

СОФИЙСКИ УНИВЕРСИТЕТ „СВ. КЛИМЕНТ ОХРИДСКИ“
Факултет по математика и информатика

КУРСОВ ПРОЕКТ
по дисциплина: „Представяне и моделиране на знания“

ТЕМА:
NBA БАСКЕТБОЛНА ОНТОЛОГИЯ
(Семантично моделиране на играчи, отбори и статистически показатели)

Изготвил:
Име: Явор Йорданов Чамов
Фак. Номер: 4MI3400787
Специалност: Изкуствен интелект

1. Въведение в предметната област

Баскетболът, и по-конкретно Националната баскетболна асоциация (NBA), е сфера, генерираща огромно количество данни. Отвъд простата статистика (точки, борби, асистенции), съвременният анализ изисква разбиране на сложни взаимовръзки – договори, роля на играча в отбора, история на трансферите и потенциал за развитие.

Традиционните релационни бази данни са отлични за съхранение на тези данни, но изпитват затруднения при извличането на скрито знание чрез логически извод. Например, заявка за „намиране на ветеран, който преследва титла в силен отбор, но никога не е печелил такава“ изисква сложни JOIN операции и предварително дефинирани флагове.

Целта на настоящия проект е да създаде формална онтология (Knowledge Base), която не просто описва данните, а позволява семантично търсене и автоматична класификация на играчи и отбори.

Проблемът, който се решава, е автоматизирането на статистиката и категоризацията. Чрез средствата на дескриптивните логики (DL), системата може сама да определи дали даден играч е например „Елитен стрелец“, базирайки се единствено на неговите показатели.

2. Преглед и анализ на решенията

В процеса на разработка бяха анализирани съществуващи подходи за спортни онтологии. Повечето достъпни решения (като BBC Sport Ontology) са твърде общи – те описват събития и резултати, но липсва дълбочина в профилирането на атлетите.

Като част от проучването бе използван и **интелигентен агент (LLM)** за генериране на идеи. Агентът предложи интересни концепции за сложни класове, но анализът показва, че генерираният OWL синтаксис често е неконсистентен или логически неиздържан (имаше циклични зависимости и липсващи връзки).

Силни и слаби страни:

- *Генерирани решения:* Добри за идеен проект и брейнсторминг на категории (напр. идеята за VeteranRingChaser). Слаби при спазване на строгите ограничения на owl:disjointWith или функционалните свойства.
- *Настоящото решение:* Комбинира идеите от AI с ръчна валидация и строга йерархия, гарантираща консистентност (Satisfiability) при пускане на Reasoner.

3. Описание на данните

Използваните данни са представителна извадка от сезон 2024-2025 на NBA, както и исторически данни за активни легенди.

Основните източници са Basketball-Reference и [NBA.com](https://www.nba.com).

Данните включват:

- **Демография:** Възраст, години опит.
- **Статистика:** Точки (PPG), борби (RPG), асистенции (APG), процент стрелба от тройката (3P%).
- **Структура:** Отбори, конференции (Изток/Запад), дивизии.
- **Постижения:** Награди (MVP, Шампионски титли).

В онтологията данните не се въвеждат като статични етикети, а като числови стойности (xsd:integer, xsd:decimal), върху които се прилагат логически рестрикции.

4. Описание на онтологията

Онтологията е проектирана на езика **OWL 2 DL**. Тя се състои от таксономия на класовете, свойства (Object & Data properties) и множество от дефинирани класове за логически извод.

4.1. Йерархия на класовете (Taxonomy)

Клас (Class)	Родителски клас (Subclass of)	Описание
BasketballProfessional	owl:Thing	Общ клас за всички професионалисти в спорта.
Player	BasketballProfessional	Активни състезатели.
Coach	BasketballProfessional	Треньори.
Organization	owl:Thing	Юридически лица и организации.
Team	Organization	Баскетболен отбор.
NBATeam	Team	Отбор, играещ в NBA лигата.
EasternConferenceTeam	NBATeam	Отбор от Източната конференция.

WesternConferenceTeam	NBATeam	Отбор от Западната конференция.
League	Organization	Спортна лига (напр. NBA).
Achievement	owl:Thing	Постижения и трофеи.
Award	Achievement	Индивидуални награди.
Championship	Achievement	Отборни титли.
Position	owl:Thing	Изброим клас (Enumerated Class).

