Studies in Engineering and Exact Sciences
ISSN: 2764-0981

Arquitectura para sustentar la integración de conocimiento externo heterogéneo en un motor de reglas

Arquitetura para apoiar a integração de conhecimentos externos heterogéneos num motor de regras

DOI: 10.54021/sesv3n2-004

Recebimento dos originais: 03/02/2022
Aceitação para publicação: 03/03/2022

Marcos Maciel
Software Engineer
Institución: Universidad Abierta Interamericana (UAI)
Dirección: Av. Montes de Oca 745, Buenos Aires, Argentina
Correo electrónico: Marcos.Maciel@uai.edu.ar

Claudia Pons
PhD in Computer Science
Institución: Universidad Abierta Interamericana (UAI) / Universidad Nacional de La Plata / CIC
Dirección: Av. Montes de Oca 745, Buenos Aires, Argentina
Correo electrónico: Claudia.pons@uai.edu.ar

RESUMEN
En un contexto de negocios globalizado donde la completitud de la información se obtiene al componer varias partes, resolver problemas se convierte en una tarea que involucra tiempo, análisis y experiencia. Una organización ve limitado su ámbito de acción porque necesita información de terceros para evaluar en forma íntegra y completa una colección de datos. Para superar estos problemas se propone implementar un motor de reglas capaz de interactuar mediante reglas con servicios usando Json como mensajería de intercambio de datos. El modelo propuesto mejora la capacidad de conocimiento al compartir información entre sistemas heterogéneos usando los estándares de la comunidad para resolver problemas complejos.

Palabras clave: motor de reglas, dss, lógica simbólica, arquitectura orientada a servicio, soa.

RESUMO
Num contexto empresarial globalizado onde a exaustividade da informação é obtida através da composição de várias partes, a resolução de problemas torna-se uma tarefa que envolve tempo, análise e perícia. Uma organização é limitada no seu âmbito de acção porque necessita de informação de terceiros para avaliar completa e completamente uma recolha de dados. Para ultrapassar estes problemas, propõe-se a implementação de um motor de regras capaz de interagir através de regras com serviços que utilizam a Json como troca de mensagens de intercâmbio de dados. O modelo proposto melhora a capacidade de conhecimento

através da partilha de informação entre sistemas heterogéneos, utilizando normas comunitárias para resolver problemas complexos.

Palavras-chave: motor de regras, dss, lógica simbólica, arquitectura orientada para os serviços, soa.

1 INTRODUCCIÓN

Un motor de reglas es una herramienta de software con reglas que evalúan de forma encadenada bloques de conocimiento y experiencia configurados por un experto en la materia. Cada regla es una porción de información expresada: **Si** condición **Entonces** conclusión.

Estos sistemas ayudan a las personas con la tarea de recolectar información para unificar conocimiento (Peña, 2006), analizar relaciones entre los datos y emitir resultados con el objetivo de apoyar la toma de decisión. Son menos propensos a cometer errores porque razonan como un humano lo haría (Agarwal, 2014), resuelven cálculos simples y complejos en un tiempo inferior comparado con una persona (Palma & Marín, 2008), después de ser configurado también puede ser usado por personas no expertas.

Algunos sistemas expertos desarrollados recientemente implementan la inteligencia con diversos métodos, por ejemplo, para diagnosticar miocardiopatía dilatada (Bahrami et al, 2014) usa un árbol de decisión de 46 reglas con respuestas del tipo si/no integrando lógica y conocimiento con el lenguaje de programación Clips (Clips, s.f). Con 16 reglas programadas y un árbol de decisión este sistema experto desarrollado en Php y Mysql genera un diagnostico sugerido sobre 16 tipos de enfermedades oftalmológica (Munaiseche et al, 2018).

En el campo de la telemedicina Massaro et al, desarrollaron un sistema para soportar decisiones de multiniveles apoyado en inteligencia artificial con una arquitectura no orientada a servicios.

En Prado et al, se propone un sistema experto para apoyar la enseñanza en combinación lineal de vectores usando Prolog como lenguaje de programación.

En relación con el aprendizaje Mohd et al, propone un sistema de apoyo de decisión basado en la predicción del estilo de aprendizaje de estudiantes, usando como input la interacción usuario-sistema a través del comportamiento corporal. El diseño técnico de la solución responde a un motor de inferencia con una base

de datos para centralizar el conocimiento. En la industria de fabricación de línea blanca se propone la recolección de datos desde sensores IoT (Internet of Things) y la ejecución de un DSS (Decision Support System) desarrollado en Minitab® con el objetivo de incrementar la productividad y eliminar el stock (Shady & Eltawil, 2018). Una arquitectura orientada a servicios permite a una herramienta CIG (Computer-Interpretable Guidelines) utilizar múltiples fuentes y formato de datos, pero delega por completo el razonamiento a una aplicación monolítica desarrollado en Prolog (Chapman & Curcin, 2019).

