance of run-time systems, access to the source code is very important.

4.2.3 Value Assignment

This activity determines the qualitative values for the properties of each evaluated platform. These values must be justified by extracted information from the study of the source documentation obtained in the previous activity or by means of the experiments with the use of the platforms. The corresponding quantitative values are based on the previous activity.

The properties evaluation step produces three tables: (i) summary of the values for comparison properties; (ii) correspondence of qualitative and quantitative values; and, (iii) quantitative values of correspondence of each integration platforms. The first registers qualitative values of the properties. Columns represent the integration platforms, lines represent the properties grouped by dimension, and cells represent the qualitative values. The second registers the correspondence between qualitative and quantitative values. There is one column for qualitative and another for quantitative values; lines represent the properties grouped by dimension, and cells represent the values. The third registers the corresponding quantitative values. Columns in the table represent the integration platform alternatives, lines represent the properties into the respective dimensions, and cells represent the qualitative values; there is a subtotal of the sum of the cells for each dimension and a total of the overall sum for each integration platform.

4.3 | Ranking Construction

This step aims to produce the results that lead to the ranking of the integration platforms and involves determining the level of relevance for each dimension, rating the capacity of the integration platforms in relation to the properties, merging the criteria to calculate the scores for the integration platforms, and realising the pairwise comparison of the integration platforms. This step is divided into the following activities: dimension prioritisation, platform judgement, score computation, and pairwise comparison. We use the TransparentChoice software to support the computation of the ranking in our methodology. This is a decision-making software which permits to build a list of alternatives, to identify of dimensions that will be used for evaluation, to judge for each dimension, to analyse the sensitivity of results, and to generate reports.

4.3.1 Dimension prioritisation

This activity consists of determining the level of relevance for each dimension. This prioritisation is made by means of the collection of opinions of experts and represented by the assignment of a weight (ω) as a measure of consistency and quality about the chosen integration platform, based on Table 1 . When properties of a dimension are fundamentals to achieve a good performance, they must be highly observed when choosing the integration platform, then the relevance for this dimension is set to *essential*, and it assumes a weight equal to three. If the dimension improves the performance, but it is not essential, then the level of relevance for this dimension is set to *important*, and it assumes a weight equal to two. If the dimension adds positive features, but its absence does not significantly impact the performance, then the level of relevance for this dimension is set to *desirable*, and it assumes a weight equal to one.

TABLE 1 Levels of relevance for dimension.

Weight (ω)	**Relevance**	**Description**
3	Essential	The dimension is essential to meet the goal.
2	Important	The dimension is important to meet the goal.
1	Desirable	The dimension contributes to the goal as an additional element.

4.3.2 Platform judgement

This activity consists of quantifying the capacity of the integration platforms regarding to the properties of the dimensions, and it is made by means of a subjective and objective evaluation. The subjective evaluation follows a scale proposed by Saaty [15], according to Figure 1 . The objective evaluation converts the qualitative values of the properties into quantitative values defined in previous step. The calculation is made by merging subjective and objective criteria. The subjective criteria consist of the judgements done by experts about the competence of each integration platform in meeting a dimension, given by Equation 5:

$$subjective\_criteria_{i_k} \equiv qualitative\_value_{i_k}, \tag{5}$$

where i represents each integration platform and k represents each dimension. The objective criteria consist of the numeric conversion of the qualitative values of the properties into quantitative values, cf. Equation 6:

$$objective\_criteria_{i_k} = \sum_{j=1}^{m} quantitative\_value_{i_k}[j], \tag{6}$$

where j represents each quantitative value and m is the number of properties of dimension. The points achieved in the degree of competence of an integration platform (α_{i_k}) is the sum of the objective and the subjective criteria, cf. Equation 7:

$$\alpha_{i_k} = objective\_criteria_{i_k} + subjective\_criteria_{i_k} \tag{7}$$

4.3.3 Score computation

This activity calculates the scores for integration platforms based on the results of the previous activities. The partial score of an integration platform is calculated using the Equation 8:

$$partial\_score_{i_k} = \alpha_{i_k} \cdot \omega_k, \tag{8}$$

where i represents each integration platform, k represents each dimension; and k assumes values from 1 to n, where n is the number of dimensions. The total score of an integration platform is calculated using the Equation 9:

$$platform[i]\_score = \sum_{k=1}^{n} \alpha_{i_k} \cdot \omega_k \tag{9}$$

4.3.4 Pairwise comparison

This activity is used to evaluate how much an integration platform is better than the another, regarding each dimension. This evaluation uses a scale of one to nine, where a value equal to one is given when integration platforms are equivalent or equal, i.e., both integration platforms have the same competence to meet the evaluated dimension. A value equal to two means that an integration platform is two times more competent than the other, a value equal to three means that an integration platform is three times more competent than the other, and successively up to value nine, when an integration platform is nine times better than the other. The results of pairwise comparison for every integration platform is organised into positive reciprocal *nxn* matrix, cf. Equation 3.

Pairwise comparisons are well-supported by the TransparentChoice software, which checks the consistency of the comparisons and points out potential errors in its graphical editor. Besides, this software can forecast values of comparisons based on those already made to show approximate results of the evaluation process, even before making all comparisons.

At the end of ranking construction step, the following tables are produced: (i) summary of relevance of dimensions; (ii) degrees of competences of the integration platforms; and (iii) summary of pairwise comparisons. The first registers the weight given to each dimension. In this table, columns represent the level of relevance: essential, important and desirable; lines represent the dimensions, and cells represent the weight in the dimension. The second registers the values of the points obtained by means of subjective evaluation. In this table, columns represent the integration platforms alternatives, lines represent the dimensions, and cells represent the qualitative values in the dimension. The third registers the values in pairwise comparison. In this table, columns and lines represent the integration platforms alternatives, with lines grouped by dimensions, and cells represent the values corresponding to how much one integration platform is better than the other.

4.4 | Results Analysis

This step discusses the results produced in previous steps and is divided into the following activities: data visualisation and data interpretation.

