Received: 26 May 2020 | Revised: 31 August 2020 | Accepted: 3 October 2020

DOI: 10.1002/spe.2916

SPECIAL ISSUE PAPER

WILEY

A cloud-based integration platform for enterprise application integration: A Model-Driven Engineering approach

Rafael Z. Frantz[1] | Rafael Corchuelo[2] | Vitor Basto-Fernandes[3] | Fernando Rosa-Sequeira[4] | Fabricia Roos-Frantz[1] | José L. Arjona[5]

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

[2]ETSI Informática, University of Seville, Seville, Spain

[3]Department of Information Science and Technology, University Institute of Lisbon, Lisbon, Portugal

[4]Informatics Engineering Department, Polytechnic Institute of Leiria, Leiria, Portugal

[5]i2Factory, S.L., Technological Scientific Park of Huelva, Huelva, Spain

Correspondence
Rafael Z. Frantz, Department of Exact Sciences and Engineering, Unijuí University, Ijuí, Brazil.
Email: rzfrantz@unijui.edu.br

Funding information
Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Grant/Award Number: 88881.119518/2016-01; CYTED Ciencia y Tecnología para el Desarrollo, Grant/Award Numbers: TIN2010-21744-C1, TIN2013-40848-R, TIN2016-75394-R; Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul, Grant/Award Number: 17/2551-0001206-2; Fundação para a Ciência e a Tecnologia, Grant/Award Number: UID/CEC/4524/2016

Abstract

This article addresses major information systems integration problems, approaches, technologies, and tools within the context of Model-Driven Software Engineering. The Guaraná integration platform is introduced as an innovative platform amongst state-of-the-art technologies available for enterprises to design and implement integration solutions. In this article, we present its domain-specific modeling language and its industrial cloud-based web development platform, which supports the design and implementation of integration solutions. A real-world case study is described and analyzed; then, we delve into its design and implementation, to finally disclose ten measures that empirically help estimating the amount of effort involved in the development of integration solutions.

KEYWORDS

domain-specific language, enterprise application integration, integration framework, integration patterns, integration platform as a service, integration systems modeling

1 | INTRODUCTION

The field of Software Engineering is involved in a paradigm shift that has important consequences on how software engineers devise and evolve systems. The discipline[1] of Model-Driven Engineering is central to this change, once it promotes models as though first-class citizens, in every stage of the software development process. Models are abstractions that allow software engineers to focus on relevant aspects of a software system and to ignore its irrelevant details.

This discipline is based on raising the level of abstraction of the overall development process; the goal is to formalize systems as collections of reusable models, to separate business logic descriptions from a particular platform implementation, and to automate the implementation stage.[2-6]

The Model-Driven Architecture approach focuses on applying Model-Driven Engineering to the design and implementation stages of the software development process. It promotes modeling a software system as a collection of interrelated models at different levels of abstraction. A model can be used as the source to produce other models at the same level or at different levels of abstraction.[7] In the first case, resulting models are typically refactorizations, whose goal is to increase the quality of original models (like for instance, their maintainability or their efficiency), while in the latter they are typically refinements or generalizations, whose goal is to introduce or remove technology-specific details.

In the Model-Driven Architecture, two kinds of models are frequently used, namely: platform-independent models and platform-specific models. The former describes systems as collections of operations, structures, and behaviors,[8] which provides a very high level of abstraction. Databases, communication channels, software patterns, component interfaces, or data structures are captured at this level, although models are not bound to any technology in particular, which facilitates the creation of different platform-specific models.[9,10] The second set of models describe systems at the lowest level of abstraction. That is, they are bound to particular implementation technologies, such as a vendor specific database (e.g., Oracle, SQL Server, PostgreSQL, and so forth), a communication protocol (e.g., HTTP, IIOP, RMI, and so forth), or an application framework (e.g., Java EE, .NET, and so forth). Therefore, operations, structure, and behavior of models must include details on how they can be implemented by employing the chosen technologies.

In recent years, domain-specific languages (DSL) are becoming increasingly popular in the software industry. Many authors argue that DSLs can bring important advantages over general-purpose languages. To name a few, they help raise the level of abstraction by providing language constructs that are very close to the problem domain; they are smaller and easier for software engineers to learn and use; they are more expressive; they increase productivity and quality, while reducing maintenance efforts.[11-16]

Enterprise application integration (EAI) is related to the development of integration solutions, which aim at helping independently designed applications collaborate to support new business processes. A typical integration solution may integrate the online sales system along with the on-premises inventory and accounting systems. Camel,[17] Spring Integration,[18] and Mule[19] shine amongst the state-of-the-art technologies available for software engineers to design and implement integration solutions. They got inspiration from the catalogue of integration patterns of Hohpe and Woolf,[20] which has been adopted by the integration community as a cookbook.

The integration community has been working hard to shift such technologies from a code-centric to a model-centric development approach. Thus, these technologies have been endowed with modeling languages in the last years. However, the resulting models are tightly coupled to the underlying integration platform.

According to a recent report by Gartner,[21] technology has evolved very fast. Not only integration platforms are frequently updated, but also new ones are expected to be launched in the short-term. Furthermore, the number of players in the market will grow considerably and the tendency is the reengineering of integration platforms to provide them cloud services.[22] In this scenario, the usage of integration technologies focused on platform-specific models to devise application integration solutions shall mean higher migration costs.

A platform-independent, DSL based on integration patterns is necessary to enable the design of platform-independent models. Such a language would have a positive impact on a model-centric development approach, and on the design of models that minimize costs for platform migration and platform independence, something achieved through the large-scale reuse that those kind of models provide. Model-Driven Engineering provides transformations that can be used to automate the software development process by translating models from one abstraction level to another, and by obtaining a skeleton or a complete executable system for a specific integration platform. The next generation of integration platforms are expected to be model-centric.[23]

In this article, we introduce i2Factory, which is a cloud-based platform to support the design, implementation, execution, and monitoring of EAI solutions by employing Guaraná DSL.[24] Together, they have proven to provide a convenient means to design and deploy EAI solutions. A cloud-based platform is a platform that is accessed through the web browser. The user can create, edit, compile and run the solutions through his or her browser, without installing any i2Factory-related piece of software on his or her computer. A major feature of this proposal is that it adheres to the principles of Model-Driven Architecture, so it makes a clear separation between platform-independent integration models and platform-dependent models into which they are transformed in order to be deployed to the cloud. The rest of the article is organized as follows: Section 2 introduces related integration platforms; Section 3 provides an overview of the Guaraná DSL; Section 4 provides an overview of the cloud-based platform that supports it, which is provided by a spin-off called i2Factory, S.L.; Section 5 demonstrates how this platform can be used to model an integration solution to a real-world integration problem; Section 6 reports on the results of our implementation; and finally, Section 7 presents our conclusions.

2 | RELATED WORK

In this section, we review the technical and scientific literature, which includes proposals developed in both: industry and academy. i2Factory is a software tool delivered for industrial purposes. In the industry, our review was driven by the 2019 Gartner Magic Quadrant,[21] which ranks the best-known providers of Enterprise Integration Platforms as Services (iPaaS) in the market, representatives of the state-of-the-art technologies. Our review for academic proposals was conducted at SCOPUS and DBLP databases and it aimed at identifying proposals of integration platforms which were either complete and providing supporting tools (that allow them to be experimented), or only conceptual proposals.

2.1 | Industry proposals

Gartner, Inc. is a world-wide company that provides information regarding technology and how it is used across the world. Its reports have driven many technology and business decisions for years. The 2019 Magic Quadrant[21] ranks many providers of Enterprise Integration Platforms as Services (iPaaS) along two axes: ability to execute and completeness of vision. The former assesses the ability of iPaaS providers to deliver platforms that fulfill the expectations of software engineers and drive success in their projects; the latter assesses the capability of iPaaS providers to support emerging requirements, lead the market, and grow a profitable and self-sustained business. Platforms are also classified into four profiles, namely: niche players, visionaries, challengers, and leaders. Niche players are recent start-ups or small companies with excellent technology and very satisfied customers. Visionaries know the specific requirements of the iPaaS market and innovate with new market strategies. Challengers have been in the market for several years and have hundreds, if not thousands, of clients. However, they have a limited perspective on how iPaaS market will evolve, which results in more narrowly focused offerings when compared with their competitors. Leaders have thousands of clients, solid reputations, notable market presence, and their platforms are well-proven and functionally rich, with regular releases in order to quickly address emerging requirements. We shall briefly present the platforms classified as leaders by Gartner, Inc. in this section.

Founded in 2000 in the USA, Boomi[25] became part of Dell's universe in 2010, but kept operating as an independent business. This platform was first introduced as an iPaaS, and it was able to support integration application processes between cloud platforms, software-as-a-service applications, and on-premise systems. Boomi offers a visual designer with prebuilt connectors, where users build integration processes by pointing-and-clicking, dragging-and-dropping, with as little coding as possible. Solutions are deployed into a dynamic run-time engine. It has a centralized management for all integration solutions, no matter if they are deployed on a cloud or on-premise. Boomi provides a tool to test and watch the process while running. It includes connectors for well-known applications such as Dropbox or Jira, as well as standard connectors such as FTP or HTTP.

Informatica[26] is a private company founded in the USA in 1993. This platform provides not only a visual designer with a drag-and-drop web interface and self-service wizards, but also connectivity to applications on the cloud, and connectivity to on-premise applications and databases. Informatica also provides wizards, preconfigured templates, and out-of-the box mappings. Developers use the design canvas to drag and drop data sources, targets, and advanced transformations. The platform allows to manage the state of integration and business processes, whether they are synchronous, asynchronous, long, or short-running. It includes connectors for well-known applications such as Dropbox, Google APIs, Jira, Microsoft Sharepoint, Salesforce, and standard protocols such as FTP, ODBC, and REST.

JitterBit[27] was founded in the USA in 2003. This platform provides a graphical interface which is able to reuse existing code and business logic, point-and-click connectivity, drag-and-drop configuration, and prebuilt templates; it also helps to endow existing applications with some artificial intelligence. Deploying a solution may be accomplished either on the cloud, on-premise, or by using a hybrid approach. It is also possible to reuse any application or code in JitterBit. This platform claims that its management data may be moved across applications, real-time analytics with consolidated data, real-time monitoring with alerts, and team permissions. It includes connectors for well-known applications, such as Dropbox, Gmail, Jira, Microsoft SQL Server, as well as standard protocols such as FTP, HTTP, and ODBC.

Microsoft[28] was founded in 1975 in the USA. With its Azure Logic Apps, it joined the enterprise iPaaS market in 2016. Nowadays its main offering is Azure integration services, consisting on four components: Azure API Management, Azure Logic Apps, Azure Service Bus, and Azure Event Grid. These cloud services may be used to integrate cloud and on-premises applications, sometimes combined with other cloud technologies. Azure Logic Apps provides hundreds of connectors, from cloud applications such as Office 365 or Salesforce CRM to on-premises technologies such as Sharepoint or SAP or even standard protocols such as FTP or SMTP. A few reference customers identified some improvement needs, especially in data transformation and mapping, EDI support, life cycle management of integration artifacts, automated testing, and metadata discovery.

Mulesoft[29] was founded back in 2006 in the USA. This platform is an open-source proposal that combines cloud-hosted and on-premise integration. Mulesoft enables integration of software as a service and on-premise applications, as well as API management. A repository for connectors, templates, and APIs is available and might be enriched by users. Mulesoft includes connectors for well-known applications such as Dropbox, Jira, Microsoft Sharepoint, or Salesforce, and standard protocols such as FTP, HTTP, or JDBC.

