



Department of Information and Computer Science

National University of Mongolia

KGE-MN 2025 - Knowledge Graph utilizing open data from the State Registration of Legal Entities of Mongolia

Document Data:

December 15, 2025

Reference Persons:

Chinzorigt.G, Enkhbayasgalan.E

Ulaanbaatar, Mongolia

This report is licensed under CC-BY-SA-NC and describes the work and results of the Knowledge Graph Engineering course (ICSI500) offered by the Department of Information and Computer Science at the National University of Mongolia. This course is initially developed in the University of Trento, Italy and its KnowDive research group.



Contents

| | | |
|----------|---|----------|
| 1 | Introduction | 1 |
| 1.1 | Project Overview | 1 |
| 1.2 | Motivation and Significance | 1 |
| 1.3 | Document Structure | 2 |
| 2 | Domain of Interest (DoI) | 2 |
| 2.1 | Spatial Boundaries | 3 |
| 2.1.1 | Primary Geographic Scope | 3 |
| 2.1.2 | International Dimension | 3 |
| 2.1.3 | Spatial Boundary Justification | 3 |
| 2.2 | Temporal Boundaries | 4 |
| 2.2.1 | Historical Scope | 4 |
| 2.2.2 | Temporal Granularity | 4 |
| 2.2.3 | Temporal Boundary Justification | 5 |
| 2.3 | Domain Boundary Summary | 5 |
| 2.4 | Out of Scope | 5 |
| 3 | Project Development | 6 |
| 3.1 | Data Production | 6 |
| 4 | Initial Resources | 6 |
| 5 | Purpose Formalization | 7 |
| 5.1 | Scenarios Definition | 7 |
| 5.1.1 | Scenario 1: Corporate Ownership Investigation | 7 |
| 5.1.2 | Scenario 2: Due Diligence for Business Partnerships | 7 |
| 5.1.3 | Scenario 3: Stock Market Investment Analysis | 7 |
| 5.1.4 | Scenario 4: Regulatory Compliance Monitoring | 8 |
| 5.1.5 | Scenario 5: Anti-Money Laundering (AML) Analysis | 8 |
| 5.2 | Personas | 8 |
| 5.2.1 | Persona 1: Investigator Batbold | 8 |
| 5.2.2 | Persona 2: Business Analyst Oyungerel | 9 |
| 5.2.3 | Persona 3: Portfolio Manager Enkhjargal | 9 |
| 5.2.4 | Persona 4: Compliance Officer Munkhbat | 9 |
| 5.3 | Competency Questions (CQs) | 10 |
| 5.3.1 | Entity Identification and Basic Information | 10 |
| 5.3.2 | Ownership and Shareholder Information | 10 |
| 5.3.3 | Management and Representation Authority | 10 |
| 5.3.4 | Ultimate Beneficial Ownership | 11 |
| 5.3.5 | Business Activities | 11 |
| 5.3.6 | Corporate Restructuring and History | 11 |
| 5.3.7 | Network Analysis and Cross-Entity Queries | 11 |

| | | |
|-----------|--|-----------|
| 5.4 | Concepts Identification | 12 |
| 5.4.1 | High Popularity Concepts (Core Entities) | 12 |
| 5.4.2 | Medium Popularity Concepts (Supporting Entities) | 12 |
| 5.4.3 | Low Popularity Concepts (Contextual Information) | 13 |
| 5.4.4 | Property Identification | 13 |
| 5.5 | ER Model Definition | 14 |
| 5.5.1 | Entity Descriptions | 14 |
| 5.5.2 | Relationship Descriptions | 14 |
| 5.5.3 | ER Diagram | 15 |
| 5.6 | Design Decisions and Rationale | 15 |
| 5.6.1 | Strengths of the Proposed Model | 15 |
| 5.6.2 | Limitations and Trade-offs | 16 |
| 5.6.3 | Alternative Approaches Considered | 16 |
| 6 | Information Gathering | 16 |
| 6.1 | Knowledge Layer | 17 |
| 6.1.1 | Sources Description | 17 |
| 6.1.2 | Informal Resources Collection | 17 |
| 6.1.3 | Knowledge Resources Classification | 17 |
| 6.2 | Data Layer | 17 |
| 6.2.1 | Sources Description | 17 |
| 6.2.2 | Source 1: Mongolia Open Data Portal | 17 |
| 6.2.3 | Source 2: Mongolian Stock Exchange | 19 |
| 6.2.4 | Resources Collection and Scraping | 19 |
| 6.2.5 | Resources Classification | 35 |
| 6.3 | Design Decisions and Rationale | 35 |
| 6.3.1 | Choice of Data Sources | 35 |
| 6.3.2 | Choice of Extraction Methods | 35 |
| 6.3.3 | Data Integration Strategy | 36 |
| 7 | Language Definition | 36 |
| 7.1 | Concept Identification | 37 |
| 7.2 | Dataset Filtering | 37 |
| 8 | Knowledge Definition | 38 |
| 9 | Data Definition | 38 |
| 10 | Evaluation | 39 |
| 11 | Metadata Definition | 39 |
| 12 | Open Issues | 40 |

Revision History:



| Revision | Date | Author | Description of Changes |
|-----------------|-------------------|------------------|--|
| 0.1 | December 2, 2025 | Chinzorigt.G | Document created |
| 0.2 | December 7, 2025 | Chinzorigt.G | Introduction & DOI section created |
| 0.3 | December 10, 2025 | Chinzorigt.G | PFsheet (XLSX) & ER Model (PNG) created |
| 0.4 | December 14, 2025 | Chinzorigt.G | Purpose Formalization section created |
| 0.5 | December 14, 2025 | Enkhbayasgalan.E | Coding web scraping and JSON data extraction |
| 0.6 | December 14, 2025 | Chinzorigt.G | Information gathering section created |
| 0.7 | December 15, 2025 | Enkhbayasgalan.E | Language definition section created |

1 Introduction

Reusability is one of the main principles in the Knowledge Graph Engineering (KGE) process defined by iTelos. The KGE project documentation plays an important role in enhancing the reusability of the resources handled and produced during the process. A clear description of the resources as well as of the process (and sub-processes) developed, provides a clear understanding of the project, thus serving such information to external readers for the future exploitation of the project's outcomes.

1.1 Project Overview

This project focuses on the construction of a Knowledge Graph utilizing open data from the State Registration of Legal Entities of Mongolia. The primary objective is to visualize, in a graph structure, the complex relationships among various stakeholders within the Mongolian corporate ecosystem. Specifically, the project addresses the following relationship categories:

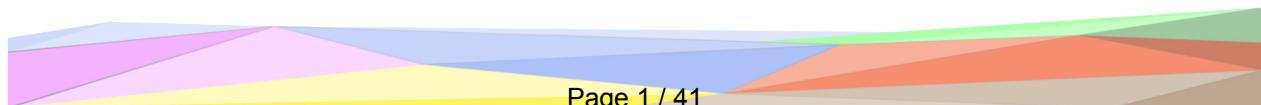
- **Shareholders and Members:** Individuals and entities holding ownership stakes in legal entities, including their classification and country of origin.
- **Officials and Controlling Entities:** Persons authorized to represent legal entities without a power of attorney, including their official positions and appointment dates.
- **Ultimate Beneficial Owners:** Natural persons who ultimately own or control legal entities, enabling transparency in corporate ownership structures.

Furthermore, for companies listed on the Mongolian Stock Exchange, the project aims to link and visualize relevant open datasets, creating an integrated view of corporate information that spans both registration data and capital market participation.

1.2 Motivation and Significance

The transparency of corporate ownership structures is essential for various stakeholders, including regulatory authorities, financial institutions, investors, and the general public. In Mongolia, as in many jurisdictions, complex corporate structures can obscure the true ownership and control of legal entities. This Knowledge Graph addresses this challenge by:

- Enabling the visualization of multi-layered ownership networks
- Facilitating the identification of individuals with significant influence across multiple entities



-
- Supporting regulatory compliance and anti-money laundering efforts
 - Providing investors and business partners with comprehensive due diligence capabilities

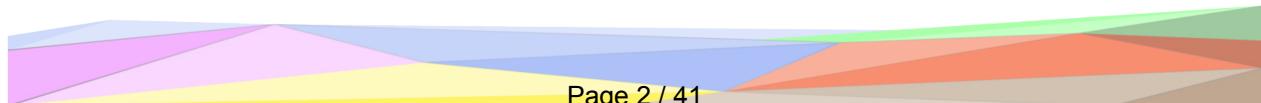
1.3 Document Structure

The current document aims to provide a detailed report of the project developed following the iTelos methodology. The report is structured as follows:

- **Section 2:** Definition of the project's purpose and its domain of interest, establishing the scope and objectives that guide all subsequent development activities.
- **Section 3:** High-level description of the project development, based on the Produce role's objectives, providing an overview of the production strategy and key milestones.
- **Sections 4, 5, 6, 7, and 8:** The description of the iTelos process phases and their activities, divided by knowledge and data layer activities. These sections detail the systematic approach taken to formalize the purpose, design the knowledge architecture, and implement the data integration processes.
- **Section 9:** The description of the evaluation criteria and metrics applied to the project's final outcome, ensuring the quality and fitness-for-purpose of the resulting Knowledge Graph.
- **Section 10:** The description of the metadata produced for all (and all kinds of) resources handled and generated by the iTelos process while executing the project, supporting long-term maintainability and reusability.
- **Section 11:** Conclusions and open issues summary, reflecting on the project outcomes and identifying opportunities for future development and enhancement.

