Adaptación del juego de cartas Clash Royale a una estructura RDF

(Web Semántica - Resource Description Framework)

Katheryn Ximena Peralta Haro
Facultad de Ciencias
Universidad Nacional de Ingeniería
Lima, Perú
katheryn.peralta.h@uni.pe

Renzo Renato Quispe Amao
Facultad de Ciencias
Universidad Nacional de Ingeniería
Lima, Perú
rrquispea@uni.pe

José Javier Campó Beraún
Facultad de Ciencias
Universidad Nacional de Ingeniería
Lima, Perú
jcampob@uni.pe

Franklin Hamer Jara Ocas
Facultad de Ciencias
Universidad Nacional de Ingeniería
Lima, Perú
frnaklinh.jara.o@gmail.com

***Abstract*—Este artículo presenta los beneficios que ofrece la web semántica y la manipulación de una estructura RDF. Utilizamos Web Scraping para obtener la información del juego Clash Royale, luego construir un esquema RDF con ayuda de RDFlib (módulo de Python), para poder expandir la base de conocimientos del juego.**

***Index Terms*—RDF, RDFlib, URI, Web Scraping.**

I. Introducción

En el año 2000, Berners-lee ofreció una conferencia en el marco de la W3C donde propuso que la información debe ser reunida de forma que un buscador pueda comprender en lugar de simplemente ponerlo en la lista. La web semántica sería una red de documentos de "meta información" que permitan, a su vez, búsquedas más inteligentes. La idea sería aumentar la inteligencia de los contenidos de las páginas web dotando de contenido semántico. Con la web actual solo es posible almacenar datos, pero no es capaz de pensar ni de entender el contenido de las páginas web. En mayo del 2001, Tim Berners, James Hendler y Ora Lassila publican un artículo en la revista Scientific American titulado "The Semantic Web: A new form of web content that is meaniful to computers will unleash a revolution of new possibilities" donde la definen como:"La Web Semántica es una extensión de la Web actual en la que a la información se le da un significado bien definido, lo que permite que las computadoras y las personas trabajen en cooperación". [1]
Para poder explotar la Web semántica, se necesitan lenguajes semánticos más potentes, esto es, lenguajes de marcado capaces de representar el conocimiento basándose en el uso de metadatos y ontologías. Utilizando anotaciones RDF y RDF Schema se pueden presentar algunas facetas sobre conceptos de un dominio del conocimiento y se puede, mediante relaciones taxonómicas, crear una jerarquía de conceptos.

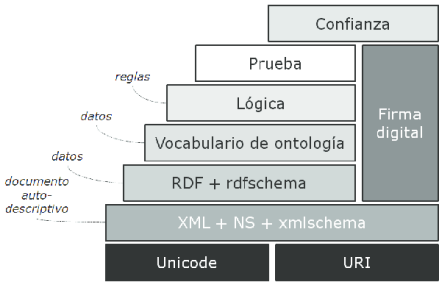

Fig. 1. RDF se encuentra en una capa intermedia ofreciendo un puente entre los recursos (accedidos por los usuarios a través del URI) y los niveles más altos de abstracción (significado y confianza sobre los datos)

II. Conceptos Previos

A. Web scraping

Web Scraping, también conocido como extracción o recolección web, es una técnica para extraer datos de la World Wide Web (WWW) y guardarlos en un sistema de archivos o base de datos para su posterior recuperación o análisis.
Esto se logra ya sea manualmente por un usuario o automáticamente por un bot o un rastreador web.
Las herramientas de scraping web de última generación no solo son capaces de analizar los lenguajes de marcado o archivos JSON, sino que también se integran con el análisis visual por computadora y el procesamiento de lenguaje natural para simular cómo los usuarios humanos navegan por el contenido web [3]. El scraping de datos de Internet se puede dividir en dos pasos secuenciales; adquirir recursos web y luego extraer la información deseada de los datos adquiridos.

