FRIGATO LUCA

PROGETTO SEMANTIC WEB E LINKED DATA

MATRICOLA 836201

CONTENTS

Progetto Modellazione Concettuale del Web Semantico 4
1 Motivazioni 4
2 Requisiti 4
2.1.1 Finalità 4
2.1.2 Task specifici e utenti target 5
3 Dominio 6
3.1.1 Descrizione 6
3.1.2 Fonti e riferimenti bibliografici 7
4 Documentazione dominio 7
4.1.1 Ispirazioni e risorse simili (standard esistenti) 7
4.1.2 Dota Ontology 7
4.1.3 League of Legends Ontology 9
4.1.4 Esempio del dominio 11
4.1.5 Allineamenti 12
4.1.6 Skos 13
4.1.7 Pattern allineamento 14
4.1.8 Checklist per l'ontologia 14
5 LodE 15
6 Visualizzazione 15
Tassonomia delle classi 15
6.1 Principiali template 16
6.1.1 Ontology Design Pattern List e 16
6.1.2 Visualizzazione del MATCH 19
7 Queries SPARQL 19
7.1 Interaction 19
7.2 Queries 22
7.2.2 Estrazione dell'ordine in cui le abilità sono state potenziate 24
7.2.3 Estrazione dell'inventario 25
8 Applicazione web 28
8.1 Connessione a Graph Db 33
9 Conclusioni e sviluppi futuri 35

PROGETTO MODELLAZIONE CONCETTUALE DEL WEB SEMANTICO

1 MOTIVAZIONI

Nel contesto della mia passione per i giochi MOBA, ho sviluppato un'ontologia per fornire una comprensione strutturata delle complesse meccaniche inerenti a questi giochi. La motivazione alla base di questo impegno deriva dal desiderio di migliorare la strategia dei giocatori, aiutare lo sviluppo dei giochi e promuovere i progressi dell'intelligenza artificiale per i giochi.

L'ontologia serve ai giocatori per comprendere i punti di forza e di debolezza dei diversi personaggi, valutare l'efficacia delle varie strategie e ideare metodi per contrastare efficacemente gli avversari. Dal punto di vista dello sviluppo del gioco, questa ontologia può essere utile per bilanciare il gameplay. Offre una comprensione completa delle relazioni e delle interazioni tra i diversi elementi del gioco, consentendo agli sviluppatori di apportare le modifiche necessarie per garantire un'esperienza di gioco equa e coinvolgente.

Inoltre, l'ontologia ha implicazioni significative per lo sviluppo dell'intelligenza artificiale nei giochi. Può essere utilizzata per creare avversari o compagni di squadra IA più sofisticati, che portano a un gameplay più impegnativo e realistico.

In termini di analisi del dominio e di valutazione delle attuali implementazioni, è stato intrapreso uno studio approfondito delle meccaniche di gioco, delle abilità dei personaggi e delle strategie dei giocatori. Sebbene le soluzioni esistenti, come **MOBAFIRE e Leaguepedia**, offrano una grande quantità di informazioni, non sono in grado di fornire la comprensione strutturata e relazionale che può offrire un'ontologia. Per questo motivo, è stato condotto un confronto tra le conoscenze ottenute dall'ontologia e queste soluzioni esistenti, dimostrando che

2 REQUISITI

2.1.1 Finalità

Data la vastità del dominio dei giochi MOBA e la complessità che esso comporta, il progetto si è concentrato sulla formalizzazione di un sottoinsieme delle peculiarità e delle caratteristiche dei giochi MOBA. Questo approccio ha permesso un'esplorazione più maneggevole e mirata del dominio, pur cogliendo gli elementi essenziali che definiscono i giochi MOBA. Questo sottoinsieme è stato accuratamente selezionato per rappresentare gli elementi e le meccaniche di gioco chiave, fornendo una base che può essere ampliata nel lavoro futuro. L'ontologia, quindi, funge da modello semi-completo e conciso del dominio dei giochi MOBA, offrendo spunti preziosi e una comprensione strutturata di questo sottoinsieme di meccaniche di gioco.

L'obiettivo principale di questa ontologia era la modellazione della composizione e della gestione della costruzione di oggetti, un aspetto fondamentale dei giochi MOBA. Le relazioni "costruito da" e "usato per

costruire" sono state modellate meticolosamente per catturare l'intricato processo di costruzione degli oggetti in questi giochi. Questo aspetto dell'ontologia fornisce una comprensione dettagliata di come i diversi oggetti interagiscono e si combinano per formare equipaggiamenti più potenti, una parte cruciale della strategia nei giochi MOBA.

Per quanto riguarda la modellazione dei personaggi, è stata prestata particolare attenzione alla rappresentazione delle abilità dei campioni e al concetto di evoluzione. Questo include l'assegnazione di punti evoluzione alle abilità per migliorarle, riflettendo la progressione e la crescita dei personaggi nel corso della partita. Questo aspetto dell'ontologia può aiutare i giocatori a comprendere i potenziali percorsi di sviluppo dei loro personaggi e a prendere decisioni strategiche sul potenziamento delle abilità.

2.1.2 Task specifici e utenti target

I compiti specifici e gli utenti target di questa ontologia sono principalmente legati alla consultazione e alla ricerca da parte degli utenti. L'ontologia potrebbe diventare uno strumento molto utile se integrata con specifiche API di gioco per alimentare pagine semantiche che assistano gli utenti alle prime armi nell'apprendimento del gioco. Integrandosi con le API per ottenere informazioni sugli utenti, l'ontologia potrebbe fornire consigli più strutturati e quantitativi per migliorare il gioco.

Inoltre, l'ontologia potrebbe essere utilizzata per creare un sistema simile a MOBAFIRE per visualizzare informazioni sulle partite e dati sulle prestazioni. Ciò consentirebbe di utilizzare

Tasks specifici di utilizzo

- **Costruzione di oggetti:** Comprendere le relazioni tra i diversi oggetti e come si combinano per formare equipaggiamenti più potenti.
- **Modellazione dei personaggi**: Rappresentazione delle abilità dei campioni e del concetto di evoluzione, compresa l'assegnazione dei punti evoluzione alle abilità.
- **Analisi del gioco:** Fornisce una comprensione strutturata delle meccaniche di gioco e delle prestazioni dei giocatori per aiutare gli utenti a prendere decisioni informate su come migliorare.
- **Strumento di apprendimento**: Assistere gli utenti alle prime armi nell'apprendimento del gioco, fornendo una comprensione strutturata delle meccaniche di gioco.
- **Visualizzazione dei dati sulle prestazioni**: Creare un sistema di visualizzazione delle informazioni sulle partite e dei dati sulle prestazioni, simile a MOBAFIRE.

Utenti Target:

- **Giocatori principianti:** Chi è nuovo ai giochi MOBA e ha bisogno di una **comprensione strutturata** delle meccaniche di gioco per imparare il gioco.
- **Giocatori esperti**: Coloro che vogliono migliorare il proprio gioco comprendendo **le relazioni tra i diversi oggetti** e il modo in cui si combinano per formare **equipaggiamenti più potenti.**
- **Sviluppatori di giochi:** Coloro che vogliono **capire le relazioni** e le interazioni tra i diversi elementi del gioco per bilanciare il gameplay.
- **Analisti di gioco**: Coloro che vogliono **analizzare il gameplay** e le **prestazioni dei giocatori** per fornire consigli sul miglioramento del gameplay.
- **Costruzione della comunità**: Una piattaforma basata sull'ontologia potrebbe fungere da hub per la **comunità dei giochi MOBA**, facilitando la condivisione delle conoscenze e la discussione. Potrebbe ospitare guide create dai giocatori, discussioni sulla strategia, analisi dei personaggi e altro ancora, il tutto strutturato intorno al quadro dell'ontologia.

L’uso di un'ontologia comune nella comunità dei giochi MOBA potrebbe effettivamente favorire l'influenza incrociata tra sviluppatori di giochi attuali e aspiranti tali. Ecco come:

- **Vocabolario condiviso**: Un'ontologia fornisce un vocabolario condiviso e un insieme di concetti su cui tutti possono concordare. Questo linguaggio comune può facilitare una comunicazione più chiara tra sviluppatori di giochi diversi, portando potenzialmente alla condivisione di idee e strategie.
- **Impollinazione incrociata delle idee**: Con una comprensione comune delle meccaniche e delle strutture di gioco, gli sviluppatori possono imparare più facilmente dai giochi degli altri. Questo potrebbe portare all'adozione di meccaniche di successo da un gioco all'altro, promuovendo l'innovazione nel genere.
- **Standardizzazione:** Un'ontologia comune potrebbe portare a un certo livello di standardizzazione nel genere MOBA, rendendo più facile per i giocatori passare da un gioco all'altro. Questo potrebbe ampliare la base di giocatori per ogni gioco e incoraggiare la competizione tra gli sviluppatori per creare caratteristiche uniche all'interno del quadro condiviso.

3 DOMINIO

3.1.1 Descrizione

Il settore dei giochi MOBA (Multiplayer Online Battle Arena) è caratterizzato da una serie di caratteristiche comuni, ma presenta anche differenze significative tra i vari titoli. Ecco un'analisi delle somiglianze e delle differenze tra i giochi MOBA più popolari, basata su informazioni provenienti da fonti come MOBAFire, Wikipedia, Leaguepedia e altre:

Similitudini:

- **Struttura di gioco**: La maggior parte dei giochi MOBA condivide una struttura simile, con due squadre avversarie che mirano a distruggere la base dell'altro mentre difendono la propria. La mappa è tipicamente composta da diverse "corsie" che collegano le basi e da aree "giungla" con mostri neutrali.
- **Ruoli dei personaggi**: Nella maggior parte dei giochi MOBA, ogni giocatore controlla un singolo personaggio, o "eroe", con abilità uniche. Questi personaggi spesso ricoprono ruoli specifici, come il damage dealer, il tank o il supporto.
- **Progressione e livellamento**: I personaggi iniziano ogni partita a un livello basso e guadagnano esperienza e oro sconfiggendo personaggi e mostri nemici. In questo modo possono salire di livello, migliorare le proprie abilità e acquistare oggetti per migliorare le proprie prestazioni.

Differenze:

- **Abilità e meccaniche dei personaggi**: Mentre i ruoli di base possono essere simili, le abilità e le meccaniche specifiche dei personaggi possono variare notevolmente da un gioco all'altro. Ad esempio, Dota 2 è noto per avere personaggi con abilità potenti e in grado di cambiare le partite, mentre League of Legends pone maggiore enfasi sui colpi di abilità e sulla mobilità.
- **Sistemi di oggetti:** Anche il modo in cui gli oggetti vengono utilizzati e acquisiti può variare da un gioco all'altro. In Dota 2, ad esempio, i personaggi possono usare un corriere per consegnare loro gli oggetti dal negozio sul campo di battaglia, mentre in League of Legends devono tornare alla loro base per acquistare gli oggetti.

- **Design delle mappe:** Mentre la struttura di base di corsie e giungle è comune, il layout specifico e le caratteristiche della mappa possono variare. Ad esempio, Heroes of the Storm presenta più mappe con obiettivi unici che possono aiutare una squadra a ottenere un vantaggio.

- **Ritmo e durata delle partite:** Anche il ritmo e la durata tipica delle partite possono variare da un gioco all'altro. Alcuni giochi, come **Pokemon Unite,** Heroes of the Storm, sono progettati per avere partite più brevi e veloci, mentre altri, come Dota 2 e League Of Legends (**DESKTOP**), hanno spesso partite più lunghe con un ritmo più lento.

Date le differenze identificate in precedenza, come le abilità e le meccaniche dei personaggi, i sistemi di oggetti, il design delle mappe, il ritmo e la durata del gioco, l'ontologia deve essere abbastanza flessibile da accogliere queste variazioni. Tuttavia, deve anche mantenere un certo livello di coerenza per fornire un quadro comune che possa essere applicato a giochi diversi.

Per esempio, anche se le abilità e le meccaniche specifiche dei personaggi possono variare notevolmente da un gioco all'altro, l'ontologia può comunque fornire una struttura comune per **la rappresentazione dei personaggi**, come la definizione di **attributi** per la salute**, i** danni**,** il **ruolo** e le **abilità**. Allo stesso modo, anche se i sistemi di oggetti possono essere diversi, l'ontologia può comunque modellare il concetto di **oggetto** e le relazioni tra di essi, come "**costruito da**" e "**usato per costruire**".

In questo modo, l'ontologia può catturare gli aspetti unici dei diversi giochi, pur fornendo un linguaggio e una struttura comuni per comprendere e discutere il genere MOBA nel suo complesso. Questo equilibrio tra espressività e fondamento è la chiave dell'utilità e dell'applicabilità dell'ontologia.

3.1.2 Fonti e riferimenti bibliografici

Queste somiglianze e differenze si basano su informazioni provenienti da varie fonti, tra cui wiki specifici del gioco come Leaguepedia e Dota 2 Wiki, siti di guide strategiche come MOBAFire e fonti di informazioni generali come Wikipedia.

4 DOCUMENTAZIONE DOMINIO

4.1.1 Ispirazioni e risorse simili (standard esistenti)

Per la definizione della MOBA ontology, ho esplorato diverse fonti di ispirazione, inclusi lavori esistenti e repository su **GitHub**, al fine di comprendere meglio i concetti chiave e le relazioni all'interno dei MOBA games. Tuttavia, è importante sottolineare che la MOBA ontology che ho sviluppato rappresenta un'interpretazione originale e non è stata direttamente integrata o basata su un repository specifico o una risorsa esistente.

4.1.2 Dota Ontology

Uno dei repository che ho esaminato è il **Dota Ontology** presente su GitHub (https://github.com/AbstractMachinesLab/dota-ontology) . Questa risorsa ha fornito un punto di partenza prezioso per comprendere gli elementi fondamentali di un MOBA game come Dota 2 e le possibili relazioni tra di essi. Anche se non ho integrato direttamente questa ontologia nel mio lavoro, è stata utile per acquisire una visione generale delle componenti tipiche di un MOBA game e delle possibili interazioni tra personaggi, abilità e oggetti.

