Linked Open Data-based Knowledge Management Process for Biodiversity

Rathachai Chawuthai

DOCTOR OF PHILOSOPHY

Department of Informatics
School of Multidisciplinary Science
SOKENDAI (The Graduate University for Advanced Studies)

September, 2016

Linked Open Data-based Knowledge Management Process for Biodiversity

Author:
Rathachai Chawuthai

Supervisor:
Hideaki Takeda

DOCTOR OF PHILOSOPHY

Department of Informatics
School of Multidisciplinary Sciences
SOKENDAI (The Graduate University for Advanced Studies)

2016

A dissertation
submitted to the department of Informatics
and the Committee on graduate studies of
SOKENDAI (The Graduate University for Advanced Studies)
In partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Advisory Committee

Prof.	Hideaki	Takeda	National Institute of Informatics, SOKENDAI
Prof.	Ken	Satoh	National Institute of Informatics, SOKENDAI
Assoc. Prof.	Ryutaro	Ichise	National Institute of Informatics, SOKENDAI
Assoc. Prof.	Asanobu	Kitamoto	National Institute of Informatics, SOKENDAI
Assoc. Prof.	Ikki	Ohmukai	National Institute of Informatics, SOKENDAI
Assoc. Prof.	Takahiro	Kawamura	The University of Electro-Communications

Approved for the University Committee on Graduate Studies.

ABSTRACT

Thanks to the advancement of Linked Open Data (LOD) technology, it enables data around the world to become explicitly and implicitly exchangeable through the Internet, so the motivation to empower the capability of Knowledge Management (KM) for global accessibility comes to be achievable. However, the shift is not just simply replacing a legacy storage with a graph database. There are challenging issues about how to deal with the change in knowledge, how to have human-/machine-readable knowledge graphs, how to realize that the structure of a knowledge graph is suitable for finding new knowledge, and how to have learners learn knowledge from a knowledge graph conveniently. Thus, this thesis takes this opportunity to study the role of LOD in any KM process through some key KM activities by using biodiversity as domain knowledge. For the Ph.D. course, knowledge capture, knowledge exchange, knowledge discovery, and knowledge presentation are studied; because they are common KM activities of and primarily found in any KM processes. To inform how importance the LOD in KM process is, three projects are introduced for the according activities.

Linked Taxonomic Knowledge (LTK) takes responsibility of capturing and exchanging of the knowledge by initiating an RDF data model to capture the change in biological taxonomy with appropriate context and publish the data with a simple and lightweight structure. This project introduces three types of taxonomic identifiers for the lightweight representation. There are Nominal Entity, Simple Nominal Entity, and Contextual Nominal Entity. The Nominal Entity is the broader term of taxon concepts and taxonomic names; the Simple Nominal Entity is used for encapsulating a taxon concept and its scientific name within a single URI; and the Contextual Nominal Entity is the versioning representation of the Nominal Entity with a single URI by including a taxon concept, its scientific name, and an aspect of time. For presenting taxonomic knowledge in a variety of uses, this project also initiates three types of knowledge graphs that are Event-Centric model, Transition model, and Snapshot model. First, The Event-Centric model is created for capturing the change in taxonomy. It includes operations of change, relations between background knowledge of the changes, aspects of time, and references. Next, the Transition model is used to present the chronological change between taxa. Last, the Snapshot model presents the temporal information of a taxon. The Event-Centric model is designed for presenting the change in taxonomy, so the model is complex by design but flexible for any other applications. Thus, the Transition model and the Snapshot model are designed to be lightweight knowledge representation for exchanging data with LOD cloud.

Link Prediction on Interspecies Interaction (LPII) is accounted for the knowledge discovery by analyzing the knowledge graph of fungus-host interactions, and then gives biologists a recommended list to discover more interspecies interactions. This project introduces a hybrid recommender system including scoring functions based on Collaborative Filtering, Community Structure, and Biological Classification. In order to capitalize on knowledge graphs, the LPII makes prediction on the basis of a bipartite graph, a projection network, and a taxonomy. These scoring functions work with the bipartite graph, the projection network, and taxonomy getting from LOD respectively. It has been found that the linear combination of the three scoring functions is more accurate than other combinations, and some missing relations have been found from the new discovery of the National Museum of Nature and Science in Tokyo and some literatures from external resources.

Biodiversity Knowledge Graph Visualization (BViz) is in charge of the knowledge presentation by simplifying and rearranging a knowledge graph for delivering a human-friendly node-link diagram to users. BViz introduces three modules that are Graph Simplification, Triple Ranking, and Property Selection. First, the Graph Simplification uses newly introduced Semantic Web rules to merge some same-as nodes, removing inferred transitive links, and eliminating the chain of inferred type-hierarchy. Second, the Triple Ranking helps to reorder all triples in the query graph from common information to topic-specific information. Last, the Property Selection allows users to display or hide some triples containing selected URIs or namespaces. A web application is implemented on the basis of these proposed methods, and all of them can be controlled by users via the interactive user interface. Thus, learners can query a knowledge graph, simplify the graph, rearrange the graph from common information to topic-specific information, and select some parts of the graph based on user preference.

The results of these projects demonstrate that the combination of LOD and KM can address the issues and provides contributions to biodiversity domain by both publications and applications. Moreover, the knowledge and experience gained from this study are intended to be a guideline for creating LOD-based KM systems to any other domains.

.....

To my family, teachers, and friends.

ACKNOWLEDGEMENT

Firstly, I would like to express my sincere gratitude to my advisor Prof. Hideaki Takeda for the continuous support of my doctoral study and related research, for his patience, motivation, and immense knowledge. The door to his office was always open whenever I ran into a problem spot or had a question about my research or development. He consistently allowed this thesis to be my own work, but steered me into the right direction whenever he thought I needed it. His guidance helped me during my time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my study.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Ken Satoh, Prof. Ryutaro Ichise, Prof. Asanobu Kitamoto, Prof. Ikki Ohmukai, and Prof. Takahiro Kawamura, for their insightful comments and encouragement, but also for the hard questions that inspired me to widen my research from various perspectives.

My sincere thanks also go to Dr. Tsuyoshi Hosoya, and Dr. Utsugi Jinbo who provided me an opportunity to learn biodiversity knowledge and access to some useful datasets of National Museum of Nature and Science, Tokyo, Japan. Without they precious support, it would not be possible to conduct this research.

I would like to show my great respect to all teachers for fundamental knowledge and hand-on experience, especially in Prof. Vilas Wuwongse for the fundamental background of Semantic Web and inspiration of organizing Linked Open Data for the academic domain.

Thanks to LODAC's and LODI's members who provided me an opportunity to join their team and gave access to the laboratory and research facilities. Also thanks to NII's members for technical support, nice recommendations, and friendship.

I also would like to thank Dr. Sirod Sirisup, Dr. Kalika Suksomboon, Dr. Rémy Cazabet, Dr. Vorapong Suppakitpaisarn, Ms. Natthaphat Akharathanaphan, and anyone whose name is not written here for additionally useful comments and supports. Thanks to colleagues from Thomson Reuters and Punsarn Asia for every fundamental technological knowledge and practice, and thanks to all my friends for making me enjoy my time during my Ph.D. course.

I gratefully acknowledge the funding sources from MEXT Honors, NII, SOKENDAI, and Laboratory that make my Ph.D. work possible and opportunity to learn Japanese culture.

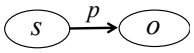
Last but not least, I must express my very profound gratitude to my family members for providing me with love, support, and continuous encouragement throughout my years of studying, researching, and writing this thesis. This accomplishment would have not been possible without them. Thank you.

Rathachai Chawuthai

Rathachai Chawuthai
Author

.....

LIST OF NOTATIONS

Notation	***Meaning***
$\langle s,p,o \rangle$	An expression of a triple including a subject (s), a predicate (p), and an object (o)
	A diagram of a triple including a subject (s), a predicate (p), and an object (o)
$\{ s_1\ p_1\ o_1.\ s_2\ p_2\ o_2.\ \}$	An expression of an RDF graph including one or more triples.
$P \rightarrow Q$	A proposition P implies a proposition Q; or if P then Q.
$f: X \rightarrow Y$	Function f maps a set X into a set Y.
$A \models B$	A sentence A entails a sentence B, that is in every model in which A is true, B is also true.
$\equiv$	Concept equivalence
$\sqsubseteq$	Concept inclusion
$\top$	Every individual as an instance
$\bot$	An empty concept
$\wedge$	Logical conjunction (and)
$\vee$	Logical disjunction (or)
$\cap$	Set-theoretic intersection
$\cup$	Set-theoretic union
$\subset$	Subset
$\in$	Set membership
2^X or $P(X)$	The power set of the set X.

.....

GLOSSARY OF ABBREVATIONS

Abbreviation	*Stands for*
BViz	Biodiversity Knowledge Graph Visualization *
KM	Knowledge Management
LPII	Link Prediction on Interspecies Interaction *
LTK	Linked Taxonomic Knowledge *
RDF	Resource Description Framework
RDFS	RDF Schema
OWL	Web Ontology Language
LOD	Linked Open Data
SPARQL	SPARQL Protocol and RDF Query Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
HTTP	Hypertext Transfer Protocol

Note: the star symbol (*) indicates an abbreviation newly introduced in this thesis.

……

NAMESPACES

Prefix	*Namespace (Reference)*
bibo:	http://purl.org/ontology/bibo/ (*Bibliographic Ontology* [133])
cka:	http://www.cka.org/2012/01/cka-onto# (*Contextual Knowledge for Archives* [22])
dct:	http://purl.org/dc/terms/ (*Dublin Core Terms Namespace* [137])
dbpedia:	http://live.dbpedia.org/resource/ (*DBpedia Namespace* [72])
dwc:	http://rs.tdwg.org/dwc/terms/ (*Darwin Core* [138])
foaf:	http://xmlns.com/foaf/0.1/ (*Friend of a Friend Ontology* [139])
gbif:	http://www.gbif.org/species/ (*Global Biodiversity Information Facility* [140])
genus: *or* ge:	http://rc.lodac.nii.ac.jp/taxon/genus/ (*Namespace for genera*)
lodac:	http://lod.ac/species/ (*LODAC Species* [146])
ltk:	http://rc.lodac.nii.ac.jp/ns/ltk# (*Linked Taxonomic Knowledge Ontology*)
skos:	http://www.w3.org/2004/02/skos/core# (*Simple Knowledge Organization System Namespace* [150])
species: *or* sp:	http://rc.lodac.nii.ac.jp/taxon/species/ (*Namespace for species*)
subspecies: *or* ssp:	http://rc.lodac.nii.ac.jp/taxon/subspecies/ (*Namespace for subspecies*)
soic:	http://rdfs.org/sioc/ns# (*Semantically-Interlinked Online Communities Core Ontology* [151])
tl:	http://purl.org/NET/c4dm/timeline.owl# (*Timeline Ontology* [157])
tmo:	http://www.yso.fi/onto/taxmeon/ (*Meta-Ontology of Biological Name* [118])

.....

LIST OF PUBLICATIONS

Journal Paper

Rathachai Chawuthai, Hideaki Takeda, Vilas Wuwongse, and Utsugi Jinbo:
"Presenting and preserving the change in taxonomic knowledge for linked data."
In: Special Issue on Semantics for Biodiversity -
Semantic Web Journal, vol. 7, no. 6, pp. 589-616. (2016)

International Conference Papers

Rathachai Chawuthai, and Hideaki Takeda:
"RDF Graph Visualization by Interpreting Linked Data as Knowledge"
In: Semantic Technology - The 5th Joint International Sematic Technology Conference (JIST 2015), Yichang, China, November 11-13, 2015.
Springer International Publishing, pp. 23-39. (2015)

Rathachai Chawuthai, Hideaki Takeda, and Tsuyoshi Hosoya:
"Link Prediction in Linked Data of Interspecies Interactions Using Hybrid Recommendation Approach."
In: Semantic Technology - The 4th Joint International Sematic Technology Conference (JIST 2014), Chiang Mai, Thailand, November 9-11, 2014.
Springer International Publishing, pp. 113-128. (2014)

International Conference Poster

Rathachai Chawuthai, and Hideaki Takeda:
"*rSim*: Simplifying an RDF Graph at the Visualization Tier for Non-Expert Users"
In: The 14th International Semantic Web Conference (ISWC 2015), Bethlehem, Pennsylvania, United States, October 11-15, 2015.
CEUR Workshop Proceedings. (2015)

.....

TABLE OF CONTENTS

.....

LIST OF TABLES

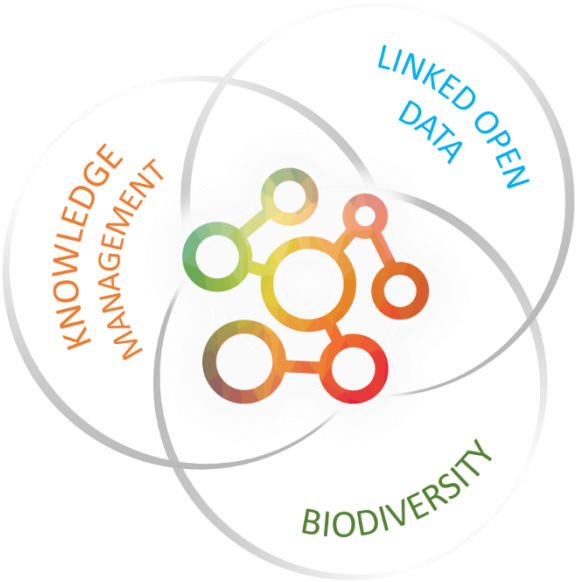
.....

LIST OF FIGURES

.....

CHAPTER 1

INTRODUCTION



Thanks to the power of Internet, Semantic Web, and Linked Open Data technologies, data around the world have opportunity to be connected and can be formed the world knowledge graph. This situation can enhance the capability of knowledge management, because the linked open data can play a key role in some activities of knowledge management process such as knowledge capture, exchange, discovery, and presentation. The domain of biodiversity informatics is primarily investigated, because it has available graph data that has high quality and are globally accessible. For this reason, the goal of this doctoral thesis is to study the role of linked open data in knowledge management process for biodiversity informatics. The overall idea of this study is introduced in this chapter.

1.1. Background

Saying “knowledge is a key to success” remains true throughout time. Knowledge is considered to be a key resource for every community, every country, every industry, and every business [31]. Many organizations have advanced themselves to stand at the advantage position in the competitive environment because they place highly important to the management on their organizational knowledge [11, 93]. Humans themselves can discover knowledge by observing, analyzing, discussing, etc. Knowledge from humans (tacit knowledge) can be written into any medias such as books, voices, videos, etc. On the other hand, knowledge recorded in media (explicit knowledge) can be transferred to humans by learning. The transformation between tacit knowledge and implicit knowledge is commonly found in academia and industry every day. The effective Knowledge Management (KM) leads to the creation of new knowledge and new innovation, and gives successful results [11, 93].

Especially in an education system, it is known that knowledge is considered as a highly important key resource. Knowledge is discovered, captured, exchanged, and presented every day from primary schools to research institutes. Biology class is a good example because it is so close to everyone during K12 [8]. Students firstly gain biological knowledge by attending lectures and reading books. After that, a teacher assigned them to observe the nature such as keeping eye on the growth of some animals or plants. Finally, they have to write a report and share with other classmates. Moreover, some students may find out some interesting points among other reports and raise some questions. Because of some arisen questions, they may have discussion and conclude into a new knowledge. In terms of KM process, this simple story can be viewed that knowledge is exchanged during teaching, discussing, and sharing the reports; knowledge is discovered during making observation and summarizing two or more reports; knowledge is captured when writing reports; and knowledge is presented when reading reports. This example demonstrates a simple story of KM process, and more details are described in Section 2.1.

In fact, moreover, we are able to acquire more knowledge from libraries or documentaries. It means that learning is not limit to any classrooms, but it can be done from other sources of knowledge. Thanks to the Internet technology, we have more opportunity to access more online contents because many organizations such as some academic institutes, archives, communities, governments, industries, libraries, and museums trend to give open access to their contents [72, 136, 140, 146, 147, 148]. It is commonly known that learning from multiple sources makes learners gain more precise understanding of their interesting subjects, so associating all pieces of knowledge around the world gives a great benefit to learners [130]. Therefore, managing the worldwide knowledge in the right direction can improve the value of knowledge and learning ability of learners in long term.

1.2. Motivation

Having the knowledge management on worldwide data is very challenging because any different formats, schemas, structures, and the interpretations of data from different sources become the obstacle of linking data. Fortunately, the Linked Open Data (LOD) technology, that creates semantic association among Internet resources, let us begin to see the possibility of having the knowledge management on the Internet data [16, 50]. LOD, which is described in Section 2.2, shows that data can be implicitly and explicitly linked, and a knowledge graph can be built. Using the key features of LOD including graph-structured data, schemas and

ontologies, reasoning, and query; the activities [11, 75] of KM process can be improved expectedly in the knowledge exchange.

To have a qualified knowledge graph, data must be atomic and structured [16, 54]. Most of linked data at the moment such as data from libraries, archives, and museums are implemented at the metadata level [120]. However, atomic and structured data inside content that are required for building a knowledge graph are rarely to be found. In consequence, for this thesis, we have considered to study on the domain of biodiversity informatics, which is detailed in Section 2.3, by the following reasons.

- Data about organism groups are atomic and structured such as biological classification, taxonomic concept description, interspecies interaction, food web, etc. [122].
- There are online repositories and some of them are graph data, for example LODAC [146], GBIF [140], uBio [106], Catalog of Life [63], ZooBank [100], MycoBank [25], etc.

In addition, biodiversity knowledge is not static knowledge, but it is dynamic knowledge, and waiting for more discovery [122]. This dynamic behavior is also found in other domains. Thus, biodiversity knowledge is a suitable domain to demonstrate the role of LOD in KM process, and this study can be a guideline for other domains when their data are ready to be constructed as knowledge graphs.

1.3. The Big Picture

We have introduced the wide perspective idea about KM process in Section 1.1 using the example of learning in the biology class. In this section, we adopt some approaches, concepts, and model from Nonaka [93], Liebowitz [75], and Becerra-Fernandez & Sabherwal [11]; and the overall idea about KM process for the context of biodiversity informatics is demonstrated in Fig. 1.1. It is noted that this figure shows a potential KM process that is observed from the biodiversity domain, but it is not the final solution. It can be improved in the future according to progress of requirements and technologies. Since this thesis studies how LOD support KM process, the wide scape of knowledge management for biodiversity has to be drawn in order to have readers view the same goal as our intention.

In the figure, first, there are three stakeholders: **Nature** can refer to any ecosystem or natural living environment; **Human** can refer to any people who consume and provide biodiversity knowledge; and **Machine** can refer to any computer systems that are developed to deal with biodiversity knowledge.

Second, **Factual evidence** is anything that occurs in nature for example the appearance and behaviors of organisms. There are two kinds of knowledge discussed in this study: **Tacit knowledge** is knowledge embodied in people such as knowing about organismal groups; and **Explicit knowledge** is knowledge stored in media by human such as images and publications.

Next, factual evidence, tacit knowledge, and explicit knowledge are transformed to each other using the following KM process [11, 75, 93]. In this case, the explicit knowledge is presented by knowledge graph in RDF format.

- **Observation** is an active activity of human for exploring more factual evidence from the nature. Any found factual evidence can be transformed into tacit knowledge in human by this step [8].

- **Socialization** is an activity of human for sharing and discussing their tacit knowledge. It can happen in any class rooms, meeting rooms, workshops, laboratories, etc. where humans can participate each other [93]. New knowledge could be created here if they found some new conclusions during their discussion.
- **Externalization** is an activity for transforming tacit knowledge into explicit knowledge [11, 93]. This thesis uses the term ***Knowledge Capture***. It captures knowledge from human into an RDF graph using some following approaches.
    - *Data Modelling* for being schema of biodiversity data in RDF [146, 153],
    - *Data Preservation* for capturing changes in biodiversity knowledge [70, 118],
    - *Natural Language Processing* (NLP) for converting unstructured text into RDF graph [53],
    - *Input Interfaces* that can be a user interface or a service interface,
    - etc.
- **Combination** is an activity that associates some pieces of explicit knowledge from multiple sources. The new summarization of the data combination can create new knowledge [11, 93]. This research separates this activity into knowledge exchange and knowledge discovery. First, ***Knowledge Exchange*** shares or transfers explicit biodiversity knowledge across taxonomic repositories using the following technologies.
    - *Linked Data* that enables connection among pieces of data via the Internet using the power of Semantic Web [50],
    - *Ontology Matching* that finds correspondences between resources from different ontologies semantically [113],
    - *Information Retrieval* that identifies and ranks relevant resources for satisfy a query expression [112],
    - *Trust* management that measures the correctness and the quality of data in order to select highly respected triples and identify some inconsistent data when a lot of data are combined from multiple sources [5],
    - etc.

  Second, ***Knowledge Discovery*** creates new explicit biodiversity knowledge from the synthesis of prior knowledge [11]. There are several approaches to support this activity.
    - *Link Prediction* that is an approach to find some potentially missing links using mathematical methods [83],
    - *Ontology Matching* [113],
    - *Trust* [5],
    - etc.
- **Internalization** is an activity that transforms explicit knowledge into tacit knowledge. Individuals can acquire tacit knowledge by reading some contents or making some experiments through simulations [11, 93]. This thesis uses the term ***Knowledge Presentation*** for delivering RDF data from a computer to human perception. There are many ways to use and present RDF data, for example:
    - *Graph Diagram* (node-link or concept-map diagram) that presents the overview of concepts in a specific topic using the relationships between entities and named links [80, 94, 110],
    - *RDF to Text* that synthesizes human-readable text from a set of triples [32],
    - *Application* that uses knowledge to take some actions such as decision making systems and e-learning systems [11],
    - etc.

Any actions for knowledge management can be done by human-nature, human-human, human-computer, and computer-computer interactions. However, the field of informatics can

take responsibility for any tasks related to computer systems that are encircled in a blue dash line in Fig. 1.1.

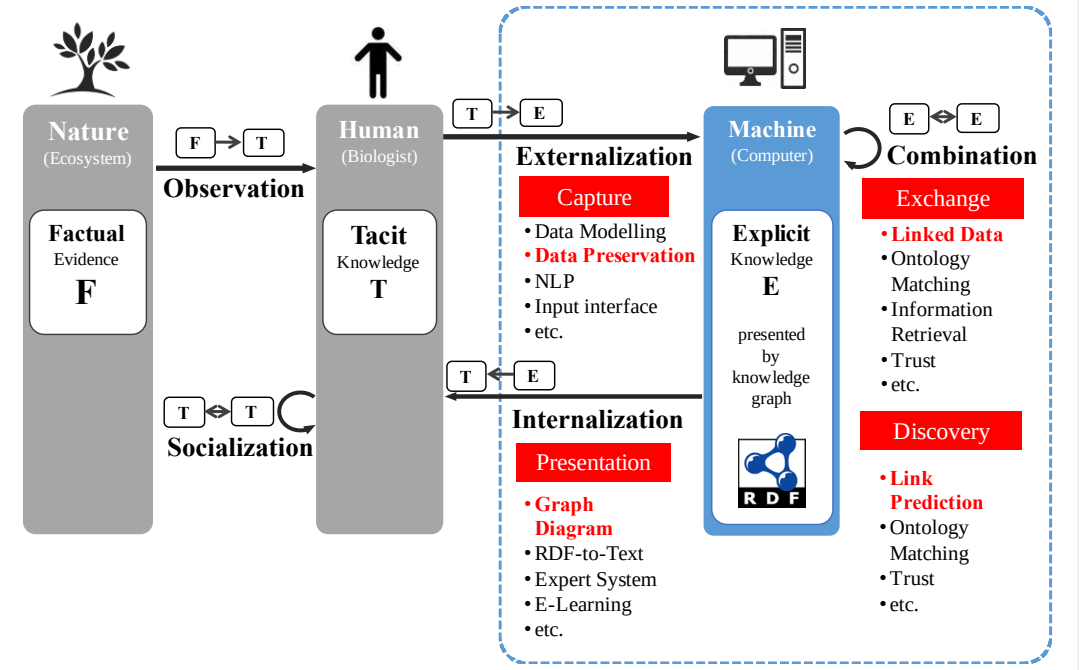


Fig. 1.1: Overall of biodiversity knowledge management activities.
- **F** is Factual evidence that is occurred in the nature or ecosystem.
- **T** is Tacit knowledge that human has a justified true belief about the nature.
- **E** is Explicit knowledge that is transformed from tacit knowledge and recorded in any medias. In this study, the explicit knowledge is presented by knowledge graph stored in computers.
- A blue dash line encircles topics under the ability of the informatics.
- Red boxes and red texts are topics under the scope of this study.

In order to have a clear understanding of the overall picture of this study, a scenario for demonstrating the usability of a biodiversity KM system is described using the study of biodiversity under the topic of fungi. It includes the process of learning knowledge about fungi, and observing fungal distribution [95]. The scenario is described by the following steps.

1) In general, fungi are growing by parasitizing some animals, plants, or other fungi that provide appropriate environment including suitable temperature, nutrient support, chemical substance, moistness, etc. It means that some groups of fungi are more likely to parasitize some specific hosts.
2) Biologists recognize the characteristics, behaviors, features, interactions, etc. of fungi and hosts from observation or experiment in laboratories. In this case, tacit knowledge of biologists such as taxonomy is increasing.
3) Biologists summarize knowledge from their observation into knowledge graphs and preserve in a KM system using ***knowledge capture***. This step shows that the tacit knowledge of biologists is transformed into explicit knowledge in a computer.
4) Biologists do observation continuously. In case some evidences are discovered and they result in the improvement of knowledge, they update the changes in knowledge into the KM system again using ***knowledge capture***.

5) Thus, the KM system contains pieces of change in biodiversity knowledge that are taken from ***knowledge capture***, and the knowledge graphs are updated according to the recently added changes.
6) Thanks to the interoperability of knowledge graphs, pieces of diversity knowledge are easily distributing with other KM systems via the Internet using ***knowledge exchange***. Thus, explicit knowledge is increasing.
7) When biologists have a plan to observe fungi again, they can use the knowledge graph as a knowledge base for predicting some potential interactions between fungi and hosts. In this case, ***knowledge discovery*** employed some up-to-date knowledge graphs from other repositories for improving a prediction model and making prediction.
8) Biologists use the predicted list of fungus-host interactions from ***knowledge discovery*** as a guideline for making observation. When they have new discovery, they update some new pieces of knowledge again into the KM system using ***knowledge capture***. At this step, explicit knowledge is increasing.
9) After that, the system can deliver the knowledge graphs to biologists using ***knowledge presentation***. When they learn biodiversity knowledge from the knowledge graphs, explicit knowledge in the computer is transformed into the tacit of biologists again.
10) Later, biologists can use knowledge from the KM system to be an instruction for observing the nature and updating new discovery into the KM system continuously.

This scenario shows the big picture of the uses of a KM process in the biodiversity domain. After conducting the research for these KM activates, this scenario is discussed again in Section 7.4.2 using the solutions that are studied and developed in this thesis.

1.4. Problem Statement

As we previously described, adopting LOD in KM systems give an advantage over traditional databases, because it can unlock the capability of exchanging knowledge from different structures among systems around the world through the Internet. This will result in the learners to have more opportunity to obtain wider and deeper knowledge from multiple sources.

Biodiversity domain is a good start because there are atomic and structured data provided by some online repositories and some of them are distributed in RDF format. These data are attempted to be ingested into KM processes, however we have faced with some challenging issues, which are also detailed in Section 2.4, as follows:

- Biodiversity knowledge is not stable and it is commonly changed due to the new discovery and the perspective of biologists. When expressing the change in knowledge, it has to concern about provenance metadata such as time, contributors, consequences, etc. However, RDF structure, which is a binary relationship, is not designed for embedding more context in a single triple. Capturing the change in taxonomy must concern about a proper way to include the context of a change. Moreover, the proposed solution should be implemented with the present-day tools or systems, and be used practically.
- Due to the change in biodiversity knowledge, there are a lot of obsoleted names, accepted names, synonyms, homonyms, etc.; linking data using scientific names alone is not proper for this situation. Since the clients are both machines and non-computer-expert users, the data structure should be lightweight and be readable by both humans especially in non-computer-expert users and computers.

- Since biologists always discover more knowledge, the utilization of the existing knowledge can create a suitable direction for helping them observe more factual evidence. Discovering more fungus-host interactions is an interesting topic because it can be applied for solving some real-world problems such as the protection of crop diseases. However, this kind of data is very sparse by nature. Using a single prediction method is not enough for making a precise result, so the combination of various techniques is likely to improve the prediction accuracy. In this case, LOD is considered, because the graph structure can supply various features for making link prediction.
- Learning biology from a node-link diagram or a concept-map diagram can enhance the learning ability of learners. Since a RDF dataset is a graph structure, visualizing the RDF diagram is possible to do. However, in practice, the generated graph diagram is too complex to be read due to many inferred triples resulting from the reasoning process. The other problem is that the graph visualization does not give the reading flow to learners, so readers cannot read from common information to topic specific information step by step. Thus, the graph has to be simplified and triples should be well-arrangement in order to be convenient to read by users.

1.5. Scope

As we discussed, there are some challenging issues for having a complete LOD-based KM system. For this thesis, we intend to study the role of LOD in KM process under the domain of biodiversity informatics. The possibility, the feasibility, and the suitability for having a RDF-based knowledge graph in KM process are demonstrated by showing some problems, proposed solutions, and outcomes. Nevertheless, we do not aim to implement a complete LOD-based KM system, create a new KM process for biodiversity domain, preserve knowledge at the hardware level, or create any full-functioned applications such as decision support systems and e-learning systems.

As shown in the red boxes and red texts in Fig. 1.1, we are focusing on the study of knowledge capture, knowledge exchange, knowledge discovery, and knowledge presentation. These four activities are directly related to computer systems, and they are important activities that are mentioned from the classic model of knowledge creation spiral [93] to the newer model of KM systems and processes [11]. In detail, since we need to satisfy some real-world matters in biodiversity domain, we scope our work on the following issues.

- For the knowledge capture, we focus on capturing the change in taxonomy in RDF.
- For the knowledge exchange, we focus on exchanging taxonomic knowledge that arises from the change in taxonomy.
- For the knowledge discovery, we focus on predicting potential interaction links among species.
- For the knowledge presentation, we focus on using graph diagram visualization for presenting taxonomic knowledge.

In addition, in this study, the term knowledge is generally referred to explicit knowledge. Since tacit knowledge cannot be totally expressed, the term tacit knowledge in this work is scoped to only explicatable tacit knowledge that can be written in an RDF expression.

1.6. Objectives

For studying the role of LOD in biodiversity KM process, we aim to accomplish the following objectives.

1) To find out an appropriate data model for capturing the change in taxonomic knowledge in RDF.
2) To find out a suitable data model and a method for globally exchanging taxonomic data, that are resulted from the change in taxonomic knowledge, with a simple and lightweight expression.
3) To demonstrate that the integration of some features from LOD can improve the accuracy of a recommender model for discovering potential links of interspecies interactions.
4) To find out a process that can present a simplified and well-rearranged RDF graph visualization for conveniently reading by non-semantic-web-expert users.

1.7. Contribution

To achieve the research objectives, three projects are created. They help to demonstrate the role of LOD in biodiversity KM process in several viewpoints.

Linked Taxonomic Knowledge (LTK)

Account to:	Knowledge Capture and Knowledge Exchange
Description:	LTK is a web application that provides functions for preserving the change in taxonomic knowledge, and presenting the change together with the temporal information of organismal groups in the RDF format. It also delivers a user interface for displaying the historical data of a taxon concept together with contextual information of any changes; and serves as a SPARQL endpoint when working with other machines.
Outcome:	LTK shows that the event of change in biodiversity knowledge can be captured and expressed as RDF format, and the event model can be transformed in to a simple model for working well with data in LOD cloud. Moreover, LTK shows that it is possible to have an identifier including human readable name and context within a single URI, so the linked data model is simple, lightweight, and easy to read by both machines and non-semantic-web-expert users. This project supports the practicability to employ LOD to enhance the knowledge capture and knowledge exchange activities.
Publication:	Rathachai Chawuthai, Hideaki Takeda, Vilas Wuwongse and Utsugi Jinbo. *"Presenting and preserving the change in taxonomic knowledge for linked data."* Special Issue on Semantics for Biodiversity - Semantic Web Journal. IOS Press, Volume 7, Number 6, Pages 589-616. (2016)
Application:	`http://rc.lodac.nii.ac.jp/ltk/`

Link Prediction on Interspecies Interaction (LPII)

Account to: Knowledge Discovery

Description: LPII is a project that supports the National Museum of Nature and Science (KAHAKU), Japan to predict some potential interactions between fungi and hosts in order to give awareness to agriculturalists for preventing their farms from crop diseases caused by fungal attacks. This project introduces a hybrid recommender system that uses the calculation based on collaborative filtering, community structure in a network, and biological classification.

Outcome: LPII shows that the graph structure of interspecies interaction and some contents from other RDF repositories contribute to the improvement of the recommender model. This project supports the possibility to consider LOD to enhance the knowledge discovery activity.

Publication: Rathachai Chawuthai, Hideaki Takeda, and Tsuyoshi Hosoya.
"Link Prediction in Linked Data of Interspecies Interactions Using Hybrid Recommendation Approach."
Semantic Technology - The 4th Joint International Sematic Technology Conference (JIST 2014), Chiang Mai, Thailand, November 9-11, 2014.
Springer International Publishing,
Pages 113-128. (2014)

Application: `http://rc.lodac.nii.ac.jp/txi/`
Note: This application is the demonstration of the relationship between fungi and hosts for data analysis, but the result of the proposed hybrid recommender model is not involved in this application.

Biodiversity Knowledge Graph Visualization (BViz)

Account to: Knowledge Presentation

Description: BViz is a web application that allows users to simplify and rearrange a query graph based on the provided three key functions: Graph Simplification, Triple Ranking, and Property Selection. These functions used the knowledge structure of Semantic Web together with statistical analysis for making an RDF graph to be more easy readable. Users can customize each function conveniently for learning knowledge from an appropriate node-link diagram.

Outcome: BViz shows that it can simplify and rearrange a query graph to generate a node-link diagram in different information levels for learners to read appropriately. This project supports the practicability to use LOD as a part of the knowledge presentation activity in order to enhance the ability to learn biology.

Publication: Rathachai Chawuthai, and Hideaki Takeda.
"RDF Graph Visualization by Interpreting Linked Data as Knowledge"
Semantic Technology - The 5th Joint International Sematic Technology Conference (JIST 2015), Yichang, China, November 11-13, 2015.
Springer International Publishing,
Pages 23-39. (2015)

Application: `http://rc.lodac.nii.ac.jp/rdf4u/`
Note: This application is not limited to the biodiversity domain but also designed for general uses.