4.4.1 Data visualisation

This activity focuses on the presentation of results by means of tables and graphics. The ranking of the integration platforms can be shown in a bar chart, in which a bar represents the evaluation of an integration platform for each dimension, and also in general, namely: partial score and total score, respectively. The width of the entire bar represents the total score of an integration platform and is composed of the partial scores in each dimension. Another way of representing the ranking of the integration platforms is through the perspective of the fulfilment of the properties in each one of the dimensions, by means of a radar graphic; the closer a vertex of a representation of a platform is of the vertices of a dimension, the more run-time system of this platform is endowed with the ideal values of the properties of this dimension. This kind of graphic allows visualising whether a platform is uniformly strong across all dimensions or fails in some of them. Therefore, with such information the enterprise is able to decide according to its interest.

4.4.2 Data interpretation

In this activity, these scores are interpreted and discussed, determining the achievement of the platform in each dimension and in overall. The consistency of the judgements should be checked regarding to the following statements:

- The platforms that reach the best quantitative values in a given dimension, will also occupy the best ranking position in such dimension.
- The best overall position in the ranking of the integration platforms must be occupied by the platform that best meets the dimension of a greater weight.
- The best ranking of the integration platforms must be consistent with the pairwise comparison, so if *Platform 1* is better than *Platform 2* and *Platform 2* is better than *Platform 3*, then *Platform 1* is better than *Platform 3*.

Usually, the analysis tools do every consistency checking incorporated the AHP method. However, in this activity, the actuation of the experts and decision-makers is fundamental, because they will be able to validate the requirements of the enterprise.

There is no restriction on which platforms and tools to analyse. There is also no limit on dimensions or properties. It is important to point out that once the properties, values, and order of preference are defined, as proposed in the attribute definition activity, they can be reused for the decision-making of any set of platforms, reducing the steps of the methodology and making the process faster.

5 | INTEGRATION PLATFORM EVALUATION

In this section, we report our experience in evaluating the performance of four open-source integration platforms, following the previously presented research methodology. The goal is to evaluate and rank the selected integration platforms to find the best in terms of performance of their run-time system in the execution of integration solutions. We structure the problem in successive levels in the form of an inverted tree, in which at the highest level is the goal of the best performance, at the second level are the dimensions, and at the lowest level are the integration platforms, cf. Figure 3 . The steps of the research methodology and their activities are detailed below.

5.1 | Components Selection

In this step, the integration platforms and the analysis tool were selected. We analysed and evaluated the platforms based on our experience of several years on the development of integration projects in real-world software ecosystems. For this reason, the expert selection and the decision-maker selection activities were completely dispensed.

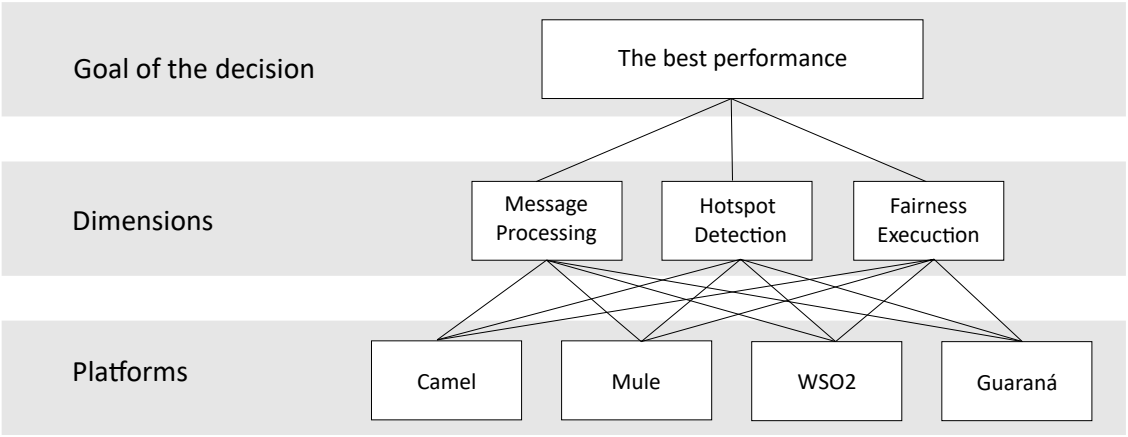

FIGURE 3 Hierarchical structure of the decision.

5.1.1 | Platform selection

In this activity, we selected the platforms Mule[5], Camel[4], WSO2[6] and Guaraná[7,8], which represent the state-of-the-art of integration platforms that provide support to the integration patterns[10] and follow the pipes-and-filters architectural style[9]. This selection was made based in collaboration with enterprises on the development of real-world projects, in which these were the technologies widely used.

5.1.2 | Analysis tool selection

In this activity, we chose as analysis tool the *TransparentChoice*[1] software, which supports the prioritisation and judgements of dimensions, as well as performs the tests of consistency for the evaluation. It also supports pairwise comparisons, checking their consistency, pointing out potential errors and predicting values of comparisons based on those already made.

5.2 | Properties Evaluation

In this step, we explored the available documentation of the four platforms in order to analyse performance properties of their run-time systems, resulting in a comparison framework composed of three dimensions, nine properties and a set of candidate values for them, which will be detailed in the following.

5.2.1 | Attribute Definition

In this activity, we defined the properties as evaluation criteria and they are organised in three dimensions: message processing, hotspot detection, and fairness execution. The message processing dimension concerns the efficiency of message processing within an integration solution. The hotspot detection dimension concerns the discovery of hotspot within an integration solution, in which message processing becomes upper to acceptable time. The fairness execution dimension concerns the balanced distribution of computational resources to message processing within an integration solution. In this article, we refer to threads as computational resources to execute tasks of an integration solution in the message processing, being a thread the basic unit of processing, i.e., the smallest sequence of programmed instructions that can be managed by the run-time system. Table 2 presents dimensions, properties, and their corresponding qualitative and quantitative values.