2 Domain of Interest (DoI)

This section defines the boundaries of the Knowledge Graph Engineering project in terms of spatial and temporal dimensions. The Domain of Interest establishes the scope within which the project purpose—visualizing relationships among shareholders, officials, controlling entities, and ultimate beneficial owners of Mongolian legal entities—will be realized.



2.1 Spatial Boundaries

2.1.1 Primary Geographic Scope

The Domain of Interest is geographically bounded to **Mongolia**, specifically encompassing:

- **National Coverage:** All legal entities registered with the State Registration of Legal Entities of Mongolia, regardless of their physical location within the country's 21 aimags (provinces) and the capital city of Ulaanbaatar.
- **Administrative Divisions:** The registered addresses of legal entities span all administrative levels, including:
 - Ulaanbaatar (capital city) and its districts (dүүregs)
 - Provincial capitals (aimag centers)
 - District subdivisions (khoroos and bags)

2.1.2 International Dimension

While the primary focus is Mongolia, the domain necessarily extends to include international elements due to the nature of corporate ownership:

- **Foreign Shareholders:** Legal entities may have shareholders from foreign countries (e.g., Singapore, Hungary, as seen in the example data with "Хайнекен Азия Пасифик Пte Ltд" from Singapore and "Steppe Beverage KFT" from Hungary). The Knowledge Graph will capture the country of origin for these foreign stakeholders.
- **Cross-Border Ownership Chains:** The graph will represent ownership relationships that cross national boundaries, though detailed information about foreign parent companies is limited to what is recorded in the Mongolian registry.
- **Boundary Limitation:** The project does not extend to foreign corporate registries. Information about foreign shareholders is limited to their name, country of origin, and relationship to Mongolian entities as recorded in the State Registration system.

2.1.3 Spatial Boundary Justification

The geographic boundaries were defined based on the following considerations:

1. **Data Availability:** The open data from the State Registration of Legal Entities covers all legally registered entities within Mongolia's jurisdiction, providing comprehensive national coverage.



-
2. **Legal Framework:** The Mongolian Company Law and relevant regulations govern entities within these boundaries, ensuring data consistency and regulatory compliance.
 3. **User Needs:** The identified personas (investigators, business analysts, portfolio managers, compliance officers) primarily operate within the Mongolian legal and business environment, making national scope most relevant to their needs.

2.2 Temporal Boundaries

2.2.1 Historical Scope

The temporal dimension of the Domain of Interest encompasses:

- **Start Date:** The Knowledge Graph will include data from **January 1, 2000** onwards. This date was selected because:
 - It captures the modern era of Mongolia's market economy development
 - Most currently active legal entities were registered after this date
 - Data quality and completeness improve significantly from this period
- **End Date:** The temporal scope extends to the **present day**, with the expectation of ongoing updates as new registrations and changes occur in the source registry.
- **Historical Records:** For entities registered before 2000 that remain active, their historical information (as available in the registry) will be included, though with the understanding that older records may be less complete.

2.2.2 Temporal Granularity

The Knowledge Graph captures temporal information at the following levels of granularity:

- **Registration Dates:** Precise dates (YYYY.MM.DD format) for:
 - Initial entity registration
 - Shareholder/member registration
 - Appointment of authorized representatives
 - Ultimate beneficial owner registration
 - Business activity registration
 - Restructuring events



-
- **Validity Periods:** For certain business activities (particularly licensed activities such as alcohol production), the data includes validity periods with start and end dates (e.g., "2019.04.14 - 2022.04.14" for alcohol production licenses).
 - **Change Tracking:** The system captures the dates when changes occurred, enabling temporal analysis of corporate evolution.

2.2.3 Temporal Boundary Justification

The temporal boundaries were established based on:

1. **Data Completeness:** Records from 2000 onwards demonstrate higher data quality and completeness compared to earlier periods.
2. **Relevance to Current Analysis:** The 20+ year historical window provides sufficient depth for:
 - Tracking corporate evolution and restructuring
 - Identifying long-term patterns in ownership and control
 - Supporting due diligence investigations requiring historical context
3. **Regulatory Evolution:** Mongolia's modern corporate governance framework, including beneficial ownership disclosure requirements, has developed primarily within this timeframe.

2.3 Domain Boundary Summary

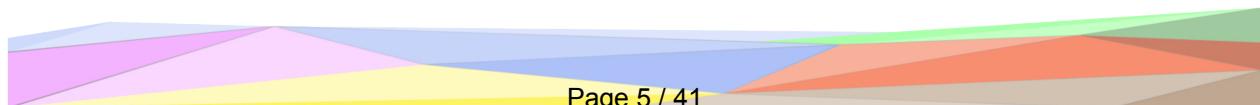
| Dimension | Boundary Definition |
|-------------------------------|---|
| Geographic Scope | Mongolia (all 21 aimags and Ulaanbaatar) |
| International Elements | Foreign shareholder countries (as recorded in Mongolian registry) |
| Institutional Scope | State Registration of Legal Entities, Mongolian Stock Exchange |
| Temporal Start | January 1, 2000 |
| Temporal End | Present (with ongoing updates) |
| Temporal Granularity | Daily (date-level precision for all recorded events) |

Table 1: Summary of Domain of Interest Boundaries

2.4 Out of Scope

To provide clarity on the Domain of Interest boundaries, the following elements are explicitly **excluded** from the project scope:

- **Foreign Registry Data:** Detailed corporate information from foreign jurisdictions (beyond what is recorded in the Mongolian registry)



-
- **Informal Enterprises:** Unregistered businesses or sole proprietorships not captured in the State Registration system
 - **Historical Records Pre-1990:** Data from the socialist period before Mongolia's transition to a market economy
 - **Non-Corporate Entities:** Government agencies, international organizations, and diplomatic missions (unless they appear as shareholders in registered companies)
 - **Real-Time Transaction Data:** Stock trading data, financial transactions, or other real-time market information beyond static company registration data

3 Project Development

This section describes, at top level, how the project's objectives (or "The Purpose") will be satisfied. More in details the current section aims at describing how the data production process is performed.

3.1 Data Production

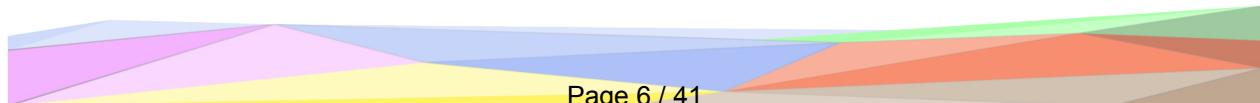
The description of which (quality) data needs to be created to satisfy the project purpose. This sub-section highlights the role of the data producer. The sub-section aims at describing how the data producer creates the data required to satisfy the project's purpose.

4 Initial Resources

This section describes the already available resources considered for the project. More in detail the resources here described, are quality resources (compliant with the quality and reusability guidelines defined by iTleos. 6*, or at least 5*) which don't need to be processed or created by a data producer. The resources described in this section are those that can be already composed by the data consumer to satisfy the project's purpose.

In this section are described both the resources selected, and the sources from which such resources have been retrieved.

This section describes the two kind of resources considered by a projects, by filling the two sub-sections here below.



-
- **Knowledge resources:** iTelos compliant reference schemas and ontologies initially collected to satisfy the purpose along the KGE process. The knowledge resources initial metadata have to be reported here.
 - **Data sources:** iTelos compliant datasets initially collected to satisfy the purpose along the KGE process. The data resources initial metadata have to be reported here.

5 Purpose Formalization

This section documents the activities and results achieved during the first phase of the iTelos methodology for constructing a Knowledge Graph based on the State Registration of Legal Entities in Mongolia. The project aims to visualize relationships among shareholders, members, officials, controlling entities, and ultimate beneficial owners, with additional linkages to Mongolian Stock Exchange data for listed companies.

5.1 Scenarios Definition

The following usage scenarios describe the multiple aspects considered by the project purpose:

5.1.1 Scenario 1: Corporate Ownership Investigation

A financial investigator needs to trace the ownership structure of a company suspected of involvement in financial irregularities. The investigator must identify all shareholders, their respective ownership percentages, and any connections to other legal entities. The system should reveal complex ownership chains, including nested corporate structures where companies own shares in other companies, ultimately leading to the identification of ultimate beneficial owners.

5.1.2 Scenario 2: Due Diligence for Business Partnerships

A business development manager at a Mongolian corporation is evaluating potential partners for a joint venture. Before entering negotiations, they need to understand the governance structure of target companies, including who has authority to represent the company without power of attorney, the company's business activities, and any organizational restructuring history that might indicate instability or strategic pivots.