Oracle was founded in the USA in 1977, and it provides an iPaaS.[30] Solutions are developed by point-and-click in a browser-based visual designer editor that allows to publish APIs, which can be externally consumed. Users have access to connectors from all Oracle SaaS applications, native SaaS adaptors to integrate with other cloud applications, and integration with on-premise applications. It is possible to monitor transactions and key performance indicators; one can also detect errors and diagnose them. Customers have access to prebuilt integrations that can be used as-is or customized, and also to a Cloud Marketplace where prebuilt adapters and integrations are traded. Developers have access to a set of connectors to well-known applications, including GMail, Microsoft SQL Server, or SAP R3, and standard protocols such as FTP, JDBC, or REST.

Snaplogic[31] was founded back in 2006 in the United States. Its platform provides a web-based user interface and a set of adapters, integration flows, and another set of patterns to be used by integrators via drag-and-drop. It integrates applications or data on the cloud, on-premise, or by using hybrid approaches. Snaplogic guarantees data requests delivery by automatically monitoring them, to ensure data delivery as well as compliance with service level agreements, company policies, and regulatory requirements. Centralized object level, granular security, and permissions enable integration to be extended to customers' organizations. This platform includes connectors to ERP, CRM, identity management, on-line storage, relational, columnar and key-value databases, as well as standard protocols such as FTP, REST, or OAuth.

Workato[32] was founded in the USA in 2013. 100% Cloud Native platform, it allows to create and operate integrations from any device or browser. Being an early adopter of Artificial Intelligence, Machine Learning, and chatbot technology, it supports more common scenarios such as data integration or API management, and more advanced ones such as chatbot-enabled human tasks or Robotic Process Automation. It also supports on-premise-centric use despite being a cloud platform. However, such solutions may become problematic in extremely low-latency requirements scenarios in which data must be processed in a customer's environment. Workato provides users with thousands of prebuilt "recipes" (solutions ready to run, easily customizable according to Workato itself). It also includes connectors ("apps") to standard protocols such as HTTP, FTP, XML parser and also to well-known applications such as SQL Server, My SQL, Google Drive, or Trello.

Table 1 presents a comparison on industry integration platforms according to their main features, and from developers' point of view: (1) Development and deployment on the Cloud or On-Premise; (2) Debugging features at the integration solution abstraction level; (3) Availability of predefined integration templates to foster the development of integration solutions; (4) Support for Offline or online/Cloud development environment; (5) Real-time monitoring support covering the integration solution and the inflow and outflow rates from tasks, and resources consumption; (6) Team work support features such as users and their profile management, as well as access control; (7) Independent DSL Platform, which allows DSL integration solutions specification to be implemented in more than one EAI development environment. We use n.a. if the integration proposal has no documentation available for that feature.

When comparing the vendor-lock property of all integration platforms, we got to the conclusion that all of them are vendor-lock, despite the fact that some of them export the solution integration specification in formats such as XML documents. As XML integration solutions use vendor specific (DSL) tasks, patterns, syntax and semantics, and there are no explicit/documented mappings for conversions between those specifications, we assume all platforms are vendor-lock in practice.

TABLE 1 Industry integration platforms

Platforms	Cloud (C) On-Premise (O) Hybrid (H)	Graphical native debug	Predefined templates	IDE type	Real-time monitoring support	Collaborative work support and team management	DSL platform independent
Boomi	COH	No	Yes	on-line/off-line	Full	Yes	No
Informatica	C	n.a.	Yes	on-line/off-line	Full	n.a.	No
JitterBit	COH	No	Yes	on-line/off-line	Basic	Yes	No
Microsoft	COH	No	n.a.	off-line	Full	n.a.	No
Mulesoft	COH	No	Yes	on-line/off-line	Full	n.a.	No
Oracle	COH	No	Yes	on-line/off-line	Basic	n.a.	No
Snaplogic	C	n.a.	No	on-line	Full	n.a.	No
Workato	C	n.a.	Yes	on-line	Full	Yes	No
i2Factory	C	Yes	Yes	on-line	Full	Yes	Yes

Abbreviation: DSL, domain-specific language.

2.2 | Academic proposals

For academic proposal research we used SCOPUS and DBLP databases. We employed the search string "(integration frameworks OR system integration OR integration tool) AND (application integration)," for articles published between 2010 and 2019, written in English, in the subject areas of Computer Science and Engineering. The search returned 108 unique results through a diverse range of journals and conferences. Next, we carefully reviewed all titles and abstracts from those 108 articles and we discarded the unrelated ones. Unrelated proposals represent articles that include at least one of the keywords from the search string, but are not related with EAI. At the end, 15 articles were left and then grouped by year, as shown in Table 2.

Asunción et al.[33] worked on a proposal to increase flexibility and to develop integration solutions based on service mediation. Their proposal was inspired by the Model-Driven Engineering and attempts to separate business rules from business processes supported by integration solutions. Although the authors show an application of their proposal in a specific scenario, it is just a conceptual model for a service integration framework. Marhaim et al.[34] proposed a method and a visual tool targeted for software designers to build integration solutions. In this proposal, integration is realized at the user interface layer, by using techniques to navigate and extract information from source applications and by filling out target applications. Li et al.[35] introduced an open-source integration platform that was based on the orchestration of services, which performed application integration while targeting networked companies. The proposed platform manages

TABLE 2 Articles found in the scientific literature review

Year	Articles found	Articles selected	Reference
2010	22	3	33, 33-35
2011	19	0	–
2012	15	1	36, 36
2013	18	0	–
2014	7	1	37, 37
2015	10	5	38-42
2016	6	2	43,44
2017	6	0	–
2018	2	1	45
2019	3	2	46,47
	108	15	

a set of services known only through unique identifications and is able to decouple services from the platform. In this approach, integration solutions are described by means of orchestration processes, which are designed through a graphical modeling tool and run on an independent Java-based run-time system. The authors also demonstrate their proposal by two cases applied in the industry.

Kovanovic and Djuric's[36] proposal focused on tool support to design platform-independent integration solutions. They argue that the key to evolve in the support for application integration is to provide abstract languages, which in turn support well-known integration patterns in the EAI community. Highway is their internal DSL, built on top of the programming language Clojure, to enable the usage of functional programming in the context of application integration. This language is accompanied by an application framework which builds integration solutions. Pinho et al.[37] proposed an extensible integration platform focusing on the integration of cloud services to be used in web applications. The authors claim that one of the main features of their proposal is that it implements access control policies and mechanisms for sharing and delegating resources. In their platform, the resulting integration solution is deployed as a cloud service, which can also be managed and integrated with other services by their platform.

Kern et al.[38] targeted the challenges of application integration by proposing an integration framework built on top of the model-driven development paradigm. They claim that building an integration solution is a time consuming, costly, and error-prone task, so that a structured and automated way to build the solutions is required. Their integration framework includes a declarative mapping language with a graphical notation that allows transformations between data schema. Reuse of integration knowledge is also targeted by their framework, which maps already developed integration solutions in a transparent way, so that they can be used in new projects. Balko and Barros[39] discussed how the adoption of in-memory databases have impacted traditional integration platforms. They propose a new integration framework that aims at connecting applications by means of a shared in-memory database. It includes a run-time system and a compiler infrastructure, which maps business process definitions into native artifacts located on the in-memory database. According to them, running in-memory integration solutions not only helps to improve performance, but also allows to reuse fundamental features such as transaction concurrency control, failover, persistence, backup, and recovery. Zheng et al.[40] proposed a domain-specific integration framework focusing on health-care, targeting the integration of clinical information systems and clinical decision support applications. The authors validate their proposal in a real-world clinical environment.

Integration of globally distributed applications is picked out by Hanson et al..[41] In their article, they introduce P2PIE, an integration platform to design and implement point-to-point integration solutions. P2PIE relies on the messaging integration style, which also allows for keeping a central control and configuration environment for managing and monitoring features, despite the point-to-point connection from integrated applications. The authors report the use of their platform for more than 6 years in the industry, by demonstrating how it outperforms other well-known commercial integration platforms. Wei[42] combines service oriented architecture (SOA) and web service technology to improve information sharing and business adaptability. He assumes every application in the software ecosystem provides a web service communication layer to be accessed; also, that it works on the transformation and adaptation of information that must be shared amongst applications. The author introduces an integration platform that relies on messaging integration style to reduce maintenance costs and improve efficiency of the integration solutions when sharing information, claiming his approach makes enterprises more competitive. Xu et al.[43] propose an integration framework based on web services to exchange data amongst different applications. In their proposal, applications are endowed with a web service layer, so that data can be either read from or written to the application, transformed by another web service and sent to a target application. According to the authors, web services can decouple integrated applications in a secure and easy way. The application integration focused on data exchange is also addressed by Chen,[44] who introduces an integration framework that enables connecting applications to the implementation of a business process, so that the process can be fed with data and also produce data to the integrated applications in the expected format. In this proposal, web services also play a central role in easy communication amongst applications and follow standards in the field of systems integration.

Beer and Hassan[45] claim that current frameworks which build integration solutions based on SOA and representational state transfer (REST) web services lack security in their architecture, so that messages exchanged may have their integrity broken. Their proposal includes a run-time component called "intelligent security engine," which can be attached to an integration platform to intercept and analyse messages with the purpose of detecting security threats. Artificial neural networks are used by the proposed engine to predict security vulnerabilities during the execution of the integration platform. Their proposal focuses on a very specific point (security) inside the run-time engine. de Souza Cimino et al.[46] propose an event-base middleware solution, which focus on the integration of Internet of Things sensors

TABLE 3 Comparison between academic integration platforms and i2Factory

Platform references	Integration patterns	DSL	Ad hoc	Integration type	Includes a support tool (IDE, Plug-in)
Asunción et al.[33]	No	Yes	Yes	Messaging	No
Marhaim et al.[34]	No	No	Yes	Record/replay engine	IDE
Li et al.[35]	No	No	Yes	ESB	Graphical Tool
Koanovic and Djuric[36]	Yes	Yes	No	Messaging	n.a.
Pinho et al.[37]	No	No	Yes	Publish and subscribe	No
Kern et al.[38]	No	Yes	n.a.	Mapping	Plugin or module
Balko and Barros[39]	No	No	Yes	In-memory database	No
Zheng et al.[40]	n.a.	n.a.	n.a.	n.a.	n.a.
Hanson et al.[41]	No	No	n.a.	Distributed Message Broker	Management Console
Wei[42]	No	No	No	n.a.	No
Xu et al.[43]	No	No	Yes	Web Services	No
Chen[44]	n.a	No	n.a	Database	No
Beer and Hassan[45]	n.a.	n.a.	n.a.	n.a.	n.a.
de Souza Cimino et al.[46]	No	No	Yes	ESB	n.a.
Huang et al.[47]	Yes	No	Yes	Messaging	No
i2Factory	Yes	Yes	No	Messaging	IDE

Abbreviation: DSL, domain-specific languages.

and actuators with high-performance computing services. The integration workflow allows the specification of asynchronous message communication with the integrated sensors and actuators—a key feature to their proposal. The authors claim their proposal has shown to be a feasible middleware with a good performance and low consumption of memory and CPU. However, they register it introduces small overheard when integrating sensors and actuator from distinct networks. Huang et al.[47] start introducing an architecture to be followed by integration platforms that target on supporting integration, in contexts of large volumes of data. They provide a high-level description on an integration system they have developed following their architectural proposal and run it in the ecosystem of a telecom company, which highly demands data exchange and has facing challenges due to the increasing volumes of data concerned. Their integration approach focuses on the integration of front-end and back-end systems, reason why they claim their proposal not only reduces the complexity involved in the design of integration solutions, but it also enables the achievement of service-level agreement at EAI when it comes to large volumes of data.

