Implementasi Model Ontologi Pada Game Menggunakan Protégé

Rifqi Fauzi Rahmadzani
Magister Teknologi Informasi
Universitas Gadjah Mada
Yogyakarta, Indonesia
Email: rifqifauzi@mail.ugm.ac.id

Abstrak—*Game* merupakan salah satu media teknologi yang banyak digemari masyarakat, mulai dari usia dini hingga orang-orang dewasa. *Game* selalu menjadi daya tarik sendiri, selain berfungsi sebagai sarana hiburan juga dapat berperan dalam menjalin sarana interaksi dan kompetisi dengan temannya. Penelitian ini bertujuan untuk membuat aplikasi ontologi mengenai *game* menggunakan protégé 5.5.0. Model ontologi dikembangkan dalam bentuk ontologi dengan struktur hirarki *class*, tipe *class*, *instance*, dan *property*. Model ontologi yang dihasilkan dapat memberikan informasi dan klasifikasi terhadap masing-masing karakteristik permainan. Dengan demikian dalam menggunakan model ini dapat didefinisikan melalui query yang direpresentasikan ke dalam bahasa SPARQL.

Kata kunci—*Game*, Ontologi, OWL, Protégé, SPARQL

I. INTRODUCTION

Seiring dengan berkembang pesatnya teknologi, dimana hampir semua lingkup bidang teknologi sekarang menggunakan basis internet. Salah satu teknologi populer yang saat ini dirasakan manfaatnya adalah *world wide web*. Hingga saat ini web menjadi penghasil sumber data terbesar bagi setiap pengguna dikarenakan konten web yang saling terhubung dan dapat diakses melalui koneksi internet. Setiap pengguna yang memiliki hak akses dapat menambahkan konten secara bebas. Semakin banyak informasi yang ditambahkan, semakin besar ukuran web. Hal ini akan mempengaruhi proses pencarian informasi yang dilakukan pada *web* dalam waktu yang relatif singkat. Selain itu, penting untuk memperhatikan faktor akurasi pencarian dalam memilah suatu informasi [1].

Semantik web merupakan bagian dari teknologi web yang ada saat ini, dengan adanya semantik web, *website* tidak hanya dimengerti oleh manusia namun juga dapat dipahami oleh mesin. Semantik web selain dapat memahami makna atau konsep kata juga dapat memahami hubungan logis antara keduanya. Salah satu teknologi pendukung dari semantik web adalah ontologi. Ontologi dapat merepresentasikan suatu pengetahuan tentang spesifik domain dan telah ditetapkan diberbagai bidang. Sehingga ontologi dapat digunakan untuk penyajian informasi secara semantik serta melakukan pemetaan informasi secara sistematis dan terstruktur [2].

Game merupakan salah satu media yang banyak digemari masyarakat oleh segala usia, baik anak-anak maupun dewasa. Di era yang serba digital ini *game* selalu menjadi daya tarik seseorang selain untuk berkompetisi dengan rekannya juga sebagai kompetisi dalam kejuaraan. Dengan demikian, *game* menjadi salah satu domain yang membutuhkan pemahaman semantik untuk memudahkan pencarian mengenai informasi. Informasi ini diharapkan dapat memberikan ketepatan dan keakuratan dalam prosesnya. Sehingga, perlu untuk menyimpan data *game* secara semantik untuk membantu mesin pencari dapat bekerja secara optimal dalam menemukan informasi.

Tujuan dari penelitian ini adalah melakukan pengembangan model ontologi pada domain *game*. Kemudian melakukan pengujian terhadap model ontologi melalui perintah-perintah yang direpresentasikan kedalam bentuk SPARQL.

II. KONSEP DASAR

2.1. Ontologi

Ontologi merupakan kunci untuk penerapan semantik web [3]. Ontologi dapat direpresentasikan dalam bentuk objek, properti dari objek, dan relasi diantara setiap objek. Adapaun bahasa yang digunakan dalam ontologi adalah *Ontology Web Language* (OWL). Pada awalnya OWL didesain untuk merepresentasikan informasi tentang kategori dari suatu objek dan bagaimana objek tersebut berhubungan. Selain itu, OWL juga dapat menyediakan informasi mengenai objek itu sendiri. Sebagai hasil bagian yang dilakukan oleh Semantik Web W3C, OWL memiliki standar dalam web semantik, yaitu bahasa yang dikelompokkan secara bersama dengan XML dan RDF [1].