5.1.3 Scenario 3: Stock Market Investment Analysis

An investment analyst researching publicly traded companies on the Mongolian Stock Exchange requires comprehensive information about company leadership, ownership con-



centration, and cross-holdings between listed entities. The analyst needs to identify potential conflicts of interest where the same individuals serve as officials across multiple companies or where significant ownership overlaps exist.

5.1.4 Scenario 4: Regulatory Compliance Monitoring

A compliance officer at a regulatory authority monitors legal entities for adherence to ownership disclosure requirements. They need to identify companies where ultimate beneficial owner information is incomplete, track changes in controlling persons over time, and detect patterns that might indicate attempts to obscure true ownership.

5.1.5 Scenario 5: Anti-Money Laundering (AML) Analysis

An AML specialist investigates networks of companies that may be used for layering illicit funds. The specialist needs to visualize connections between entities through shared shareholders, officials, and beneficial owners, identifying clusters of related companies and individuals who appear across multiple entities in patterns suggesting coordinated control.

5.2 Personas

5.2.1 Persona 1: Investigator Batbold

- **Role:** Financial Crimes Investigator at the Financial Regulatory Commission
- **Age:** 42 years old
- **Background:** 15 years of experience in financial investigation, former police detective
- **Technical Skills:** Moderate; comfortable with databases but prefers visual interfaces
- **Goals:** Quickly identify ownership networks, trace beneficial owners, and document evidence chains for legal proceedings
- **Pain Points:** Currently relies on manual searches through multiple registries; difficulty connecting entities across different data sources
- **Primary Scenario:** Scenario 1, Scenario 5



5.2.2 Persona 2: Business Analyst Oyungerel

- **Role:** Senior Business Development Manager at a mining corporation
- **Age:** 35 years old
- **Background:** MBA graduate, 10 years in corporate strategy
- **Technical Skills:** High; proficient in data analysis tools and visualization platforms
- **Goals:** Conduct thorough due diligence on potential partners, understand corporate governance structures, assess business stability
- **Pain Points:** Time-consuming process to gather information from multiple sources; difficulty assessing company history and restructuring events
- **Primary Scenario:** Scenario 2

5.2.3 Persona 3: Portfolio Manager Enkhjargal

- **Role:** Portfolio Manager at an investment fund
- **Age:** 38 years old
- **Background:** CFA charterholder, specializes in Mongolian equities
- **Technical Skills:** Very high; uses quantitative analysis tools daily
- **Goals:** Identify investment opportunities, assess governance risks, understand ownership concentration in listed companies
- **Pain Points:** Limited integration between stock exchange data and corporate registry information; manual effort required to build comprehensive company profiles
- **Primary Scenario:** Scenario 3

5.2.4 Persona 4: Compliance Officer Munkhbat

- **Role:** Senior Compliance Officer at the General Authority for State Registration
- **Age:** 45 years old
- **Background:** Legal background, 20 years in public administration
- **Technical Skills:** Moderate; familiar with government databases
- **Goals:** Monitor compliance with disclosure requirements, identify incomplete registrations, generate compliance reports



-
- **Pain Points:** Difficulty tracking changes over time; no automated alerting for suspicious patterns
 - **Primary Scenario:** Scenario 4

5.3 Competency Questions (CQs)

The following competency questions were developed based on the personas and scenarios defined above:

5.3.1 Entity Identification and Basic Information

- CQ1: What is the registration number of a legal entity given its name?
- CQ2: What is the registration date of a specific legal entity?
- CQ3: What is the legal form (Хэлбэр) of a given company?
- CQ4: What is the type (Төрөл) classification of a legal entity?
- CQ5: What is the registered address of a legal entity?

5.3.2 Ownership and Shareholder Information

- CQ6: Who are all the shareholders and members of a specific legal entity?
- CQ7: What is the classification (Ангилал) of each shareholder in a company?
- CQ8: Which country does each shareholder belong to?
- CQ9: What is the gender distribution of shareholders in a given company?
- CQ10: When was each shareholder registered as a member of the company?
- CQ11: Which companies share common shareholders?
- CQ12: What legal entities does a specific individual hold shares in?

5.3.3 Management and Representation Authority

- CQ13: Who are the officials authorized to represent a company without power of attorney?
- CQ14: What position does each authorized representative hold?
- CQ15: Which companies does a specific individual have authority to represent?



CQ16: Are there individuals who serve as authorized representatives in multiple companies?

CQ17: What is the registration date of each authorized representative's appointment?

5.3.4 Ultimate Beneficial Ownership

CQ18: Who are the ultimate beneficial owners of a specific legal entity?

CQ19: What is the classification of each ultimate beneficial owner?

CQ20: Which companies share the same ultimate beneficial owner?

CQ21: For a given individual, in which companies are they listed as an ultimate beneficial owner?

CQ22: Which companies lack complete ultimate beneficial owner information?

5.3.5 Business Activities

CQ23: What are the registered business activities of a legal entity?

CQ24: What is the status (active/inactive) of each business activity?

CQ25: Which companies operate in the same business sector?

CQ26: When was a specific business activity registered for a company?

5.3.6 Corporate Restructuring and History

CQ27: Has a legal entity undergone any organizational restructuring?

CQ28: What was the previous name of a restructured company?

CQ29: What type of restructuring occurred (merger, division, transformation)?

CQ30: What is the chronological history of changes for a given company?

5.3.7 Network Analysis and Cross-Entity Queries

CQ31: What is the network of companies connected through shared ownership?

CQ32: Which individuals appear in multiple roles (shareholder, official, beneficial owner) across different companies?

CQ33: What is the degree of separation between two legal entities through ownership or management relationships?

CQ34: Which clusters of companies exhibit patterns of coordinated control?



5.4 Concepts Identification

Based on the scenarios, personas, and competency questions, the following concepts have been identified and classified according to their popularity and relevance to the project purpose.

5.4.1 High Popularity Concepts (Core Entities)

| Concept (English) | Concept (Mongolian) | Description |
|---------------------------|----------------------------|---|
| Legal Entity | Хуулийн этгээд | The primary entity representing registered companies and organizations |
| Person | Хүн | Individual persons who can be shareholders, officials, or beneficial owners |
| Shareholder/Member | Хувьцаа эзэмшигч/Гишүүн | Persons or entities holding ownership stakes |
| Authorized Representative | Итгэмжлэлгүй төлөөлөгч | Officials with authority to represent without power of attorney |
| Ultimate Beneficial Owner | Эцсийн өмчлөгч | The final natural person who ultimately owns or controls the entity |

5.4.2 Medium Popularity Concepts (Supporting Entities)

| Concept (English) | Concept (Mongolian) | Description |
|-------------------|------------------------|---|
| Business Activity | Үйл ажиллагааны чиглэл | Types of business operations registered for an entity |
| Position/Title | Албан тушаал | Official positions held by authorized representatives |
| Legal Form | Хэлбэр | The legal structure type of the entity (LLC, JSC, etc.) |
| Entity Type | Төрөл | Classification type of the legal entity |

| Concept (English) | Concept (Mongolian) | Description |
|-------------------|---------------------|--|
| Country | Улс | Country of origin for foreign shareholders |

5.4.3 Low Popularity Concepts (Contextual Information)

| Concept (English) | Concept (Mongolian) | Description |
|---------------------|---------------------|--|
| Restructuring Event | Өөрчлөлт | Corporate reorganization events |
| Address | Хаяг | Physical location of the legal entity |
| Classification | Ангилал | Category classification for shareholders and beneficial owners |
| Activity Status | Төлөв | Current status of business activities |

5.4.4 Property Identification

| Property (English) | Property (Mongolian) | Data Type | Applies To |
|---------------------|------------------------|-----------|---------------------------|
| Registration Number | Регистрийн дугаар | Number | Legal Entity |
| Name | Оноосон нэр | String | Legal Entity |
| Registration Date | Бүртгэсэн огноо | Date | Multiple entities |
| First Name | Нэр | String | Person |
| Patronymic | Эцэг/эх/-ийн нэр | String | Person |
| Gender | Хүйс | String | Person |
| Country Name | Улсын нэр | String | Person |
| Position Title | Албан тушаал | String | Authorized Representative |
| Activity Direction | Үйл ажиллагааны чиглэл | String | Business Activity |
| Status | Төлөв | String | Business Activity |

| Property (English) | Property (Mongolian) | Data Type | Applies To |
|--------------------|-------------------------------------|-----------|---------------------|
| Restructuring Type | Өөрчлөгдөн зохион байгуулсан хэлбэр | String | Restructuring Event |
| Previous Name | Өөрчлөлтийн өмнөх оноосон нэр | String | Restructuring Event |
| Change Notes | Өөрчлөлтийн тэмдэглэл | String | Restructuring Event |

5.5 ER Model Definition

Based on the concepts and properties identified above, the following Entity-Relationship model has been designed to represent the purpose of the Knowledge Graph.