1) Proceso de extracción: Específicamente, un programa de scraping web comienza componiendo una solicitud HTTP para adquirir recursos de un sitio web específico. Esta solicitud puede formatearse en una URL que contiene una consulta GET

o en un mensaje HTTP que contiene una consulta POST. Una vez que la solicitud sea recibida y procesada con éxito por el sitio web de destino, el recurso solicitado se recuperará del sitio web y luego se enviará de vuelta al programa de scraping web.

El recurso puede estar en múltiples formatos, como páginas web creadas a partir de HTML, fuentes de datos en formato XML o JSON, o datos multimedia como imágenes, audio o archivos de vídeo. Una vez que se descargan los datos web, el proceso de extracción continúa analizando, formateando y organizando los datos de forma estructurada.

Hay dos módulos esenciales de un programa de web scraping: un módulo para componer una solicitud HTTP, como `Urllib2` o `seleniu`, y otro para analizar y extraer información de código HTML sin procesar, como `BeautifulSoup` o `Pyquery`.

- **Urllib2:** Define un conjunto de funciones para manejar solicitudes HTTP, como autenticación, redirecciones, cookies, etc. Mientras que Selenium es un contenedor de navegador web que crea un navegador web, como Google Chrome o Internet Explorer, y permite a los usuarios Automatizar el proceso de navegación por un sitio web mediante programación.
- **Beautiful Soup:** Está diseñado para scraping HTML y otros documentos XML. Proporciona funciones `Pythonic` convenientes para navegar, buscar y modificar un árbol de análisis; `atoolkit` para descomponer un archivo HTML y extraer la información deseada a través de lxml o html5lib. `Beautiful Soup` puede detectar automáticamente la codificación del análisis en proceso y convertirlo en una codificación legible por el cliente.

B. Expresiones regulares

En el área de programación las expresiones regulares son un método por medio del cual se pueden realizar búsqueda de cadenas de caracteres.Sin embargo la amplitud de la búsqueda requerida de un patrón definido de caracteres, las expresiones regulares proporcionan una solución practica al problema,Adicionalmente, un uso derivadas de la búsqueda de patrones es la validación de un formato especifico en una cadena de caracteres dada, como por ejemplo flechas o identificadores. Para poder utilizar las expresiones regulares en el programa solo necesitamos una lista de datos donde evaluar. [4]

III. Resource Description Framework

A. ¿Qué es RDF?

RDF es un lenguaje para la representación de la información sobre recursos en la *World Wide Web*. Está especialmente diseñado para representar metadatos sobre recursos web. RDF se basa en la idea de identificar cosas mediante identificadores web (llamados *identificadores uniformes de recursos o URI*) y describir los recursos en términos de propiedades similares y valores de propiedad. Esto permite que RDF represente declaraciones simples sobre recursos como un *gráfico* de nodos y arcos que representan los recursos, sus propiedades y valores. [5]

B. Estructura RDF

RDF proporciona un vocabulario de modelado de datos para datos RDF. La estructura RDF es bastante similar a otros modelos de datos, como pueden ser los diagramas de clases en la programación orientada a objetos. El esquema RDF se diferencia que en lugar de definir una clase en términos de las propiedades que pueden tener sus instancias. El esquema RDF describe las propiedades en términos de la clases de recurso a las que se aplican. La idea de RDF se basa en expresar enunciados simples sobre recursos, donde cada enunciado consta de un sujeto, predicado y un objeto. En términos de RDF el sujeto puede ser un recurso o propiedad, el predicado igualmente un recurso o propiedad a diferencia del objeto que puede ser un recurso o valor. [6]

- Un **recurso** es cualquier cosa que puede ser identificada unívocamente por un URI(Uniform Resource Identifier). Identificadores universales para cualquier recurso de la red o de fuera de ella.
- Una **propiedad** es un recurso que tiene un nombre y que puede usarse como una propiedad, por ejemplo, autor o título. En muchos casos todo lo que nos importa en realidad es el nombre, pero una propiedad necesita ser un recurso de forma tal que pueda tener sus propias propiedades.
- Un **valor** es la representación que toma la propiedad.

