9° Congreso Nacional

CoNaIISI 2021

de Ingeniería Informática y Sistemas de Información

4 5 NOVIEMBRE

confedi
Consejo Federal de Decanos de Ingeniería

UTN FACULTAD REGIONAL MENDOZA
UNIVERSIDAD TECNOLÓGICA NACIONAL

Departamento de Ingeniería en Sistemas de Información

Red de Ingenierías en Informática / Sistemas de Información
RIISIC
ConFeDI

9° Congreso Nacional de Ingeniería Informática/Sistemas de Información

CoNaIISI

4 y 5 de noviembre de 2021

Universidad Tecnológica Nacional

Facultad Regional Mendoza

Memoria de Trabajos

9° Congreso Nacional de Ingeniería Informática/Sistemas de Información
9° CoNaIISI: Memorias de trabajos / Compilación de Marcela Fernandez; Matilde Césari; María Gabriela Martínez. - 1a ed. –
Ciudad Autónoma de Buenos Aires: Universidad Tecnológica Nacional, 2022.
Libro digital, PDF

Archivo Digital: descarga y online
ISBN 978-950-42-0213-4

1. Sistemas de Información. 2. Ingeniería Informática. I. Fernandez, Marcela, comp. II. Césari, Matilde, comp. III. Martínez, María Gabriela, comp. IV. Título
CDD 004.0711

Compilación de: Fernandez, Marcela
César, Matilde
Martínez, María Gabriela

Revisado por: Carbonari, Daniela
Caymes Scutari, Paola
Bianchini, Germán

ISBN 978-950-42-0213-4

Auspiciantes:

Orquestación de Reglas para Integrar Conocimiento Heterogéneo

Marcos Maciel[1], Claudia Pons[2]
[1] CAETI UAI, Buenos Aires, Argentina
Mmaciel03@hotmail.com
[2] Universidad Nacional de La Plata, UAI, Buenos Aires, Argentina
Claudia.pons@uai.edu.ar

Resumen

Para un ser humano tomar decisiones en base al conocimiento experto de un dominio es una tarea que involucra tiempo, análisis y experiencia. Sumado a estos factores la globalización agrega complejidad debido al incremento en los volúmenes de información y al esfuerzo que exige ordenarla en tiempo y forma. Para asistir en la toma de decisiones se propone un motor de reglas para integrar conocimiento heterogéneo, haciendo uso del desarrollo de modelos de negocios dinámicos desde mensajes Json y la orquestación de reglas nativas y reglas que procesan la respuesta de servicios, todo como parte del mismo dominio.

Keywords: *Motor de Reglas, DSS, Arquitectura Orientada a Servicio, SOA, Lógica Simbólica.*

1. Introducción

Los sistemas expertos nacen con la idea de asistir a personas en la toma de decisiones, datan de fines de los 50's y principios de los 60's para convertirse en implementaciones de sistemas para los 80's [1]. Como herramienta de software su tarea es procesar y/o analizar información previamente procesada por un experto del dominio basada en hechos, con el objetivo de facilitar y apoyar la toma de decisiones [2]. También son denominados sistemas inteligentes o Sistemas de Soporte a las Decisiones (DSS por sus siglas en inglés Decision Support System) porque resuelven problemas en forma similar a como una persona lo haría, con un tiempo de respuesta considerablemente menor sobre grandes volúmenes de información [3].

Estos sistemas en general están compuestos de 3 partes: un repositorio, hechos o conocimiento representados por un modelo y por último la interfaz de usuario [1]. El repositorio materializa la información o conocimiento utilizada por el motor de ejecución de reglas/inferencia y puede estar contenido en bases de datos y/o archivos, por ejemplo, xls, txt, xml, etc. Existen una variedad de dominios donde estos sistemas son usados, entre los que se pueden mencionar el campo de la medicina, economía, agricultura, educación, en industrias como bancos y seguros solo por nombrar algunos casos. A continuación, se listan algunos sistemas expertos: Para diagnosticar miocardiopatía dilatada en [4], usan un árbol de decisión de 46 reglas con respuestas del tipo si/no desarrolladas en Clips [5]. Con 16 reglas y un árbol de decisión, este sistema experto desarrollado en Php y Mysql genera un diagnostico sugerido sobre 16 tipos de enfermedades oftalmológica [6]. En [7], 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 [8], 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.

Para el campo de la telemedicina, en [9] se desarrolla un sistema para soportar decisiones de multiniveles apoyado en inteligencia artificial con una arquitectura no orientada a servicios.