The result of these projects demonstrates that LOD can play an important role in biodiversity KM process by showing that LOD can enhance the capability of knowledge capture, knowledge exchange, knowledge discovery, and knowledge presentation. Thus, a KM system handling the Internet data can be built by utilizing the practical outcome of this study as a guideline for other domains and other KM processes.

1.8. Outline

This thesis organizes contents into the following chapters. It is important to inform in the beginning that Chapters 3-6 are key chapters that describes the main detail of each activity in KM process.

Chapter 2: Literature Review

This chapter gives the background knowledge used in this study. There are the overviews of knowledge management, LOD technology, biodiversity informatics, and the current issues about these topics. In order to give a proper reading flow, any pieces of related work of each project are written individually in each chapter.

Chapter 3: Knowledge Capture

Since knowledge capture and knowledge exchange are continuous stories and some definitions (such as a simple nominal entity and a contextual nominal entity) used by both activities are described in this chapter, Chapter 3 and 4 should be read respectively. This chapter explains the LTK project by the viewpoint of knowledge capture. An approach to the preservation of the change in taxonomy in form of the event model (an event-centric model) in RDF is written here.

Chapter 4: Knowledge Exchange

This chapter discusses the perspective of knowledge exchange of the LTK project, so LOD is more discussed here. The proposed Semantic Web rules transform an event model for capturing the change in taxonomy into chronological and temporal models (a transition model and a snapshot model) that are proper for LOD. In addition, a prototype of LTK project, which is an experiment of Chapters 3 and 4, are detailed in this chapter.

Chapter 5: Knowledge Discovery

This chapter describes the LPII project in terms of knowledge discovery. It aims to introduce a link prediction model for finding potential interspecies interaction. The analysis of the fungus-host interaction dataset from National Museum of Nature and Science, Japan is detailed here. The overview of collaborative filtering and community detection methods are also explained in order to lead the readers to understand our hybrid recommender model for interspecies interaction.

Chapter 6: Knowledge Presentation

This chapter gives a detail of the BViz project that provides a proper RDF graph visualization of biodiversity data that is easily to read by users. It explains about the problems of using a query RDF graph directly for visualization. A set of Semantic Web rules are introduced for sparsifying a complex graph. In addition, an approach to the rearrangement of a graph from common information to topic-specific information is proposed. In this project, we use the term “RDF4U” to be the name of application because it intends to use beyond the domain of biodiversity informatics.

Chapter 7: Discussion

The outcomes of the knowledge capture, knowledge exchange, knowledge discovery, and knowledge presentation are written in this chapter. However, due to the close relationship of knowledge capture and knowledge exchange, both activities are discussed in the same section. The overall outcome of this study including the relationship among the proposed projects is pictured here.

Chapter 8: Summary

The overall information about this study are concluded in the final chapter with the plan for the future of this thesis.

Appendix

All additional contents, configurations, and big chunks of supplementary content are written in the appendix in order to keep a well-organized, easy-to-navigate document, and the appendix is referred by some contents in some chapters.

.....

CHAPTER **2**

LITERATURE REVIEW

This thesis aims to study the role of linked open data in knowledge management process in biodiversity domain. The main contribution is to use a knowledge graph of biodiversity information as a knowledge base in knowledge management process. In order to have the precise understanding of the following chapters, necessary background knowledge has to be reviewed. This chapter provides principle contents about knowledge management, linked open data technology, and biodiversity informatics. The challenges of using linked open data in knowledge management activities for biodiversity domain are also indicated. It is noted that pieces of related work are not written here, because they are described in the following four chapters.

2.1. Knowledge Management

Saying "knowledge is a key to success" remains true throughout time. Many organizations, for example governments, academic institutes, industries, etc., have advanced themselves to be successful by placing important to the power of knowledge [11, 93]. Knowledge is considered to be a key resource for every community, every country, every industry, and every business in order to fine the advantage position in the completive environment [31]. One classic example of the most significant achievements of mankind is to send the man to walk on the moon in 1969. Many scientific knowledge and technologies were discovered for this mission. Some pieces of knowledge and technologies have been continuously developed for our lives such as portable devices and cellular phones [11]. Another example is about Japanese companies such as Honda, Canon, Matsushita, and Nissan. They have been successful in terms of innovation creation because they can use the organizational knowledge to manage their skills and expertise of employees effectively, make decision, and turn knowledge and innovation into a lot of successful products [93]. Moreover, there are so many other success stories of using knowledge that can be found in everyday life such as studying in classes, discussing with other people, reading from books, magazines, newspapers, Internet, etc. Most of stories have been driven by knowledge and a way to use the power of knowledge [11, 31, 93]. Thus, it can be say that Knowledge Management (KM) is a key player for enhancing the use of knowledge.

This section gives review about knowledge, KM activities, and KM processes. The review is mainly based on the following three books: "*The Knowledge Creating Company*" of Nonaka and Takeuchi (1995) [93], "*Knowledge Management Handbook*" of Liebowitz (1999) [75], and "*Knowledge Management Systems and Processes*" of Becerra-Fernandez and Sabherwal (2014) [11].

2.1.1. Knowledge

The definition of the term *knowledge* has a long history. Due to many perspectives of authors and fields of studies, there is no globally accepted definition of this term. In this thesis, we review the term knowledge briefly for giving a background before describing KM Process. As reviewing from [11, 75, 93] knowledge is related to information that is organized for problem and solving; the set of insight, experiences, and procedures that is considered as true; reasoning about information and data to solving problems, decision-making, learning, and teaching; personal map/model of the world; and a justified true belief. Although the definition of knowledge is somewhat essential, the more importance of this research is how to use it. Thus, we study more about the characteristics of knowledge.

Data, Information, and Knowledge

One much talked issue is about what different between data, information, and knowledge. Liebowitz [93] gave an interesting summarization as follows:

- **Data** are texts, facts, codes, images, or sounds.
- **Information** is organized structured, interpreted, or summarized data. In other words, it can be presented that *information = data + meaning + structure*.
- **Knowledge** is case, rule, process, or model. It is simply said that *knowledge = information + reasoning + abstraction + relationships + application*.

In this study, we present knowledge by an RDF model, so the graph of data is viewed as a knowledge graph.

Types of Knowledge

There are several viewpoints for classifying knowledge. Most types of knowledge in KM are tacit knowledge and explicit knowledge [11, 75, 93], so this study focuses on these two types.

- **Tacit knowledge** sometimes refers to know-how of human including insights, intuitions, and hunches. It is more likely to be personal because it is based on individual experiences. It is located in human's brain, so it cannot be completely expressed, formalized, and shared.
- **Explicit knowledge** generally referred to knowledge that has been expressed in texts, numbers, images, sounds, etc. and stored in proper medias such as stones, leathers, textures, woods, and papers, etc. In the computer age, explicit knowledge is recording in a digital storage, so it is easy to be identified, stored, and exchanged.

Besides knowledge, **factual evidence** is also mentioned in this thesis. It can be viewed as raw data that are observed from either laboratory or nature. For example, in biodiversity, the factual evidence can be the appearances or the behaviors of living things.

2.1.2. Knowledge Management Activities

The principle of KM has been discussed for about two decades. The growing of technologies and businesses contributes to the evolution of KM. Liebowitz [75] collected that KM can be an application or system that enhance a business by its knowledge assets; KM is a process that captures knowledge of organizational expertise into digital format and distributes it to other ones; KM gives the right knowledge to the right person at the right time and context; KM formalizes and accesses know-how of organizational staffs and then turns into innovation in order to create values for customers. In addition, Becerra-Fernandez [11] offered interesting definition of KM:

> *"Knowledge management can be defined as performing the activities involved in discovering, capturing, sharing, and applying knowledge so as to enhance, in cost-effective fashion, the impact of knowledge on the unit's goal achievement."*

Moreover, the use KM in any organization is much more focus on KM processes [11, 75]. In informatics, KM process includes activities between humans, pieces of knowledge, and computer systems. There is no standard for naming a KM activity. Each activity can be named differently depended on the viewpoint of authors. Some authors used different names for the same activity, whereas some use the same name but different scope as shown in Fig. 2.1. Regarding [11, 75, 111], KM activities are individually listed as follows:

- **Identifying** knowledge is to consider the goal of an organization, and find out that where know-how is or who has necessary tacit knowledge in order to achieve the goal.
- **Acquiring** knowledge is to find out necessary knowledge from both inside and outside an organization.
- **Organizing** knowledge is to enter, systemize, categorize, and codify both tacit and explicit knowledge in order to make knowledge be complete, standardized, and accessible.
- **Storing** knowledge is to preserve explicit knowledge into medias. At the moment, explicit knowledge can be digitalized and stored in the storage of a computer.
- **Accessing** knowledge is to access the right explicit knowledge at right time. A computer system can help human to search for some relevant pieces of knowledge from a huge storage.

- **Sharing** knowledge is to distribute explicit knowledge across KM systems. In this case, it needs communication between machines, so a common language and a common protocol are required.
- **Creating** knowledge is to find out or discover more knowledge. This activity can be done by both humans and computers. New tacit knowledge is sometimes created during discussion. Further, a computer can use some algorithms to discover new explicit knowledge.
- **Applying** knowledge is to learn, adopt, and adapt knowledge for creating an innovation or solving a problem. Thus, explicit knowledge should be presented in the right format in order to transform into tacit knowledge completely. Applications such as e-learning systems and decision making systems are also built to achieve this activity.

It is noted that the names and descriptions KM activities are summarized from several sources, so it does not need to stick with any names but places important to the ability of a KM system. Integrating KM activities into a KM process is very important for a KM system [11, 75]. However, the way to select and integrate should primarily satisfy the goal and the strategy of an organization. Next topic is the discussion about KM processes.

Source	Knowledge Management Activities						
Alavi and Leidner (2001)	Creation	Storage	Transfer	Application			
Alle (1997)	Collect	Identify	Create	Share	Apply	Organize	Adapt
Argote (1999)	Share	Generate	Evaluate	Combine			
Bartezzaghi et al. (1997)	Abstraction and Generalization	Embodiment	Dissemination	Application			
Davenport and Prusak (1998)	Determine Requirements	Capture	Distribute	Use			
Despres and Chauvel (1999)	Mapping	Acquire Capture Create	Package	Store	Apply Share Transfer	Reuse Innovate Evolve Transform	
Dixon (1992)	Acquire	Distribute	Interpret	Making Meaning	Organizati-onal Memory	Retrieve	
Huber (1991)	Acquisition	Distribution	Interpretation	Organizational Memory			
Nevis et al. (1995)	Acquisition	Sharing	Utilization				
Stein and Zwass (1995)	Acquisition Learning	Retention	Maintenance	Retrieval			
Szulanski (1996)	Initiation	Implementation	Ramp-up	Integration			
Walsh and Ungson (1991)	Acquisition	Storage	Retrieval				
Wiig (1997)	Creation	Capture	Transfer	Use			

Fig. 2.1: Determining the activities of KM Processes. Summarized by Sedera [111].

2.1.3. Knowledge Management Processes

As introduced in the previous part, there is no single solution for knowledge management event the scope of each KM activity. Many organizations demonstrate their KM strategy into KM process, and it is often represented by a flowchart diagram of KM activities. KM process transforms knowledge into a valuable organizational asset, by formalizing, distributing, sharing, and applying knowledge, experience, and expertise [75]. In KM process, most steps for managing knowledge are KM activities that are sometime repeated and do not to be arranged

in linear sequence [11, 75]. However, there is also no single solution for any KM process. It is depended on the vision, strategy, requirement, technology, budget, and culture of an individual organization. There are many proposed models as shown in Fig. 2.1. In this part, some well-discussed processes are reviewed.

SECI Model (1995)

SECI (Socialization, Externalization, Combination, and Internalization) is introduced by Nonaka and Takeuchi in 1995 [93]. As introduced in the beginning of this section, Nonaka and Takeuchi studied the processes of Japanese companies and summarized the principle of the management of their knowledge into the SECI model as shown in Fig. 2.2. SECI model was the earlier KM process that aims to be a model of the knowledge creating process. It is working by understanding the dynamic nature of knowledge creation, which are the transformation between tacit and explicit knowledge, and to manage such a process well. There are four activities which are arranged in a spiral flow, and their scale become larger in the next round of knowledge creation [93]. The model uses a human as a primary entity, so internalization means moving knowledge into the human and externalization mean moving knowledge out of the human.

- **Socialization** is an activity for sharing tacit knowledge through face-to-face communication or shared experience such as meeting and training. New tacit knowledge can be created here.
- **Externalization** is an activity for developing model that can express tacit knowledge into explicit knowledge.
- **Combination** is an activity for combining various pieces of explicit knowledge for any other sources. New knowledge can be created here.
- **Internalization** is an activity for transforming explicit knowledge into tacit knowledge. It is the same way as presenting knowledge to learners.

Since this model comes up from what industries do, the model becomes up-to-date and practicable until now, and it also becomes a principle model for any other KM systems.

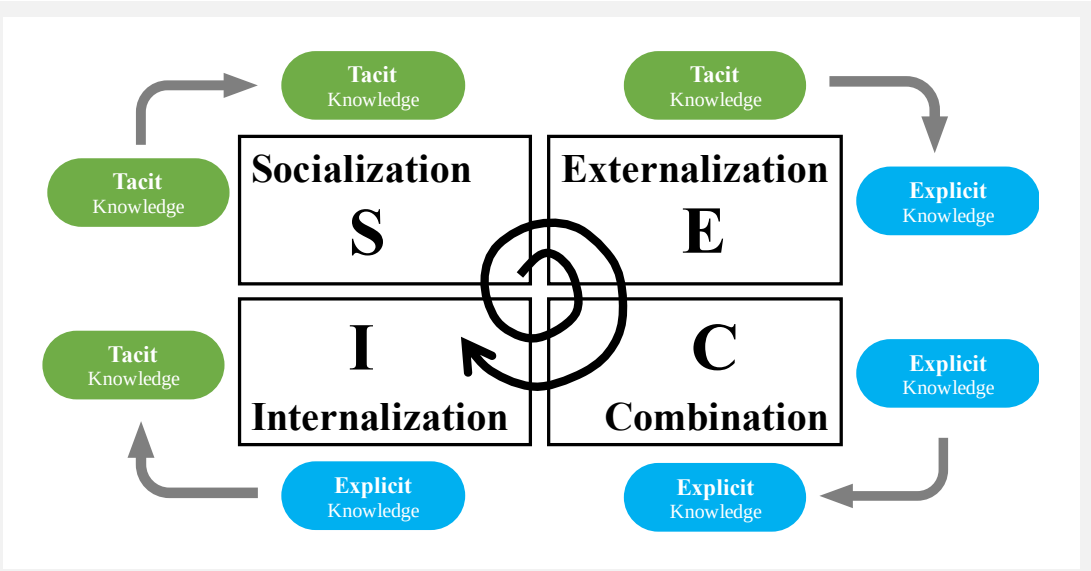


Fig. 2.2: SECI Model

Beckman's Model (1997)

Another KM process, which is always discussed, is Beckman's model [12]. This is a circle process including eight main stages or activities as demonstrated in Fig. 2.3.

- **Identify** or determine core capabilities, strategies, and knowledge domains.
- **Capture** or formalize existing knowledge.
- **Select** or measure knowledge relevance and value, and resolve some conflicting information.
- **Store** or preserve the presentation of knowledge in memory.
- **Share** or distribute knowledge with collaborates.
- **Apply** or use knowledge for solving problems, making decisions, supporting education, or training.
- **Create** or discover new knowledge through research, experiment, and creative thinking.
- **Sell** or develop new knowledge-based products and services, and market them.

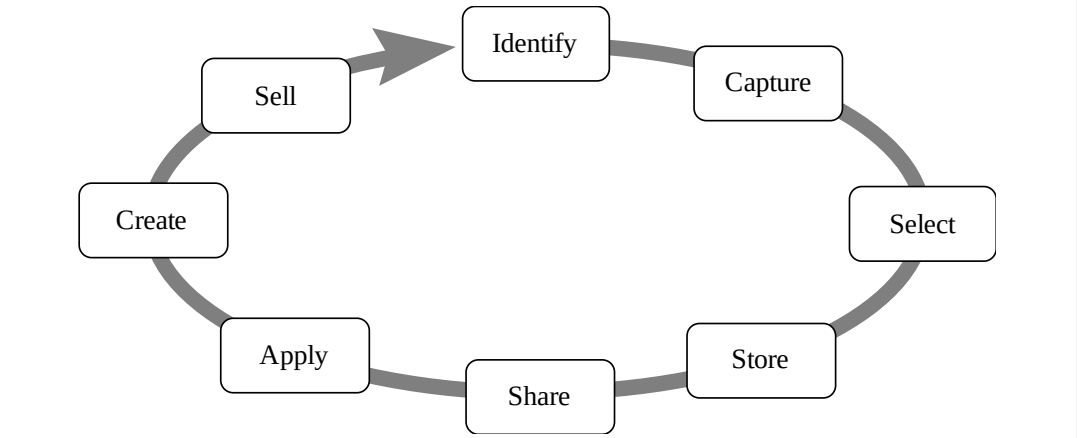


Fig. 2.3: Beckman's Model

Holsapple's and Joshi's Model (2002)

Holsapple and Joshi [57] introduced a framework including six KM activities that are designed on top of what business needs. Fig. 2.4 shows the framework including a KM process (above part) and business (below part). The following list describes only the KM process. Moreover, the framework can interact with external sources and targets such as customers, competitors, suppliers, universities, consultants, government agencies in order to acquire and exchange knowledge.

- **Acquiring** knowledge is to extract, interpret, and transfer knowledge from external sources.
- **Selecting** knowledge is to locate, retrieve, and transfer knowledge from an organization.
- **Internalizing** knowledge is to assess, target, and deposit knowledge into an organization.
- **Using** knowledge includes two activities: Generating knowledge and Externalizing knowledge.
- **Generating** knowledge is to monitor, evaluate, and produce transferring knowledge to the activity externalizing knowledge.

- **Externalizing knowledge** is to target, produce, and transfer output knowledge to external targets.

This model uses similar terms as SECI [93] model, however some descriptions are different because the external and internal elements of this model refers to own organization and other organizations respectively, whereas internal and external elements of SECI model refers to humans and machines respectively. Thus, the "Selecting" of this model is similar to the "Externalization" of SECI model, the "Internalizing" of this model is similar to the "Internalization" of SECI model, and the "Acquiring" and "Externalizing" are similar to "Combination" of SECI model.

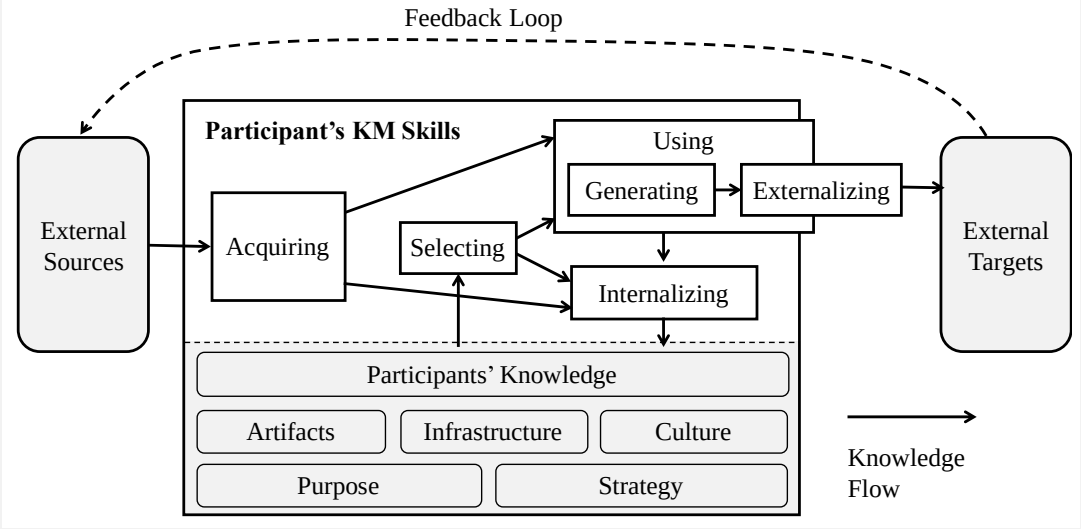


Fig. 2.4: Holsapple's and Joshi's Model

Becerra-Fernandez's and Sabherwal's Model (2014)

Becerra-Fernandez and Sabherwal [11] introduced a model for KM processes and KM systems. The model is a workflow including four main KM activates that is demonstrated in Fig. 2.5. Each activity involved by both humans and computers, so its sub-activates can be either human's or machine's tasks, and the outcome of both clients becomes input of applications.

- **Discovery** is an activity for creating new knowledge. It includes combination and socialization that are similar to the SECI [93] model. The combination is to combine knowledge from many sources for creating new explicit knowledge, while the socialization is to create tacit knowledge by face-to-face communication.
- **Capture** is broader than the term "Capture" from Beckman's [12] model, which is just only capture knowledge from humans. The capture in this model aims to be moving knowledge between humans and computers, so it comprises of the internalization and the externalization that have same meanings as SECI [93] model.
- **Sharing** is an activity that establish communication between tacit and explicit knowledge. The socialization is to share knowledge between humans, whereas the exchange is to share knowledge between computers.
- **Application** is the final result of this process. It includes direction and routines. The direction is a task of top management for solving problem in organization such as

expert systems and business intelligence systems, whereas the routines are everyday task of small segments.

The overall picture expresses that the discovery and the capture produce and handle both tacit and explicit knowledge. After that, the sharing exchanges and communicates knowledge among multiple sources. Then, both tacit and explicit knowledge becomes a raw material for applications to compute, decide, recommend, and take some appropriate actions.

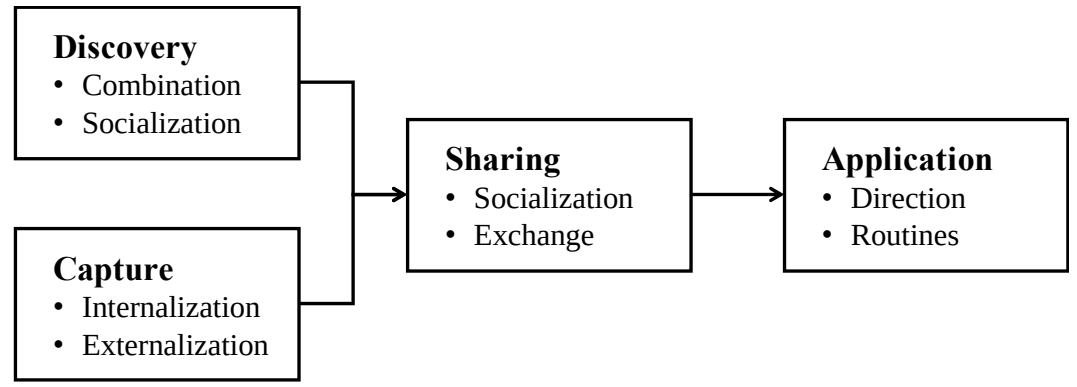


Fig. 2.5: Becerra-Fernandez's and Sabherwal's Model

Using knowledge management for an organization or a domain is interesting and challenging because it is depended on a context and there is no the best solution at all. It is a science that uses a creative way to align technology with business. Defining, choosing, or introducing KM activities and scheming KM process must be considered by the capability of businesses and technologies including organizational visions, business requirements, staffs, cultures, working processes, budgets, tools, hardware, software, infrastructure, and environment. Thus, KM processes are not developed similarly among different organizations and/or different time.

As we summarized from these studies, we found that there are four important KM activities in most KM systems.

- Preserving knowledge into digital objects.
- Exchanging knowledge across systems.
- Adding value to knowledge.
- Learning knowledge from digital objects.

Actors of these activities can be either humans or computers, and some of these activities can be selected, named, enhanced, decorated, merged, split, re-arranged, etc. according to the conditions and requirements of an individual organization and a domain.

2.2. Linked Open Data

Linked Open Data (LOD) is one practice approach of Sematic Web. One significant ability is to integrate open data from different schemas through the Internet. A graph is a data structure that is used in LOD. The reasoning of graph data together with schemas and ontologies can improve the connection and accessibility of data. Thus, we would like to provide background knowledge of RDF model, ontology, reasoning, query, and LOD cloud.

This review on the basis of two well-known books: "*Foundation of Semantic Web Technolgies*" of Hitzler, Krötzsch & Rudolph (2009) [54], and "*Linked Data: Evolving the Web into a Global Data Space*" of Heath & Bizer (2011) [50].

2.2.1. Resource Description Framework (RDF)

Graph data model in LOD is presented by a direct graph. Nodes can be either named resources or literals, while links are named properties. Every resource and property is written using a Uniform Resource Identifier (URI), and the structure of graph is modeled by Resource Description Framework (RDF) [108].

Uniform Resource Identifier (URI)

Every resource and property must be identified. In the RDF model, the URI is used to be the identifier of resources and properties. For example,

```
http://www.example.org/universities/Sokendai
http://www.example.org/students#Rathachai
http://www.example.org/terms#studiesAt
```

are resources describing the terms "*Sokendai*", "*Rathachai*", and "*studies at*" respectively. URIs are used to identified real-world objects and abstract concepts. LOD recommended that the URI should be HTTP URI, and clients can access each URI using the HTTP protocol and then get a returned document according to requested format.

RDF allows having short-hand writing, for example if a prefix is defined as

```
@prefix unv: <http://www.example.org/universities/> .
@prefix std: <http://www.example.org/students#> .
@prefix :    <http://www.example.org/terms#> .
```

The former URIs can be shortened to be `unv:Sokendai`, `std:Rathachai`, and `:studiesAt`.

In addition, it is possible to use a blank node if no URI is decided, such as `_:a1`, but we do not recommend because it is difficult to give a reference in the practical uses.

It is noted that URIs in this thesis are commonly written in shortened form, and example URIs in this review are not dereferencable.

Literal

Literals are texts that are typed and untyped. The untyped literal can be any string such as *"Rathachai Chawuthai", "December 1983", "18.12",* and *"+8180-7999-1818"*. A string can be ended with a language tag in order to inform the language of a text. In addition, the typed literal contain a property value together with a URI of datatype. For example

- `"Linked Open Data"@en` presents that this string is in English.
- `"555"^^xsd:integer` presents an integer of 555.
- `"1983-12-18"^^xsd:date` presents a date of December 18th, 1983.

Triple

Triple is a binary relation between two entities, and it is the fundamental unit of an RDF graph. It includes the sequence of a subject, a predicate, and an object. Subjects and predicates are resources represented by URIs, but objects can be either resources or literals.

The formal representation of triple is $\langle s,p,o \rangle$ where *s* is a subject, *p* is a predicate or a property, and *o* is an object. In the graph diagram, the URI of a resource is commonly written inside an eclipse, a literal is presented by a quoted text inside a rectangle, and an arrows with the URI of predicate are drawn from a subject to an object.

Fig. 2.6(a) is interpret that the resource *std:Rathachai* is a subject, the *:studiesAt* is a predicate, and the *unv:Sokendai* is an object; and humans can simply interpret that *Rathachai studies at Sokendai*.

For Fig. 2.6(b), the resource *std:Rathachai* is a subject, the *foaf:name* is a predicate, and the "*Rathachai Chawuthai*" is an object literal, and humans can simply interpret that *Rathachai's name is "Rathachai Chawuthai"*.

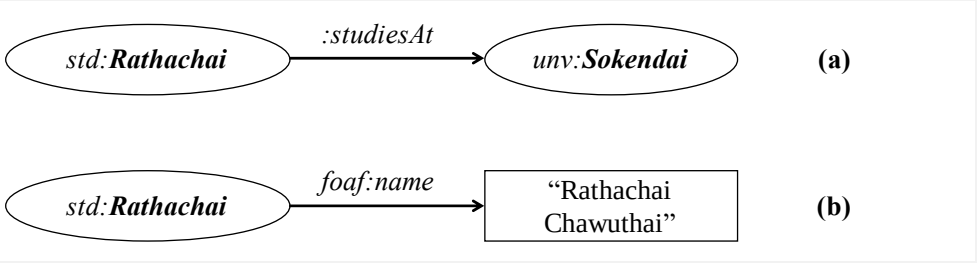


Fig. 2.6: Example Triples
A triple (a) is interpreted that *Rathachai studies at Sokendai*.
A triple (b) is interpreted that *Rathchai's name is "Rathachai Chawuthai"*.

A triple can be expressed by several formats such as Turtle, XML, JSON-LD, and N-Triples, etc. In this thesis, we commonly use the Turtle (or RDF/Turtle) because it is convenient to read and write by humans. The diagram in Fig. 2.6(a) can be expressed in Turtle format by

```
<http://www.example.org/students#Rathachai>
  <http://www.example.org/terms/studiesAt>
    <http://www.example.org/universities/Sokendai> .
```

or it can be shortened into

```
@prefix unv: <http://www.example.org/universities/> .
@prefix std: <http://www.example.org/students#> .
@prefix :    <http://www.example.org/terms/> .

std:Rathachai  :studiesAt  unv:Sokendai .
```

For Fig. 2.6(b) or it can be expressed by

```
@prefix unv:  <http://www.example.org/universities/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

std:Rathachai  foaf:name "Rathachai Chawuthai" .
```

RDF Graph

RDF graph that can be called as RDF model is a set of triples. It can be written in Turtle, so triples can be stated in curly brackets such as { *:s1 :p1: o1* . *:s2 :p2 :o2* . }, and the expression can be abbreviated as the following list. It is noted that the declaration of prefixes is sometime avoid in some expressions in order to make it be simpler and easier for reading.

- If triples share the same subject and predicate, it can put objects in an object list and separate them by comma (,). For example, a graph { *:s1 :p1 :o1* . *:s1 :p1: o2* . } can be abbreviated as { *:s1 :p1 :o1, o2* . }.
- If triples share the same subject, it can put predicates and objects in a predicate list and separate them by semi-colon (;).For example, a graph { *:s1 :p1 :o1* . *:s1 :p2 :o2* . } can be abbreviated as { *:s1 :p1 :o1; p2 o2* . }.
- If triples contain the chain of same blank node, it can put the relative cause in square brackets. For example, a graph { *ex:Ryu foaf:knows _:a1* . *_:a1 foaf:name "Ken"* . } can be abbreviated as { *ex:Ryu foaf:knows* [*foaf:name "Ken"*] . }, which can be read as *Ryu knows someone whose name is "Ken"*.

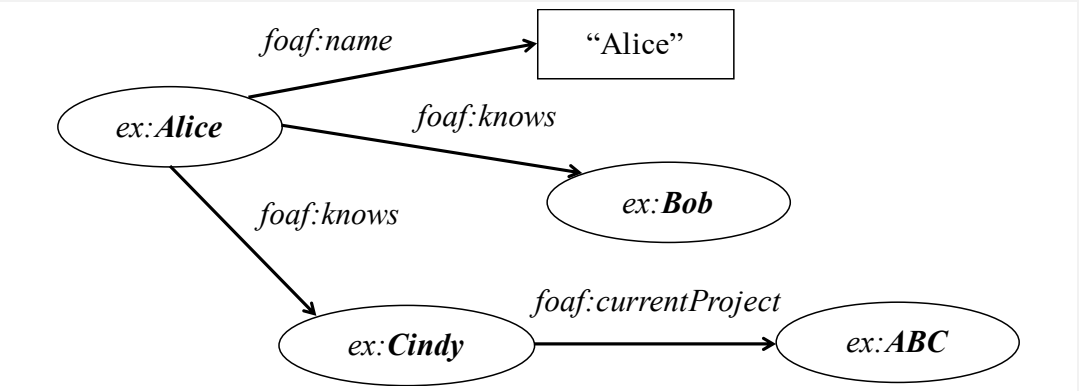


Fig. 2.7: Example RDF Graph
This graph is interpreted that *Alice whose name is "Alice" knows Bob and Cindy whose current project is ABC*.

Fig. 2.7 is an example of RDF graph. It can be expressed in the simple Turtle format as follows:

```
ex:Alice   foaf:name             "Alice" .
ex:Alice   foaf:knows            ex:Bob .
ex:Alice   foaf:knows            ex:Cindy .
ex:Cindy   foaf:currentProject   ex:ABC .
```

In this case, it can be abbreviated as the following expression.

```
ex:Alice   foaf:name             "Alice" ;
           foaf:knows            ex:Bob , ex: Cindy .
ex:Cindy   foaf:currentProject   ex:ABC .
```

2.2.2. The Interpretation on Ontology

Without interpretation, an RDF statement is just a string of characters. Adding meaning to an RDF graph is a key feature of Semantic Web, and this feature makes an RDF graph become

excellently machine-readable. Ontology is a formal, explicit specification of a shared conceptualization. Formal refers to machine-readable; explicit specification refers to concepts, properties, relations, constants, taxonomies, and axioms; and shared conceptualization refers to consensual knowledge of abstract model and real-world objects. An RDF graph is also one part of ontology. With the higher interpretation of RDF data with either RDF Schema or ontologies, the graph is more expressive.

RDF Schema (RDFS)

RDFS provide basic entailments for interpreting ontologies [17]. It mainly includes the interpretation of classes, properties, the hierarchies of classes, and the hierarchies of properties. There are 13 RDFS deduction rules for RDFS-Entailment, but we describe only rules that use in this thesis.

It is noted that the statement ⟨ *x, rdf:type, C* ⟩ where *x* and *C* are any URIs and they can be interpreted that the instance *x* is a member of a class or a set *C*. In this section, the name of a class is recommended to begin with an uppercase letter, while the name of either an instance or a property begins with a lowercase letter.

Domain and Range

rdfs2: if ⟨ *p, rdfs:domain, C* ⟩ and ⟨ *s, p, o* ⟩ ,
then ⟨ *s, rdf:type, C* ⟩ .

rdfs3: if ⟨ *p, rdfs:range, C* ⟩ and ⟨ *s, p, o* ⟩ ,
then ⟨ *o, rdf:type, C* ⟩ .

Subproperties

rdfs5: if ⟨ *p2, rdfs:subPropertyOf, p1* ⟩ and ⟨ *p1, rdfs:subPropertyOf, p0* ⟩ ,
then ⟨ *p2, rdfs:subPropertyOf, p0* ⟩ .

rdfs7: if ⟨ *p1, rdfs:subPropertyOf, p0* ⟩ and ⟨ *s, p1, o* ⟩ ,
then ⟨ *s, p0, o* ⟩ .

Subclasses

rdfs9: if ⟨ *C1, rdfs:subClassOf, C0* ⟩ and ⟨ *x, rdf:type, C1* ⟩ ,
then ⟨ *x, rdf:type, C0* ⟩ .

rdfs11: if ⟨ *C2, rdfs:subClassOf, C1* ⟩ and ⟨ *C1, rdfs:subClassof, C0* ⟩ ,
then ⟨ *C2, rdfs:subClassOf, C0* ⟩ .