Table 3 presents a comparison on academic integration platforms proposals according to their main features and i2Factory platform reference features, from the developers point view: whether it uses (1) Integration patterns; (2) New or adopted existing DSL; (3) Ad hoc integration development; (4) Integration type (message-based, enterprise service bus, publish and subscribe, mapping, in-memory database, distributed message broker, web services); (5) Provides a full integrated development environment, a plug-in for existing IDE or specific tools. We use n.a. when the integration proposal does not present any documentation available for that feature. Although i2Factory integration platform does not belong to the academic platforms group, for the sake of comparison between academic and i2Factory platform features, we also included i2Factory in this table.

3 | THE GUARANÁ LANGUAGE

Guaraná is a DSL that provides graphical notation in order to design platform-independent models for EAI solutions. It is the result of a 6-year research project to provide new languages, methodologies, and tools to help integration engineers reduce the costs involved in the design and implementation of EAI solutions. A detailed discussion of the Guaraná DSL and an introduction on the academic integration framework to support the implementation and execution of models designed with Guaraná is presented by Frantz et al.[24,48] In this article, we focus on the cloud-based platform to be used in the industry to design, implement, execute, and monitor integration solutions on the Internet. In the following, we provide an overview of the main constructors of Guaraná in order to understand our proposal, namely:

Message:	An abstraction of a piece of information that is exchanged amongst applications and transformed across an integration solution. It is composed of a header, a body, and one or more attachments. The header includes custom properties and some predefined properties: message identifier, correlation identifier, sequence size, sequence number, return address, expiration date, and message priority. The body holds payload data. Attachments allow messages to carry extra pieces of data associated with the payload, for example, an image or an e-mail message.
Task:	An implementation of an integration pattern,[20] for example, split, aggregate, translate, chop, filter, correlate, merge, resequence, replicate, dispatch, enrich, slim, promote, demote, or delay. Roughly speaking, a task may have one or more inputs from which it receives messages rom ports or tasks, and one or more outputs by means of which messages are delivered to other tasks or ports. Tasks are classified as stateless or stateful, depending on whether the work that they carry out on a particular message is independent from previous and future messages or not.
Slot:	A buffer that connects the output of a task or a port with the input of another task or port. They support several policies to serve messages, including priority-based and first-come, first-served. They are the key for tasks to process messages as asynchronously as possible.
Port:	An interface that abstracts the required details to interact with resources within a software ecosystem. There are four types of ports: entry, exit, solicitor, and responder.
Integration Solution:	A workflow that aggregates a number of ports, tasks, and slots. Conceptually, it is a workflow that routes messages read from entry ports through a process that transforms them and writes the results to exit ports; through the workflow, additional information may be either gathered by means of solicitor ports or delivered on-demand by responder ports.
Resource:	Represents an information source or sink that usually belongs to an application, such as data files, databases, APIs, or even user interfaces. Resources exist prior to integration solutions and are not changed at all when they are integrated.

Table 4 shows the concrete syntax for the constructors in Guaraná. The small rounded connectors on the sides of each task icon represent the inputs and the outputs. Slots are connected to tasks through those rounded connectors. Note that we have included the notation we use, in order to represent resources which are being integrated. In the table, this notation does not specify which layer (database, channel, file, API, user interface, and so forth) is being used to communicate the integration solution to the resource. Messages do not appear in this table, because they are not part of the conceptual model; they only exist when they flow at run-time.

We classify tasks according to their intended semantics, namely: routers, which do not change the state of the messages they process, but route them through a whole process; modifiers, which help to add or remove data from messages, but do not alter their schemata; and transformers, which help to transform one or more messages into a new one with a different schema. In the appendix, Table A1 presents the routers, Table A2 presents the modifiers, and Table A3 presents the transformers.

TABLE 4 Guaraná concrete syntax

Notation	Concept	Notation	Concept
▭	Task	┌▲▼┐	Solicitor Port
……▶	Slot	└▲▼┘	Responder Port
▶	Entry Port	□	Integration solution
◀	Exit Port	▭–ⓧ	Resource

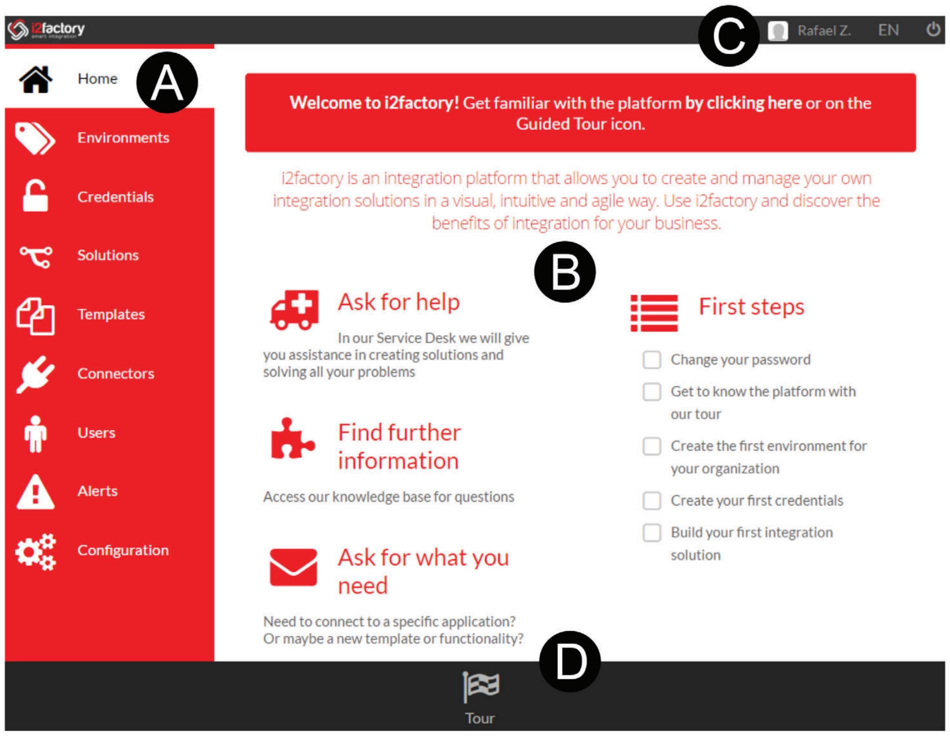
4 | THE CLOUD-BASED PLATFORM

i2Factory is a cloud-based web development environment, available under a commercial license, that allows for the design, execution, and monitoring of integration solutions in Guaraná. It is a joint effort between academy and industry in a research project [1]to develop a disrupting tool, built on the principles of the Model-Driven Architecture—to create integration solutions on the Web at a high level of abstraction. The technology was developed by the first couple of authors with the aid of other researches. Its original and academic name was replaced by the commercial name i2Factory and it is currently own by investors. i2Factory has an intuitive web-based user interface, cf. Figure 1. Panel (A) provides the main modules. Panel (B) provides access to information regarding Guaraná and the features of i2Factory, which includes detailed documentation, tutorials, sample integration solutions, and on-line support. Panel (C) provides information regarding the user who is logged in, as well as a button to change the language of the interface and an exit button. Panel (D) presents a contextual toolbar, which lists the set of commands available in each interface. In the start interface, this toolbar provides a single command that leads users through all main features available to design, run, and monitor integration solutions. A help command is provided in every toolbar (except for the start interface), so that it starts a wizard that explains the current interface and helps users with the available commands. Currently, the platform provides the following modules:

Environment:	the virtual server where integration solutions run. The platform allows for the creation of two types of servers depending on the cloud where they are going to be deployed: *i2Factory Cloud*, which is supported by i2Factory cloud machines, and *Customer Cloud*. The latter simply requires the installation of i2Factory server in customer's machines. The environment provides all the required computational resources to run integration solutions. In the server it is possible to enable debugging support, specify hardware settings, and choose a geographic location.
Credentials:	a repository of authentication data which allows connectors to perform their communication protocol. This module stores authentication data that can be used by integration solutions. A variety of credentials are supported for interoperation with several cloud and Internet services, as well as enterprise software services and packages, including Dropbox, Twitter, GMail, Alfresco, Salesforce, Zoho CRM, and so forth. A wizard helps users to set up protocols and to configure credentials, so as to facilitate integrating in as many applications as possible.
Solutions:	a repository of integration solutions. This module provides features to design, debug, run, and monitor integration solutions. Users can create them by starting either with an existing template or from scratch.
Templates:	a repository of reusable solutions. This module stores integration solutions or pieces of them as reusable assets, which in turn serve as the starting point for the design of a new integration solution.
Connectors:	a connector implements a specific communication protocol that a port requires to interact with a resource being integrated. This module gives access to the list of available connectors, allowing users to purchase new connectors or to disable the ones that are no longer needed.
Users:	a set of system users to whom access to the integration platform is granted. This module allows to manage user accounts (creation, settings, removal, and so forth).
Alerts:	an alert is a message containing information to the users. This module provides access to notifications and alerts of important events raised by integration solutions.
Configuration:	the arrangement of complementary information regarding the user who owns and manages the server and the solutions. This module allows to configure additional user profile details, for example, addresses and contacts.

We focus on the solutions module because it is the core of our proposal, cf. Figure 2. Panel (A) shows the list of integration solutions available. Every column displays the name of the solution, the state (both at design time and execution time), creation date, the user who created it, and the environment by which it is associated. The design status indicates whether the solution is syntactically valid or not, if it has been modified and needs to be redeployed, or if it is locked because another user is working on it. The running status indicates whether the solution is running or not. A contextual toolbar is available in Panel (B); when the *new* command is clicked, Panel (C) is opened in the right side to ask users whether they wish to design a solution from scratch or start it with a predefined template. The toolbar in this interface also provides commands to edit, move, copy, delete, detail, start, stop, restart, and clear integration solutions.

[1]i2Factory on European Commission Horizon 2020: https://cordis.europa.eu/project/id/762147.



Welcome to i2factory! Get familiar with the platform **by clicking here** or on the Guided Tour icon.

i2factory is an integration platform that allows you to create and manage your own integration solutions in a visual, intuitive and agile way. Use i2factory and discover the benefits of integration for your business.

Ask for help

In our Service Desk we will give you assistance in creating solutions and solving all your problems

Find further information

Access our knowledge base for questions

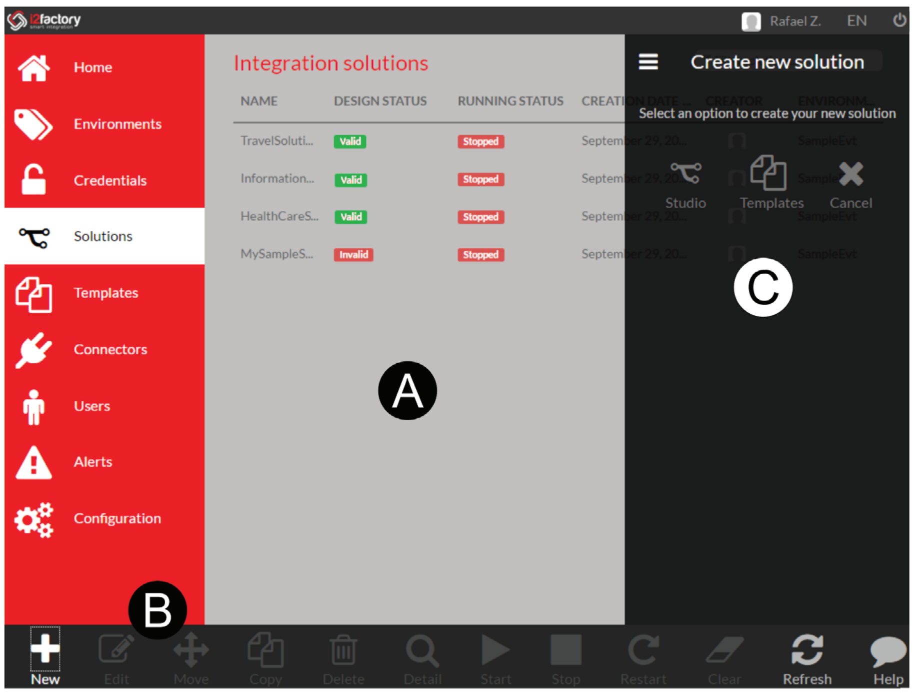
Ask for what you need

Need to connect to a specific application? Or maybe a new template or functionality?