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 desarrollado en Minitab® con el objetivo de incrementar la productividad y eliminar el stock [10]. Una arquitectura orientada a servicios que 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 [11].

En [12], se 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 en [13] se 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 en un software monolítico. Para el campo de la agricultura se desarrolló un DSS para asistir a los granjeros con información basada en el clima y envío de SMS (Short Message Service) como mecanismos de comunicación [14], el prototipo usa una arquitectura tradicional de sistema experto con una base de datos y un módulo de predicción. En la industria 4.0 en [15] se 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 [16], usa una arquitectura de consulta a múltiples repositorios con información divida por área de interés.

El desafío de las organizaciones es disponer de información en tiempo real para tomar decisiones y un motor de reglas es un medio adecuado para integrar y procesar conocimiento. Para asistir a las personas en la toma de decisiones, se propone utilizar mensajes en formato Json para crear modelos de negocios genéricos y dinámicos sin la necesidad de programar clases u objetos. En segundo lugar, desarrollar reglas que procesen conocimiento en código tradicional y/o invocando servicios internos o externos. Por último, implementar un motor de reglas como orquestador de esta integración de información heterogénea.

Este articulo está organizado de la siguiente forma: en la Sección 2 se trata del Desarrollo de modelos de negocios dinámicos, una propuesta de Orquestación de Reglas para Integrar Conocimiento es presentada en la Sección 3, Pruebas y evaluación es desarrollado en la Sección 4, por último, se hallan conclusiones y trabajos futuros en la Sección 5.

2. Desarrollo de modelos de negocios dinámicos

Un DSS está compuesto por: repositorio, hechos o conocimiento representados por modelos e interfaz de usuario [1]. Estos sistemas internamente representan el modelo de negocios mediante objetos o clases, dependiendo de la tecnología subyacentes estos pueden estar construidos con algún lenguaje de programación como python, nodejs, C#, Java, etc. Existen motores de reglas [4] [6] [8] que usan este diseño para construir modelos e inyectar dichos objetos en las reglas. La regla recupera los valores de cada atributo para procesarlos según la validación programada.

Haciendo uso del formato de intercambio de datos Json [17] se propone crear objetos dinámicos para cada dominio de negocios, usando un esquema padre-hijo que representen la relación objeto-atributo de los modelos orientados a objetos tradicionales.

El primer paso es construir un modelo desde un mensaje Json (ver Tabla 1), para ello se definen 2 tipos de objetos: Nivel(level) y Entidad(entity) que representan una raíz con entidades asociadas. Se clasifican como entidades básicas ej. nombre, apellido porque dependen de la raíz en forma directa y otras más complejas que agrupan entidades como tipo y nro. documento, por ejemplo, identificación (identification). Esta configuración inicial es detallada en Tabla 2.

Tabla 1. Desarrollo de modelos con Json

```
"payer": {
      "firtsname": "Charles",
      "surname": "Luevano",
      "email": "charles@hotmail.com",
      "date_created": "2015-06-
02T12:58:41.425-04:00",
      "phone": {
         "area_code": "011",
         "number": "949 128 866"
        },
      "identification": {
         "type": "DNI",
         "number": "12345678"
      },
      {
      "card_number": "4507990000004905",
      "card_expiration_month": "08",
      "card_expiration_year": "20",
      "security_code": "123",
      "card_holder_name": "John Doe",
      "card_holder_identification": {
          "type": "dni",
          "number": "25123456"
       }
     },
      "address": {
          "street_name": "Cuesta Miguel A
rmendáriz",
          "street_number": "1004",
          "zip_code": "11020"
        }
    }
```

Tabla 2. Desarrollo de modelos con Json

Level	Entity	
payer		firstname
		surname
		email
	phone	area_code
	phone	number
	identification	type
	identification	number
		card_number
		card_expiration_year
		security_code
		card_holder_name
	card_holder_identification	type
	card_holder_identification	number
	address	street_name
	address	street_number
	address	zip_code

El segundo paso es asociar un tipo de dato a cada entidad, donde los tipos básicos son números, fechas o valores alfanuméricos. La Tabla 3 contiene un modelo de negocios con entidades asociados a tipo de datos.