2.2. SPARQL

SPARQL merupakan bahasa query untuk mengakses dokumen RDF [4]. *Graph* RDF terdiri dari triple yang terbentuk dari subjek, predikat, dan objek. Ekspresi RDF dapat disimpin dalam format lain seperti XML dan relasional *database*. Struktur bahasa yang digunakan pada SPARQL hampir mirip dengan SQL pada relasional database. SPARQL dalam mendapatkan informasi dari *graph* RDF menyediakan fasilitias seperti, mengekstrak informasi dalam bentuk URI, blank node, dan literal, selain itu mengekstrak RDF *subgraph*, serta dapat membangun *graph* RDF baru berdasarkan query *graph*. Contoh SPARQL dapat dilihat pada gambar 1.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT ?subject ?object
         WHERE { ?subject rdfs:subClassOf ?object }
```

Gambar 1. Contoh query SPARQL

2.3. Protégé

Protégé merupakan perangkat lunak yang dikembangkan oleh otorisasi Standford, khususnya dalam bidang ontologi. Protégé digunakan untuk membuat sebuah domain ontologi, menyesuaikan form untuk *entry* data dan memasukan data. Beberapa format yang didukung seperti OWL, RDF, XML, dan HTML [5]. Protégé dibuat menggunakan bahasa pemrograman Java. Fitur-fitur yang terdapat dalam Protégé

sudah ditampilkan melalui *Graphical User Interface* (GUI) sehingga lebih interaktif dan komunikatif. Setiap tab *class* dalam editor ontologi berfungsi untuk mendefinisikan *class* dan hirarki class, *property*, dan nilai *property*, relasi antara class dan *property* dari relasi tersebut.

III. PEMBAHASAN

3.1. Perancangan Ontologi

Pada tahapan ini dilakukan perancangan ontologi, hasil rancangan ontologi dapat dilihat pada tabel 1.

TABEL 1. CLASS, SUBCLASSOF, DAN OBJECTPROPERTY