First steps

- ☐ Change your password
- ☐ Get to know the platform with our tour
- ☐ Create the first environment for your organization
- ☐ Create your first credentials
- ☐ Build your first integration solution

FIGURE 1 Start interface of i2Factory [Color figure can be viewed at wileyonlinelibrary.com]



Integration solutions

NAME	DESIGN STATUS	RUNNING STATUS	CREATION DATE
TravelSoluti...	Valid	Stopped	September 29, 20...
Information...	Valid	Stopped	September 29, 20...
HealthCareS...	Valid	Stopped	September 29, 20...
MySampleS...	Invalid	Stopped	September 29, 20...

Select an option to create your new solution

FIGURE 2 Solutions module [Color figure can be viewed at wileyonlinelibrary.com]

When a new integration solution is created, or an existing one is edited, the interface shown in Figure 3 pops up to the user. The integration solution is modeled in Panel (B) by dragging and dropping ports and tasks from Panel (A). There is a great variety of connectors available in the icon expanded in Panel (D), for example, Dropbox, GMail, Blogger, WordPress, HTTP, SSH, POP3, LDAP, and so forth. Connectors implement a communication protocol inside ports to interact with different applications which are being integrated. A set of tasks is available in Panel (C). As of the time of writing this article, i2Factory supported six router tasks (correlator, dispatcher, filter, merger, replicator, and threader), seven modifier tasks (context-based enricher, context-based slimmer, header demoter, header promoter, set correlation ID, set return address, and slimmer), and five transformer tasks (aggregator, assembler, chopper, splitter, and translator).

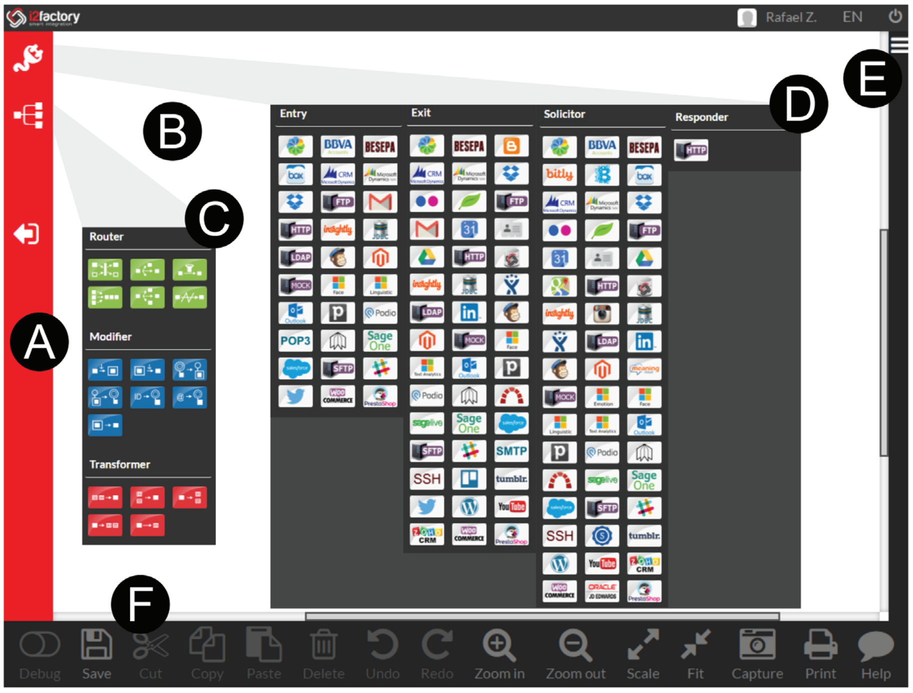
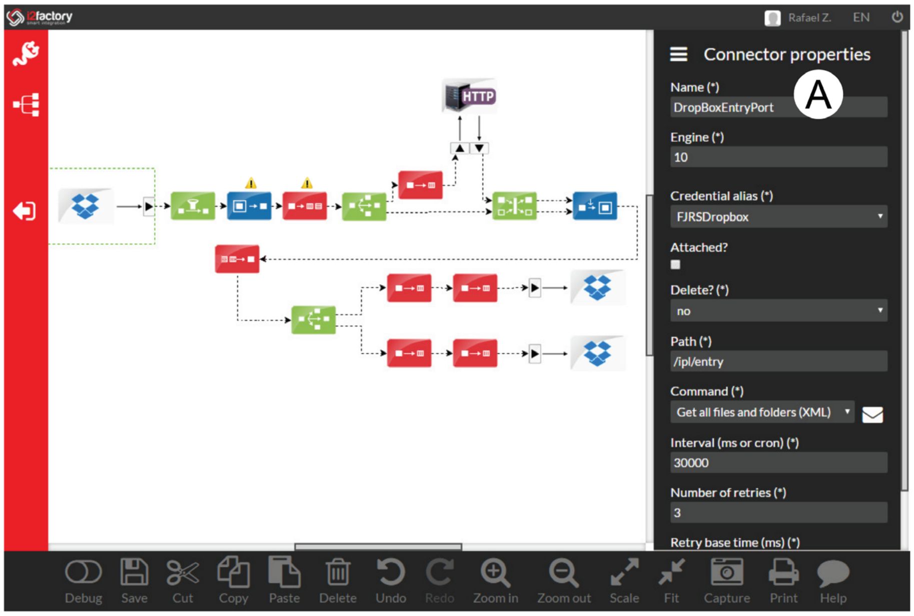


FIGURE 3 User interface to model integration solutions [Color figure can be viewed at wileyonlinelibrary.com]

The three horizontal white bars in Panel (E) open a form to set the properties (name, description, and environment) from the integration solution. The toolbar in Panel (F) provides commands to debug, save, cut, copy, paste, delete, undo, redo, zoom in, zoom out, scale, fit, capture image, and print the integration solution modeled in Panel (B).

In the drawing panel, software engineers use the mouse to link ports and tasks with slots in order to set up the integration flow from the modeled integration solution, cf. Figure 4. Every port and task must be configured by clicking on them, so that contextual Panel (A) pops up on the right side of the screen to show available properties for the selected element. The state of an integration solution changes to *valid* only after completing the design and the configuration of every element; meanwhile the platform presents a warning icon on elements that have not been fully configured yet.

Users can activate debug mode for an integration solution, only by clicking on the first command (*debug*) in the contextual toolbar shown in Figure 4. This toolbar changes and adapts itself to show other commands that are necessary to control a debugging process, for example, run, step-by-step, stop, add breakpoint, or delete breakpoint. Breakpoints



Connector properties

Name (*)
DropBoxEntryPort

Engine (*)
10

Credential alias (*)
FJRSDropbox

Attached?

Delete? (*)
no

Path (*)
/ipl/entry

Command (*)
Get all files and folders (XML)

Interval (ms or cron) (*)
30000

Number of retries (*)
3

Retry base time (ms) (*)

FIGURE 4 An example of integration solution modeled using Guaraná [Color figure can be viewed at wileyonlinelibrary.com]

can be set to any port or task, just by selecting the element and clicking on the command. The debug mode allows the inspection of messages that entered the solution and were either stored in slots or processed by tasks and ports. Figure 5 shows the user interface for the execution of an integration solution in debug mode. In this mode, tasks are decorated with the number of messages that they have processed and an icon that indicates whether they have executed them properly or not.

Ports, tasks, and slots can be inspected by users just by clicking on them. Contextual Panel (A) on the right side shows the inspected element properties. Figure 5 shows the results of inspecting a slot; it is necessary to select one from the stored messages in Panel (B) and then its contents are shown in Panel (C). When in debug mode, users cannot modify the integration solution model until such mode is deactivated. This is done to preserve the model and to keep the current running state of the integration solution.

Integration solutions can be monitored so that users can have information about their executions, including their performance, the number of incoming and outgoing messages, details on messages, event logs, and so forth. Figure 6 presents the monitoring interface, which is opened by clicking on the *detail*command in the interface shown in Figure 2. The menu

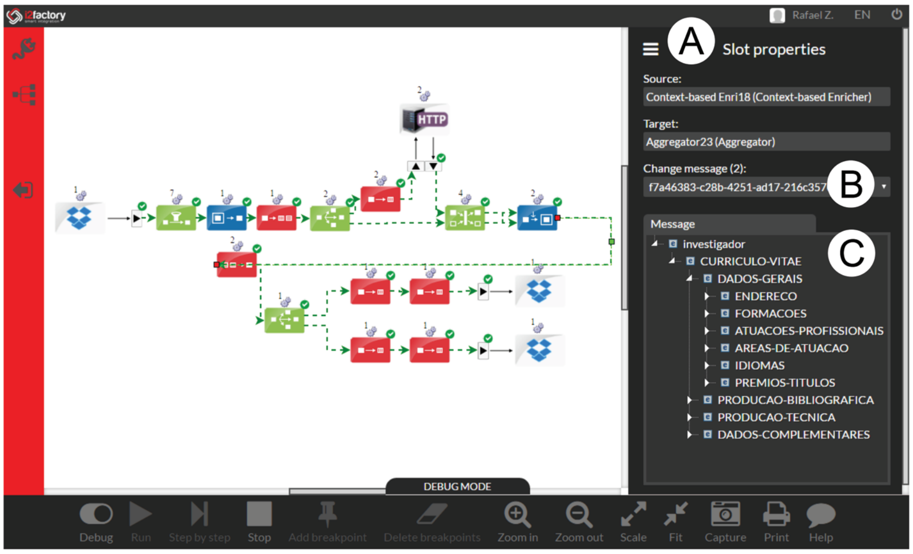


FIGURE 5 User interface to debug integration solutions [Color figure can be viewed at wileyonlinelibrary.com]