For example, if the following RDF graph exist

```
foaf:Person      rdfs:subClassOf       foaf:Agent .
foaf:homepage    rdfs:subPropertyOf    foaf:page ;
                 rdfs:range            foaf:Document .
ex:dan           rdf:type         foaf:Person ;
                 foaf:homepage    <http://dan.info> .
```

After entailment with RDFS rules, the inferred triples are generated as follows:

- rdfs3 entails { *<http://dan.info> rdf:type foaf:Document* . } .
- rdfs7 entails { *ex:dan foaf:page <http://dan.info>* . } .
- rdfs9 entails { *ex:dan rdf:type foaf:Agent* . } .

Web Ontology Language (OWL)

OWL is a recommended standard for the modelling of ontologies [29]. It gives higher interpretation over RDF and RDFS, so the representation of a knowledge graph becomes more expressive. There are many features and details in OWL, however, this review selects some features that are commonly used in this thesis, for example equality and property characteristics.

It is firstly informed that every instance is belong to *owl:Thing*.

Equality

same-as: if ⟨ *s1, owl:sameAs, s2* ⟩ and ⟨ *s1, p1, o1* ⟩ ,
then ⟨ *s2, p1, o1* ⟩ .

if ⟨ *o1, owl:sameAs, o2* ⟩ and ⟨ *s1, p1, o1* ⟩ ,
then ⟨ *s1, p1, o2* ⟩ .

equivalence class:
if ⟨ *C1, owl:equivalentClass, C2* ⟩ and ⟨ *x, rdf:type, C1* ⟩ ,
then ⟨ *x, rdf:type, C2* ⟩ .

equivalence property:
if ⟨ *p1, owl:equivalentProperty, p2* ⟩ and ⟨ *s1, p1, o1* ⟩ ,
then ⟨ *s1, p2, o1* ⟩ .

Property Characteristics

transitive property:
if ⟨ *p, rdf:type, owl:TransitiveProperty* ⟩ , ⟨ *x, p, y* ⟩ , and ⟨ *y, p, z* ⟩ ,
then ⟨ *x, p, z* ⟩ .

symmetric property:
if ⟨ *p, rdf:type, owl:SymmetricProperty* ⟩ and ⟨ *x, p, y* ⟩
then ⟨ *y, p, x* ⟩ .

inverse property:
if ⟨ *p1, owl:inverseOf, p2* ⟩ , and ⟨ *x, p1, y* ⟩
then ⟨ *y, p2, x* ⟩ .

For example, if the following RDF graph exists

```
skos:broaderTransitive  rdf:type  owl:TransitiveProperty .
skos:broader  owl:reverseOf  skos:narrower .

ex:football   owl:sameAs   ex:soccer .
ex:football   skos:broader ex:sport .

ex:dog        skos:broaderTransitive      ex:mammal .
ex:mamal      skos:broaderTransitive      ex:animal .
```

After interpreting with OWL, the inferred RDF graph is generated as follows:

```
ex:soccer     skos:broader  ex:sport .
ex:sport      skos:narrower ex:football .
ex:sport      skos:narrower ex:soccer .

ex:dog        skos:broaderTransitive  ex:animal .
```

Well-Known Ontologies

There are many ontologies that are used for specific purposes and domains. In this study we review some ontologies that are commonly used in this thesis.

DC-Term

DC-Term (DCMI metadata term) [137], which is the extension of Dublin Core Metadata, contains metadata for describing a documents and their relationships. For example

- *dct:isVersionOf* is a property that identify the previous version of a term.
- *dct:source* is a property that identify the reference.

BIBO

BIBO (Bibliographic ontology) [133] provides main concepts and properties for describing citations and bibliographic references. For example

- *bibo:performer* is a property that identify who did a given document.
- *bibo:issuer* is a property that identify who issued or released a given document.

FOAF

FOAF (Friend of a Friend) ontology [139] provides vocabularies for describing humans, organizations, and documents, and relationships among them. For example

- *foaf:Person* is a class of persons, and it is the sub class of *foaf:Agent*.
- *foaf:knows* is a symmetric property that identifies persons knowing each other.
- *foaf:depiction* is a property that points to a picture that is the instance of the class *foaf:Image*.

SKOS

SKOS (Simple Knowledge Organization System) ontology [150] provides a model for expressing the basic structure and content of concept schemes. For example

- *skos:broader* is a property that identifies the board concept of a given concept.
- *skos:broaderTransitive* is a transitive property that identifies the transitively board concept of a given concept.
- *skos:narrower* is a property that identifies the narrow concept of a given concept and it is the inverse property of *skos:broader*.
- *skos:narrowerTransitive* is a property that identifies the transitively narrow concept of a given concept and it is the inverse property of *skos:broaderTransitive*.

DBpedia

The previous ontologies generally provide vocabularies and schemas. In other words, they are acting as a meta-ontology. However, DBpedia [72] that is an ontology extracted from a large number of Wikipedia entries, provides both schemas and data, so DBpedia becomes famous in terms of being a big RDF dataset.

2.2.3. Semantic Web Reasoning

RDFS and OWL provide necessary rules to entail an RDF graph. In case some specific requirements beyond RDFS and OWL are needed, developers can create some Semantic Web

rules. For creating own rules, the book of Semantic Web programming [51] recommended to use Apache Jena [143] as a reasoning engine that is a library of Java.

For example, in case the condition “An uncle is a brother of one’s father” is defined, it can be simply expressed that:

If ⟨ *x, :hasFather, y* ⟩ and ⟨ *y, :hasBrother, z* ⟩ , then ⟨ *x, :hasUncle, z* ⟩ .

This rule can be written in a dialog program as follows:

```
hasFather(?x, ?y) ∧ hasBrother(?y, ?z) → hasUncle(?x, ?z)
```

Moreover, it can be written for executing by Jena in the following expression.

```
[rule_identify_uncle:
  (?x  :hasFather   ?y),
  (?y  :hasBrother  ?z)
->(?x  :hasUncle    ?z)]
```

For demonstrate the result of Semantic Web rules, if the following RDF graph exists

```
ex:john  :hasFather  ex:smith .
ex:smith :hasBrother ex:adam .
```

the inferred model is generated after executing the RDF graph with the according rule.

```
ex:john  :hasUncle ex:adam .
```

2.2.4. SPARQL

SPARQL (SPARQL Protocol and RDF Query Language) is a protocol and query language for querying RDF data [152]. The syntax includes prefixes, scheme or result, and condition.

For example, there is the following RDF graph.

```
ex:Movie      rdfs:subClassOf      ex:Entertainment .
ex:TVSeries   rdfs:subClassOf      ex:Entertainment .

ex:starwarsVII       rdf:type      ex:Movie ;
                     ex:starring   ex:daisy, ex:adam .

ex:gameOfThrones     rdf:type      ex:TVSeries .
```

If all documents are needed to be list, the SPARQL expression can be written as follows:

```
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ex:   < http://www.example.org/ex-terms/>

SELECT ?ent  WHERE  { ?ent  rdf:type  ex:Entertainment .}
```

The following result is retrieved.

```
+-----------------------------------------------+
|                      ?ent                     |
+-----------------------------------------------+
| http://www.example.org/ex-terms/starwarsVII   |
| http://www.example.org/ex-terms/gameOfThrones |
+-----------------------------------------------+
```

In addition, the query can result in a graph if *CONSTRUCT* is used instead of *SELECT*. For example, if actors or actresses played in the same story, they should know each other. Thus, the query expression is written as follows:

```
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ex:   < http://www.example.org/ex-terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

CONSTRUCT { ?a    foaf:knows     ?b .}
WHERE     { ?ent  ex:starring    ?a , ?b .}
```

Then, the following graph is retrieved.

```
<http://www.example.org/ex-terms/daisy>
  <http://xmlns.com/foaf/0.1/knows>
    <http://www.example.org/ex-terms/adam> .
```

2.2.5. Linked Open Data Cloud

Linked Open Data (LOD) is the right progress of Sematic Web. It aims to have structured data around the world be linked through the Internet using Semantic Web rules and queries. However, this story is so far [16]. In order to have high quality LOD, Tim Berners-Lee introduced the five stars rating scheme for LOD.

★	Data are available online with public license such as PDF.
★★	Data are structured and some software can read such as Excel.
★★★	Data are non-proprietary format such as CSV.
★★★★	Data are in RDF format.
★★★★★	Data can link to or reuse identifiers from other datasets especially in well-known datasets.

In order to have five-star data, Tim Berners-Lee also introduced the principle of linked data as follows:

- Uses URIs to identifying things
- Use HTTP URIs, so these things can be referred to and looked up (“dereferenced”) by people and user agents.
- Provide useful information about the thing when its URI is dereferenced, leveraging standards such as RDF, RDFS, OWL, and SPARQL.
- Include links to other URIs, so that they can discover more things.

Based on this principle, LOD Cloud is materialized from a single triple to an RDF dataset and then subsequently to the associations among numerous graph datasets as shown in Fig. 2.8. The growth of LOD Cloud is impressive from a small number of datasets in 2007 until more

than 500 datasets by the 2014 [145]. The journey of LOD is a piece of evidence to prove the vision of Berners-Lee and the power of Semantic Web.

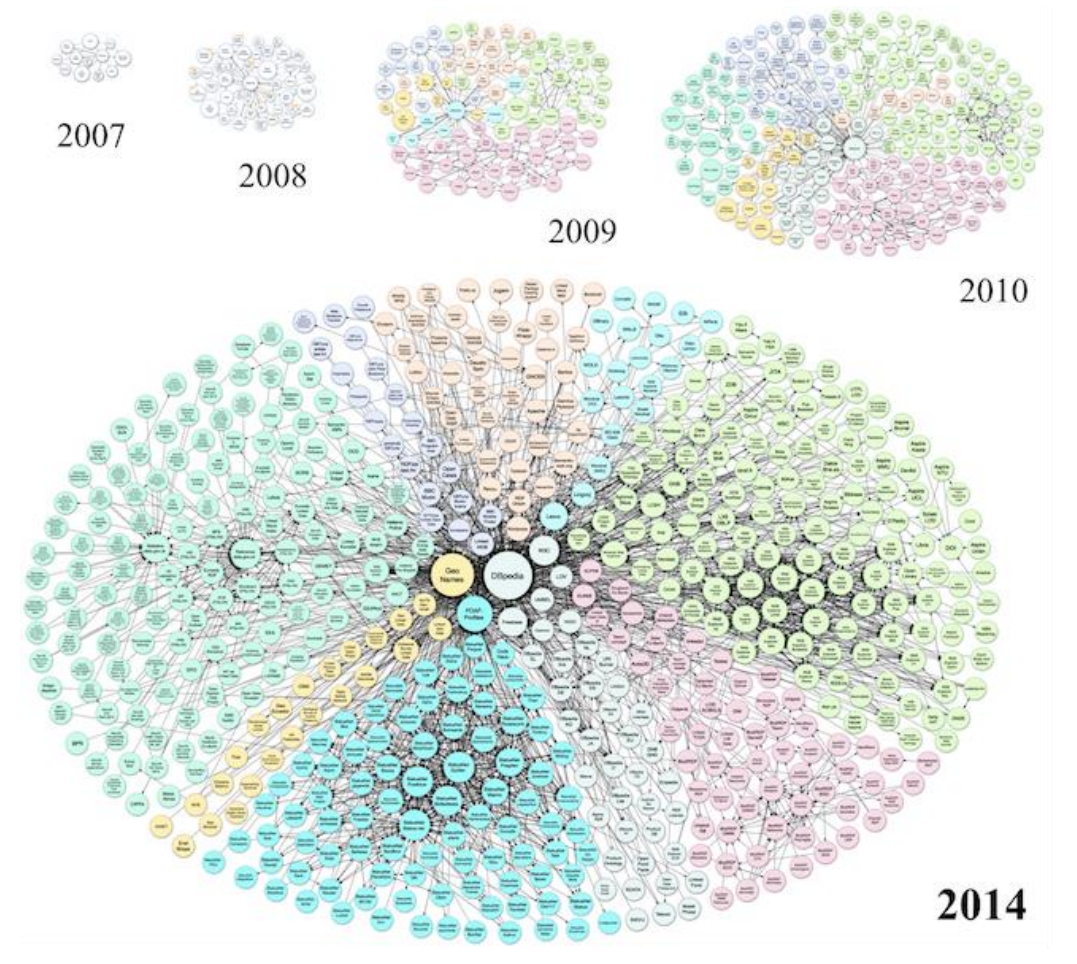


Fig. 2.8: Linked Open Data Cloud Diagrams
The LOD Cloud diagrams from the first version on 2007-05-01 (12 datasets) to the last updated version on 2014-08-30 (570 datasets).
(Source: *http://lod-cloud.net/*)

2.3. Biodiversity Informatics

Biodiversity is the shortened-form of the term “Biological Diversity”. Biodiversity means the variety of organisms in all manifestations. The main elements of biodiversity are genetic diversity, ecological diversity, and organismal diversity; and the study of biodiversity includes genes, species, assemblages, taxonomy, ecological process, ecological components, ecosystems, and their interactions [45].

Biodiversity informatics is a field that relates to biodiversity information and information technologies. This field comprises of information management applications, algorithms, analysis, and the interpretation of data regarding organisms, especially in the knowledge organization of the species level [115].

Since our work relates to a knowledge graph, we have to give the background of taxonomy and interspecies interaction in order to be a ground for creating a knowledge graph.

2.3.1. Taxonomy

Taxonomy is the science of defining organismal groups. It includes the naming of organismal groups, the hierarchy of groups on the basic of shared characteristics of members in a group, and association between names. This review is based on the books titled "Describing Species: Practical Taxonomic Procedures for Biologists" of Winston [122], and "Naming Nature: The Clash between Instinct and Science" of Yoon [126].

Nomenclature (System of Naming)

Nomenclature is a rule for naming organisms or living things and maintaining a name system, for example, the names of animals, plants, fungi, etc. The "name" in this case, called "scientific name", is used as a label of an organismal group. At the moment, biologists agree to use scientific names for describing living things. It is noted that the following examples are for animals, but names for plants and fungi are under control by another nomenclature which has different name structures and terms.

The organismal group is defined in hierarchy called biological classification that is described in the next part. The name at any rank above species uses only a single Latin letters, such as Animalia, Arthoropoda, Insecta, Lepidoptera, Saturniidae, *Saturnia*, etc.

The name for species is commonly written by the binomial nomenclature proposed by Linnaeus in 18th century. The name of each species is composed of two parts. The first part is the genus, that is the name of an immediate group of species. The second part, called specific name in zoology, is the identifier of a species in that genus. Each part is in Latin letters and the meaning relates to the specific description of that part. For example, the scientific name of the Japanese giant silkworms is

Saturnia japonica

where "*Saturnia*" is a genus name and "*japonica*" is an identifier under this genus. The species name can be shortened into *S. japonica* for the second or more time of writing.

In addition, subspecies, which is a subgroup of a species, can be named using three parts. The first and second parts are elements of a species name, and the third part, called a subspecific name, is the identifier under that species. For example, the scientific name of one subspecies under *S. japonica* is

Saturnia japonica ryukyuensis

where "*Saturnia japonica*" is a species name and "*ryukyuensis*" is a specific name that is an identifier under this species. The species name can be shortened into *S. japonica ryukyuensis* for the second or more time of writing.

For names of animals, scientific names can include an author's name and year, for example

Saturnia japonica ryukyuensis Inoue, 1984

means that this name has been introduced by Inoue in 1984.

Biological Classification

Taxonomists classify organismal groups into hierarchy based on the shared characteristics of members or groups. In this field, the term "taxon" (plural taxa) is used instead of the term "group". For example, Animalia, Arthoropoda, Insecta, Lepidoptera, Saturniidae, *Saturnia*, *Saturnia japonica*, and *Saturnia japonica ryukyuensis* are names of taxa.

The classification is done in hierarchy, and each level in the hierarchy is called “Rank”. The traditional ranks are kingdom, phylum, class, order, family, genus, and species that are ordered from the highest level to the lowest level [132]. The example of the classification is shown in Fig. 2.9. It is possible to give some more general or more specific ranks for each traditional rank, for example superspecies and subspecies are more general and specific than the rank of species respectively.

The biological classification has a long story. In the beginning, it was done by estimating the appearances of specimens, such as organs, sizes, colors, behaviors, etc. At the moment, genetic analysis is trended to use to estimate distance between specimens. The classification shows that living things under the same lower rank are very closely related.

Ranks	Example Taxon	Description
Kingdom	Animalia	Organisms able to move on their own.
Phylum	Chordata	Animals with a back bone.
Class	Mammalia	Chordates with fur or hair and milk glands.
Order	Primates	Mammals with collar bones and grasping fingers
Family	Haplorhini	Primates with relatively flat faces and three-dimensional vision.
Genus	*Homo*	Hominids with upright posture and large brains.
Species	*Homo sapiens*	Member of the genus Homo with a high forehead and thin skull bones

Fig. 2.9: Biological Classification
Example hierarchical classification of human

Association between Names

Since biologists worked in different places and different time and they might have different perspectives, the separated definitions of names and classifications are commonly found.

In the Zoological Nomenclature, the synonym is defined as “each of two or more (scientific) names of the same rank used to denote the same taxonomic taxon.” Another definition is that the two names must have been established separately when they are regarded as synonyms. *Saturnia japonica* and *Caligula japonica* are different names which denoted same taxon, but the origin of two names are same. Some other reasons of synonymy are as follows:

- Two names were published by different biologists, then they become synonym.
- When a rank of a taxon is changed, the suffix of its name is sometimes changed.
- When taxa are merged into a new taxon, the names before and after this change are synonym.

On the other hand, the definition of synonym in botany is different from that in zoology. In Botany, such cases as *Saturnia japonica* and *Caligula japonica* are also included in the range

of synonym and called homotypic synonym, while the case of *Caligula japonica* and *Dictyoploca manonis* is called heterotypic synonym.

Moreover, it is possible to see the same scientific name that points to different taxa for example *Echidna* is the name of the genus of moray eels and African snakes. In zoology, the earlier homonym becomes valid. Thus, *Echidna* J. R. Forster, 1788 is a genus of moray eels while *Echidna* Merrem, 1820 is junior homonym for a genus of African snakes.

Table 2.1: Types of Interspecies Interactions
Note: + indicates population growth increased; - indicates population growth decreased; and 0 indicates population growth not affected.
(Source: Odum, 1959 [95])

Type of Interaction	Effect of Relationship on Growth and Survival of Two Populations				Genera Result of Interaction
	When No Interacting		When Interacting		
	A	***B***	***A***	***B***	
Neutralism *(A and B independent)*	*0*	*0*	*0*	*0*	Neither population affects the other.
Competition *(A and B competitors)*	*0*	*0*	+	-	Population most effected eliminated from niche.
Mutualism *(A and B partners, or symbionts)*	-	-	+	+	Interaction obligatory to both.
Protocooperation *(A and B cooperators)*	*0*	*0*	+	+	Interaction favorable but not obligatory to both.
Commensalism *(A, commensal; B, host)*	-	*0*	+	*0*	Obligatory for *A*; *B* not affected.
Amensalism *(A, amensal; B, inhibitor in allelopathy, or antibiotic in antibiosis)*	*0*	*0*	-	*0*	*A* inhibited; *B* not affected.
Parasitism *(A, parasite; B, host)*	-	*0*	+	-	Obligatory for *A*; *B* inhibited
Predation *(A, predator; B, prey)*	-	*0*	+	-	Obligatory for *A*; *B* inhibited.

2.3.2. Interspecies Interaction

The field of interspecies interactions is an area of ecology. The area studies the responses of species to their environments [95] or the interaction between species. The latter may result in several outcomes depended on which species involved in the interaction. The interaction between the two species sometimes provides benefits to both, but the interaction may sometimes result in negative consequence. Eight types of interactions have been known as shown in Table 2.1.

In this study, we focused on the parasitism interaction between fungi and hosts because it is an important area in agriculture, and not only the knowledge, but together with the benefit of this kind of knowledge is also worthy to be adopted in the knowledge management as well.

Fungi (singular: fungus) are diverse group of organisms that can be found in land, water, and air. Since they are not animals or plants, they have been classified in the kingdom of fungi. This kingdom includes mushrooms, molds, and yeasts. Since they cannot produce nutrients by

themselves, they obtain nutrients from other organism, namely animals, plants, or other fungi. Some fungi live on dead bodies of other organisms, but they also have interaction with living organisms through various types of interactions. The fundamental body structure of fungi is hyphae, a fibrous structure that enables them to penetrate into the substrates (materials on which fungi grows or attached). When living substrates are considered, they are called “hosts”. They provide digestive enzymes in their hosts and absorbing dissolved molecules from hosts. Biologists observed that the similar fungal species are more likely to parasitize on the specific range of hosts, so it is possible to determine the distribution of them. The fungi may produce spores for asexual or sexual production on the host, or they may produce fruiting body (mushrooms) recognizable in naked eyes.

In agriculture, the attack of fungi, especially rust fungi, is a serious issue that results in fungal diseases of plant. The life cycle of example of a fungus is demonstrated in Fig. 2.10. Because fungal parasitism damages the cells of healthy plants through their life cycle [27], the distribution of fungi provides the negative impact to economy in the large area. It would be a significant contribution if farmers understand and predict the host-range, so they can put effort to protect their commodity.

Although according review of this section is a small piece of the whole study of biodiversity, it is enough for being background of this thesis. The biodiversity knowledge is traditionally conserved in books and published papers, but the shift in information technology brings the knowledge into digital representation. Many information systems for biodiversity are implemented for managing biodiversity information, for example LODAC [146], GBIF [140], uBio [106], Catalog of Life [63], ZooBank [100], MycoBank [25], etc. Integrating data among systems is one of timely issues in biodiversity informatics.

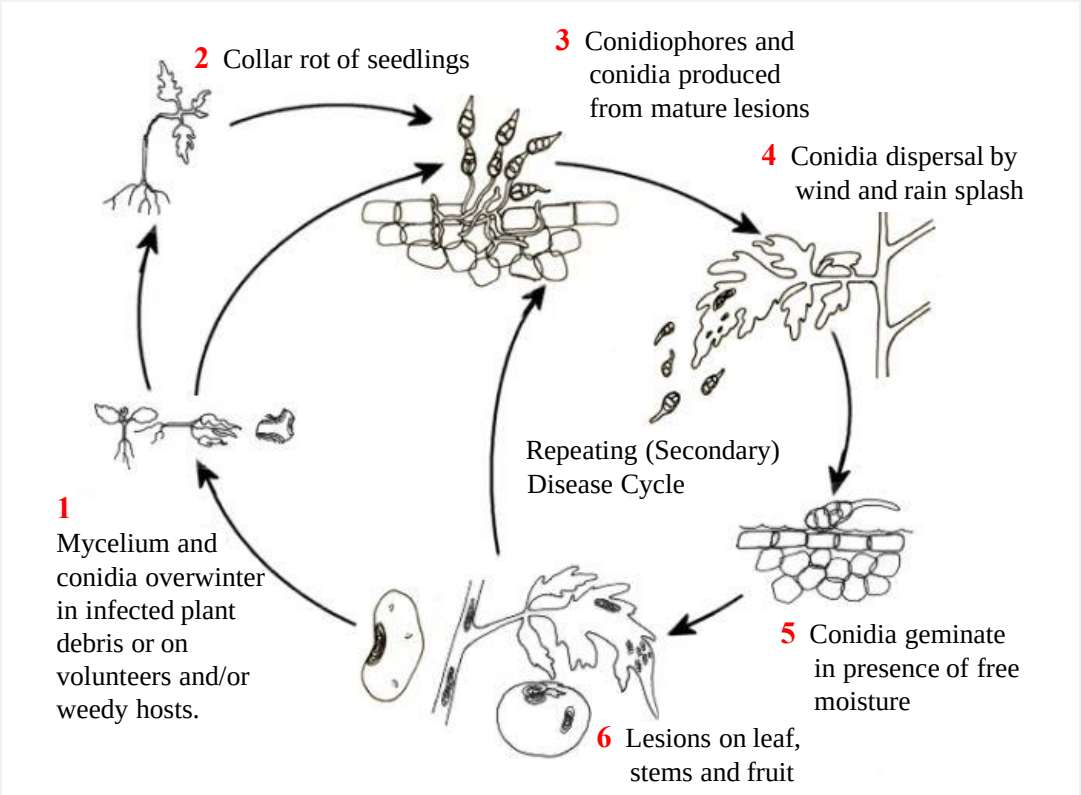


Fig. 2.10: Life cycle of *Alternaria solani*.
(Source: Early blight of potato and tomato [65])

2.4. The Challenge of Managing Knowledge Graph in Biodiversity Informatics

Enhancing KM with LOD is possible to develop for managing knowledge graph. This thesis aims to study this approach to KM and LOD using the biodiversity domain; because the biodiversity knowledge is very close to human life, and we have the direct and indirect benefit of biodiversity every day, for example food, medicines, clothes, buildings, etc. Although some structured data and RDF data of biodiversity are existing and open access, the study of the available information and the behavior of biodiversity knowledge let us recognize the following challenges.

2.4.1. Change in Biodiversity Knowledge

The effective of the research activities in biology resulted in a lot of the new discovery of biodiversity knowledge, and together with the various perspectives of biologists, biodiversity knowledge is commonly changed [70, 118]. For example, the merging between some genera resulted in that all species under the old genera have to be transferred into the new genus and renamed for satisfying the zoological nomenclature [121]. When capturing any change in knowledge, it has to describe contextual information about what prior knowledge is, what new knowledge is, when knowledge change, who change it, etc. [21]. However, a single triple cannot include context due to the limitation of the binary relation [54]. When one fact is presented by a single triple and context is needed to be added, much more triples have to be entered in order to present that fact together with context, so the model becomes more complex. There is no a single solution for solving this issue. Taxon Meta-Ontology (TaxMeOn) [118] also proposed an appropriate model for presenting the change in taxonomy, but the model is not suitable for linking association between changes which is necessary for learning taxonomy. In order to support the according requirement, a new schema has to be designed. The designed structure of data should satisfy requirements that are related to any actual cases about the changes in taxonomy. The degree of complexity of the model and information granularity should be suitable for being recognized by users, applying to a wide range of applications, and being implemented by current tools and technologies.

2.4.2. Linking Biodiversity Data

It is known that LOD is mainly introduced for enhancing the ability of exchanging data through the Internet. In order to have a precise link, an identification of every datum must be assigned. In biodiversity, there are several specifications of taxonomic identifier such as names [25, 97, 100], globally unique identifiers [153, 155], URIs [109, 155], and human-readable URIs [72, 88]. Each approach has individual advantages and disadvantages. When concerning about the change in knowledge, identifying a taxon become more challenging. There are a lot of obsoleted names, accepted names, synonyms, homonyms, etc. [122] that make the meaning of identifier become dynamic, so the integration of contextual information and an identifier is necessary to be studied and implemented. Moreover, the clients can be either machines or non-computer-expert users, so the data structure should be lightweight and be read by both humans especially in non-computer-expert users and computers, and the complexity and information granularity have to be concerned as well.

2.4.3. Interspecies Interaction

To gain advantages of the nature efficiency and avoid some negative impacts from any misuses, biologists have to discover more biodiversity knowledge by working in laboratories

and observing more factual evidence from nature. Some nature observations consume high cost including time and budget, but they give benefit to the human life especially in agriculture domain. Finding more evidence about relationship between fungi and hosts supports the protection of crop diseases [90]. Using a computer to predict the potential fungus-host interactions is possible to do, and the prediction result becomes a guideline for biologists for making observation. However, the existing dataset is too sparse to make a prediction by one scoring function alone [102]. Thus, it needs to study about how some features and structure of linked data such as link prediction, network analysis and some metadata can support the prediction of fungus-host interactions.

2.4.4. Node-Link Diagram for Biodiversity Data

It is known that learning biology from a node-link diagram or a concept-map diagram can enhance the learning ability of learners especially in biology [80]. Generating a graph diagram from RDF data is straightforward. However, in practice, many RDF repositories such as DBpedia [72] and LODAC [88] use some schemas that rely on the principle of RDFS and OWL, so a lot of inferred triples are generated by the entailment process. The densely complex graph is good for data retrieval, however it becomes a big problem in visualization due to a lot of semantically related resources and predicates displayed like a hairball. Many researches focused on how to summarize the big RDF data into a chart, a map, or an interactive visualization [33, 46, 85, 99, 119]. However, any automatic mechanism for simplifying a query graph has not been mentioned yet. As we analyzed, there are two main problems. (1) A query graph is too complicated to read due to a lot of inferred triples. (2) Lacking of reading flow of RDF data, so users cannot rearrange data for reading from introduction to main content. Thus, in order to be convenient to read by users, the graph has to be simplified and triples should be well-arrangement automatically.

……

CHAPTER 3

BIODIVERSITY KNOWLEDGE CAPTURE

"Scientific knowledge is in perpetual evolution;
it finds itself changed from one day to the next."

- Jean Piaget

One capital issue of knowledge capture in biodiversity domain is that taxonomic knowledge is not stable. The new discovery and the various perspectives of the classification systems for organismal groups led to inconsistent information among different taxonomic databases. This issue results in the ambiguity in taxon interpretation and consequently affects the other knowledge management activities. Although some pieces of research in earlier stages employed the Linked Open Data (LOD) technique to establish links in taxonomy transition, they overlooked the temporal representation of taxa and underlying knowledge of the change in taxonomy that is necessary for learning biodiversity. To this end, this chapter is aimed at developing a model for capturing the change in taxonomic knowledge using the Resource Description Framework (RDF). The Linked Taxonomic Knowledge (LTK) approach is developed, and this approach intends to initiate a simple data model for supporting the real-world changes in taxonomy. It includes the declaration of taxonomic entities, event-centric model, and operations of changes in knowledge. The results show that the proposed model is able to handle various practical cases of changes in taxonomic activities. In addition, it is noted that the proposed data model mainly benefits to the knowledge exchange, so the description of the application on this approach is transferred to the next chapter.

3.1. Overview

There are large number of species in the world, and they are described and classified with standard naming according to their characteristics such as morphological characters, living behaviors, and DNA sequences [78, 122]. Many taxonomists have described living things in terms of organismal groups and published them for more than hundred years. Due to the limitation of technology, biodiversity knowledge has not always been shared among all researchers around the world completely. In addition, there is no consensus on classification systems among all taxonomists. In other words, taxonomists might have different perspectives when it comes to classifying and naming organismal groups. As a consequence, the name and the classification of a single species is assigned differently [122].

To describe this situation more clearly, we demonstrate cases of change in taxonomic knowledge in Section 3.2. Most case studies regard that the change in taxonomic knowledge are regular the change in name and change in classification [41, 78, 118, 122]. The example cases demonstrated the problems that occur when each name reflects particular details observed by each researcher. Due to such a change in taxonomic names, when learners who, studies biodiversity knowledge, accesses only information containing only the present scientific name, they sometimes miss important information that was written with its previous scientific names. It can say that the scientific names and taxonomy lack a single interpretation in biology [84, 128]. Thus, to understand biodiversity especially in taxonomy thoroughly, learners need to know all synonyms across multiple datasets and then link their associated information together via the Internet [50]. Thus, in order to have the precise knowledge of taxonomy, researchers have to pay attention to the significance of the change in taxa over time. Finding associations among the background knowledge of changes is also needed to be studied in order to understand the taxonomic knowledge more correctly.

This chapter describes the first half of Linked Taxonomic Knowledge (LTK) project to demonstrate the role of LOD in knowledge capture. Related work, case studies, LTK approach, evaluation, and summary are mentioned hereafter.

3.2. Case Study

We have studied some changes in taxonomic knowledge and collected some following examples.

First, the Chinese yellow swallowtail, named *Papilio xuthus* Linnaeus, 1767 is written in different names among different research institutes. For example *P. xuthulus* Bremer, 1861, *P. chinensis* Neuburger, 1900, *P. koxinga* Fruhstorfer, 1908, and *P. neoxuthus* Fruhstorfer, 1908 [122].

Second, when two or more taxa were recognized as the same organismal group, the only original name is accepted [132]. Thus, some species have to be reclassified and renamed due to the naming system [78, 122]. For example, in 2008, Hoare established the genus *Kendrickia* (ostracods). Then, in 2010, Kempf found that this genus was a primary junior homonym for *Kendrickia* Solem, 1985 (gastropods) and proposed the name *Dickhoarea* as a replacement name for the *Kendrickia* Hoare, 2008. This led to the subsequent change in species names; for instance, *Kendrickia asketos* has subsequently been renamed *Dickhoarea asketos* since Kempf announced the name in 2010 [122].

Next, the circumscription of a taxon can be changed due to the progress of taxonomic research [122]. Sometimes, it results in the change in species name. For example, the genus *Columba* (pigeons) has been split into five genera, *Patagioenas*, *Chloroenas*, *Lepidoenas*, *Oenoenas*, and *Columba*, where the latter *Columba* is narrower than the former one. Some species of the genus *Columba* have been assigned to one of these newly separated genera, for instance, *Columba speciosa* was changed to *Patagioenas speciosa* [7].

Another situation is to merge taxa such as on the genus level. When some genera were decided to be merged into a single taxon, their lower taxa such as species had to be transferred to the newly accepted genus [132]. According to nomenclature, these species had to be renamed to be consistent with the new genus name [78, 122]. For instance, two genera of owls, *Bubo* and *Nyctea*, were merged into the prior genus *Bubo*. Following the change in these genera, the scientific name of the snowy owl *Nyctea scandiaca* has been subsequently changed to *Bubo scandiacus* in order to satisfy the zoological nomenclature [121].

Moreover, some researchers may have an incorrect understanding of some taxon concepts as a result of them having been reclassified frequently, for example, a reclassification of the Baltimore oriole (*Icterus galbula* Linnaeus, 1758) and the Bullock's oriole (*I. bullockii* Swainson, 1827). In 1964, Sibley and Short argued that these two species should be merged into a single species [114]. As a result, the former name, *I. galbula*, became the accepted name, whereas *I. bullockii* was a junior synonym of *I. galbula*. In contrast, in 1995, research results regarding the DNA sequences of the two species led to the splitting of *I. galbula* into *I. galbula* and *I. bullockii* again [44]. Although these two species are currently separate, some information on *I. galbula*, especially which recorded between 1964 and 1995, might include important details on *I. bullockii*. Researchers sometimes obtain imprecise information when they simply search for information by using the name *I. galbula* only.

3.3. Related Work

In light of the issue of the change in biodiversity knowledge in RDF, this study is an attempt to capture this kind of knowledge for presenting the correct interpretation of taxonomic data. An approach to linking taxonomic data along with the precise context and preservation of their background information is clearly needed.