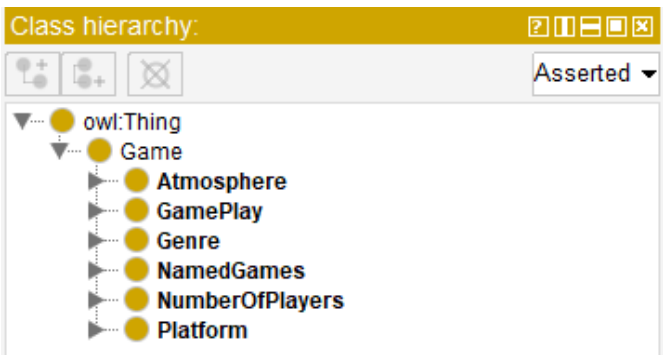
Class	Subclassof
Atmosphere	Thing
Adventure	Atmosphere
Aggressive	Atmosphere
Peaceful	Atmosphere
Gameplay	Thing
BasicGamePlayRule	Gameplay
GoalOriented	Gameplay
Genre	Thing
GameOfAction	Genre
GameOfInformation	Genre
NamedGames	Thing
CallOfDuty	NamedGames
Chess	NamedGames
GrandTheftAuto	NamedGames
QuizPlanet	NamedGames
Sudoku	NamedGames
SuperMario	NamedGames
WorldOfThanks	NamedGames
NumberOfPlayers	Thing
MultiPlayer	NumberOfPlayers
SinglePlayer	NumberOfPlayers
Platform	Thing
GamingConsole	Platform
MobileDevices	Platform
PersonalComputer	Platform
Web	Platform

3.2. Implementasi Ontologi

Setelah dilakukan perancangan ontologi, langkah selanjutnya adalah implementasi konsep ontologi menggunakan software protégé. Pada tahap ini dibuat class, instances, properties, dan query.

3.2.1. *Class*

Class didefinisikan sebagai sekumpulan individu yang memiliki persyaratan sesuai untuk menjadi anggota *class* itu. *Class* dikenal sebagai komponen kunci dari ontologi OWL. Dalam ontologi *game* ini dibuat enam kelas utama. Karena kelas "owl:Thing" mewakili semua individu, semua kelas tersebut adalah sub-kelas dari owl:Thing. Gambar 2 menunjukkan hierarki kelas yang juga dikenal sebagai Taksonomi.



Gambar 2. Hirarki *Game Class*

a. *Atmosphere class*

Seperti yang ditunjukkan pada gambar 3, seluruh individu adalah *instance* dari *Game class*. *Game class* dapat diklasifikasikan sesuai dengan *Atmosphere* yang dialami dalam game oleh pemain. Pemain yang mewakili dari *Atmosphere class* ini terdapat *adventurous, aggressive,* dan peaceful. Jadi dalam hal ini, permainan setidaknya harus memiliki satu atmosfer.