Berona et al, propone un DSS para predecir la calidad ecológica necesaria para el cultivo de microalgas en exterior, para ello usan una arquitectura monolítica de machine learning y algoritmos en Python con resultados expuestos en la web. Relacionado a medio ambiente (Yu et al,.2020) presento una arquitectura con GIS (sistema de información geográfico) para monitorear y predecir la calidad del agua en pozos privados, aunque hace uso de servicios para extender el sistema implementado este se encuentra acoplado a un software propietario. Para el ámbito de la agricultura se desarrolló un DSS para asistir a los granjeros con información basada en el clima y envió de SMS (Short Message Service) como mecanismos de comunicación (Soyemi & Adesola, 2018), el prototipo usa una arquitectura tradicional de sistema experto con una base de datos y un módulo de predicción. Thaker & Nagori analizo las funciones usadas en sistemas expertos que utilizan lógica difusa en sistemas de recomendaciones tradicionales con base de conocimiento centralizada. En la industria 4.0 Cheng et al, propone una arquitectura basada en la nube donde combina IoT con árboles de decisión para mejorar la producción de prótesis dentales. El uso de DSS para el desarrollo de smart cities elaborado en Bartolozzi et al, usa una arquitectura de consulta a múltiples repositorios con información divida por área de interés.

Para un ser humano tomar decisiones en base al conocimiento experto de un dominio es una tarea que involucra tiempo, análisis y experiencia. Decidir conlleva asociado un proceso cognitivo divido en etapas: invertir tiempo para recolectar grandes volúmenes de información, depurar y/o clasificar mediante sesgos y heurísticas, determinar la probabilidad de ocurrencia y un porcentaje de exactitud sobre cada posible respuesta encontrada (Rampello, 2019), por último, asegurar el respaldo de la información procesada en soporte digital etc. Todas

estas tareas pueden resultar difíciles de gestionar debido a que, con la globalización las personas cada día generan más contenido y el volumen de información se torna incontrolable. Los motores de reglas son una solución ideal para gestionar y almacenar el conocimiento.

En este trabajo se propone desarrollar el diseño de la arquitectura propuesta en (Maciel, 2020), integrando conocimiento mediante la combinación de regla ejecutadas en código tradicional y regla ejecutadas en servicios Api Rest (propietarios o externos) para dar soporte a una variedad de funcionalidades de negocios desplegadas en la nube e independiente unas con otras. El objetivo es extender las fronteras del conocimiento incorporando una variada gama de sistemas heterogéneos en la ejecución de reglas, establecer relaciones de negocios globales, compartir conocimiento, reusar desarrollos tecnológicos propios y de terceros y optimizar los procesos de validaciones, entre otras.

Este artículo está organizado de la siguiente forma: integración de conocimiento mediante Json son detallados en la sección 2, implementación técnica es presentada en la sección 3, testing y evolución es descripta en la sección 4, y por último se hallan conclusiones y trabajos futuros en la sección 5.

2 INTEGRACIÓN DE CONOCIMIENTO MEDIANTE JSON

Json (JavaScript Object Notation) (JSON, s.f). es una representación lógica, organizada y de fácil lectura que tienen como finalidad intercambiar datos entre sistemas sin importar el software subyacente que lo creo. Los valores o datos están contenidos por atributos que representan un modelo completo o parcial de un negocio, por ejemplo, {"first_name": "Jhon"} donde {atributo: valor}. Es posible leer jerárquicamente su estructura para recuperar el par atributo-valor como padres e hijos usando las llaves ({}). Los servicios SOA (por sus siglas en ingles Service-Oriented Architectures) permite aprovechar las ventajas de este formato porque comparten contrato formal, son débilmente acoplados, abstractos, sin estado y reusables (Rosado Gomez & Jaimes Fernández, 2018). Las características antes mencionadas hacen del formato de intercambio Json y de los servicios un medio indicado para unir sistemas heterogéneos sin dependencia del lenguaje de programación, de una representación exacta de objetos de negocios, del medio de almacenamiento que soporta los objetos de negocios

(base de datos, xml, archivos, etc.), del tipo de base de datos (relacional o no relacional), etc. Para integrar conocimiento de distintas fuentes de datos sin las preocupaciones de la tecnología que los soportan se propone un sistema experto capaz de llamar a servicios usando Json como medio para el intercambio de información con los siguientes pasos:

2.1 CONFIGURACIÓN DE ENTIDADES Y TIPO DE DATOS A PARTIR DE UN JSON NO PROPIETARIO

A continuación, se presenta un mensaje Json recuperado de una plataforma de pagos, primero se configura al sistema experto mapeando las entidades y atributos como level-entity donde cada entity tiene un tipo de datos asociado. El sistema tiene preconfigurado una lista de funciones o hechos que se asignan dependiendo del tipo de datos (Maciel, 2020), por ejemplo "area_code" se asocia al tipo de datos numérico que tiene preconfigurado algunas funciones matemáticas como por ejemplo sumas, restas o comparaciones del tipo x > 0 < x. Configurados los pasos previos se pueden construir las reglas desde un mensaje u objeto Json de terceros.

```
curl -X POST \ 'https://api.client.com/v1/customers' \
   -H 'Authorization: Bearer ACCESS_TOKEN_ENV' \ -d '

"customer": {"email": "jhon@doe.com", "first_name": "Jhon", "last_name": "Doe",
  "phone": {"area_code": "55", "number": "991234567"},
  "identification": { "type": "CUIT", "number": "12345678900" },
"default_address": "Home",
  "address": {"id": "123123", "zip_code": "01234567",
"street_name": "Av Corrientes", "street_number": "123"},
  "date_registered": "2000-01-18", "description": "Description user",
"default_card": "None",
  "file": "JVBERi0xLjQKJeLjz9MKMiAwIG…"}
```

La tabla 1 contiene la categorización de cada valor del atributo(entity) con un tipo de datos donde date, number y string son los tipos básicos, por otro lado, Identity e Invoice son tipos de datos agrupadores de atributos por ej.

{"identification": {"type": "CUIT", "number": "12345678900"}} == Identity. Este tipo de datos propietario del motor de reglas permite asignar una función que valida un conjunto de datos como una unidad o si es necesario llamar a un servicio externo componer el input. Las funciones son porciones de código capaces de ejecutar lógica simbólica mediante programación tradicional, expresiones regulares, matemáticas o algebraicas o llamar a servicios externos al motor de reglas.

Table 1. Lista de atributos recuperado de un mensaje json.

Atributo	Date	Number	Identity	Invoice	String
email					x
first_name					x
last_name					x
phone.area_code		x			
phone.number		x			
identification.type			x		
identification.number		x	x		
default_address					x
address.id		x			
address.zip_code					x
address.street_name					x
address.street_number		x			
date_registered	x				
description					x
default_card					x
file				x	

Por ejemplo, el tipo de datos Invoice del atributo file permite ejecutar una regla intermedia, invocando a un endpoint cuyo input es un string en base64(factura afip pdf) y usando un procedimiento OCR (Optical Character Recognition) retorna un json que el motor de reglas inyecta al mensaje principal según:

"customer": {
"invoice": {"category": "CAE", "identity_type": "80", "identity_number": "20000000001",
"sales_point": "1", "type": "A", "number": "00141787", "date": "20101014",
"amount": "300.8", "authorization_code": "60428000005029", "receptor_identity_type": "80",
"receptor_identity_number": "300000000007"}}

Después de la configuración de un mensaje Json a un modelo de negocios el motor de reglas esta disponible para crear paquetes de reglas.

2.2 DESARROLLO DE UN PAQUETE DE REGLAS A PARTIR DE UN OBJETO DE NEGOCIOS

A partir de cada par entidad-atributo(level-entity) y con el tipo de dato asociado se pueden crear las reglas de negocios, como se observa a continuación. Una función tiene como objetivo validar una condición de verdad mediante rutinas matemáticas, expresiones de comparación booleanas, validaciones del tipo expresiones regulares o llamadas a servicios.

Lista de reglas configuradas en base al json de trabajo – Paquete de regla:

Regla 1: first name is not null

Regla 2: last name is not null

Regla 3: phone.area_code is not null Λ phone.area_code is number Λ phone.area_code == 2 dígitos

Regla 4: phone.number is not null Λ phone.number is number Λ phone.number > 7 dígitos Λ phone.number < 12

Regla 5: email is not null Λ email is valid (api reg exp)

Regla 6: trusth email domain (api externa)

Regla 7: email not in black list (api interna)

Regla 8: address.zip_code is not null Λ address.zip_code is number Λ address.street_name is not null Λ address.street_number is not null Λ address.street_number is number

Regla 9: phone.area_code belongs to address(api interna)

Regla 10: Afip invoice (api interna)