In this section, we review several pieces of research that are likely to solve the issue. A poor data model leads to the lack of linkability among different datasets [106]. A scientific name alone is not enough for introducing a precise link [13, 63, 66, 70, 96, 97, 106, 109, 128]. The International Organization for Plant Information (IOPI) model [13] used taxonomic names together with circumscription references as potential taxa for linking data among multiple taxonomic views. The Biodiversity Information Standard (TDWG) [153, 155] developed a standard for taxonomic data sharing among different datasets, adopted Life Science Identifiers (LSIDs) as Globally Unique Identifiers (GUIDs) for indexing taxa, and allowed having versions of taxon concepts. It also provided Darwin Core schema [138] containing vocabularies for describing taxonomic data. Page [96] and Jones et al. [63] employed LSIDs for taxonomic databases, and the links of LSIDs can associate information among various data sources. The Universal Biological Indexer and Organizer (uBio) also gave LSIDs to taxa for enhancing the power of federated search engines [106]. As every taxon has been indexed with an ID, relations between taxa can be given by using links between IDs [66]. Schulz et al. [109] embedded the taxonomy of living things into an ontology by using semantic technology. The hierarchy of taxon concepts was represented in the Resource Description Framework (RDF) [109, 155].

The pieces of research above have not yet mentioned about the preservation of changes in taxonomic knowledge. For this reason, TaxMeOn [118] developed a Semantic Web-based meta-ontology of biological names that managed and presented the changes in the scientific preposition of biological names and taxonomies such as splitting and lumping, and it emphasized how the biological names were published by referring to related publications. However, to the best of our knowledge, there is less discussion about the information structure of associations between any reasons behind changes or background knowledge, which is needed to make a clear understanding of taxonomy.

This challenge puts forward the view that an underlying knowledge of the changes in taxonomic knowledge is required for the correct interpretation of taxonomic data. The study of biodiversity informatics should focus on the inclusion of the historical changes in taxa and the context information that is essential for understanding the situation regarding their changes and how names are related as well.

3.4. Linked Taxonomic Knowledge (LTK): Data Model

Regarding the mentioned issue, here, we present a logical model named "Linked Taxonomic Knowledge" (LTK) for preserving and presenting the change in taxonomic knowledge for linked data. To achieve the goals and issues addressed in the previous sections, our logical model was developed on the basis of the following points.

- The model can capture the changes in biodiversity knowledge.
  - The model preserves the changes as an event along with aspects of time and provenance.
  - The model supports the changes in either taxa or association between taxa.
  - The model allows tracing the background knowledge of the changes by linking the cause and effect between them.
- The model can exchange biodiversity knowledge changed using a suitable format for a dataset for linked open data.
  - The linked data model deals with simple identifiers of Semantic Web resources in order to make the linked data be easily recognized by both humans and machines.
  - The model provides a sequence of changes in taxa.
  - The model presents temporal data on the basis of a given time point.

This chapter, Knowledge Capture, intends to describe the logical model according to the first point, the data model for managing the change in taxonomic knowledge; but the second point is described in the next chapter, Knowledge Exchange.

In this section, we illustrate the types of changes in taxonomic knowledge, terms and descriptions, LTK data model, and a method for utilizing our approach in the Resource Description Framework (RDF).

3.4.1. Categorization of Changes in Taxonomic Knowledge

On the basis of the actual case studies [41, 44, 60, 62, 68, 114, 118, 121, 122], the changes can be categorized into the tree in Fig. 3.1. The figure shows that there are three main categories: changes in nomenclature, taxon concept, and relationship.

First, the category nomenclature refers to the change in taxonomic names including renaming, synonym, and homonym.

- **New name** is used when initially giving a name to a taxon.
- **Spelling sequence** means a Latin name is typo, so a correct name is published.
- **Synonym** is used when different names are assigned for the same taxon. When one name is accepted, that name becomes a senior synonym and the other names become junior synonyms.
- **Homonym** is used when the same name is assigned to different taxa.

The second category is about the change in taxon concept that denotes the change in the description of a taxon. It includes the life span of the name of taxa that are initially stated (creating) and made obsolete (ending), and it also involves with the replacement of taxa in different checklists. Moreover, the change in the scope of taxa that are merging, splitting, and change in circumscription are also included.

- **Creating** means to introduce a new taxon.
- **Ending** means to obsolete a taxon.
- **Replacing** means a taxon is replaced by another taxon.
- **Merging** means to lump taxa into a single taxon.
- **Splitting** is to separate a taxon into several taxa.
- **Change in circumscription** means to modify the scope of a single taxon.

In terms of zoology [132], the taxa before the change are assumed to be obsolete from the dataset, and then, the other taxa after the change become newly created.

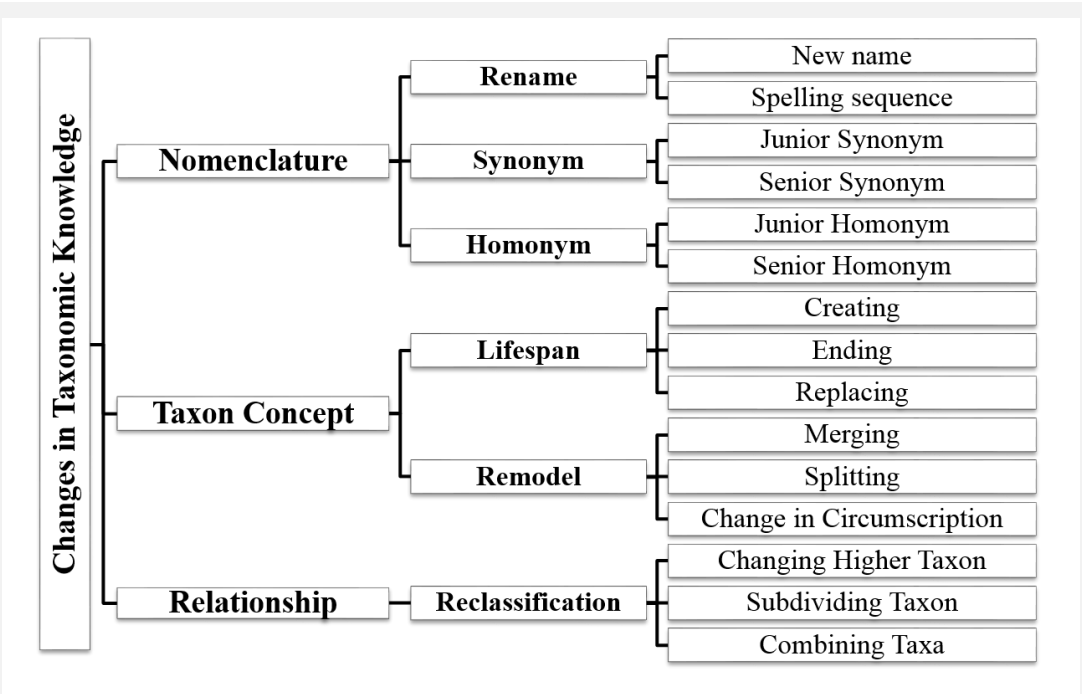


Fig. 3.1: Analysis of changes in taxonomic knowledge.

Last, the change in relationship means a modification made to a link between concepts. In terms of the Semantic Web, it is a change in a triple. In this figure, three changes are mentioned.

- **Change in higher taxon** transfers a lower taxon to a new a higher taxon.
- **Subdividing taxon** is to create new sub-taxa under the given taxon. It differs from splitting because the given taxon remains accepted, and its description does not change.

For example, a species *Aus aus* was subdivided into subspecies *A. aus aus* and *A. aus bus*.

- **Combining taxa** is the opposite of the subdividing. The sub-taxa of a given taxon are no longer used when all sub-taxa are combined into one concept and no subdivision is applicable. For example, when the two subspecies *A. aus aus* and *A. aus bus* are combined into one subspecies and there are no other valid subspecies, all subspecies names are no longer used.

3.4.2. Preliminary Definitions

A way to describe the changes in taxonomy along with context knowledge is a challenging task. In this project, we primarily employed the data model for digital archives of CKA [22], which offers a logical model for presenting the change in the underlying community knowledge on the basis of the theory of Flouris and Meghini [40]. The data model offers a schema for presenting a change with aspects of time and references in RDF. Since the approach of CKA conforms to our goal, we utilize the idea of CKA and enhance that framework for satisfying the specific requirements of biodiversity informatics. Here we introduce entities, operations, and data models used by this LTK project.

Entities for LTK

An entity in LTK is a URI for responding to specific positions, for example, entities for representing taxa, operations of changes, and events describing the changes. In this case, some terms are needed to be defined and clarified.

Nominal Entity

Semantic technology encourages that everything should be represented as an Internet resource identified by a URI [50]. Since there are no clear about the boundary of taxon concepts and taxonomic names, it is difficult to reuse these terms again in our framework. Thus, this research has to introduced a new term name nominal entity. This entity is a concept and an Internet resource used for taxonomic knowledge, and we can say that this entity includes taxon concepts and taxonomic names. Let *TC* be a set of taxon concepts, *TN* be a set of taxon names, and *IR* be a set of Internet resources, the set of nominal entities (*NOM*) is defined as follows:

$$TC \cup TN \subset NOM \subset IR$$

Simple Nominal Entity

This research moreover introduces a simple nominal entity as a subset of the nominal entity, and each of these entities corresponds to a single scientific name. Due to the change in knowledge, the role of a taxon has a lifespan. The simple nominal entity, which is an Internet resource, can act as either a taxon concept or a name according to the following situations. If a scientific name of any entity had been accepted for a certain period, that entity could be viewed as a taxon concept at any time in that period. In contrast, it becomes viewed as a taxonomic name when it is mentioned in another time. Thanks to the advantage of DBpedia [72] and LODAC [88], a human-readable URI makes a knowledge graph be human friendly, for example, *dbpedia:Bubo* and *lodac:Bubo*. We recommend using simple nominal entities for several reasons. A model is simple and lightweight, presented data are easily recognized by normal users, and a triple in a knowledge graph is more understandable. In addition, the issue of homonyms can be solved by using a different namespace.

Since the simple nominal entity (*SIM*) is a part of the nominal entity, the scope of the set of these entities is described as follows:

$$SIM \subset NOM$$

Contextual Nominal Entity

The change in knowledge sometimes has an impact on some representative taxa, and their circumscription or their name may be changed. Our work deals with this problem by applying the idea of TaxMeOn [118], which creates different URIs for the same taxon when its description is changed. We additionally define that every representation of taxonomy used in LTK is viewed as a version of a nominal entity.

In other words, the contextual nominal entity (*CON*) can be defined by the following expression.

$$CON \cup SIM \subset NOM$$

In the case of working with a simple nominal entity, this research provides the following recommendations.

1) A URI should include a scientific name and a version. We recommend using a year of the change as a version number such as *genus:Bubo_1999*.
2) If a change affects the change in nomenclature, a new URI should be created, and a link between the former and the latter URIs is developed to show the relationship between them.
3) In case that a new URI of a taxon concept is recreated for some purposes without a change in scientific name, the version number in the URI string should be updated.

Operations of Change

An operation of a change is a type of a change in taxonomic knowledge. As we previously described, there are changes in conception and relation. The change in conception is to capture the chronological change of a taxon such as replacing, merging, and splitting; while the change in relation the temporal link between taxa such as reclassifying, subdividing, combining, having synonym, etc.

Let *OPR* be the top operation of change, *OPRC* is the operation of change in conception, and *OPRR* is the operation of the change in relation, the relations between these entities are

$$OPRC \cup OPRR \subset OPR$$

where *OPRC* and *OPRR* are disjoint. In practice, writing the terms *OPRC* and *OPRR* in RDF is done by reusing the vocabularies from the CKA framework. Thus, the OPRC is represented by the URI named *cka:ConceptEvolution*, while the *OPRR* is represented by the URI named *cka:RelationEvolution*.

Next, let *TaxonReplacement*, *TaxonMerger*, and *TaxonSplitter* be the operation of replacing, merging, and splitting respectively, the relations with the operation of change in concept are

$$\begin{aligned} TaxonReplacement &\subset OPRC \\ TaxonMerger &\subset OPRC \\ TaxonSplitter &\subset OPRC \end{aligned}$$

where *TaxonReplacement*, *TaxonMerger*, and *TaxonSplitter* are disjoint.

For operations of changes in relation between taxa, let *ChangeHigherTaxon*, *SubdivideTaxon*, *CombineTaxa*, and *SynonymLink*, be the operation of reclassifying, subdividing, combining, and having synonym, some operations of change in relation are

$$\begin{aligned} ChangeHigherTaxon &\subset OPRR \\ SubdivideTaxon &\subset OPRR \\ CombineTaxa &\subset OPRR \\ SynonymLink &\subset OPRR \end{aligned}$$

where *ChangeHigherTaxon*, *SubdivideTaxon*, *CombineTaxa*, and *SynonymLink* are disjoint. Since these operations are examples of the changes in relation, when there are more operations, the new operations must be defined as subsets of *OPRR* as well.

The practical use of these operations is to create an instance of one operation and gives some parameters. Thus, it is possible to give a link between operations, and then the link can be viewed as a link between background knowledge. The way to add parameters of and links between operations is fully described in section 3.4.6.

Event Entity

There are many possible ways to attach some aspects of time and references into an operation. Adding them to any operation can be done directly, however, many operations sharing the same context may create a lot of duplicate data. To reduce data redundancy, an event entity is created to group some operations that share the same aspects of time and provenance. Thus, the time interval and references are assigned to the event entity. The set event entities (*EVENT*) is defined by

$$EVENT \subset IR$$

For the use of each entity, it is noted that our work does not restrict the representation of URIs. A simple nominal entity, an unfriendly identifier, or the separation of a scientific name and a taxon concept are possible to use as a URI in our model.

In addition, in this research, we view the nominal entity, simple nominal entity, and contextual nominal entity as concepts, which are subclasses of *skos:Concept*. Because a change usually performs an action with concepts, from now on, when we mention the term “concept” in the context of change or with an operation of change, we mostly refer to the contextual nominal entity.

Last, since each entity is a Semantic Web resource, we added symbols to the figures in order to distinguish the types of entities:

- *(nom)* is an instance of a nominal entity,
- *(sim)* is an instance of a simple nominal entity,
- *(con)* is an instance of a contextual nominal entity,
- *(OPR)* is a class of a change entity (operation),
- *(opr)* is an instance of an operation, and
- *(event)* is an instance of an event entity.

Data Models for LTK

In addition, to have researchers interpret data precisely, our knowledge management scheme introduces some various models of the representations of biodiversity knowledge.

Event-Centric Model

The event-centric model is a data structure that is used to preserve the change in taxonomic knowledge in RDF. It is based on the idea of CKA [22] that uses the *n*-ary relation for creating context-dependent RDF statements including operations, time intervals, and references [22, 40, 47]. Thus, the RDF presentation of this model is quite complicated by design. Although the model is expensive due to a lot of triples required, it is advantageous to various applications, especially in KM systems.

The event-centric model (M^{EVENT}) is simply defined as

$$M^{EVENT} = (EVENT, OPR, INV, REF, R_{EVENT\text{-}OPR}, R_{OPR\text{-}OPR}, R_{EVENT\text{-}INV}, R_{EVENT\text{-}REF})$$

where

- $EVENT$ is a set of event entities.
- OPR is a set of operations whose parameters are already assigned.
- INV is a set of time intervals. Each interval contains begin and end time points.
- REF is a set of references such as contributors and publications.
- $R_{EVENT\text{-}OPR}$ contains named relations between event entities and operations.
- $R_{OPR\text{-}OPR}$ contains named relations between two operations.
- $R_{EVENT\text{-}INV}$ contains named relations between event entities and intervals.
- $R_{EVENT\text{-}REF}$ contains named relations between event entities and references.

The detail of the event-centric model is described in section 3.4.3.

Transition Model

The transition model is a model for presenting the chain of changes in contextual nominal entities. This model is transformed from the event-centric model by using Semantic Web rules. This model is presented as a general knowledge graph including only contextual nominal entities and their links, so it is simpler than the event-centric model and it is easily exchanged with other regular linked data, but the representation of background knowledge of a change in detail is limited. It is noted that the description and the use of the transition model are described in the Chapter 4 knowledge exchange.

Snapshot Model

The snapshot model is a set of simple RDF statements like the transition model, but it is generated according to a given time point. This model demonstrates how the information of a taxon changes over time. It is noted that the description and the use of the snapshot model are also described in the Chapter 4 knowledge exchange.

3.4.3. Formal Model for Change in Taxonomy

As mentioned in the previous section, the change in contextual nominal entities and the change in the relation between them are key players in biodiversity knowledge capture and linking taxonomic knowledge. To present general definitions for the change in taxonomic knowledge, we propose a formal model for preserving and presenting the change in taxa for linked data. Our formal model enhances the data model for digital archives and reuses some vocabularies such as upper classes from CKA framework [22] to create an operation of the change in concepts and an operation of the change in relation between two concepts and how to map an operation with a Semantic Web property.

Operation of Changes (*OPR*)

Change in Conception

For the operation of change in taxonomic conception, main operations in this framework are *TaxonReplacement*, *TaxonMerger*, and *TaxonSplitter*.

- *TaxonReplacement* handles how to replace one taxon by another one taxon.
- *TaxonMerger* handles how to merge two or more taxa into one taxon.
- *TaxonSplitter* handles how to split one taxon into two or more taxa.

In addition, relations (or properties or predicates) indicating the parameters of these operation are described as follows:

- *before* is a relation that maps between an operation and a contextual nominal entity. This entity means a taxon that was accepted before the change, but it has been obsolete after the change. This relation is defined by
  $before : OPRC \times CON \rightarrow \{T,F\}$

- *majorBefore* has similar meaning with the *before*, but the *majorBefore* points to a taxon that is dominant in the change. In other words, that taxon holds a big part before merging. This relation is defined by
  $majorBefore : OPRC \times CON \rightarrow \{T,F\}$

- *after* is a relation that maps between an operation and a contextual nominal entity. This entity means a taxon that becomes accepted after the change. This relation is defined by
  $before : OPRC \times CON \rightarrow \{T,F\}$

- *majorAfter* has similar meaning with the *after*, but the *majorAfter* points to a taxon that is dominant in the change. In other words, that taxon hold the major part after splitting. This relation is defined by
  $majorAfter : OPRC \times CON \rightarrow \{T,F\}$

It is noted that all taxa presented in the same operation of change in conception must be the same taxonomic rank. For example, if the taxa *a1* and *b1* are merged into *a2*; *a1*, *b1*, and *a2* must be the same taxonomic rank, such as they must be only species, only genus, only family, etc. Thus, it cannot be done across taxonomic ranks, for example, merging a species and a genus into a genus is invalid.

In order to work with each operation, it needs to follow the according steps. In this case, we simulate how to capture the merging between the contextual nominal entities *a1* and *b1* into *a2*, and also assume that *a1* is the major part of *a2*.

1) To prepare instances of the contextual nominal entity that are used in operation. For example, *CON(a1), CON(b1),* and *CON(a2).*

2) To create an instance of an operation, for example, let *mg* is a member of *TaxonMerger*, this task can be expressed as *TaxonMerger(mg)*.

3) To inform that *a1* is the major part before merging with another concept into a new concept by using the relation named *majorBefore*. In this case, we express *majorBefore(mg, a1)*.

4) To inform that *b1* is a concept before merging with another concept into a new concept by using the relation named *before*. In this case, we express *before(mg, b1)*.

5) To inform that *a2* is the result after merging some taxa by using the relation named *after*. In this case, we express *after(mg, b1)*.

From the above steps, the formal expression of this change can be gathered as follows:

```
CON(a1) ∧ CON(b1) ∧ CON(a2)
∧ TaxonMerger(mg)
∧ majorBefore(mg, a1) ∧ before(mg, b1)
∧ after(mg, a2)
```

Change in Relationship between Taxa

In addition to the change in conception, the operations of the change in relation between two concepts is formally defined. The operation can be changing a higher taxon, subdividing a taxon, combining taxa, synonym link etc. The number of operations is not limited by these four operations, but developers are possible to define more operations for own purposes. The guideline is discussed in Section 3.4.7.

In this section, only four operations are described.

- *ChangeHigherTaxon* handles how to change the biological classification of a taxon. For example, a genus *ge1* has been reclassified from a family *fam1* to a new family *fam2*.
- *SubdivideTaxon* handles how to subdivide a higher taxon into two or more lower taxa. For example, a species *sp1* has been subdivided into two subspecies *subsp1* and *subsp2*. This case differs from splitting because no taxa are obsolete. At first, there is *sp1* exist without any subspecies. After that, taxonomists have announced to create two new subspecies *subsp1* and *subsp2* under the *sp1*.
- *ConbineTaxa* handles how to combine lower taxa into a higher taxon. For example, two subspecies *subsp1* and *subsp2* have been combined into a species *sp1*. In this case, at first, the species *sp1* has two subspecies *subsp1* and *subsp2*. After that, taxonomists preferred to merge both subspecies and found that the species *sp1* would has only one subspecies. Since a single subspecies under a single species is redundant, both subspecies were obsolete by combining into that species. In this case the species *sp1* is still the same before and after the change.
- *SynonymLink* handles how to give a synonym for taxa.

In addition, relations (or properties or predicates) indicating the parameters of these operation are described as follows:

- *subject* is a relation that maps between an operation and a nominal entity. This entity means the subject of an operation, and this value is not changed. This property is defined by
  subject : $OPRR \times NOM \rightarrow \{T,F\}$

- *objectBefore* is a relation that maps between an operation and a nominal entity. This entity means the object side of an operation before the change. This property is defined by
  objectBefore : $OPRR \times NOM \rightarrow \{T,F\}$

- *objectAfter* is a relation that maps between an operation and a nominal entity. This entity means the object side of an operation after the change. This property is defined by
  before : $OPRR \times CON \rightarrow \{T,F\}$

In addition, the terms *subject*, *objectBefore*, and *objectAfter* may not be friendly for readers, the LTK prepares some other easily-recognizable relations as follows:

For the operation *ChangeHigherTaxon*, we recommend to use the term *child*, *parentBefore*, and *parentAfter* instead of subject, *objectBefore*, and *objectAfter* respectively; because LTK offers that *child* $\sqsubseteq$ *subject*, *parentBefore* $\sqsubseteq$ *objectBefore*, and *parentAfter* $\sqsubseteq$ *objectAfter*.

For the operation *SubdivideTaxon*, *CombineTaxa*, *SynonymLink*, and other operations having meaning as a linking relation such as *HomonymLink*, *CorrectSpellingLink*, etc.; we recommend to use the term *sourceTaxon* and *targetTaxon* instead of *subject* and *objectAfter*; because *sourceTaxon* $\sqsubseteq$ *subject*, and *targetTaxon* $\sqsubseteq$ *objectAfter*. These operations do not mention about the *objectBefore* because they refer to a newly created relation rather than change from something to another thing.

To make it more clear, we introduce some steps for working with these operations. In this case, we simulate how to reclassify a lower taxon *x1* from the higher taxon *b1* to the higher taxon *a2*.

1) To prepare instances of the nominal entity that are used in operation. For example, *CON(x1), CON(b1),* and *CON(a2)*.

2) To create an instance of an operation, for example, let *reclass* is a member of *ChangeHigherTaxon*, this task can be expressed as *ChangeHigherTaxon(reclass)*.

3) To inform a subject of this operation, in this case, it is *x1* because it is about the change of *x1*. we express *subject(reclass, x1)*.

4) To inform an object before the change of this operation, in this case, the old higher taxon is *b1*, so we express *parentBefore(reclass, b1)*.

5) To inform an object after the change of this operation, in this case, the new higher taxon is *a2*, so we express *parentAfter(reclass, a2)*.

From the above steps, the formal expression of this change can be gathered as follows:

```
CON(x1) ∧ CON(b1) ∧ CON(a2)
∧ ChangeHigherTaxon(reclass)
∧ subject(reclass, x1)
∧ parentBefore(reclass, b1)∧ parentAfter(reclass, a2)
```

Relation between two Operations ($R_{OPR\text{-}OPR}$)

When one operation contributes to another operation, a link between operation can be defined. In this case, there are three relations.

- *effect* is used when an operation in the first argument contributes to another operation in the second argument. This relation is defined by
  $effect : OPR \times OPR \rightarrow \{T,F\}$

- *cause* is an inverse meaning of the *effect*. This relation is defined by
  $cause : OPR \times OPR \rightarrow \{T,F\}$

- *detail* is sometimes used for linking details of a newly created concept after a change such as adding higher taxon of a new taxon. This relation is defined by
  $detail : OPR \times OPR \rightarrow \{T,F\}$

For example, regarding the previous example, the instance *mg* explains the merging between *a1* and *b1* into *a2*, after that the instance *reclass* explain the reclasfying of *x1* from *b2* to *a2*. In this case, we assume that the operation *mg* contributes to the operation *reclass*, so it can be expressed as *effect(mg, reclass)* or *cause(reclass, mg)*.

Event Entity (*EVENT*)

The event entity is an event of a set of change together with aspect of times and references. For example, if the instance *ev* is an event, it can be defined as *EVENT(ev)*.

Relation between Event Entities and Operations ($R_{EVENT\text{-}OPR}$)

Next, the relation between an event entity and an operation can be expressed by the relation named *assure*. This relation is defined by

$assure : EVENT \times OPR \rightarrow \{T,F\}$

For example, the event *ev* assures the operation *mg* and *reclass* can be written by *assure(ev,mg)* and *assure(ev,reclass)*.

Relation between Event Entities and Intervals ($R_{EVENT\text{-}INV}$)

For defining the relation between an event entity and an interval, it firstly needs to describe about the interval.

- *INV* is a set of interval time that includes a begin time point and an end time point.

- *TP* is a set of time points.

- *begins* is a relation between an interval and a begin time point. It is defined by
  $begins : INV \times TP \rightarrow \{T,F\}$

- *ends* is a relation between an interval and an end time point. It is defined by
  $ends : INV \times TP \rightarrow \{T,F\}$

For example, an instance *inv* is an interval that begins at a time point *t1* and ends at a time point *t2*. This statement can be expressed by

$INV(inv) \wedge TP(t1) \wedge TP(t2) \wedge begins(inv, t1) \wedge ends(inv, t2)$

After that, the relation between an event entity and interval can be defined by a relation named *interval* where

$interval : EVENT \times INV \rightarrow \{T,F\}$

For example, the interval *inv* of the event *ev* can be written by *interval(ev, inv)*.

Relation between Event Entities and References ($R_{EVENT\text{-}REF}$)

To map an event entity with references, there are some classes and relations as follows:

- *DOC* is a set of documents including publications.
- *PERSON* is a set of persons.
- *source* is a relation between an event and a document. It is defined by
  $source : EVENT \times DOC \rightarrow \{T,F\}$
- *performer* is a relation between an event and a person who discovers this event of change. This relation is defined by
  $performer : EVENT \times PERSON \rightarrow \{T,F\}$
- *issuer* is a relation between an event and a person who report this event of change. If it is the same person as performer, this relation can be ignored. This relation is defined by
  $issuer : EVENT \times PERSON \rightarrow \{T,F\}$

For example, the event *ev* has been discovered by a person named *john* and the detail has been written in an academic paper *doc1*. This sentence can be expressed by

$DOC(doc1) \wedge PERSON(john) \wedge source(ev, doc1) \wedge performer(ev, john)$

Example of an Event-Centric Model

According to each example from each topic above, they can be combined into the following statement in the predicate logic.

```
1  CON(a1) ∧
2  CON(b1) ∧
3  CON(a2) ∧
4  CON(x1) ∧

5  EVENT(ev) ∧

6  INV(inv) ∧
7  TP(t1) ∧ begins(inv, t1) ∧
8  TP(t2) ∧ ends(inv, t2) ∧
9  interval(ev, inv) ∧
```

```
10  DOC(doc1) ∧
11  source(ev, doc1) ∧

12  PERSON(john) ∧
13  performer(ev, john) ∧

14  TaxonMerger(mg) ∧
15  majorBefore(mg, a1) ∧
16  before(mg, b1) ∧
17  after(mg, a2) ∧
18  assures(ev, mg) ∧

19  ChangeHigherTaxon(reclass) ∧
20  subject(reclass, x1) ∧
21  parentBefore(reclass, b1) ∧
22  parentAfter(reclass, a2) ∧
23  assures(ev, reclass) ∧

24  effect(mg, reclass)
```

This expression can be explained by the following list.

- Lines 1-4 are the declarations of taxa.
- Line 5 is the declaration of an event entity.
- Lines 6-9 show the interval of this event.
- Lines 10-11 show the reference of this event.
- Lines 12-13 show the person involving with this event.
- Lines 14-17 show the operation of merging between taxa.
- Line 18 shows that the merger is assured by this event.
- Lines 19-22 show the operation of reclassifying a taxon.
- Line 23 shows that the reclassification is assured by this taxon.
- Line 24 is the relation between both operations.

3.4.4. RDF Vocabularies for LTK

After introducing the formal model for the change in taxonomy, this section gives mapping between formal terms and RDF vocabularies as shown in Table 3.1. Some of vocabularies are reused from some known ontologies. It is noted that the full meta-ontology including data constraint is written in Appendix.

Table 3.1: Mapping between formal terms and RDF vocabularies I

Descriptions	Terms	RDF Vocabularies
Classes		
Nominal entity	*NOM*	*ltk:NominalEntity*
Simple nominal entity	*SIM*	*ltk:SimpleNominalEntity*
Contextual nominal entity	*CON*	*ltk:ContextualNominalEntity*
Event entity	*EVENT*	*cka:CommunityKnowldge*
Interval	*INV*	*tl:interval*
Document	*DOC*	*foaf:Document*
Person	*PERSON*	*foaf:Person*
Operation	*OPR*	*cka:Operation*

Descriptions	Terms	RDF Vocabularies
Operation of the change in conception	*OPRC*	*cka:ConceptEvolution*
Operation of the change in relation between taxa	*OPRR*	*cka:RelationEvolution*
Operation for replacing a taxon	*TaxonReplacement*	*ltk:TaxonReplacement*
Operation for merging taxa	*TaxonMerger*	*ltk:TaxonMerger*
Operation for splitting a taxon	*TaxonSplitter*	*ltk:TaxonSplitter*
Operation for changing higher taxon	*ChangeHigherTaxon*	*ltk:ChangeHigherTaxon*
Operation for subdividing a taxon	*SubdivideTaxon*	*ltk:SubdivideTaxon*
Combining taxa	*CombineTaxa*	*ltk:CombineTaxa*
Linking synonym	*SynonymLink*	*ltk:SynonymLink*
Properties		
sub set of a class	⊂	*rdfs:subClassOf*
is element of a class	*Class(element)*	*rdf:type*
has interval	*interval*	*tl:interval*
begins at a time point	*begins*	*tl:beginsAtDateTime*
ends at a time point	*ends*	*tl:endsAtDateTime*
has a source document	*source*	*dct:source*
has a performer	*performer*	*bibo:performer*
has a reporter	*issuer*	*bibo:issuer*
assures an operation	*assures*	*cka:assures*
a concept before changing	*before*	*cka:conceptBefore* *ltk:taxonBefore*
a major concept before changing	*majorBefore*	*cka:majorConceptBefore* *ltk:majorTaxonBefore*
a concept after changing	*after*	*cka:conceptAfter* *ltk:taxonAfter*
a major concept after changing	*majorAfter*	*cka:majorConceptAfter* *ltk:majorTaxonAfter*
subject	*subject* *subjectTaxon* *child* *sourceTaxon*	*cka:subject* *ltk:subjectTaxon* *ltk:child* *ltk:sourceTaxon*
an object before changing	*objectBefore* *objectTaxonBefore* *parentBefore* *higherTaxonBefore*	*cka:objectBefore* *ltk:objectTaxonBefore* *ltk:parentBefore* *ltk:higherTaxonBefore*
an object after changing	*objectAfter* *objectTaxonAfter* *parentAfter* *higherTaxonAfter* *targetTaxon*	*cka:objectAfter* *ltk:objectTaxonAfter* *ltk:parentAfter* *ltk:higherTaxonAfter* *ltk:targetTaxon*
effect	*effect*	*cka:effect*
cause	*cause*	*cka:cause*
detail	*detail*	*cka:detail*
Datatypes		
Time point	*TP*	*xsd:DateTime*

After some RDF vocabularies are introduced, here is the RDF expression for an example of the event centric model from section 3.4.3. The comparison between the formal expression

and the RDF statement can be done line by line. Every instance in the example is transformed into a URI using a pseudo prefix *ex:*.

```
ex:a1       rdf:type    ltk:ContextualNominalEntity .
ex:b1       rdf:type    ltk:ContextualNominalEntity .
ex:a2       rdf:type    ltk:ContextualNominalEntity .
ex:x1       rdf:type    ltk:ContextualNominalEntity .

ex:ev       rdf:type    cka:CommunityKnowledge .

ex:inv      rdf:type    tl:Interval ;
            tl:beginsAtDateTime   "t1"^^xsd:DateTime ;
            tl:endsAtDateTime     "t2"^^xsd:DateTime .
ex:ev       tl:interval      ex:inv .

ex:doc1     rdf:type         foaf:Document .
ex:ev       dct:source       ex:doc1 .

ex:john     rdf:type         foaf:Person .
ex:ev       bibo:performer   ex:john .

ex:mg       rdf:type               ltk:TaxonMerger ;
            ltk:majorTaxonBefore   ex:a1 ;
            ltk:taxonBefore        ex:b1 ;
            ltk:taxonAfter         ex:a2 .
ex:ev       cka:assures            ex:mg .

ex:reclass  rdf:type               ltk:ChangeHigherTaxon ;
            ltk:child              ex:x1 ;
            ltk:parentBefore       ex:b1 ;
            ltk:parentAfter        ex:a2 .
ex:ev       cka:assures            ex:reclass .

ex:mg       cka:effectex:reclass .
```

3.4.5. Working with a Simple Scenario

In this part, we present how to work with the even-centric model in order to capture the change in biodiversity knowledge. Here, we suppose the simple scenario of the change in biodiversity knowledge by the following steps.

- There are two families: Audae and Buidae.
- The family Buidae includes one genus named *Xus*.
- At time *t1*, Buidae is merged into Audiae.
- Subsequently, the genus *Xus* is regarded as a member of a new URI of Auidae.
- This scenario is assumed to end at time *t2*; however, the end time point can be ignored if this event is still active.

In this case, we have to do by the following steps.

- Assign URIs of contextual nominal entities for Auidae, Buidae, and *Xus*, which are *ex:Auidae_1*, *ex:Buidae_1*, and *ex:Xus_1*, respectively.

- When two families are merged into Audae at time *t1*, according the use of the contextual nominal entity, the model has to create a new URI of Auidae to be *ex:Auidae_2*.
- Then, the genus *ex:Xus_1* is transferred to the newer accepted family *ex:Auidae_2*.