Gambar 3. Hirarki *Atmosphere Class*

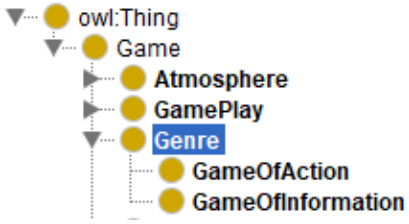
b. *GamePlay class*

GamePlay Class menentukan bagaimana *game* itu dimainkan. *Class* ini menunjukkan aturan dan proses permainan. Terdapat *subclass BasicGamePlayRule* dan *GoalOriented*. Setiap *game* harus memiliki setidaknya satu kelas *gameplay*.

Gambar 4. Hirarki *Gameplay Class*

c. *Genre class*

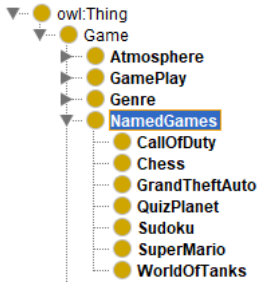
Genre adalah kategori *game* tertentu yang terkait dengan karakteristik *gameplay*. *Genre* biasanya tidak ditentukan oleh pengaturan atau cerita permainan atau media permainannya, namun oleh cara pemain berinteraksi dengan permainan. Sebagai contoh, *game* dapat diklasifikasikan ke dalam *genre* seperti *Game of action* maupun *Game of control*.



Gambar 5. Hirarki *Genre Class*

d. *NamedGames class*

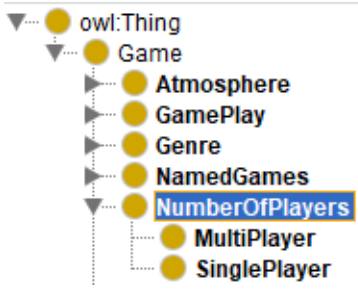
Pada *NamedGames class* terdiri dari beberapa daftar nama permainan yang dimasukkan. Hal ini meliputi *Call of Duty, Chess, Grand Theft Auto, Quiz Planet, Sudoku, Super Mario,* dan *World of Tanks*.



Gambar 6. Hirarki *NamedGame Class*

e. *NumberOfPlayers class*

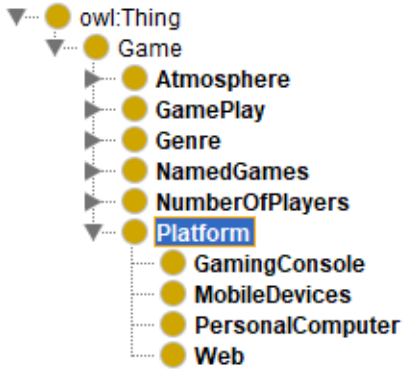
Sebuah *game* dapat diklasifikasikan berdasarkan jumlah pemain yang dimainkan pada *game* tersebut. Dua klasifikasi tersebut adalah *Multi Player* dan *Single Player*.



Gambar 7. Hirarki *NumberOfPlayers Class*

f. *Platform class*

Platform class menentukan perangkat operasional yang dimainkan pada *game*. Beberapa *platform* umumnya dapat berupa *gaming console, mobile devices, personal computer,* dan *web*. Setiap *game* setidaknya harus memiliki satu *platform*.