Regla 11: category is not null Λ category between CAE-CAI-CAEA Λ identity_number is not null Λ identity_number is Identity Λ sales_point is not null Λ sales_point is number Λ type is not null Λ type between A-B-C Λ number is not null number is number Λ date is not null Λ date is date Λ amount is not null Λ amount > 0 Λ authorization_code is not null Λ authorization_code is number Λ receptor_document_type is not null Λ receptor_document_type is number Λ receptor_identity_number is not null Λ receptor_identity_number is Cuit

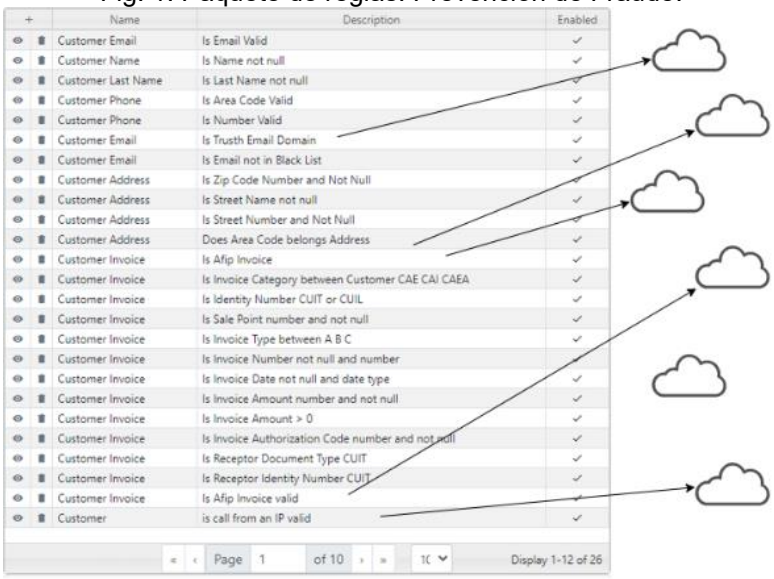
Regla 12: AFIP invoice valid (SOA Service externo)

Regla 13: IP not in Blacklist

Regla n: …

Cada paquete de reglas se puede configurar en forma independiente a otros paquetes, reutilizando los tipos de datos y las funciones parametrizadas la Fig. 1 resume las reglas construidas para el paquete llamado Prevención. La lista contiene reglas que se ejecutan en código del motor de reglas para validaciones base y otras reglas que consumen servicios señaladas con flechas a la nube. Este paquete de reglas demuestra la flexibilidad y creatividad para construir validaciones robustas que involucren no solo código y datos propietarios sino código y datos externos.

Fig. 1. Paquete de reglas: Prevencion de Fraude.



3 IMPLEMENTACIÓN TÉCNICA

La Fig. 2 expone la comunicación entre el motor de reglas (Maciel, 2020) con servicios propietarios hosteados tanto en intranet y otros no propietarios hosteados en internet. Un cliente envía un mensaje json al sistema, el motor recupera el paquete de reglas correspondiente y ejecuta regla a regla según su definición (ver Fig. 3). En los casos de invocaciones hacia endpoint no propietario se dispara una llamada a un servicio de intranet cuya responsabilidad es interactuar con el servicio externo.

Fig. 2. Arquitectura del motor de reglas.

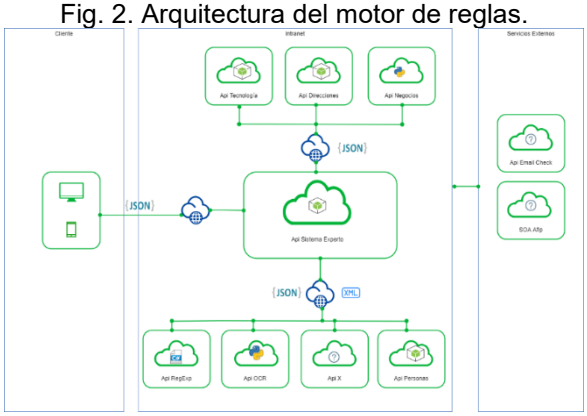
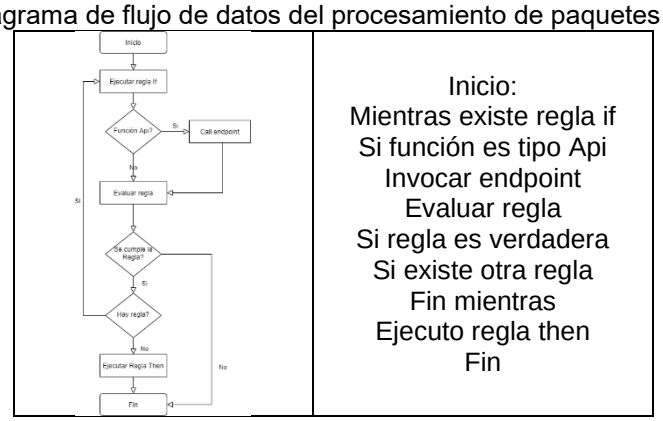


Fig. 3. Diagrama de flujo de datos del procesamiento de paquetes de reglas.