5.5.1 Entity Descriptions

LegalEntity - The central entity representing registered legal entities in Mongolia.

- Primary Key: registrationNumber
- Attributes: name, registrationDate, legalForm, entityType, address

Person - Represents natural persons who participate in legal entities.

- Attributes: firstName, patronymic, gender, countryName

BusinessActivity - Represents registered business activity directions.

- Attributes: activityDirection, status, registrationDate

Position - Represents official positions/titles.

- Attributes: positionTitle

RestructuringEvent - Represents corporate reorganization events.

- Attributes: restructuringType, registrationDate, previousName, changeNotes

5.5.2 Relationship Descriptions



| Relationship | From Entity | To Entity | Cardinal-ity | Attributes |
|--------------------|-------------|--------------------|--------------|----------------------------------|
| hasShareholder | LegalEntity | Person | 1:N | classification, registrationDate |
| hasAuthorizedRep | LegalEntity | Person | 1:N | position, registrationDate |
| hasBeneficialOwner | LegalEntity | Person | 1:N | classification, registrationDate |
| hasActivity | LegalEntity | BusinessActivity | 1:N | - |
| hasRestructuring | LegalEntity | RestructuringEvent | 1:N | - |
| holdsPosition | Person | Position | N:M | - |

5.5.3 ER Diagram

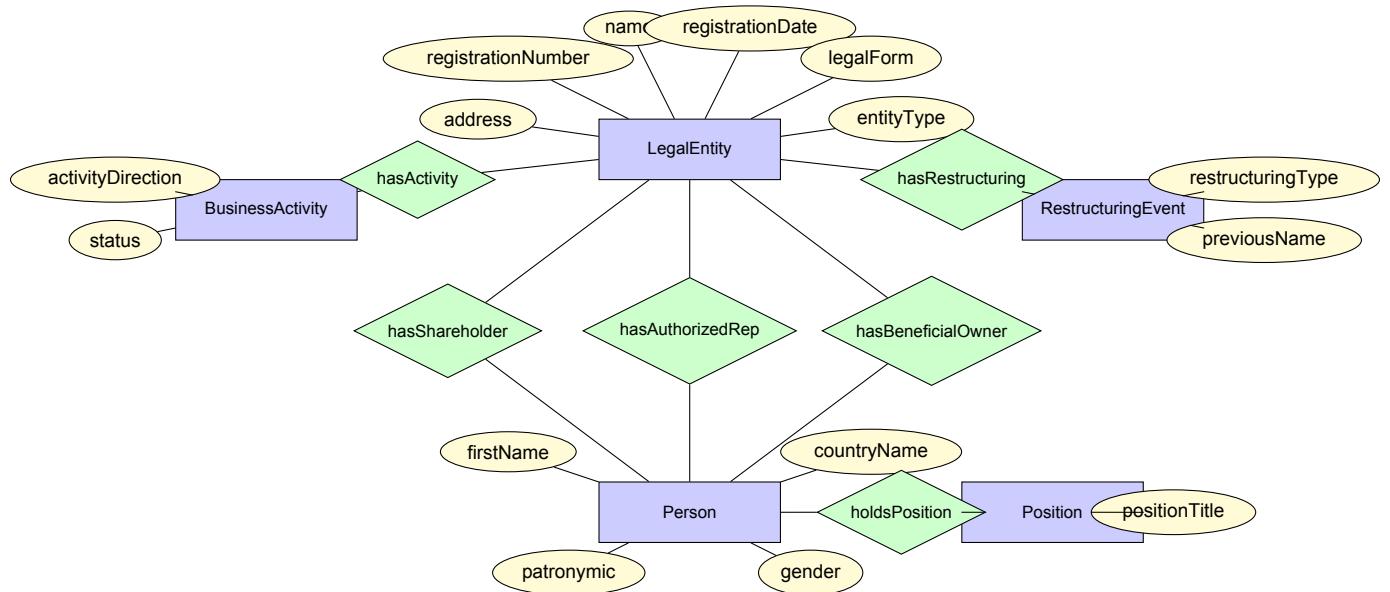
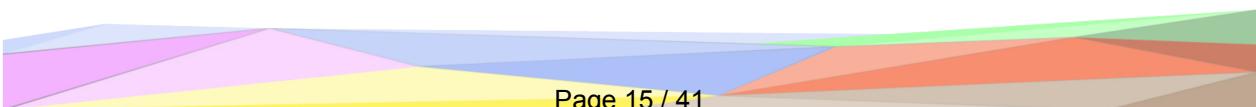


Figure 1: Entity-Relationship Diagram for Mongolian Legal Entities Knowledge Graph

5.6 Design Decisions and Rationale

5.6.1 Strengths of the Proposed Model

- Comprehensive Coverage:** The model captures all key aspects of the source data, including ownership, management, beneficial ownership, business activities, and corporate history.



-
2. **Network Analysis Support:** The design enables complex graph queries to identify relationships between entities and persons across multiple companies, supporting the investigative and analytical use cases identified in the scenarios.
 3. **Temporal Tracking:** Registration dates are captured for all major relationships, enabling historical analysis and change tracking.
 4. **Flexibility for Extension:** The model can be extended to incorporate Mongolian Stock Exchange data through additional entities and relationships without major restructuring.

5.6.2 Limitations and Trade-offs

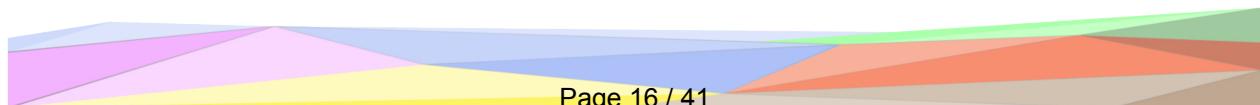
1. **Person Identification:** The source data does not include unique identifiers for persons, making it challenging to definitively link the same individual across different roles and companies. Name-based matching may be required.
2. **Ownership Percentages:** The current data format does not explicitly include shareholding percentages, which limits quantitative ownership analysis.
3. **Historical Completeness:** Changes in shareholders, officials, and beneficial owners over time are not fully captured in the source data structure, limiting historical analysis capabilities.

5.6.3 Alternative Approaches Considered

1. **Single Person Entity vs. Role-Specific Entities:** We considered creating separate entities for Shareholders, Officials, and Beneficial Owners. However, the unified Person entity was chosen to better support network analysis and identify individuals appearing in multiple roles.
2. **Address as Entity vs. Attribute:** Address could be modeled as a separate entity to enable geographic analysis. This was deferred to keep the initial model simpler, but remains a candidate for future extension.

6 Information Gathering

This section reports the execution of the activities involved in the Information Gathering iTelos phase. This section describes both the resources selected and the sources from which such resources have been retrieved. The project focuses on gathering data about Mongolian legal entities from government open data portals and supplementary information from the Mongolian Stock Exchange for publicly listed companies.



6.1 Knowledge Layer

6.1.1 Sources Description

The knowledge layer resources provide the conceptual foundation and domain understanding necessary for modeling Mongolian legal entity structures.

6.1.2 Informal Resources Collection

Knowledge resources were collected through manual review of documentation, legal frameworks, and domain expert consultations. Key terminological resources include:

- Legal entity classification schemas from the State Registration Authority
- Ownership structure definitions from Mongolian Company Law
- Financial reporting terminology from the Mongolian Stock Exchange
- Business activity classification codes (aligned with ISIC standards)

6.1.3 Knowledge Resources Classification

6.2 Data Layer

6.2.1 Sources Description

Two primary data sources were identified for this project:

6.2.2 Source 1: Mongolia Open Data Portal

Description: The Mongolia Open Data Portal (<https://opendata.gov.mn>) is the official government platform for publishing open datasets. The legal entities dataset (ID: 5372) contains comprehensive registration information including:

- Basic entity information (registration number, name, date, form, type, address)
- Shareholder and member information
- Authorized representatives (officials without power of attorney)
- Ultimate beneficial owners
- Business activity registrations
- Organizational restructuring history

Access Method: Direct JSON file download via API endpoint.