Йерархията на класовете (Taxonomy) в онтологията е проектирана така, че да разграничава ясно основните субекти и обекти в предметната област. В основата на структурата, наследявайки корена owl:Thing, стоят четири главни концепции. Човешките участници са обобщени в класа BasketballProfessional, който се специализира в два основни подкласа: Player (за активни състезатели) и Coach (за треньори), позволявайки споделяне на общи атрибути. Организационните единици са представени чрез класа Organization, като специално внимание е обърнато на структурата на отборите (Team); те са детайлизирани до NBATeam, който от своя страна се разклонява на EasternConferenceTeam и WesternConferenceTeam, за да отрази географското разделение в лигата. За моделиране на успехите е въведен абстрактният клас Achievement, обединяващ индивидуалните награди (Award) и отборните титли (Championship), а игралните позиции са дефинирани като отделен изброим клас Position за строга типизация на ролите на терена.

4.2. Свойства (Properties)

Свойство (Property)	Домейн (Domain)	Рейндж (Range)	Характеристики / Обратно свойство
playsFor	Player	Team	InverseOf: hasRosterPlayer
hasRosterPlayer	Team	Player	InverseOf: playsFor
draftedBy	Player	Team	Функционално (всеки е драфтен веднъж).
hasWon	BasketballProfessional	Achievement	Транзитивно (по желание).
hasWonAward	BasketballProfessional	Award	SubPropertyOf: hasWon

Секцията с обектни свойства (Object Properties) дефинира семантичните връзки между инстанциите от различните класове. Основната релация в онтологията е връзката между играч и отбор, която е моделирана като двупосочна чрез двойката инверсни свойства playsFor (играе за) и hasRosterPlayer (има играч в състава). Това гарантира, че ако заявим, че даден състезател е част от даден отбор, системата автоматично ще може да изведе, че отборът притежава този състезател. За моделиране на историческия произход на играчите се използва свойството draftedBy. То е дефинирано като функционално (Functional), тъй като в контекста на NBA драфта, един играч може да бъде

избран първоначално само от един единствен отбор. Постиженията са организирани йерархично. Общото свойство `hasWon` свързва всеки баскетболен професионалист (както играчи, така и треньори) с класа `Achievement`. За по-голяма детайлност е въведено специализираното свойство `hasWonAward` (под-свойство на `hasWon`), което конкретизира връзката само към индивидуалните награди (`Award`). Този подход позволява гъвкавост при справките – можем да търсим всички успехи на дадена личност или да филтрираме само конкретни типове награди.

Свойство (Property)	Домейн (Domain)	Рейндж (Range)	Характеристики
<code>age</code>	<code>Player</code>	<code>xsd:integer</code>	Functional
<code>pointsPerGame</code>	<code>Player</code>	<code>xsd:decimal</code>	Functional
<code>assistsPerGame</code>	<code>Player</code>	<code>xsd:decimal</code>	Functional
<code>reboundsPerGame</code>	<code>Player</code>	<code>xsd:decimal</code>	Functional
<code>threePointPercentage</code>	<code>Player</code>	<code>xsd:decimal</code>	Functional (0.00 - 1.00)
<code>winPercentage</code>	<code>Team</code>	<code>xsd:decimal</code>	Functional

За да се осигури възможност за извършване на логически извод върху количествени характеристики, онтологията дефинира набор от свойства на данните (Data Properties). По-голямата част от тях са съсредоточени върху класа `Player`, описвайки физически и статистически атрибути. Свойството `age` приема целочислени стойности (`xsd:integer`) за възрастта на играча. Ключовите показатели за представянето на терена — `pointsPerGame` (точки), `assistsPerGame` (асистенции) и `reboundsPerGame` (борби) — са

дефинирани с тип `xsd:decimal`, което позволява записването на осреднени стойности с плаваща запетая. Специфично внимание е отделено на ефективността на стрелбата чрез свойството `threePointPercentage` (процент стрелба от тройката), което също е десетично число в интервала от 0.00 до 1.00. За класа `Team` е въведено свойството `winPercentage`, отразяващо успеваемостта на отбора през сезона. Всички изброени свойства са маркирани като `Functional`, което налага логическото ограничение, че един играч или отбор може да има само една единствена стойност за даден показател в рамките на модела (напр. играчът не може да е едновременно на 25 и на 30 години).

4.3. Логически дефиниции и извод (Defined Classes)

Това е същинската част на проекта. Вместо ръчно да класифицираме играчите, ние дефинираме правила, а `Reasoner`-ът извършва класификацията.