Gambar 8. Hirarki *Platform Class*

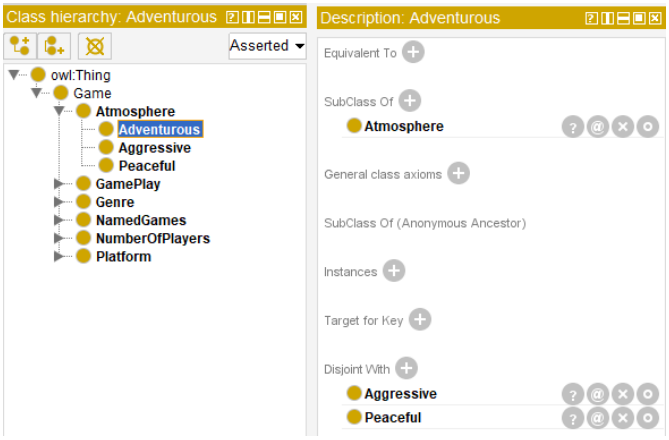
3.2.2. Tipe *Class*

a. *Sibling class*

Jika *class* A dan B adalah *sibling class*, berarti dapat memiliki kelas induk yang sama dan inherits terhadap *properties* dan *characteristics*. Jadi *class* A dan B ada kemungkinan memiliki karakterisitik khusu namun juga ada karakteristik yang diturunkan oleh *parent class*. Sebagai contoh, pada Gambar 2 *Atmosphere* dan *Gameplay* merupakan *sibling class*.

b. *Disjoint class*

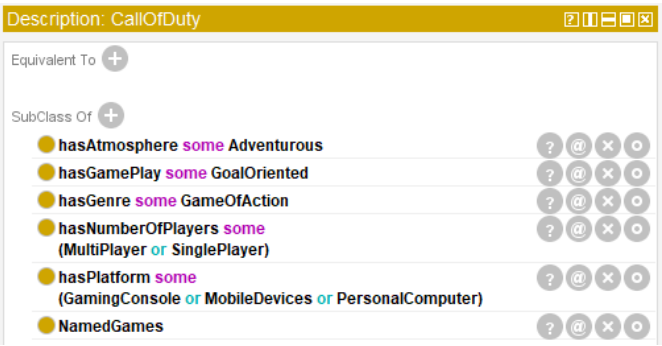
Jika dua *class* dikatakan sebagai *class* terpisah, maka individu tertentu tidak dapat menjadi turunan lebih dari satu *class* tersebut. Sebagai contoh dalam ontologi *game*, *subclass* dari kelas *atmosphere* yaitu *adventuruous* mengalami *disjoint* dengan *aggressive* dan *peaceful*. Hal ini menyatakan bahwa sebuah *game* tidak boleh memiliki lebih dari satu atas tipe atmosfer.



Gambar 9. *Disjoint class*

c. *Primitive class*

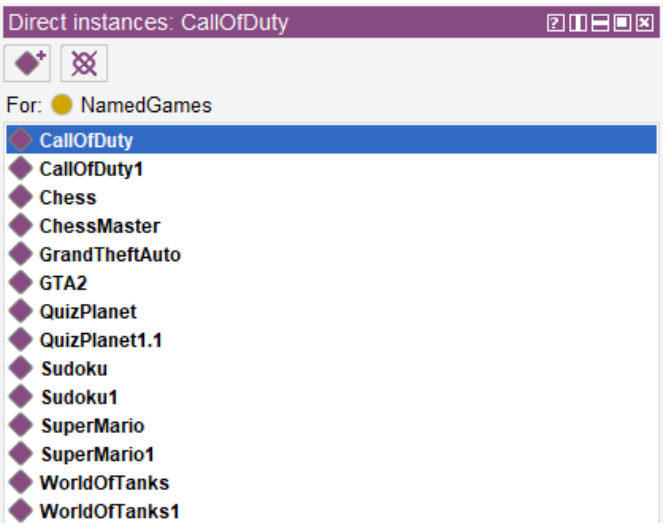
Primitive class adalah kondisi yang diperlukan dalam memberikan persyaratan untuk menjadi anggota *class*. Berarti jika suatu individu adalah anggota *class*, maka individu tersebut harus memenuhi persyaratan. Seperti halnya pada *call of duty* harus memenuhi persyaratan seperti yang ditunjukkan pada Gambar 10.



Gambar 10. *Primitive class*

3.2.3. *Instance*

Instance menampilkan daftar individu yang dinyatakan sebagai *instance* dari kelas yang dipilih. Pada gambar 11 menunjukkan daftar individu nama permainan. Beberapa individu-individu terlihat yang merupakan sebagai turunan dari *class* ini.