Tabla 3. Relación de tipo de datos con entidades

Entity		Data Type
	firstname	string
	surname	string
	email	email
phone	area_code	phone
phone	number	phone
identification	type	identification
identification	number	identification
	card_number	number
	card_expiration_year	number
	security_code	number
	card_holder_name	string
card_holder_identification	type	number
holder	number	number
address	street_name	string
address	street_number	number
address	zip_code	number

Por defecto cada tipo de datos tiene asociada funciones predefinidas, que son bloque de código que procesan un valor según su tipo. Hay funciones que ejecutan servicios rest, por ejemplo, para la entidad identificación. La Tabla 4 contiene un breve resumen de las funciones existentes.

Tabla 4. Funciones del motor de reglas

Data type	Function
number	< > == !=, not null
string	contiene, not null, comienza, finaliza
date	< > == !=, not null, is date,
email	not null, is email
identification	not null, is dni, is cuit, is cuil, api rest
phone	not null, is phone

En la Figura. 1 se define una regla para el modelo desarrollado de ejemplo. Nombre es una etiqueta que representa al objeto utilizado y descripción contiene el detalle genérico de la regla, ambos datos son descriptivos para uso de usuarios. Nivel es un objeto de negocios, Entidad un atributo, Función un bloque de código que procesa un valor de un atributo de acuerdo con su tipo de dato. Constante es un valor usado para validar si la regla es verdadera.

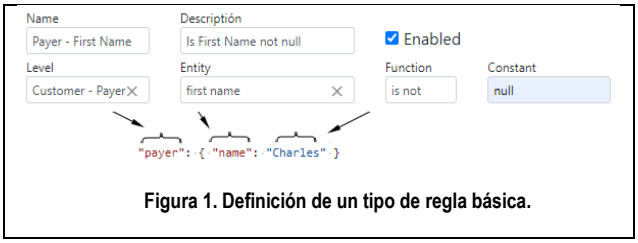

Figura 1. Definición de un tipo de regla básica.

En resumen, a partir de un mensaje en formato Json se propone generar un modelo de negocios sin desarrollar código, pero donde cada modelo es almacenado en base de datos. Una regla es creada a partir de un Nivel y una Entidad, la función asociada determina que y como se procesa un valor para comparar el resultado contra el valor de la constante.

3. Orquestación de Reglas para Integrar Conocimiento

Los DSS tienen una base de conocimiento centralizada que caracteriza el dominio del problema a resolver y un motor de inferencia encargado de dar respuesta a una pregunta mediante búsqueda, análisis y razonamiento inteligente [18].

Un motor de reglas dispone de información para dar estas respuestas, sin embargo, en un mundo globalizado una organización puede no disponer de todos los datos necesarios para el análisis completo de un dominio. Para citar un ejemplo, en prevención de fraude es necesario consultar información de personas en servicios externos dedicados al análisis crediticio.

Ejemplo de orquestación de reglas de negocios sobre prevención de fraude.

Caso de Uso: para prevenir fraudes en la venta de los artículos x1…xn entre los días 20 y el último día hábil del mes por un monto superior a los u$s 15.000 que provengan de clientes que no son Vip y la edad este entre los 18 y 35 años con estado crediticio en rojo o presenten veraz, la venta debe No ser aprobada en forma automática y pasa al área de investigación.

De acuerdo con [19], el fraude es una actividad delictiva difícil de estimar cuantitativamente ya que la metodología para cometer fraudes cambia continuamente. El desafío es desarrollar un motor de reglas flexible a los cambios del contexto, que permita consultar a una gran variedad de repositorios para ampliar el espacio de análisis y con capacidad de adaptación a nuevos entornos.

Variables.
a=20; b=[ultimo_dia_habil_mes]; c=u$s; d=15.000; e=18; f=35; h=Se recomienda Investigar; i= [20,23,25,34,10]; CustomerStatus = ApiExterna.getCustomerStatus({body}); k=Aprobado

Variables Proposicionales: p= día>=a; q= día<=b; r= Moneda == c; m= Importe >d; o= EsClienteVP == True; z= Edad >= e; t=Edad <= f; u= CustomerStatus.Status == "RED"; l= CustomerStatus.Veraz == True; s= [i1], [i2], [in]; j= Alert=h; k= VentaAprobada=True

Representación Simbólica:
((p Λ q) Λ (r Λ m) Λ (┐o Λ z Λ t) Λ (u Λ l) Λ (s1 v s2 v sn)) -> (h Λ ┐k)

Los símbolos clarifican el entendimiento de la lógica del problema, evitando la vaguedad o ambigüedad propios del lenguaje natural [20].

```
curl --location --request POST 'https://{url}/api/v0/getcustomerstatus' \
--header 'Content-Type: application/json' \
--header 'Accept: application/json' \
--data-raw '{"apiKey":"1325851d15g9xw561x4h",
    "type": {identification.type}, "number": {identification.number}}'
Respuesta: {"status": "GREEN-RED", "veraz": True/False, "name": " Charles"}
```

Figura 2. Servicio externo de evaluación de riesgo crediticio.