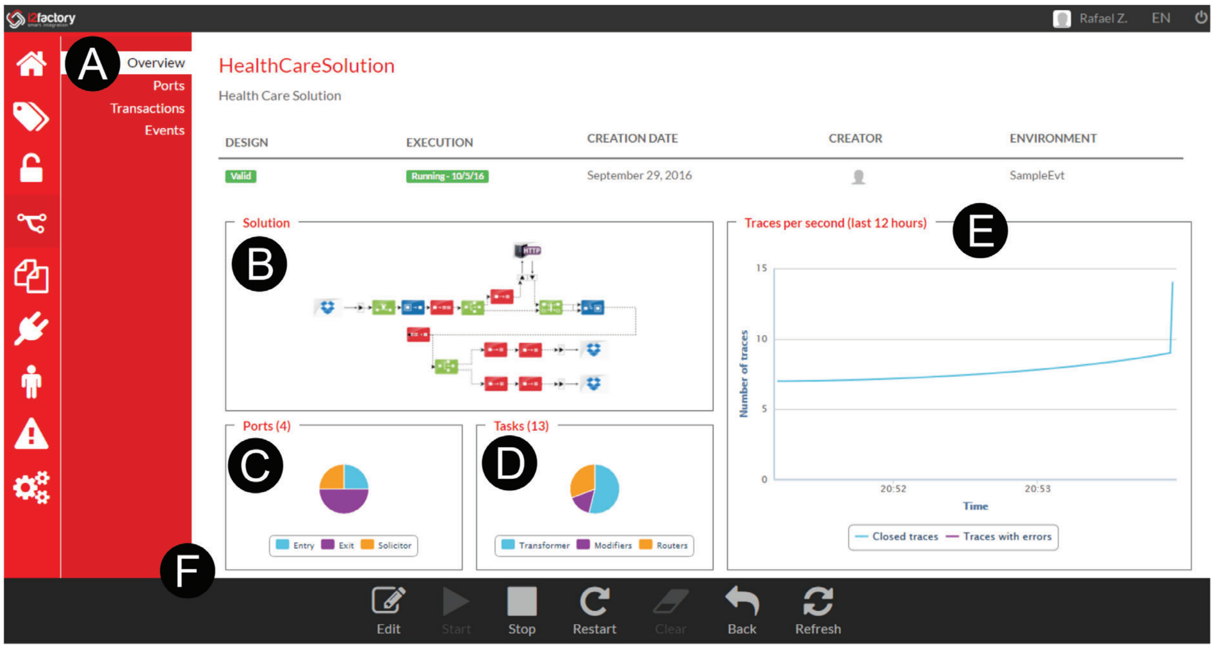


FIGURE 6 User interface to monitor integration solutions [Color figure can be viewed at wileyonlinelibrary.com]

listed in Panel (A) has the *overview* option selected by default, but it also provides options to get detailed monitoring information regarding ports, message transactions, and event logs. In the interface used to overview the integration solution, Panel (B) shows the complete model for the running integration solution; Panel (C) summarizes the types of ports used in the integration solution; Panel (D) summarizes the types of tasks used in the integration solution; and Panel (E) provides a real-time graph from the number of messages per second that were processed the last 12 h. In the contextual toolbar located in Panel (F) of the monitoring interface, there are commands to edit, start, stop, and restart integration solutions.

5 | CASE STUDY

In this section, we show Guaraná and i2Factory in action. We present a solution to a real-world integration problem: the research outcomes management from a research unit in Portugal. A major share of scientific research in Portugal is carried out in Research and Development Units and Associate Laboratories, which are funded and evaluated by the Foundation for Science and Technology (FCT). In 2014, there were 292 Research and Development Units and 26 Associate Laboratories.[49] Some of these are public, and they are hosted by higher education institutions, whereas others are private, nonprofit organizations. There were more than 22,000 affiliated researchers.[50] Several research units are supported directly by FCT, both at national and international levels.[49]

To get funding from the FCT, research units must prove their relevance by means of research results and their scientific impact. From the perspective of management, this requires an accurate and efficient way to collect, organize, compile, summaries, and report information that is spread across many researchers' curricula and other national and international scientific repositories.

The integration solution presented in this section was deployed as a pilot project in the Computer Science and Communications Research Centre (CIIC)[2]of the Polytechnic Institute of Leiria (Portugal). It aims to replace the previous human-based process that consisted of collecting, computing, and updating the research units' production, and it was basically performed by researchers and civil servants with the help of some office-related software tools. The integration solution will facilitate that researchers have their list of publications up-to-date and that their participation in scientific events shall be properly recorded; it will also help classify publications according to their ranks, and then aggregate, summaries, and generate annual reports.

In the following section, we introduce the software ecosystem involved in the integration solution, then we present the Guaraná DSL model for this solution and its output.

5.1 | Software ecosystem

The integration solution involves four applications, namely: Researchers Registry, CIIC Web, DeGóis, and Scopus. Figure 7 abstracts the integration problem involving the applications in the software ecosystem. Researchers Registry and CIIC Web are in-house applications and were designed without integration concerns in mind; interacting with them demands access to their data layers. DeGóis[51] and Scopus[52] are external applications available on the Web; they provide REST APIs for higher education institutions and public purpose research projects.

Researchers Registry manages a local repository that provides basic data about researchers who are associated with a research unit. Amidst this data, there is information regarding their work contract and the URL for their curriculum vitae in DeGóis. Data layer of this application consists of large XML documents, which are stored in a file system that can be accessed via Dropbox.

CIIC Web is a content management application that helps publish information about a research unit on the Web; it includes publications, projects, awards, events, partnerships, and advanced training provided by its researchers. In this case study, the CIIC Web application was customized to the needs of this research center.

DeGóis is one of the core curricula repositories of Portuguese researchers. It was developed and is currently maintained by the FCT. In Portugal, every researcher has to be registered in this application and fill out information about his or her scientific publications and academic activities.

Scopus is amongst the most important international data sources of scientific research results. It belongs to Elsevier, a publishing and analytics company who runs this data source. Scopus offers a bibliography repository that not only

[2]Available at http://ciic.ipleiria.pt.

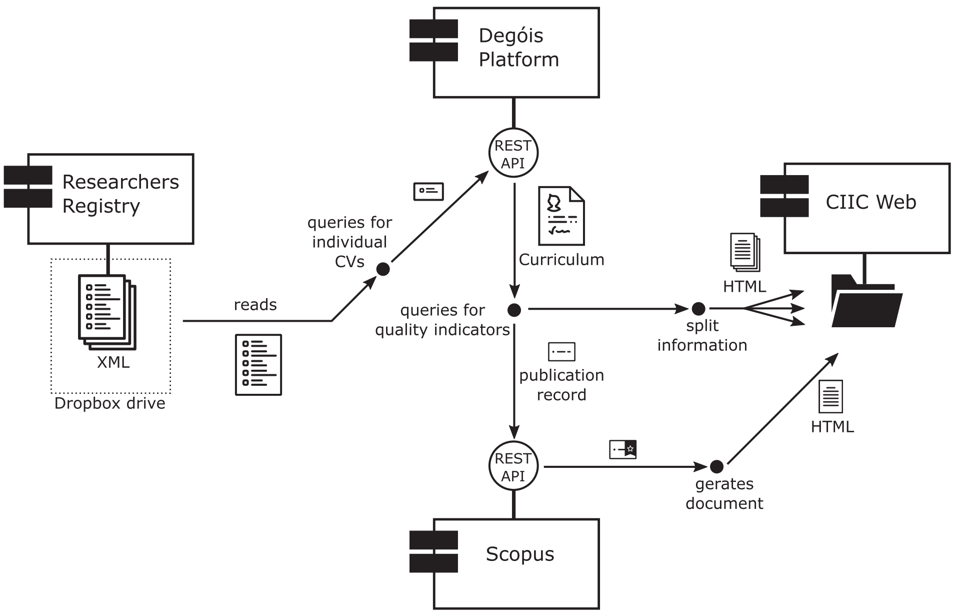


FIGURE 7 Abstraction of the case study integration problem

provides a search engine to find publications, but also includes citation and ranking information for articles, journals, and conferences.

The integration solution takes a list of researcher names as input; it then collects their curricula from DeGóis and relates the publications to quality indicators that are found in Scopus. The output consists of a set of HTML documents that are published by CIIC Web.

5.2 | Integration solution

The integration solution that we have devised is composed of nine ports to interact with applications that are being integrated in the software ecosystem, and 24 tasks to implement its integration logic. Figure 8 shows the integration solution designed with the Guaraná DSL through the i2Factory platform. Our goal with this case study is to show i2Factory in action to develop a solution for a real-world integration problem.

This solution involves collecting, enriching, and transforming data. Workflow starts at the entry port *P*1, which reads an XML file from a shared folder in Dropbox containing a list of researchers' names who work with the CIIC. A splitter task *T*3 is used to break this list into individual messages that carry the researchers' names. These messages are then routed to solicitor port *P*2, which makes requests to DeGóis. After fetching the corresponding curricula, messages are

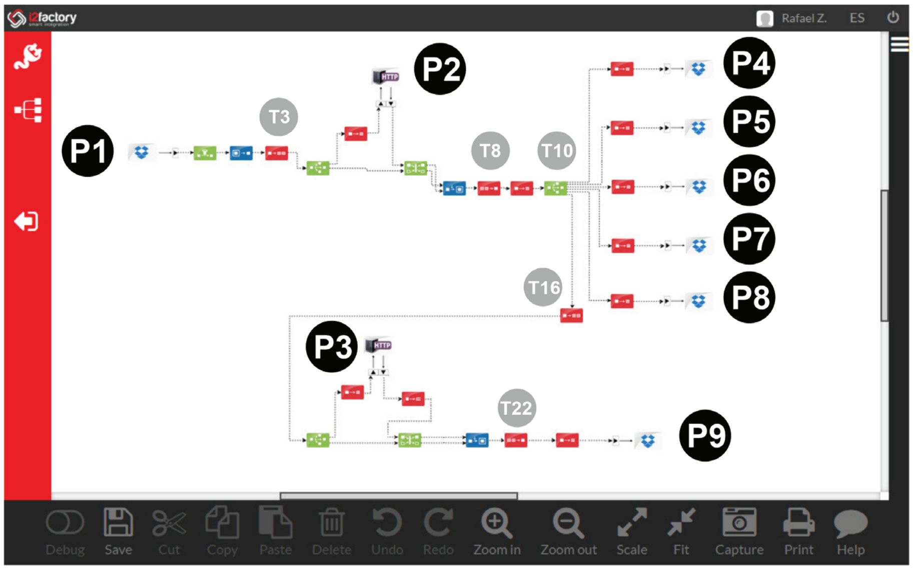
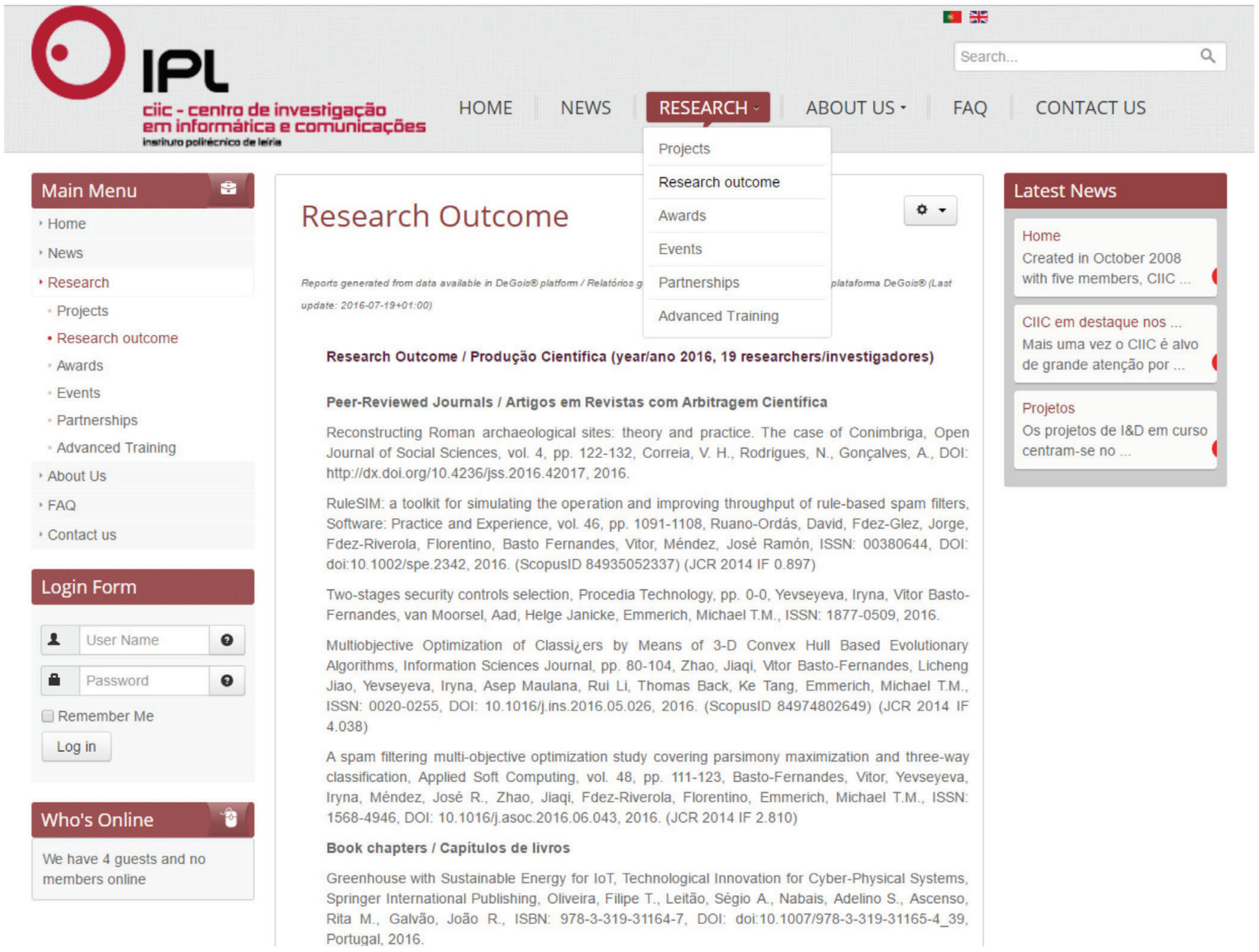