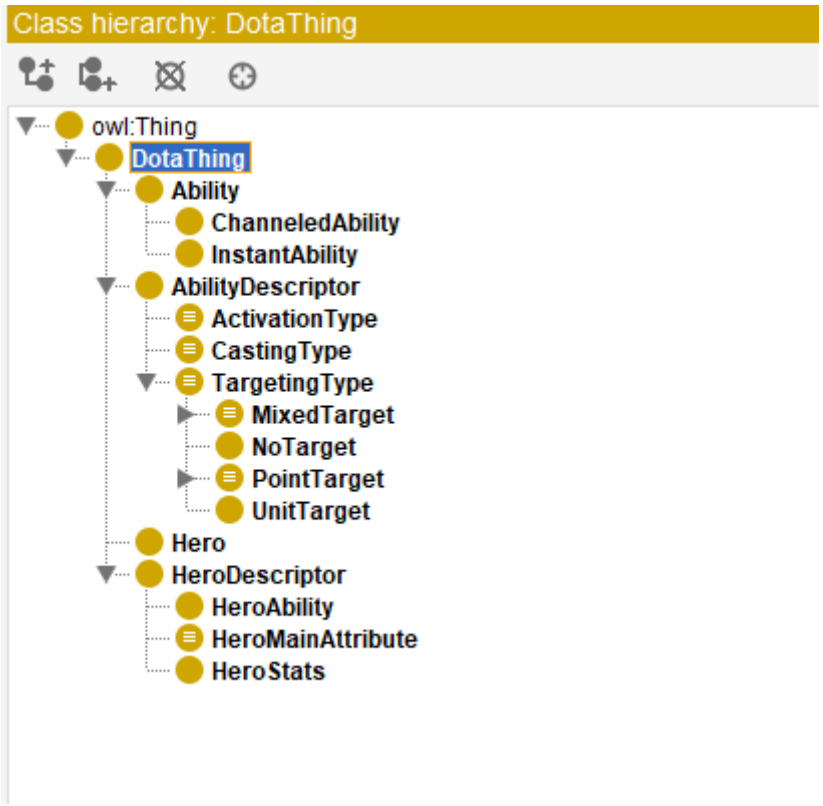


Figura 1 Estrattodella Dota Ontology

La Dota Ontology, come presupposto e finalità, è stata progettata per supportare il processo di "Drafting" nel gioco Dota 2. Il Drafting è una fase strategica in cui le squadre selezionano e scelgono i personaggi (chiamati eroi) per comporre la loro squadra prima dell'inizio della partita. L'obiettivo della Dota Ontology è quello di fornire un'ontologia limitata che rappresenti gli eroi, le loro abilità e le interazioni tra di essi durante il processo di Drafting.

Sebbene la Dota Ontology sia utile per la specifica finalità del Drafting nel gioco Dota 2, può essere limitata quando si cerca di definire un'ontologia più ampia per i MOBA games in generale. Poiché la Dota Ontology è focalizzata esclusivamente su Dota 2, le sue definizioni e relazioni potrebbero non essere facilmente generalizzabili ad altri titoli di MOBA games. Ciò significa che potrebbero mancare aspetti importanti dei MOBA games che sono specifici di altri titoli.

Per definire un'ontologia più generale per i MOBA games, è necessario considerare un insieme più ampio di giochi e le caratteristiche comuni che si ritrovano tra di essi. Questo permette di catturare in modo più completo le relazioni, le meccaniche e le dinamiche di gioco che si verificano nei MOBA games nel loro insieme. Pertanto, mentre la Dota Ontology può essere utile per la sua finalità specifica, è necessario adottare un approccio più ampio e generale per definire un'ontologia dei MOBA games che sia rappresentativa di tutto il genere.

4.1.3 League of Legends Ontology

Un altro repository che ho esplorato è il SPARQL Query Generator for League of Legends Ontology (https://github.com/sirmarcis/SPARQL-Query-Generator-for-League-of-Legends-Ontology/tree/master). Questo repository ha offerto uno strumento per generare query SPARQL sull'ontologia di League of Legends. Anche se non ho utilizzato direttamente questo generatore di query, mi ha permesso di approfondire l'approccio semantico utilizzato nell'ontologia di un MOBA game specifico come League of Legends.

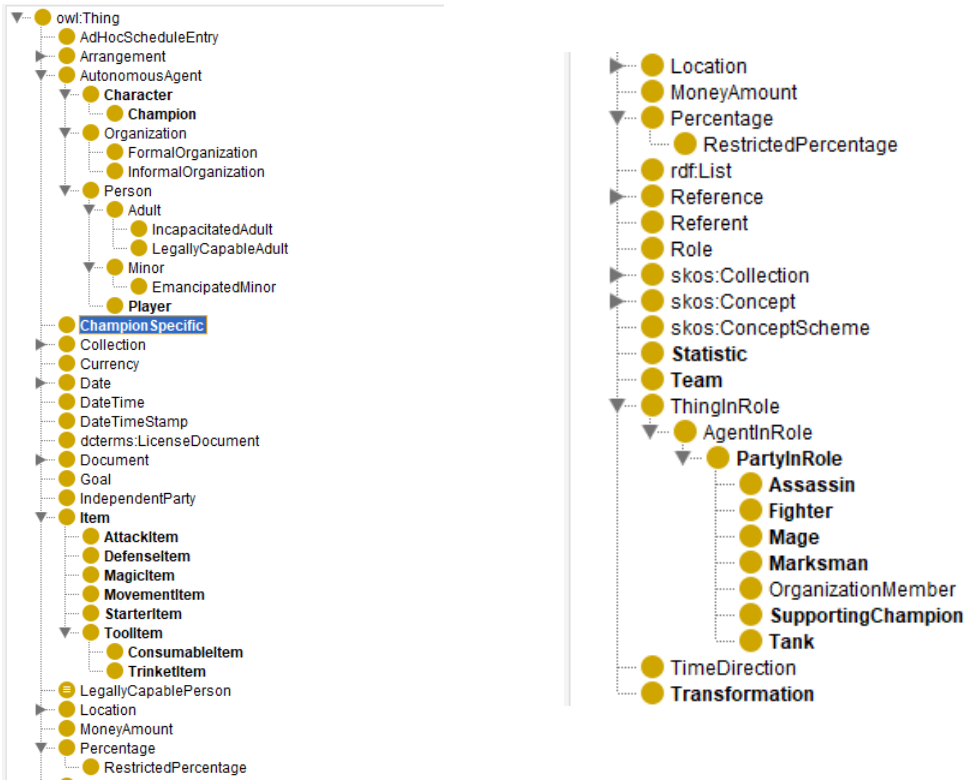


Figura 2 League of legends Ontology nel dettaglio

Nonostante la versione 5.1.3 della League of Legends Ontology presentasse una maggiore precisione nell'organizzazione delle classi all'interno della T-BOX, è importante sottolineare che non erano state effettuate integrazioni o allineamenti con SKOS o altre risorse, né erano presenti istanze rappresentative nel progetto stesso. È possibile che si tratti di un refuso da parte dell'autore del repository o che il progetto non sia stato completamente depositato.

Nonostante queste limitazioni, la struttura e la gerarchia dei concetti **come l'Agente** (autonomo o meno) e **il Ruolo** possono comunque essere estremamente utili per la definizione del dominio dei MOBA games. La presenza di una struttura organizzata e nidificata dei concetti offre una comprensione più approfondita dei ruoli e delle dinamiche di gioco presenti nei MOBA games. Ad esempio, l'uso di una gerarchia dei Ruoli consente di distinguere tra i diversi tipi di personaggi presenti nel gioco, come i personaggi di supporto, gli assassini o i tank. Questa struttura offre una base solida per l'elaborazione dell'ontologia dei MOBA games, consentendo una rappresentazione più accurata delle relazioni tra i concetti e fornendo una visione organizzata del dominio.

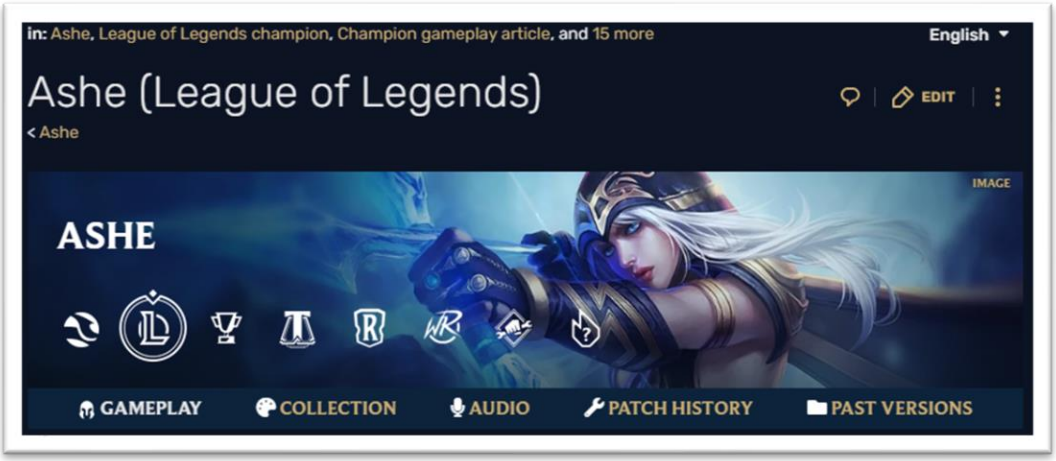
Oltre a queste risorse specifiche, ho condotto una ricerca su documenti accademici, consultato articoli scientifici e documentazioni ufficiali sui MOBA games. Questo approccio mi ha consentito di acquisire una

conoscenza approfondita dei concetti chiave dei MOBA games, come la struttura del gioco, i **ruoli** dei **personaggi**, la **progressione dei livelli** e le meccaniche di gioco.

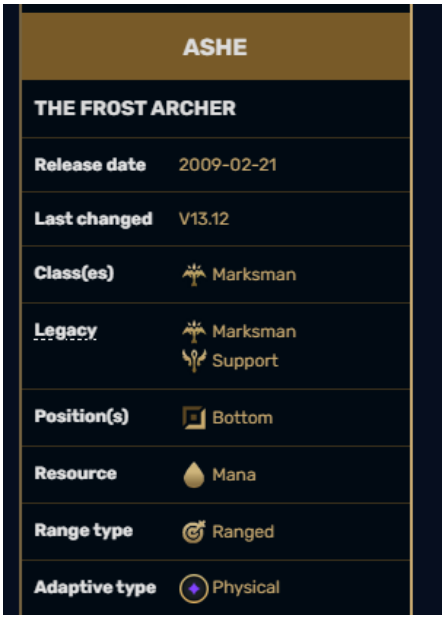
Mentre ho tratto ispirazione da diverse fonti, ho scelto di sviluppare un'ontologia che fosse più generale e rappresentativa del dominio dei MOBA games nel suo insieme. Ho cercato di evitare l'integrazione di ontologie esistenti specifiche di un singolo gioco MOBA perché avrebbero potuto risultare incomplete o troppo specifiche per rappresentare l'intero spettro dei MOBA games.

Successivamente, ho esteso la mia ontologia MOBA per includere tre istanze specifiche dei diversi giochi MOBA: League of Legends (LoL), Dota 2 e Pokémon Unite. Ho riconosciuto che ogni gioco MOBA ha le proprie caratteristiche uniche, meccaniche di gioco e personaggi, e ho voluto creare istanze specifiche dell'ontologia per catturare queste differenze.

4.1.4 Esempio del dominio

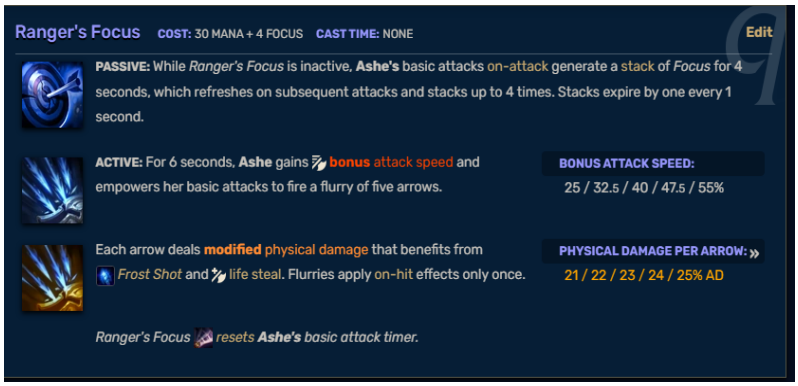


Ashe è un personaggio chiave nel gioco MOBA League of Legends (LoL). Prendendo in considerazione le caratteristiche presenti sulla pagina di Ashe su **League of Legends Fandom**, ecco una descrizione completa dei **punti chiave** del personaggio:

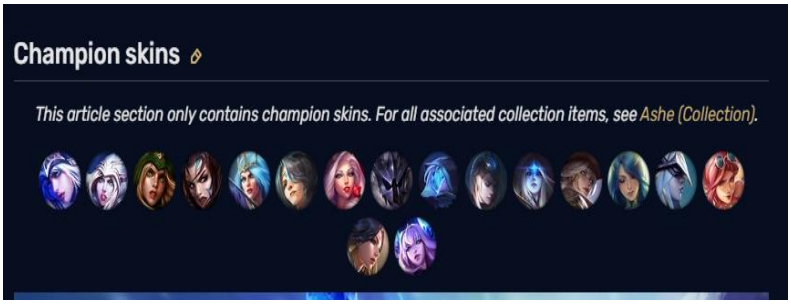


- **Ruolo**: Ashe è un tiratore (ADC) di ruolo, specializzato nel fornire danni a distanza ai nemici attraverso i suoi attacchi base e abilità.
- **Posizione di gioco e Stile di gioco**: elemento caratteristico del gameplay (corsia in cui il gioco consiglia di giocare il personaggio)
- **Risorsa utilizzata**: ogni Champion per poter utilizzare le proprie abilità consuma una qualche risorsa, nel caso di Ashe è il **Mana**
- **Range:** una caratteristica legata dalla combinazione di abilità e forme di attacco del campione. **Ranged** generalmente indica che il campione effettua degli attacchi **"**base" (senza utilizzo di abilità) che hanno una portata (in termini di unità di gioco) ampia mentre il suo opposto è "**Melee**".
- **Build e oggetti raccomandati**
- **Strategie di gioco**

- **Statistiche:** Le statistiche di un personaggio come Ashe in League of Legends sono un aspetto importante per valutarne la forza e l'efficacia in combattimento. Nel box riportate alcune delle statistiche chiave di Ashe.
- E' fondamentale indicare che queste caratteristiche possono variare nel corso della partita in base agli oggetti acquisiti, livelli ottenuti dal player.



- **Abilità uniche**: in tutti i giochi MOBA ogni campione è caratterizzato da ABILITA uniche che possono essere PASSIVE o ATTIVE (attivate su richiesta esplicita dal giocatore). La descrizione è esaustiva nel descrivere le interazioni con **items** ma possono anche esserci casi di **interazione** con specifici altri **campioni.** Probabilmente sarebbe interessante modellare il concetto di **interazione** in questo frangente.



