Computer Games Ontology

Documentation

Jan Sawicki

sawickij@student.mini.pw.edu.pl

Semantic Data Processing
Faculty of Mathematics and Information Science
at Warsaw University of Technology

Schedule

26.11.2019	Implementation plan - second version Documentation draft
17.12.2019	Alpha implementation + alpha documentation
07.01.2020	Beta implementation + beta documentation
21.01.2020	Submit final working code and documentation
28.01.2020	Final submission + presentations

Requirements

Alpha stage

- General description
- 400 axioms
- Known considerations noted

Beta stage

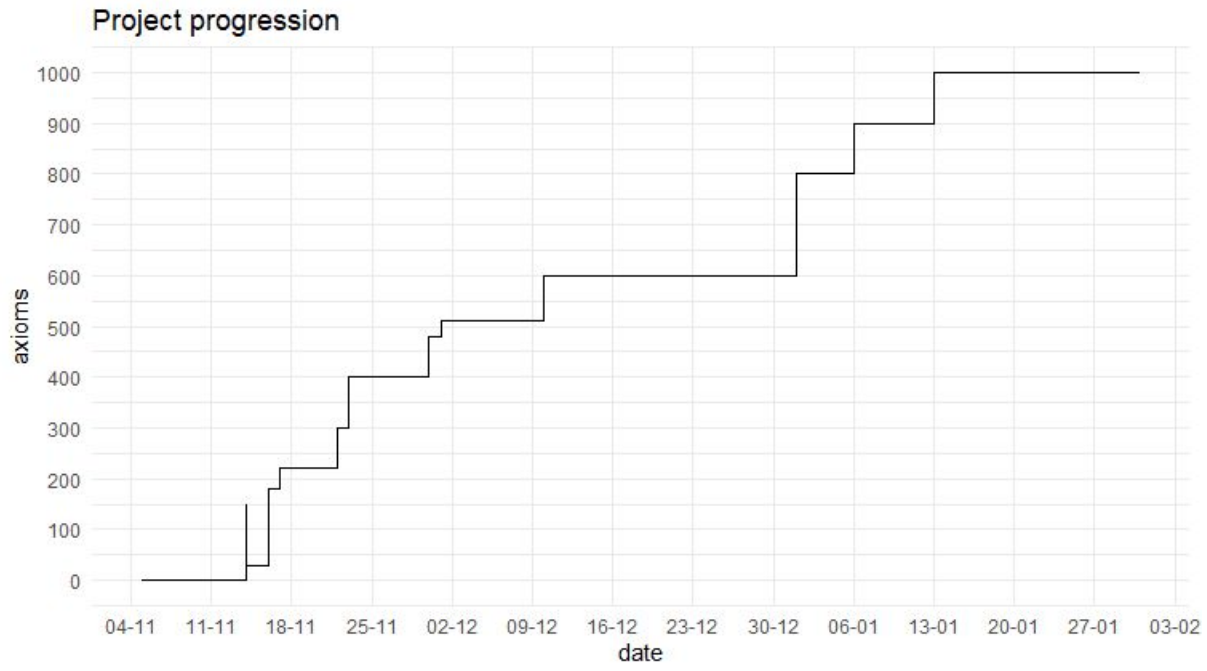
- 1000 axioms
- Known considerations noted and described

Final version

- Ontology metrics (axiom types by count) with comments
- Rendered images - class graph, etc.
- HTML documentation generated from ontology file, including annotations (voaf, dublin core, rdfs)
- At least 10 examples of Individuals described using the ontology (preferably in Turtle format) – the individuals should NOT be included in the ontology file, and will not count towards the minimum axiom count
- Short instructions about using the ontology to describe Individuals
- Short instructions about extending the ontology (if applicable, i.e. if ontology contains stub classes)

Project progression

Overall, the development has been executed according to schedule with rather irrelevant mismatches between the requirements and the reality. The following chart represents number of axioms of the ontology in time.