Message Processing. This dimension addresses the efficiency of the run-time system to process a message, which is related to the average number of messages processed per unit of time. Message processing is comprised by the following properties that relate to the capacity of reducing the real-time demanded by the integration solutions in order to completely process a message:

[1] https://www.transparentchoice.com/ahp-software

- **Thread pool creation.** This property indicates how the run-time system makes available the thread pools to execute the integration solutions, allowing to create one or more thread pools[4]. This property may take the following values, in ascendant order of preference: `global` or `local`. `global` value indicates that a single thread pool executes every task of an integration solution. `local` value indicates that the run-time system allows configuring local thread pools to execute a task or a group of tasks of an integration solution.
- **Message storage.** This property indicates how the run-time system deals with the storing of messages during the execution of an integration solution. Storing messages in-memory is faster[31] but can be more expensive. Messages that contain a big amount of data impact the amount of memory required for their processing inside the integration solutions. In such cases, rather than storing messages only in-memory, the run-time system can store them on disk. This property may take the following values, in ascendant order of preference: `in-memory`, or `hybrid`. The `in-memory` value indicates that the run-time system store messages only in-memory. The `hybrid` value indicates that the run-time system adopts different strategies for storing messages and do not limit to store only in-memory.
- **Thread pool configuration.** This property indicates how the run-time system tunes the size of the thread pool to deal with different workloads of messages. The run-time system allows to configure and manage thread pools to the integration solution. This property may take the following values, in ascendant order of preference: `static` or `dynamic`. The `static` value indicates that the thread pool is configured with a fixed number of threads, which was defined at design time by the software engineer. The `dynamic` value indicates that during run-time, the run-time system tune dynamically the number of threads of the pool, increasing or decreasing, according to the demand of the workload, within a range of values established at design time[32].

Hotspot Detection. This dimension addresses the detection of hotspots within the integration solutions by the run-time system. A hotspot is a piece of an integration solution where there is a bottleneck, therefore, messages are not being processed in an acceptable time. They are characterised by the overload of work to tasks and indicate that there is a lack of threads for that point of the integration solutions, what increase of the actual time required to process a message. The hotspot detection can help the run-time system allocate threads more efficiently to tasks in the integration solutions. The following properties can contribute to the detection of hotspots:

- **Detection stage.** Indicates the stage in which it is possible to determine the existence of hotspots in the integration solutions[33]. This property may take the following values, in ascendant order of preference: `design time`, `run-time`. If the detection stage takes the value `design time`, it means that software engineers must use their domain knowledge and modelling experience to forecast hotspots during design time; `run-time` means that the run-time system is endowed with intelligence to detect hotspots during the execution of the integration solutions.
- **Abstraction level.** This property indicates the level of decomposition in which the integration solutions can be broken to detect hotspots. The hotspot detection can be done by dividing the integration solution into pieces, which are groups of tasks chained or only a task. This property may take the following values, in ascendant order of preference: `per group`, `per task`. `per group` means that the piece of the integration solution analysed equates to a group of the task. `per task` means that the piece of the integration solution analysed equates to a single task, which allows a finer grained control[34].
- **Pattern identification.** This property indicates if the run-time system can detect patterns[35] that can lead to hotspots. Such patterns may determine: (i) known behaviours that occur during execution, increasing the average waiting time of the tasks that are ready to be executed; (ii) a particular combination of tasks identified in the design of the integration solutions. This property may take the following values, in ascendant order of preference: `no` or `yes`. `yes` indicates that the run-time system can detect patterns that can cause hotspots in the execution of pieces of an integration solution; in case it does not have this ability, the value is `no`.

Fairness Execution. This dimension addresses the assignment of threads to tasks, in a balanced way, in order to minimise the average time that a message takes to be processed in an integration solution. The following properties provide means that contribute to have a fair execution of tasks:

- **Execution policy.** This property indicates if the run-time system has the ability to adopt intervention policies to execute groups of tasks of an integration solution. Such intervention policies may: (i) start or stop the execution of a group of tasks chained; (ii) dictate the order in which the execution of a group of tasks should be started; (iii)

create additional groups of tasks chained and insert them within the integration solution[5]. This property may take the following values, in ascendant order of preference: `no` or `yes`. `yes` indicates that the run-time system can adopt intervention policies for the execution of groups of tasks chained of an integration solution; in case it does not have this ability, the value is `no`.

- **Scheduling policy.** This property indicates the policy followed by the run-time system to schedule the execution of tasks of an integration solution by computational resources. Tasks usually wait in a queue until there are available threads to execute them. In cloud environments, scheduling of an integration solution becomes challenging, because its performance must result in reduced scheduling overhead, minimised cost, and maximise resource utilisation while still meeting the specified deadline[36]. This property may take the following values, in ascendant order of preference: `fifo`, `priority` or `mapping`. `fifo` means that the run-time system follows first-in-first-out policy, in which the oldest (first-in) task into queue is executed first; `priority` means that the run-time system allows tasks to have priority associated, so that, a task with high priority is executed first; and, `mapping` means that the run-time system follows a mapping based on a mathematical model or optimisation method that allows finding an optimal scheduling policy to task execution by previously evaluating the integration solution.
- **Throttling controller.** This property indicates if the run-time system allows controlling the rate of incoming messages in an integration solution so that when this rate exceeds a previously determined limit, the run-time system can adopt suitable policies to preserve the execution of the integration solution. Such intervention policies may be: (i) refusing new messages; (ii) buffering at input the incoming messages or persisting them in a repository. This property may take the following values, in ascendant order of preference: `no` or `yes`. `yes` indicates that the run-time system can control the rate of incoming messages, that is, it has a throttling controller[10]; in case it does not have a throttling controller, the value is `no`.

TABLE 2 Performance attributes of run-time system of integration platforms.

Dimension	Property	Value	
		Qualitative	**Quantitative**
Message Processing	Thread pool creation	`global`	1
		`local`	10
	Message store	`in-memory`	1
		`hybrid`	10
	Thread pool configuration	`static`	1
		`dynamic`	10
Hotspot Detection	Detection stage	`design time`	1
		`run-time`	10
	Abstration level	`per group`	1
		`per task`	10
	Pattern identification	`no`	1
		`yes`	10
Fairness Execution	Execution policy	`no`	1
		`yes`	10
	Scheduling policy	`fifo`	1
		`priority`	10
		`mapping`	100
	Throttling controller	`no`	1
		`yes`	10

5.2.2 | Platform study

In this activity, we studied the platforms based on publicly available documentation of their source code and website, and on books and articles, taking into account our technical experience on using them on real-world projects.

Message Processing.