Data Quality Assessment:



Table 7: Knowledge Layer Sources

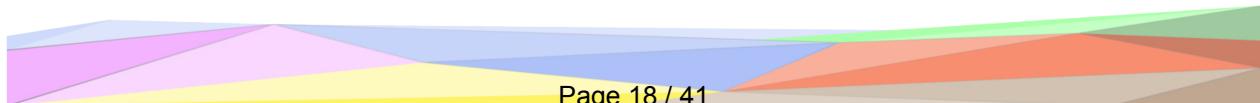
| Source Name | URL | Type | Description |
|--|---|-----------------------|---|
| Mongolia Open Data Portal | https://opendata.gov.mn | Government Portal | Official open data platform providing metadata and data dictionaries |
| General Authority for State Registration | https://burtgel.gov.mn | Government Website | Legal entity registration authority with terminology definitions |
| Mongolian Stock Exchange | https://mse.mn/mn | Financial Institution | Stock exchange with listed company information and financial terminology |
| Company Law of Mongolia | Legal Document | Legislation | Defines legal entity types, ownership structures, and governance requirements |

Table 8: Knowledge Resources Classification

| Resource | Classification | Justification |
|-------------------------|-----------------------|---|
| Legal Entity Types | Core | Fundamental classification for all registered entities |
| Person/Organization | Common | Standard distinction used across multiple domains |
| Ownership Relations | Core | Central to the project purpose of visualizing ownership |
| Business Activity Codes | Contextual | Supporting information for entity characterization |
| Restructuring Types | Contextual | Historical information for tracking changes |
| Stock Exchange Listings | Contextual | Additional data for publicly traded companies |

Table 9: Data Layer Sources

| Source | URL | Format | Access | Update Freq. |
|-------------------------------------|---|---------------|-----------------|---------------------|
| Mongolia Open Data - Legal Entities | https://opendata.gov.mn/dataview/5372 | JSON | Direct Download | Periodic |
| Mongolian Stock Exchange | https://mse.mn/mn | HTML | Web Scraping | Real-time |



-
- **Strengths:** Official government source, structured JSON format, comprehensive coverage, regular updates
 - **Weaknesses:** Some fields may contain null values, Mongolian language requires encoding handling, no unique person identifiers

6.2.3 Source 2: Mongolian Stock Exchange

Description: The Mongolian Stock Exchange (MSE) website provides information about publicly listed companies, including trading data, company profiles, financial reports, and corporate announcements.

Access Method: Web scraping required as no official API is available.

Data Quality Assessment:

- **Strengths:** Real-time market data, detailed company profiles for listed entities, financial performance metrics
- **Weaknesses:** Requires web scraping, HTML structure may change, limited to publicly traded companies only

6.2.4 Resources Collection and Scraping

6.2.4.1 JSON Data Extraction from Open Data Portal The following Python code extracts legal entity data from the Mongolia Open Data Portal:

```
1 import requests
2 import json
3 import os
4 from datetime import datetime
5 from typing import Optional, List, Dict, Any
6
7 class MongoliaOpenDataExtractor:
8     """
9         Extractor for Mongolia Open Data Portal - Legal Entities Dataset
10        Source: https://opendata.gov.mn/dataview/5372
11    """
12
13    def __init__(self, output_dir: str = "data"):
14        self.base_url = "https://opendata.gov.mn"
15        self.dataset_id = "5372"
16        self.output_dir = output_dir
17        self.session = requests.Session()
18        self.session.headers.update({
19            'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36',
20            'Accept': 'application/json',
```

```

21         'Accept-Language': 'mn-MN,mn;q=0.9,en;q=0.8'
22     })
23
24     # Create output directory if not exists
25     if not os.path.exists(output_dir):
26         os.makedirs(output_dir)
27
28     def get_dataset_metadata(self) -> Optional[Dict[str, Any]]:
29         """Retrieve dataset metadata including download links"""
30         try:
31             metadata_url = f"{self.base_url}/api/3/action/package_show?id={self.
32 dataset_id}"
33             response = self.session.get(metadata_url, timeout=30)
34             response.raise_for_status()
35             return response.json()
36         except requests.RequestException as e:
37             print(f"Error fetching metadata: {e}")
38             return None
39
40     def download_json_data(self, url: str = None) -> Optional[List[Dict]]:
41         """
42             Download JSON data from the open data portal
43             Direct download from dataview endpoint
44         """
45         try:
46             # Direct download URL for dataset 5372
47             if url is None:
48                 url = f"{self.base_url}/dataview/{self.dataset_id}"
49
50             print(f"Downloading data from: {url}")
51             response = self.session.get(url, timeout=120)
52             response.raise_for_status()
53
54             # Handle different response formats
55             content_type = response.headers.get('Content-Type', '')
56
57             if 'application/json' in content_type:
58                 data = response.json()
59             else:
60                 # Try to parse as JSON anyway
61                 data = json.loads(response.text)
62
63             print(f"Successfully downloaded {len(data)} if isinstance(data, list)
64 else 1} records")
65             return data
66
67         except requests.RequestException as e:

```

```

66         print(f"Error downloading data: {e}")
67     return None
68 except json.JSONDecodeError as e:
69     print(f"Error parsing JSON: {e}")
70     return None
71
72 def download_with_pagination(self, limit: int = 1000) -> List[Dict]:
73     """
74     Download data with pagination support for large datasets
75     """
76     all_data = []
77     offset = 0
78
79     while True:
80         try:
81             # API endpoint with pagination
82             url = f"{self.base_url}/api/3/action/datastore_search"
83             params = {
84                 'resource_id': self.dataset_id,
85                 'limit': limit,
86                 'offset': offset
87             }
88
89             response = self.session.get(url, params=params, timeout=60)
90             response.raise_for_status()
91             result = response.json()
92
93             if result.get('success'):
94                 records = result.get('result', {}).get('records', [])
95                 if not records:
96                     break
97
98                 all_data.extend(records)
99                 print(f"Downloaded {len(all_data)} records...")
100                offset += limit
101            else:
102                break
103
104        except requests.RequestException as e:
105            print(f"Error during pagination: {e}")
106            break
107
108    return all_data
109
110 def parse_legal_entity(self, raw_data: Dict) -> Dict[str, Any]:
111     """
112     Parse raw JSON data into structured legal entity format

```

```

113     """
114
115     entity = {
116         'registration_number': raw_data.get(' '),
117         'name': raw_data.get(' '),
118         'registration_date': raw_data.get(' '),
119         'legal_form': raw_data.get(' '),
120         'entity_type': raw_data.get(' '),
121         'address': raw_data.get(' '),
122         'shareholders': [],
123         'authorized_representatives': [],
124         'beneficial_owners': [],
125         'business_activities': [],
126         'restructuring_events': []
127     }
128
129
130     # Parse shareholders/members
131     shareholders_data = raw_data.get(' ', ' ', [])
132     if isinstance(shareholders_data, list):
133         for sh in shareholders_data:
134             entity['shareholders'].append({
135                 'classification': sh.get(' '),
136                 'country': sh.get(' '),
137                 'patronymic': sh.get('//-' '),
138                 'name': sh.get(' '),
139                 'gender': sh.get(' '),
140                 'registration_date': sh.get(' ')
141             })
142
143     # Parse authorized representatives
144     auth_rep_data = raw_data.get(
145         ' ', ' ', ' ',
146         []
147     )
148     if isinstance(auth_rep_data, list):
149         for rep in auth_rep_data:
150             entity['authorized_representatives'].append({
151                 'position': rep.get(' '),
152                 'country': rep.get(' '),
153                 'patronymic': rep.get('//-' '),
154                 'name': rep.get(' '),
155                 'gender': rep.get(' '),
156                 'registration_date': rep.get(' ')
157             })
158
159     # Parse beneficial owners
160     bo_data = raw_data.get(' ', [], [])
161     if isinstance(bo_data, list):

```

```

160     for bo in bo_data:
161         entity['beneficial_owners'].append({
162             'classification': bo.get(''),
163             'patronymic': bo.get('//-' ),
164             'name': bo.get(''),
165             'gender': bo.get(''),
166             'registration_date': bo.get(' ')
167         })
168
169     # Parse business activities
170     activity_data = raw_data.get(' ', [])
171     if isinstance(activity_data, list):
172         for act in activity_data:
173             entity['business_activities'].append({
174                 'status': act.get(''),
175                 'registration_date': act.get(' '),
176                 'activity_direction': act.get(' ')
177             })
178
179     # Parse restructuring events
180     restructure_data = raw_data.get(' ', [])
181     if isinstance(restructure_data, list):
182         for rst in restructure_data:
183             entity['restructuring_events'].append({
184                 'restructuring_type': rst.get(' '),
185                 'registration_date': rst.get(' '),
186                 'previous_name': rst.get(' '),
187                 'change_notes': rst.get(' ')
188             })
189
190     return entity
191
192 def save_to_json(self, data: List[Dict], filename: str = None) -> str:
193     """Save extracted data to JSON file"""
194     if filename is None:
195         timestamp = datetime.now().strftime('%Y%m%d_%H%M%S')
196         filename = f"legal_entities_{timestamp}.json"
197
198     filepath = os.path.join(self.output_dir, filename)
199
200     with open(filepath, 'w', encoding='utf-8') as f:
201         json.dump(data, f, ensure_ascii=False, indent=2)
202
203     print(f"Data saved to: {filepath}")
204     return filepath
205
206 def extract_all(self) -> List[Dict]:

```

```

207     """
208     Main extraction method - downloads and parses all data
209     """
210
211     print("Starting data extraction from Mongolia Open Data Portal...")
212
213     # Download raw data
214     raw_data = self.download_json_data()
215
216     if raw_data is None:
217         print("Attempting paginated download...")
218         raw_data = self.download_with_pagination()
219
220     if not raw_data:
221         print("No data retrieved")
222         return []
223
224     # Parse entities
225     parsed_entities = []
226     for item in raw_data:
227         parsed_entity = self.parse_legal_entity(item)
228         parsed_entities.append(parsed_entity)
229
230     # Save to file
231     self.save_to_json(parsed_entities)
232
233     print(f"Extraction complete. Total entities: {len(parsed_entities)}")
234     return parsed_entities
235
236 # Main execution
237 if __name__ == "__main__":
238     extractor = MongoliaOpenDataExtractor(output_dir="extracted_data")
239     entities = extractor.extract_all()
240
241     # Print sample
242     if entities:
243         print("\nSample entity:")
244         print(json.dumps(entities[0], indent=2, ensure_ascii=False))