In nomenclature, a taxon at the genus level or above does not need to change its scientific name when it is transferred to another higher taxon [78, 132]. Thus, the current URI of the genus *ex:Xus1* is retained. However, if a change in taxonomy contributes to the change in scientific name of a taxon, a new contextual nominal entity has to be created, and a link between an old concept and a new concept has to be identified. Fig. 3.2 demonstrates the changes in taxa, the change in relationship between them, and the event entity.

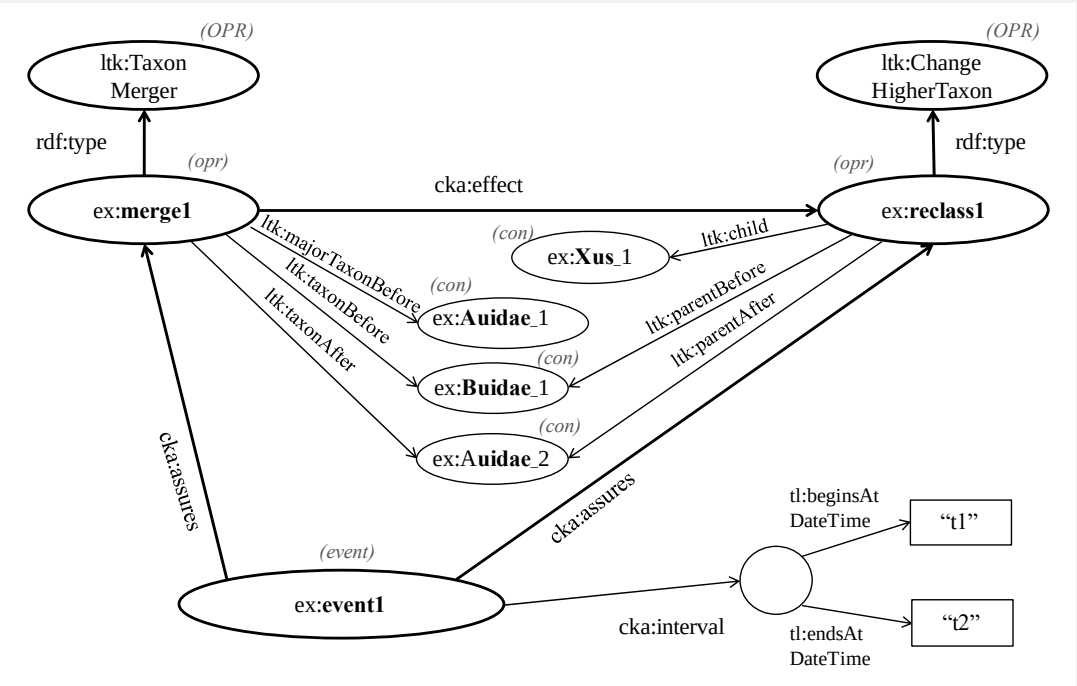


Fig. 3.2: LTK Model: Event-Centric Model

First, the operation, *ex:merge1*, is the merging of *ex:Auidae_1* and *ex:Buidae_1* into *ex:Auidae_2*. Thus, the given values of *ltk:taxonBefore* are *ex:Auidae_1* and *ex:Buidae_1*, while the given value of *ltk:taxonAfter* is *ex:Auidae_2*. However the *ex:Auidae_1* is dominant in this change, so it should use the property *ltk:majorTaxonBefore*.

Second, the change in relationship between contextual nominal entities, *ex:reclass1*, is the reclassification of *ex:Xus_1* from *ex:Buidae_1* to *ex:Auidae_2*. Hence, *ex:Xus_1*, *ex:Buidae_1*, and *ex:Auidae_2* are assigned to the properties *ltk:child*, *ltk:parentBefore*, and *ltk:parentAfter*, respectively.

Moreover, according to this scenario, *ex:merge1* contributes to *ex:reclass1*, so it can use *cka:effect* to map from *ex:merge1* to *ex:reclass1*, or use *cka:cause* to map from *ex:reclass1* to *ex:merge1*.

Last, the event entity named *ex:event1* assures the two changes as mentioned above by linking with a property named *cka:assures*, and it identifies a temporal identity by using a property named *cka:interval*. The temporal identity uses the property named

tl:beginsAtDateTime to assign the begin time point "*t1*", and uses the property named *tl:endsAtDateTime* to assign the end time point "*t2*".

3.4.6. Event-Centric Model with a Real Case

To link data with the LOD Cloud, we proposed useful operations that specify the changes in concepts, the changes in the details of a concept, the changes in relations between concepts, and the background information of the changes. All operations are defined by extending vocabularies from the well-known ontology such as SKOS and properties from LODAC and CKA. The namespaces and example properties used by our model are described in Appendix. As a result, the data from our approach can be linked to data from other repositories.

For instance, the old concepts *genus:Nyctea_1826* and *genus:Bubo_1805* have been merged into a new concept named *Bubo*. As stated previously, the new identifier of the genus *Bubo* has to be initiated because its new scope is larger than the former one. According to our recommendation, the identifier should be ended with a string representing the year in which the new URI was created, so the new identifier of *genus:Bubo_1805* becomes *genus:Bubo_1999*. To link between concepts before and after the change, LTK provides the property named *ltk:mergedInto* to represent the relation between a concept before and a concept after merging. As a result, the relation between *genus:Nyctea_1826* and *genus:Bubo_1999* remains to be represented by the property *ltk:mergedInto*. Moreover, in the case where the former concept and the latter concept have the same name or their circumscriptions are very close, the property *ltk:majorMergedInto* is recommended for demonstrating the very close relationship between them, such as *genus:Bubo_1805* and *genus:Bubo_1999*. To handle this situation, the model allows the use of the property *ltk:majorTaxonBefore* for the operation of merging and the property *ltk:majorTaxonAfter* for the operation of splitting. As the genus *Nyctea* was merged into the genus *Bubo*, all species under the genus *Nyctea*, such as *N. scandiaca*, have to be transferred into the genus *Bubo*; in this case, the name of this species has to be changed to *B. scandiacus* according to the nomenclature [78, 121, 122, 132]. The following RDF statements describe the merging of two genera, the renaming of a species under the genus *Nyctea*, and the change in a species under the genus *Bubo*. In this case, the *species:Bubo_scandiacus_1999* is newly generated without any higher taxa, so this event has to show the higher taxon of the according species by using the operation *ltk:HigherTaxonAddition* to originate a higher taxon of a newly generated URI. In addition, some references can be assigned to the event entity. Fore example, they are researchers who discovered the changes (*bibo:performer*), researchers who published the changes (*bibo:issuer*), and some publications (*dct:source*).

```
ex:event1999    bibo:performer  pp:Wing, pp:Heidrich ;
    bibo:issuer pp:Richard ;
    dct:source        pub:5224773 ;
    cka:interval      [tl:beginsAtDateTime "1999"] ;
    cka:assures       ex:mg1, ex:rp1, ex:ac1 .

ex:mg1    rdf:type              ltk:TaxonMerger ;
     ltk:majorTaxonBefore       genus:Bubo_1805 ;
     ltk:taxonBefore            genus:Nyctea_1826 ;
     ltk:taxontAfter            genus:Bubo_1999 .

ex:rp1    rdf:type            ltk:TaxonReplacement ;
          ltk:taxonBefore     species:Nyctea_scandiaca_1826 ;
          ltk:taxonAfter      species:Bubo_scandiacus_1999 .
```

```
ex:ac1    rdf:type            ltk:HigherTaxonAddition ;
          ltk:child           species:Bubo_scandiacus_1999 ;
          ltk:parentAfter     genus:Bubo_1999 .

ex:mg1  cka:effect  ex:rp1 .
ex:rp1  cka:detail  ex:ac1 .
```

3.4.7. Working with other Operations

Technically, the CKA framework allows other ontologies to customize their own operations of changes for particular purposes. This work is done by extending either the class *cka:ConceptEvolution* for changing a concept's scope or the class *cka:RelationEvolution* for changing a binary relation between two concepts. For example, the operations of the change in taxon concepts, such as *ltk:TaxonMerger* and *ltk:TaxonSplitter*, are descended from *cka:ConceptEvolution*. Thus, when there are new properties that are not a part of either CKA or LTK, such as morphological, molecular, or ecological traits; new operations need to be initiated by extending one of the mentioned classes from CKA and then binding the new operations with related properties. The following statement is the pattern to give a new operation, where *ex:changeSomething* is an example operation.

```
ex:changeSomething  rdfs:subClassOf cka:ConceptEvolution .
```

or

```
ex:changeSomething  rdfs:subClassOf cka:RelationEvolution .
```

In addition, although this research focuses on the change in taxonomic data, some triples that are not changed over time are recommended to be preserved by the even-centric model because it can present essential metadata such as a date added and references. Moreover, if some domains require more operations of changes, the operations can be created by extending *cka:RelationEvolution*. This method is also compatible with systems that separate a taxon concept and a name. Our model also allows having operations for either the object property or datatype property. Example properties or attributes are those such as *skos:prefLabel* [150], *foaf:depiction* [139], *dwc:identificationID* [138], *dwc:taxonID* [138], *dwc:scientificNameID* [138], *dwc:scientificName* [138], and *lodac:hasCommonName* [146]. However, for the datatype property, the object should be come with the properties *cka:valueBefore* and *cka:valueAfter* instead of the properties *ltk:objectTaxonBefore* and *ltk:objectTaxonAfter* Some details of them are described in Appendix.

In conclusion, the introduced logical model includes the data model for the change in taxonomic knowledge. It also presents how to use the model for real-world cases of the change in taxonomic knowledge in RDF. However, if more properties are needed for a specific purpose, developers can customize their operations by extending this framework.

3.5. Evaluation

The evaluation of this chapter aims to demonstrate the practicality of the introduced event-centric model and the nominal entity that are feasible and possible to capture the actual changes

in biodiversity knowledge. Thus, we use some real test cases from literatures that are agreed by domain experts to verify our proposed model.

We imported the example cases from the case study in Section 3.2 and some data on Japanese moths of the family Saturniidae published as three checklists (list of names): Inoue in 1982 [60], Jinbo in 2008 [62], and Kishida in 2011 [68]. One of the authors, Jinbo, analyzed the difference among these three checklists and finalized them into the changes in taxa among these checklists. The data cover operations of changes, which are creating a concept, making a concept obsolete, replacing a taxon, merging taxa, splitting a taxon, linking synonym, changing a higher taxon, subdividing a taxon, and combining taxa. This experiment contains 40 instances of operations together with 60 taxa from several taxonomic ranks: family, subfamily, genus, species, and subspecies. Here, we choose one example. In [60], the species *Caligula boisduvalii* has two subspecies, *Caligula boisduvalii fallax* and *Caligula boisduvalii jonasii*. In the subsequent study, this species was transferred from the genus *Caligula* to *Saturnia*, one of its subspecies *jonasii* was raised into a distinct species, and another subspecies, *fallax*, was regarded as a subspecies of *jonasii*. Hence, in that study, *Caligula boisduvalii* in [60] was redefined as two species, *Saturnia boisduvalii* and *Saturnia jonasii*. At the same time, the latter species was split into two subspecies, *Saturnia jonasii jonasii* and *Saturnia jonasii fallax*. These changes were adopted in the second checklist [62]. After a few years, both subspecies were combined into the species *S. jonasii* in [68]. These changes resulted in many links of synonyms. Even though these events are described in taxonomic papers, information on events is not included in each name and thus cannot be captured by the databases of scientific names. Some entities of background knowledge of the change in *S. jonasii* were linked so users could browse the accurate history of taxa, which is difficult to access for non-taxonomic experts. Therefore, the benefit of managing the change in concepts, such as presenting the links between concepts in the chain of the changes in taxonomic knowledge, temporal information about them, and the underlying knowledge of that change, made gathering correct data along with the precise context convenient. Therefore, it reduced confusion and helped avoid misunderstanding arising with respect to taxonomic data. This experiment proved that the LTK approach could deal with a real-world situation of changes in taxonomy.

Here is an RDF statements describing some changes in taxonomy of Moths in Japan in order to demonstrate the practicability of the LTK data model.

Change in Conception

There are three cases for the changes in conceptions.

Case 1: Replacing a taxon

This case shows that the species *Caligula boisduvalii* has been replaced to the species *Saturnia boisduvalii* since 2008.

```
ex:case1
      rdf:type         cka:CommunityKnowledge ;
      cka:interval     ex:case1_inv ;
      dct:source
            <http://www.jstor.org/discover/10.2307/4083714> ;
      bibo:performer
            <https://twitter.com/mothprog> .

ex:case1_inv
  tl:beginsAtDateTime  "2008-12-31T00:00:00"^^xsd:dateTime .
```

```
ex:case1   cka:assures     ex:case1_rp .

ex:case1_rp
      rdf:type         ltk:TaxonReplacement ;
      ltk:taxonBefore  species:Caligula_boisduvalii_1847 ;
      ltk:taxonAfter   species:Saturnia_boisduvalii_2008 .
```

Case 2: Merging taxa

This case shows that the subspecies *Antheraea yamamai yamamai* and the subspecies *A. yamamai ussuriensis* were decided to be merged into the subspecies *A. yamamai yamamai* in 2011.

```
ex:case2
      rdf:type         cka:CommunityKnowledge ;
      cka:interval     ex:case2_inv ;
      dct:source
            <http://www.jstor.org/discover/10.2307/4083714> ;
      bibo:performer
            <https://twitter.com/mothprog> .

ex:case2_inv
  tl:beginsAtDateTime  "2011-12-31T00:00:00"^^xsd:dateTime .

ex:case2   cka:assures     ex:case2_mg .

ex:case2_mg
      rdf:type         ltk:TaxonMerger ;
      ltk:majorTaxonBefore
            subspecies:Antheraea_yamamai_yamamai_1861 ;
      ltk:taxonBefore
            subspecies:Antheraea_yamamai_ussuriensis_1953 ;
      ltk:taxonAfter
            subspecies:Antheraea_yamamai_yamamai_2011 .
```

Case 3: Splitting a taxon

This case shows that the species *Loepa sakaei* has been split into two species *L. sakaei* and *L. katinka* since 2008, where the new *L. sakaei* holds the big part after splitting.

```
ex:case3
      rdf:type         cka:CommunityKnowledge ;
      cka:interval     ex:case3_inv ;
      dct:source
            <http://www.jstor.org/discover/10.2307/4083714> ;
      bibo:performer
            <https://twitter.com/mothprog> .

ex:case3_inv
  tl:beginsAtDateTime  "2008-12-31T00:00:00"^^xsd:dateTime .
```

```
ex:case3   cka:assures     ex:case3_sp .

ex:case3_sp
      rdf:type             ltk:TaxonSplitter ;
      ltk:taxonBefore      species:Loepa_sakaei_1965 ;
      ltk:majorTaxonAfter  species:Loepa_sakaei_2008 ;
      ltk:taxonAfter       species:Loepa_katinka_2008 .
```

Change in Relation between Taxa

There are four cases for the changes in relations between taxa.

Case 4: Changing a higher taxon

This case is the change of the family of a genus of birds. It shows that the family of the genus *Saxicola* has been changed from the family Turdidae to the family Muscicapidea since 2010.

```
ex:case4
      rdf:type         cka:CommunityKnowledge ;
      cka:interval     ex:case4_inv ;
      dct:source
            <http://www.ncbi.nlm.nih.gov/pubmed/20656044>  ;
      bibo:performer   pp:Sangster_G .

ex:case4_inv
  tl:beginAtDateTime  "2010-10-31T00:00:00"^^xsd:dateTime .

ex:case4   cka:assures     ex:case4_reclass .

ex:case4_reclass
      rdf:type             ltk:ChangeHigherTaxon ;
      ltk:child            genus:Saxicola_1802 ;
      ltk:parentBefore     family:Turdidae_1815 ;
      ltk:parentAfter      family:Muscicapidae_1825 .
```

Case 5: Dividing a taxon

This case shows that the species *Attacus atlas* was subdivided into two subspecies *A. atlas atlas* and *A. atlas ryukyuensis* during the time between 2008 and 2011.

```
ex:case5
      rdf:type         cka:CommunityKnowledge ;
      cka:interval     ex:case5_inv ;
      dct:source
            <http://www.jstor.org/discover/10.2307/4083714> ;
      bibo:performer
            <https://twitter.com/mothprog> .

ex:case5_inv
  tl:beginAtDateTime  "2008-12-31T00:00:00"^^xsd:dateTime ;
  tl:endAtDateTime    "2011-12-31T00:00:00"^^xsd:dateTime .
```

```
ex:case5   cka:assures     ex:case5_subdv .

ex:case5_subdv
      rdf:type         ltk:SubdivideTaxon ;
      ltk:sourceTaxon  species:Attacus_atlas_1758 ;
      ltk:targetTaxon
            subspecies:Attacus_atlas_atlas_2008 ,
            subspecies:Attacus_atlas_ryukyuensis_2008 .
```

Case 6: Combining taxa

This case shows that two subspecies *Saturnia jonasii fallax* and *S. jonasii jonasii* has been combined under the species *S. jonasii* since 2011

```
ex:case6
      rdf:type         cka:CommunityKnowledge ;
      cka:interval     ex:case6_inv ;
      dct:source
            <http://www.jstor.org/discover/10.2307/4083714> ;
      bibo:performer
            <https://twitter.com/mothprog> .

ex:case6_inv
  tl:beginAtDateTime  "2011-12-31T00:00:00"^^xsd:dateTime .

ex:case6   cka:assures     ex:case6_comb .

ex:case6_comb
      rdf:type         ltk:CombineTaxa ;
      ltk:sourceTaxon
            subspecies:Saturnia_jonasii_fallax_2008 ,
            subspecies:Saturnia_jonasii_jonasii_2008 ;
      ltk:targetTaxon  species:Saturnia_jonasii_2008 .
```

Case 7: Linking synonym

This case shows that the species *Loepa sakaei* has become a synonym of the species *L. Katinka* since *1965*.

```
ex:case7
      rdf:type         cka:CommunityKnowledge ;
      cka:interval     ex:case7_inv ;
            [tl:beginAtDateTime
                  "1965-12-31T00:00:00"^^xsd:dateTime] ;
      dct:source
            <http://www.jstor.org/discover/10.2307/4083714> ;
      bibo:performer   <https://twitter.com/mothprog> .

ex:case7_inv
  tl:beginAtDateTime  "1965-12-31T00:00:00"^^xsd:dateTime .
```

```
ex:case7   cka:assures     ex:case7_syn .

ex:case7_syn
      rdf:type         ltk:SynonymLink ;
      ltk:sourceTaxon  species:Loepa_katinka_1848 ;
      ltk:targetTaxon  species:Loepa_sakaei_1965 .
```

3.6. Summary

This chapter described the biodiversity knowledge capture that is the first half of the LTK project. Taxonomic entities: the nominal entity, the simple nominal entity, and the contextual nominal entity are introduced to be Internet resources for taxa. We also introduced the event-centric model that includes the operation of change in conception, the operation of change in relation, an aspect of time, and references. The key operations are replacing a taxon, merging taxa, splitting a taxon, changing a higher taxon, subdividing a taxon, combining taxa, and linking synonym, and the framework allows to create more operations according to specific requirements. The event-centric model uses RDF statement that is a binary relation to present the *n*-ary relation, so the model becomes complicated by design, but it is suitable for embedding contextual information and flexible for being applied by various applications. In terms of utilization, we showed that the LTK data model can handle the real cases of the changes in taxonomy of Japanese moths under the family Saturniidae and some other cases demonstrated in this chapter. The effort of this chapter helps to confirm that it is possible and feasible to use LOD in terms of knowledge graph and schema for capturing the change in biodiversity knowledge. Although the RDF statement presenting a change with context is not simple by design, the event-centric model can be transformed into a simple statement with less context, as discussed in the next chapter.

.....

CHAPTER **4**

BIODIVERSITY KNOWLEDGE EXCHANGE

"Simplicity is the glory of expression."

- Walt Whitman

Due to the change in biodiversity knowledge, taxonomic information especially in biological classifications are different and sometimes inconsistent among different taxonomic repositories. To have a precise understanding of taxonomy, one needs to integrate relevant data across taxonomic databases. This is difficult to establish due to the ambiguity in taxon interpretation. The previous chapter described about how to capture the change in taxonomic knowledge in RDF. The event-centric data model is proper for demonstrating a change in rich detail, but it is quite complex due to the integration of context information. However, for LOD, the model should be simplified in order to be easy to access and read by non-computer-expert users. Thus, two simpler models: transition model and snapshot model, are introduced to be view related models of the event-centric model for the purpose of linked data. Semantic web rules and a guideline for development are described. We also demonstrate the feasibility and the performance of this approach by implementing a prototype.

4.1. Overview

The previous chapter, biodiversity knowledge capture, mainly mentioned about entities representing taxa and the event-centric model. The event-centric model contains the change in taxonomy and context. The change in taxonomy is presented by an operation of change, for example, replacing a taxon, merging taxa, splitting a taxon, changing a higher taxon, subdividing a taxon, combining taxa, and linking synonym. The context includes a begin time point, an end time point, publications, contributors, etc. The principle of the design of our data model is similar to the idea of the normalization of database design and the flexible-reusable components of the object-oriented design. Thus, it becomes advantage in terms of knowledge preservation; however, in terms of knowledge exchange, this design seems complex for querying and exchanging, because other RDF repositories at the moment do not care much about integrating triples along with context.

Since the power of LOD itself is mainly focusing on data exchange, this project does not need to demonstrate the ability to exchange knowledge. Thus, the main objective of this chapter is to publish an appropriate representation of the change in taxonomy to LOD cloud.

In this case, two simple data models that are a transition model and a snapshot model are proposed. They are simple views of the event-centric model of which context is overlooked, and they can be transformed from the event-centric model directly using Semantic Web rules. The transition model is about the chronological change of taxa, while the snapshot model presents RDF graph corresponding to a specific time point. We also discuss more about the simple representation of URIs for presenting taxa. The outcome of this study is that an RDF statement become lightweight, simple, and easy to understand by non-computer-expert users, and is compatible with other RDF repositories.

This chapter describes the remaining part of Linked Taxonomic Knowledge (LTK) project in order to demonstrate the role of LOD in biodiversity knowledge exchange. Related work, case study, LTK approach, prototype, evaluation, and summary are written henceforward.

4.2. Related Work and Case Study

To materialize the formation of linked data, in this part, we studied how to utilize an Internet resource for representing the identifier of a taxon. There are several views on using identifiers such as Life Science Identifiers (LSIDs) or URIs, human-readable or non-human-readable identifiers, and representation of biodiversity knowledge, which are reviewed as follows.

4.2.1. Unique Identifier

The use of LSIDs as Globally Unique Identifiers (GUIDs) promoted by TDWG [153, 155] resulted in taxonomic data becoming globally available and linkable. Several information models adopted the LSID as a unique key representing a taxon in their databases [13, 63, 66, 70, 96, 97, 106, 109]. Jones et al. [63] resolved the multiple names by assigning separated LSIDs for a name (NAMELSID) and for a taxon (TAXONLSID), and they integrated an LSID into a URI. In addition, the authors of [70] compared the differences between the LSID and the URI and recommended using a URI as an Internet resource for taxonomic datum in order to gain benefit from LOD cloud. TaxMeOn [118] also put forward the view that taxon concepts are always changed, so a fixed identifier might not proper for every concept. Therefore, when a taxon's circumscription was changed, that concept needed to be recognized as a new

identifier. For instance, the genus *Bubo*, before merging with the genus *Nyctea*, must not have the same Semantic Web-based identifier as the *Bubo* after merging because the latter *Bubo* is broader than the former one [121, 122]. The model also allowed having a URI for a taxon concept and a URI for its name. It therefore had minimal redundancy and was flexible for updating either names or concepts. Nevertheless, TaxMeOn propounded the view that a taxon concept and its name were treated as one unit in a name collection. The domain or the range of properties is allowed to be a union of scientific names and taxon concepts.

4.2.2. Name-Centric Identifier

Patterson et al. [97] additionally introduced the Global Names Architecture (GNA) and supported the view that names were keys to access biological information. GNA, which mainly treats names with implicit taxon concepts, has three layers, but two layers are related to this topic. One is the Global Names Index (GNI), which is aimed at collecting name strings used in various information sources and normalized spellings. Another one is the Global Names Usage Bank (GNUB). It is aimed at describing name uses, which is a combination of a name and a reference, and nomenclatural issues. This name-centric model also provided features for identifying relationships between names, and it was integrated into online official repositories of names such as ZooBank [100] and MycoBank [25]. The authors of [70] argued that it was very challenging to combine a name and a taxon concept into a single unit because doing so decreased the granularity of information but gave high simplicity. In addition, naming conventions for identifiers are different among various systems. The Global Biodiversity Information Facility (GBIF), which is an international organization aiming to construct an information infrastructure for sharing information on biodiversity globally, gave a reference guide for GNA. It is a guideline for an information system to select some accepted names among all names used for living beings, and it recommends using an unfriendly label for a persistent identifier because a taxonomic name is not stable [26, 101, 103, 140]. The authors of [70] used non-human-readable local names in URIs. Whereas, TaxMeOn [118] does not specify the format of the URIs for data instances, so it is possible to use either human-readable or non-human-readable URIs. Furthermore, LODAC [88], which provided a linked data hub for biodiversity, denoted a URI as an Internet resource for representing a piece of taxonomic data. LODAC also considered including a human-readable label in URI in order to make the model be lightweight and human-friendly such as *lodac:Bubo*. It is consistent with the URIs of Internet resources used by DBpedia [72]. In this case, the human-readable URI is sometimes viewed as either a name or a taxon concept depending on the context. It also gives an advantage to humans, especially biologists, who involve with linked data, because the human-readable URI reduces the gap between machines and normal users.

4.2.3. Triple representing Knowledge

It is known that one triple contains only a single subject, a single predicate, and a single object, and this is the smallest component of an RDF graph. A single triple presents a relation from one thing to another thing. Some cases use a single triple to present one proposition while some other cases use two or more triples for presenting a proposition.

In case of the LTK project, it needs several triples to present a single proposition. For example, the replacement from the species *Nyctea scandiaca* to the species *Bubo scandiacus* consumes three triples as shown in the following RDF statements.

```
ex:rp1    rdf:type            ltk:TaxonReplacement .
ex:rp1    ltk:taxonBefore     species:Nyctea_scandiaca_1826 .
ex:rp1    ltk:taxonAfter      species:Bubo_scandiacus_1999 .
```

It means that if we need to find what is the replacing taxon of the species *Nyctea scandiaca*, we have to give a query statement as follows:

```
SELECT ?after
WHERE {
        ?opr rdf:type         ltk:TaxonREplacement .
        ?opr ltk:taxonBefore  species:Nyctea_scandiaca_1826 .
        ?opr ltk:taxonAfter   ?after .
}
```

This SPARQL expression can help users to find the right answer, however the expression of the condition is somehow complicated because it requires several steps to get the answer.

For the graph structure of TaxMeOn [118], a particular knowledge sometimes can be accessed by one hop in the query statement such as the property *tmo:congruentTaxon*, while some changes need more than two triples to present such as merging and splitting. If we need to find some taxa after splitting the genus *Galba*, we need to query by the following expression.

```
SELECT ?after
WHERE {
        ?x rdf:type           tmo:Split .
        ?x tmo:before         ex:Galba .
        ?x tmo:after          ?after .
}
```

In addition, some RDF repositories such as DBpedia [72] and LODAC [88] use a single triple for representing a particular knowledge, so the relation between taxon can be presented by a single subject, a single predicate, and a single object. Thus, for the similar question as above, the query statement is very simple. For example, if we need to find the synonym of a species *Argynnis aglaja*, the query statement is simply indicated as follows:

```
SELECT ?taxon
WHERE { species:Argynnis_aglaja lodac:hasSynonym ?taxon . }
```

The previous case is somehow simple because a URI contains a human-readable label. In case of an unfriendly label for a persistent identifier [26, 101, 103, 140], the query statement may not be simple as the previous case. For example, the query for the previous case may be advanced to be the following statement.

```
SELECT ?nameOfTaxon2
WHERE {
 ?taxon1  lodac:hasScientificName species:Argynnis_aglaja .
 ?taxon1  lodac:hasSynonym ?taxon2 .
 ?taxon2  lodac.hasScientificName ?nameOfTaxon2 .
}
```

or

```
SELECT ?nameOfTaxon2
WHERE {
 ?taxon1  rdf:label           "Argynnis aglaja" .
 ?taxon1  lodac:hasSynonym    ?taxon2 .
 ?taxon2  rdf:label           ?nameOfTaxon2 .
}
```

As can be seen, there are several solutions for presenting taxonomic knowledge due to the requirement of any systems. Some features about taxonomic databases such as a simple structure, information granularity, global linkablity, need of a resolver, etc. are needed to be considered and adjusted for inducing a data model.

4.3. Linked Taxonomic Knowledge (LTK): Linked Data

This section describes the remaining part of the LTK project in terms of biodiversity knowledge exchange using LOD. To achieve the objective that the model can be used to publish an appropriate representation of the change in taxonomy in LOD cloud, it needs to concern about the following points.

- The linked data model deals with simple identifiers of Semantic Web resources in order to make the linked data be easily recognized by both humans and machines.
- The model provides a sequence of changes in taxa.
- The model presents temporal data on the basis of a given time point.

In order to make data exchange be consistent with other repositories such as DBpedia [72] and LODAC [88], we consider to use a traditional triple ⟨ *subject, predicate, object* ⟩ where all predicates and objects surround a subject forming a star-like model. To address these points, simple URIs for taxonomy, a transition model, a snapshot model, meta-ontology of LTK, methods to create both models are described together with the role of LTK in LOD cloud.

4.3.1. Simple URIs for Taxonomy

To have simple URIs for taxonomy, we have recalled the two introduced entities: the simple nominal entity (*SIM*) and the contextual nominal entity (*CON*). Both entities include human readable name e.g. a scientific name.

For the simple nominal entity, a scientific name is included at the end of URI directly. For example, the scientific names are written in bold letters as follows:

- *http://live.dbpedia.org/resource/**Bubo***
- *http://live.dbpedia.org/resource/**Bubo_scandiacus***
- *http://lod.ac/species/**Bubo_virginianus***
- *http://lod.ac/species/**Strix_virginiana***
- *http://rc.lodac.nii.ac.jp/taxon/genus/**Bubo***
- *http://rc.lodac.nii.ac.jp/taxon/genus/**Icterus***
- *http://rc.lodac.nii.ac.jp/taxon/species/**Icterus_galbula***

Thus, when they are in short-hand writing, they become easily readable by humans such as *genus:Icterus*, *species:Icterus_galbula*, etc.

In case of merging between the *genus:Bubo* and the *genus:Nyctea* into the *genus:Bubo* in 1999, using the same *genus:Bubo* before and after merging leads to the misunderstanding of learners when they query information about the *genus:Bubo*. In other words, the *Bubo* before merging and the *Bubo* after merging are not the same thing, because the latter is wider than the former. Thus, we have to integrate a URI along with context called the contextual nominal entity. This entity is formerly introduced in the section 3.4.2. According to that section, we recommend using a year of the change as a version number such as *genus:Bubo_1999* if a change affects the change in name and circumscription. For example,

- *http://rc.lodac.nii.ac.jp/taxon/genus/**Bubo_1805***
- *http://rc.lodac.nii.ac.jp/taxon/genus/**Nyctea_1826***
- *http://rc.lodac.nii.ac.jp/taxon/genus/**Bubo_1999***

The created contextual nominal entity can link to nominal entities from external datasets in order to make the knowledge graph be globally linkable. According to the standard of TDWG [153], our research uses the property *dct:isVersionOf* for linking between a contextual nominal entity and a nominal entity.

In practice, we make a simple nominal entity be a representative of an external URI for maintaining links between the LTK dataset and external datasets. It is possible to link a contextual nominal entity with other taxonomic data such as the URIs or LSIDs from TDWG [153], GBIF [26, 103, 140], Catalog of Life (CoL) [63], LODAC [88], and DBpedia [72] via those representatives. For example, the following statement addresses an association among the contextual nominal entity (*genus:Bubo_1999*), the simple nominal entity as the representative of any external URIs (*genus:Bubo*), and the external URIs and LSIDs viewed as the nominal entity (*gbif:5959091*, *lodac:Bubo*, and *urn:lsid:ubio.org:namebank:2473659*).

```
genus:Bubo_1999 dct:isVersionOf genus:Bubo .

genus:Bubo
     owl:sameAs       gbif:5959091 , lodac:Bubo ,
                      <urn:lsid:ubio.org:namebank:2473659>.
```

Since our approach to the identifier is designed for being simple and there is no the best solution for naming identifier, the outcome of this approach is discussed in the section 7.1.

4.3.2. Transition Model

Transition model includes a set of triples where a single triple ⟨ *subject, predicate, object* ⟩ itself is meaningful for learners and many of them can construct a useful knowledge graph. A single triple of this model is also interpreted as a chronological change in taxa. This contain only information of the change between taxa, but any context is eliminated.

For example, the replacement from the species *Nyctea candiaca* to the species *Bubo scandiacus* can be simply written by a single proposition as

replacedTo(Nyctea_scandiaca, Bubo_scandiacus)

or a single triple as

⟨ *species:Nyctea_scandica_1826, ltk:replacedTo, species:Bubo_scandiacus_1999* ⟩

Definition

Let *CON* be the set of contextual nominal entities, IP_{CHRON} be a set of chronological properties, $IEXT_{CHRON}$ be a set of chronological relations, CON_{STATUS} is a status of a taxon; the definition of the transition model (M^{TRANS}) is

$$M^{TRANS} = (CON, IP_{CHRON}, IEXT_{CHRON}, CON_{STATUS})$$

where

$$IEXT_{CHRON} : IP_{CHRON} \rightarrow 2^{CON \times CON}$$

It means that the $IEXT_{CHRON}$ is a function mapping between a contextual nominal entity and a contextual nominal entity.

In order to describe the cardinality of the following chronological properties with concepts before and after changing, two entities named CON_{OLD} and CON_{NEW} are introduced to be sets of old concepts (before changing) and sets of new concepts (after changing) respectively. Both of them are contextual nominal entities, which can be defined by $CON_{OLD}, CON_{NEW} \subset CON$.

Chronological Property (IP_{CHRON})

IP_{CHRON} contains relations indicating chronological changes. In this work, there main relations: *replacedTo, mergedInto, splitedInto* are introduced.