3.1 DISGREGACIÓN DE REGLAS – ENDPOINT:

Regla: Formato de Email

Entidad: email {customer.email}

parámetros: {customer.email: mmaciel03@hotmail.com }

Función: is email format: curl --location

--request GET 'https://{domain}:105/emailformat/{parámetros}'

Tipo servicio: Api Regular Expression

Respuesta:

{"email": "mmaciel03@hotmail.com","isok": "true","type": "hotmail.com","user": "mmaciel"}

Cuando se ejecuta esta regla la función extrae el valor del customer.email y pasa el parámetro al servicio validando la respuesta con status 200 y el valor isok==true.

Regla: Dominio de email confiable.

Entidad: email {customer.email}

parámetros: {customer.email : mmaciel03@hotmail.com }
Función: is trusth email domain: curl --location --request GET 'https://{domain}:106/esdominioemailconfiable/{parámetros}'
Invocación a Api externo:
curl --location --request GET 'https://mailcheck.p.rapidapi.com?domain=mmaciel03@hotmail.com' \
--header 'x-rapidapi-key: xxx' \ --header 'x-rapidapi-host: mailcheck.p.rapidapi.com'
Respuesta:
{"email": "mmaciel03@hotmail.com","isok": "true","type": "hotmail.com","user": "mmaciel"}
Tipo servicio: Api Business

Cuando se ejecuta esta regla la función extrae el valor del {customer.email} y pasa el parámetro al servicio validando la respuesta con status 200 y el valor isok==true. En este ejemplo además existe una llamada a un servicio externo y sobre la respuesta se crea un nuevo objeto.

Regla: Factura Afip
Entidad: Factura {customer.file}
parámetros: “JVBERi0xLjQKJeLjz9MKMiAwIG…”
Función: is AFIP invoice: curl --location --request POST 'https://{domain}:108/getafipinvoice /' \
--header 'Content-Type: application/json' \ --data-raw '{"data": {parámetros}'
Tipo servicio: Api OCR
Respuesta:
"invoice": {"category": "CAE", "identity_number": "20000000001","sales_point": "1", "type": "A", "number": "00141787", "date": "20101014", "amount": "300.8", "authorization_code": "60428000005029", "receptor_document_type": "80",

"receptor_identity_number": "300000000007","isok": "true"}

Esta regla la función extrae el valor del customer. file y pasa el parámetro al servicio validando la respuesta con status 200 y el valor isok==true, adicionalmente inyecta un objeto nuevo para usar en la siguiente regla.

Regla 12: factura aprobada x afip
Entidad: Factura {customer.invoice}
parámetros: {customer.invoice}
Función: is AFIP invoice valid: curl --location
--request POST 'https://{domain}:106/checkafipinvoice /' \
--header 'Content-Type: application/json' \ --data-raw '{customer.invoice}'
Invocación a servicio externo
https://wswhomo.afip.gov.ar/WSCDC/service.asmx?op=ComprobanteConstatar
Tipo servicio: Api Business
Respuesta: {"response": "A", "isok": "true"}

En este último, caso la regla usa información generada por una regla anterior y llama a un servicio que a su vez se comunica con un ente gubernamental para validar información que originalmente llego en base64, logrando un procesamiento más completo.

4 TESTING Y EVALUACIÓN

Se realizaron pruebas de performance, tiempo de respuesta y kb de datos enviados y recibidos, el motor de reglas propuesto actúa como evaluador de datos entre un sistema cliente que envía json y un sistema backend responsable del negocio por este motivo el tiempo incurrido para ejecutar un paquete de reglas no debe ser considerable.