1) RDF Schema: Un esquema RDF proporciona información sobre la interpretación de una sentencia dada en un modelo de datos RDF. Mientras un esquema XML puede utilizarse para validar la sintaxis de una expresión XML, un esquema sintáctico sólo no es suficiente para los objetivos de RDF. Los esquemas RDF pueden también especificar restricciones que deben seguirse por estos modelos de datos. El trabajo futuro en torno al esquema RDF y al esquema XML podría facilitar la sencilla combinación de reglas sintácticas y semánticas para ambos. [6] La RDF Schema es una extensión de RDF permite especificar una jerarquía explícita de clases de recursos y propiedades que describen estas clases, junto con las restricciones sobre las combinaciones permitidas de clases, propiedades y valores. En general, RDF Schema permite:

- Definir no sólo las propiedades de un recurso (ej. título, autor, materia, tamaño, color, etc.) sino que también definir los tipos de recursos que se describirán (libros, páginas web, personas, empresas, etc.).
- Proporcionar información sobre la interpretación de una sentencia en un modelo de datos RDF (semántica).
- Definir la herencia de clases para crear una taxonomía del modelo; Esta es una poderosa característica de RDF Schema dado que en ella radica la extensibilidad en cuanto a elaboración de nuevos esquemas.
- Definir las relaciones entre recursos y propiedades, lo cual ayudará a inferir información del modelo y a la vez mejorar los procesos de búsqueda.

- Especificar restricciones que deben seguirse por estos modelos de datos.

C. Clases, propiedades y restricciones

1) Clases: Los recursos siguientes son las clases principales que se definen como parte del vocabulario del Esquema RDF. Cada modelo RDF que se traza sobre el `namespace` del Esquema RDF (implícitamente) los incluye. Estas son las principales:

- **Rdfs:Resource:** Toda sentencia RDF se considera una instancia de esta clase, por ello debe poseer URI que lo identifique y permita el acceso a su descripción.
- **Rdf:Property:** Es la clase de todas las propiedades utilizadas en la caracterización de las instancias de `Rdfs:Resource`. Son utilizados como predicados de los triples, la semántica de un triple depende de la `property` utilizada como predicado.
- **Rdfs:Class:** Este corresponde con el concepto genérico de un tipo o categoría semejante a la noción de clase en los lenguajes de programación orientados a objetos como Java. Cuando un esquema define una nueva clase, el recurso que representa esa clase debe tener una propiedad `Rdf:type` cuyo valor es el recurso `Rdfs:Class`. Las clases RDF puede definirse para representar cualquier cosa, como páginas web, personas, tipos de documento, bases de datos o conceptos abstractos.

2) Propiedades: Son objetos específicos de una categoría (instancias) de la clase `Rdf:Property` y proporcionan un mecanismo para expresar las relaciones entre las clases y sus objetos específicos de categoría (instancias) o superclases. A continuación se enumera las principales:

- **Rdf:type:** Esta propiedad indica que un recurso es miembro de una clase, de tal forma que tiene todas las características que se obtiene de un miembro de esa clase. Es un objeto específico de la categoría (instancia) de la clase especificada.
- **Rdfs:subClassOf:** Modela jerarquía de clases, donde una clase puede ser subclase de otras subclases. La propiedad `Rdfs:subClassOf` es transitiva. Si la clase A es una subclase de otra clase B más amplia, y B es una subclase de C, entonces A es también implícitamente una subclase de C. Por lo tanto, los recursos que son objetos específicos de una categoría(instancias) de la clase A serán también (instancias) de C, puesto que A es un subconjunto de ambas, tanto de B como de C.
- **Rdfs:subPropertyOf:** Es un objeto específico de una categoría(instancias) de `Rdf:Property` que se utiliza para especificar que una propiedad es una especialización de otra. En tal caso se aplica la propiedad de transitividad por tanto si una propiedad P2 es una subpropiedad de `Rdfs:subPropertyOf` otra propiedad P1, y si un recurso R tiene una propiedad P2 con valor V, esto implica que el recurso R también tiene la propiedad R1 con valor V.
- **Rdfs:label:** Es para dar un nombre al sujeto legible por humanos.
- **Rdfs:comment:** Es para proveer de una descripción más larga del recurso.
- **Rdfs:seeAlso:** Especifica un recurso que podría proporcionar información adicional sobre el recurso sujeto.
- **Rdfs:isDefinedBy:** La propiedad `Rdfs:isDefinedBy` es una subpropiedad de `Rdfs:seeAlso` e indica el recurso que define el recurso sujeto.