General description

The aim

The aim of this project is to create an ontology of video games. It should contain all important information, relations and dependencies to fulfil its purpose, being an advanced search tool for gamers.

The perspective

The ontology is supposed to serve gamers and in particular people looking for a game. Therefore, the main espoused perspective is the view of a gamer seeking a game to play. However, it is not designed to be an equivalent to a search engine for people who know what they are looking for. The information buried in the ontology

should allow finding games matching the “taste” of the gamer.

The “taste” of the gamer

The process of looking for a game is similar to chasing the taste of a meal that one once ate and totally loved. One may not know what actually was behind the fact that they enjoyed it so much, but they definitely want to find something similar.

There are many aspects that may be useful while looking for a game that one does not know about. That means, looking for a game that is, for example, “similar” to another game. The “taste” is not only limited to game-with-game similarity. One may also look for a game that has a resembling vibe to a movie, or a book or perhaps a particular feature that sets the whole climate in some way. Perhaps one may also look for games created by a particular company. This is what the Computer Games Ontology aims to do - help find something similar to what someone already enjoyed.

The practical requirements

One may also claim that we want to find something of a particular “taste” only if we can actually eat it.

The ontology is also contains information about the requirements of the game - both the very technical like RAM or processor requirement but also whether a game is published (sold) in a specific country or what rating or age restrictions it has.

The repository

The progress of the ontology is saved in the repository hosted under the following link:
https://github.com/JanSawicki/cgo
The access is private to avoid plagiarism. The author suggests direct contact to acquire the access.

Data sources

Although the following list describes various sources of knowledge about games, video games, their taxonomy, categorization and other related phenomena, the most fundamental source was the self experience and own knowledge of the creator.

Data sources:

- Own experience
- https://en.wikipedia.org/wiki/List_of_video_game_genres
- https://en.wikipedia.org/wiki/Fictional_universe
- https://en.wikipedia.org/wiki/List_of_video_game_developers
- https://en.wikipedia.org/wiki/List_of_video_game_publishers
- https://www.gamepressure.com/games/
- https://store.steampowered.com/games/
- https://en.m.wikipedia.org/wiki/List_of_fictional_universes_in_literature
- https://en.m.wikipedia.org/wiki/Glossary_of_video_game_terms
- https://www.metacritic.com/game
- https://gamefaqs.gamespot.com/games/rankings
- https://en.wikipedia.org/wiki/Video_game_content_rating_system#Rating_systems
- http://dbpedia.org/page/PC_game
- http://dbpedia.org/ontology/VideoGame
- http://dbpedia.org/page/Game

From scratch

The main question to answer when creating an ontology is whether or not it is supposed to be a completely new project or an extension of an already existing one. After a thorough research it is concluded that there are not many game ontologies (see the list below) and there is none that would fit even vaguely the purpose of this ontology. Therefore, the video games ontology will be created from the very beginning.

Known ontologies:

- http://vocab.linkeddata.es/vgo/
  A small ontology with the focus on the player, describing the gaming world, in terms of e.g. achievements, fandom, difficulty etc.
- https://lov.linkeddata.es/dataset/lov/vocabs/game
  Very small ontology related to game states.

Note that although the ontology was created from scratch, it is supposed to have “connections” with other ontologies. It should be treated as rather a module (or in gaming world a DLC) than an extension.

Technology stack

The following list describes the main technologies used in the process of ontology creation.