```

Listing 1: JSON Data Extraction from Mongolia Open Data Portal

6.2.4.2 Web Scraping from Mongolian Stock Exchange The following Python code scrapes company information from the Mongolian Stock Exchange website:

```

1 import requests
2 from bs4 import BeautifulSoup
3 import json

```

```

4 import time
5 import os
6 from datetime import datetime
7 from typing import Optional, List, Dict, Any
8 from urllib.parse import urljoin
9 import re
10
11 class MSEScraper:
12     """
13         Web Scraper for Mongolian Stock Exchange (MSE)
14         Source: https://mse.mn/mn
15     """
16
17     def __init__(self, output_dir: str = "data"):
18         self.base_url = "https://mse.mn"
19         self.language = "mn" # Mongolian language
20         self.output_dir = output_dir
21         self.session = requests.Session()
22         self.session.headers.update({
23             'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36',
24             'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
25             'Accept-Language': 'mn-MN,mn;q=0.9,en-US;q=0.8,en;q=0.7',
26             'Connection': 'keep-alive'
27         })
28
29         # Rate limiting
30         self.request_delay = 2 # seconds between requests
31
32         # Create output directory
33         if not os.path.exists(output_dir):
34             os.makedirs(output_dir)
35
36     def _make_request(self, url: str, params: Dict = None) -> Optional[BeautifulSoup]:
37         """Make HTTP request with rate limiting and error handling"""
38         try:
39             time.sleep(self.request_delay)
40             response = self.session.get(url, params=params, timeout=30)
41             response.raise_for_status()
42             response.encoding = 'utf-8'
43             return BeautifulSoup(response.text, 'html.parser')
44         except requests.RequestException as e:
45             print(f"Request error for {url}: {e}")
46             return None
47

```

```

48     def get_listed_companies(self) -> List[Dict[str, Any]]:
49         """
50             Scrape list of all companies listed on MSE
51         """
52         companies = []
53
54         # Main securities listing page
55         listing_url = f"{self.base_url}/{self.language}/security"
56
57         print(f"Fetching listed companies from: {listing_url}")
58         soup = self._make_request(listing_url)
59
60         if not soup:
61             return companies
62
63         # Find company listing table
64         # Note: Actual selectors depend on MSE website structure
65         company_tables = soup.find_all('table', class_='table')
66
67         for table in company_tables:
68             rows = table.find_all('tr')[1:] # Skip header
69
70             for row in rows:
71                 cols = row.find_all('td')
72                 if len(cols) >= 3:
73                     company = {
74                         'symbol': cols[0].get_text(strip=True),
75                         'name': cols[1].get_text(strip=True),
76                         'sector': cols[2].get_text(strip=True) if len(cols) > 2
77                     }
78                 else:
79                     company = None
80
81                     'detail_url': None
82
83
84                     # Extract detail link
85                     link = cols[1].find('a')
86                     if link and link.get('href'):
87                         company['detail_url'] = urljoin(self.base_url, link['href'])
88
89
90         companies.append(company)
91
92         print(f"Found {len(companies)} listed companies")
93         return companies
94
95
96     def get_company_details(self, company_url: str) -> Optional[Dict[str, Any]]:
97         """
98             Scrape detailed information for a specific company

```

```
93     """
94
95     if not company_url:
96         return None
97
98     print(f"Fetching company details from: {company_url}")
99     soup = self._make_request(company_url)
100
101    if not soup:
102        return None
103
104    details = {
105        'url': company_url,
106        'scraped_at': datetime.now().isoformat()
107    }
108
109    # Company profile section
110    profile_section = soup.find('div', class_='company-profile')
111    if profile_section:
112        details['profile'] = self._parse_profile(profile_section)
113
114    # Trading information
115    trading_section = soup.find('div', class_='trading-info')
116    if trading_section:
117        details['trading'] = self._parse_trading_info(trading_section)
118
119    # Company information table
120    info_table = soup.find('table', class_='company-info')
121    if info_table:
122        details['company_info'] = self._parse_info_table(info_table)
123
124    # Shareholders information
125    shareholders_section = soup.find('div', {'id': 'shareholders'})
126    if shareholders_section:
127        details['major_shareholders'] = self._parse_shareholders(
128            shareholders_section)
129
130        # Board members / Management
131        management_section = soup.find('div', {'id': 'management'})
132        if management_section:
133            details['management'] = self._parse_management(management_section)
134
135        # Financial highlights
136        financial_section = soup.find('div', {'id': 'financials'})
137        if financial_section:
138            details['financials'] = self._parse_financials(financial_section)
139
140    return details
```

```

139
140     def _parse_profile(self, section) -> Dict[str, str]:
141         """Parse company profile section"""
142         profile = {}
143
144         # Company name
145         name_elem = section.find(['h1', 'h2', 'h3'], class_='company-name')
146         if name_elem:
147             profile['name'] = name_elem.get_text(strip=True)
148
149         # Description
150         desc_elem = section.find('p', class_='description')
151         if desc_elem:
152             profile['description'] = desc_elem.get_text(strip=True)
153
154         # Extract key-value pairs
155         info_items = section.findall(['div', 'li'], class_='info-item')
156         for item in info_items:
157             label = item.find(['span', 'strong'], class_='label')
158             value = item.find(['span', 'div'], class_='value')
159             if label and value:
160                 key = label.get_text(strip=True).rstrip(':')
161                 profile[key] = value.get_text(strip=True)
162
163         return profile
164
165
166     def _parse_trading_info(self, section) -> Dict[str, Any]:
167         """Parse trading information"""
168         trading = {}
169
170         # Current price
171         price_elem = section.find('span', class_='current-price')
172         if price_elem:
173             price_text = price_elem.get_text(strip=True)
174             trading['current_price'] = self._extract_number(price_text)
175
176         # Price change
177         change_elem = section.find('span', class_='price-change')
178         if change_elem:
179             trading['price_change'] = change_elem.get_text(strip=True)
180
181         # Volume
182         volume_elem = section.find('span', class_='volume')
183         if volume_elem:
184             trading['volume'] = self._extract_number(volume_elem.get_text(strip=True))

```

```

185     # Market cap
186     mcap_elem = section.find('span', class_='market-cap')
187     if mcap_elem:
188         trading['market_cap'] = mcap_elem.get_text(strip=True)
189
190     return trading
191
192     def _parse_info_table(self, table) -> Dict[str, str]:
193         """Parse information table"""
194         info = {}
195         rows = table.find_all('tr')
196
197         for row in rows:
198             cells = row.find_all(['th', 'td'])
199             if len(cells) >= 2:
200                 key = cells[0].get_text(strip=True).rstrip(':')
201                 value = cells[1].get_text(strip=True)
202                 info[key] = value
203
204         return info
205
206     def _parse_shareholders(self, section) -> List[Dict[str, Any]]:
207         """Parse major shareholders information"""
208         shareholders = []
209
210         table = section.find('table')
211         if table:
212             rows = table.find_all('tr')[1:] # Skip header
213
214             for row in rows:
215                 cols = row.find_all('td')
216                 if len(cols) >= 2:
217                     shareholder = {
218                         'name': cols[0].get_text(strip=True),
219                         'shares': self._extract_number(cols[1].get_text(strip=True))
220                     } if len(cols) > 1 else None,
221                         'percentage': cols[2].get_text(strip=True) if len(cols) > 2
222                     else None
223                     }
224                     shareholders.append(shareholder)
225
226             return shareholders
227
228     def _parse_management(self, section) -> List[Dict[str, str]]:
229         """Parse management/board members information"""
230         management = []

```

```

230     members = section.find_all(['div', 'li'], class_='member')
231     for member in members:
232         name_elem = member.find(['h4', 'span'], class_='name')
233         position_elem = member.find(['p', 'span'], class_='position')
234
235         if name_elem:
236             management.append({
237                 'name': name_elem.get_text(strip=True),
238                 'position': position_elem.get_text(strip=True) if position_elem
239             else None
240         })
241
242     # Alternative: table format
243     table = section.find('table')
244     if table and not management:
245         rows = table.find_all('tr')[1:]
246         for row in rows:
247             cols = row.find_all('td')
248             if len(cols) >= 2:
249                 management.append({
250                     'name': cols[0].get_text(strip=True),
251                     'position': cols[1].get_text(strip=True)
252                 })
253
254
255     return management
256
257
258     def _parse_financials(self, section) -> Dict[str, Any]:
259         """Parse financial highlights"""
260         financials = {}
261
262         table = section.find('table')
263         if table:
264             rows = table.find_all('tr')
265             for row in rows:
266                 cells = row.find_all(['th', 'td'])
267                 if len(cells) >= 2:
268                     metric = cells[0].get_text(strip=True)
269                     value = cells[1].get_text(strip=True)
270                     financials[metric] = value
271
272
273         return financials
274
275     def _extract_number(self, text: str) -> Optional[float]:
276         """Extract numeric value from text"""
277         if not text:
278             return None
279         # Remove non-numeric characters except decimal point