- **Thread pool creation.** We identified that Mule, WSO2 and Guaraná have a global thread pool to execute all tasks in the integration solution. However, Camel allows for the creation of local thread pools, which can be dedicated to executing a task or a group of tasks in an integration solution.
- **Message store.** We realised that the run-time systems are advanced in storing messages in the communication channels of an integration solution. Mule and Camel can store data in-memory and can adopt strategies to deal with other storage types, such as, in a file or a database. In contrast, WSO2 and Guaraná are able to store messages only in-memory. The storage types adopted by the platform can influence its message processing. Usually, disk storage can increase the total time of message execution, so that it should be used only in scenarios in which this type of storage is really needed, such as to deal with big data[37], that is, larger data in size or volume.
- **Thread pool configuration.** Only Mule has strategies to dynamically tune the size of the thread pool according to the demand of the execution of an integration solution. This ability allows run-time systems to deal with message processing peaks more efficiently by means of assigning threads to tasks with high demand, as well as, releasing threads, when the workload is lower.

Hotspot Detection.

- **Detection stage** The success of hotspot detection in the design phase depends on the expertise of the software engineer. In the case of the detection stage, every run-time system evaluated provides information that can indicate the presence of a hotspot in the execution of an integration solution, but only Camel is able to detect them at run-time.
- **Abstraction level.** Every run-time system, except Camel, is able to detect hotspots at task level. In all of the run-time systems, it is possible to specify tasks to be observed.
- **Pattern identification.** None of them can reveal bottleneck cues. Thus, the identification of such patterns depends on the quality of the information provided by the monitoring mechanisms and of the experience of the software engineer. Most of the integration platforms use some monitoring mechanism, which provides information to help detecting bottlenecks. However, they do not have any automated way to do this, leaving such detection to the software engineer.

Fairness Execution.

- **Execution policy.** Mule and Camel can control parts of the integration solution at run-time, starting or interrupting paths as execution policy.
- **Scheduling policy.** Mule, Camel, and Guaraná use first-in-first-out, and WSO2 allows tasks to have an associated priority to influence their scheduling, so that it is possible to set a high priority to tasks that need to be executed first or more often.
- **Throttling controller.** Only Mule and Camel allow tasks to actively call a thread at regular intervals of time, which can help ensure a fairer execution when the rate of incoming messages is high.

5.2.3 | Value Assignment

In this activity, we assigned the values for each property obtained from the study of the platforms. The values found for every property in each of the three dimensions are presented in Table 3 . The quantitative values for each property are presented in Table 4 . This values are based on Table 2 , in which the lowest value in ascendant order of preference receives the lowest quantitative value, and the highest value in ascendant order of preference receives the highest quantitative value. The quantitative values for each dimension are summed in Table 4 . In the message processing dimension, Mule and Camel achieved 21 points, and WSO2 and Guaraná achieved 3. In the hotspot detection dimension, Mule, Camel, WSO2 and Guaraná achieved 12 points. In the fairness execution dimension, Mule and Camel achieved 21 points, WSO2, 12, and Guaraná, 3.

TABLE 3 Summary of the values for comparison properties.

		Integration Platforms			
Dimension	**Property**	***Mule***	***Camel***	***WSO2***	***Guaraná***
Message Processing	Thread pool creation	`global`	`local`	`global`	`global`
	Message store	`hybrid`	`hybrid`	`in-memory`	`in-memory`
	Thread pool configuration	`dynamic`	`static`	`static`	`static`
Hotspot Detection	Detection stage	`design time`	`run-time`	`design time`	`design time`
	Abstraction level	`per task`	`per group`	`per task`	`per task`
	Pattern identification	`no`	`no`	`no`	`no`
Fairness Execution	Execution policy	`yes`	`yes`	`no`	`no`
	Scheduling policy	`fifo`	`fifo`	`priority`	`fifo`
	Throttling controller	`yes`	`yes`	`no`	`no`

TABLE 4 Quantitative values of competence for each integration platform.

		Integration Platforms			
Dimension	**Property**	***Mule***	***Camel***	***WSO2***	***Guaraná***
Message Processing	Thread pool creation	1	10	1	1
	Message store	10	10	1	1
	Thread pool configuration	10	1	1	1
Total		21	21	3	3
Hotspot Detection	Detection stage	1	10	1	1
	Abstraction level	10	1	10	10
	Pattern identification	1	1	1	1
Total		12	12	12	12
Fairness Execution	Execution policy	10	10	1	1
	Scheduling policy	1	1	10	1
	Throttling controller	10	10	1	1
Total		21	21	12	3

5.3 | Ranking Construction

In this step, we build the ranking for the integration platforms, following the activities proposed in the research methodology. Dimensions were prioritised, the platform competencies were found by means of the objective and subjective criteria, the score of each evaluated integration platform was calculated, and integration platforms were pairwise compared.

5.3.1 | Dimension Prioritisation

In this activity, we assigned the level of relevance to each dimension, based on Table 1 . The level of relevance is the component of greater weight in the score of an integration platform. The message processing was judged as an *essential* dimension, thus good values to their properties are necessary to achieve a good performance in the execution of an integration solution. Hotspot detection was judged as a *desirable* dimension, therefore this dimension adds elements to increase the performance, but they are not essential. Fairness execution was judged as an *important* dimension, so the values of their properties have a medium level of relevance. The prioritisation of the dimension resulted in the values shown in Table 5 .

5.3.2 | Platform Judgement

In this activity, we quantified the competence of each integration platform to achieve a good performance in the execution of an integration solution, and for this, subjective and objective evaluations were done. The subjective evaluation of the competence

TABLE 5 Summary of relevance of dimensions.

Dimension	Essential	Important	Desirable
Message Processing	3	-	-
Hotspot Detection	-	-	1
Fairness Execution	-	2	-

of the integration platforms was made with the help of the scale introduced in Figure 1 and took into account the degree and the manner in which the integration platforms meet the performance requirements represented by properties of the dimension. We made the judgement based on the study of the of integration platforms, in which it was possible to identify how every one of them is endowed to meet the properties. Then, we assigned a value from 0 to 9 and filled Table 6 with the results.

TABLE 6 Degrees of competence of the integration platform.

	Integration Platforms			
Dimension	***Mule***	***Camel***	***WSO2***	***Guaraná***
Message Processing	4	8	1	1
Hotspot Detection	3	3	3	3
Fairness Execution	6	6	2	1

In the following, we detail the computation of the individual pointing (α_{i_k}) to each integration platform, where i assumes values from 1 to 4, corresponding to Mule, Camel, WSO2 or Guaraná; and k assumes values from 1 to 3, corresponding to the dimensions of message processing, hotspot detection, and fairness execution, respectively.