Technology	Details
Git	Used for version control
GitHub	Used for hosting the Git repository
Protégé	Used for ontology creation and edition
Protégé plugins: Ontology Debugger, OntoGraf, Ontology visualization, SWRLTab, HermiT reasonerMatrix, OWLViz	Used to enhance the work with Protégé and create visualization for final artifacts of the project.
VOWL, OWLGrEd	Used for ontology visualization

Architecture

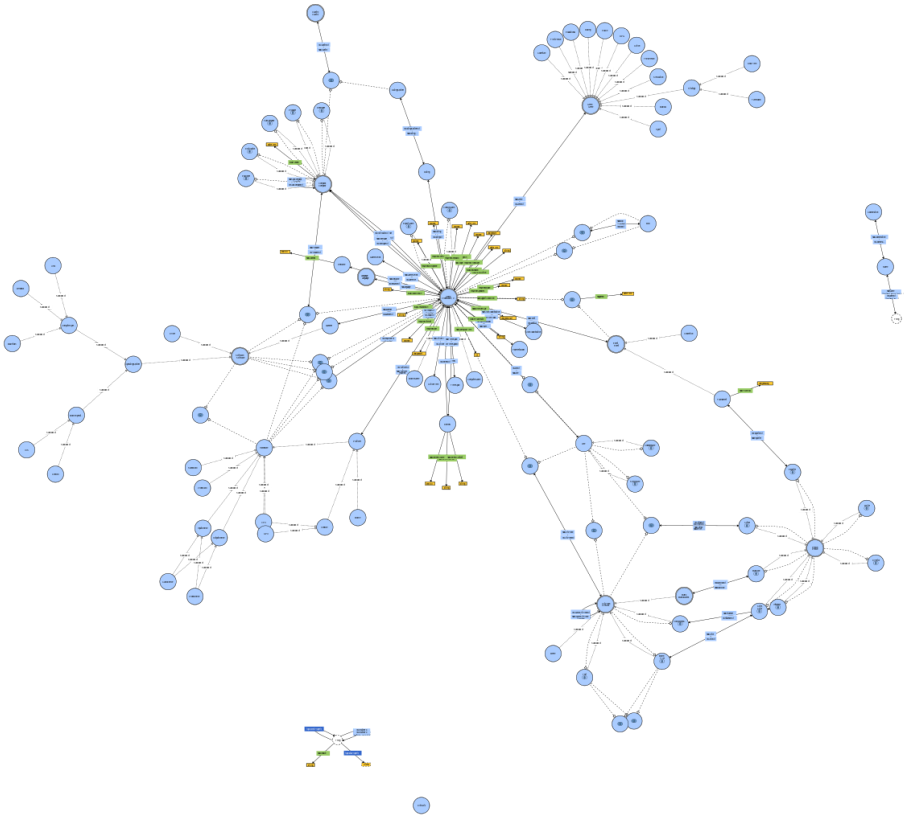
All of the classes of the ontology orbit around the main class - Game. It is the central class and the main focus of the ontology. However, it is not the only focus. As intended the ontology is supposed to have a connection to the “outer world”. This connection is realized via a.o. the ArtPiece class. This and some other very vague terms and allow the ontology to be easily connected to other ontologies - which its very comfortable byproduct feature. For example, the class Person allows to link Computer Games Ontology with e.g. FOAF[1].