```

```

276     cleaned = re.sub(r'[^\\d.]', ' ', text.replace(',', ' '))
277     try:
278         return float(cleaned) if cleaned else None
279     except ValueError:
280         return None
281
282     def get_market_indices(self) -> List[Dict[str, Any]]:
283         """Scrape market indices data"""
284         indices = []
285         index_url = f"{self.base_url}/{self.language}/index"
286
287         soup = self._make_request(index_url)
288         if not soup:
289             return indices
290
291         index_cards = soup.find_all('div', class_='index-card')
292         for card in index_cards:
293             name_elem = card.find(['h3', 'h4'], class_='index-name')
294             value_elem = card.find('span', class_='index-value')
295             change_elem = card.find('span', class_='index-change')
296
297             if name_elem and value_elem:
298                 indices.append({
299                     'name': name_elem.get_text(strip=True),
300                     'value': self._extract_number(value_elem.get_text(strip=True)),
301                     'change': change_elem.get_text(strip=True) if change_elem else
302                         None
303                 })
304
305         return indices
306
307     def scrape_all(self) -> Dict[str, Any]:
308         """
309             Main scraping method - collects all available data
310         """
311
312         print("Starting MSE web scraping...")
313
314         result = {
315             'scraped_at': datetime.now().isoformat(),
316             'source': self.base_url,
317             'indices': [],
318             'companies': []
319         }
320
321         # Get market indices
322         print("\n1. Scraping market indices...")
323         result['indices'] = self.get_market_indices()

```

```

322
323     # Get listed companies
324     print("\n2. Scraping listed companies...")
325     companies = self.get_listed_companies()
326
327     # Get details for each company
328     print("\n3. Scraping individual company details...")
329     for i, company in enumerate(companies):
330         print(f"Processing {i+1}/{len(companies)}: {company.get('name', 'Unknown')}")
331
332         if company.get('detail_url'):
333             details = self.get_company_details(company['detail_url'])
334             if details:
335                 company.update(details)
336
337             result['companies'].append(company)
338
339     # Save results
340     self.save_to_json(result)
341
342     print(f"\nScraping complete. Total companies: {len(result['companies'])}")
343     return result
344
345 def save_to_json(self, data: Dict, filename: str = None) -> str:
346     """Save scraped data to JSON file"""
347     if filename is None:
348         timestamp = datetime.now().strftime('%Y%m%d_%H%M%S')
349         filename = f"mse_data_{timestamp}.json"
350
351     filepath = os.path.join(self.output_dir, filename)
352
353     with open(filepath, 'w', encoding='utf-8') as f:
354         json.dump(data, f, ensure_ascii=False, indent=2)
355
356     print(f"Data saved to: {filepath}")
357     return filepath
358
359
360 # Main execution
361 if __name__ == "__main__":
362     scraper = MSEScraper(output_dir="mse_extracted_data")
363
364     # Scrape all data
365     data = scraper.scrape_all()
366
367     # Print summary

```

```

368     print("\n" + "="*50)
369     print("SCRAPING SUMMARY")
370     print("="*50)
371     print(f"Market Indices: {len(data.get('indices', []))}")
372     print(f>Listed Companies: {len(data.get('companies', []))}")
373
374     # Print sample company
375     if data.get('companies'):
376         print("\nSample company data:")
377         print(json.dumps(data['companies'][0], indent=2, ensure_ascii=False))

```

Listing 2: Web Scraping from Mongolian Stock Exchange

6.2.4.3 Data Integration Script

The following script integrates data from both sources:

```

1 import json
2 from typing import Dict, List, Any, Optional
3 from datetime import datetime
4
5 class DataIntegrator:
6     """
7         Integrates data from Mongolia Open Data Portal and MSE
8         Links legal entity data with stock exchange information
9     """
10
11     def __init__(self):
12         self.legal_entities = []
13         self.mse_companies = []
14         self.integrated_data = []
15
16     def load_legal_entities(self, filepath: str) -> None:
17         """Load legal entities data from JSON file"""
18         with open(filepath, 'r', encoding='utf-8') as f:
19             self.legal_entities = json.load(f)
20             print(f"Loaded {len(self.legal_entities)} legal entities")
21
22     def load_mse_data(self, filepath: str) -> None:
23         """Load MSE data from JSON file"""
24         with open(filepath, 'r', encoding='utf-8') as f:
25             data = json.load(f)
26             self.mse_companies = data.get('companies', [])
27             print(f"Loaded {len(self.mse_companies)} MSE companies")
28
29     def find_matching_entity(self, mse_company: Dict) -> Optional[Dict]:
30         """
31             Find matching legal entity for MSE company
32             Uses name matching with normalization

```

```

33     """
34     mse_name = mse_company.get('name', '').strip().upper()
35
36     for entity in self.legal_entities:
37         entity_name = entity.get('name', '').strip().upper()
38
39         # Exact match
40         if entity_name == mse_name:
41             return entity
42
43         # Partial match (company name contained)
44         if mse_name in entity_name or entity_name in mse_name:
45             return entity
46
47     return None
48
49 def integrate(self) -> List[Dict[str, Any]]:
50     """
51     Integrate data from both sources
52     """
53     print("Starting data integration...")
54
55     for mse_company in self.mse_companies:
56         integrated_record = {
57             'mse_data': mse_company,
58             'legal_entity_data': None,
59             'integration_status': 'no_match',
60             'integrated_at': datetime.now().isoformat()
61         }
62
63         # Find matching legal entity
64         matching_entity = self.find_matching_entity(mse_company)
65
66         if matching_entity:
67             integrated_record['legal_entity_data'] = matching_entity
68             integrated_record['integration_status'] = 'matched'
69
70         self.integrated_data.append(integrated_record)
71
72         # Statistics
73         matched = sum(1 for r in self.integrated_data
74                         if r['integration_status'] == 'matched')
75         print(f"Integration complete: {matched}/{len(self.integrated_data)} matched")
76
77     return self.integrated_data
78

```

```

79     def save_integrated_data(self, filepath: str) -> None:
80         """Save integrated data to JSON file"""
81         with open(filepath, 'w', encoding='utf-8') as f:
82             json.dump(self.integrated_data, f, ensure_ascii=False, indent=2)
83             print(f"Integrated data saved to: {filepath}")
84
85
86 # Usage example
87 if __name__ == "__main__":
88     integrator = DataIntegrator()
89     integrator.load_legal_entities("extracted_data/legal_entities.json")
90     integrator.load_mse_data("mse_extracted_data/mse_data.json")
91     integrated = integrator.integrate()
92     integrator.save_integrated_data("integrated_data/combined_data.json")

```

Listing 3: Data Integration Script

6.2.5 Resources Classification

6.3 Design Decisions and Rationale

6.3.1 Choice of Data Sources

Decision: Use Mongolia Open Data Portal as primary source with MSE as supplementary source.

Strengths:

- Official government data ensures reliability and legal validity
- Direct JSON download eliminates complex parsing requirements
- Open data license permits unrestricted use for the project
- MSE data enriches information for publicly traded companies

Weaknesses:

- Government data may have update delays
- MSE website structure may change, requiring scraper maintenance
- No direct linkage keys between the two data sources

6.3.2 Choice of Extraction Methods

Decision: Direct API/download for Open Data Portal; web scraping for MSE.

Rationale: The Open Data Portal provides structured JSON access, making direct download the most efficient approach. MSE lacks a public API, necessitating web scraping with appropriate rate limiting to avoid server overload.

6.3.3 Data Integration Strategy

Decision: Name-based matching for linking MSE companies to legal entities.

Strengths:

- Simple implementation without requiring additional external data
- Effective for well-known listed companies with consistent naming

Weaknesses:

- May miss matches due to name variations
- Cannot automatically resolve ambiguous matches
- Manual verification recommended for critical linkages

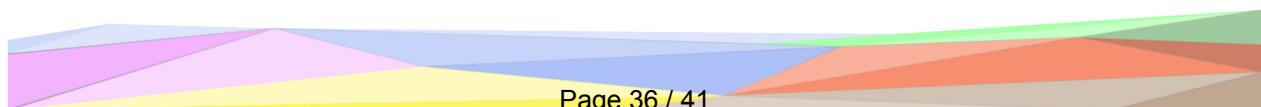
7 Language Definition

This section describes the Language Definition phase of the methodology adopted for the construction of the Knowledge Graph. As in the previous phase, the goal is to clearly describe the sub activities performed by the team members, as well as the outcomes produced. This phase is particularly important as it establishes the semantic foundations of the Knowledge Graph, defining how real-world entities and relationships are represented and ensuring that the resulting model is aligned with the project objectives.