Дефиниран Клас (Defined Class)	Логическо изражение (Equivalent To / OWL Syntax)	Значение на естествен език
Position	<code>{PointGuard, ShootingGuard, SmallForward, PowerForward, Center}</code>	Класът се състои точно от тези 5 индивида.
EliteShooter	<code>Player \sqcap \exists threePointPercentage \geq 0.40]</code>	Играч с успеваемост от тройката над 40%.
VeteranPlayer	<code>Player \sqcap \exists age \geq 32]</code>	Играч на 32 или повече години.
TripleDoublePlayer	<code>Player \sqcap \exists pointsPerGame \geq 10] \sqcap \exists reboundsPerGame \geq 10] \sqcap \exists assistsPerGame \geq 10]</code>	Играч с двуцифрени показатели в 3 категории.

Champion	$\text{Player} \sqcap \exists \text{hasWon.Championship}$	Играч, който е печелил титла.
PureGuard	$\text{Player} \sqcap \forall \text{hasPosition}.\{\text{PointGuard}, \text{ShootingGuard}\}$	Играч, който играе само на гардови позиции.
CelticsPlayer	$\text{Player} \sqcap (\text{playsFor value BostonCeltics})$	Играч на Бостън Селтикс (Value restriction).
ContenderTeam	$\text{NBATeam} \sqcap \exists \text{winPercentage}[\geq 0.60]$	Силен отбор с над 60% победи.
VeteranRingChaser	$\text{VeteranPlayer} \sqcap \exists \text{playsFor.ContenderTeam} \sqcap \neg \text{Champion}$	Ветеран в силен отбор, който не е шампион.

Най-съществената функционална част от онтологията са логическите дефиниции (Defined Classes), които позволяват на машината за логически извод (Reasoner) да извършва автоматична класификация на инстанциите. Вместо статично да приписваме етикети на играчите, са създадени класове с необходими и достатъчни условия (Equivalent To), базирани на дескриптивни логики.

Ограничения на стойности (Data Property Restrictions): Класове като EliteShooter и VeteranPlayer използват математически оператори (\geq), за да филтрират играчи на базата на техните статистически показатели и възраст.

Конюнкция (Intersection): Класът TripleDoublePlayer демонстрира обединяването на три отделни условия (за точки, борби и асистенции) в една логическа структура чрез оператора за сечение.

Квантори и Отрицание: За по-сложни зависимости са използвани универсални квантори

(напр. PureGuard, използващ \forall), които ограничават обхвата на възможните връзки, както и логическо отрицание.

Вложени дефиниции: Най-комплексният пример е класът VeteranRingChaser. Той стъпва върху дефиницията за VeteranPlayer, проверява връзката му с отбор, който отговаря на условията за ContenderTeam (над 60% победи), и изрично изисква играчът да не е шампион (\neg Champion).

4.4. Индивиди

За валидация е въведен индивидът StephenCurry.