Pseudocódigo parcial del paquete de reglas.

```
Inicio
 Si(Regla 1 -- > Si (Rango de
Fecha(día))
y
 Regla 2 -- > Si (Valor y Moneda de
Venta (moneda, importe))
y
 Regla 3 -- > Si (Tipo de
Cliente(idcliente))
y
 Regla 4 -- > Si (Rango de Edad(edad))
y
 Regla 5 -- > Si (Estado
crediticio({body}))
y
 Regla 6 -- > Si (Lista de artículos
(lista)))
y
 Regla 7 -- > Venta No Aprobada + Alert)
 De lo contrario
 Regla 8 -- > Venta aprobada
Fin
```

Las reglas 1 al 4 y 6 procesan código tradicional en este caso en nodejs, acá llamadas reglas-código. La regla 5 tiene como objetivo validar el estado crediticio de un cliente e involucra el uso de información externa. La función {is customer status} invoca a un servicio responsable del análisis y la respuesta es inyectada al modelo de negocios, ver Figura. 2. Este tipo de funciones tiene responsabilidades acotadas y bien definidas, ejecutan el siguiente ciclo de vida:

a) Recupera de tabla la definición del curl Figura. 2.

b) Mapear los valores de la entidad payer.identification.[type,number] en el curl {"type": {identification.type}, "number": {identification.number}}

c) Temas de seguridad: recuperar token si es necesario y mapea el mismo dentro de la definición del curl.

d) Invocar al servicio, gestionar toda respuesta no satisfactoria o donde http status code sea distinto de 200 (respuesta ok) [21]. La gestión incluye alertas internas y generación de un valor por default, por ejemplo, {payer.status='GREEN', veraz = false}

e) En caso de http status code == 200, mapea la respuesta con la entidad asociada, por ejemplo, {payer.status = servicio.body.status, payer.veraz = servicio.body.veraz}

Ejecutar la validación definida en la función, en este caso payer.status == 'RED'

Los pasos antes enumerados son ejecutados por la regla de evaluación de riesgo crediticio, Figura. 3.

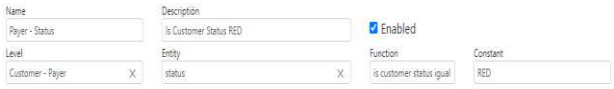

Figura 3. Regla de Ejecución de Evaluación de Riesgo Crediticio.

En las subsiguientes reglas se puede usar la respuesta generada por el servicio (puntos d o e) para validar nuevas condiciones, como por ejemplo en la Figura. 4.

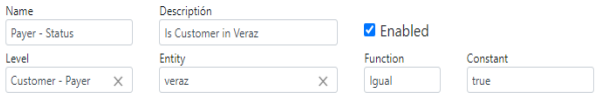

Figura 4. Regla de Evaluación sobre Respuesta Dinámica.

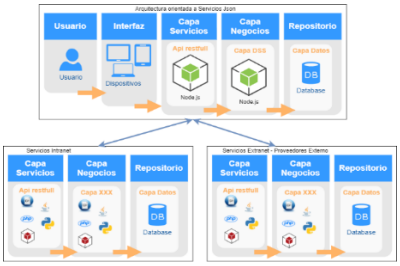

Figura 5. Arquitectura del Motor de Reglas Orientado a Servicios.