FIGURE 8 Solution to the case study integration problem [Color figure can be viewed at wileyonlinelibrary.com]

aggregated into a single message by the aggregator task $T8$, which is then replicated by task $T10$, so that translator tasks bound to exit ports $P4$–$P8$ build output HTML report files for each type of research activity, namely: projects, awards, events, partnerships, and advanced training. The sixth copy of the message is then split again by task $T16$. This operation creates several messages, containing each a single publication, so that Scopus is requested by HTTP REST solicitor port $P3$ to check whether publications are indexed or not; in the former, citations and other quality indicators are retrieved. Every message is aggregated back again by aggregator task $T22$, so that an HTML report file is generated by translator tasks bound to exit port $P9$, which outputs the list of the research outcomes for the research unit that is then available on the CIIC Web under the *Research* menu, cf. Figure 9.

5.3 | Code generation

One of the key features of i2Factory is that it promotes a model-centric development, relieving software engineers from the burden of dealing with rather low-level constructs provided by programming languages. The design and configuration of the integration solution is then done at a high level of abstraction on the model and the generation of executable code is achieved by means of model-to-text transformations. This process of code generation is supported by a set of scripts written in MOFScript[53] that, roughly speaking, scan the model, read it, and generate the corresponding Java implementation for the integration solution; not only this, but also its Ports and the internal workflow containing Tasks and Slots. Unfortunately, these scripts are very large and verbose, which makes them not appropriate to be listed in a research article. Thus, in this section we provide three excerpts with a simplified notation that abstracts the generation of code for an integration solution, its ports and tasks, cf. Listings 1–3. Please refer to Frantz et al.[24] for a complete discussion



We have 4 guests and no members online

Home
Created in October 2008 with five members, CIIC ...

CIIC em destaque nos ...
Mais uma vez o CIIC é alvo de grande atenção por ...

Projetos
Os projetos de I&D em curso centram-se no ...

Research Outcome

Reports generated from data available in DeGois® platform / Relatórios g plataforma DeGois® (Last update: 2016-07-19+01:00)

Research Outcome / Produção Científica (year/ano 2016, 19 researchers/investigadores)

Peer-Reviewed Journals / Artigos em Revistas com Arbitragem Científica

Reconstructing Roman archaeological sites: theory and practice. The case of Conimbriga, Open Journal of Social Sciences, vol. 4, pp. 122-132, Correia, V. H., Rodrigues, N., Gonçalves, A., DOI: http://dx.doi.org/10.4236/jss.2016.42017, 2016.

RuleSIM: a toolkit for simulating the operation and improving throughput of rule-based spam filters, Software: Practice and Experience, vol. 46, pp. 1091-1108, Ruano-Ordás, David, Fdez-Glez, Jorge, Fdez-Riverola, Florentino, Basto Fernandes, Vitor, Méndez, José Ramón, ISSN: 00380644, DOI: doi:10.1002/spe.2342, 2016. (ScopusID 84935052337) (JCR 2014 IF 0.897)

Two-stages security controls selection, Procedia Technology, pp. 0-0, Yevseyeva, Iryna, Vitor Basto-Fernandes, van Moorsel, Aad, Helge Janicke, Emmerich, Michael T.M., ISSN: 1877-0509, 2016.

Multiobjective Optimization of Classi¿ers by Means of 3-D Convex Hull Based Evolutionary Algorithms, Information Sciences Journal, pp. 80-104, Zhao, Jiaqi, Vitor Basto-Fernandes, Licheng Jiao, Yevseyeva, Iryna, Asep Maulana, Rui Li, Thomas Back, Ke Tang, Emmerich, Michael T.M., ISSN: 0020-0255, DOI: 10.1016/j.ins.2016.05.026, 2016. (ScopusID 84974802649) (JCR 2014 IF 4.038)

A spam filtering multi-objective optimization study covering parsimony maximization and three-way classification, Applied Soft Computing, vol. 48, pp. 111-123, Basto-Fernandes, Vitor, Yevseyeva, Iryna, Méndez, José R., Zhao, Jiaqi, Fdez-Riverola, Florentino, Emmerich, Michael T.M., ISSN: 1568-4946, DOI: 10.1016/j.asoc.2016.06.043, 2016. (JCR 2014 IF 2.810)

Book chapters / Capítulos de livros

Greenhouse with Sustainable Energy for IoT, Technological Innovation for Cyber-Physical Systems, Springer International Publishing, Oliveira, Filipe T., Leitão, Ségio A., Nabais, Adelino S., Ascenso, Rita M., Galvão, João R., ISBN: 978-3-319-31164-7, DOI: doi:10.1007/978-3-319-31165-4_39, Portugal, 2016.

FIGURE 9 Output published on CIIC Web [Color figure can be viewed at wileyonlinelibrary.com]

regarding transformations and code generation. In Listing 4 we provide an excerpt of the Java code generated for the case study integration solution. Since this integration solution has several ports, tasks and slots, we have simplified the code to show only part of it, which includes the generated code for entry port P1 and the initial four tasks - Filter (T1), Slimmer (T2), Splitter (T3), Replicator (T4)—and their connecting slots in the integration workflow. The complete implementation of the use case integration solution has 1129 lines of code, including those lines required to configure and run the solution, which would have to be written by a software engineer in the absence of i2Factory.

```
package <Package.name>;
<Import classes>
public class <Process.name> extends Process {
   <Tasks declaration>
   <Slots declaration>
   <Ports declaration>

   public <Process.name>() {
      <Slots initialization>
      <Ports construction>
      <Tasks construction>

   }
}
```

Listing 1: Transforming process

Integration solutions are implemented extending the `Process` class, cf. Listing 1. This script generates the Java code to the whole integration solution. Those parts enclosed within angle brackets are generated by other scripts; text out of angle brackets is assumed to be copied verbatim to the final code generated. Please note that the code generation process is driven by all data provided by the software engineer for each element on the model in its corresponding properties pane, such as shown in Figures 4 and 5.

```
<EntryPort.name> = new <EntryPort.getTypeName()>(‘‘<EntryPort.name>’’) {
   @Override
   public void initialize() {
      generateCode(<EntryPort.name>)
   }
};
addPort(<p.name>);
```

Listing 2: Constructing entry ports

For each type of port there is a corresponding script that transforms it into Java code. Listing 2 shows the script used to transform entry ports. This script is executed on every entry port found in the model and it calls auxiliary scripts to generate the implementation code for the `initialize()` operation. Please, note that it is in this operation that the port creates and uses the corresponding connector to communicate with the integrated application (see generated code in Listing 4). The last line in this script corresponds to the required code to add that port to the integration solution. The script in Listing 3 is executed on every task found in the model and generates not only the corresponding Java code of the task but also the required code to bind every input and output of the task to its previous and next task or port. The last line in this script corresponds to the code required to add the task to the integration solution.

```
<Task.name> = new <Task.getTypeName()>(‘‘<Task.name>’’, <Task.inputs.size()>,
               <Task.outputs.size()>) {
   @Override
   public void doWork(Exchange exchange) throws TaskExecutionException {
      generateCode(<Task.name>)
   }
};
<Bind input slots>
<Bind output slots>
addTask(<Task.name>);
```

Listing 3: Constructing tasks

```
package spe.solution;
// imports were intentionally omitted to save space
public class IntegrationSolution extends Process {

   protected Task[] task;
   protected Slot[] slot;
   protected EntryPort p1;

   public IntegrationSolution() {
      super(``Case Study Integration Solution'');

      slot = new Slot[24];
      for (int i=0; i<slot.length; i++) {
         slot[i] = new Slot(``Slot #''+i);
      }

      p1 = new EntryPort(``Entry Port P1'') {
         @Override
         public void initialize() {
            setInterSlot(new Slot(``InterSlot''));
            Communicator com = new DropboxConnector();
            com.output[0].bind(getInterSlot());
            setCommunicator(com);
         }
      }; addPort(p1);

      task = new Task[23];

      task[0] = new Filter(``Filter T1'', 1, 1) {
         @Override
         public void doWork(Exchange exchange) throws TaskExecutionException {
            Message msg = exchange.input[0].poll();
            // apply filtering policy
            exchange.output[0].add(msg);
         }
      }; task[0].input[0].bind(p1.getInterSlot()); task[0].output[0].bind(slot[0]);
      addTask(task[0]);

      task[1] = new Slimmer(``Slimmer T2'', 1, 1) {
         @Override
         public void doWork(Exchange exchange) throws TaskExecutionException {
            Message msg = exchange.input[0].poll();
            // slims the message
            exchange.output[0].add(msg);
         }
      }; task[1].input[0].bind(slot[0]); task[1].output[0].bind(slot[1]);
      addTask(task[1]);

      task[2] = new Splitter(``Splitter T3'', 1, 1) {
         @Override
         public void doWork(Exchange exchange) throws TaskExecutionException {
            Message msg = exchange.input[0].poll();
            // splits the message into a list of output messages
            for (Message outMsg : msgList) {
               exchange.output[0].add(outMsg);
            }
         }
      }; task[2].input[0].bind(slot[1]); task[2].output[0].bind(slot[2]);
      addTask(task[2]);

      task[3] = new Replicator(``Replicator T4'', 1, 2);
      task[3].input[0].bind(slot[2]);
      task[3].output[0].bind(slot[3]); task[3].output[1].bind(slot[4]);
      addTask(task[3]);

   }
```

Listing 4: Excerpt of generated Java code for the integration solution

6 | DEVELOPMENT EVALUATION

In this section, we report on measures and data collected that indicate the efforts on using i2Factory to develop the case study integration solution. We do not aim to approach efficiency when it comes to the use of i2Factory, neither to compare it with others. We think i2Factory is a viable tool to design, implement, run and monitor integration solutions according to the times collected during this development.