First, subjective criteria were determined, based on Table 6 , which provides the resulting values of our judgement of the competence of each integration platform to meet a dimension.

Equation 10 presents a matrix whose elements correspond to the qualitative resulting values. In this matrix, i is the index of the lines and represents the integration platform; and k is the index of the columns and represents the dimension.

$$subjective\_criteria_{i_k} = \begin{bmatrix} 4 & 3 & 6 \\ 8 & 3 & 6 \\ 1 & 3 & 2 \\ 1 & 3 & 1 \end{bmatrix} \tag{10}$$

Second, objective criteria were determined, based on Table 4 , in which we provided the summation of quantitative values of the properties. Equation 11 presents a matrix whose elements correspond to the quantitative resulting values. In this matrix, i is the index of the lines and represents the integration platform; and k is the index of the columns and represents the dimension.

$$objective\_criteria_{i_k} = \begin{bmatrix} 21 & 12 & 21 \\ 21 & 12 & 21 \\ 3 & 12 & 12 \\ 3 & 12 & 3 \end{bmatrix} \tag{11}$$

Third, the points achieved in the degree of competence of an integration platform is equal to the sum of objective and subjective criteria, according to Equation 12. The points achieved in the degree of competence is represented by Equation 12, whose

elements correspond to the achieved points in the dimension. In these matrices, i is the index of the lines and represents the integration platform; and k is the index of the columns and represents the dimension.

$$\alpha_{i_k} = \overbrace{\begin{bmatrix} 4 & 3 & 6 \\ 8 & 3 & 6 \\ 1 & 3 & 2 \\ 1 & 3 & 1 \end{bmatrix}}^{subjective\_criteria_{i_k}} + \overbrace{\begin{bmatrix} 21 & 12 & 21 \\ 21 & 12 & 21 \\ 3 & 12 & 12 \\ 3 & 12 & 3 \end{bmatrix}}^{objective\_criteria_{i_k}} \tag{12}$$

$$\alpha_{i_k} = \begin{bmatrix} 25 & 15 & 27 \\ 29 & 15 & 27 \\ 4 & 15 & 14 \\ 4 & 15 & 4 \end{bmatrix} \tag{13}$$

The degree of competence of a platform indicates how many points an integration platform achieved in each dimension, independent of the level of relevance of the dimension. It is important to observe that the maximum quantitative value is greater than the maximum value of the scale of competence, which ensures that an integration platform with a high summation of the quantitative values is better ranked, even though the value assigned to the integration platform by means of our subjective judgement is not quite high. Thus, objective criteria will prevail over subjective criteria.

5.3.3 | Score Computation

In this activity, we calculated the score for the integration platforms. This calculation was made by the merging of subjective and objective criteria regarding the competence of the integration platforms, as well as the weight of each dimension. At the end of this activity, each integration platform achieved a partial score ($partial\_score_{i_k}$) and a total score ($platform[i]\_score$), in which i assumes values from 1 to 4, corresponding to Mule, Camel, WSO2 or Guaraná, respectively; and k assumes values from 1 to 3, corresponding to the dimensions message processing, hotspot detection, and fairness execution, respectively.

In the following, the score computation is detailed step-by-step. First, the partial score of the integration platforms is equal to the multiplication of the points achieved in the degree of competence of the platform in meeting a dimension (α_{i_k}) by the weight of the respective dimension (ω_k), according to Equation 14.

$$partial\_score_{i_k} = \overbrace{\begin{bmatrix} 25 & 15 & 27 \\ 29 & 15 & 27 \\ 4 & 15 & 14 \\ 4 & 15 & 4 \end{bmatrix}}^{\alpha_{i_k}} \times \overbrace{\begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix}}^{\omega_k} \tag{14}$$

$$partial\_score_{i_k} = \begin{bmatrix} 75 & 15 & 54 \\ 87 & 15 & 54 \\ 12 & 15 & 28 \\ 12 & 15 & 8 \end{bmatrix} \tag{15}$$

The partial score is represented by Equation 15, with elements that correspond to the achieved score in the dimension. In these matrices, i is the index of the lines and represents the integration platform; and k is the index of the columns and represents the dimension. Finally, the total score of the integration platforms is equal to the sum of the partial scores, i.e., the sum of the elements of the lines of matrix 15. Equation 16 presents a matrix whose elements correspond to the score of the integration platforms. In this matrix, i is the index of the lines and represents the integration platform. Equation 17 expresses analytically

the total score of each integration platform i, by means of the summation of the multiplication of the points achieved for the degree of competence of of the platform in a dimension k by the weight of this dimension k.

$$platform[i]\_score = \begin{bmatrix} 144 \\ 156 \\ 55 \\ 35 \end{bmatrix} \tag{16}$$

$$\begin{aligned} Mule\_Score &= 25 \cdot 3 + 15 \cdot 1 + 27 \cdot 2 = 75 + 15 + 54 = 144 = 144 \\ Camel\_Score &= 29 \cdot 3 + 15 \cdot 1 + 27 \cdot 2 = 87 + 15 + 54 = 156 = 156 \\ WSO2\_Score &= 4 \cdot 3 + 15 \cdot 1 + 14 \cdot 2 = 12 + 15 + 28 = 55 = 55 \\ Guaraná\_Score &= 4 \cdot 3 + 15 \cdot 1 + 4 \cdot 2 = 12 + 15 + 8 = 35 = 35 \end{aligned} \tag{17}$$

5.3.4 | Pairwise Comparison

In this activity, we compared each pair of integration platforms, judging how many times an integration platform is better than another, using a scale from one to nine, where a value equal to one is given when the integration platforms are equivalent or equal, and other values indicate the number of times an integration platform is better than another. When an integration platform is compared with itself, the value 1 was assigned.

Integration platforms were compared two-by-two regarding each dimension, taking into account the values of Table 6 . This comparison was supported by the analysis tool, resulting in Table 7 . In the message processing dimension, Camel was judged as two times better than Mule. Mule was judged as four times better than WSO2 and Guaraná. In this dimension, Camel was judged as eight times better than WSO2 and Guaraná; WSO2 was judged as equivalent to Guaraná. In the hotspot detection dimension, Camel, Mule, WSO2 and Guaraná were judged as equivalent. In the fairness execution dimension, Camel was judged as equivalent to Mule; Mule was judged three times better than WSO2 and six times better than Guaraná. In this dimension, Camel was judged three times better WSO2 and six times better than Guaraná; in this same dimension WSO2 was judged two times better than Guaraná.