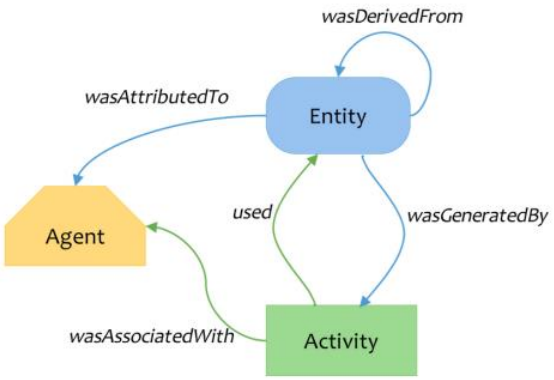
- **LORE:** è un elemento importante che contribuisce a dare profondità al personaggio e ad arricchire l'esperienza di gioco. La storia di Ashe è ambientata nel vasto mondo immaginario di Runeterra, dove coesistono diverse regioni, popoli e poteri magici.
- **Skin:** sono personalizzazioni dell'aspetto del personaggio che possono essere assunte durante la partita e spesso fanno riferimento al LORE.

4.1.5 Allineamenti

4.1.5.1 PROV

Nel mio processo di modellazione delle azioni nei MOBA games, ho trovato ispirazione nel framework Provenance e, in particolare, nel PROV Model. Utilizzando questo framework, ho definito sottoclassi più specifiche e sottoproprietà che collegano le nuove classi create nell'ontologia.

Ho utilizzato il framework Provenance (PROV) per modellare diverse attività di gioco all'interno dei MOBA games, come l'eliminazione di minion, l'eliminazione di altri personaggi, l'aggiunta di un giocatore a un team (che può essere considerata una sorta di astrazione del matchmaking) e l'aggiunta di un giocatore a una partita.



Nella modellazione degli agenti, ho identificato il giocatore umano o **PlayerAgent** come l'agente principale nel contesto dei MOBA games. Tuttavia, ho considerato la possibilità di creare una sottoclasse "Organization" per rappresentare il **Team** a cui il giocatore appartiene e una sottoclasse "Software Controlled" per rappresentare il giocatore non umano (l'Intelligenza Artificiale del gioco che controlla i personaggi non giocanti).

Ho utilizzato i concetti forniti dal framework PROV per gestire l'interazione tra le attività e gli agenti. Ad esempio, ho modellato l'attività di formazione del team come una generazione del team da parte di un'attività "Team formation". Inoltre, ho considerato l'assegnazione di ricompense come un evento che si verifica al termine di un'attività di eliminazione, consentendo di tracciare e registrare i risultati ottenuti dai giocatori.

L'adozione dei termini PROV nella modellazione delle attività e degli agenti mi ha permesso di creare un quadro più strutturato e coerente delle dinamiche di gioco nei MOBA games. Questo approccio consente di tracciare l'origine delle attività, le interazioni tra gli agenti e il flusso delle informazioni all'interno del gioco, fornendo una maggiore comprensione del contesto e delle relazioni all'interno dell'ontologia dei MOBA games.

4.1.5.2 Ontology patterns

L'integrazione delle ontologie fondazionali legate ai pattern di modellazione, come le ontologie delle liste, dei bag e delle sequenze, può arricchire l'ontologia MOBA fornendo una struttura e un supporto per la rappresentazione di concetti legati all'organizzazione e alla gestione dei dati.

Le ontologie delle liste, dei bag e delle sequenze sono comunemente utilizzate per modellare collezioni di elementi in diversi contesti. Nell'ambito dei MOBA games, queste ontologie possono essere utilizzate per rappresentare collezioni di oggetti, abilità, personaggi o altre entità pertinenti al gioco.

Nel caso specifico, le ho utilizzate principalmente per descrivere:

- Sequenze di oggetti (in riferimento all'inventario)
- Sequenze di potenziamento abilità
- Lista di oggetti necessari alla creazione di un altro

4.1.6 Skos

Con SKOS avrei potuto utilizzare le relazioni come broader, narrower ed equivalence per definire gerarchie più complesse tra le classi della mia ontologia dei MOBA games.

Con la relazione broader, avrei potuto stabilire una gerarchia in cui una classe più generale comprende classi più specifiche. Ad esempio, avrei potuto definire una classe generale "Personaggio" e sottoclassi più specifiche come "Tank", "Support" o "Assassin".

La relazione narrower, invece, avrebbe permesso di definire classi più specifiche che sono sottoclassi di una classe più generale. Ad esempio, potrei definire una classe generale "Abilità" e sottoclassi più specifiche come "Abilità di attacco", "Abilità difensiva" o "Abilità di cura".

Infine, la relazione equivalence avrebbe consentito di stabilire l'equivalenza tra classi che rappresentano lo stesso concetto, ma con terminologie diverse o provenienti da ontologie diverse. Questo avrebbe facilitato l'allineamento e l'integrazione delle informazioni tra diverse ontologie dei MOBA games.

Ho maturato la volontà di introdurre Skos in un momento un po troppo avanzato del progetto e alla fine ho scelto di non utilizzare SKOS nell'ontologia principalmente a causa di problemi di lentezza riscontrati con Protégé, che dopo un certo punto, mi ha causato non pochi problemi di performance e stabilità. Probabilmente avrei dovuto partire fin da subito con la l'idea di modellare tramite Skos queste relazioni.

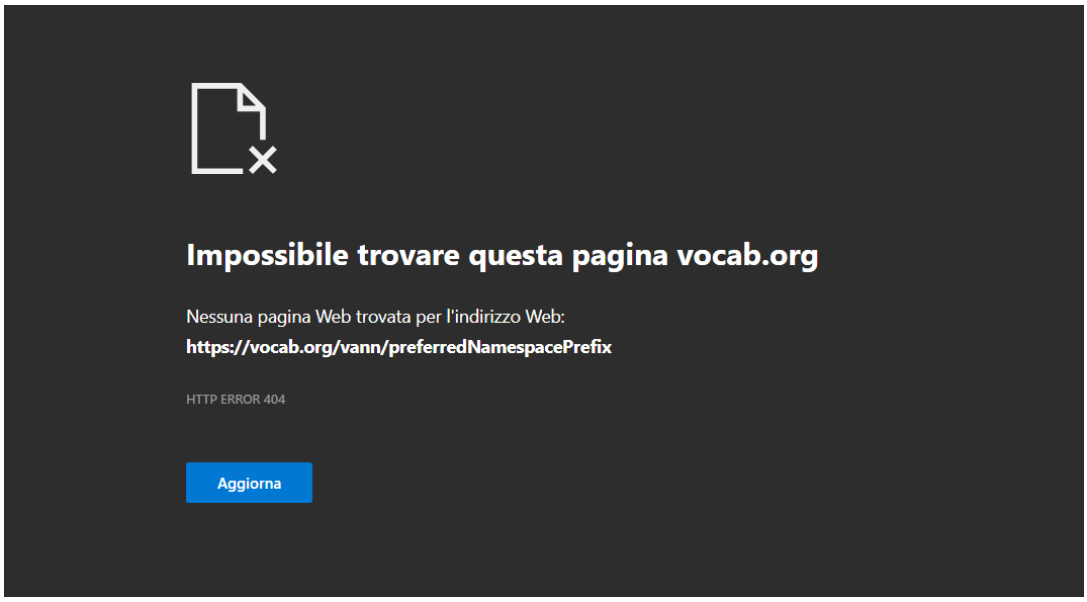
4.1.7 Pattern allineamento

Per quanto riguarda l'allineamento con queste risorse esterne, ho utilizzato principalmente la relazione **rdfs:subClassOf** e **owl:equivalentClass** nel seguente modo:

- **owl:equivalentClass**: Ho adottato la relazione **owl:equivalentClass** anche per nominare specifiche risorse nel contesto della mia ontologia. Ho stabilito una corrispondenza diretta tra le classi della mia ontologia MOBA e le risorse esterne, utilizzando lo stesso nome o un nome equivalente per rappresentare il concetto specifico nella mia ontologia.**rdfs:subClassOf**:
- **rdfs:subClassOf** quando ho introdotto classi più specifiche rispetto a quelle presenti nelle risorse esterne, ma volevo ancora beneficiare della struttura delle relazioni della risorsa esterna. Questo collegamento indica che le classi introdotte sono sottoclassi delle classi esterne, estendendo così la gerarchia delle classi.

4.1.8 Checklist per l'ontologia

Non sono riuscito a trovare un modo per importare vann ed applicare le annotazioni vann:**preferredNamespacePrefix** e vann:p**referredNamespaceUri.**
Navigando su https://vocab.org/vann/ ho anche verificato non risulta più funzionare il puntamento a https://vocab.org/vann/preferredNamespacePrefix e https://vocab.org/vann/preferredNamespaceUri



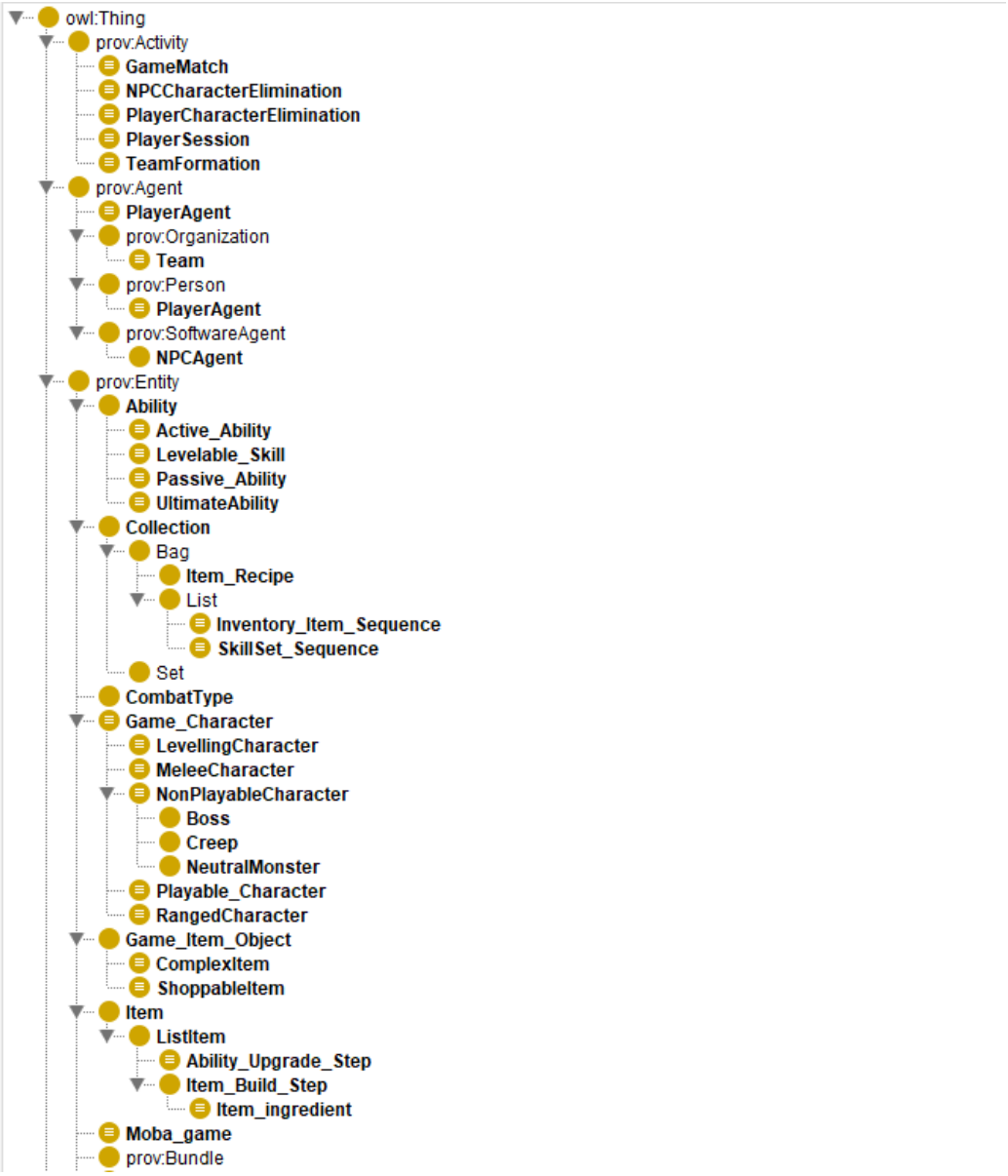
5 LODE

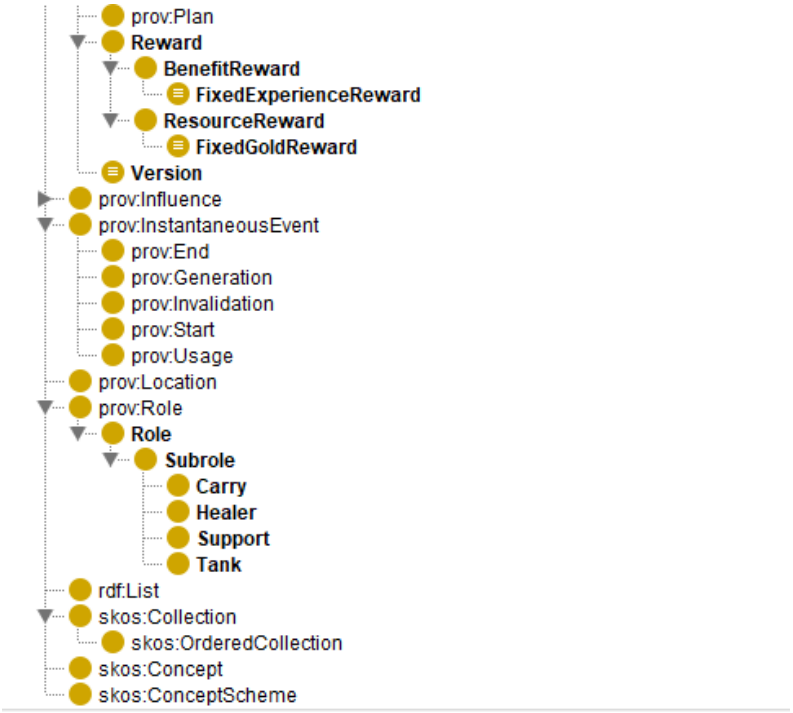
La documentazione è stata estratta e si trova nella cartella LODE.

6 VISUALIZZAZIONE

Nei successivi paragrafi verranno illustrate le visualizzazioni richieste.