3) Restricciones: La especificación de Rdfs presenta un vocabulario RDF para hacer sentencias sobre restricciones en el uso de las propiedades y clase en datos RDF. Por ejemplo, un esquema RDF podría describir las limitaciones de los tipos de valores que son válidos para una propiedad, o las clases para las que tiene sentido asignar tales propiedades. El Esquema RDF proporciona un mecanismo para describir tales restricciones pero no dice si, ni cómo, una aplicación puede procesar la información restringida. ht

- **Rdfs:ConstraintResource:** Define la clase de todas las restricciones.
- **Rdfs:ConstrainProperty:** Es un subconjunto de rdfs:ConstraintResource la cual tiene dos instancias:`Rdfs:range` y `Rdfs:domain`.
- **Rdfs:range:** Se usan para restringir el rango (un conjunto de valores válidos para la propiedad). No se permite expresar más de una restricción de rango sobre una propiedad.
- **Rdfs:domain:** Se usan para restringir el dominio (el conjunto de recursos que puede tener una determina propiedad). En dominios si se permite expresar más de una restricción de dominio, y se interpreta como una unión de dominios.

4) Otras Clases:

- **Rdfs:Literal:** Valores atómicos como conjuntos de caracteres (string) textuales, son ejemplos literales RDF.
- **Rdfs:value:** Identifica el principal valor (normalmente un string) de una propiedad cuando el valor de la propiedad es un recurso estructurado.

IV. Descripción de recursos usando RDF

A. Descripción del juego

Clash Royale es el juego de lucha de ritmo rápido en el que recolectas cartas y luchas contra otros jugadores en tiempo real. Defiende tus torres y destruye las de tu enemigo con tus mejores estrategias de ataque y defensa para ganar la partida [8].

1) ¿Qué tipos de cartas existen?: Existen cuatro tipos de cartas dependiendo de su poder que podrás ir desbloqueando según avances en el juego: **comunes**, **especiales**, **épicas** y **legendarias**. Cada una de ellas tiene un **nivel de poder**. Para invocar una carta en la pelea tendrás que tener en cuenta el costo de elixir que consume, que será mayor cuanto más poder tenga. ¡Cuidado! si usas muchas cartas con un consumo de elixir muy elevado, será muy difícil invocar varias cartas seguidas.

2) ¿Cómo se lucha?: En las batallas, los dos adversarios disponen de sus respectivos mazos en **rotaciones de cuatro**. Según utilices las cartas, irán apareciendo otras nuevas para que las sigas combinando a tu criterio. Cada jugador tiene una barra de **diez unidades de elixir** que se va a recargando progresivamente.

3) ¿Cuál es el objetivo de las batallas?: En la arena tu campo es la **mitad inferior** y el del oponente, la **mitad superior**. Tus tropas tendrán que alcanzar las **torres del enemigo** para derribarlas a toda costa antes de que se acaben los **tres minutos de tiempo**. Para ello tendrás que derrotar a los enemigos que el rival invoque al campo de batalla con el objetivo de defenderte o atacar sus tropas.

4) ¿Qué es el elixir?: El elixir es el **recurso más importante** en Clash Royale pues se usa para poner cartas en el campo de batalla, cada carta tiene su respectivo **costo de elixir**, por esta razón saber utilizarlo en el momento indicado puede cambiar la marea de la partida, en función a esta mecánica es que se desarrollan muchos tipos de estrategias.
Cada jugador empieza con **5 de elixir** y se genera cada cierto tiempo hasta juntar un **máximo de 10**, el tiempo de **abastecimiento de elixir** sube a medida que la partida avanza, por ejemplo, en los primeros 2 minutos se agrega 1 gota de elixir cada 2.8 segundos, pasados los 2 minutos la velocidad de abastecimiento se duplica y pasados los 4 minutos se triplica. En lo general se recomienda tener un maso de cartas con una **media de 2.5 a 4.5 de elixir** para tener un óptimo desempeño en el juego.