Language Definition sub activities:

- The iTelos data Producer, in this phase, aims at fixing the language (concepts and words) used to represent the information required to satisfy the project purpose. Moreover, the resources collected in earlier phases are filtered to remove noisy or irrelevant data that do not contribute to the intended analysis. With these objectives, the knowledge and data resources are handled according to the following activities:
 - Concept identification
 - UKC alignment
 - Dataset filtering

The report of the work carried out during this phase also includes a description of the main design choices made by the team, together with their strengths and weaknesses. In this way, the reader is provided with a clear and transparent account of the reasoning process that guided the definition of the language and the selection of the data resources.



7.1 Concept Identification

This project focuses on the construction of a Knowledge Graph using open data from the State Registration of Legal Entities of Mongolia. The primary objective is to visualize, through a graph structure, the complex relationships among stakeholders within the Mongolian corporate ecosystem. In order to achieve this goal, the Concept Identification activity concentrates on selecting the real-world entities and relationships that are essential to represent corporate ownership, control, and governance.

The identified concepts reflect the main relationship categories addressed by the project. In particular, *Shareholders and Members* are modeled to represent individuals and organizations holding ownership stakes in legal entities, together with their classification and country of origin. *Officials and Controlling Entities* are identified as natural persons authorized to represent legal entities without a power of attorney, characterized by their official positions and appointment dates. *Ultimate Beneficial Owners* are modeled as natural persons who ultimately own or control legal entities, enabling transparency in corporate ownership structures.

For companies listed on the Mongolian Stock Exchange, additional concepts and relationships are introduced to link registration data with capital market participation. This integration allows the Knowledge Graph to provide a unified view of corporate information that spans both legal registration and market-related data.

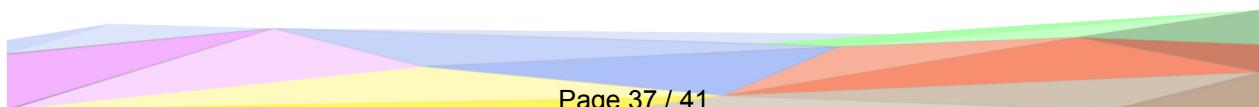
A major strength of this concept selection is its direct alignment with the project's transparency and analytical goals. A limitation is that the conceptual model is constrained by the level of detail and consistency available in the source data.

7.2 Dataset Filtering

The Dataset Filtering activity aims to ensure that only data resources relevant to the defined concepts and project objectives are retained for Knowledge Graph construction. Given the breadth of the available open datasets, a selective filtering process was necessary to reduce noise and improve semantic clarity.

Core datasets sourced from the Open Data Portal, including Legal Entity Basic Information, Shareholder Information, Authorized Representatives, and Ultimate Beneficial Owners, were retained as they are essential for modeling ownership, control, and representation relationships. Contextual datasets, such as Business Activities and Restructuring History, were also preserved to provide operational and historical context. For listed companies, datasets from the Mongolian Stock Exchange were filtered and retained only when they could be reliably linked to registered legal entities.

The main advantage of this filtering strategy is the creation of a focused and purpose-



driven dataset that directly supports the defined analytical questions. However, this approach may exclude certain auxiliary details that could be useful for future extensions of the Knowledge Graph. Despite this limitation, the filtered datasets provide a solid and coherent foundation for subsequent modeling and analysis phases.

8 Knowledge Definition

This section is dedicated to the description of the Knowledge Definition phase. Like in the previous section, it aims to describe the different sub activities performed by all the team members, as well as the phase outcomes produced.

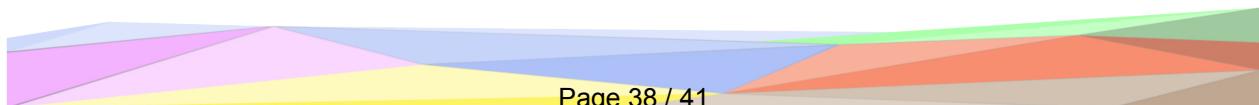
Knowledge Definition sub activities:

- The iTelos data Producer, in this phase, aims at defining the knowledge structure of the information to be considered to satisfy the project purpose. More in details, the producer process, in this phase, aims at defining the knowledge structure for each dataset (thus for each entity and properties) to be formalized, singularly. The data within such datasets, are then aligned with the structure of knowledge. The above objectives should be described by the following activities.
 - Teleology definition
 - Teleontology definition
 - Dataset cleaning and formatting

The report of the work done during this phase of the methodology, has to include also the description of the different choices made, with their strong and weak points. In other words the report should provide to the reader, a clear description of the reasoning conducted by all the different team members.

9 Data Definition

This section is dedicated to the description of the Data Definition phase. Like in the previous section, it aims to describe the different sub activities performed by all the team members, as well as the phase outcomes produced. In this phase the knowledge and data layers, composing the final KG, are merged to form a single data structure. The obtained result is a structured Knowledge Graph including both the two layers, and implicitly the language layer composed by the concepts and terms adopted to define the KG's teleontology.



Data Definition sub activities:

- The iTelos data Producer, in this phase, aims at merging the knowledge layer of a single dataset with the data values present within such a dataset, for each dataset collected. During this operation, the Producer has to consider the identification of the entities within each datasets, and, if different representation of the same entity exist, the Producer has to merge them. The above objectives should be described by the following activities.
 - Entity identification
 - Data mapping

The report of the work done during this phase of the methodology, has to includes also the description of the different choices made, with their strong and weak points. In other words the report should provide to the reader, a clear description of the reasoning conducted by all the different team members.

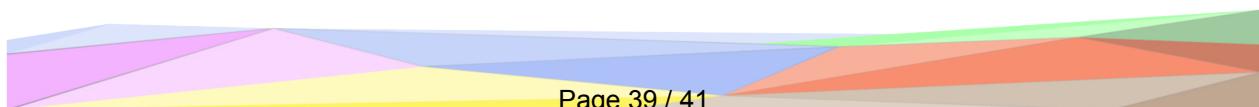
10 Evaluation

This section aims at describing the evaluation performed at the end of the whole process (producer plus consumer) over the final outcome of the iTelos methodology. More in details, this section as to report:

- the final Knowledge Graph information statistics (like, number of etypes and properties, number of entities for each etype, and so on).
- Knowledge layer evaluation: the results of the application of the evaluation metrics applied over the knowledge layer of the final KG.
- Data layer evaluation: the results of the application of the evaluation metrics applied over the data layer of the final KG.
- Query execution: the description of the competency queries executed over the final KG in order to test the suitability of the KG to satisfy the project purpose.

11 Metadata Definition

In this section the report collects the definitions of all the metadata defined for the different resources produced along the whole process (producer and consumer). The metadata defined in this phase describes both the final outcome of the project, and the intermediate



outcome of each phase.

The definition of the metadata, is crucial to enable the distribution (sharing) of the resource produced. For this reason it is important to describe also where such metadata will be published to distribute the resources it describes (for example the DataScientia catalogs).

In particular the structure of this section is organized as follows, with the objective to describe the metadata relative to all the type of resources produced by the project.

- Language resources metadata description
- Knowledge resources metadata description
- Data resources metadata description

12 Open Issues

This section concludes the current document with final conclusions regarding the quality of the process and final outcome, and the description of the issues that (for lack of time or any other cause) remained open.

- Did the project respect the scheduling expected in the beginning ?
- Are the final results able to satisfy the initial Purpose ?
 - If no, or not entirely, why ? which parts of the Purpose have not been covered ?

Moreover, this section aims to summarize the most relevant issues/problems remained open along the iTelos process. The description of open issues has to provide a clear explanation about the problems, the approaches adopted while trying to solve them and, eventually, any proposed solution that has not been applied.

- which are the issues remained open at the end of the project ?

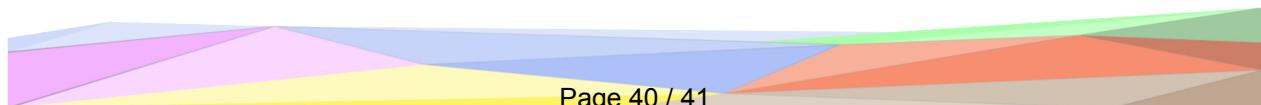


Table 10: Data Resources Classification

| Data Resource | Source | Classification | Justification |
|----------------------------|------------------|----------------|--|
| Legal Entity Basic Info | Open Data Portal | Core | Essential identification data for all entities |
| Shareholder Information | Open Data Portal | Core | Central to ownership visualization purpose |
| Authorized Representatives | Open Data Portal | Core | Key governance and representation data |
| Ultimate Beneficial Owners | Open Data Portal | Core | Critical for transparency objectives |
| Business Activities | Open Data Portal | Contextual | Supporting information for entity characterization |
| Restructuring History | Open Data Portal | Contextual | Historical context for entity changes |
| Stock Trading Data | MSE | Contextual | Supplementary market information |
| Company Financial Data | MSE | Contextual | Additional context for listed companies |