- ***replacedTo*** is a one-to-one relation that maps a taxon before the replacement with the taxon after replacement, so it can be defined by
  $CON_{OLD} \sqsubseteq =1 replacedTo.CON_{NEW}$ and
  $CON_{NEW} \sqsubseteq =1 replacedTo^{-}.CON_{OLD}$.
  For example, if a taxon *x* is replaced by a taxon *y*, this change can be written by *replacedTo(x, y).*

- ***mergedInto*** is a many-to-one relation that maps a taxon before merging with the taxon after merging, so it can be defined by
  $CON_{OLD} \sqsubseteq =1 mergedInto.CON_{NEW}$ and
  $CON_{NEW} \sqsubseteq \geq 2 mergedInto^{-}.CON_{OLD}$.
  For example, if taxa *x* and *y* are merged into a taxon *z*, this change can be written by *mergedInto(x, z)* and *mergedInto(y, z).*

- ***splitInto*** is a one-to-many relation that maps a taxon before splitting with the taxon after splitting, so it can be defined by
  $CON_{OLD} \sqsubseteq \geq 2 splitInto.CON_{NEW}$ and
  $CON_{NEW} \sqsubseteq =1 splitInto^{-}.CON_{OLD}$.
  For example, if a taxon *x* is split into taxa *y* and *z*, this change can be written by *splitInto(x, y)* and *splitInto(x, z).*

In addition to the *mergedInto* and *splitInto*, two more relations: *majorMergedInto* and *majorSplitInto* are introduced to indicate that two taxa before and after the change are very close to each other.

- ***majorMergedInto*** is a one-to-one relation that infers the relation *mergedInto* where the taxon in the first argument is very close to the taxon in the second argument. This relation is defined by
  $majorMergedInto \sqsubseteq mergedInto$,
  $CON_{OLD} \sqsubseteq =1 majorMergedInto.CON_{NEW}$, and
  $CON_{NEW} \sqsubseteq =1 majorMergedInto^{-}.CON_{OLD}$.
  For example, if taxa *x* and *y* are merged into a taxon *z* where *x* is the big part of *z*, this change can be written by
  majorMergedInto(x, z) and *mergedInto(y, z),*
  after that, *mergedInto(x, z) is inferred.*

- ***majorSplitInto*** is a one-to-one relation that infers the relation *splitInto* where the taxon in the second argument is very close to the taxon in the first argument. This relation is defined by
  $majorSplitInto \sqsubseteq splitInto$,
  $CON_{OLD} \sqsubseteq =1 majorSplitInto.CON_{NEW}$, and
  $CON_{NEW} \sqsubseteq =1 majorSplitInto^{-}.CON_{OLD}$.
  For example, if a taxon *x* is split into taxa *y* and *z* where *y* obtains the big part of *x*, this change can be written by *majorSplitInto(x, y)* and *splitInto(x, z)*, after that, *splitInto(x, y)* is inferred.

Status of Taxon (CON_{STATUS})

CON_{STATUS} contains relations indicating the entered date and the expired date of a taxon using relations named *entered* and *expired* respectively. The domain of both relations is *CON* and the range of them is *TP*. For example, if a taxon *x* is entered at a time point *t1* and expired at a time point *t2*, it can be written by *entered(x, t1)* and *expired(x, t2)*.

Generating Transition Model

The transition model is a view of the event-centric model. It is generated by transforming an operation of change in conception into a single triple using Semantic Web rules. For this task, five rules are introduced. The definitions in this part conforms to the predicate logic.

Rule RT1: Taxon Replacement

This is a rule to generate members of $IEXT_{CHRON}(replacedTo)$. If the event-centric model (M^{EVENT}) entails that there exist an operation (*?rp*) of the *TaxonReplacement*, a taxon (*?x*) is an argument of the parameter named *before* of this operation (*?rp*), and a taxon (*?y*) is an argument of the parameter named *after* of this operation (*?rp*); then the transition model (M^{TRANS}) entails that the relation named *replacedInto* consists the order paired of the former taxon (*?x*) and the latter taxon (*?y*).

`TaxonReplacement(?rp)` `∧ before(?rp,?x) ∧ after (?rp,?y)`	*existing in* M^{EVENT}
`→`	
`replacedTo(?x,?y)`	*generated in* M^{TRANS}

Rule RT2: Taxon Merger

This is a rule to generate members of $IEXT_{CHRON}(mergedInto)$. If the event-centric model (M^{EVENT}) entails that there exist an operation (*?mg*) of the *TaxonMerger*, a taxon (*?x*) is an argument of the parameter named *before* of this operation (*?mg*), and a taxon (*?y*) is an argument of the parameter named *after* of this operation (*?mg*); then the transition model (M^{TRANS}) entails this the relation named *mergedInto* consists the order paired of the former taxon (*?x*) and the latter taxon (*?y*).

`TaxonMerger(?mg)` `∧ before(?mg,?x) ∧ after (?mg,?y)`	*existing in* M^{EVENT}
`→`	
`mergedInto(?x,?y)`	*generated in* M^{TRANS}

Rule RT3: Taxon Splitter

This is a rule to generate members of $IEXT_{CHRON}$*(splitInto)*. If the event-centric model (M^{EVENT}) entails that there exist an operation (*?sp*) of the *TaxonSplitter*, a taxon (*?x*) is an argument of the parameter named *before* of this operation (*?sp*), and a taxon (*?y*) is an argument of the parameter named *after* of this operation (*?sp*); then the transition model (M^{TRANS}) entails this the relation named *splitInto* consists the order paired of the former taxon (*?x*) and the latter taxon (*?y*).

`TaxonMerger(?mg)`	
`∧ before(?mg,?x) ∧ after (?mg,?y)`	*existing in* M^{EVENT}
→	
`splitInto(?x,?y)`	*generated in* M^{TRANS}

Rule RT4: Major Taxon Merger

This is a rule to generate members of $IEXT_{CHRON}$*(majorMergedInto)*. If the event-centric model (M^{EVENT}) entails that there exist an operation (*?mg*) of the *TaxonMerger*, a taxon (*?x*) is an argument of the parameter named *majorBefore* of this operation (*?mg*), and a taxon (*?y*) is an argument of the parameter named *after* of this operation (*?mg*); then the transition model (M^{TRANS}) entails this the relation named *majorMergedInto* consists the order paired of the former taxon (*?x*) and the latter taxon (*?y*).

`TaxonMerger(?mg)`	
`∧ majorBefore(?mg,?x) ∧ after (?mg,?y)`	*existing in* M^{EVENT}
→	
`majorMergedInto(?x,?y)`	*generated in* M^{TRANS}

Rule RT5: Major Taxon Splitter

This is a rule to generate members of $IEXT_{CHRON}$*(majorSplitInto)*. If the event-centric model (M^{EVENT}) entails that there exist an operation (*?sp*) of the *TaxonSplitter*, a taxon (*?x*) is an argument of the parameter named *before* of this operation (*?sp*), and a taxon (*?y*) is an argument of the parameter named *majorAfter* of this operation (*?sp*); then the transition model (M^{TRANS}) entails this the relation named *majorSplitInto* consists the order paired of the former taxon (*?x*) and the latter taxon (*?y*).

`TaxonSplitter(?mg)`	
`∧ before(?mg,?x) ∧ majorAfter (?mg,?y)`	*existing in* M^{EVENT}
→	
`majorSplitInto(?x,?y)`	*generated in* M^{TRANS}

Rule RT6: Status of Taxon

This is a rule to generate members of CON_{STATUS}. If the event-centric model (M^{EVENT}) entails that there exist an event (*?ev*), this event (*?ev*) has an interval (*?inv*) that begins at a beginning time point (*?t1*) and ends at an ending time point (*?t2*), this event (*?ev*) ensures an operation (*?oprc*) of the *OPRC*, a taxon (*?x*) is an argument of the parameter named *before* or *majorBefore* of this operation (*?oprc*), and a taxon (*?y*) is an argument of the parameter named *after* or *majorAfter* of this operation (*?oprc*); then the transition model (M^{TRANS}) entails that the relation named *entered* consists the order paired of the former taxon (*?x*) and the beginning time point (*?t1*), and entails that the relation named *expired* consists the order paired of the former taxon

(*?x*) and the ending point (*?t2*) and the order paired of the latter taxon (*?y*) and the beginning time point (*?t1*).

EVENT(?ev) ∧ INV(?inv) ∧ TP(?t1) ∧ TP(?t2) ∧ interval(?ev,?inv) ∧ begins(?inv,?t1) ∧ ends(?inv,?t2) ∧ OPRC(?oprc) ∧ ensures(?ev,?oprc) ∧ before(?oprc,?x) ∧ after(?oprc,?y)	*existing in* M^{EVENT}
→ created(?x,?t1) ∧ expired(?y,?t1) ∧ expired(?x,?t2)	*generated in* M^{TRANS}

It is noted that the examples of these rules are described hereafter and the validity of them is demonstrated by test cases in the section 4.5.1.

4.3.3. Snapshot Model

The purpose of the snapshot model is to present a set of triples that are valid at a specific time point, so this data model includes a set of triples along with intervals. It would be great if a triple can be presented by ⟨ *subject, predicate, object, begin-time-point, end-time-point* ⟩, but it is impossible due to the limitation of a binary relation used by RDF. For this reason, the named graph is considered. Named graph is a technique to have multiple RDF graphs in a single repository and a name of each graph is provided by a URI. For example, a graph*:g1* that includes two triples ⟨ *:s1, :p1, :o1* ⟩ and ⟨ *:s2, :p2, :o2* ⟩ is written by *GRAPH :g1 { :s1 :p1 :o1 . :s2 :p2 :o2}.*

Definition

Let *IR* is a set of resources, *IP* is a set of properties, *INV* is a set of intervals, *IEXT* is used as a set of triples, and *NG* is a set of names of named graphs; the snapshot model (M^{SNAPS}) is defined by

$$M^{SNAPS} = (IR, IP, INV, IEXT, NG)$$

The term *NG* is newly introduced here. It includes relation that maps a name of graph to a set of triple. In order to give a temporal information to a graph, the snapshot model reuses an interval to be a name of a graph. Thus, for example, *NG(g1)* returns a set of triples under a named graph *g1*. In this project, we declare that M^{SNAPS} entails *NG(gi)* if the interval *gi* covers a given time point. It is noted that we prefer to use the term $M^{SNAPS}[gi]$ rather than the term *NG(gi)* in order to emphasize that this is the snapshot model.

Next, example properties for the change in relation between taxa are defined. In this case, there are *higherTaxon*, *subdividedInto*, *combinedInto*, and *synonym*.

- ***higherTaxon*** is a relation that indicates the higher rank of a taxon. For example, if a taxon *x* is a member of a higher taxon *y*, this scenario can be written by *higherTaxon(x, y).*

- ***subdividedInto*** is a relation that indicates the subdivision of a taxon. For example, if a taxon *x* is subdivided into lower taxa *y* and *z*, this scenario can be written by *subdividedInto(x, y)* and *subdividedInto(x, z).*

- ***combinedInto*** is a relation that indicates the combination of taxa. For example, if taxa *x* and *y* are combined into a higher taxon *z*, this scenario can be written by *combinedInto(x, y)* and *combinedInto(x, z)*.

- ***synonym*** is a relation that indicates a synonym between taxa. For example, if taxa *x* and *y* are synonym, this scenario can be written by *synonym(x, y)*.

Generating Snapshot Model

The snapshot model is also a view of the event-centric model by transforming the operation of change in a relation between taxa into a single triple using rules and assigning an interval as a named graph. The expression of the following rules conforms to the predicate logic.

Rule RS1: Change Higher Taxon

This is a rule to create a member of the *IEXT(higherTaxon)*. If the event-centric model (M^{EVENT}) entails that there exist an event (*?ev*), this event (*?ev*) has an interval (*?inv*), this event (*?ev*) ensures an operation (*?reclass*) of the *ChangeHigherTaxon*, a taxon (*?x*) is an argument of the parameter named *child* of this operation (*?reclass*), and a taxon (*?y*) is an argument of the parameter named *parentAfter* of this operation (*?reclass*); then the named graph *(?inv*) of the transition model (M^{TRANS}) entails that the latter taxon (*?y*) is the higher taxon of the former taxon (*?x*).

`EVENT(?ev) ∧ INV(?inv)∧ interval(?ev,?inv)`
`∧ ChangeHigherTaxon(?reclass)`
`∧ ensures(?ev,?reclass)`
`∧ child(?reclass,?x)`
`∧ parentAfter(?reclass,?y)`
	existing in M^{EVENT}

`→` `higherTaxon(?x,?y)`	*generated in a named graph (*`?inv`*) of* M^{SNAP}

Rule RS2: Subdivide a Taxon

This is a rule to create a member of the *IEXT(subdividedInto)*. If the event-centric model (M^{EVENT}) entails that there exist an event (*?ev*), this event (*?ev*) has an interval (*?inv*), this event (*?ev*) ensures an operation (*?subdv*) of the *SubdivideTaxon*, a taxon (*?x*) is an argument of the parameter named *source* of this operation (*?subdv*), and a taxon (*?y*) is an argument of the parameter named *target* of this operation (*?subdv*); then the named graph *(?inv*) of the transition model (M^{TRANS}) entails that the former taxon (*?x*) is subdivided into the latter lower-taxon (*?y*).

`EVENT(?ev) ∧ INV(?inv)∧ interval(?ev,?inv)`
`∧ SubidivideTaxon(?subdv)`
`∧ ensures(?ev,?subdv)`
	existing in M^{EVENT}

`→` `subdividedInto(?x,?y)`	*generated in a named graph (*`?inv`*) of* M^{SNAP}

Rule RS3: Combine Taxa

This is a rule to create a member of the *IEXT(combinedInto)*. If the event-centric model (M^{EVENT}) entails that there exist an event (*?ev*), this event (*?ev*) has an interval (*?inv*), this event (*?ev*) ensures an operation (*?comb*) of the *SubdivideTaxon*, a taxon (*?x*) is an argument of the

parameter named *source* of this operation (*?comb*), and a taxon (*?y*) is an argument of the parameter named *target* of this operation (*?comb*); then the named graph *(?inv*) of the transition model (M^{TRANS}) entails that the former taxon (*?x*) is combined into the latter higher-taxon (*?y*).

`EVENT(?ev) ∧ INV(?inv)∧ interval(?ev,?inv)`
`∧ CombineTaxa(?comb)`
`∧ ensures(?ev,?comb)`
`∧ source(?comb,?x) ∧ target(?comb,?y)`
	existing in M^{EVENT}

`→`	
`combinedInto(?x,?y)`	*generated in a named graph (`?inv`) of* M^{SNAP}

Rule RS4: Linking Synonym

This is a rule to create a member of *IEXT(synoym)*. If the event-centric model (M^{EVENT}) entails that there exist an event (*?ev*), this event (*?ev*) has an interval (*?inv*), this event (*?ev*) ensures an operation (*?syn*) of the *SynonymLink*, a taxon (*?x*) is an argument of the parameter named *source* of this operation (*?syn*), and a taxon (*?y*) is an argument of the parameter named *target* of this operation (*?syn*); then the named graph *(?inv*) of the transition model (M^{TRANS}) entails that the former taxon (*?x*) and the latter taxon (*?y*) are synonym.

`EVENT(?ev) ∧ INV(?inv)∧ interval(?ev,?inv)`
`∧ SynonymLink(?syn)`
`∧ ensures(?ev,?syn)`
`∧ source(?syn,?x) ∧ target(?syn,?y)`
	existing in M^{EVENT}

`→`	
`synonym(?x,?y)`	*generated in a named graph (`?inv`) of* M^{SNAP}

It is also noted that the examples of these rules are described hereafter and the validity of them is demonstrated by test cases in the section 4.5.1 as well.

4.3.4. Meta-Ontology for LTK

In order to materialize the according data models, meta-ontology for LTK is built using RDF statements. In this section, there are vocabularies for LTK and schemas for operations.

Table 4.1: Mapping between formal terms and RDF vocabularies II.

Descriptions	Terms	RDF Vocabularies
Properties		
is replaced to	*replacedTo*	*ltk:replacedTo*
is merged into	*mergedInto*	*ltk:mergedInto*
is major merged into	*majorMergedInto*	*ltk:majorMergedInto*
is split into	*splitInto*	*ltk:splitInto*
is major split into	*majorSplitInto*	*ltk:majorSplitInto*
is entered at	*entered*	*ltk:entered*
is expired at	*expired*	*ltk:expired*
has higher taxon	*higherTaxon*	*ltk:higherTaxon*
is subdivided into	*subdividedInto*	*ltk:subdividedInto*
is combined into	*combinedInto*	*ltk:combinedInto*
has synonym	*synonym*	*ltk:synonym*

RDF Vocabularies for LTK

In this topic, some formal terms and RDF vocabularies are mapped and shown in Table 4.1. It is noted that this table is in addition to Table 3.1 from Chapter 3.

Schema for Operations

It can be seen that an operation of a change presented in the event-centric model corresponds to a property in either the transition model or the snapshot model. Thus, in order to generate the transition model and the snapshot model, some specific rules are required for every operations of change. It means that if a new operation is created, then a new rule is also defined.

To reduce this routine task, we adapt the data models to be more generic by binding any corresponding property with an operation. For example, the operation *ltk:TaxonReplacement* corresponds to the property *ltk:replacedTo*, so these two resources are linked by a property *ltk:linkingProperty*. There are three linking properties used by LTK as follows:

- ***ltk:linkingProperty*** is for linking between a an operation of change in conception and a property indicating a relation between a concept before and after change such as *ltk:mergedInto*.
- ***ltk:majorLink*** has the same purpose as the *ltk:linkingProperty*, but it indicates a very close relation such as *ltk:majorMergedInto*.
- ***ltk:relation*** is for linking between an operation of change in a relation between taxa and a property used by the snapshot model such as *ltk:higherTaxon*.

Next step, all proposed operations are redefined as follows:

Taxon Replacement

```
ltk:TaxonReplacement
     rdfs:subClassOf       cka:ConceptEvolution ;
     ltk:linkingProperty   ltk:replacedTo .
```

Taxon Merger

```
ltk:TaxonMerger
     rdfs:subClassOf       cka:ConceptEvolution ;
     ltk:linkingProperty   ltk:mergedInto ;
     ltk:majorLink         ltk:majorMergedInto .
```

Taxon Splitter

```
ltk:TaxonSplitter
     rdfs:subClassOf       cka:ConceptEvolution ;
     ltk:linkingProperty   ltk:splitInto ;
     ltk:majorLink         ltk:majorSplitInto .
```

Change Higher Taxon

```
ltk:ChangeHigherTaxon
      rdfs:subClassOf  cka:RelationEvolution ;
      ltk:relation     ltk:higherTaxon .
```

Subdivide a Taxon

```
ltk:SubdivideTaxon
      rdfs:subClassOf  cka:RelationEvolution ;
      ltk:relation     ltk:subdividedInto .
```

Combine Taxa

```
ltk:CombineTaxa
      rdfs:subClassOf  cka:RelationEvolution ;
      ltk:relation     ltk:combinedInto .
```

Linking Synonym

```
ltk:LinkingSynonym
      rdfs:subClassOf  cka:RelationEvolution ;
      ltk:relation     ltk:synonym .
```

4.3.5. Working with Semantic Web Rules

The previous section gives a generic representation of all operations of changes, all rules are redefined for the general purpose. From now on, a rule is expressed by the syntax of the Jena's rule [143]. A Jena's rule is written by the following template. In this template, *rule_name* is the name of a rule, a variable begins with the question mark (*?*) such as *?s*, a URI can be defined by a shot-hand form such as *:p1* and *:p2*, any triple (*?s :p1 ?o*) that comes before the arrow symbol -> is an element of a set of conditions, and any triple (*?s :p2 ?o*) that comes after the arrow symbol -> is an element of a set of inferred triples.

```
[rule_name: (?s1 :p1 ?o1) -> (?s1 :p2 ?o1)]
```

Generating Transition Model

There are Jena's rules that cover the previously introduced rules *RT1 – RT7*.

Rule Rx1: Normal Link

This rule generates a normal link between a concept before and after a change. It covers all criteria of the rules *RT1 – RT3*.

```
[Rx1_transition:
  (?OPR  rdfs:subClassOf  cka:ConceptEvolution),
  (?OPR  ltk:linkingProperty  ?predicate),
  (?opr  rdf:type  ?OPR),
  (?opr  ltk:taxonBefore  ?before),
  (?opr  ltk:taxonAfter   ?after)

->(?before  ?predicate  ?after) ]
```

Rule Rx2: Major Link

These rules generate a link indicating the close relation between a concept before and after a change. It covers all criteria of the rules *RT4 – RT5*.

```
[Rx2_major_before:
  (?OPR  rdfs:subClassOf  cka:ConceptEvolution),
  (?OPR  ltk:majorLink  ?predicate),
  (?opr  rdf:type  ?OPR),
  (?opr  ltk:majorTaxonBefore  ?before),
  (?opr  ltk:taxonAfter        ?after)

->(?before  ?predicate  ?after) ]
```

```
[Rx2_major_after:
  (?OPR  rdfs:subClassOf  cka:ConceptEvolution),
  (?OPR  ltk:majorLink  ?predicate),
  (?opr  rdf:type  ?OPR),
  (?opr  ltk:taxonBefore      ?before),
  (?opr  ltk:majorTaxonAfter  ?after)

->(?before  ?predicate  ?after) ]
```

Rule Rx3: Taxon Status

These rules generate a link indicating the entered data and the expired data of a taxon concept. It covers all criteria of the rules *RT6*.

```
[Rx3_begin_of_before:
  (?event tl:interval ?inv),
  (?inv   tl:beginsAtDateTime ?t1),
  (?event cka:ensures  ?opr),
  (?opr   rdf:type  cka:ConceptEvolution),
  (?opr   ltk:taxonBefore  ?before)

->(?before  ltk:entered  ?t1) ]
```

```
[Rx3_end_of_before:
  (?event tl:interval ?inv),
  (?inv   tl:endsAtDateTime ?t2),
  (?event cka:ensures  ?opr),
  (?opr   rdf:type  cka:ConceptEvolution),
  (?opr   ltk:taxonBefore  ?before)

->(?before  ltk:expired  ?t2)]
```

```
[Rx3_end_of_after:
  (?event tl:interval ?inv),
  (?inv   tl:beginsAtDateTime ?t1),
  (?event cka:ensures  ?opr),
  (?opr   rdf:type  cka:ConceptEvolution),
  (?opr   ltk:taxonAfter   ?after)

->(?after   ltk:expired  ?t1) ]
```

Generating Snapshot Model

For generating the snapshot model, we introduce a rule *Rx4* to work instead of the previously introduced rules *RS1 – RS6*.

Rule Rx4: Snapshot Model

```
[Rx4_snapshot:
  (?event tl:interval ?inv),
  (?OPR   rdfs:subClassOf  cka:RelationEvolution),
  (?OPR   ltk:relation  ?predicate),
  (?event cka:ensures   ?opr),
  (?opr   rdf:type      ?OPR),
  (?opr   ltk:subjectTaxon      ?subject),
  (?opr   ltk:objectTaxonAfter  ?object

->(?subject  ?predicate  ?object) ]
```

After that, the inferred triples are assigned to the name graph of the variable *?inv*. The named graph is supported by Sesame [149] and Jena [143].

In case some repositories do not support the named graph, the reification of a triple can be employed, so the rule can be as follows:

```
[Rx4_snapshot_reification:
  (?event tl:interval ?inv),
  (?OPR   rdfs:subClassOf  cka:RelationEvolution),
  (?OPR   ltk:relation  ?predicate),
  (?event cka:ensures   ?opr),
  (?opr   rdf:type      ?OPR),
  (?opr   ltk:subjectTaxon      ?subject),
  (?opr   ltk:objectTaxonAfter  ?object

->(?inv  rdf:type       rdf:Statement),
  (?inv  rdf:subject    ?subject),
  (?inv  rdf:predicate  ?predicate),
  (?inv  rdf:object     ?object)]
```

4.3.6. Working with Simple Scenarios

This section demonstrates examples of generating the transition model and the snapshot model from the event centric model by simple scenarios.

Generating Transition Model

This scenario is one part of what we described in the even-centric model in Fig. 3.2. The case of merging two families *ex:Auidae_1* and *ex:Buidae_1* into the family *ex:Auidae_2* during a time point *t1* and a time point *t2* has been presented in the even-centric model in Section 3.4.5. After executing with the rules *Rx1 – Rx3*, it results in the linked data of taxa comprising of associations among *ex:Auidae_1*, *ex:Buidae_1*, and *ex:Auidae_2* as demonstrated in Fig. 4.1.

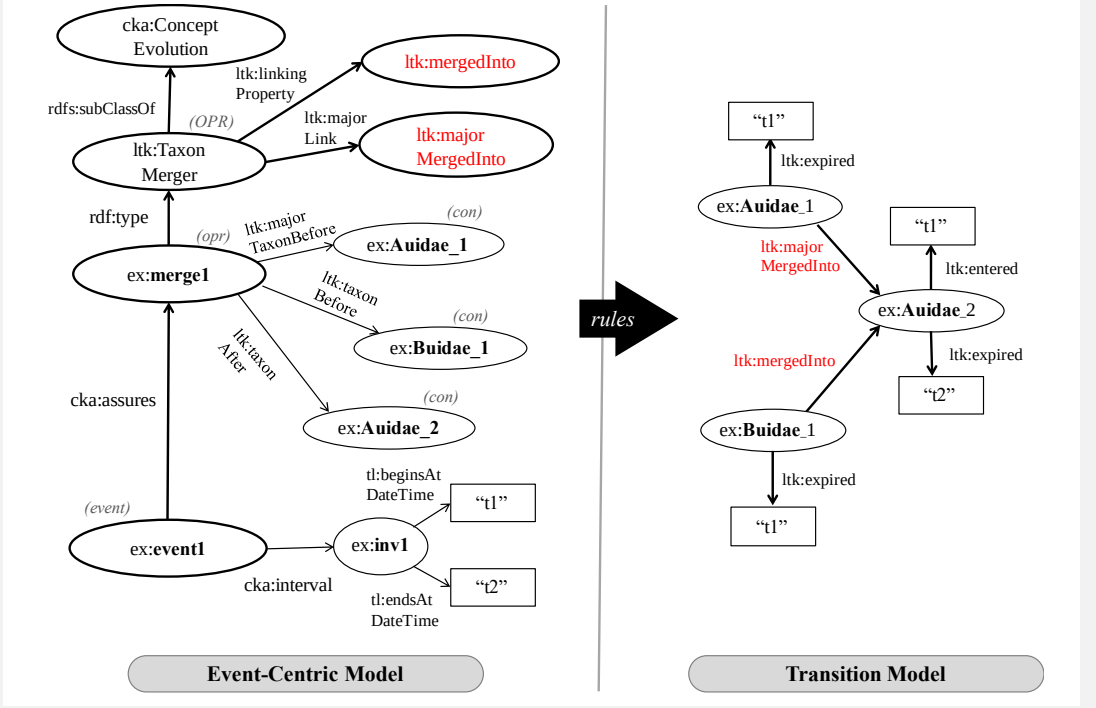


Fig. 4.1: LTK Rule: Transforming an event-centric model into a transition model.

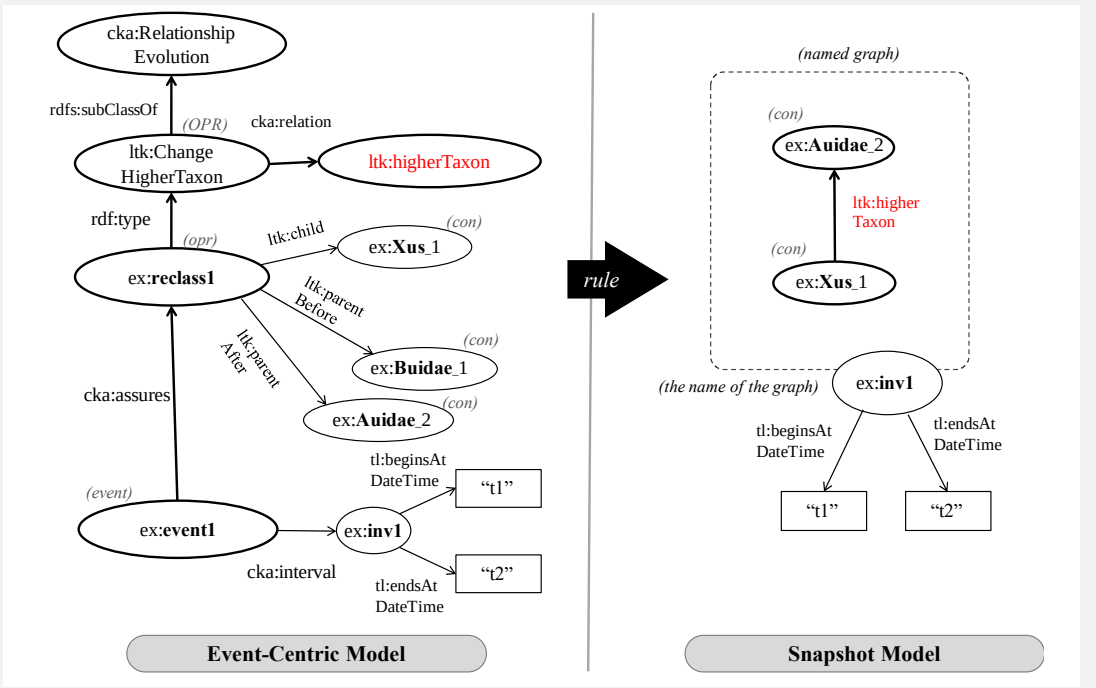


Fig. 4.2: LTK Rule: Transforming an event-centric model into a snapshot model

Generating Snapshot Model

This scenario is one part of what we described in Fig. 3.2 of the even-centric model. The event-centric model presents that the genus *ex:Xus_1* was reclassified from the family *ex:Buidae_1* to the family *ex:Auidae_2* during a time point *t1* and a time point *t2*. After

executing the rule *Rx4*, the snapshot model containing the result triple and *ex:inv1* as the name of named graph is demonstrated in Fig. 4.2.

4.3.7. Semantic Web Rules in Practice

Having proposed the formal descriptions and rules, we now demonstrate how to utilize the RDF model to present and execute the change in taxonomy that is described in the previous sections.

The Revision of the Genus *Columba*

In 2003, the genus *Columba* (pigeons) has been split into five genera, *Patagioenas*, *Chloroenas*, *Lepidoenas*, *Oenoenas*, and *Columba* [7]. According to this change, the following statement give the data of *Columba* in the RDF format. Initially, our work presents the relationship between a species and a genus by using the property *ltk:higherTaxon*.

```
species:Columba_speciosa_1789
     ltk:higherTaxon
          genus:Columba_1758 .
```

Then, the following RDF statements express the event entity and operation for splitting the genus *Columba* together with a reference time point.

```
ex:event2003 cka:interval   [tl:beginsAtDateTime "2003"] ;
             cka:assures    ex:split1 .

ex:split1  rdf:type               ltk:TaxonSplitter ;
           ltk:taxonBefore        genus:Columba_1758 ;
           ltk:majorTaxonAfter    genus:Columba_2003 ;
           ltk:taxonAfter         genus:Patagioenas_2003,
                                  genus:Chloroenas_2003,
                                  genus:Lepidoenas_2003,
                                  genus:Oenoenas_2003 .
```

Furthermore, the framework provides a technique for transforming the event-centric model into the transition model along with a given concept. For example, links between the genus *Columba* and the new concepts after splitting can be expressed as:

```
genus:Columba_1758    ltk:majorSplitInto
                           genus:Columba_2003 ;
                      ltk:splitInto
                           genus:Patagioenas_2003 ,
                           genus:Chloroenas_2003 ,
                           genus:Lepidoenas_2003 ,
                           genus:Oenoenas_2003 .

genus:Columba_1758         ltk:expired  "2003" .
genus:Columba_2003         ltk:entered  "2003" .
genus:Patagioenas_2003     ltk:entered  "2003" .
genus:Chloroenas_2003      ltk:entered  "2003" .
genus:Lepidoenas_2003      ltk:entered  "2003" .
genus:Oenoenas_2003        ltk:entered  "2003" .
```

The Revision of the Genus *Chatopsis*

The genus *Chatopsis* (scorpions) has been reclassified several times. In 2001, Soleglad and Sissom reclassified this genus from the family Chactidae to the family Euscorpiidae, and then in 2011, Rein moved it back to the family Chactidae again. In this case, there are two events.

The first statement is the revision in 2001.

```
ex:event2001 cka:interval        ex:inv1 ;
             bibo:performer      pp:Soleglad, pp:Sissom .

ex:inv1      tl:beginsAtDateTime "2001" ;
             tl:endsAtDateTime   "2011" .

ex:event2001 cka:assures   ex:reclass1 .

ex:reclass1  rdf:type            ltk:ChangeHigherTaxon ;
             ltk:child           genus:Chatopsis_1912 ;
             ltk:parentBefore    family:Euscorpiidae_1896 ;
             ltk:parentAfter     family:Chactidae_1893 .
```

The second statement is the revision in 2011.

```
ex:event2011 cka:interval        ex:inv2 ;
             bibo:performer      pp:Rein .

ex:inv2      tl:beginsAtDateTime "2011" .

ex:event2011 cka:assures   ex:reclass2 .

ex:reclass2  rdf:type            ltk:ChangeHigherTaxon ;
             ltk:child           genus:Chatopsis_1912 ;
             ltk:parentBefore    family:Chactidae_1893 ;
             ltk:parentAfter     family:Euscorpiidae_1896 .
```

After executing with the rule *Rx4*, two named graphs of the snapshot model are generated.

```
ex:inv1 tl:beginsAtDateTime "2001";tl:endsAtDateTime "2011".
ex:inv1 {
             genus:Chatopsis_1912
                 ltk:higherTaxon
                       family:Euscorpiidae_1896 . }


ex:inv2 tl:beginsAtDateTime "2011".
ex:inv2 {
             genus:Chatopsis_1912
                 ltk:higherTaxon
                       family:Chactidae_1893 . }
```

After that, a learner can query information of a particular taxon using a specific time point. For example, the graph of the taxon *genus:Chatopsis_1912* in 2005 can be constructed using the following example SPARQL statement.

```
CONSTRUCT { ?s ?p ?o . }
WHERE {
  ?g tl:beginsAtDateTime ?t1 ; tl:endsAtDateTime ?t2 .
  FILTER(  ?t1 >= "2005-01-01T00:00:00Z"^^xsd:dateTime &&
           ?t2 <  "2005-01-01T00:00:00Z"^^xsd:dateTime )

  GRAPH ?g {
    ?s ?p ?o .
    FILTER ( ?s = genus:Chatopsis_1912 ||
             ?o = genus:Chatopsis_1912)
  }
}
```

A simple RDF statement containing a subject, a predicate, and an object is useful for a client. The data with simple format is easier for exchanging with well-known ontologies in order to query by well-known properties as defined in Appendix. For example, the properties *skos:exactMatch* and *lodac:hasSuperTaxon* in query statements can produce the same results as the ones from *ltk:synonym* and *ltk:higherTaxon*, respectively. This approach also allows users to check the existence of a concept by inquiring about either the property *cka:entered* or the property *cka:expired*.