Fig. 2. Portada del juego móvil de clash Royale

B. Recursos

Los recursos utilizados son de la página Stats Royale [6]. Con ayuda del Web scraping recolectamos los recursos de las cartas a utilizar en la creación de la base de conocimiento del modelo RDF para Clash Royale. El diagrama a continuación muestra las principales relaciones entre las clases principales y la clase característica que tiene las propiedades de las cartas del juego.

- Card
  - -Troop
    - -Giant
    - -Dead
    - -Creature
    - -Dragon
    - -Goblin
    - -Magician
    - -Human
  - -Build
    - -Hut
    - -Tower
  - -Spell
- characteristic
  - -Name
  - -Level
  - -Quality
    - -Rare
    - -Epic
    - -Common
    - -Legendary
  - -Objective
    - -Air
    - -Ground
    - -Building
  - -Scope
    - -Melee
    - -Near
    - -Far
  - -Speed
    - -Slow
    - -Medium
    - -Fast
    - -Very Fast
  - -Radio
  - -Timeoflife
  - -Lifepoints
  - -Hurt
  - -Damage
  - -Areadamage

V. Manipulación de RDF

Como ya se ha explicado anteriormente, los RDF permiten almacenar los datos de manera efectiva, se puede decir que almacena los datos en partes, esto para que se puedan manipular dichos datos de mejor manera de la que normalmente se hace, por ejemplo los RDF nos permiten obtener las subclases de una clase, el nombre de cada objeto almacenado, verificar si un elemento del RDF pertenece a cierta clase, en fin, manipular cualquier característica definida en el grafo para obtener las relaciones que deseemos.

- **Obtener sujeto, predicado y objeto:**
  A través del bucle **for** se obtienen los triples creados (sujeto, predicado y objeto).

```
#imprimir tripletas
for s, p, o in g:
    print(s, p, o)
```

```
https://statsroyale.com/es/card/Princess https://statsroyale.com/es/areada
mage 140
https://statsroyale.com/es/card/Elixir+Golem https://statsroyale.com/es/hu
rt 211
https://statsroyale.com/es/card/Electro+Wizard https://statsroyale.com/es/
areadamage 75
https://statsroyale.com/es/card/Ice+Golem https://statsroyale.com/es/objec
tive https://statsroyale.com/es/building
https://statsroyale.com/es/ground http://www.w3.org/2000/01/rdf-schema#sub
PropertyOf https://statsroyale.com/es/objective
https://statsroyale.com/es/duende https://statsroyale.com/es/name duende
https://statsroyale.com/es/card/Goblin+Giant https://statsroyale.com/es/sp
eed https://statsroyale.com/es/medium
https://statsroyale.com/es/card/Lava+Hound https://statsroyale.com/es/name
Sabueso de lava
```

Fig. 3. Resultado de obtener los triples del grafo RDF creado.

- **Obtener clases superiores:**
  A través del método **transitive_objects** podemos obtener todas las clases que estén en la escala jerárquica arriba de otra clase.

Clases superiores

```
clase_base = n.giant
for s in g.transitive_objects(clase_base, RDFS.subClassOf):
    print(g.value(s, name))
```

```
gigante
tropas
cartas
```

Fig. 4. Resultado de obtener todas las clases superiores dependientes de una clase.