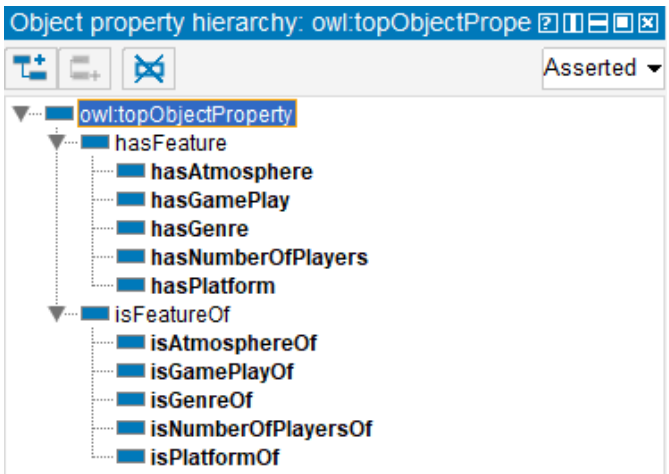
Gambar 11. Daftar *instance*

3.2.4. *Properties*

Properti OWL digunakan dalam mewakili hubungan. Di OWL ada dua jenis properti utama yaitu properti Objek dan properti Tipe Data. Selain itu juga memiliki jenis properti ketiga yang disebut properti Anotasi. Properti anotasi digunakan dalam menambahkan informasi ke *class*, individu, dan properti tipe objek / data.

a. *Object properties*

Properti objek menunjukkan hubungan antar individu. Dalam ontologi *game* yang dikembangkan, ada dua belas properti objek yang dibuat. Kelas "owl:topObjectProperty" mewakili semua properti objek, semua properti yang dikembangkan pada ontologi ini adalah sub-properti dari owl:topObjectProperty. Gambar 12 merupakan daftar properti objek dari ontologi *game*.



Gambar 12. Daftar *object properties*

b. *Property domain* dan *ranges*

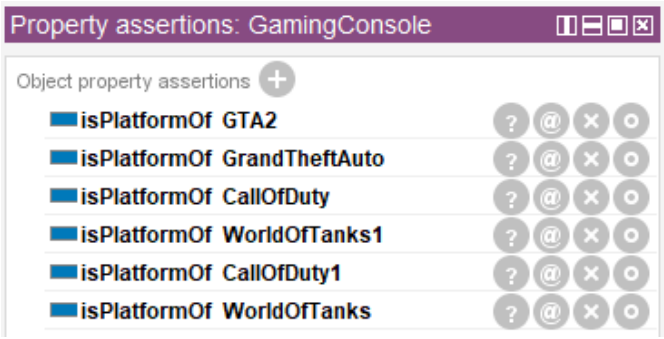
Setiap properti objek memiliki *domain* dan *range* yang ditentukan. Individu domain dikaitkan ke individu-individu dari *range* melalui properti. Untuk setiap properti yang ada pada ontolgi *game* dapat direpresentasikan dalam *domain* dan *range* seperti pada tabel 2 berikut.

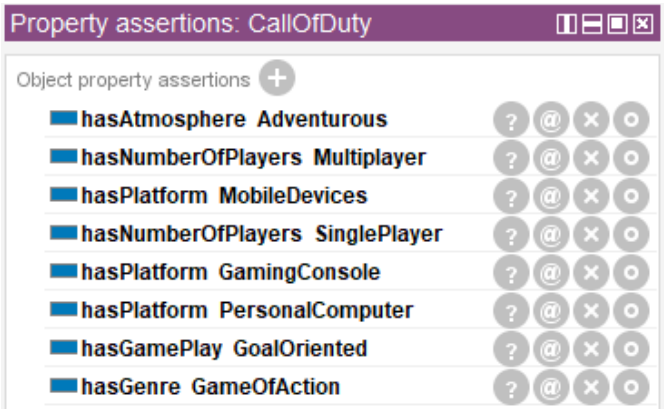
TABEL 2. OBJECT PROPERTY, DOMAIN, DAN RANGE

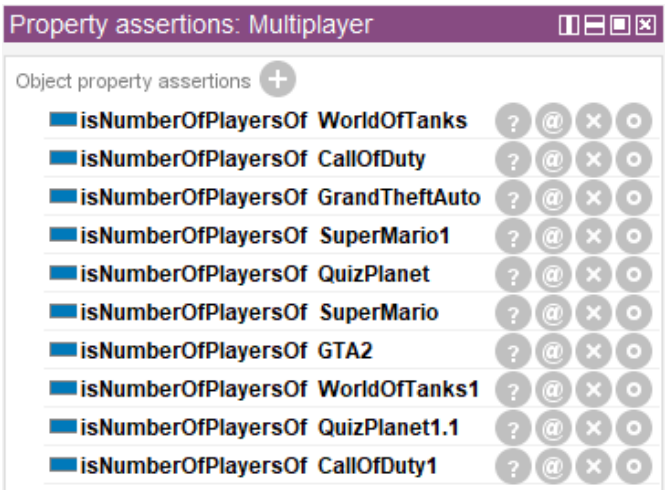
Object Property	Domain	Range
hasAtmosphere	Game	Atmosphere
hasGamePlay	Game	GamePlay
hasGenre	Game	Genre
hasNumberOfPLayers	Game	NumberOfPlayers
isAtmosphereOf	Atmoshpere	Game
isGamePlayOf	Gameplay	Game

c. *Inverse properties*