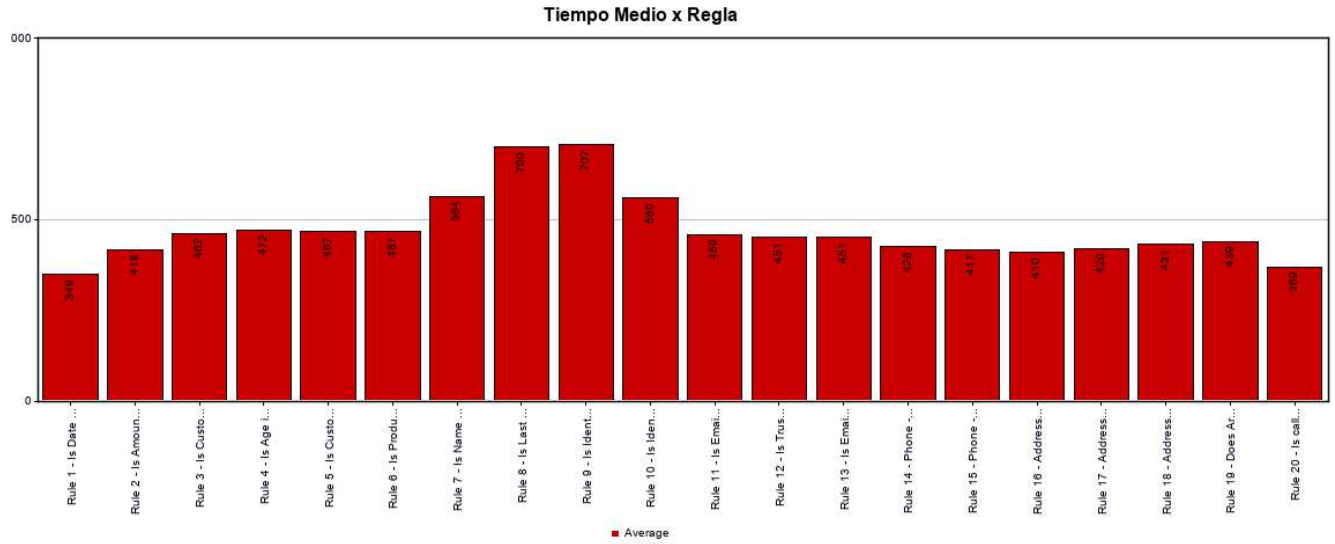

Figura 6. Tiempo promedio por regla.

Tabla 5. Medición de performance para paquete de reglas

Regla	Avg	Min	Max	Std. Dev.	Throughput	Receive dKB/sec	Sent KB/sec	Avg. Bytes
Rule 1 - Is Date In Range	747	490	1056	144.34	39.96803	7.92	4.88	203
Rule 2 - Is Amount & Currency > u$s 15.000	592	549	653	28.69	40.98361	7.96	5.56	199
Rule 3 - Is Customer VIP	660	608	702	30.3	40.35513	19.82	5.16	503
Rule 4 - Is Age in Range	580	553	627	18	42.44482	21.97	5.8	530
Rule 5 - Is Customer Status RED	580	548	618	18.84	43.59198	21.54	11.37	506
Rule 6 - Is Customer in Veraz	597	576	632	12.45	41.32231	20.42	10.77	506
Rule 7 - Is Product In Range	656	626	696	18.49	40.03203	19.66	8.17	503
Rule 8 - Is First Name not null	580	536	641	29.25	43.32756	22.04	5.8	521
Rule 9 - Is Last Name not null	531	509	562	11.85	46.04052	23.42	6.16	521
Rule 10 - Is Identity Number CUIT or CUIL	570	67	1058	99.6	45.41326	23.5	6.21	530
Rule 11 - Is Identity Number CUIT	520	492	551	12.74	47.57374	25.46	6.78	548
Rule 12 - Is Email Valid	530	492	564	15.94	47.52852	25.99	6.87	560
Rule 13 - Is Trusth Email Domain	549	521	579	16.85	45.57885	25.06	6.63	563
Rule 14 - Is Email not in Black List	556	516	586	24.25	46.55493	25.87	6.87	569
Rule 15 - Phone - Is Area Code Valid	495	479	513	6.25	50.1002	25.05	6.56	512
Rule 16 - Phone - Is Number Valid	554	490	595	26.47	46.59832	24.39	6.46	536
Rule 17 - Address - Is Zip Code Number and Not Null	578	550	609	10.17	43.40278	21.57	5.64	509
Rule 18 - Address - Is Street Name not null	533	489	591	31.85	46.3392	24.93	6.65	551
Rule 19 - Address - Is Street Number and Not Null	512	483	547	15.85	49.50495	24.46	6.38	506
Rule 20 - Does Area Code belongs Address	551	522	573	10.95	46.81648	24.09	10.84	527
Rule 21 - Is call from an IP valid	523	487	557	19.22	48.78049	25.53	6.76	536
TOTAL	571	67	1058	71.25	84.28996	40.92	13.14	497.1

La orquestación se lleva a cabo integrando dos tipos de reglas, las tradicionales que ejecutan código nativo en nodejs, y las reglas que invocan a servicios sean estos propios (internos) o no propios (externos). La función asociada a la regla tiene la inteligencia para comunicarse con los servicios, crear el {body} necesario y mapear la respuesta a una entidad ya configurada en formato nivel/entidad. Con la respuesta incorporada al modelo la regla puede usar esta información externa para validar si la misma es verdadera o falsa. La Figura. 5 representa la arquitectura de esta propuesta, haciendo énfasis en la comunicación desde la regla en la capa de servicios del motor hacia los servicios disponibles.