[1] http://www.foaf-project.org/

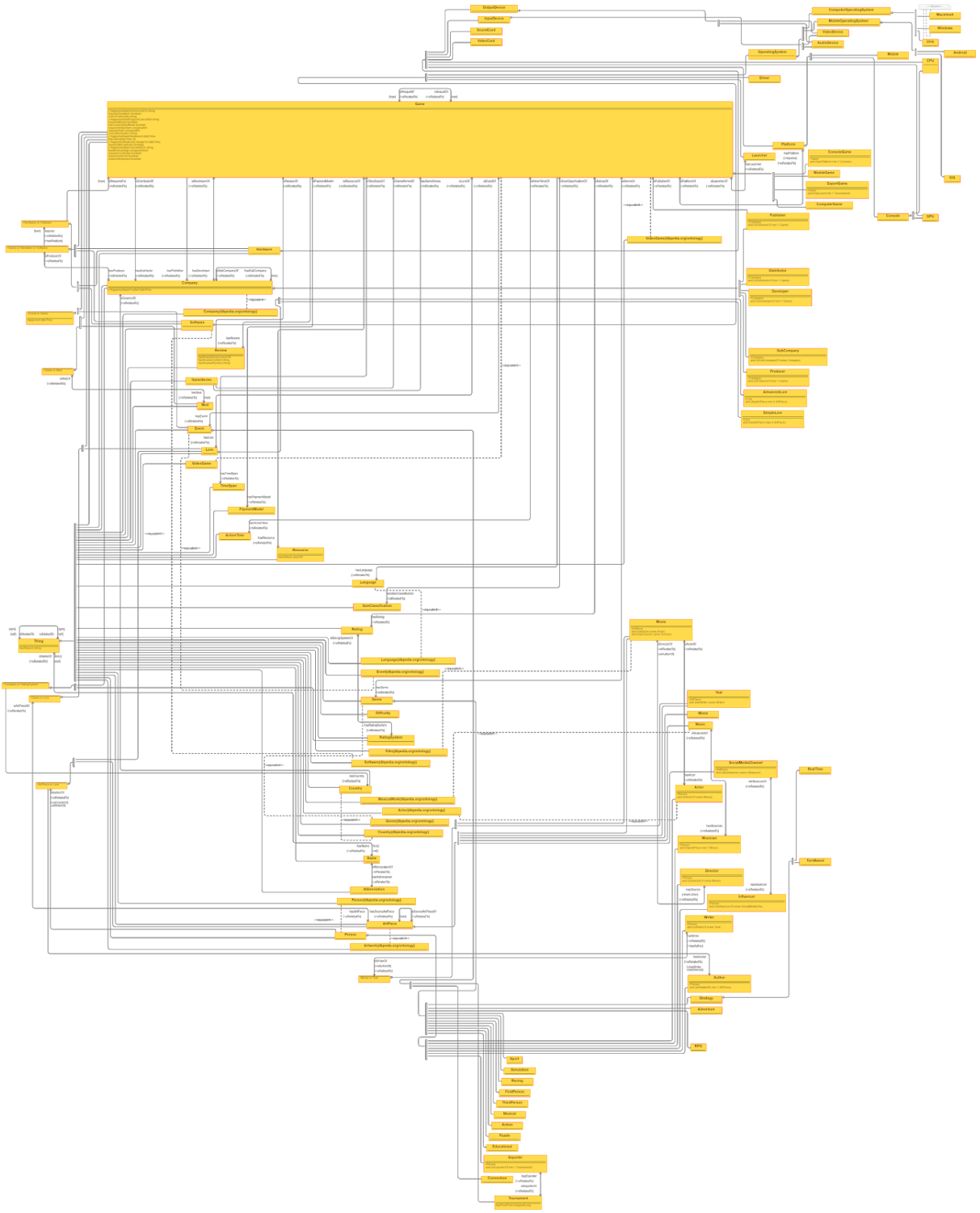
Visualization

The graphs below represents the Ontology as a network-like object. Due to their illegibility, the graphs are also attached in a higher resolution/vector graphic form.

Visualization with WebVOWL:



Visualization with OWLGrEd:



Metrics

The following table contains axioms types by count with explanations (if necessary).

Axiom type	Count	Comment
Total	1032	The number may be misleading because it contains the axioms that are technically not part of the ontology but are necessary to allow reasoning from the Protégé/HermiT point of view. Those axioms are all classes mostly from DBpedia (e.g. dbo:VideoGame), that are used to “link” (using EquivalentTo) to existing classes (e.g. cgo:Game).
Logical	436	The logic in ontology has been the biggest focus. The creator assumed the analogy that “it is better to define 100 relations between two people than define 100 people”.
Declaration	211	The number of declaration axiom is the direct result of all other axioms counts.
Class	98	Second biggest focus was to cover a big part of the domain or at least the main aspects according to the general purpose. Hence the number of declarations.
Object Property	71	Note that most of object properties have an analogous inverse property.
Data Property	27	It would be a lot simpler to add many data properties, but the authors wanted to focus on relations than a pure declarative description. The point was to create a reasonable ontology, not a dictionary with long definitions.