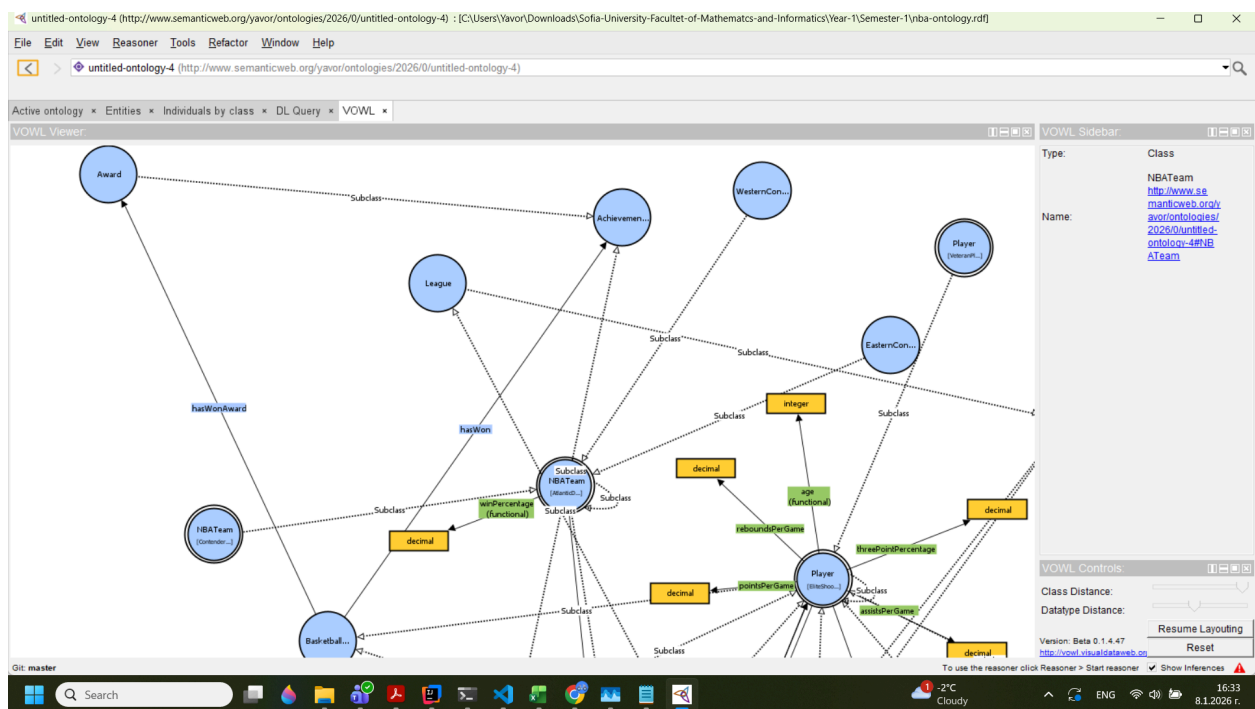
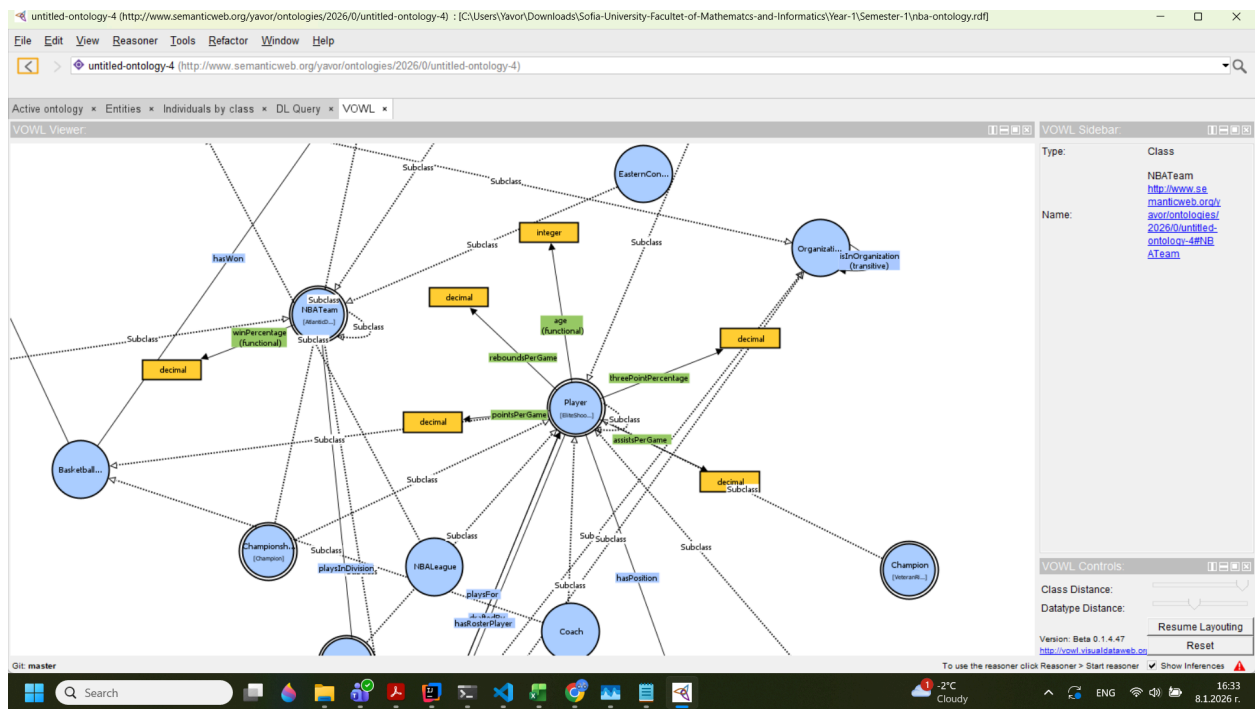
Индивид (Instance)	Тип (Type Asserted)	Стойности на свойства (Property Assertions)
StephenCurry	Player	hasPosition: PointGuard playsFor: GoldenStateWarriors draftedBy: GoldenStateWarriors hasWon: NBACHampionship

		age: 35 threePointPercentage: 0.42
GoldenStateWarriors	WesternConferenceTeam	winPercentage: 0.55 hasRosterPlayer: StephenCurry
BostonCeltics	EasternConferenceTeam	winPercentage: 0.78

Разделът с индивидите (Individuals) служи за практически тест на онтологията (A-Box) и валидиране на дефинираните по-рано правила. В таблицата са представени ключови екземпляри, чрез които се демонстрира работата на машината за логически извод. Централен пример е индивидът StephenCurry, който е експлицитно дефиниран само като Player. Чрез обектни свойства той е свързан с позицията PointGuard и отбора GoldenStateWarriors.