4.3.8. LTK connecting LOD Cloud

The LTK project publishes different views of taxonomic information in terms of the event centric model, transition model, and snapshot model. The event-centric model is developed on the basis of the Semantic Web and the underlying community knowledge [22, 40], so it can view as a knowledge base that collects the changes in biodiversity knowledge across repositories, and provides background knowledge about how taxon concepts are changed or linked. In order to enable global access on data, This project also maintains links among contextual nominal entities with external nominal entities from known datasets that are commonly referred by many applications and publications such as GBIF [140], CoL [63], uBio [106], and LODAC [146] by using the property *dct:isVersionOf*.

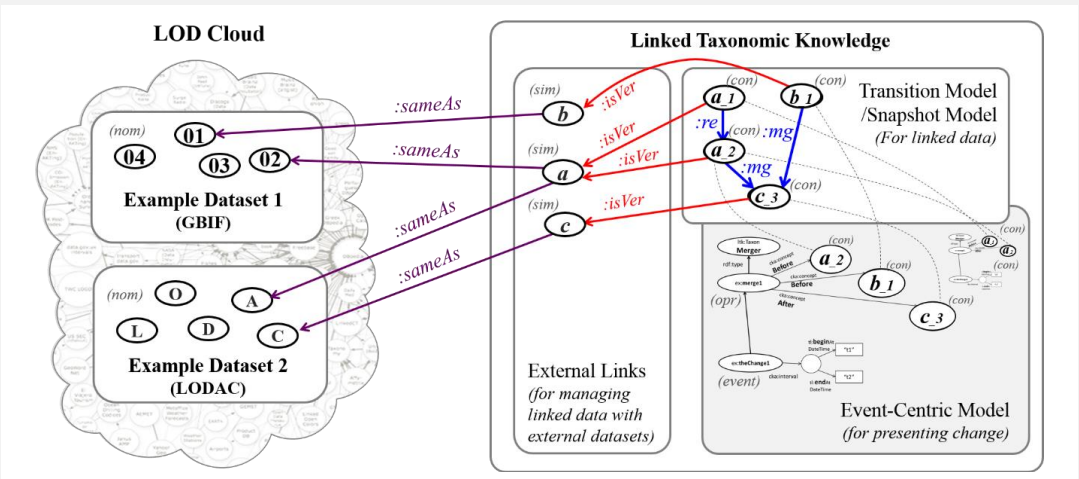


Fig. 4.3: Role of LTK in LOD Cloud

In terms of data exchange, the position of LTK in terms of linked data is demonstrated in Fig. 4.3. In the figure, LTK is positioned to be a portal of linked data collecting the change in

biodiversity knowldge. It contains three parts. The first part consists of external links for representative concepts and links to external datasets. The second part includes the transition model and the snapshot model. The third part contains the event-centric model that acts as the background knowledge of change. Our approach can publish data to the LOD Cloud by using open access data via SPARQL, making URIs be dereferenceable, and linking data to known datasets [50].

Moreover, the properties presented in this chapter are used and extended from some well-known ontologies SKOS [150], LODAC [88], and TaxMeOn [118] as shown in Table 4.2. In this case, the triples under the LTK ontology are globally exchanged with LOD cloud as well.

Table 4.2: Relations between LTK's properties and other ontologies.

Properties	***rdfs:subPropertyOf***
ltk:replacedTo	*tmo:congruentWithTaxon* *skos:exactMatch*
ltk:mergedInto	*skos:broadMatch*
ltk:majorMergedInto	*skos:closeMatch*
ltk:splitInto	*skos:narrowMatch*
ltk:majorSplitInto	*skos:closeMatch*
ltk:higherTaxon	*skos:broaderTransitive* *tmo:isPartOfHigherTaxon* *lodac:hasSuperTaxon*
ltk:subdividedInto	*skos:narrowMatch*
ltk:combinedInto	*skos:broadMatch*
ltk:synonym	*skos:exactMatch* *lodac:hasSynonym*

4.4. Prototype

Our proposed approach to the LTK model described in both Chapters 3 and 4 intends to capture and exchange the change in taxonomic knowledge and represents any changes in RDF format. To verify the possibility and feasibility of our work, a web application was developed. The main purpose of its implementation is to execute and present changes in taxonomic knowledge. The system architecture and a demonstration of this web application are also presented. Information on our prototype is available at the website:

http://rc.lodac.nii.ac.jp/ltk/

4.4.1. Functionalities

The prototype is implemented on the basis of two key functions: defining and executing the change in taxonomic knowledge and presenting the temporal information of an Internet resource used in taxonomic knowledge.

The first function allows users to input changes in taxonomic knowledge by recording a list of operations, their parameters, and metadata. It also offers a bulk load feature for importing the event-centric model in RDF into the system directly. When the input data is submitted, rule-based reasoning produces the relationships between concepts that are the result of a change in taxonomic knowledge, and then, the system collects the RDF data in an RDF data store.

In addition to the execution of the event-centric model, the second function offers an interface for presenting temporal information and linked data of a given concept. The prototype lets users browse the URI of a given concept with a given time point in *xsd:dateTime* format, and it then displays the temporal information of the concept together with its related concepts that are resulted from the change and any background information regarding those changes.

4.4.2. Implementation

To accomplish these key activities, we analyzed the functions, designed the system architecture, employed well-known open source tools, and did the programming to implement the web application for end users and service interfaces for client applications. The architecture of the prototype is a web-based system, as shown in Fig. 4.4, comprising three layers: a presentation layer, business logic layer, and data access layer.

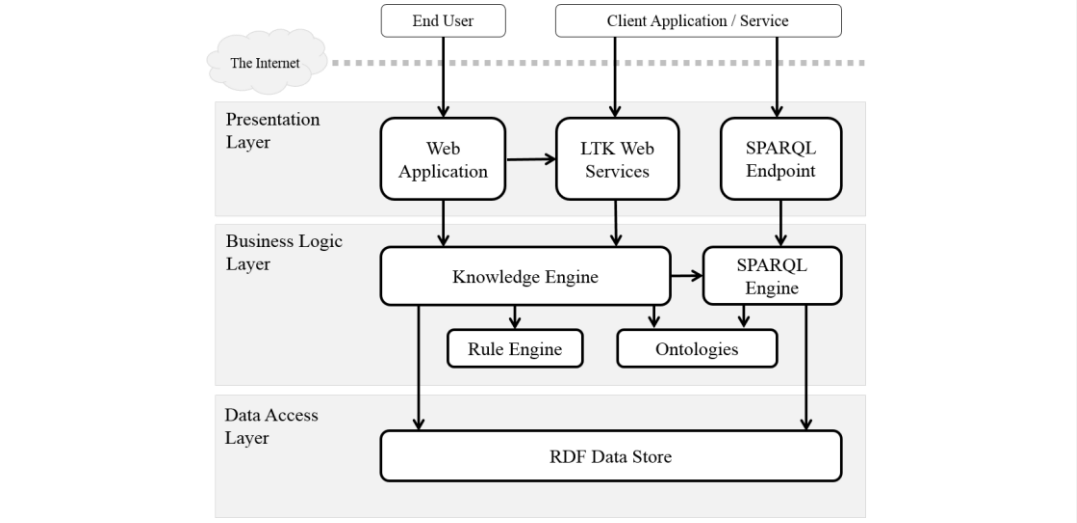


Fig. 4.4: LTK Prototype: System Architecture.

The presentation layer displays information related to such services as creating and executing the change in a given concept and presenting the taxonomic knowledge. It communicates with other service endpoints by outputting results to users or client applications. The user can browse the information by using a web application created by PHP, whereas the client applications can access the data by using LTK web services written in Java and SPARQL endpoint, which is provided by Sesame framework [149].

In addition to the presentation layer, the business logic layer controls an application's functionality by performing data processing. Knowledge Engine, a Java-based component, is the main module that manages the RDF-based event-centric model together with Semantic Web rules and related ontologies in order to construct taxonomic knowledge and linked data of Internet resources for taxonomic data. Technically, this component normalizes and forwards RDF data to the data store directly. It also queries RDF data via the SPARQL engine with an API from Sesame. Moreover, a Semantic Web rule engine developed by using Apache Jena [143] transforms the event-centric model into the transition model and the snapshot model.

Last, the data access layer built for the storage and retrieval of triples collects subject-predicate-objects from components in the upper layers. Our experiment uses Sesame, which offers high capacity with great performance. It additionally offers an API that performs well with Jena.

All of these layers run on a server that is connected to the Internet, so the system is ready to provide LTK services to end users or client applications. Moreover, the system architecture is flexible to enable application to other domains. Developers can customize Semantic Web rules and ontologies to their own requirements and publish their data for open access.

4.4.3. LTK Services

As a result of the services provided in the presentation layer, all interfaces are conveniently accessible over the Internet. In this section, we illustrate how to use services from this prototype by describing web application and web services.

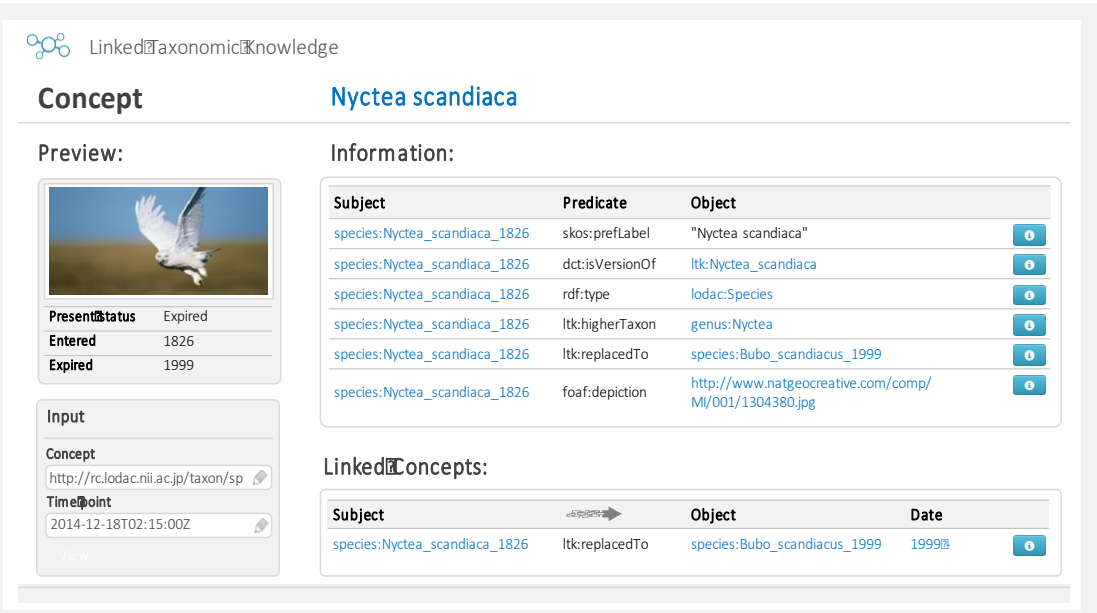


Fig. 4.5: LTK Prototype: Taxonomic Knowledge of a Taxon.

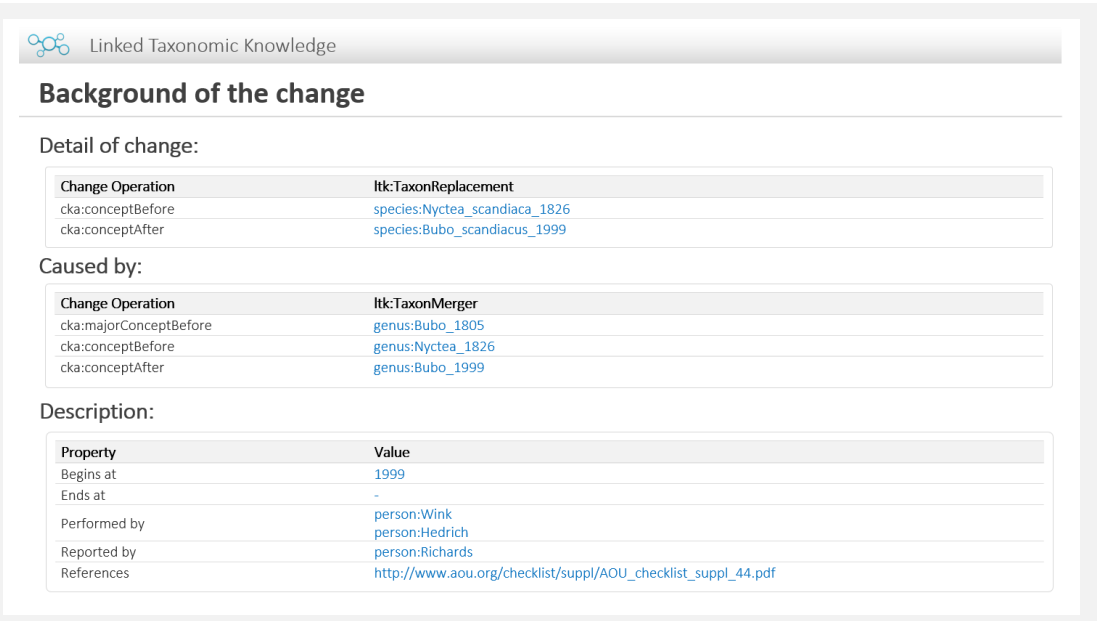


Fig. 4.6: LTK Prototype: Background information about change.

Web Application

Beginning with the web application, it contains two main parts, an administration interface and a user interface.

The administration interface provides a tool for importing a list of changes in concepts. Every change can be done by choosing an operation such as merging, replacing, and splitting, and then assigning a concept or a value to the required properties. After that, users can state the relationship between changes in the case where one change relates to another change by linking them with properties named "cause," "effect," or "detail." Finally, the prototype allows users to prepare metadata of these changes, such as a begin time point, an end time point, performers, e.g., researchers, who discovered the change, reporters who announced the change, and references such as publications.

Apart from the administration interface, the user interface is implemented as a browser for presenting the information of a given concept. The web page shows historical information of a taxon concept including point temporal data, its related concepts that result from the change, and links of its related concepts. The user has to specify a URI of a concept together with a particular time. For this prototype, the URL pattern "*http://[ltk_domain]/*" denotes the domain name of our prototype, where the term "*[ltk_domain]*" is "*rc.lodac.nii.ac.jp*" in our experiment. The pattern of a request for displaying information of a given concept in a given time point is

http://[ltk_domain]/ltk/concept.php?concept=[concept]&date=[time_point] ,

where "*[concept]*" is a URI of a given concept and "*[time_point]*" is a given time point in the format *xsd:dateTime*. For example, browsing the species *Bubo virginianus* at a given time point "*1998-01-01T00:00:00Z*" results in that this species was classified into *genus:Bubo_1805*. After the merging of the two genera, *Bubo* and *Nyctea* in 1999, the species *B. virginianus* was technically reclassified into the newer genus *genus:Bubo_1999*. Thus, a request with time points after 1999 shows that the genus of this species is *genus:Bubo_1999*. In addition, users can request only

http://[ltk-domain]/taxon/[rank]/[name]

in the web browser directly, where "*[rank]*" is a taxonomic rank and "*[name]*" is a taxonomic name string including a version label. The accept request-header, which is "text/html," redirects to a webpage with the current date and time, while sending a request with a header "text/plain" results in retrieving response data as RDF/Turtle format. Another example is indicated in Fig. 4.5, which shows the temporal information of the species *Nyctea scandiaca*. This page includes three main sections. First, a photo of the species is displayed together with its present status, entered date, and expired date. Second, the section "Information" displays temporal data, which can be classification, description, label, etc., that are the snapshot model and the transition model at the given time point. The last section, "Linked Concepts," demonstrates the transition model of the given concept. Moreover, the background knowledge of the change in concepts is described when a button labeled "i" is chosen by a user. A web document titled "Background of the Change" appears and reveals the detail of change, reason behind the change, and metadata. Fig. 4.6 shows the changes in *Nyctea scandiaca* that were caused by the merging of the two genera, *Bubo* and *Nyctea*. It also gives reference information, such as, researchers, academic papers, website, etc., in order to provide evidence for that particular change.

Web Services

In addition to the web application, there are LTK web services and a SPARQL endpoint that provide data to client applications. Example datasets were loaded into Sesame framework

[149] storage via LTK web service. The SPARQL endpoint for querying the links between concepts resulting from the changes can be accessed at the following URL.

http://[ltk_domain]/ltk-service/sparql/ltk

This endpoint also offers the ability to query for the temporal data of a given concept. However, LTK-Service provides a service to present the temporal information of a given concept at a given time point in the RDF/Turtle format by requesting the following URL.

http://[ltk_domain]/ltk-service/context?concept=[concept]&date=[time_point]

The background knowledge of the change that relates to a link of two concepts is available at

http://[ltk_domain]/ltk-service/reason?subj=[subject_concept]&obj=[object_concept] ,

where *"[subject_concept]"* and *"[object _concept]"* are URIs of two associated concepts.

4.5. Evaluation

To keep on the practicability evaluation of the LTK model from the previous chapter, this chapter demonstrates how suitability and feasibility of our proposed models for implementation. The key part of the practicability for implementation has been exhibited using the prototype (Section 4.4). In this section, we test that the event-centric model can be transformed into the transition model and the snapshot model. In this case, the actual result should be lightweight expression and satisfy the expectation from Semantic Web users and domain experts. In addition, we test the memory for storing data and the time for accessing data in order to make sure that our data model does not create any critical system performance issues during development and production.

4.5.1. Evaluation against test cases

For testing the LTK model against the real-world situation, the seven test cases from the evaluation section (Section 3.5) of Chapter 3 are reused to be transformed into the transition model and the snapshot model. The RDF statements presents the results after executing the rules *Rx1 – Rx4* of the all event-centric model in the previous chapter on a case-by-case basis.

Creating a Transition Model

The first three cases are transformed into the transition model.

Case 1: Replacing a taxon

The event-centric model of the *Case 1* in the previous chapter explained that the species *Caligula boisduvalii* has been replaced to the species *Saturnia boisduvalii* since 2008. After executing with the rules *Rx1 – Rx3*, the transition model is created as follows:

```
species:Caligula_boisduvalii_1847
     ltk:replacedTo
          species:Saturnia_boisduvalii_2008 .

species:Caligula_boisduvalii_1847
     ltk:expired  "2008-12-31T00:00:00"^^xsd:dateTime .

species:Saturnia_boisduvalii_2008
     ltk:entered  "2008-12-31T00:00:00"^^xsd:dateTime .
```

Case 2: Merging taxa

The event-centric model of the *Case 2* in the previous chapter explained that the subspecies *Antheraea yamamai yamamai* and the subspecies *A. yamamai ussuriensis* were decided to be merged into the subspecies *A. yamamai yamamai* in 2011. After executing with the rules *Rx1 – Rx3*, the transition model is created as follows:

```
subspecies:Antheraea_yamamai_yamamai_1861
     ltk:majorMergedInto
          subspecies:Antheraea_yamamai_yamamai_2011 .

subspecies:Antheraea_yamamai_ussuriensis_1953
     ltk:mergedInto
          subspecies:Antheraea_yamamai_yamamai_2011 .


subspecies:Antheraea_yamamai_yamamai_1861
     ltk:expired  "2011-12-31T00:00:00"^^xsd:dateTime .

subspecies:Antheraea_yamamai_ussuriensis_1953
     ltk:expired  "2011-12-31T00:00:00"^^xsd:dateTime .

subspecies:Antheraea_yamamai_yamamai_2011
     ltk:entered  "2011-12-31T00:00:00"^^xsd:dateTime .
```

Case 3: Splitting a taxon

The event-centric model of the *Case 3* in the previous chapter explained that the species *Loepa sakaei* has been split into two species *L. sakaei* and *L. katinka* since 2008, where the new *L. sakaei* holds a big part after splitting. After executing with the rules *Rx1 – Rx3*, the transition model is created as follows:

```
species:Loepa_sakaei_1965
     ltk:majorSplitInto
          species:Loepa_sakaei_2008 .

species:Loepa_sakaei_1965
     ltk:splitInto
          species:Loepa_katinka_2008 .

species:Loepa_sakaei_1965
     ltk:expired  "2008-12-31T00:00:00"^^xsd:dateTime .

species:Loepa_sakaei_2008
     ltk:entered  "2008-12-31T00:00:00"^^xsd:dateTime .

species:Loepa_katinka_2008
     ltk:entered  "2008-12-31T00:00:00"^^xsd:dateTime .
```

Creating a Snapshot Model

The last four cases are the conversion of the event centric model into the snapshot model.

Case 4: Changing a higher taxon

The event-centric model of the *Case 4* in the previous chapter explained that the family of the genus *Saxicola* has been changed from the family Turdidae to the family Muscicapidea since 2010. After executing with the rule *R4*, the snapshot model is created as follows:

```
ex:case4_inv  {
      genus:Saxicola_1802
            ltk:higherTaxon
                  family:Muscicapidae_1825.
}

ex:case4_inv
  tl:beginAtDateTime  "2010-10-31T00:00:00"^^xsd:dateTime .
```

Case 5: Dividing a taxon

The event-centric model of the *Case 5* in the previous chapter explained that the species *Attacus atlas* was subdivided into two subspecies *A. atlas atlas* and *A. atlas ryukyuensis* during the time between 2008 and 2011. After executing with the rule *R4*, the snapshot model is created as follows:

```
ex:case5_inv  {
      species:Attacus_atlas_1758
            ltk:subdividedInto
                  subspecies:Attacus_atlas_atlas_2008 .

      species:Attacus_atlas_1758
            ltk:subdividedInto
                  subspecies:Attacus_atlas_ryukyuensis_2008 .
}

ex:case5_inv
  tl:beginAtDateTime  "2008-12-31T00:00:00"^^xsd:dateTime ;
  tl:endAtDateTime    "2011-12-31T00:00:00"^^xsd:dateTime .
```

Case 6: Combining taxa

The event-centric model of the *Case 6* in the previous chapter explained that two subspecies *Saturnia jonasii fallax* and *S. jonasii jonasii* has been combined under the species *S. jonasii* since 2011. After executing with the rule *R4*, the snapshot model is created as follows:

```
ex:case6_inv  {
      subspecies:Saturnia_jonasii_fallax_2008
            ltk:combinedInto
                  species:Saturnia_jonasii_2008 .

      subspecies:Saturnia_jonasii_jonasii_2008
            ltk:combinedInto
                  species:Saturnia_jonasii_2008 .
}
```

```
ex:case6_inv
  tl:beginAtDateTime  "2011-12-31T00:00:00"^^xsd:dateTime .
```

Case 7: Linking synonym

The event-centric model of the *Case 7* in the previous chapter explained that the species *Loepa sakaei* has become a synonym of the species *L. Katinka* since *1965*. After executing with the rule *R4*, the snapshot model is created as follows:

```
ex:case7_inv  {
      species:Loepa_katinka_1848
            ltk:synonym
                  species:Loepa_sakaei_1965.
}

ex:case7_inv
  tl:beginAtDateTime  "1965-12-31T00:00:00"^^xsd:dateTime .
```

All results of the transition model and the snapshot model still preserve the presentation of the change in taxonomy by the simply use of relations between nominal entitles although some pieces of context information are eliminated. These models are easy to read and query, so they satisfy the intention of biodiversity knowledge exchange.

4.5.2. Performance Analysis

In addition to the usability evaluation, the performance of the prototype was tested. There are memory and the query execution time.

Memory

For comparing the memory used by the related models, the number of triples of each scenario is simply counted. Since the most context information of TaxMeOn [118] is not structured data, it cannot use the event centric model and the snapshot model in the comparison. Thus, only the transition model is used to evaluate in this part, and the cases 1-3 are focused.

In case of the transition model of LTK, since the contextual nominal entity presents a taxon and name in a single URI, the number of triple of each scenario is the maximum number between taxon concepts before and after the change. For the TaxMeOn, since the taxon concept and its name are linked by a single property, it firstly requires the same number of triples as the number of taxon concepts. In addition, if a change is just replacing or changing circumscription, it requires only 1 triple; whereas, if it is either merging or splitting, the number of triples is same as the number of concepts involving in the particular change.

Thus, for the *Case 1* (replacing a taxon), the transition model of LTK uses 1 triple, and TaxMeOn uses 3 triples. For both *Case 2* (merging two taxa into one taxon) and *Case 3* (splitting one taxon into two taxa), the transition model uses 2 triples, and TaxMeOn uses 6 triples. The summary of this comparison together with the test cases from domain experts (changes among three checklists: Inoue in 1982 [60], Jinbo in 2008 [62], and Kishida in 2011 [68]) and the 1,000,000 simulated changes are recorded in Table 4.3.

Table 4.3: Memory comparison between LTK and related work.

Cases	**LTK** (Transition Model)	**TaxMeOn**
Replacing one taxon to another one taxon	1 triple	3 triples
Merging two taxa into one taxon or Splitting one taxon into two taxa	2 triples	6 triples
Test cases from domain experts	33 triples	86 triples
Simulating 1,000,000 changes	272.13 MB	760.91 MB

Note: The event-centric model consumes 1,055.06 MB, and the snapshot model consumes 287.50 MB for presenting the 1,000,000 changes.

Query Execution Time

The LTK model essentially transforms a basic triple containing a subject, a predicate, and an object into a complex structure to express an event of a change in either a concept or a triple along with the reference time. As it consumes many more triples than the traditional form to present the same fact, the issue of performance becomes a key point in this research. We therefore verified the model with a great number of data and evaluated the query execution time by comparing our approach and a simple query as a baseline.

According to the data model, one event-centric model including 10 operations requires about 100 triples. In this experiment, the number of test data in the repository was increased up to 1,000,000 triples. For every increase of 100,000 triples, we measured the performance and recorded all the results in a chart. All steps in this experiment were performed on Linux 3.11.0-12 (64 bit) installed on an Intel quad-core i5 3.40-GHz PC with 32 GB of memory. The changes in data were stored in OpenRDF SESAME Ver. 2.7.7. To optimize query performance, RDF schema and direct type hierarchy inferencing were enabled, so sequence triples were automatically generated from ones containing the properties *rdf:type*, *rdfs:subClassOf*, and *rdfs:subPropertyOf*. As a result, the dataset contains more than 5 million triples including inferred statements. The RDF repository additionally built two indexes: a subject-predicate-object-context (spoc) key pattern and a predicate-object-subject-context (posc) key pattern, where a context is generally viewed as a graph name [149].

Our verification step was performed by comparing the result from our approach with the baseline speed. To determine the basic speed of the SPARQL engine in our test, a baseline experiment was conducted by using the following simple SPARQL statement for searching information on a species where the term <*taxon_uri*> is a given URI of a taxon.

```
SELECT ?p ?o
WHERE {
 <taxon_uri> ?p ?o .
}
```

Afterward, we compare the baseline query with an example scenario that a client accesses the same information on the same species by querying from the even-centric model directly using the following SPARQL statements, where the term <*taxon_uri*> is any URI of a taxon and "*tx*" is a specific time point.

```
SELECT ?p ?o
WHERE
{
    {
       ?event   tl:beginsAtDateTime          ?t1 ;
                tl:endsAtDateTime            ?t2 ;
                cka:assures                  ?opr .
       ?opr     rdf:type                     ?OPR ;
                ltk:subjectTaxon             <taxon_uri> ;
                ltk:objectTaxonAfter         ?o .
       ?OPR     ltk:relation                 ?p .

       FILTER(  ?t1 >= "tx"^^xsd:dateTime &&
                ?t2 <  "tx"^^xsd:dateTime  )
   }
 UNION
   {
       ?opr     rdf:type                     ?OPR ;
                ltk:taxonBefore              <taxon_uri> ;
                ltk:taxonAfter               ?o .
       ?OPR     ltk:linkingProperty          ?p .
   }
}
```

The performance was measured by recording the response time of the web method. For having more accuracy, data caching was disabled, and a given concept and a given time point were changed for every service request. The results of the experiment are shown in Fig. 4.7, which shows that the execution time from our approach was almost constant at about 0.039 seconds for every 100,000 input triples added into the repository, while the value from the baseline was approximately 0.016 seconds. A closer look at the result indicates that our approach consumed slightly more execution time than did a simple query by a millisecond unit. The results of our experiment provide confirmatory evidence that our framework does not cause application performance problems in terms of memory and access time for the current software development even if dealing with millions of pieces of changes.

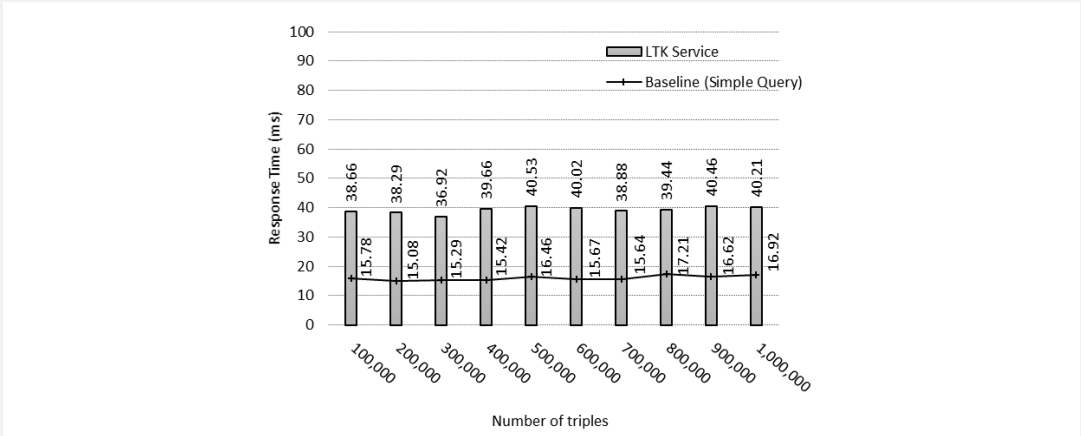


Fig. 4.7: Query execution time in a dataset.

4.6. Summary

This chapter, Biodiversity Knowledge Exchange, described the LTK framework in terms of knowledge exchange. This framework comprises the use of a taxonomic identifier corresponding to a single scientific name, the transition model and the snapshot model. For the purpose of linking data, we developed our model by employing an ontology of contextual knowledge for archives together with widely accepted ontologies such as SKOS. The contextual nominal entity that is a single and readable Internet resource for representing a version of a concept used in taxonomic knowledge is proposed to be viewed as either a name or a taxon concept. The result is that triples become lightweight, simple, and easy to understand by both machines and non-computer-expert users. Our model can deal with both the complex format of the event-centric model and easily-linkable triples from the transition model and snapshot model in RDF, and hence, tipples of both models can be formed a biodiversity knowledge graph. In addition, we implemented a prototype that utilizes the proposed model for managing the change in taxonomic knowledge and offering open access in order to give an opportunity to link our data to the LOD Cloud. As a consequence, other applications that need linked taxon concepts can give associations to these data. By giving links to and reusing existing URIs from well-known taxonomic databases, it is possible to associate our dataset with the large amount of taxonomic data across biodiversity KM systems in order to discover a broader knowledge of biology. The effort of this chapter helps to confirm that it is possible and feasible to use LOD in terms of knowledge graph, schema, reasoning, and query for exchanging the change in biodiversity knowledge.

……

CHAPTER 5

BIODIVERSITY KNOWLEDGE DISCOVERY

"New discoveries in science will continue to create a thousand new frontiers for those who still would adventure."

- Herbert Hoover

When knowledge is captured into a knowledge graph and the schema of the knowledge graph is appropriate for exchanging, the opportunity to create new knowledge from the existing knowledge graph using a computational mechanism is possible to do. For biodiversity knowledge, thanks to the collaboration between Linked Open Data for ACademia (LODAC) and National Museum of Nature and Science (KAHAKU), collecting linked data of interspecies interaction and making link prediction for future observations have started. The initial data is very sparse and disconnected, so it is very difficult of make the prediction of potential missing links when using a single scoring function alone. In this chapter, we introduce Link Prediction on Interspecies Interaction network (LPII) using a hybrid recommendation approach. Our prediction model is a combination of three scoring functions based on three types of graphs, and takes into accounts of different algorithms: collaborative filtering (using bipartite graph), community structure (using projection network), and biological classification (using taxonomy). We have found that our approach to the use of the structure of knowledge graph and the power of LOD is feasible for making prediction on fungus-host interactions and gives higher accuracy than any other scoring functions. Using weight adjustment, we can observe that each scoring function plays different roles depending on the conditions of linked data. This chapter shows that using a knowledge graph and linked data can be applied to deal with other real-world situations of link prediction.

5.1. Overview

Thanks to collaboration between Linked Open Data for ACademia (LODAC) and National Museum of Nature and Science (KAHAKU), we have opportunity to study the role of LOD in the knowledge discovery activity by initiating the project named Link Prediction on Interspecies Interaction (LPII). In this section, background about fungus-host interactions is clearly informed, and the outline of this project is drawn.

5.1.1. Background

Every organism lives on other organisms with some relationship. The range of relationship is vast: predation, parasitism, symbiosis, etc. as shown in Table 2.1. These biological relationships support the biodiversity and evoke evolution. To understand the functional aspects of biodiversity, it is essential to understand biological relationship.

The requirement of the accumulation of relationship information is being claimed [55], and databases are established [142, 158]. The cumulated database will help estimating the keystone species in the interaction [87], help decision making in conservation biology [125], and estimation of epidemic development in biosecurity [89], but the progress is limited [55]. Biological relationships also give insights to understand evolution, such as potential route in horizontal transfer of genes [123].

Fungi are one of the most diverse organisms that requires interaction between other organisms. In contrast to plants that are autotrophic, fungi are heterotrophic (living on other organisms being unable to produce their nutrients by themselves like plants), and require various external nutritional sources [3]. Although some fungi live saprotrophically, a number of fungi are known to be biotrophic, and requires living host [2]. The range of host is also diverse: limited to a single species to a genus or even wider (host selectivity) [38]. The host selectivity is sometimes recognized even in saprotophic fungi. Although most fungi are known to interact between plants through parasitism and symbiosis, some fungi interact with animals and fungi though saprophytism, parasitism, and symbiosis. In fact, fungi are important organisms in nutritional support of plants, pathogens of plants and animals, and prey for insects [90]. Thus, fungi are an attractive source of biological relationships.

Although each fungal-host relationship is given by a single set of fungus-host data, the complexity between multiple fungi and hosts can be better analyzed by incorporating network analysis, because the relationship between fungi and hosts are continuous (one fungus has relationships between plants that may have other relationships with other fungi) [9]. The network analysis provides intuitive and holistic understanding of the complex relationships [43], and even provides estimates for unknown relationship. Although network analysis originated in the social science, it is now being applied to ecological analysis [34, 52, 127].