Table 2 Testing de performance

Rules	Time avg	Min	Max	Std. Dev.	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Rule 1- Is Email Valid	528	146	960	232.44	5.735.5928	3.13	0.83	560
Rule 2 - Is Name not null	538	9513	961	275	278.95.1103	2.85	0.75	521
Rule 3 - Is Last Name not null	583	814	949	7	288.54.7510	2.6	0.68	521
Rule 4 - Is Area Code Valid	623	915	982	1	282.94.4014	2.38	0.62	512
Rule 5 - Is Number Valid	660	818	990	3	288.94.0960	2.3	0.61	536
Rule 6 - Is Trusth Email Domain	705	5	107	4	8	1	2.25	0.6
Rule 7 - Is Zip Code Number and Not Null	743	166	113	2	289.32	3.8022	1.89	0.49
Rule 8 - Is Street Name not null	787	223	117	1	286.01	3.5852	1.93	0.51
Rule 9 - Is Street Number and Not Null	798	266	116	9	264.44	3.4326	1.7	0.44
Rule 10 - Does Area Code belongs Address	806	298	116	2	241.78	3.3209	1.71	0.77
Rule 11 - Is Afip Invoice	801	317	116	9	222.36	3.2379	1.57	61.73 497
Rule 12 - Is Invoice Category between Customer CAE CAI CAEA	799	398	118	0	205.01	3.1601 6	1.67	0.44 542
Rule 13 - Is Identity Number CUIT or CUIL	780	386	116	4	194.59	3.1179 8	1.68	0.45 551
Rule 14 - Is Sale Point number and not null	777	440	116	6	183.56	3.0792	1.51	0.39 503
Rule 15 - Is Invoice Type between A B C	762	535	116	3	166.63	3.0484 1	1.49	0.39 500
Rule 16 - Is Invoice Number not null and number	756	565	114	8	160.04	3.0365 6	1.58	0.42 533
Rule 17 - Is Invoice Date not null and date type	739	528	112	5	161.49	3.0257 2	1.52	0.4 515
Rule 18 - Is Invoice Amount number and not null	734	541	118	1	151.26	3.0378 5	1.53	0.4 515
Rule 19 - Is Invoice Amount > 0	718	145	116	2	173.31	3.0831 8	1.6	0.42 530
Rule 20 - Is Invoice Authorization Code number and not null	690	385	111	3	186.96	3.1623	1.63	0.43 527
Rule 21 - Is Receptor Document Type CUIT	650	280	112	7	214.69	3.2829 9	1.64	0.43 512
Rule 22 - Is Receptor Identity Number CUIT	611	189	116	4	239.21	3.4582 9	1.85	0.49 548
Rule 23 - Is Afip Invoice valid	1954	1098	296	2	468.94	3.2148 1	1.35	1.66 429
Rule 24 - Is call from an IP valid	396	18	118	3	370.88	3.7017 8	1.94	0.51 536
Rule 25 - Is Email not in Black List	351	18	111	7	346.89	4.0167 1	2.23	0.59 569
Total	732	18	296	2	378.49	48.769 07	24.98	44.51 524.5

La tabla 2 resume las métricas de la prueba de performance sobre 50 ejecuciones concurrentes evaluado con la herramienta Apache JMeter (JMeter, sf). Average es el tiempo promedio tomado por las 50 ejecuciones, el Min es el tiempo más corto ocupado y el Max el tiempo más largo, Std Dev es la desviación estándar también sobre el tiempo, Throughput numero de request procesados por unidad de tiempo a mayor valor mejor resultado. Por último, Received KB/sec, Sent KB/sec, Avg. Bytes informa el tamaño del mensaje intercambiado ida y vuelta. A priori el tiempo de respuesta se mantiene por abajo del segundo excepto el caso de dependencia con un servicio externo.

La Fig. 4 resume el tiempo medio de ejecución de cada regla para una prueba de 50 corridas y la Fig. 5 gráfica el tiempo individual de la regla todo medido en ms.

Fig. 4. Gráfico de tiempo medio de resolución de paquete de regla.