- **Obtener clases inferiores:**
  A través del método **transitive_subjects** podemos obtener todas las clases que estén en la escala jerárquica debajo de otra clase.

```
clase_base = n.card
for s in g.transitive_subjects (RDFS.subClassOf,clase_base):
    print(s,g.value(s, name))
```

```
https://statsroyale.com/es/card cartas
https://statsroyale.com/es/build estructuras
https://statsroyale.com/es/tower torre
https://statsroyale.com/es/hut None
https://statsroyale.com/es/spell hechizos
https://statsroyale.com/es/troop tropas
https://statsroyale.com/es/dead muerto
https://statsroyale.com/es/creature criatura
https://statsroyale.com/es/human human
https://statsroyale.com/es/magic magico
https://statsroyale.com/es/duende duende
https://statsroyale.com/es/dragon dragon
https://statsroyale.com/es/giant gigante
```

Fig. 5. Resultado de obtener todas las clases inferiores dependientes de una clase.

- **Clasificar cada elemento por su debida clase:**
  A través del método **subjects** podemos obtener las instancias de una misma clase.

```
#Clasificar por clase tropas, estructuras y hechizos
tipos = [n.troop,n.build,n.spell]
for tipo in tipos:
    print("..............",(g.value(tipo, name)).upper(),"..............")
    for carta in g.subjects(RDF.type,tipo):
        print(g.value(carta, name))
```

```
.............. TROPAS ..............
Gólem de hielo
Leñador
Murciélagos
Chispitas
Esbirros
Globo bombástico
Gigante noble
Montacarneros
Barril de esqueletos
Dragón infernal
```

Fig. 6. Resultado de obtener las instancias de una misma clase.

- **Combate entre dos cartas:**
  Se crea una simulación usando las inferencias anteriores. Se definen 5 cartas, el mazo de los jugadores serán 4 de estas cartas. Se escoge que cartas van a enfrentarse hasta que todas las cartas del enemigo tengan 0 de vida o escojan la opción salir.

```
CARTAS DE JUGADOR 1:
Nombre:  Minero , vida:  1000.0 , daño:  133.0
Nombre:  Príncipe oscuro , vida:  1030.0 , daño:  158.0
Nombre:  Duende gigante , vida:  2616.0 , daño:  97.0
Nombre:  Valquiria , vida:  1654.0 , daño:  147.0
CARTAS DE JUGADOR 2:
Nombre:  Mosquetera , vida:  598.0 , daño:  164.0
Nombre:  Príncipe oscuro , vida:  1030.0 , daño:  158.0
Nombre:  Duende gigante , vida:  2616.0 , daño:  97.0
Nombre:  Valquiria , vida:  1654.0 , daño:  147.0
INICIA EL COMBATE!:
Se encontró la unidad
Se encontró la unidad
Comienza el combate entre  Minero  y  Príncipe oscuro
Vida de  Minero  :  1000.0 , Daño:  133.0
Vida de  Príncipe oscuro  :  1030.0 , Daño:  158.0
Carta de jugador 1 ha ganado, Minero  ha ganado!, le queda 69.0  de vida
INICIA EL COMBATE!:
Se encontró la unidad
Se encontró la unidad
Comienza el combate entre  Minero  y  Mosquetera
Vida de  Minero  :  69.0 , Daño:  133.0
Vida de  Mosquetera  :  598.0 , Daño:  164.0
```

Fig. 7. Jugadores eligen invocar una carta de sus mazos y combaten.

```
Carta de jugador 2 ha ganado,  Mosquetera  ha ganado!, le queda 434.0  de vida
INICIA EL COMBATE!:
Se encontró la unidad
Se encontró la unidad
Comienza el combate entre  Príncipe oscuro  y  Mosquetera
Vida de  Príncipe oscuro  :  1030.0 , Daño:  158.0
Vida de  Mosquetera  :  434.0 , Daño:  164.0
Carta de jugador 1 ha ganado, Príncipe oscuro  ha ganado!, le queda 556.0  de vida
CARTAS DE JUGADOR 1:
Nombre:  Minero , vida:  0 , daño:  133.0
Nombre:  Príncipe oscuro , vida:  556.0 , daño:  158.0
Nombre:  Duende gigante , vida:  2616.0 , daño:  97.0
Nombre:  Valquiria , vida:  1654.0 , daño:  147.0
CARTAS DE JUGADOR 2:
Nombre:  Mosquetera , vida:  0 , daño:  164.0
Nombre:  Príncipe oscuro , vida:  0 , daño:  158.0
Nombre:  Duende gigante , vida:  2616.0 , daño:  97.0
Nombre:  Valquiria , vida:  1654.0 , daño:  147.0
INICIA EL COMBATE!:
Se encontró la unidad
Se encontró la unidad
Comienza el combate entre  Duende gigante  y  Duende gigante
Vida de  Duende gigante  :  2616.0 , Daño:  97.0
```

Fig. 8. Los jugadores siguen combatiendo mientras sus cartas mueren.