The development was conducted by a software engineer who was familiar with previous versions of Guaraná and i2Factory implemented in the case study. He was provided with a system user to get access to the integration platform from his own computer, by using a web browser and a short introduction on the integration problem described in the previous section. He was required to measure the following variables with a hand clock to estimate the effort involved in the development cf. Table 5:

Time to study the integration problem:	the amount of time spent reviewing the case study, which involves studying the applications to integrate, that is, their communications layer, their data layers, their schemata, and so forth.
Time to sketch the integration solution:	the amount of time the designer spent to devise a complete and ready-to-implement design of an integration solution for our case study.
Time to chain building blocks:	the amount of time spent dragging and dropping, positioning, and connecting building blocks, without setting anything up.
Time to set up credentials:	the amount of time spent creating and setting up credentials for the many applications that require authentication.
Time to set up ports:	the amount of time spent setting up all the input/output ports in the solution.
Time to set up tasks:	the amount of time spent setting up all the tasks in the solution.
Time spent debugging:	the amount of time spent debugging the solution.
Number of errors in tasks:	the number of errors found in tasks.
Number of errors in ports:	the number of errors found in ports.
Number of chained errors:	the number of errors that appeared when previous errors were corrected.

The time to study the integration problem is mandatory and independent from the integration technology applied. It includes the study of data sources, data schemata, and other technology-independent analyses. The time to chain building blocks reflects the benefit from the integration problem study performed in previous stages.

We used for our case study solution-specific credentials (e.g., Dropbox credentials) and i2Factory platform credentials. As shown in Figure 10, the solution uses one Dropbox entry port for data entry (*P*1) and six Dropbox exit ports (*P*4-*P*9) for data output. The time used to set up the credentials includes the process applied to the platform and also the connection to Dropbox and the confirmation via web browser. i2Factory allows credentials to be reused, which helps reduce the time spent on setting up ports and translators. The time spent on setting up Dropbox entry ports is greater than the one spent on setting up Dropbox exits port, once the former have more parameters than the latter. In our case, despite not mandatory, the path in the Dropbox entry port was set up so that the solution had a better performance. Regarding Dropbox exit ports, we set up the credentials, the command to be used, and the paths to the appropriate files.

HTTP solicitor ports *P*2 and *P*3 got similar times, despite connecting to different resources even in similar settings. In our case, the differences were set previously in Translators *T*5 and *T*18. The time taken to set up filter *T*1 is the time needed to build a condition with an XPath expression builder, which is built into i2Factory. The software engineer only had to type the filename, which helped reduce the possibility of errors. The XPath expression builder was used to configure slimmer *T*2 and splitter *T*3. Configuring replicator *T*4, correlator *T*6, replicator *T*17, and correlator *T*20 also took little time because they were mostly set up with default values; the only requirement was to change their names. Despite being used in similar situations, translators *T*5 and *T*18 took different times to be set up because of the complexity of queries involved. The times taken regarding context-based enrichers *T*7 and *T*21 are also different because there were errors when the former was configured for the first time; we did learn from those errors, which helped us configure the second one faster. The same applies to aggregators *T*8 and *T*22: once this kind of task is easy to set up because it only requires software engineers to know the schemata of input messages (splitters *T*3 and *T*16), completing the task for the second time was even faster. Translator *T*9 is a task designed to prepare information for future tasks. It did not take much time since configuring it was not difficult at all; the same applies to other translators. Replicator *T*10 was no difficult to configure either, once it only requires to be bound with translators in its output, and message replication is automatically done . Translators *T*11, *T*12, *T*13, *T*14, and *T*15 are used to create the contents of the files that are uploaded to Dropbox, which means that they

TABLE 5 Effort to develop the integration solution

Measure	Time (hh:mm:ss)
Time to study the integration problem	00:44:28
Time to sketch the integration solution	00:25:09
Time to chain building blocks	00:22:07
Time to set up Dropbox credentials	00:02:02
∑ Time to set up ports	00:02:00
Entry port *P*1	00:00:40
Solicitor port *P*2	00:00:17
Solicitor port *P*3	00:00:11
Exit port *P*4	00:00:11
Exit port *P*5	00:00:08
Exit port *P*6	00:00:08
Exit port *P*7	00:00:08
Exit port *P*8	00:00:09
Exit port *P*9	00:00:08
∑ Time to set up tasks	04:25:33
Filter *T*1	00:00:40
Slimmer *T*2	00:00:55
Splitter *T*3	00:00:37
Replicator *T*4	00:00:01
Correlator *T*6	00:00:16
Context-based enricher *T*7	00:03:47
Aggregator *T*8	00:01:21
Translator *T*9	00:03:20
Replicator *T*10	00:00:21
Translator *T*11	01:27:45
Translator *T*12	00:18:34
Translator *T*13	00:35:19
Translator *T*14	00:17:46
Translator *T*15	01:18:16
Splitter *T*16	00:01:11
Replicator *T*17	00:00:08
Translator *T*18	00:08:21
Translator *T*19	00:01:10
Correlator *T*20	00:00:10
Context-based enricher *T*21	00:01:12
Aggregator *T*22	00:00:45
Translator *T*23	00:01:04
Time spent debugging	00:31:51
∑ Number of errors in tasks	5
Number of errors in filter	1
Number of errors in translators	2
Number of errors in context-based enrichers	2
Number of errors in the solicitor port	1
∑ Number of chained errors	2
Number of errors as a consequence of redesign	1
Number of errors as a consequence of bugs	1

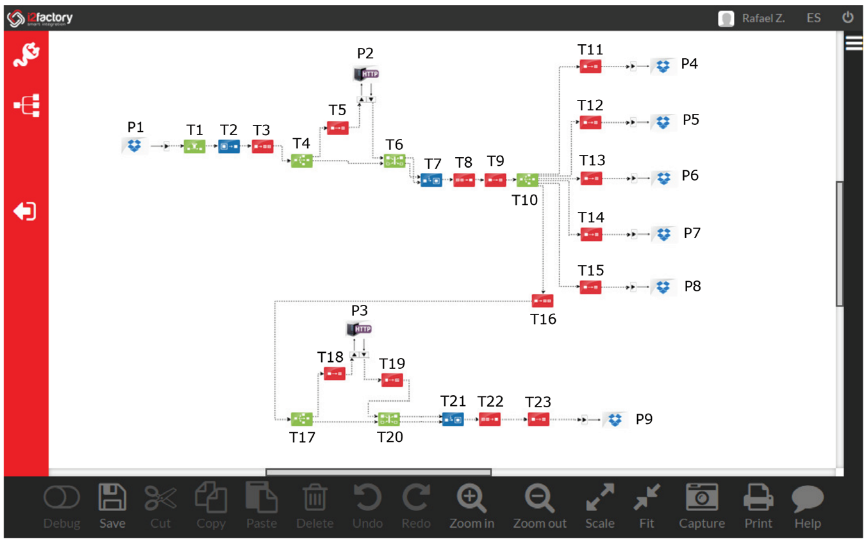


FIGURE 10 Tasks and Ports within the integration solution [Color figure can be viewed at wileyonlinelibrary.com]

have many commonalities that help reduce configuration time. However, configuring translators $T1$, $T12$, $T13$, and $T14$ was easier than configuring translator $T15$. Splitter $T16$ took longer when compared with splitter $T3$ because the XPath expression builder was not yet available, and settings had to be manually set. Translator $T19$ is used after HTTP solicitor port $P3$ to include XML namespaces required by the HTTP reply; nothing similar was used after HTTP solicitor port $P2$ since namespaces were not required. Translator $T23$ is similar to translators T11 through T15; since producing the content from the resulting files is relatively easy and similar to previous translators, it helped shorten the configuration time.

After the engineer finished setting up the solution, it was run in debug mode and some errors were found. This fact required us to reexamine the solution, and reset some configuration parameters to perform the redesign. Half-an-hour later, the solution was running and the results were correct. The errors included a misspelled file name in filter $T1$, a wrong connection between the correlator and the context-based enricher (which happened twice) and some errors in the XSLT specifications used to configure all translators.

7 | CONCLUSIONS

Companies are always designing new business processes or optimizing their current ones to boost their businesses. They have typically developed in-house or purchased third-party software applications to support their business. As a result, they have ended up with heterogeneous software ecosystems composed of applications developed by the use of different technologies, which may run on different operating systems and have different data models. Moreover, such applications did not use to be designed by taking integration into account. EAI focuses on providing methodologies, techniques and tools to design, implement, and monitor EAI solutions. The integration community has been working hard on shifting current tools from code-centric to model-centric development approaches.

In this article, we have introduced i2Factory, which is a cloud-based web development platform that allows for the design, implementation, execution, and monitoring of integration solutions devised through Guaraná. Integration solutions can be designed by the use of a DSL, which results in platform-independent models for EAI solutions. They are the result of a joint effort between academy and industry to help integration engineers reduce costs involved in the design and implementation of integration solutions, by taking models to the center of the development process.

We have presented an integration solution in the context of managing research outcomes to demonstrate how our language and our platform can be used to solve real-world problems in the industry. The i2Factory development environment was tested and proved convenient to develop, deploy, monitor, and manage integration solutions in a systematic, cost effective, and at a high level of abstraction. Its fully web-based development environment and easy deployment of solutions to public or private cloud infrastructures demonstrated the core features to be present in the next generation of integration platforms. This information system integration approach was eventually evaluated by ten measures, which empirically estimate the amount of effort involved in the development of the proposed integration solution. We consider the times achieved in each metric during the development of the solution acceptable, even though they may be quite long regarding the total of time to set up tasks (4 h, 25 min, and 33 s). Please note that this time was dramatically increased by the complexity in the development of XPath transformations into translator tasks T11 and T12, and not due to the use of i2Factory. This time shall be reduced with software engineers more experienced with XPath. An inconvenience of i2Factory might afflict the developer when the connector is not already available on the platform, and, therefore, must first be developed and deployed on i2Factory for use. Note that, once this connector has been developed, it becomes available for use in other integration solutions, thus allowing a reduction in time and cost for future solutions. Panel (D) on Figure 3 shows the actual variety of connectors available at i2Factory, which covers a wide range of integration problems.

It is desirable that an integration platform provided as a service in the cloud would be efficient in terms of resources usage, chiefly because the cloud follows the pay-as-you-go model. For future work, we intend to optimize our cloud-based integration platform, specifically the run-time system, which is the piece of software responsible for executing integration solutions, in order to process more messages by consuming less computing resources. It is still a challenge when it comes to provide software systems in the cloud - the same goes for cloud-based integration platforms.[54]

ACKNOWLEDGMENTS
This work was supported by the Brazilian Coordination Board for the Improvement of University Personnel (CAPES) under project grant 88881.119518/2016-01; the Research Support Foundation of the State of Rio Grande do Sul (FAPERGS) under project grant 17/2551-0001206-2, by the Spanish R&D programme through project grants TIN2016-75394-R, TIN2013-40848-R, and TIN2010-21744-C1, and by the Portuguese Foundation for Science and Technology, I.P., under project UID/CEC/4524/2016.