TASSONOMIA DELLE CLASSI



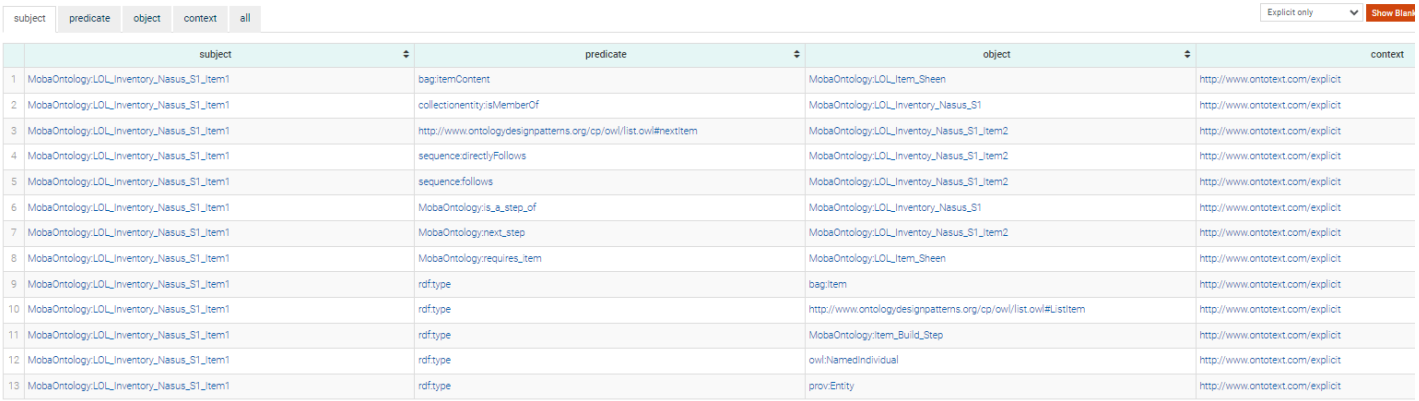


6.1 PRINCIPIALI TEMPLATE

6.1.1 Ontology Design Pattern List e

Nell'implementazione dell'Ontology Design Pattern (ODP) LIST, ho seguito l'approccio standardizzato per la rappresentazione di una sequenza di elementi, mantenendo un ordine specifico tra questi.

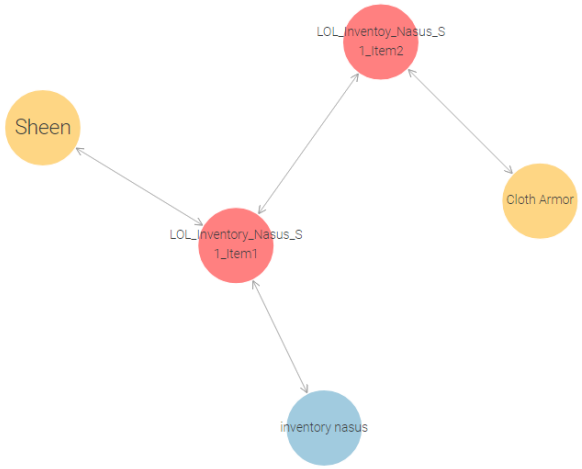
Ho utilizzato questo pattern per rappresentare due tipi di liste: una per gli oggetti dell'inventario dei personaggi Ashe e Nasus, e un'altra per le sequenze di aggiornamenti delle abilità dei medesimi personaggi.


subject predicate object context all

Explicit only Show Blank

	subject	predicate	object	context
1	MobaOntology:LOL_Inventory_Nasus_S1_Item1	bag:itemContent	MobaOntology:LOL_Item_Sheen	http://www.ontotext.com/explicit
2	MobaOntology:LOL_Inventory_Nasus_S1_Item1	collectionentity:isMemberOf	MobaOntology:LOL_Inventory_Nasus_S1	http://www.ontotext.com/explicit
3	MobaOntology:LOL_Inventory_Nasus_S1_Item1	http://www.ontologydesignpatterns.org/cp/owl/list.owl#nextitem	MobaOntology:LOL_Inventoy_Nasus_S1_Item2	http://www.ontotext.com/explicit
4	MobaOntology:LOL_Inventory_Nasus_S1_Item1	sequence:directlyFollows	MobaOntology:LOL_Inventoy_Nasus_S1_Item2	http://www.ontotext.com/explicit
5	MobaOntology:LOL_Inventory_Nasus_S1_Item1	sequence:follows	MobaOntology:LOL_Inventoy_Nasus_S1_Item2	http://www.ontotext.com/explicit
6	MobaOntology:LOL_Inventory_Nasus_S1_Item1	MobaOntology:is_a_step_of	MobaOntology:LOL_Inventory_Nasus_S1	http://www.ontotext.com/explicit
7	MobaOntology:LOL_Inventory_Nasus_S1_Item1	MobaOntology:next_step	MobaOntology:LOL_Inventoy_Nasus_S1_Item2	http://www.ontotext.com/explicit
8	MobaOntology:LOL_Inventory_Nasus_S1_Item1	MobaOntology:requires_item	MobaOntology:LOL_Item_Sheen	http://www.ontotext.com/explicit
9	MobaOntology:LOL_Inventory_Nasus_S1_Item1	rdf:type	bag:Item	http://www.ontotext.com/explicit
10	MobaOntology:LOL_Inventory_Nasus_S1_Item1	rdf:type	http://www.ontologydesignpatterns.org/cp/owl/list.owl#ListItem	http://www.ontotext.com/explicit
11	MobaOntology:LOL_Inventory_Nasus_S1_Item1	rdf:type	MobaOntology:Item_Build_Step	http://www.ontotext.com/explicit
12	MobaOntology:LOL_Inventory_Nasus_S1_Item1	rdf:type	owl:NamedIndividual	http://www.ontotext.com/explicit
13	MobaOntology:LOL_Inventory_Nasus_S1_Item1	rdf:type	prov:Entity	http://www.ontotext.com/explicit

Per rappresentare le liste degli inventari, ho creato un individuo per ogni oggetto nell'inventario, associandolo all'elemento precedente tramite la proprietà **is_a_step_of** e, se presente, all'elemento successivo tramite la proprietà **next_step**. Si tratta di relazioni costruite a partite da quelle offerte dalla libreria ma ristrette sul dominio della mia ontologia.



Nell'implementazione delle sequenze di **aggiornamenti delle abilità**, ho creato un individuo per ogni step di aggiornamento, associandolo alla sequenza di aggiornamenti tramite la proprietà **isMemberOf** e, sepresente, all'elemento successivo tramite la proprietà **nextItem**.

Ad esempio, l'elemento "LOL_Inventory_Ashe_S1_Item1" è associato all'inventario "LOL_Inventory_Ashe_S1" e richiede l'oggetto "LOL_Item_Boots".

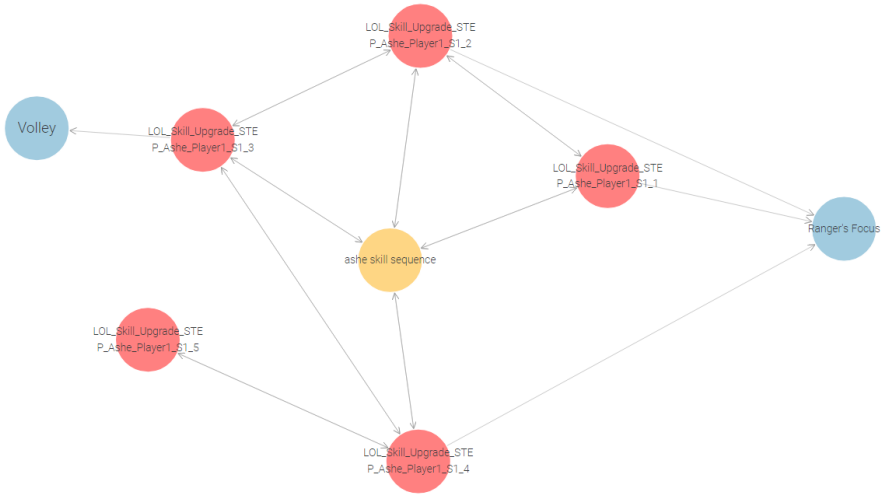


Figura 3 Nell immagine i nodi blu sono le ABILITA, i nodi rossi sono i List Item e il nodo giallo è la lista delle skills

subject | predicate | object | context | all

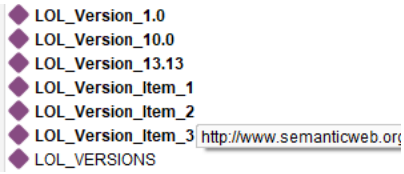
Explicit only | Show Blank Nodes | Download as | Visual graph

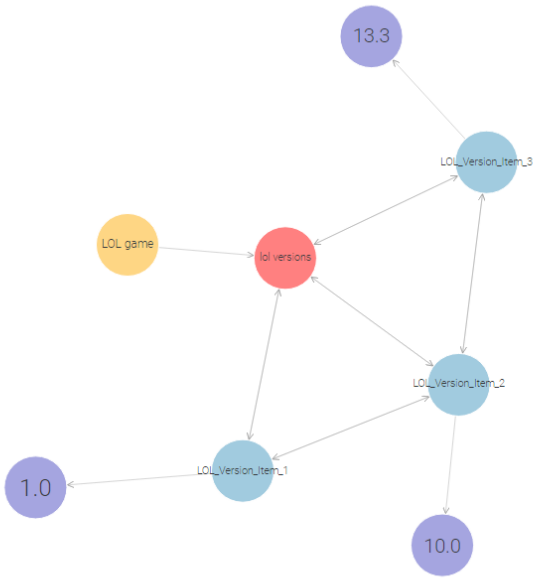
	subject	predicate	object	context
1	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	collectionentity:isMemberOf	MobaOntology:LOL_Skill_Upgrade_Sequence_Nasus_Player1_S1	http://www.ontotext.com/explicit
2	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	http://www.ontologydesignpatterns.org/cp/owl/list.owl#nextItem	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_2	http://www.ontotext.com/explicit
3	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	sequence:directlyFollows	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_2	http://www.ontotext.com/explicit
4	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	sequence:follows	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_2	http://www.ontotext.com/explicit
5	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	sequence:follows	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_3	http://www.ontotext.com/explicit
6	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	sequence:follows	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_4	http://www.ontotext.com/explicit
7	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	sequence:follows	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_5	http://www.ontotext.com/explicit
8	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	sequence:follows	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_6	http://www.ontotext.com/explicit
9	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	sequence:follows	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_7	http://www.ontotext.com/explicit
10	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	sequence:follows	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_8	http://www.ontotext.com/explicit
11	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	sequence:follows	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_9	http://www.ontotext.com/explicit

Lo stesso approccio l'ho adottato per quanto riguarda il versioning del Moba Game. Ho effettivamente definito una classe Versione con queste data property:

- has-version
- release_date

Per cui anche in questo caso, creo gli oggetti List Item, li rendo membri della Collezione **LOL_Versions** e successivamente collego ciascun Wrapper alla specifica istanza di versione e, al wrapper successivo.