TABLE 7 Summarise pairwise comparison.

		Integration Platforms			
Dimension		***Mule***	***Camel***	***WSO2***	***Guaraná***
Message Processing	***Mule***	1x	1/2x	4x	4x
	Camel	2x	1x	8x	8x
	WSO2	1/4x	1/8x	1x	1x
	Guaraná	1/4x	1/8x	1x	1x
Hotspot Detection	***Mule***	1x	1x	1x	1x
	Camel	1x	1x	1x	1x
	WSO2	1x	1x	1x	1x
	Guaraná	1x	1x	1x	1x
Fairness Execution	***Mule***	1x	1x	3x	6x
	Camel	1x	1x	3x	6x
	WSO2	1/3x	1/3x	1x	2x
	Guaraná	1/6x	1/6x	1/2x	1x

5.4 | Results Analysis

In this step, the results were analysed and the ranking of the integration platforms was graphically represented. First, we present a visualisation of the score and then an interpretation of the data.

5.4.1 | Data Visualisation

In this activity, we built graphics to represent the results. Figure 4 shows the total ranking for these integration platforms, and also presents both the total score and the partial score in each of the three dimensions.

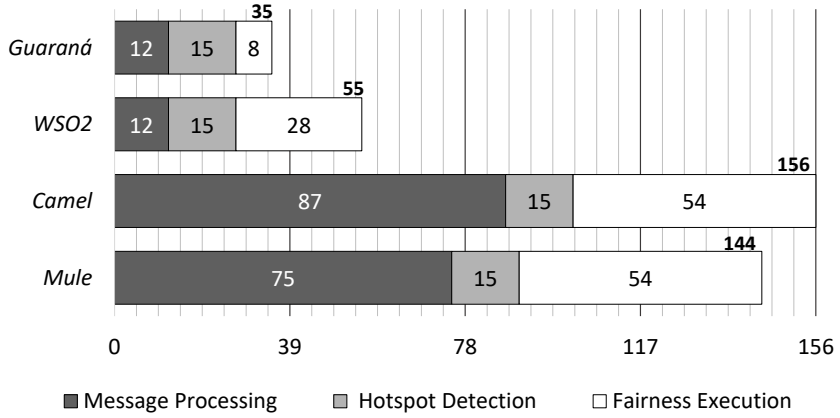

FIGURE 4 Ranking of integration platforms.

The ranking for Mule, Camel, WSO2, and Guaraná indicates that Camel has the best performance amongst the analysed integration platforms, reaching a score of 156, followed by Mule, WSO2, and Guaraná, respectively with scores 144, 55, and 35. Figure 5 shows the ranking by means of a radar graphic. At the vertices of the outermost triangle of dotted lines are the dimensions. The inner triangles represent the fulfilment of the properties by integration platforms. The biggest triangle represents the fulfilment of the properties by Camel; the second biggest represents the fulfilment by Mule; the third represents the fulfilment by WSO2; and, the smallest black triangle represents the fulfilment by Guaraná.

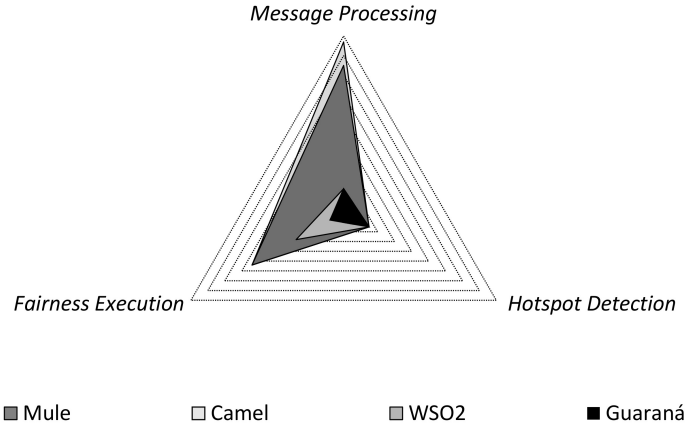

FIGURE 5 Fulfilment of the properties of the dimensions.

5.4.2 | Data Interpretation

In this activity, we analysed and interpreted the results with the help of the produced graphs. Camel was the integration platform with the highest score in the message processing dimension, reaching a score of 87. Mule reached a score of 75, while WSO2 and Guaraná reached a score of 12. In this dimension, the level of relevance was defined as equal to *essential*, thus the weight equalled 3. In the hotspot detection dimension, every integration platform had the same score, Camel, Mule, WSO2, and Guaraná reached a score of 15. In this dimension, the level of relevance was defined as equal to *desirable*, thus the weight equalled 1. In the fairness execution dimension, Camel and Mule reached the highest score and Guaraná the lowest. Camel and Mule reached a score of 54, WSO2 reached a score of 28, and Guaraná reached a score of 8. In this dimension, the level of relevance was defined as equal to *important*, thus the weight equalled 2.

In the fairness execution dimension, Mule achieved 27 points in the degree of competence, which is better than its points in in the message processing dimension, which achieved 25 points; however, when the points of degree of competence were multiplied by the weight of the respective dimension, Mule's partial score reached a better result in the message processing dimension than in the fairness execution dimension, because the level of relevance of the former dimension was considered *essential* to achieve a good performance in the execution of an integration solution, while the latter dimension was considered *important* to achieve this goal. In the hotspot detection dimension, WSO2 achieved 15 points in the degree of competence, which is better than its points in the fairness execution dimension, which achieved 14 points; however, when the points of degree of competence were multiplied by the weight of the respective dimension, WSO2's partial score reached a better result in the fairness execution dimension than in the hotspot detection dimension, because the level of relevance of the former dimension was considered *important* to achieve a good performance in the execution of an integration solution, while the latter dimension was considered only *desirable* to achieve this goal.

Summarising, Mule achieved more points in the degree of competence in the fairness execution dimension, which achieved 27 points. Camel was better in the degree of competence in the message processing dimension, which achieved 29 points. WSO2 and Guaraná were better in the degree of competence in the hotspot detection dimension, which achieved 15 points. It is important to note that the high partial score in the message processing dimension leveraged the total score of Mule and Camel, since this dimension has priority over the others. Similarly, the high partial score in the fairness execution increased the total score of the four integration platforms, since this dimension has priority over hotspot detection dimension.