The following list contains some of the architectural remarks:

- Other ontology extensions

  Due to lack of a satisfying state-of-the-art basis, the ontology was developed from scratch. However, it still is a kind of extension of other ontologies. It contains many classes (e.g. Person, Game, Movie) which equivalents are already present in existing ontologies (e.g. DBpedia). Those classes that have equivalents in external ontologies have been marked using the “Equivalent

To" relation (e.g. cgo:Person isEquivalentTo db:Person).

- Object properties
    - Naming

      Nearly all of the object properties follow the naming convention of "hasXxx" and isXxxOf". The relations described with properties are rather easy to understand even by just looking at their domains and ranges. Such convention was introduced to make it easier for the user to "guess" what the relation is called when they know what they are looking for.

      Note, however, that there are some exceptions, like the object property requires
    - Inverse properties

      Nearly all of the object properties have their inverses (e.g. hasGenre and isGenreOf). One may say that that one in such pair is redundant and it can be omitted without loss of ontological knowledge. However, the author is aware of how difficult it may sometimes be to create a "inverse of a query" and provides a simple inverse properties to make the ontology easier to use with e.g. SPARQL[2].

Individuals

The process of describing individuals can be done in two ways. Be advised that one of the artifacts of the project is the ontology with some exemplary individuals described in Turtle format[3].

DFS approach

The advised way of adding new individuals is to perform a depth-first-search-like approach. A semantist[4] should start with a game they want to describe. A game is always the starting point. They should focus on one aspect of it, e.g.: its esport

[2] https://www.w3.org/2001/sw/wiki/SPARQL
[3] https://www.w3.org/TR/turtle/
[4] The word "semantist" (a semantic scientist) has been invented independently by the ontology author and (provided it is legally licente) all rights to it are reserved.

status. Assuming it is an esport game, they should focus on one aspect of esport scene of this game, e.g. a player. They should again start with describing one aspect of the player, e.g. the tournaments (of the described game) they played in. Describing a tournament should start again with one aspect of it and so on. The path ends when the description of something does not need anything else than a simple property to be described, e.g. a tournament name which is just a string. A string does not need any further description (does not depend on some class to be described etc.). After finding the “leaf”, the semantist should go up “one level”, i.e. from the name to the tournament and again describe one aspect of it, e.g. when it happened.

BFS approach

The other, less advised approach is to act analogically to how it is described in DFS approach, but instead of going deeper first, the semantist should go wider. First describe all games, then for each game describe all its esport players, for each player all their tournaments then for each tournament their names and so on.

Advice

This second (BFS) approach is less advised due to the following cause and effect. A semantist stops describing individuals in an ontology when one of the following happens (logical OR):

- the ontology is complete (never)
- some time expires
- they die

Let’s assume the second - finishing after 2 hours. In BFS approach a semantist has created basically a very long list of classes representing many games. In DFS approach they have more or less described one whole game maybe two. BFS result has no semantic value, because it is just a list and HermiT[5] is very sad, because it cannot reason anything from it. DFS served a better purpose in creating something that, although not complete, allows some reasoning.

[5] http://www.hermit-reasoner.com/

Considerations

During the development there were many unclear aspects - the considerations that had to be dealt with. This section sums up the main design decisions and the justifications for them.