Dalam tabel 2 menunjukkan beberapa properti yang *invers* dari properti lainnya. Jenis-jenis properti itu disebut *invers property*. *Invers property* memiliki *invers domain* dan rentang properti terkait. Sebagai contoh, properti "hasAtmoshere". *Domain*-nya adalah "Game" dan *range*-nya adalah "Atmosphere". Maka *invers property*-nya adalah "isAtmosphere". (Gambar 13)



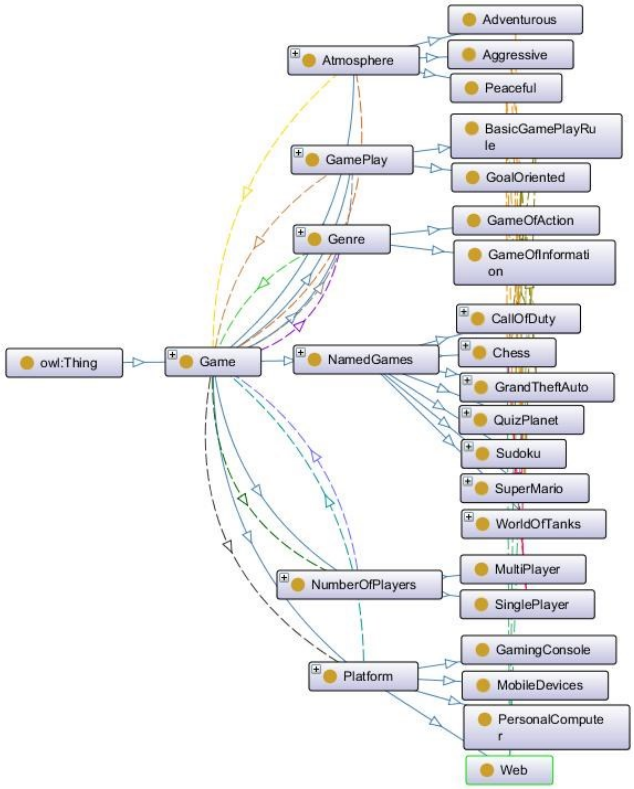




Gambar 13. Daftar *invers properties*

3.3. Diagram ontologi

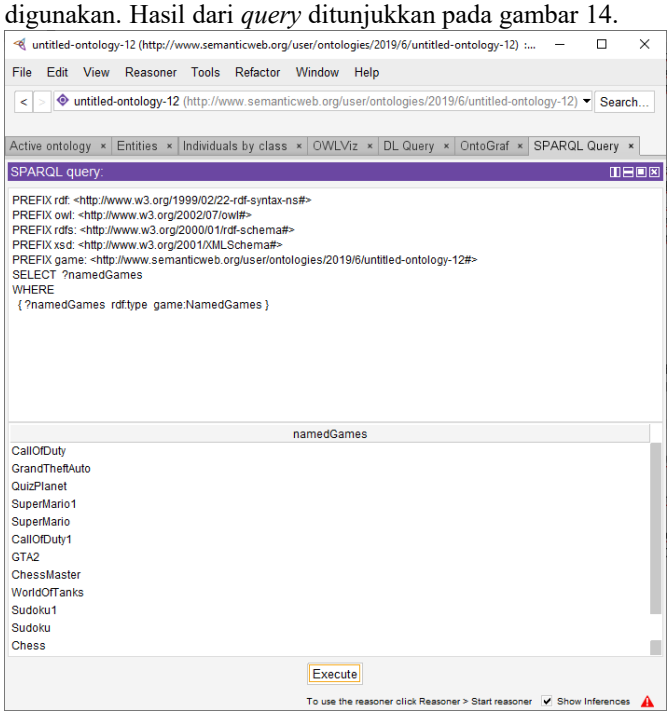
Pada diagram ontologi dapat menampilkan relasi tiap *class* dan *subofclass* hingga *instance* yang telah dibuat. Relasi yang terbentuk berdasarkan relasi yang telah diatur pada *object property* dan *rule*. Diagram *ontologi* mengenai model *game* ini ditunjukkan pada gambar 14.



Gambar 13. Hirarki *class* diagram ditunjukkan menggunakan ontograph

3.4. *Query* menggunakan SPARQL *query*

Setelah dilakukan implementasi ontologi dengan menentukan *class, tipe class, instance,* dan *property.* Tahapan selanjutnya sebelum ontologi di implementasikan ke pengembangan *semantic web* yaitu mendefinisikan *query* untuk mengetahui apakah model ontologi sudah bisa digunakan. Hasil dari *query* ditunjukkan pada gambar 14.



Gambar 14 merupakan contoh *query* untuk menampilkan nama permainan

Sebelum membuat query perlu menambahkan prefix yang digunakan untuk mendefinisikan URI (Universal Resource Identifier). URI yaitu semacam identitas unik yang tidak harus dapat menghubungkan atau mengakses sumber daya (resource) [6]. Prefix yang digunakan sebagaimana berikut.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX game:
<http://www.semanticweb.org/user/ontologies/2019/6/untitled-ontologi-12#>
```