La creación de modelos de negocios heterogéneos y dinámicos permite construir reglas más robustas de forma creativa y flexible. La integración de conocimiento se logra mediante funciones que ejecutan código tradicional con información local y funciones que se comunican con servicios expuestos sean internos o externos.

4. Pruebas 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 [22] por este motivo el tiempo incurrido para ejecutar un paquete de reglas no debe ser considerablemente alto.

La Tabla 5 resume las métricas de la prueba de performance sobre 50 ejecuciones concurrentes evaluado con la herramienta Apache JMeter [23]. Avg 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. Los tiempos de respuestas son aceptables para un motor de reglas expuesto como servicio, aunque el tiempo total dependerá del tiempo consumido por los servicios externos utilizados. La Figura. 6 resume el tiempo medio por cada regla.

Ambiente de pruebas: OS Name Microsoft 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

Los DSS nacen para ayudar a las personas a tomar decisiones, los primeros fueron mono programas escritos en lenguajes como Prolog o Clips, pero a medida que la tecnología y las necesidades evolucionaron estos sistemas se fueron adaptando para cubrir nuevas necesidades. Se migraron programas de escritorio a plataformas web con accesos a base de datos para enriquecer la validación de reglas. Por la popularidad adquirida se sumaron muchas áreas de la industria y cada una fue sumando distintos algoritmos acordes a sus requisitos de negocios. La globalización implica nuevos retos para estos sistemas como el acceso a datos almacenados en ubicaciones externas, el procesamiento de conocimiento externo, uso de librerías de terceros y la implementación de servicios para aprovechar el acceso a estos recursos, etc. En esta propuesta se implementa una mejora al motor de reglas desacoplado presentado en [22], sumando el acceso a servicios expuestos en internet. De esta forma, se puede recuperar de una fuente externa información para validar reglas, logrando abstraer al DSS de las particularidades de los pasos ejecutados por el servicio para formar la respuesta.

Este diseño innovador maximiza la creatividad en el desarrollo de reglas, permite crear modelo de negocios dinámicos y robustos, minimiza el acoplamiento entre dominios, aumentar la escalabilidad de los paquetes de reglas, maximiza la confiabilidad de las respuestas y elimina los programas monolíticos de difícil mantenimiento. Como trabajo futuro se puede mencionar la incorporación de lógica no simbólica para soportar algoritmos de machine learning e implementar microservicios con service bus para mejorar la experiencia de crear funciones.

Referencias

[1] R. Jain, Decision Support Systems: An Overview, Decision Support System in Agriculture using Quantitative Analysis, Udaipur, Agrotech Publishing Academy, 2016, pp. 42-50.

[2] I. Nizetic, K. Fertalj, and B. Milasinovic, “An Overview of Decision Support System Concepts,” Race, vol. 3, no. 2, pp. 1–14, 2007.

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

[4] 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.

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

[6] 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.

[7] 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).

[8] 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.

[9] 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.

[10] 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.

[11] Chapman, M. D, & Curcin, V. (2019). A Microservice Architecture for the Design of Computer-Interpretable.

[12] 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.

[13] 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

[14] 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.

[15] 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.

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

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

[18] García, A. (2013) Inteligencia Artificial, Fundamentos, práctica y aplicaciones, pp 171-175. México: AlfaOmega.

[19] FRISS. Strong Market Demand for Fraud Analytics in North America Drives FRISS to Expand Operations. https://www.friss.com/noticias/strong-market-demand-for-fraud-analytics-in-north-america-drives-friss-to-expand-operations/ [Acceso Junio 2021]

[20] Copi, I. (2001). Lógica simbólica. México: Compañía Editorial Continental.

[21] W3C - World Wide Web Consortium, https://www.w3.org/Protocols/HTTP/HTRESP.html [Acceso Agosto 2021]

[22] 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

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

9° CoNaIISI
2021

Congreso Nacional de
Ingeniería Informática y
Sistemas de Información