6 | CONCLUSIONS

Companies frequently need to integrate different applications that compose their software ecosystem, and so need to rely on integration platforms that provide appropriate performance. Integration platforms are specialised software tools that allow information to be kept consistent and synchronised on all applications. The run-time system is part of these tools and is responsible for running integration solutions; therefore, its performance often influences the decision of companies in choosing an integration platform. There is a growing number of integration platform offers on the market, so it has become a challenge for software engineers to select one of them when performance is a central requirement. In this article, we presented an evaluation and ranking of four open-source integration platforms, which follow the pipes-and-filters architectural style and support several integration patterns documented by Hohpe and Woolf[10]. First, we structured our research methodology, which takes into account subjective and objective criteria. After, we related a set of properties regarding the performance of run-time systems of integration platforms and evaluated the four platforms according to these properties. Finally, we ranked the these integration platforms, with Camel achieving the best performance, followed by Mule, WSO2, and Guaraná, respectively.

6.1 | Scientific Contribution

We made the comparison of the integration platforms with the support of a research methodology based on the AHP method. This methodology allowed us to model a complex decision problem in a hierarchical structure divided into three levels. At the top of this hierarchy is the goal of the decision, which was finding the integration platform with the best performance in the execution of integration solutions; below of it are the decision criteria that were defined as dimensions of performance properties; and, at the lowest level are the alternatives of choice that were defined as the chosen platforms. Although we used specific decision criteria, in this research methodology it is possible to add new criteria and sub-criteria, such as price, ease of use, learning curve, as well as, adapt it to other goals. The hierarchical structuring of the problem facilitated the judgement of

a dimension regarding its priority in relation to another dimension. We seek a methodology that considered either objective or subjective aspects. We seek a methodology that considered either objective or subjective aspects. AHP was chosen because an important aspect of it is that it takes into account human judgments, so supporting our methodology to consider the preferences of experts in the evaluation of the integration platforms. AHP has been widely used applications related to decision-making [16,17] because of its strong mathematical and psychological background that allow to tackle complex decisions by breaking them into small and affordable decisions. According to Ishizaka and Labib [38], psychologically, for achieving a consensus, it is easier and more accurate to express an opinion about two alternatives than about all the alternatives at the same time. We used pairwise comparisons to focus on a small and well-defined action of comparing two platforms in a given dimension. We compared each pair of platforms by judging how many times one is more competent than the other to meet each dimension, and registered the values of this judgement. The level of relevance of a dimension was the factor of highest influence on the ranking of an integration platform, being able to either leverage or worsen the ranking. It is important to observe that the inclusion of objective criteria, which is not part of the original AHP method, ensure that an integration platform that has better objective values for properties to be better ranked, even though the subjective judgement is not quite high. So, objective criteria prevail over subjective criteria. This means that if the same set of platforms are judged for different experts, considering the same objective values for properties of these platforms, the ranking is not changed. The contribution of the subjective criteria in the raking computation is in the insertion of a measure of how much the platforms are endowed of the quality attributes represented by the performance properties. As quality attributes, this measure cannot assume only discrete values, so we use the Saaty scale that allows experts to assign values of intensity ranging from 0 to 9. Briefly, objective criteria define ranking, while subjective ones define how much a platform is better than the others.

6.2 | Practical contribution

In our evaluation of the integration platforms, we took into account three dimensions of performance properties, representing the rating criteria of the platform: message processing, hotspot detection, and fairness execution. Each one contained a set of properties that can impact the performance of the integration platforms. Message processing focuses on efficient message processing within an integration solution; hotspot detection focuses on the detection of bottlenecks in an integration solution; and fairness execution focuses on the fair execution of tasks within an integration solution. In this evaluation, we gave priority to the message processing dimension compared to fairness execution dimension; and we gave priority to the fairness execution dimension compared to hotspot detection dimension. We assigned values to properties, then we achieved an overall ranking of the performance of the platforms, in which it is possible to observe the results in each one of the dimensions individually and the integration platform that achieved the best total score in relation to performance in the execution of integration solutions. Regarding the partial score, Camel was the integration platform with the highest score in the message processing dimension; every integration platform had the same score in the hotspot detection dimension; and, both Camel and Mule reached the highest score in the fairness execution dimension. The ranking of integration platforms can help enterprises in the complex task of choosing an integration platform that has a better performance in the execution of an integration solution, and consequently, providing an improvement to their business processes. The properties and their values were found based in our experience on the development of integration projects in real-world software ecosystems, in which performance is a key feature. We have been using and developing them for more than 10 years working on real-world integration problems focusing on the performance of the platforms. We believe this set of properties is a good starting point to evaluate integration platforms, but can also be extended to make decision process more accurate. The practical contribution that this article provided motivates us to continue our research, extending this evaluation to a larger number of platforms.

ACKNOWLEDGEMENTS

This work was supported by the Brazilian Co-ordination Board for the Improvement of University Personnel (CAPES) under grants 73318345415 and 88881.119518/2016-01; and, the Research Support Foundation of the State of Rio Grande do Sul (FAPERGS) under grant 17/2551-0001206-2. We would like to thank Dr. Rafael Corchuelo and Dr. Inma Hernández from the University of Seville (Spain) and Ms. Elizabeth Thornton Rush from the Pennsylvania State University (United States) for their helpful comments in earlier versions of this article.