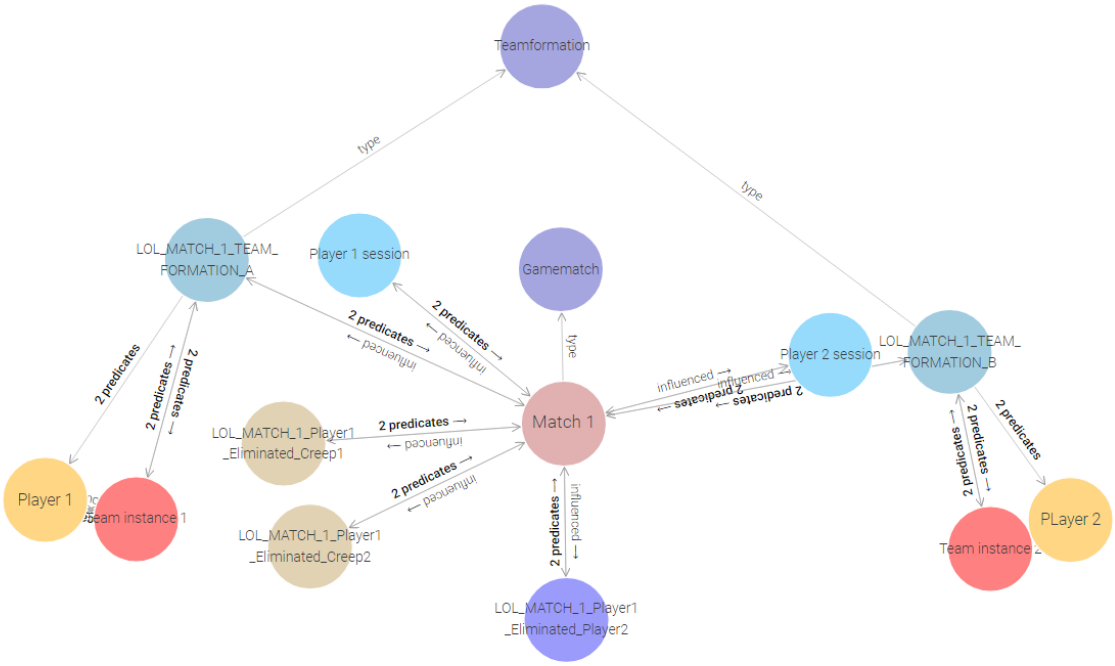


subject | predicate | object | context | all

Explicit only | Show Blank Nodes | Download as | Visual grap

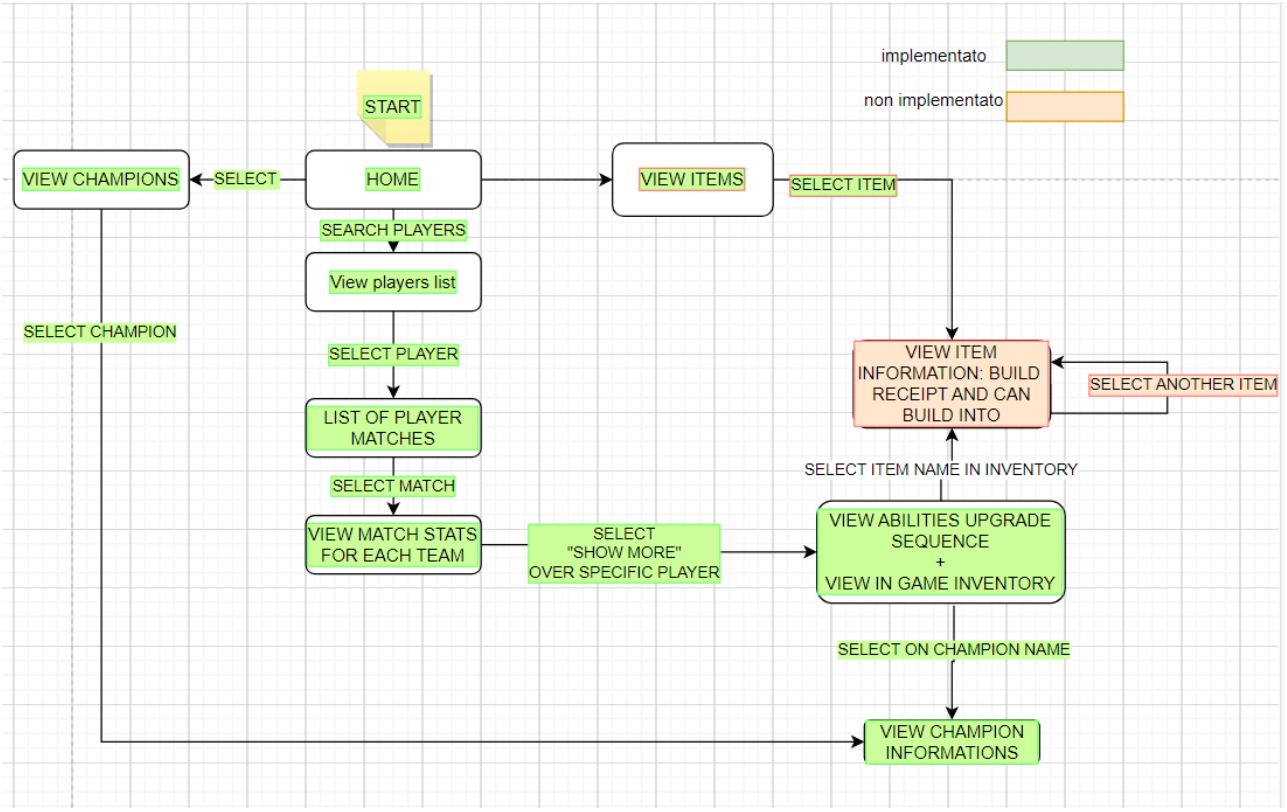
	subject	predicate	object	context
1	MobaOntology:LOL_VERSIONS	collectionentity:hasMember	MobaOntology:LOL_Version_Item_1	http://www.ontotext.com/explicit
2	MobaOntology:LOL_VERSIONS	collectionentity:hasMember	MobaOntology:LOL_Version_Item_2	http://www.ontotext.com/explicit
3	MobaOntology:LOL_VERSIONS	collectionentity:hasMember	MobaOntology:LOL_Version_Item_3	http://www.ontotext.com/explicit
4	MobaOntology:LOL_VERSIONS	rdf:type	bag:Bag	http://www.ontotext.com/explicit
5	MobaOntology:LOL_VERSIONS	rdf:type	collectionentity:Collection	http://www.ontotext.com/explicit
6	MobaOntology:LOL_VERSIONS	rdf:type	http://www.ontologydesignpatterns.org/cp/owl/list.owl#List	http://www.ontotext.com/explicit
7	MobaOntology:LOL_VERSIONS	rdf:type	owl:NamedIndividual	http://www.ontotext.com/explicit
8	MobaOntology:LOL_VERSIONS	rdf:type	prov:Entity	http://www.ontotext.com/explicit
9	MobaOntology:LOL_VERSIONS	rdfs:comment	"all lol verions"	http://www.ontotext.com/explicit
10	MobaOntology:LOL_VERSIONS	rdfs:label	"lol versions"@en	http://www.ontotext.com/explicit

6.1.2 Visualizzazione del MATCH

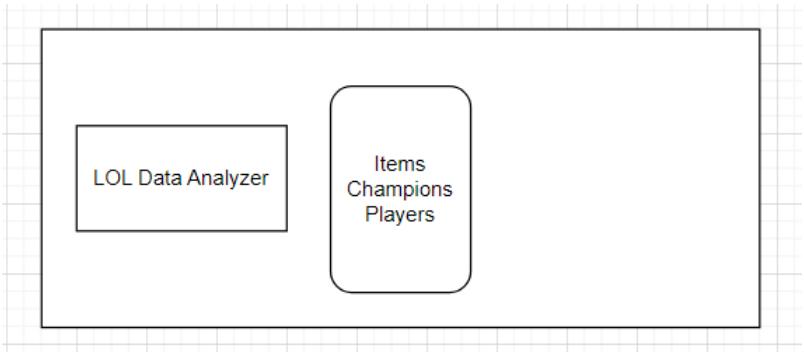


7 QUERIES SPARQL

7.1 INTERACTION



Home: L'utente viene accolto da una schermata iniziale che gli chiede di selezionare “**visualizza emporio oggetti**”, “**visualizza campioni**”, “**visualizza statistiche**”.



visualizza emporio oggetti

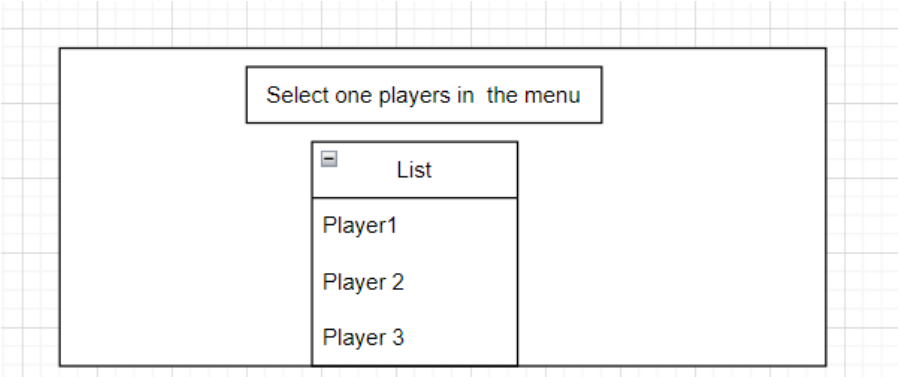
1. L'utente viene accolto da una schermata con una lista di oggetti, gli viene chiesto di selezionare un Oggetto specifico per il quale vuole visualizzare le informazioni.
2. **Visualizzazione dettaglio Oggetto:** L'utente visualizza un insieme di dettagli sull’oggetto e sui componenti di ciascun oggetto. (può selezionarli per aprire un dettaglio ulteriore)

visualizza campioni

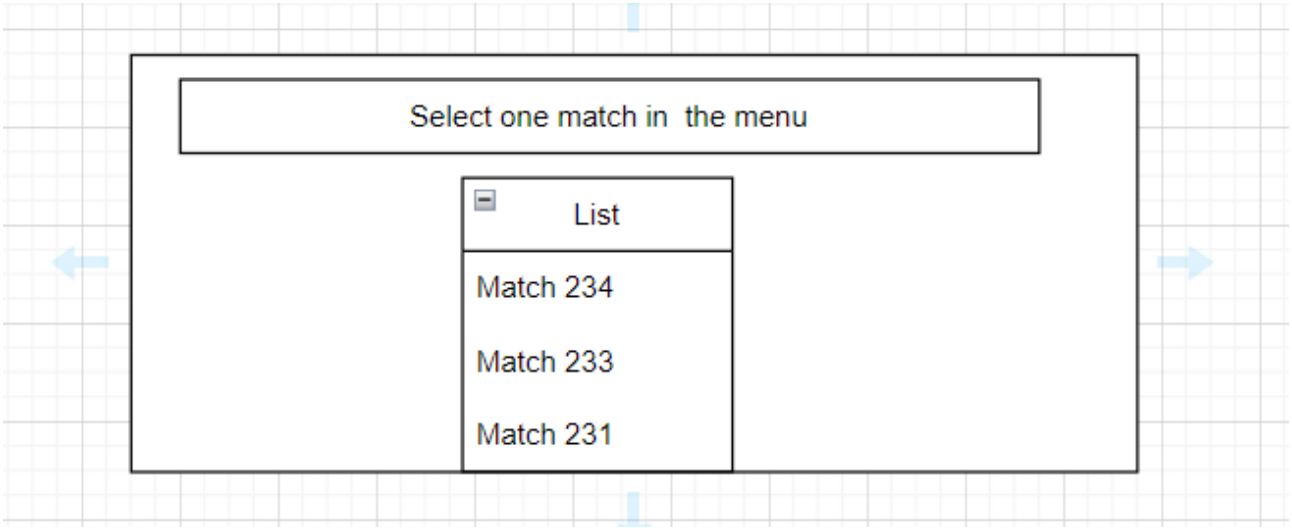
1. L'utente viene accolto da una schermata con una lista di campioni, gli viene chiesto di selezionare un Campione specifico per il quale vuole visualizzare le informazioni.
2. **Visualizzazione Campione:** L'utente visualizza un insieme di dettagli sull’oggetto e sui componenti di ciascun oggetto. (può selezionarli per aprire un dettaglio ulteriore)

visualizza statistiche

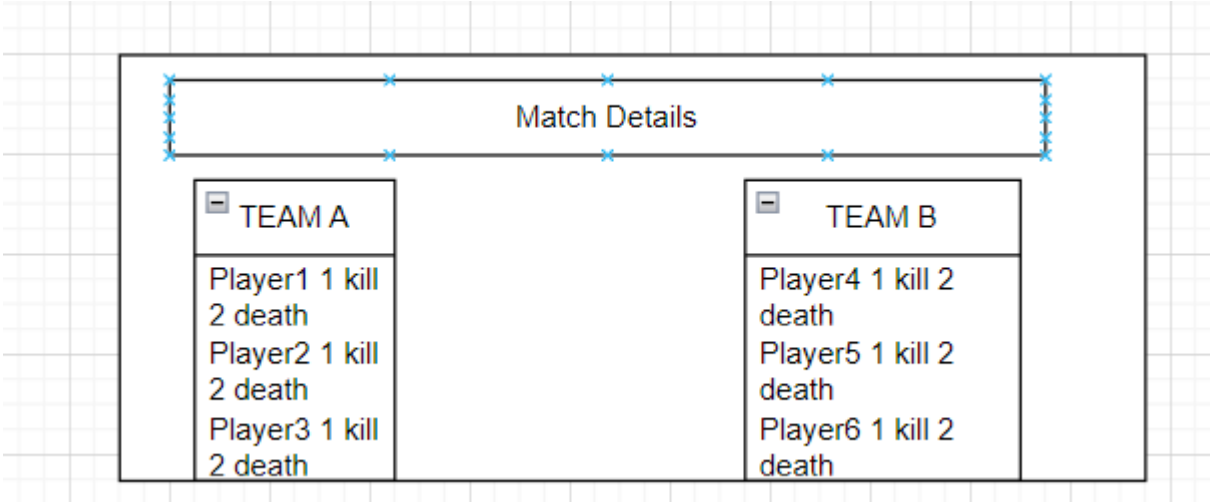
1. L'utente viene accolto da una schermata con una lista di utenti, gli viene chiesto di selezionare un Player Agent specifico per il quale vuole visualizzare le statistiche.



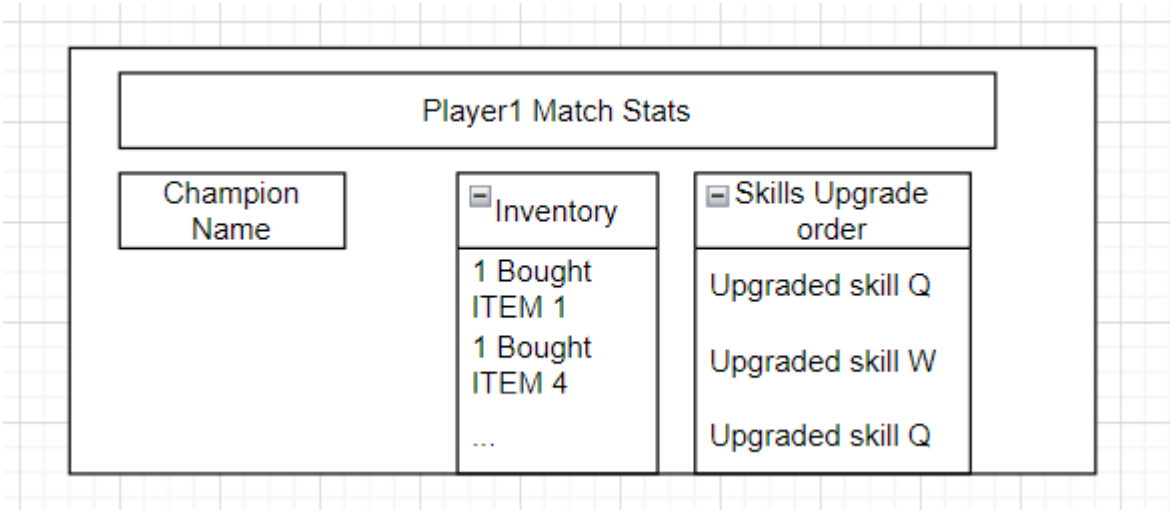
2. **Selezione del Match:** Dopo aver selezionato un giocatore, all'utente viene chiesto di selezionare un match specifico da analizzare.



3. **Visualizzazione delle Statistiche del Match:** Una volta selezionato il match, vengono visualizzate le statistiche del match per il giocatore selezionato. Questo include il numero di eliminazioni NPC, eliminazioni di personaggi giocatori, numero di volte in cui il giocatore è stato eliminato e il totale dell'oro guadagnato.



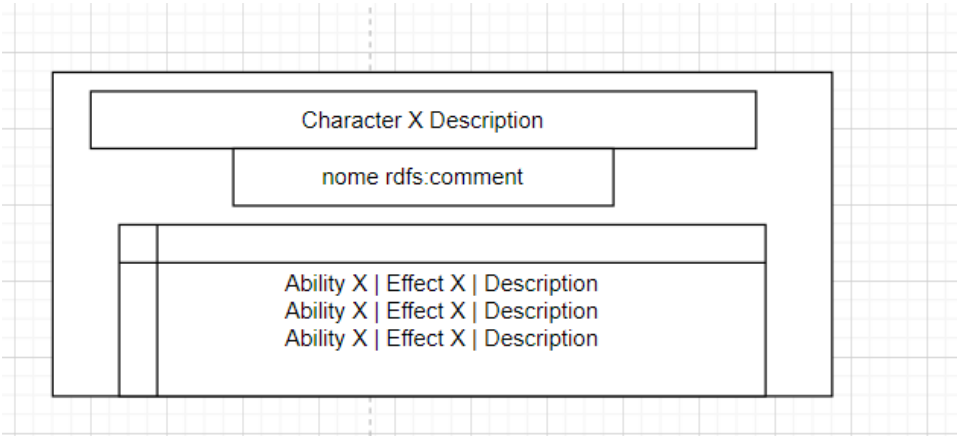
4. **Dettaglio sul Personaggio utilizzato**: L'utente ha poi la possibilità di cliccare sul nome del personaggio di gioco per visualizzare informazioni più dettagliate sulle abilità del personaggio. **(**Vedi **Visualizzazione Campione)**



5. **Visualizzazione della Build di Oggetti:** L'utente può inoltre selezionare di visualizzare la "build" di oggetti utilizzata dal giocatore durante il match.