ORCID
Rafael Z. Frantz https://orcid.org/0000-0003-3740-7560
Rafael Corchuelo https://orcid.org/0000-0003-1563-6979
Vitor Basto-Fernandes https://orcid.org/0000-0003-4269-5114
Fabricia Roos-Frantz https://orcid.org/0000-0001-9514-6560

REFERENCES
1. Da Silva AR. Model-driven engineering: a survey supported by the unified conceptual model. *Comput Lang Syst Struct*. 2015;43:139-155.
2. Linington PF. Automating support for e-business contracts. *Int J Cooperat Inf Syst*. 2005;14(2-3):77-98.
3. Bézivin J. On the unification power of models. *Softw Syst Model*. 2005;4(2):171-188.
4. Kent S. Model driven engineering. Paper presented at: Proceedings of the International Conference on Integrated Formal Methods (IFM), Berlin, Heidelberg; 2002:286-298.
5. Schmidt DC. Guest editor's introduction: model-driven engineering. *IEEE Comput*. 2006;39(2):25-31.
6. Hailpern B, Tarr PL. Model-driven development: the good, the bad and the ugly. *IBM Syst J*. 2006;45(3):451-462.
7. Mens T, Van Gorp P. A Taxonomy of model transformation. *Electron Notes Theory Comput Sci*. 2006;152:125-142.
8. Bézivin J, Hammoudi S, Lopes D, Jouault F. Applying MDA approach for web service platform. Paper presented at: Proceedings of the Enterprise Distributed Object Computing Conference (EDOC), Monterey, CA; 2004:58-70.
9. Miller Joaquin, Mukerji Jishnu. *MDA Guide Version 1.0.1*. Needham, Massachusetts, EUA: Object Management Group; 2003. http://www.omg.org/cgi-bin/doc?omg/03-06-01.
10. Kurtev I, Bézivin J, Jouault F, Valduriez P. Model-based DSL frameworks. Paper presented at: Proceedings of the Object-Oriented Programming, Systems, Languages & Applications Conference (OOPSLA), Portland, Oregon, EUA; 2006:602-616.
11. Kosar T, Bohra S, Mernik M. Domain-specific languages: a systematic mapping study. *Inf Softw Technol*. 2016;71:77-91.
12. Estublier J, Vega G, Ionita AD. Composing domain-specific languages for wide-scope software engineering applications. Paper presented at: Proceedings of the International Conference on Model Driven Engineering Languages and Systems (MODELS), Montego Bay, Jamaica; 2005:69-83.
13. Deursen A, Klint P. Little languages: little maintenance? *J Softw Maintenan*. 1998;10(2):75-92.
14. Baker P, Loh S, Weil F. Model-driven engineering in a large industrial context: motorola case study. Paper presented at: Proceedings of the International Conference on Model Driven Engineering Languages and Systems (MODELS), Montego Bay, Jamaica; 2005:476-491.
15. Tolvanen JP, Kelly S. Defining domain-specific modeling languages to automate product derivation: collected experiences. Paper presented at: Proceedings of the Software Product Line Conference (SPLC), Rennes, France; 2005:198-209.
16. Voelter M. *DSL Engineering: Designing Implementing and Using Domain-Specific Languages*. Scotts Valley, Califórnia, EUA: CreateSpace Independent Publishing Platform; 2013.
17. Ibsen C, Jonathan A. *Camel in Action*. Shelter Island, Nova York, EUA: Manning Publications; 2018.
18. Gutierrez F. *Spring Boot Messaging: Messaging APIs for Enterprise and Integration Solutions*. New York, NY: Apress publishing; 2017.
19. Dossot David, D'Emic John, Romero Victor. Mule in Action. Manning Publications; 2018.
20. Hohpe G, Woolf B. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Boston, Massachusetts, EUA: Addison-Wesley; 2003.
21. Thoo E, Pezzini M, Guttridge K, Bhullar B. *Magic Quadrant for Enterprise Integration Platform as a Service*. Stamford, Connecticut, EUA: Gartner; 2019.

22. Li Q, Wang Z, Li W, Cao Z, Du Ruiyang LH. Model-based services convergence and multi-clouds integration. *Comput Ind*. 2013;64(7):813-832.
23. Panetto H, Zdravkovic M, Jardim-Goncalves R, Romero D, Cecil J, Mezgár I. New perspectives for the future interoperable enterprise systems. *Comput Ind*. 2016;79:47-63.
24. Frantz RZ, Reina-Quintero AM, Corchuelo R. A domain-specific language to design enterprise application integration solutions. *Int J Cooperat Inf Syst*. 2011;20(2):143-176.
25. Boomi iPaaS solutions & tools for cloud connected business; 2020. https://boomi.com/. Accessed May 21, 2020.
26. Data integration cloud | informatica; 2020. https://www.informatica.com/products/cloud-integration/integration-cloud.html Accessed May 21, 2020.
27. Unleash the transformative power of APIs and integration | Jitterbit; 2020. https://www.jitterbit.com. Accessed February 12, 2020.
28. Azure integration services, microsoft azure; 2020. https://azure.microsoft.com/en-us/product-categories/integration/. Accessed May 20, 2020.
29. MuleSoft, integration platform for connecting SaaS and enterprise applications; 2020. https://www.mulesoft.com/. Accessed February 10, 2020.
30. Application integration and API management| oracle; 2020. https://cloud.oracle.com/integration. Accessed March 11, 2020.
31. Enterprise integration, data integration tools | snaplogic; 2019. https://www.snaplogic.com/. Accessed December 18, 2019.
32. Workato app integration & automation - automate everything; 2020. https://www.workato.com/. Accessed May 20, 2020.
33. Asuncion CH, Iacob ME, Sinderen MJ. Towards a flexible service integration through separation of business rules. Paper presented at: Proceedings of the IEEE International Enterprise Distributed Object Computing Conference (EDOC), Vitória, ES, Brazil ; 2010:184-193.
34. Marhaim I, Mordechai E, Bartolini C, Bergman R, Ariel O, Peltz C. A visual tool for rapid integration of enterprise software applications. Paper presented at: Proceedings of the International Conference on Web Engineering (CWE), Vienna, Austria; 2010:430-444.
35. Li Q, Zhou J, Peng Q-R, et al. Business processes oriented heterogeneous systems integration platform for networked enterprises. *Comput Ind*. 2010;61(2):127-144.
36. Kovanovic V, Djuric D. Highway: a domain specific language for enterprise application integration. Paper presented at: Proceedings of the India Software Engineering Conference (ISEC), Kanpur, India; 2012:33-36.
37. Pinho E, Silva LB, Costa C. A cloud service integration platform for web applications. Paper presented at: Proceedings of the International Conference on High Performance Computing Simulation (HPCS), Bologna, Italy; 2014:366-373.
38. Kern H, Stefan F, Fähnrich KP, Dimitrieski, V. A mapping-based framework for the integration of machine data and information systems. Paper presented at: Proceedings of the International Conference Information Systems (IADIS), Madeira, Portugal; 2015:286-298.
39. Balko S, Barros A. In-memory business process management. Paper presented at: Proceedings of the IEEE International Enterprise Distributed Object Computing Conference (EDOC), Adelaide, Australia; 2015:74-83.
40. Zheng X, Zhang Y, Huang Z, Jia Z, Duan H, Li H. An Integration framework for clinical decision support applications. *Ubiquitous Computing Application and Wireless Sensor*. 331. Jeju, Korea: Springer, Dordrecht; 2015:543-552.
41. Hanson J, Ohlsson J, Ertan N, et al. P2PIE: a new enterprise application integration solution. Paper presented at: Proceedings of the CAiSE Industry Track co-located with 27th Conference on Advanced Information Systems Engineering, Stockholm, Sweden; 2015:60-70.
42. Wei N. Research on an information application integration platform based on SOA and web service. *Int J Multimed Ubiquit Eng*. 2015;10(12):165-274.
43. Xu A, Ling G, Xu W, Zhang X. Research on automation integration technology of application systems based on web services. Paper presented at: Proceedings of the International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), Changsha, China; 2016:2210-2215.
44. Chen C. The framework of web services to information integration. Paper presented at: Proceedings of the International Conference on Electronic Information Technology and Intellectualization (ICEITI), Guangzhou, China; 2016:365-368.
45. Beer MI, Hassan MF. Adaptive security architecture for protecting RESTful web services in enterprise computing environment. *SOCA*. 2018;2(12):111-121.
46. de Souza Cimino L, de Resende JEE, Silva LHM, et al. A middleware solution for integrating and exploring IoT and HPC capabilities. *Softw Pract Exp*. 2019;49(4):584-616.
47. Huang CC, Kuo CY, Chen JH, Huang CW. A low-cost enterprise application integration architecture for large-scale environment. Paper presented at: Proceedings of the 2019 20th Asia-Pacific Network Operations and Management Symposium, Matsue, Japan; 2019:1-4.
48. Frantz RZ, Corchuelo R, Roos-Frantz F. On the design of a maintainable software development kit to implement integration solutions. *J Syst Softw*. 2016;111(1):89-104.
49. FCT *Establishing Portugal as a Global Reference for Research and Innovation*. Lisboa, Portugal: Fundação para a Ciência e a Tecnologia; 2019. http://www.fct.pt/documentos/Brochura_FCT_web.pdf.
50. FCT About the FCT. Fundação para a Ciência e a Tecnologia; 2019. http://www.fct.pt/linhasatividadefct.
51. National Platform for Science and Technology; 2020. http://www.degois.pt/. Accessed August 06, 2020.
52. Scopus home; 2020. http://www.scopus.com/. Accessed August 06, 2020.
53. Oldevik J, Neple T, Grønmo R, Aagedal J, Berre AJ. Toward standardised model to text transformations. Paper presented at: Proceedings of the European Conference on Model Driven Architecture-Foundations and Applications; 2005:239-253; Springer, Berlin, Heidelberg.
54. Harman M, Lakhotia K, Singer J, White DR, Yoo S. Cloud engineering is search based software engineering too. *J Syst Softw*. 2013;86(9):2225-2241.

How to cite this article: Frantz RZ, Corchuelo R, Basto-Fernandes V, Rosa-Sequeira F, Roos-Frantz F, L. Arjona J. A cloud-based integration platform for enterprise application integration: A Model-Driven Engineering approach. *Softw Pract Exper*. 2020;1–24. https://doi.org/10.1002/spe.2916

APPENDIX

TABLE A1 Router tasks

Icon	Task	Description
	Correlator	Analyses inbound messages and outputs sets of correlated ones.
	Merger	Merges messages from different input slots into one output slot.
	Resequencer	Reorders messages into sequences with a preestablished order.
	Filter	Filters out unwanted messages.
	Idempotent Transfer	Removes duplicated messages.
	Dispatcher	Dispatches a message to exactly one slot.
	Distributor	Distributes messages to one or more slots.
	Replicator	Replicates a message to all of the output slots.
	Semantic Validator	Validates the semantics of a message.
	Threader	Increases the number of threads to run tasks in the model.
	Custom Router	Allows for routing a message according to custom semantics.

TABLE A2 Modifier tasks

Icon	Task	Description
	Slimmer	Removes contents from the body of a message according to a static policy.
	Context-based Slimmer	Removes contents from the body of a base message according to a dynamic policy that is provided by a context message.
	Content Enricher	Adds static contents to the body of a message.
	Context-based Content Enricher	Adds dynamic contents from a context message to the body of a base message.
	Header Enricher	Adds static contents to the header of a message.
	Context-based Header Enricher	Adds dynamic contents from a context message to the header of a base message.
	Header Promoter	Promotes a part of the body of a message to its header.
	Header Demoter	Demotes a part of the header of a message to its body.
	Set Correlation ID	Sets a correlation ID value to the corresponding property in the header of a message.
	Set Return Address	Sets a return address value to the corresponding property in the header of a message.
	Custom Modifier	Allows to modify the header and body of a message according to custom semantics.

TABLE A3 Transformer tasks

Icon	Task	Description
	Translator	Transforms the body of a message from one schema into another.
	Splitter	Splits a message that contains repeating elements into several messages.
	Aggregator	Constructs a new message from several messages produced previously by a Splitter.
	Chopper	Breaks a message into two or more messages.
	Assembler	Constructs a new message from two or more messages.
	Cross Builder	Constructs a new message that contains the Cartesian product of all inbound messages.
	Custom Transformer	Allows for transformation of a message according to custom semantics.