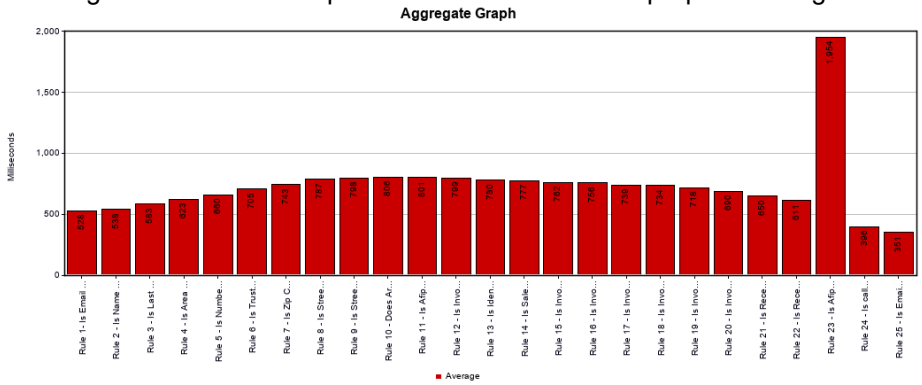
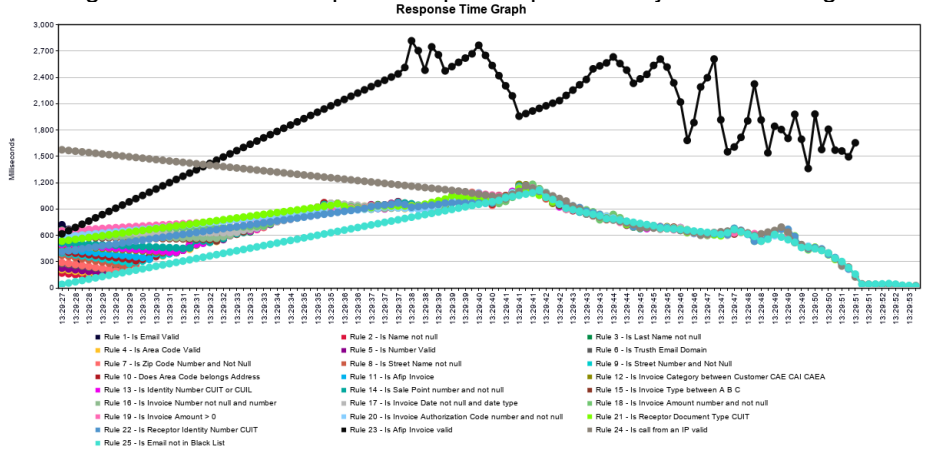


Fig. 5. Gráfico del tiempo de respuesta por cada ejecución de regla.



El Ambiente de pruebas estuvo conformado de la siguiente forma: OS NameMicrosoft Windows Server 2012 R2 Standard Version 6.3.9600 Build 9600,

System Type x64-based PC, Processor Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz, 2195 Mhz, 6 Core(s), 6 Logical Processor(s), Installed Physical Memory (RAM) 38.0 GB, Drive C: File System NTFS Size 79.48 GB (85,343,596,544 bytes), Free Space 22.40 GB (24,051,150,848 bytes).

5 CONCLUSIÓN Y TRABAJOS FUTUROS

En un contexto globalizado donde analizar información ya no depende exclusivamente de datos propios, esta propuesta está orientada a acortar la distancia entre sistemas extendiendo las fronteras del ámbito de la información y la tecnología, hacienda uso de servicios y mensajería para el intercambio de información. El motor de reglas permite configurar una combinación de reglas tanto para validar datos de forma tradicional como para recuperar nueva información proveniente de fuentes externas y reutilizarla en una instancia posterior mediante otras reglas. Esta flexibilidad para construir validaciones de negocios potencia el acceso a datos externos, mejora el acceso a soluciones externas independientemente de la tecnología subyacente, permite reutilizar conocimiento, desarrollar reglas de negocios más robustas e incrementar la certidumbre de los resultados optimizando la toma de decisione,s entre otras mejoras. Como trabajo futuro se puede mencionar la incorporación de lógica no simbólica para soportar algoritmos de machine learning, y la implementación de microservicios con un service bus para mejorar la experiencia de acceso a servicios.

REFERÊNCIAS

Agarwal, M. & Goel, S. (2014) Expert System and it's Requirement Engineering Process. International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014), Jaipur, 2014, pp. 1-4. 2015 de http://ieeexplore.ieee.org [Acceso Junio 2021]

Bahrami, A., Roozitalab, N., Jafari, S. and Bahrami, A. 2014. An expert system for diagnosing dilated cardiomyopathy. International Journal of Engineering Science Invention. 3,3 (March, 2014), 38--42.

Bartolozzi, Marco & Bellini, Pierfrancesco & Nesi, Paolo & Pantaleo, Gianni & Santi, Luca. (2015). A Smart Decision Support System for Smart City. 10.1109/SmartCity.2015.57.

Berona, Elyzer & Buntag, Daibey & Tan, Mary Jane & Coronado, Armin. (2016). Web-Based Decision Support System for Water Quality Monitoring and Prediction for Outdoor Microalgae Cultivation. IOSR Journal of Computer Engineering. 18. 2278-661. 10.9790/0661-1803061620.

Chapman, M. D., & Curcin, V. (2019). A Microservice Architecture for the Design of Computer-Interpretable Guideline Processing Tools. In 18th IEEE International Conference on Smart Technologies IEEE Computer Society Press. https://doi.org/10.1109/EUROCON.2019.8861830

Cheng, Yu-Jie & Chen, Ming-Huang & Cheng, Fu-Chi & Cheng, Yu-Chi & Lin, Yu-Sheng & Yang, Cheng-Jung. (2018). Developing a Decision Support System (DSS) for a Dental Manufacturing Production Line Based on Data Mining. Applied System Innovation. 1. 17. 10.3390/asi1020017.