1. Ogni oggetto della build è navigabile nel dettaglio.
2. **Vedi - Visualizzazione dettaglio Oggetto**

6. **Visualizzazione delle Mosse Potenziate**: Infine, l'utente può scegliere di visualizzare quali mosse sono state potenziate dal giocatore durante il match.



7.2 QUERIES

In seguito, le query usate nell'implementazione dell'interazione con l'utente precedentemente descritta. Tutte le query definiscono i seguenti prefissi:

```
PREFIX : <http://www.semanticweb.org/user/ontologies/2023/6/MobaOntology#>
PREFIX LIST: <http://www.ontologydesignpatterns.org/cp/owl/list.owl#>
PREFIX col:
<http://www.ontologydesignpatterns.org/cp/owl/collectionentity.owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX BAG: <http://www.ontologydesignpatterns.org/cp/owl/bag.owl#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
```

7.2.1.1 Recupero dello scoreboard in una partita

```
SELECT ?player ?team  ?playerEliminationCount   ?playerDeathCount ?
totalGoldEarned ?npcEliminationCount{
    SELECT ?player ?team
    (COUNT(DISTINCT ?npcElimination) AS ?npcEliminationCount)
    (COUNT(DISTINCT ?playerElimination) AS ?playerEliminationCount)
    (COUNT(DISTINCT ?playerDeaths) AS ?playerDeathCount)
    (SUM(?npcGold) + SUM(?playerGold) AS ?totalGoldEarned)
    WHERE {
        # Fetching all player characters
        ?player a :PlayerAgent .
        ?teamFormation a :TeamFormation ;
        prov:generated ?team ;
        prov:qualifiedAssociation ?player .

        {
            # Counting all NPC character eliminations by each player and
gold earned
            ?npcElimination a :NPCCharacterElimination ;
            :wasEliminatedBy ?player ;
            prov:wasInformedBy :LOL_Match_1  ;
            prov:generated ?npcReward .
            ?npcReward :gold_amount ?npcGold .
        }
        UNION
          {
            # Counting all player character eliminations by each player
 and gold earned
            ?playerElimination a :PlayerCharacterElimination ;
            :wasEliminatedBy ?player ;
            prov:wasInformedBy :LOL_Match_1  ;
            prov:generated ?playerReward .
            ?playerReward :gold_amount ?playerGold .
         }
         UNION
         {
            # Counting all player deaths
           ?playerDeaths a :PlayerCharacterElimination ;
             prov:wasInformedBy :LOL_Match_1  ;
            :hasEliminatedPlayerCharacter ?player .
         }
     }
     GROUP BY ?player ?team
 }
```

La query identifica all'interno di uno **specifico MATCH** gli score relativi ad ogni giocatore, di ogni squadra. Per ogni giocatore e squadra identificati, la query conta le eliminazioni distinte di PNG e giocatori, calcola l'oro totale guadagnato da queste eliminazioni e conta il numero di volte in cui il giocatore è stato eliminato durante la partita specificata. Output:

- Il numero di personaggi non giocabili (PNG) eliminati da ciascun giocatore.
- Il numero di altri giocatori eliminati da ciascun giocatore.
- Il numero di volte in cui ogni giocatore è stato eliminato.
- L'ammontare totale di oro guadagnato da ciascun giocatore attraverso le eliminazioni.

7.2.1.2 Risultati

I risultati sono raggruppati per ogni giocatore e per ogni squadra, fornendo un'analisi dettagliata delle prestazioni di ciascun giocatore all'interno delle rispettive squadre. E' stato sostituito :LOL_Match_1 come Match da cui estrarre lo scoreboard.

7.2.2 Estrazione dell'ordine in cui le abilità sono state potenziate

```
SELECT  ?step ?ability ?character
WHERE {
  ?playerSession a :PlayerSession ;
     prov:wasInformedBy ?match ;
    prov:qualifiedAssociation ?player;
    :usedSkillSet ?skillset;
    prov:used ?inventory;
    prov:used ?character.
  ?inventory a :Inventory_Item_Sequence.
  ?character a :Game_Character.
  ?skillset a :SkillSet_Sequence.

  ?skillset col:hasMember ?step .
  ?step  :has_levelled_ability ?ability

  VALUES ?player { :LOL_Agent_Player_1 }
  VALUES ?match { :LOL_Match_1 }
}
```

La query estrae l'ordine in cui le abilità sono state potenziate durante un **Match** di un dato **Giocatore**.

7.2.2.1 Risultati

	step	ability	character
1	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_1	MobaOntology:LOL_Skill_Nasus_Q	MobaOntology:LOL_Character_Nasus
2	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_2	MobaOntology:LOL_Skill_Nasus_Q	MobaOntology:LOL_Character_Nasus
3	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_3	MobaOntology:LOL_Skill_Nasus_W	MobaOntology:LOL_Character_Nasus
4	MobaOntology:LOL_Skill_Upgrade_STEP_Nasus_Player1_S1_4	MobaOntology:LOL_Skill_Nasus_Q	MobaOntology:LOL_Character_Nasus

7.2.3 Estrazione dell'inventario

```
SELECT  ?step ?item
WHERE {
  ?playerSession a :PlayerSession ;
     prov:wasInformedBy ?match ;
    prov:qualifiedAssociation ?player;
    :usedSkillSet ?skillset;
    prov:used ?inventory;
    prov:used ?character.
  ?inventory a :Inventory_Item_Sequence.
  ?character a :Game_Character.
  ?skillset a :SkillSet_Sequence.

  ?inventory :has_build_step ?step .
  ?step  :requires_item ?item

  VALUES ?player { :LOL_Agent_Player_1 }
  VALUES ?match { :LOL_Match_1 }
}
```

Sono estratti dall'inventario di un **giocatore** in un suo **specifico Match,** tutti gli item acquisiti.

7.2.3.1 Risultati

	step	item
1	MobaOntology:LOL_Inventory_Nasus_S1_Item1	MobaOntology:LOL_Item_Sheen
2	MobaOntology:LOL_Inventoy_Nasus_S1_Item2	MobaOntology:LOL_Item_Cloth_Armor

7.2.3.2 Recupero informazioni relative alle abilità del Character

```
SELECT  ?character ?ability ?label ?description ?type
WHERE {

  ?character a :Game_Character.
  ?character :has_ability ?ability;
      :character_belongs_to_moba_game ?game
                .

    OPTIONAL { ?ability rdfs:label ?label. }
    OPTIONAL { ?ability rdfs:comment ?description. }
    ?ability :ability_type ?type


    VALUES ?game {:_GAME_League_Of_Legends}
    VALUES ?character {:LOL_Character_Ashe}

}
```

A partire dalla annotazioni vengono estratte per ogni abilità la **descrizione** e il suo ability_type che indica se l'abilità sia passiva o meno.

7.2.3.3 Risultati

	character	ability	label	description	type
1	MobaOntology:LOL_Character_Ashe	MobaOntology:LOL_Skill_Ashe_P	"Frost Shot"	"Innate - Frost Shot: Ashe's basic attacks and ability hits apply Frost to enemies for 2 seconds, which slows them by 20% \u2212 30% (based on level) for the duration. Basic attacks against enemies with Frost are modified to deal 120% (+ (75% + 35%) of critical strike chance) damage.Innate - Critical Slow: Ashe's	"PASSIVE"

7.2.3.4 Estrazione items

```
SELECT ?item  ?label ?description

WHERE {
    # Fetching all items
    ?game a :Moba_game.
    ?game  :has_item ?item.


    OPTIONAL { ?item rdfs:label ?label. }
    OPTIONAL { ?item rdfs:comment ?description. }


    VALUES ?game {:_GAME_League_Of_Legends}
}
```

Query molto semplice per l'estrazione degli Items per uno specifico gioco.

7.2.3.5 Risultati

	item	label	description
1	MobaOntology:LOL_Item_Chain_Vest	"Chain Vest"	"Chain Vest is an epic item in League of Legends icon League of Legends."
2	MobaOntology:LOL_Item_Iceborn_Gauntlet	"Iceborn Gauntlet"	"Iceborn Gauntlet is a mythic item in League of Legends icon League of Legends."
3	MobaOntology:LOL_Item_Kindlegem	"Kindlegem"	"Kindlegem is an epic item in League of Legends icon League of Legends."
4	MobaOntology:LOL_Item_Plated_Steelcaps	"Plated steelcaps"	"Plated Steelcaps is a boots item in League of Legends icon League of Legends."
5	MobaOntology:LOL_Item_Boots	"Boots"	"Boots is a boots item in League of Legends icon League of Legends."
6	MobaOntology:LOL_Item_Sheen	"Sheen"	"Sheen is a basic item in League of Legends icon League of Legends."
7	MobaOntology:LOL_Item_Cloth_Armor	"Cloth Armor"	"Cloth Armor is a basic item in League of Legends icon League of Legends."
8	MobaOntology:LOL_Item_Ruby_Crystal	"Ruby Crystal"	"Ruby Crystal is a basic item in League of Legends icon League of Legends."

8 APPLICAZIONE WEB

Come estensione, ho scelto di realizzare un'applicazione web utilizzando Angular come frontend e GraphDB come database per implementare l'interazione con l'utente descritta nel paragrafo 8.1. Le

Le figure di seguito rappresentano delle schermate dell'applicazione che ho sviluppato e rappresentano un 1 a 1 con il corrispettivo mock. Il frontend dell'applicazione è stato realizzato utilizzando Angular versione 12 e comprende tutta la logica dell'applicazione in quanto interroga direttamente GraphDB senza passare attraverso un backend intermediario (ad eccezione di un proxy utilizzato per evitare problemi con CORS).

L'applicazione è molto semplice ed è composta dai 3 moduli sopra descritti (di cui sono 2/3 implementati completamente):

- View Items
- View Characters
- View Players

Ciascuno si mappa completamente sui mock e la descrizione dell'interazione vista del capitolo 2.

E' stata fatta una piccola integrazione:

- Dal menu in alto è possibile selezionare una istanza specifica della Moba Ontologia per il dominio di League Of Legends (più espresso e con più esempi), Dota 2 e Pokemon Unite.

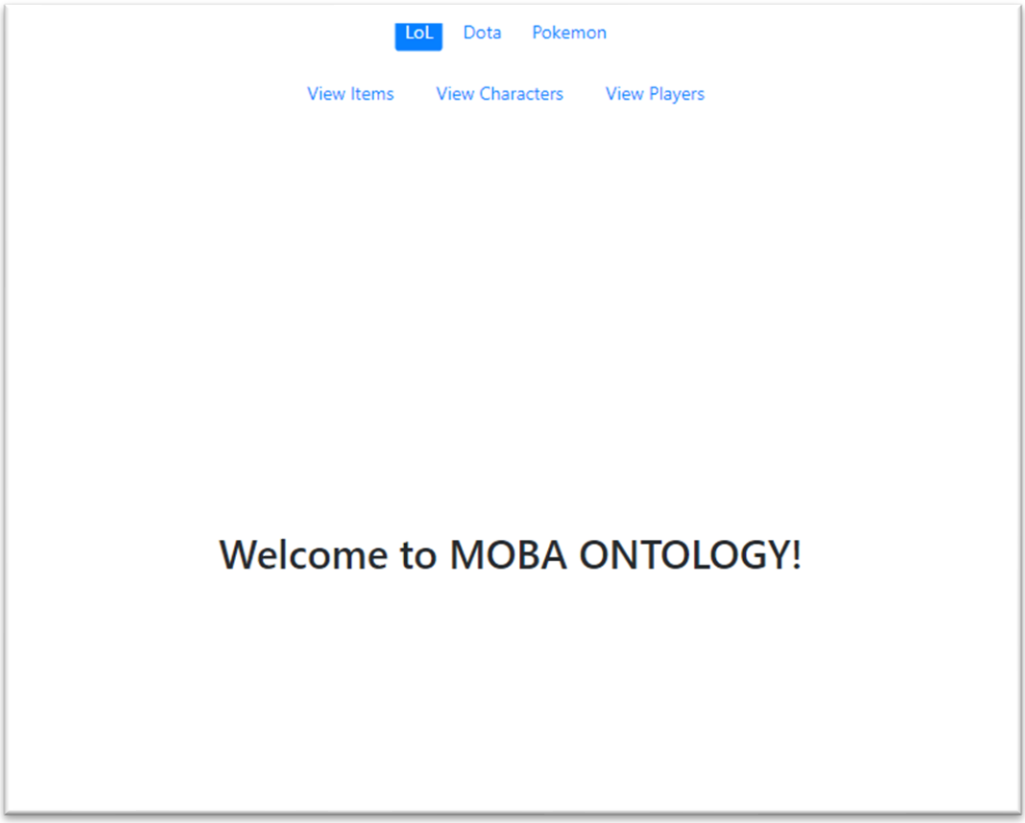


Figura 4 Home page

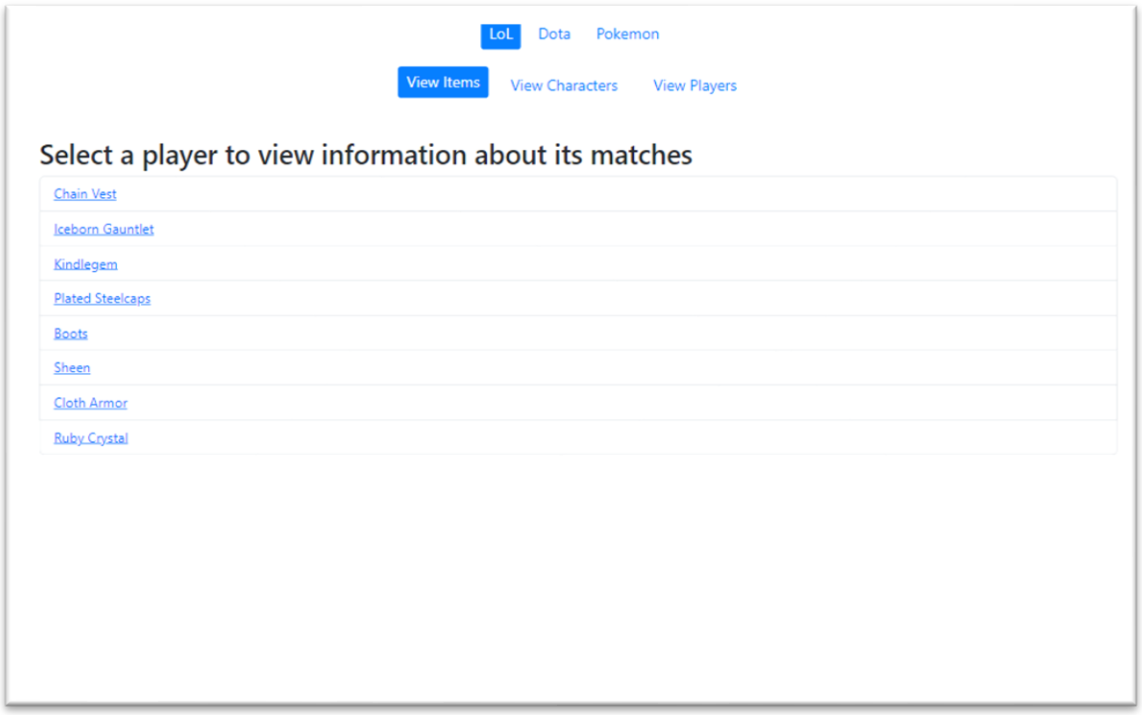


Figura 5 View items page per il gioco LOL. Purtroppo non è stato gestito il livello di navigazione successivo