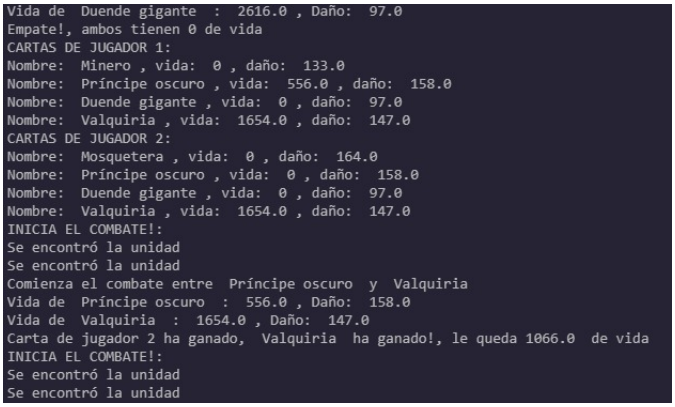

Fig. 9. Los jugadores combaten entre sus cartas disponibles.

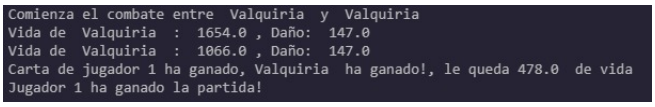

Fig. 10. Jugador 1 gana la partida.

VI. Conclusiones

A medida que las tecnologías avanzan la complejidad en cuanto a relacionar datos incrementa, esto hace que técnicas normales se vean anticuadas o sufran de deficiencia, por esta razón siempre se está en busca de nuevas técnicas que permitan una óptima clasificación de datos. Los RDF nacen como una manera para mejorar esta manipulación de datos, en el presente proyecto se ha llegado a comprobar ello, a través de un grafo en el cual se han creado elementos que poseen múltiples características, clasificados en clases se ha organizado de manera óptima para poder obtener inferencias del juego Clash Royale como por ejemplo obtener todos los triples del grafo, obtener clases superiores, clases inferiores, obtener la calidad de la carta, obtener los atributos de la carta, clasificar las cartas por los daños, clasificar las cartas por tipos, emular un combate, entre otras, llegando a realizar el objetivo de clasificar de mejor manera datos y relacionarlos.

References

[1] Frank Manola, Erick Millar. RDF Primer (2003). Disponible en: https://www.w3.org/TR/rdf-primer/.
[2] Case, Karl E.Quigley, John M.Shiller, Robert J. Comparing Wealth Effects: The Stock Market versus the Housing Market.(2005). Disponible en: https://escholarship.org/uc/item/28d3s92s.
[3] J. Yi, T. Nasukawa, R. Bunescu, W. Niblack. Sentiment analyzer: extracting sentiments about a given topic using natural language processing techniques.(2003). Disponible en: https://ieeexplore.ieee.org/abstract/document/1250949
[4] Raúl López Briega. Expresiones Regulares con Python (2015).https://relopezbriega.github.io/blog/2015/07/19/expresiones-regulares-con-python/.
[5] Tim Bernes-Lee. Why RDF model is different from the XML model (1998). Disponible en: https://www.w3.org/DesignIssues/RDF-XML.html.
[6] Dan Brickle, R.V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema (2003). Disponible en: https://www.w3.org/TR/rdf-schema/.
[7] https://statsroyale.com/es.
[8] https://www.rtve.es/playz/clash-royale/como-jugar/

Enlace GitHub
https://github.com/kx22p/IA-PracticaCalificada1.git