Clips C Language Integrated Production System http://www.clipsrules.net/ [Acceso Junio 2021].

JMeter. Apache JMeter™ https://jmeter.apache.org/ [Acceso Agosto 2021]

JSON (JavaScript Object Notation - Notación de Objetos de JavaScript) https://www.json.org/json-es.html [Acceso Agosto 2021]

Maciel, Marcos (2020). Motor de Reglas desacoplado orientado a formato JavaScript Object Notation. XXVI Congreso Argentino de Ciencias de la Computación, CACIC Buenos Aires, Argentina. 489-498 ISBN: 0201633612

Massaro, Alessandro & Galiano, Angelo & Scarafile, Domenico & Vacca, Angelo & Frassanito, Antonella & Melaccio, Assunta & Solimando, Antonio & Ria, Roberto & Calamita, Giuseppe & Bonomo, Michela & Vacca, Francesca & Gallone, Anna & Attivissimo, F.. (2020). Telemedicine DSS-AI Multi Level Platform for Monoclonal

Gammopathy Assistance. 1-5. 10.1109/MeMeA49120.2020.9137224.

Mohd, Fatihah & Yahya, Wan & Ismail, Suryani & Jalil, Masita & Maizura, Noor. (2019). An Architecture of Decision Support System for Visual-Auditory-Kinesthetic (VAK) Learning Styles Detection Through Behavioral Modelling. International Journal of Innovation in Enterprise System. 3. 24-30. 10.25124/ijies.v3i02.37.

Munaiseche, Cindy & Kaparang, Daniel & Rompas, Parabelem. (2018). An Expert System for Diagnosing Eye Diseases using Forward Chaining Method. IOP Conference Series: Materials Science and Engineering. 306. 012023. 10.1088/1757-899X/306/1/012023.

Palma J., Marín R (2008). Inteligencia Artificial: Métodos, técnicas y aplicaciones, pp. 83-97 Madrid: McGraw-Hill.

Peña, A.A.: Sistemas basados en conocimiento: Una Base para su Concepción y Desarrollo. Instituto Politécnico Nacional México (2006)

Power, Daniel J., "Decision Support Systems: Concepts and Resources for Managers" (2002). Faculty Book Gallery. 67.

Prado, C. S., León, A. D. L. C. C., & Martín, T. R. T. (2020). AUTOAPRENDIZAJE SOBRE COMBINACIÓN LINEAL DE VECTORES UTILIZANDO UN SISTEMA EXPERTO. Revista Tecnología Educativa, 5(2).

Rampello, S. (2019). “Los sesgos en la toma de decisiones”. Revista Perspectivas de las Ciencias Económicas y Jurídicas, Vol. 9, N° 1 (enero-junio). Santa Rosa: FCEyJ (UNLPam); EdUNLPam; ISSN 2250-4087, e-ISSN 2445-8566. DOI http://dx.doi.org/10.19137/perspectivas-2019-v9n1a06

Rosado Gomez, Alveiro & Fernández, Juan. (2018). REVISIÓN DE LA INCORPORACIÓN DE LA ARQUITECTURA ORIENTADA A SERVICIOS EN LAS ORGANIZACIONES.. REVISTA COLOMBIANA DE TECNOLOGIAS DE AVANZADA (RCTA). 1. 10.24054/16927257.v31.n31.2018.2769.

Shady Salama, Amr B. Eltawil, A Decision Support System Architecture Based on Simulation Optimization for Cyber-Physical Systems, Procedia Manufacturing, Volume 26, 2018, Pages 1147-1158, ISSN 2351-9789, https://doi.org/10.1016/j.promfg.2018.07.151.

Soyemi, Jumoke & Adesola, Adesi. (2018). A Web-based Decision Support System with SMS-based Technology for Agricultural Information and Weather Forecasting. International Journal of Computer Applications. 180. 1-6. 10.5120/ijca2018916338.

Thaker, Shaily & Nagori, Viral. (2018). Analysis of Fuzzification Process in Fuzzy

Expert System. Procedia Computer Science. 132. 1308-1316. 10.1016/j.procs.2018.05.047.

Yu Lan , Wenwu Tang , Samantha Dye & Eric Delmelle (2020) A web-based spatial decision support system for monitoring the risk of water contamination in private wells, Annals of GIS, 26:3, 293-309, DOI: 10.1080/19475683.2020.1798508