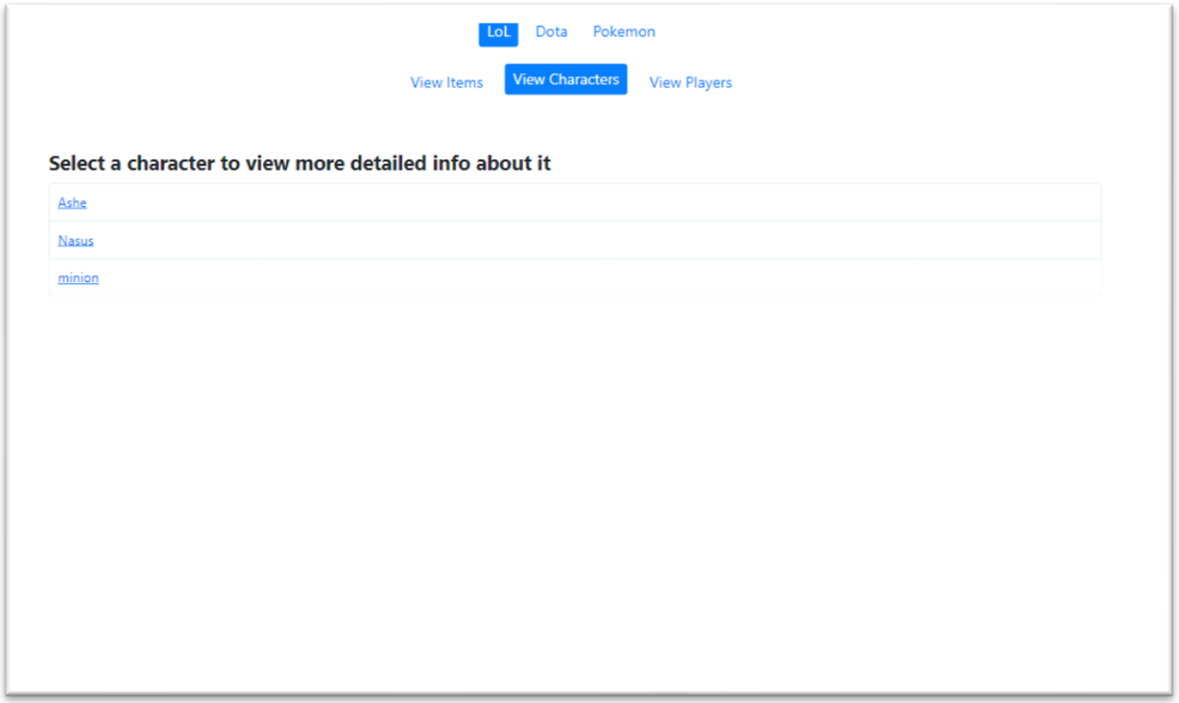


Figura 6 Pagine View characters per Lol

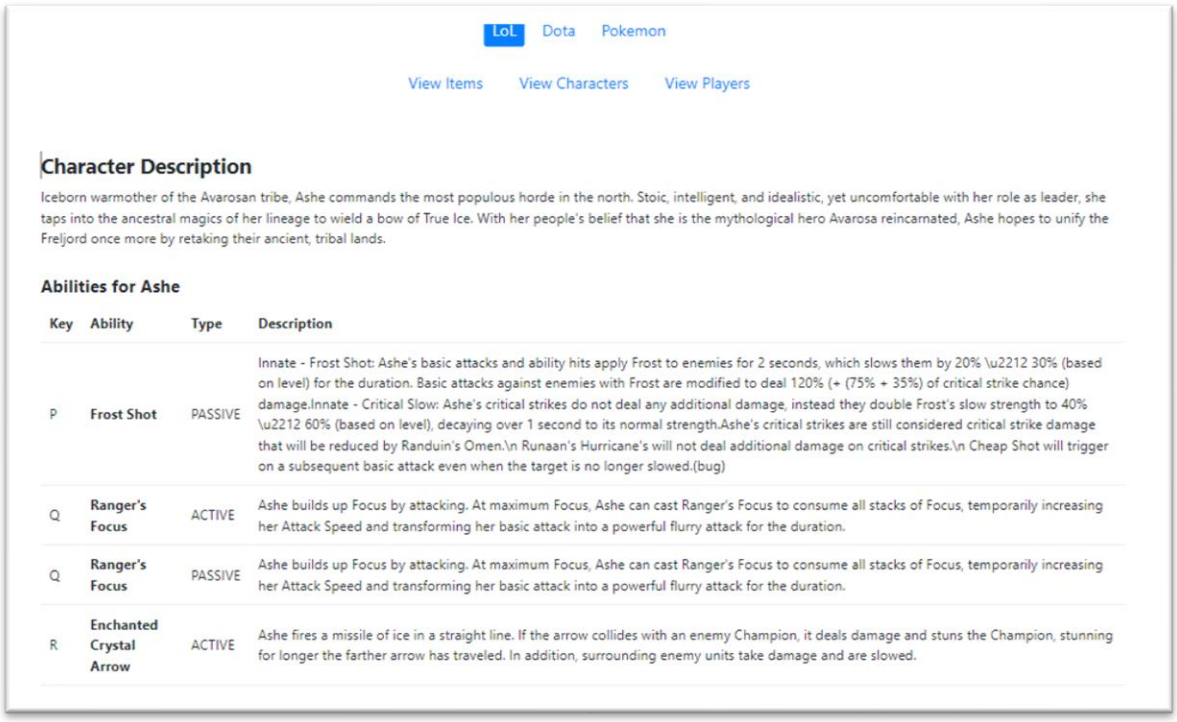


Figura 7 La pagina di dettaglio per un Character del gioco LOL selezionato

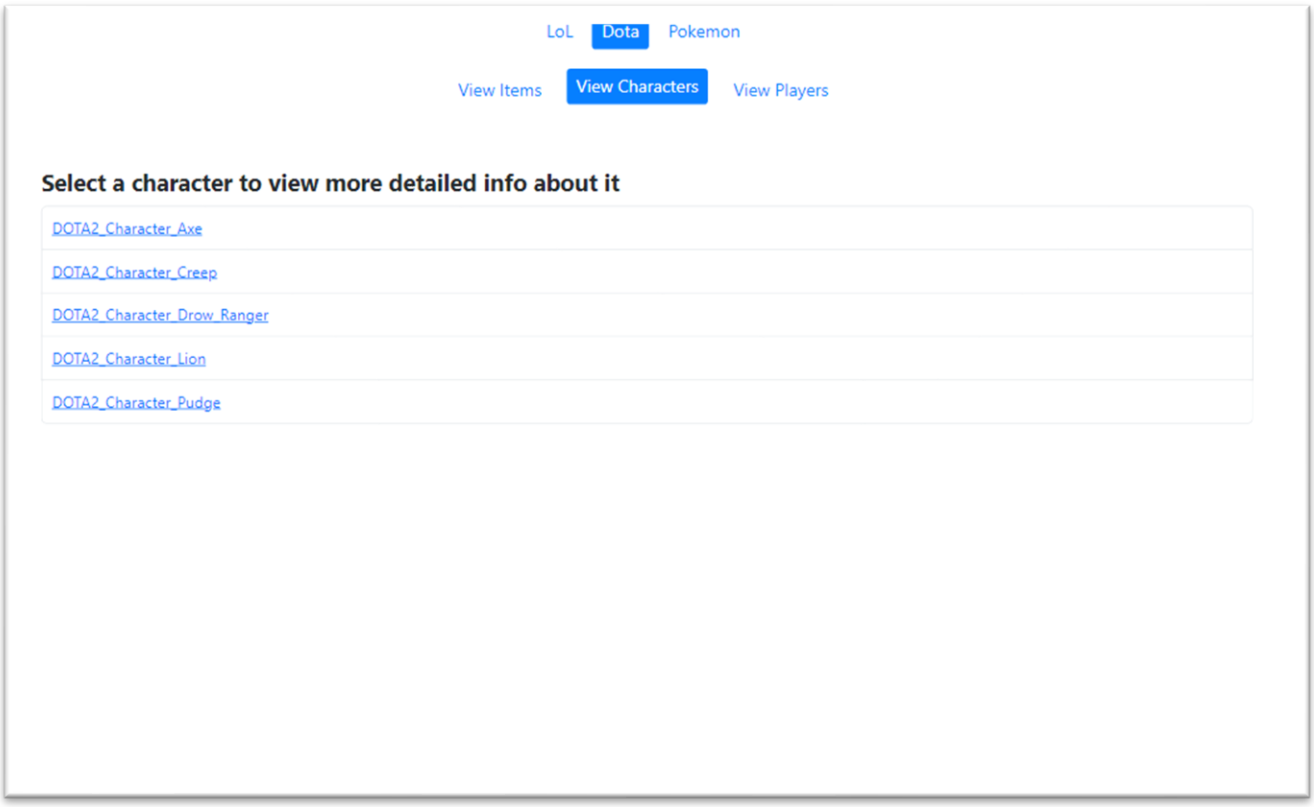


Figura 8 Pagine View characters perDota

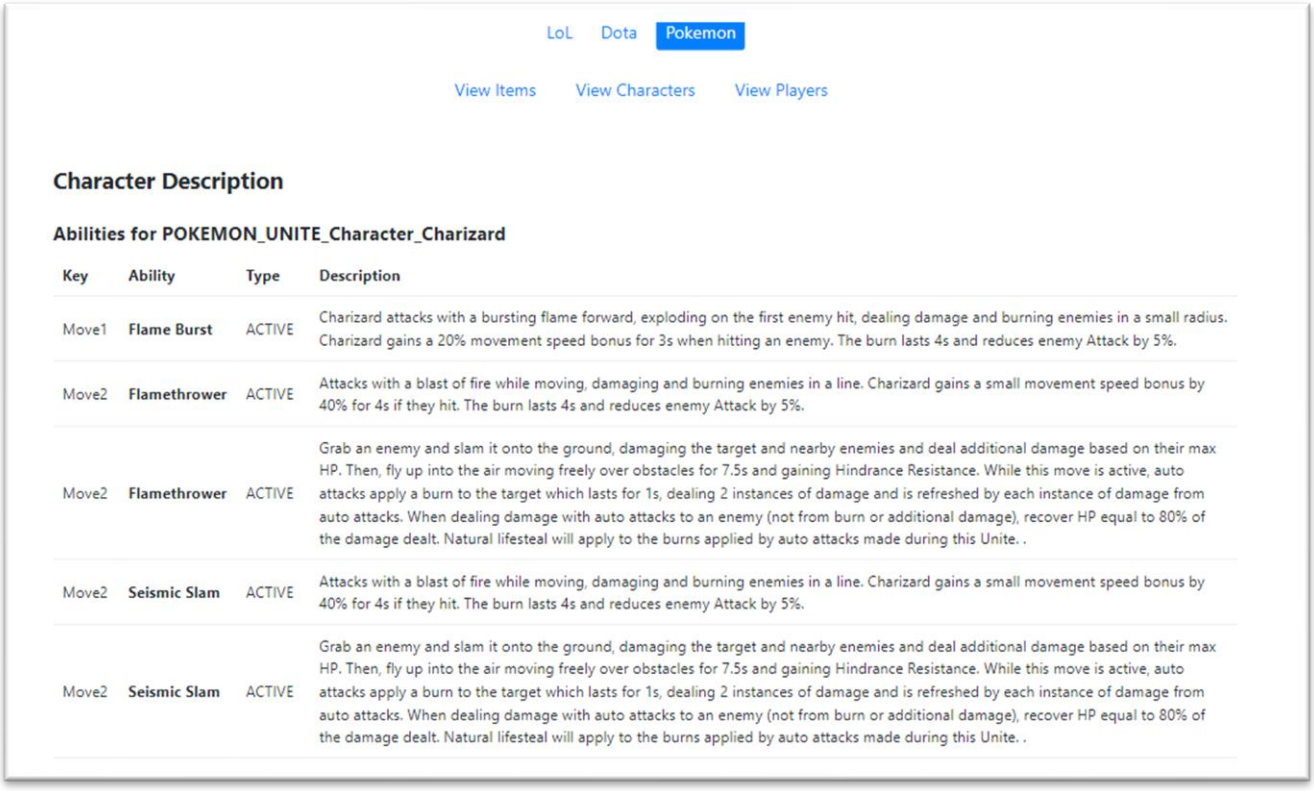


Figura 9 pagina di dettaglio per il characters Charizard nel gioco Pokemon

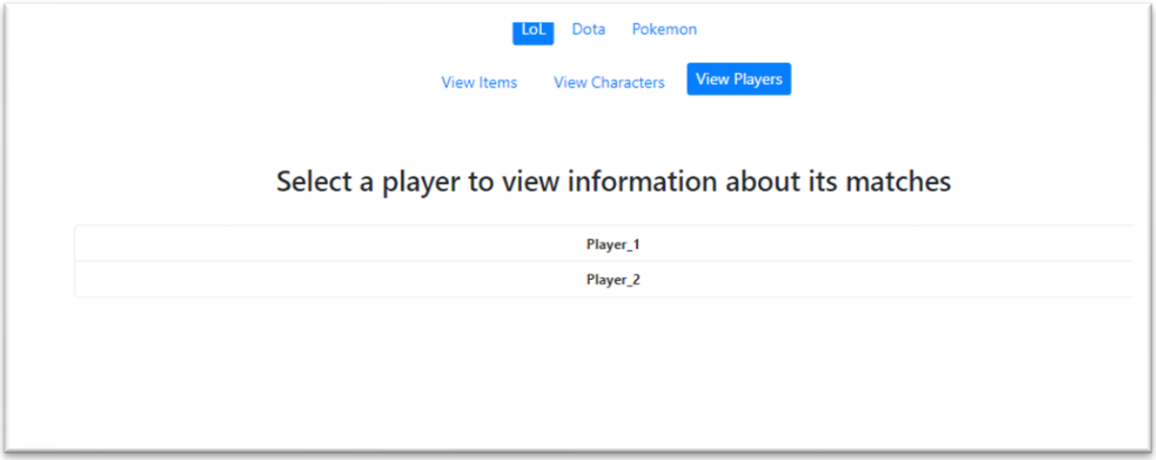


Figura 10 Navigazione nella sezione Players.