Untuk *query* yang lain dapat diimplementasikan sebagaimana berikut.

Perintah 1 : Tampilkan semua atmosphere yang ada dalam game.

```
SELECT ?atmosphere
    WHERE {
    ?atmosphere rdf:type game:Atmosphere
}
```

Hasil : Adventurous, Aggressive, Peaceful

Perintah 2 : Tampilkan semua game play yang ada dalam game.

```
SELECT ?gamePlay
    WHERE {
    ?gamePlay rdf:type game:GamePlay
}
```

Hasil : Adventurous, Aggressive, Peaceful

Perintah 3 : Tampilkan semua genre permainan.

```
SELECT ?genre
    WHERE {
    ?genre rdf:type game:Genre
}
```

Hasil : GameOfAction, GameOfInformation

Perintah 4 : Tampilkan semua daftar permainan

```
SELECT ?namedGames
    WHERE {
    ?namedGames rdf:type game:NamedGames
}
```

Hasil : CallOfDuty, CallOfDuty1, Chess, GrandTheftAuto, GTA2, QuizPlanet, QuizPlanet1.1, Sudoku, Sudoku1, SuperMario, SuperMario1, WorldOfTanks, WorldOfTanks1

Perintah 5 : Tampilkan semua jenis player

```
SELECT ?numberOfPlayers
    WHERE {
    ?numberOfPlayers rdf:type game:NumberOfPlayers
}
```

Hasil : MultiPlayer, SinglePlayer

Perintah 6 : Tampilkan semua jenis platform permainan

```
SELECT ?platform
    WHERE {
    ?platform rdf:type game:Platform
}
```

Hasil : GamingConsole, MobileDevices, PersonalComputer, Web

Perintah 7 : Tampilkan daftar permainan yang berjenis multiplayer

```
SELECT ?game
    WHERE {
    ?game rdf:type game:NamedGames
    FILTER (?NumberOfPlayers >1)
}
```

Hasil : GrandTheftAuto, SuperMario

Perintah 8 : Tampilkan game yang mempunyai atmosphere adventurous

```
SELECT ?game
    WHERE {
    ?game game:hasAtmosphere cake:Adventurous
}
```

Hasil : CallOfDuty, CallOfDuty1, WorldOfTanks, WorldOfTanks1

Perintah 9 : Tampilkan game yang mempunyai atmosphere aggressive dengan jenis player adalah multiplayer yang memiliki platform dan genre

```
SELECT ?game
    WHERE {
    ?game game:hasAtmosphere cake:Aggressive;
    game:hasPlatform ?Platform;
    cake:hasGenre ?Genre
    cake:hasNumberOfPlayers ?NumberOfPlayers
    FILTER (?NumberOfPlayers >1)
}
```

Hasil : GrandTheftAuto, SuperMario

Perintah 10 : Cari game yang bernama GrandTheftAuto

```
SELECT ?subject
    WHERE {
    if(!(Atmosphere.equals("GrandTheftAuto")))
    queryString += "?subject sw:hasAtmosphere game:" +
Atmosphere
}
```

Hasil : GrandTheftAuto

IV. KESIMPULAN

Pada penelitian yang telah dilakukan dalam mengembangkan ontologi terhadap game, menunjukkan bahwa inferensi pengetahuan dengan menggunakan ontologi dapat dilakukan dengan baik. Sehingga dapat diimplementasikan ke dalam semantic web. Selain itu, model ontologi yang dibuat telah berhasil menjawab sebagian besar pertanyaan-pertanyaan mengenai beberapa karakteristik dalam game.

V. SARAN

Setelah melakukan perancangan, implementasi, dan pengujian ontologi mengenai game, masih terdapat keterbatasan dalam pengerjaan penelitian. Sehingga dapat diimplementasikan dalam pengembangan selanjutnya yaitu dengan menambah domain yang digunakan, mengingat domain dalam pengembangan ontologi ini masih terbatas.

REFERENSI

[1] S. Kasus, P. Studi, and S. Informasi, "Penerapan Web Semantik Untuk Aplikasi Pencarian Pada Repositori Koleksi Penelitian, Studi Kasus: Program Studi Sistem Informasi Stmik Mikroskil Medan," *JSM STMIK Mikroskil*, vol. 15, no. 1, pp. 51–60, 2014.

[2] D. I. K. Banyumas *et al.*, "Model Ontologi Untuk Informasi Pariwisata Di Kabupaten Banyumas," *Seminar Nasional Teknologi Informasi dan Multimedia*, pp. 37–42, 2016.

[3] A. Gali, C. X. Chen, K. T. Claypool, and R. Uceda-sosa, "From Ontology to Relational Databases," *Conceptual Modeling for Advanced Application Domains*, pp. 1–12, 2004.

[4] Bob DuCharme . Learning SPARQL. O'reilly. 2011

[5] A. Nugroho, "Membangun ontologi jurnal menggunakan protégé (Build Journal Of Use Protege Ontology)," *Jurnal Transformatika*, vol. 10, no. 1, pp. 20–25, 2012.

[6] M. Yani and P. N. Indramayu, "Implementasi Metoda Penyimpanan dan Pengambilan Ontologi Berbasis File Menggunakan Java dan Jena," *Eksplora Informatika*, vol. 4, no. 2, pp. 165–174, 2015.