References

1. Manikas Konstantinos. Revisiting software ecosystems Research: A longitudinal literature study. *Journal of Systems and Software.* 2016;117:84–103.
2. Varghese Blesson, Buyya Rajkumar. Next generation cloud computing: New trends and research directions. *Future Generation Computer Systems.* 2018;79:849–861.
3. Frantz Rafael Z., Corchuelo Rafael, Molina-Jiménez Carlos. A proposal to detect errors in Enterprise Application Integration solutions. *Journal of Systems and Software.* 2012;85(3):480–497.
4. Ibsen Claus, Anstey Jonathan. *Camel in action*. Manning Publications Co.; 2010.
5. Dossot David, DÉmic John, Romero Victor. *Mule in action*. Manning Publications Co.; 2014.
6. Indrasiri Kasun. *Introduction to the WSO2 ESB*. Springer; 2016.
7. Frantz Rafael Z., Quintero Antonia M. Reina, Corchuelo Rafael. A Domain-Specific Language to Design Enterprise Application Integration Solutions. *International Journal of Cooperative Information Systems.* 2011;20(02):143–176.
8. Frantz Rafael Z., Corchuelo Rafael, Roos-Frantz Fabricia. On the design of a maintainable software development kit to implement integration solutions. *Journal of Systems and Software.* 2016;111:89–104.
9. Alexander Christopher, Ishikawa Sara, Silvertein Murray. *A pattern language: towns, buildings, construction*. Oxford University Press; 1977.
10. Hohpe Gregor, Woolf Bobby. *Enterprise integration patterns: Designing, building, and deploying messaging solutions*. Addison-Wesley Professional; 2004.
11. Khoumbati Khalil, Themistocleos Marinos, Irani Zahir. Evaluating the Adoption of Enterprise Application Integration in Health-Care Organizations. *Journal of Management Information Systems.* 2006;22:69–108.
12. Botta Alessio, Donato Walter, Persico Valerio, Pescapé Antonio. Integration of Cloud computing and Internet of Things: A survey. *Future Generation Computer Systems.* 2016;56:684–700.
13. Ebert Nico, Weber Kristin, Koruna Stefan. Integration Platform as a Service. *Business & Information Systems Engineering.* 2017;59:375–379.
14. Brahmi Zaki, Gharbi Chaima. Temporal reconfiguration-based orchestration engine in the cloud computing. In: International Conference on Business Information Systems (ICBIS):73–85; 2014.
15. Saaty Thomas L.. How to make a decision: the analytic hierarchy process. *European Journal of Operational Research.* 1990;48:9–26.
16. Vaidya Omkarprasad S., Kumar Sushil. Analytic hierarchy process: An overview of applications. *European Journal of Operational Research.* 2006;169:1–29.
17. Saaty Thomas L. Relative measurement and its generalization in decision making why pairwise comparisons are central in mathematics for the measurement of intangible factors the analytic hierarchy/network process. *Revista de la Real Academia de Ciencias Exactas, Fisicas y Naturales.* 2008;102:251–318.
18. Freire Daniela L., Frantz Rafael Z., Roos-Frantz Fabricia, Sawicki Sandro. (in-press). Survey on the run-time systems of enterprise application integration platforms focusing on performance. *Software: Practice and Experience.* ;DOI: 10.1002/spe.2670:1-20.
19. Corchuelo Rafael, Frantz Rafael Z., González Jesús. Una Comparación de ESBs desde la Perspectivade la Integración de Aplicaciones. In: Jornadas de Ingeniería del Software y Bases de Datos (JISBD):403–408; 2008.

20. Tan Suo, Milhim Hamzeh K Bani, Chen Bo, Schiffauerova Andrea, Zeng Yong. Enterprise Applications Integration Using Environment Based Design (EBD). In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (ASME):1245–1255; 2011.

21. More Vishal R., Bartere M. M.. Enterprise Integration using Boomi Tool. *International Journal of Advanced Information Science and Technology.* 2013;11:18–22.

22. Schlauderer Sebastian, Overhage Sven. Selecting Cloud Service Providers - Towards a Framework of Assessment Criteria and Requirements. In: Wirtschaftsinformatik (WI2015):75–90; 2015.

23. Ebert Nico, Weber Kristin. Integration Platform as a Service in der Praxis: Eine Bestandsaufnahme. In: Multikonferenz Wirtschaftsinformatik (MKWI):1675–1685; 2016.

24. Guttridge Keith, Pezzini Massimo, Golluscio Elizabeth, Thoo Eric, Iijima Kimihiko, Wilcox Mary. *Magic quadrant for enterprise integration platform as a service 2017.* : Gartner, Inc; 2017.

25. Sharma Saurabh. *Ovum Decision Matrix highlights the growing importance of iPaaS and API platforms in hybrid integration.* : Ovum Consulting; 2017.

26. Triantaphyllou Evangelos. *Multi-criteria Decision Making Methods: A Comparative Study*. Springer Science & Business Media; 2013.

27. Zimmermann Hans-Jürgen. *Fuzzy Set Theory - and Its Applications*. Springer Publishing Company, Incorporated; 3 ed.2014.

28. Fishburn Peter C.. Additive Utilities with Incomplete Product Sets: Application to Priorities and Assignments. *Operations Research.* 1967;(3):537–542.

29. Bridgman Percy Williams. *Dimensional analysis*. Yale University Press; 1922.

30. JiříFranek , Kresta Aleš. Judgment scales and consistency measure in AHP. *Procedia Economics and Finance.* 2014;12:164–173.

31. Balko S., Barros A.. In-Memory Business Process Management. In: International Conference on Enterprise Distributed Object Computing (EDOC):74–83; 2015.

32. Gleyzer Gene, Howes Jason. *System and method for supporting dynamic thread pool sizing in a distributed data grid.* US Patent 9,547,521 B2; 2017.

33. Ahmad Sheraz, Bahdur Faisal, Kanwal Faiza, Shah Riaz. Load balancing in distributed framework for frequency based thread pools. *Computational Ecology and Software.* 2016;6(4):150–164.

34. Sudarsanam Arvind, Srinivasan Mayur, Panchanathan Sethuraman. Resource estimation and task scheduling for multi-threaded reconfigurable architectures. In: International Conference on Parallel and Distributed Systems (ICPADS):323–330; 2004.

35. Ritter Daniel, May Norman, Rinderle-Ma Stefanie. Patterns for emerging application integration scenarios: a survey. *Information Systems.* 2017;67:36–57.

36. Anwar Nazia, Deng Huifang. Elastic scheduling of scientific workflows under deadline constraints in cloud computing environments. *Future Internet.* 2018;10(1):1–23.

37. Chen Min, Mao Shiwen, Liu Yunhao. Big data: a Survey. *Mobile Networks and Applications.* 2014;19:171–209.

38. Ishizaka Alessio, Labib Ashraf. Review of the main developments in the analytic hierarchy process. *Expert systems with applications.* 2011;38:14336–14345.

How to cite this article: Daniela L. Freire, Rafael Z. Frantz, Fabricia Roos-Frantz (2018) Ranking Enterprise Application Integration Platforms from a Performance Perspective: an Experience Report, *Software: Practice and Experience*, *2018;00:1–6*.