Въведените стойности на свойствата на данните, като възраст (35) и процент стрелба от тройката (0.42), служат като входни параметри за автоматичната класификация. Именно наличието на тези конкретни стойности позволява на Reasoner-а да "засече" Curry и автоматично да го причисли към динамичните класове VeteranPlayer и EliteShooter. Аналогично, за индивидите на отборите (GoldenStateWarriors и BostonCeltics) е попълнен атрибутът winPercentage, който е решаващият критерий за класифицирането им като ContenderTeam (отбори претенденти).

5.1. Визуализация в Protege Desktop



6. Демонстрация с работещ Reasoner HerMiT

The screenshot shows the Protégé interface for the ontology 'untitled-ontology-4'. The 'Individuals' tab is active, showing a list of individuals under the class 'StephenCurry'. The 'Annotations' tab is also visible, showing the following annotations for StephenCurry:

- hasPosition: PointGuard
- hasWon: NBACHampionship
- playsFor: GoldenStateWarriors

The 'Property assertions' tab shows the following data property assertions for StephenCurry:

- age: 35
- pointsPerGame: 28.4
- threePointPercentage: 0.42

The screenshot shows the Protégé interface for the ontology 'untitled-ontology-4'. The 'Individuals' tab is active, showing a list of individuals under the class 'NikolaJokic'. The 'Annotations' tab is also visible, showing the following annotations for NikolaJokic:

- draftedBy: DenverNuggets
- hasPosition: Center
- hasWon: NBACHampionship
- playsFor: DenverNuggets

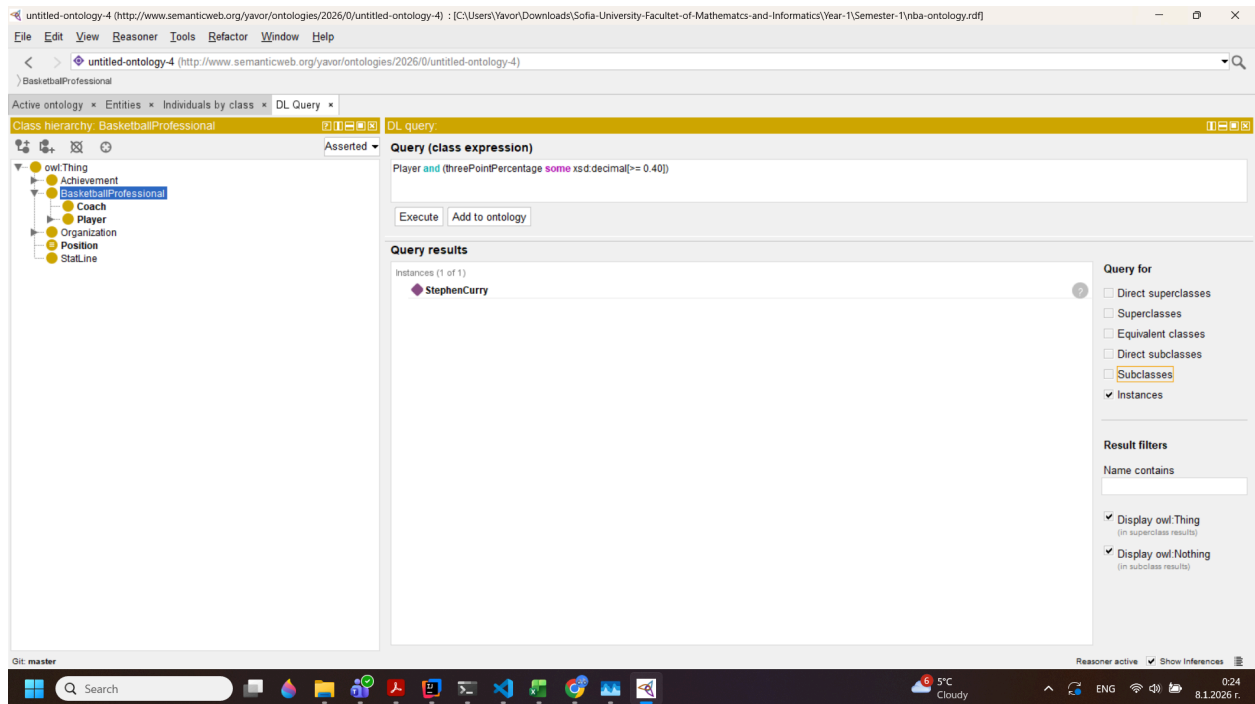
The 'Property assertions' tab shows the following data property assertions for NikolaJokic:

- age: 29
- assistsPerGame: 10.2
- pointsPerGame: 26.4
- reboundsPerGame: 12.1
- threePointPercentage: 0.36

7. Демонстрация на изпълнение на DL заявки

The screenshot shows the Protégé interface with the 'BasketballProfessional' ontology active. The 'Class hierarchy' on the left lists classes: owl:Thing, Achievement, BasketballProfessional, Coach, Player, Organization, Position, and StaffLine. The 'DL query' panel on the right shows a query for 'VeteranPlayer'. The 'Query results' section displays one instance: StephenCurry. The 'Query for' section on the right has checkboxes for 'Direct superclasses', 'Superclasses', 'Equivalent classes', 'Direct subclasses', 'Subclasses', and 'Instances' (checked). The 'Result filters' section has a 'Name contains' field and checkboxes for 'Display owl:Thing' and 'Display owl:Nothing'.

The screenshot shows the Protégé interface with the 'BasketballProfessional' ontology active. The 'Class hierarchy' on the left lists classes: owl:Thing, Achievement, BasketballProfessional, Coach, Player, Organization, Position, and StaffLine. The 'DL query' panel on the right shows a query for 'EliteShooter'. The 'Query results' section displays one instance: StephenCurry. The 'Query for' section on the right has checkboxes for 'Direct superclasses', 'Superclasses', 'Equivalent classes', 'Direct subclasses', 'Subclasses', and 'Instances' (checked). The 'Result filters' section has a 'Name contains' field and checkboxes for 'Display owl:Thing' and 'Display owl:Nothing'.



Примерни DL Queries:

VeteranPlayer and not Champion

Player and (playsFor some (NBATeam and (playsInDivision value PacificDivision)))

Player and (draftedBy value GoldenStateWarriors)

Team and (hasRosterPlayer value StephenCurry)

Player and (hasPosition min 2 Position)

Player

and (hasPosition value Center)

and (age some xsd:integer[< 23])

and (reboundsPerGame some xsd:decimal[>= 8.0])

VeteranPlayer and not Champion

Player

and (hasPosition value PointGuard or hasPosition value ShootingGuard)

and (playsFor some EasternConferenceTeam)

8. Използвани технологии и ресурси

Проектът е реализиран чрез следните инструменти:

1. **Protégé 5.6.3:** Основна среда за разработка на онтологии.
2. **OWL API:** За сериализация във формат RDF/XML.
3. **HermiT Reasoner:** Използван за проверка на консистентност и класификация на йерархията. Избран е заради бързината си при работа с Data Properties.
4. **WebVOWL:** За визуализация на графа на зависимостите.

6. Заключение и насоки за бъдещо развитие

Разработената онтология успешно демонстрира как знанията в спорта могат да бъдат формализирани. Основното предимство пред стандартните бази данни е възможността за дефиниране на абстрактни понятия (като „Елитен стрелец“), които автоматично се адаптират при промяна на данните на играча.

Бъдещо развитие:

Тази разработка поставя основите за по-мощен проект в курса по „Бази от знания“. Възможните разширения включват:

1. **Темпорални данни:** Добавяне на валидност на фактите във времето (напр. playsFor с атрибут season=2015). Това би позволило проследяване на кариерата.
2. **Автоматично попълване:** Разработване на Python скрипт, който чрез owlready2 библиотеката извлича данни от *NBA API* в реално време и ги влива в онтологията като триплети.
3. **Разширени метрики:** Включване на "Advanced Analytics" (PER, TrueShooting%), за да се откриват подценени играчи (Moneyball подход).

7. Източници

1. *NBA Stats Reference*. (2025). Retrieved from <https://www.basketball-reference.com/>
2. W3C OWL 2 Web Ontology Language Document Overview. (2012). <https://www.w3.org/TR/owl2-overview/>