- Genre

  One of the possible approaches was to define all genres as classes, but it would kill the purpose of instances of the generes. Therefore, the instances of the Genre class are the particular genres that can be a subset of the main genres (the subclasses of Genre class).
  e.g.: An instance “MMORPG” belongs to the class RPG (a subclass of class Genre)
- Name

  There were various ways to represent names in the ontology. All in all they play a key role in finding the thing a gamer is looking for. The final solution was to define a class Name which can be related to an object property isNameOf (or hasName inversely) to anything (owl:Thing). An instance of Name is suppose to have a rdfs:label with the appropriate name. This way the Instances all nicely displaying in Protégé and it is easier to not make the mistake of duplicating something (because if it exists the Name instance already exists).
  Although the justifications persuades the author and their overly ordering obsession (OCD), they have doubts if it is truly the best solution.
  Note that cgo:Name is not an equivalent for dbo:Name, because it is a name of any entity - not necessarily a name (first name and last name) of a human.
- Abbreviation

  Shortcuts and abbreviations are very popular in gaming environment .
  Therefore an Abbreviation is represented as a separate class in order to connect names (instances of class Name) to its Abbreviation.
  e.g.: Instance of class Abbreviation “MMORPG” is connected to Name “Massive Multiplayer Online Role-Playing Game” with object property

isAbbrevationOf.

- Company

  The split of class Company into subclasses (Producer, Developer etc.) could be avoided, but is necessary to reflect the real world more appropriately. A company (usually called a studio) which produces a game (i.e. pays for its creation) may have very little influence on how the game looks like (an important aspect for a gamer). This is dependent mostly on the developer. On the other hand, a particular production studio may be a good indicator of the game “size” or production “scale”. Both of those (and all other derived) pieces of information may be useful to a gamer in their gaming research. Therefore, the Company class has its subclasses.

- Price

  Price or in general all financial aspects of the gaming industry has been completely omitted in the ontology on purpose. As a time-dependant fast changing phenomena the financial sector is not appropriate for semantic representation in ontologies.

- Language, Country, etc.

  Concepts such as language or country can be represented with a simple object property (such as “hasLanguage”) with domain xsd:string. However, that approach may lead to ambiguities such as “US” vs “United States” vs “United States of America”. Moreover, the approach with classes and instances allows extending and complying with outer well known state-of-the-art ontologies (e.g. DBpedia). Therefore classes such as Language and Country have been introduced.

- RatingSystem

  It would be possible to introduce a unified rating system (e.g. Entertainment Software Rating Board). However, in the ontology each rating system is represented with an instance of the class RatingSystem in order to allow more rating systems to be included and make the ontology more general and international.

- ComputerGame, ConsoleGame, MobileGame

Due to the nature of this distinction the logic behind the rules for a Game to be a ComputerGame, ConsoleGame, MobileGame is not analogical. That means that to be a ComputerGame or MobileGame, a Game has to be in relation (have object property hasOperatingSystem) with a ComputerOperatingSystem or MobileOperatingSystem accordingly. For a Game to be a ConsoleGame, however, it has to have an object property hasPlatform with a Console. This difference is due to the fact that rarely does anyone speak about the “operating system” of a console, because a particular console can only have a particular distribution of a particular operating system. On the other hand computers and phones (as physical devices) may have different systems and may even change in time.

The list of consideration can go on depending on how many details a user needs. The authors suggest direct contact in case of further questions.

Deliverables

The project artifacts are:

- OWL file with the ontology
- OWL file (in Turtle format) with the ontology and some exemplary individuals
- HTML documentation (generated with Protégé)
- SVG of WebVOWL visualization and high resolution PNG of OWLGrEd visualization
- DOC and PDF documentation (this very file)

Future works

It has been created so that it can be extended or “built on top of”. As any other topic, the topic of computer games may have much more aspects that have been represented. The examples of possible extensions regard but are not limited to:

- Number of users playing a game (e.g. on release)
- Number of main and side quests for games that it applies to (e.g. RPG games)
- Friendliness and toxicity of the game’s community
- Split screen availability
- Launcher account sharing (e.g. family sharing)
- Multiplayer details (e.g. number of players)
- Toys market related to a game (an aspect perhaps important for a parent looking for a game for their child)

Asked about the completeness of the ontology, the authors wishes to finished with the semantic mantra:

The ontology is as complete as it is not complete.