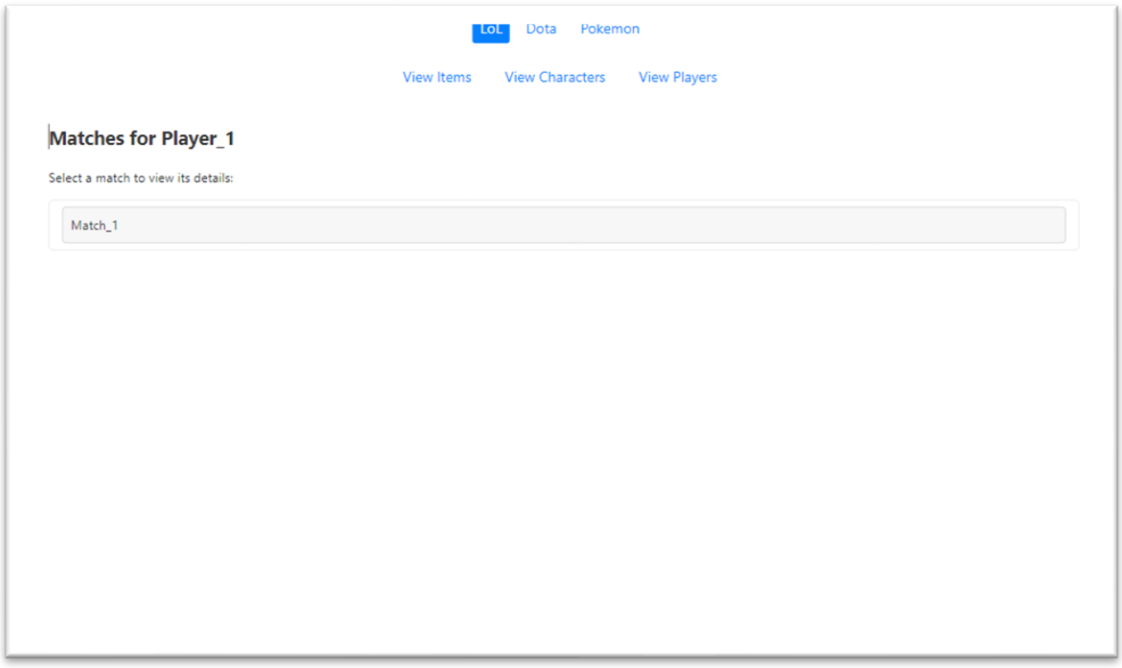


Figura 11 Selezione di un match specifico dell utente

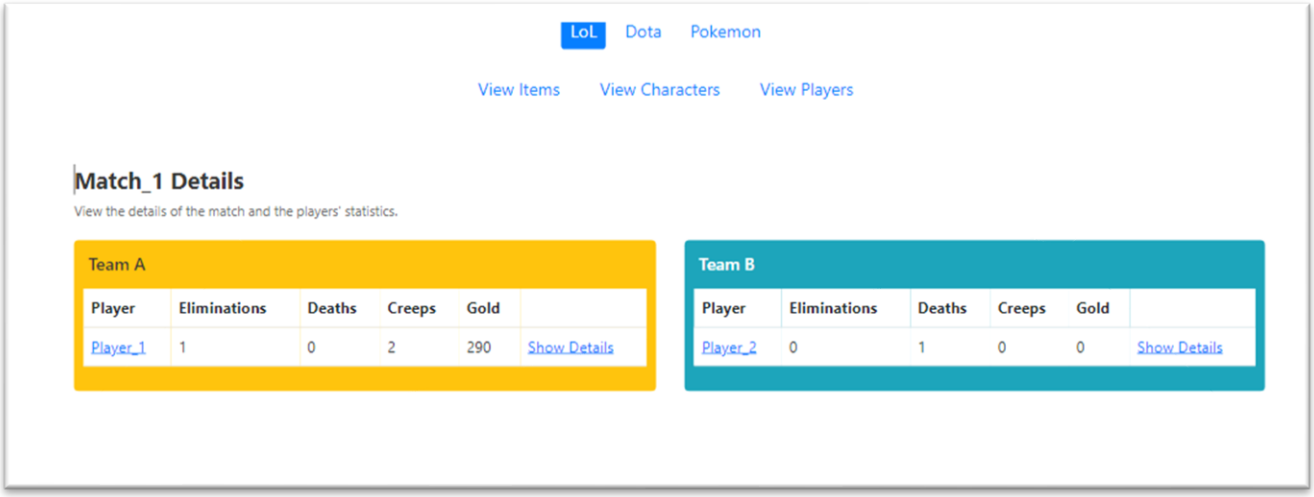


Figura 12 visualizzazione dei dettagli per il match

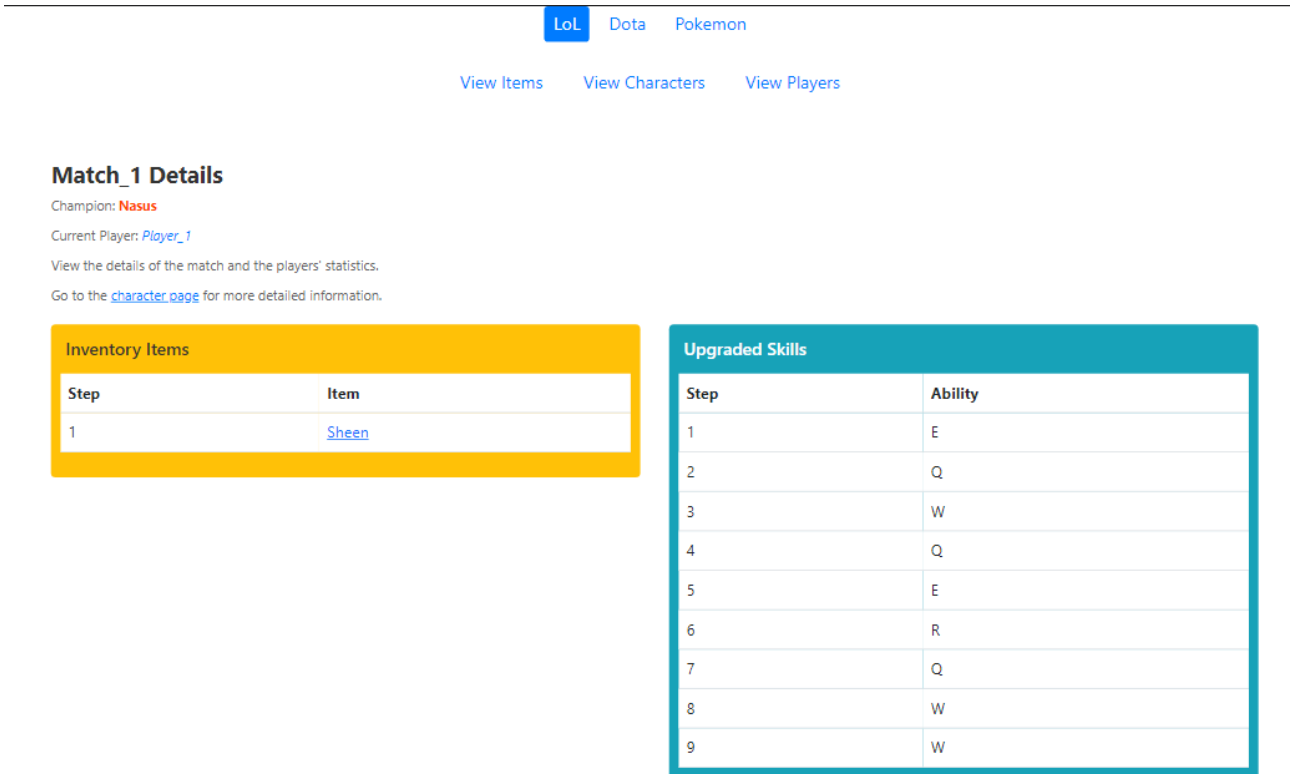


Figura 13 Dopo aver clickato su SHOW MORE l'utente può visualizzare più informazioni. Se preme su "Character Page" viene rimandato alla pagina di dettaglio per lo specifico Character

8.1 Connessione a Graph DB

Un semplice servizio Angular viene utilizzato da ciascun Page Component per contattare Graph DB.

In base alla rotta navigata, specifici Query Param sono aggiunti all'URL e sono utilizzati per guidare il filling dei contenuti del componente.

```
ngOnInit(): void {
  this.route.queryParamMap.subscribe((params) => {
    this.player_id = params.get('playerID')|| '';
    console.log(this.player_id);

    this.service.getMatches(this.player_id).subscribe((data) => {
      console.log(data);
      this.matches = data.results.bindings;
    });
  });
```

Figura 14 A tempo di init viene richiamato il servizio in modo da popolare la pagina con il dettaglio voluto

```
@Injectable({
  providedIn: 'root'
})
export class GraphDbService {

  private BASE_URL = 'http://localhost:4200/';
  private REPOSITORY = 'repositories/MobaOntology';
  GAME = '${this.GAME}';
  prefixes = `
        PREFIX : <http://www.semanticweb.org/user/ontologies/2023/6/MobaOntology#>
        PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
        PREFIX col: <http://www.ontologydesignpatterns.org/cp/owl/collectionentity.owl#>
        PREFIX prov: <http://www.w3.org/ns/prov#>
        PREFIX BAG: <http://www.ontologydesignpatterns.org/cp/owl/bag.owl#>
        PREFIX LIST:<http://www.ontologydesignpatterns.org/cp/owl/list.owl#>
        PREFIX owl: <http://www.w3.org/2002/07/owl#>
        PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

Figura 15 Per semplicità vengono sempre specificati tutti i prefissi

```
getMatches(player: string): Observable<any> {
  const query = `
    SELECT  ?match
    WHERE {
      # Fetching all player characters
      ?teamFormation a :TeamFormation ;
      prov:wasInfluencedBy ?match ;
      prov:qualifiedAssociation ?player .
    }
    VALUES ?player { :${player} }
  `;
  console.log(query)
  return this.executeQuery(query);
}
```

Figura 16 Definizione della query

```
executeQuery(query: string): Observable<any> {
  query = `${this.prefixes} \n ${query}`
  console.log(query)
  const headers = new HttpHeaders()
    .set('Accept', 'application/sparql-results+json,*/*;q=0.9')
    .set('Content-Type', 'application/x-www-form-urlencoded');
  const body = `query=${encodeURIComponent(query)}`;
  const url = `${this.BASE_URL}${this.REPOSITORY}`;

  return this.http.post<any>(url, body, { headers });
}
```

Figura 17 Esecuzione delle query

9 CONCLUSIONI E SVILUPPI FUTURI

L'ontologia progettata per i giochi MOBA si presta molto bene ai requisiti iniziali, tuttavia, può essere perfezionata in vari modi, ad esempio:

- **Descrizione e annotazione dettagliata dei personaggi**: Potrebbe essere utile includere informazioni più dettagliate sui personaggi disponibili in ogni gioco, come le loro abilità, le statistiche, la storia del personaggio e i ruoli preferiti.
- **Informazioni sulle modalità di gioco**: Oltre alla descrizione del gameplay generale, si potrebbero includere dettagli sulle diverse modalità di gioco disponibili.
- **Dettagli delle mappe:** Si potrebbero descrivere le diverse mappe presenti nel gioco, compresi i punti di interesse, gli obiettivi e le strategie comuni associate a ciascuna mappa.
- **Metadati del gioco**: L'ontologia potrebbe beneficiare dell'aggiunta di informazioni come il team di sviluppo, la data di rilascio, le patch e gli aggiornamenti importanti.
- **Informazioni sulla comunità**: Si potrebbero includere dati sulla comunità del gioco, come il numero di giocatori attivi, tornei e competizioni ufficiali, e la presenza di streamer o influencer noti associati al gioco.
- **Interazioni tra personaggi e oggetti**: Potrebbe essere utile descrivere come i diversi personaggi interagiscono con gli oggetti nel gioco, o come certi oggetti influenzano le capacità di un personaggio. Purtroppo, data la complessità non è stata esplorata la gestione delle Statistiche anche se la classe è stata definita
- **Strategie e tattiche:** L'ontologia potrebbe anche includere le strategie e tattiche comuni associate ai personaggi o alle squadre, dando **un'immagine più completa del gameplay**.
- **Informazioni sui ranking e sulle classifiche**: Dettagli sul sistema di classificazione del gioco, come il punteggio Elo o il sistema di classificazione MMR (Matchmaking Rating), potrebbero essere un'aggiunta utile.
- **Dettagli sul sistema di matchmaking**: Una descrizione del sistema di matchmaking del gioco, inclusi dettagli su come vengono formate le squadre e come viene determinato il livello di difficoltà delle partite.

Per quanto riguarda l'applicazione basata sull'ontologia, potrebbero essere apportati i seguenti miglioramenti:

- Una grafica accessibile e completa: attualmente si tratta di poco più di mock, l'impatto grafico non è dei migliori ma come piccolo prototipo può dare l'idea di dove potrebbe essere portato un progetto.
- Implementazione di funzionalità per l'inserimento dei dati sul gioco.
- Miglioramento della sicurezza implementando un backend che controlla i permessi di modifica degli utenti.
- Ampliamento delle funzionalità di ricerca, ad esempio, rendendo possibile la ricerca in base allo sviluppatore del gioco, alla data dell'ultimo aggiornamento, e molto altro.
- Implementazione della possibilità di ordinare i risultati in base a vari criteri, come la rilevanza, il numero di "mi piace", il numero di stelle su Github, ecc.
- Implementazione di paginazione o altri metodi per gestire grandi quantità di dati.