Among fungi, rust fungi (or simply rusts) are one of the characteristic, parasitic fungi on plants in its lifecycle. Because rusts are absolute parasites which require living hosts, and some of them change the host during the life cycle, producing different kinds of spores. The host range is diverse: from herbal plants to trees, angiosperms to gymnosperms [27]. Because of its complex life cycle, complete understanding of the life cycles of some rusts are lacking. Rusts are also known as a major pathogen of plant resources.

5.1.2. Outline

The LPII project is introduced to be a prediction model for finding potentially missing fungus-host interactions. The interaction data are in a bipartite graph. The prediction based on the bipartite graph using the widely used method such as the collaborative filtering [58] alone gave low accuracy because the dataset is very sparse. Thus, we have to consider the knowledge structure of species that can be the community and the types of them, and then give link prediction based on these characteristics. The community of species is evaluated by reforming the bipartite graph into the projection or the similarity network of species, and then the community structure of the network of species is constructed. After that, we can make a prediction base on the frequent pattern of interactions under the same community. Besides having the groups of species using community detection, the groups of species formed by biologists can be retrieved from the LODAC system [88], which provides RDF data about species' information.

In this project, we consider three main scoring functions: collaborative filtering, community structure, and biological classification. These functions employ the power of knowledge graph and LOD. Then, the combination of these three functions becomes a hybrid recommender system. The roles and the importance of these scoring function in the proposed prediction model is studied. The related work, data analysis, LPII model, and evaluation are described hereafter.

5.2. Related Work

To create a link prediction model for interspecies interaction, some relevant techniques and solutions are studied. There are the background of the recommender system, issues about link prediction in biological domain, and some evaluation methods.

5.2.1. Recommender System

Finding potential fungus-host relationships can be considered as a problem of a user-item recommendation that is studied under the area of a recommender system. In this study, fungi and hosts are viewed as users and items respectively or vice versa. On the basis of how recommendations are made, some categories of recommender systems are described. There are a collaborative, a content-based, a social-based, and hybrid methods [1].

Collaborative Filtering Model

A collaborative recommender system is commonly used by industries such as the book recommendation by Amazon [77]. This method decides to offers a host to a fungus when that host is shared by some similar fungi [58]. There are several indices that evaluate how similar of things such as Common Neighbors [83], Jaccard Index [49], Sørensen Index [117], Hub Depressed Index [83], and Resource Allocation Index [131]. These indices generally estimate that two fungi are very close when they are found at many same hosts and/or a few disjoint hosts. In addition, a link prediction via matrix factorization approach extended latent feature method in the link prediction problem in order to overcome several issues such as data sparsity, imbalance classes that number of unknown links is much greater than existent links, and a large graph [86].

Content-Based Recommender Model

A content-based recommender system adopts some profiles of fungi and/or hosts for recommending appropriate links. For this method, the term frequency in the field of the

information retrieval is employed in order to measure the similarity between fungi based on their features [105]. In addition, it can estimate a possibility to find fungi-host pairs under the same cluster of fungi or hosts using Bayesian classifier [81]. The advantage of this technique is that it is transparency, so the model is more explainable than some collaborative methods, but the accuracy is depended on the quality of feature extraction.

Social-Based Recommender Model

In the study of network clustering, a network structure can be found in the complex network using a community detection method [39]. Most community detection techniques use Modularity [92] to measure the quality of the division of a graph into communities. There are known methods that are Walktrap [98], Fast Greedy [98], Edge Betweeness [91], and InfoMap [104].

- Walktrap employs random walk and estimates a total Modularity in every step before merging clusters [98].
- Fast Greedy uses the random walk technique as same as what Walktrap done but calculated a local Modularity only [98].
- Edge Betweenness is a top-down clustering method where edges are removed in the decreasing order of their edge-betweenness scores [91].
- InfoMap uses an information theoretic clustering on a graph to be a map of random walks representing information flow on a network [104].

In the community detection problem, a cost function is a key player. As we previously informed, a well-known cost function is the Modularity [13] and the modularity function *Q* is defined by

$$Q = \frac{1}{2m} \sum_{ai,aj \in C} \left(w(ai, aj) - \frac{d(ai)\, d(aj)}{2m} \right) \quad 5.1$$

where the node *ai* and the node *aj* trend to stay in the same community, *m* is the half of the number of edges appearing in an undirected graph, the function *w(ai,aj)* is the appropriate weight function, and the signatures *d(aj)* and *d(aj)* return degrees of the node *ai* and the node *aj* in the projection graph respectively.

Hybrid Recommender Model

Last, most recommender systems such as some works from Balabanovicn & Shoham [6], Basu et al. [10], and Schein & Popescul [107] hybridized two or more methods to increase the accuracy of the result. The combinations of hybrid method, which commonly combine the collaborative, content-based, and other methods, are weighted, mixed, and switching recommender systems [1, 48, 19].

- The weighted system uses a linear combination of all prediction scores with different weights to produce a single score. It is simple and straightforward, so it is flexible for plugging with new methods, but each score has to be in the same space.
- The mixed system adopts different recommenders to produce separated results at the same time.
- The switching system switches among recommender methods based on criteria of the dataset.

The mixed and switching systems are useful when the scores are in different space, but it is difficult to produce a single result set. In order to have a proper hybrid model, each scoring function is considered as a feature and it has to be weighted. A simple algorithm, perceptron, is possible to use for finding the weight of each scoring function [79]. A single layer perceptron uses a linear threshold unit with multiple input neurons and one output neuron for showing how features are used in the model. Every weight is adjusted in every learning iteration with a small learning rate until the literation error is less than the specified threshold.

5.2.2. Link Prediction in Biology Domain

In addition, there are applications that employ a link prediction for the biological domain. There are studies of finding the patterns of fungal distribution, for example Wollan & et al. used a linear regression model to study the patterns of fungal distribution based on herbarium information including geographic data, and found that temperature is the main factor of the distribution characteristics of fungi in Norway [124]; and Andersson & et al. studied the patterns of nematode-trapping fungi, and concluded that the pattern of genes of these fungi making impact to the different hosts [4]. They however relied on rich background knowledge such as gene and temperature that is limited in our dataset. The use of a recommender system is found in the food web of animals, for example Berlow & et al. used a simple collaborative recommender method to predict predator-prey interactions in food webs [14]. Moreover, there are researches about the link prediction in medical domains. For example, Deng & et al. employed a Bayesians model to prediction protein-protein interactions of yeasts using a biochemical function, a subcellular location, and a cellular role as features [30]. Lin & et al. introduced the combination of common-neighbor and Bayesian to find the pasterns of a protein function and found that it improved the high false-positive and false-negative rates of protein-protein interaction data [76]. Fakhraei & et al. predicted interactions between drugs and targets by estimating the similarity on each side of drugs and targets based on chemical data, ligand structures, gene expressions, and side effects [37]. Cobanoglu & et al. adopted a matrix factorization technique and Bayes' rules to predict drug-target interactions based on the chemical similarity generated from orders of millions of proteins and compounds [24].

In addition to the use of recommender techniques, some works about the link prediction in biological domain trended to integrate the network analysis with the prediction model. For example, Cheng & et al. analyzed similarity networks of drugs and targets based on molecules and protein targets for predicting drug-target interactions [23]. Emig studied a topology of hierarchical clustering of disease gene expression signatures using random walks in order to discover new drug-target interactions [36]. Li also predicted interactions of drugs and targets based on the graph clustering on drugs and targets [82]. According research demonstrated that the network clustering could be an additional approach to the prediction of missing links [39].

5.2.3. Evaluation Methods

According to the study of the hybrid recommender model, our previous work [20] utilized the combination of the collaborative filtering, the network analysis, and the biological classification. The model mainly placed importance on the characteristic of the fungi's side rather than their hosts. For example, the network clustering and biological classification were utilized for fungi only but not hosts. As a result, the paper demonstrated that the combination of three methods gave higher Area Under the receiver-operating-characteristic Curve (AUC) when comparing to each individual method and each dual combination.

However, the high AUC shows that the overall test data have higher prediction score among predicted pairs. It does not guarantee how many test data found in the top-*k* list [74]. For

example, biologists might not find anything if they picked up the top 100 ranking result, although the AUC is very high. Thus, in order to provide a practical predicted list, a link predication model has to give priority to Precision as well as AUC. More detail about these evaluation methods are described in Section 5.4.2.

5.3. Data Analysis

Linked Open Data for Academia (LODAC) [146] and National Museum of Nature and Science (KAHAKU) [144] have developed structured data of rust fungus-host interactions that is mainly from a literature of a list of fungi recorded in Japan [64]. For example,

```
sp:Blastospora_itoana    lodac:foundAt  sp:Prunus_grayana.
sp:Blastospora_itoana    lodac:foundAt  sp:Prunus_persica.
sp:Blastospora_itoana    lodac:foundAt  sp:Morus_alba.
sp:Puccinia_coronata     lodac:foundAt  sp:Triticum_aestivum.
sp:Puccinia_coronata     lodac:foundAt  sp:Carex_japonica.
sp:Puccinia_acetosae     lodac:foundAt  sp:Rumex_acetosella.
sp:Uredinopsis_filicina  lodac:foundAt  sp:Abies_firma.
sp:Puccinia_acetosae     lodac:foundAt  sp:Rumex_acetosella.
```

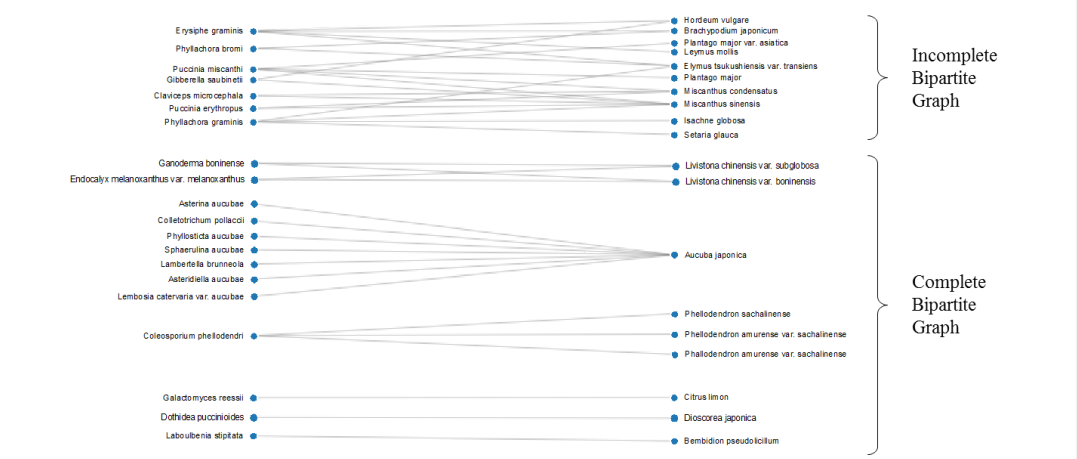


Fig. 5.1: Behavior of the bipartite graph of fungus-host interactions

The dataset is a bipartite graph containing 9,151 interactions between 3,884 fungi and 2,582 hosts, and density being about 0.0009, so it is considered as a very sparse dataset. A closer look at the data indicates that about 70% of nodes and 50% of edges form a lot of small complete bipartite networks that cannot be used as a training dataset. As shown in Fig. 5.1, there are incomplete graphs and complete graphs. The incomplete graphs offer some opportunities to find some missing links, whereas the complete graphs do not allow to find more edges. Moreover, the original dataset contains only fungus-host interactions without any background information of each species such as chemical substances, genetic aspects, and geographical data. Due to the limitation, a prediction model in this research is scoped on only the data from the interaction network.

Owing to the according issue, the dataset has to be cleaned by maintaining only incomplete bipartite graphs that are appropriate for making a prediction. Only species having at least three

degrees are selected in order to have enough data for training and testing during evaluation process. The updated dataset contains 3,846 interactions between 804 fungi and 638 hosts, and the density become 0.007. This dataset is still sparse as shown by an adjacency matrix in Fig. 5.2. In the figure, rows are fungal species, columns are host species including animals and plants, a red dot indicate an existing interaction between a fungus of that row and a host of that column, and white ones are unknown interactions. Henceforth, this research uses this cleaned dataset for doing pattern recognition, hidden features extraction, and prediction.

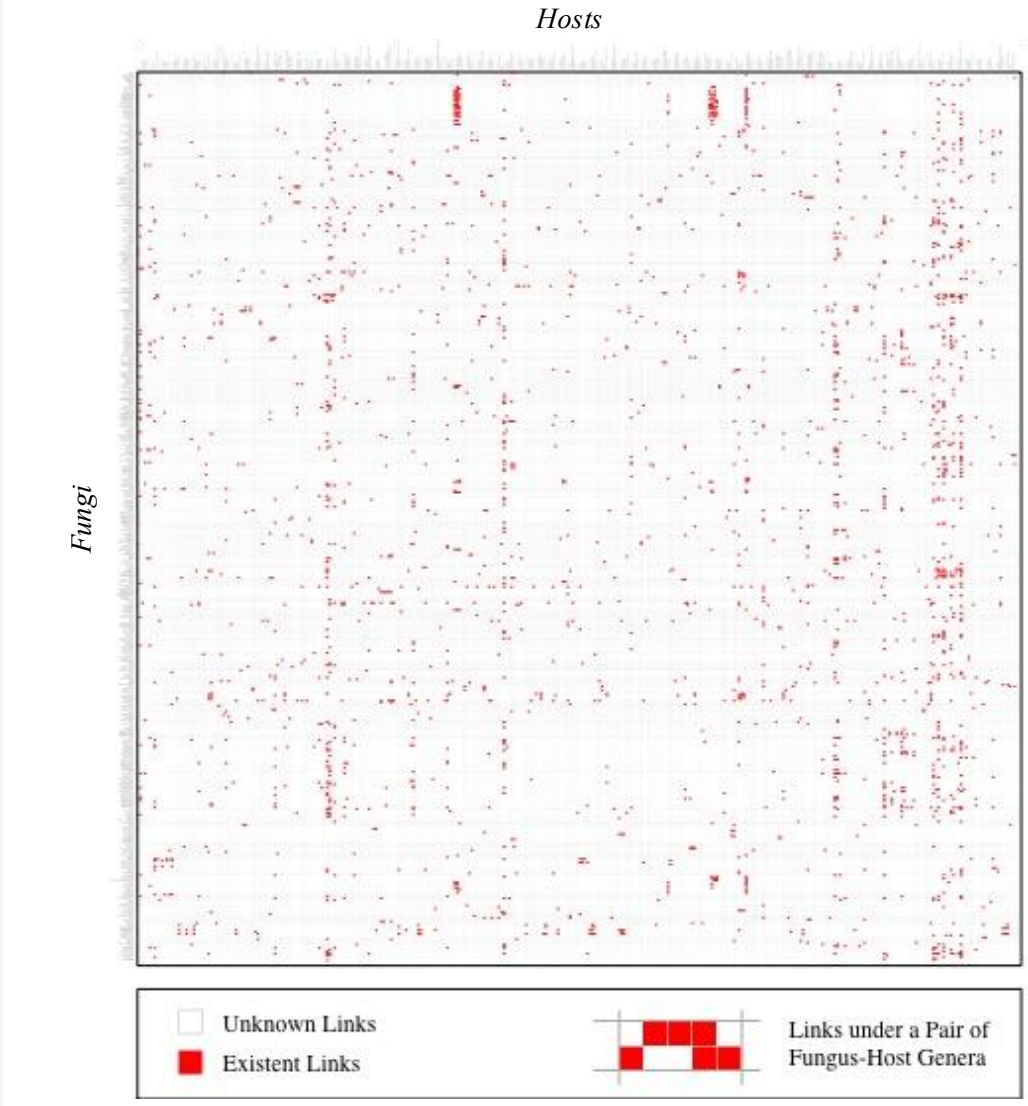


Fig. 5.2: Adjacency Matrix of fungus-host interactions

Based on this dataset, the preliminary experiment for recommending potential fungus-host interactions is done using collaborative filtering methods: the neighborhood similarity [58] with Jaccard index [49] and the matrix factorization [86]. Then, the average Precision for top 100 ranking fungus-host pairs is measured. According to the behavior of this dataset, it found that the neighborhood similarity method [58] and the matrix factorization method [86] gave very low Precisions which are about 0.297 and 0.167 respectively. The result of this case indicated that using the collaborative filtering approach alone is not practical for making recommendation. This issue is commonly found in many pieces of research that mentioned

about the problem of link prediction with a sparse matrix. Thus, other useful metadata should be considered in order to improve the accuracy of the recommender model. As we analyze the adjacency matrix of the bipartite graph as shown in Fig. 5.2, the dataset also has meaningful patterns of the interaction under the same biological classification of both fungi and hosts. Some blocks, which contain interactions under the same genera of both fungi and hosts, are dense; so they should be considered for improving the prediction model.

5.4. Linked Prediction on Interspecies Interaction (LPII)

According to the previous sections, the link prediction approach to collaborative filtering alone does not well address the issue of our dataset. A hybrid approach consisting of the collaborative filtering and other suitable methods is regularly presented by some studies such as the content-based and social-based recommender models. This section describes about some prerequisite definitions, evaluation methods, scoring functions, and the combination of the scoring functions for creating a hybrid model for link prediction on interspecies interaction.

5.4.1. Definition

Before describing the link prediction model, some definitions about graphs, learning processes, and functions used in this chapter are defined as follows:

Elements for Bipartite Graph:

- N^{Fungi} is a set of fungal species.
  Example: $N^{Fungi} = \{ f1, f2, f3, f4, f5 \}$

- N^{Hosts} is a set of host species.
  Example: $N^{Hosts} = \{ h1, h2, h3, h4, h5 \}$

- L^{Exist} is a set of existent links between fungi and hosts.
  Definition: $L^{Exist} \subset N^{Fungi} \times N^{Hosts}$
  Example: $L^{Exist} = \{ (f1,h1), (f2,h1), (f2,h2), (f3,h2), (f3,h3), (f4,h4), (f5,h5) \}$

- $L^{Unknown}$ is a set of unknown or missing links that do not appear in the dataset.
  Definition: $L^{Unknown} = N^{Fungi} \times N^{Hosts} - L^{Exist}$
  Example: $L^{Unknown} = \{(f1,h2), (f1,h3), (f1,h4), (f1,h5), (f2,h3), (f2,h4), (f2,h5), (f3,h1), (f3,h4), (f3,h5), (f4,h1), (f4,h2), (f4,h3), (f4,h5), (f5,h1), (f5,h2), (f5,h3), (f5,h4) \}$

- G^{II} is a bipartite graph of interspecies interactions between fungi and hosts.
  Definition: $G^{II} = (N^{Fungi}, N^{Hosts}, L^{Exist})$

Elements for Projection Graph:

- $\Gamma(n)$ is a function that returns a set of nodes that interact with the node *n*.
  Example: $\Gamma(f2) = \{ h1, h2 \}$

- $E_{\perp}{}^{Fungi}$ is a set of edges of a projection graph of fungi.
  Definition: $E_{\perp}{}^{Fungi} = \{ (fx,fy) \mid fx,fy \in N^{Fungi} \text{ and } \Gamma(fx) \cap \Gamma(fy) \neq \varnothing \}$

where $\varnothing$ is an empty set.

Example: $E_{\perp}^{Fungi} = \{ (f1, f2), (f2, f3) \}$

- $G_{\perp}^{Fungi}$ is a fungus-projection of the bipartite graph G^{II}.

  Definition: $G_{\perp}^{Fungi} = (N^{Fungi}, E_{\perp}^{Fungi})$

- $E_{\perp}^{Hosts}$ is a set of edges of a projection graph of hosts.

  Definition: $E_{\perp}^{Hosts} = \{ (hx,hy) \mid hx,hy \in N^{Hosts} \text{ and } \Gamma(hx) \cap \Gamma(hy) \neq \varnothing \}$

  where $\varnothing$ is an empty set.

  Example: $E_{\perp}^{Hosts} = \{ (h1, h2), (h2, h3) \}$

- $G_{\perp}^{Hosts}$ is a host-projection of the bipartite graph G^{II}.

  Definition: $G_{\perp}^{Hosts} = (N^{Hosts}, E_{\perp}^{Hosts})$

Elements for Learning Process:

- L^{Train} is a set of existent links that is used for learning by a recommender system.
  Definition: $L^{Train} \subset L^{Exist}$

- L^{Test} is a set of existent links that is not trained, and is used for evaluating a prediction model.
  Definition: $L^{Test} = L^{Exist} - L^{Train}$

- $L^{Missing}$ is a set of missing links of which a prediction model assigns a prediction score.
  Definition: $L^{Missing} \subseteq L^{Unknown} \cup L^{Test}$

Functions for Prediction Model:

- $P^{II}(f,h)$ is a prediction function of interspecies interaction that is introduced by this project.
  Input: An ordered pair of a fungus and a host
  Output: A prediction score
  It is noted that $P^{CF}(f,h)$, $P^{CS}(f,h)$, and $P^{CS}(f,h)$ are scoring functions that are introduced hereafter.

5.4.2. Evaluation Methods

Before introducing a recommender model, some evaluation methods are described in order to be the goal for the proposed model. In the link prediction problem, Precision and Area Under the receiver-operating-characteristic Curve (AUC) are recommended to evaluate any link prediction models [83, 74].

Precision

Precision is generally used in any classifier problems by measuring true positive and false positive. In the link prediction problem, Precision in top-*k* ranking is evaluated by ordering $L^{Missing}$ by prediction scores, and then counting how many members of L^{Test} found under the top-*k* links [83].

AUC (Area Under the receiver-operating-characteristic Curve)

AUC is always used to measure the performance of any algorithms by comparing to a random classifier. It considers true positive rate and false positive rate including true positive, false negative, false positive, and true negative. In link prediction, AUC compares rankings of links in $L^{Missing}$ by

$$AUC = \frac{n' + 0.5n''}{n} \qquad 5.2$$

where there are n comparisons among links in L^{Test} and $L^{Unknown}$, n' is times of rankings of links in L^{Test} being higher than $L^{Unknown}$, and n'' is times they have the same score [83]. For example, the prediction result of $L^{Missing}$ is

- $P^{II}(f1,h1) = 0.5$, where $(f1,h1) \in L^{Test}$,
- $P^{II}(f4,h5) = 0.4$, where $(f4,h5) \in L^{Unknown}$,
- $P^{II}(f2,h2) = 0.3$, where $(f2,h2) \in L^{Test}$, and
- $P^{II}(f3,h4) = 0.3$, where $(f3,h4) \in L^{Unknown}$.

According to this list, the comparisons of L^{Test} to $L^{Unknown}$ are 4 times:

- $P^{II}(f1,h1) > P^{II}(f4,h5)$,
- $P^{II}(f1,h1) > P^{II}(f3,h4)$,
- $P^{II}(f2,h2) < P^{II}(f4,h5)$, and
- $P^{II}(f2,h2) = P^{II}(f3,h4)$.

There are 2 times that links in L^{Test} are higher than $L^{Unknown}$ and 1 time equal, so AUC value is

$$AUC = \frac{2\times1 + 1\times0.5}{4} = 0.625$$

AUC evaluates rankings all possible missing links, and the improvement of an algorithm can be done by upgrading most test links to the higher positions, but it does not care how many test links found in the top-*k* ranking. Besides, Precision is more practical for a recommender application; because in practice, an application should recommend some potential links among large number of possible links. Thus, both measurements are considered in this research.

5.4.3. Scoring Function based on Collaborative Filtering (P^{CF})

This scoring function uses the feature of the **Bipartite Graph**. For making link prediction in a bipartite graph, the collaborative filtering method [58], which gives predictive scores for missing fungus-host pairs based on the number of common hosts among fungi or the neighborhood similarity, is used. It transforms one side of a bipartite graph into a classical graph using a similarity index, it finds some close neighbors, and it gives scores to all predicted links. To transform the bipartite graph G^{II} into the projection of fungi $G_{\perp}{}^{Fungi}$, it needs to use one of the following similarity indices to calculate a weight of each edge.

- Common Neighbors (CN) [83]: $|\Gamma(x) \cap \Gamma(y)|$
- Jaccard Index [49]: $\frac{|\Gamma(x)\cap\Gamma(y)|}{|\Gamma(x)\cup\Gamma(y)|}$
- Sørensen index [117]: $\frac{|\Gamma(x)\cap\Gamma(y)|}{|\Gamma(x)|+|\Gamma(y)|}$

- Hub Depressed Index (HDI) [83]: $\frac{|\Gamma(x)\cap\Gamma(y)|}{\max(\Gamma(x),\Gamma(y))}$
- Resource Allocation Index (RA) [131]: $\sum_{Z\in\Gamma(x)\cap\Gamma(y)} \frac{1}{|z|}$

Equation

The idea of the link prediction approach to the collaborative filtering [58] is to sum up all weights between neighbors that are found on with the given host. The weight is calculated by a similarity index. Let *f* be a given fungus, *h* be a given host, f_i be the neighbor of the given *f* under the projection of fungi $G_{\perp}^{Fungi}$ and f_i be found on the given host *h*; the scoring function P^{CF} is defined by

$$P^{CF}(f,h) = \sum_{\substack{f_i \in \text{neighborsOf(f)} \\ \cap\ \text{linksTo(h)}}} w(f,f_i) \qquad 5.3$$

where *neighborsOf(f)* returns a set of neighbors of the node *ai* under the projection $G_{\perp}^{Fungi}$, *linksTo(h)* gives a set of nodes that have direct links to the host *h, and w(f,f_i)* be a weight between *f* and f_i under the projection $G_{\perp}^{Fungi}$.

Example

This example is to evaluate the possibility to find the fungus *Valsa japonica* on the host *Prunus salicina* based on the following dataset.

```
sp:Valsa_japonica     lodac:foundAt   sp:Prunus_mume.
sp:Fomes_torulosus    lodac:foundAt   sp:Prunus_mume.
sp:Fomes_torulosus    lodac:foundAt   sp:Prunus_salicina.
sp:Polyporus_ikadoi   lodac:foundAt   sp:Prunus_mume.
sp:Polyporus_ikadoi   lodac:foundAt   sp:Prunus_salicina.
```

First, the projection of fungi is constructed by using the following SPARQL expression. The property *lpii:neighbor* is introduced for indicating the neighbors of nodes in any projection graph.

```
CONSTRUCT  { ?f1 lpii:neighbor ?f2 . }
WHERE      { ?f1 lodac:foundAt ?h . ?f2 lodac:foundAt ?h .
             FILTER (?f1 <> ?f2) }
```

In this case, the result can be the following graph.

```
sp:Valsa_japonica   lpii:neighbor  sp:Fomes_torulosus .
sp:Valsa_japonica   lpii:neighbor  sp:Polyporus_ikadoi .
sp:Fomes_torulosus  lpii:neighbor  sp:Polyporus_ikadoi .
```

Next, the weight of each pair is calculated. According to the example dataset, the result is as follows

- *w(sp:Valsa_japonica, sp:Fomes_torulosus)* = $\frac{1}{2}$ = 0.5
- *w(sp:Valsa_japonica, sp:Polyporus_ikadoi)* = $\frac{1}{2}$ = 0.5

- *w(sp:Fomes_torulosus, sp:Polyporus_ikadoi)* $= \frac{2}{2} = 1.0$

Finally, the interaction between the *V. japonica* and the *P. salicina* is

$$P^{CS}(sp{:}Valsa\_japonica, sp{:}Prunus\_salicina) = 0.5 + 0.5 = 1.0$$

Since the maximum value of this scoring function is not limited, it needs to be normalized when combining with other prediction models.

5.4.4. Scoring Function based on Community Structure (P^{CS})

This scoring function uses the feature of the **Projection Network**. As we inform in the section of data analysis, the prediction based on the collaborative filtering or neighborhood similarity alone gave a low Precision, so it needs to consider the prediction based on the group similarity. The groups of fungi can be constructed by the nature of data (cluster) or defined by domain experts (biological classification). This section uses the a community detection method such as Walktrap [98], Fast Greedy [98], Edge Betweenness [98], InfoMap [104], to find the community structure or the cluster of fungi.

Equation

The calculation of the scoring function based on a community structure (P^{CS}) is expressed in the equation 5.4. The idea is that the possibility to find a given fungus (*f*) on a given host (*h*) is calculated based on how popular of the given host in the community of the given fungi is. This idea is simply described in the following steps.

1) Detect the cluster of fungi based on the projection of fungi $G_{\perp}^{Fungi}$ using a proper community detection method.
2) Enumerate every fungus (f_i) that is belong to the same cluster as the given fungus (*f*).
3) Count the number of interactions between a fungus (every f_i in the step 2) and the given host (*h*).
4) Divide the result from the step 3 by the number of the fungi from the step 2.

Let *f* be the given fungus, *h* be the given host, *CS(f)* return a set of fungi that are in the same cluster, f_i be any fungus in or member of *CS(f)*; the scoring function P^{CS} is defined by

$$P^{CS}(f,h) = \frac{\sum_{f_i \in CS(f)} 1\{(f_i,h) \in L^{Exist}\}}{\sum_{f_i \in CS(f)} 1} \qquad 5.4$$

where $1\{(f_i,h) \in L^{Exist}\}$ returns the value 1 if there exist the interaction between the fungus f_i on a given host (*h*), otherwise the value is 0.

Example

This example is to evaluate the possibility to find the fungus *Puccinia acetosae* on the host *Prunus grayana*. If the dataset is

```
sp:Blastospora_itoana    lodac:foundAt  sp:Prunus_grayana.
sp:Caeoma_radiatum       lodac:foundAt  sp:Prunus_grayana.
sp:Caeoma_radiatum       lodac:foundAt  sp:Prunus_maximowiczii.
sp:Thekopsora_areolata   lodac:foundAt  sp:Prunus_grayana.
sp:Puccinia_acetosae     lodac:foundAt  sp:Rumex_acetosella.
```

and a community detection method detects that the fungi *B. itoana*, *C. radiatum*, *T. areolate*, and *P. acetosae* are in the same cluster; the possibility to find the fungus *P. acetosae* on the host *P. grayana* is as follows:

$$P^{CS}(sp{:}\,Puccinia\_actosae, sp{:}\,Prunus\_grayana) = \frac{1+1+1+0}{1+1+1+1} = \frac{3}{4} = 0.75$$

5.4.5. Scoring Function based on Biological Classification (P^{BC})

This scoring function uses the feature of the **Taxonomy**. Besides the community detection, the latter perspective is to make a prediction using a group of fungi based on expert knowledge. Some reviews indicated that some groups of fungi are mostly found at some particular plants, for instance, fungi from the genus *Cyttaria* are always found at plants from the genus *Nothofagus*. This fact seems to suggest that grouping based on biological classification is meaningful for finding missing associations between fungi and hosts.

Equation

The equation of the scoring function based on biological classification (P^{BC}) that is expressed in the equation 5.5 is similar to the equation of P^{CS}, but the group of fungi is from the taxonomy defined by taxonomists. The taxonomy is retrieved by linking data with LODAC database [88]. The idea is that the possibility to find a given fungus (*f*) on a given host (*h*) is calculated based on how popular of the given host in the biological classification of the given fungi is. This idea is simply described in the following steps.

1) Retrieve the classification of fungi from the LODAC database or others.
2) Enumerate every fungus (f_i) that are belong to the same classification as the given fungus (*f*). In this case, the genus level is enough because the higher taxonomic ranks contain too many species.
3) Count the number of interactions between a fungus (every f_i in the step 2) and the given host (*h*).
4) Divide the result from the step 3 by the number of fungi from the step 2.

Let *f* be the given fungus, *h* be the given host, *BC(f)* return a set of fungi that are in the same biological classification, f_i be any fungus in or the member of *BC(f)*; the scoring function P^{BC} is defined by

$$P^{BC}(f,h) = \frac{\sum_{f_i \in BC(f)} 1\{(f_i, h) \in L^{Exist}\}}{\sum_{f_i \in BC(f)} 1} \qquad 5.5$$

Example

This example is to evaluate the possibility to find the fungus *Amanita melleiceps* on the host *Drosophila_bizonata*. If the dataset is

```
sp:Amanita_neoovoidea    lodac:foundAt sp:Drosophila_bizonata.
sp:Amanita_neoovoidea    lodac:foundAt sp:Drosophila_angularis.
sp:Amanita_longistriata  lodac:foundAt sp:Drosophila_bizonata.
sp:Amanita_ibotengutake  lodac:foundAt sp:Drosophila_bizonata.
sp:Amanita_melleiceps    lodac:foundAt sp:Muscina_angustifrons.
sp:Amanita_pantherina    lodac:foundAt sp:Megaselia_flava.
```

and the biological classification of the according fungi is

```
sp:Amanita_neoovoidea    lodac:superTaxon     ge:Amanita.
sp:Amanita_longistriata  lodac:superTaxon     ge:Amanita.
sp:Amanita_ibotengutake  lodac:superTaxon     ge:Amanita.
sp:Amanita_melleiceps    lodac:superTaxon     ge:Amanita.
sp:Amanita_pantherina    lodac:superTaxon     ge:Amanita.
```

the possibility to find the fungus *A. melleiceps* on the host *D. bizonata* is as follows:

$$P^{CS}(sp{:}\,Amanita\_melleiceps, sp{:}\,Drosophila\_bizonata) = \frac{1+1+1+0+0}{1+1+1+1+1} = \frac{3}{5} = 0.6$$

5.4.6. The Importance of each Scoring Function

Using the features of a knowledge graph and LOD, three scoring functions: P^{CF}, P^{CS}, and P^{BC} are built. The argument of each function is not only *(f, h)* but also *(h, f)* that uses the projection of hosts ($G_{\perp}^{Hosts}$) for implementing the collaborative filtering and the community detection, so six scoring functions: $P^{CF}(f, h)$, $P^{CS}(f, h)$, $P^{BC}(f, h)$, $P^{CF}(h, f)$, $P^{CS}(h, f)$, and $P^{BC}(h, f)$ are candidates for constructing the hybrid model. In order to choose a proper combination of the scoring functions, the perceptron algorithm [79] is used to find the suitable weight of each function and it shows how importance of every function. Let X be a vector of the six values of all scoring functions, μ be a vector (which is firstly randomized) of six weights of all scoring functions, $\hat{y}$ is the calculated value, and the model is

$$\hat{y} = \mu \cdot X^T \qquad 5.6$$

Next, to train the model is to fine-tune each weight μ_i such that its predictive accuracy is maximum. Each μ_i is adjusted repeatedly using the following equation until the result satisfies the user-specified threshold. Let $\mu_i(t)$ be a weight at time t, y be the expected value (1=found, and 0=unknown), α be the user-specified learning rate ($0 < \alpha \leq 1$), the updated weight is

$$\mu_i(t+1) = \mu_i(t) + \alpha(y - \hat{y})x_i \qquad 5.7$$

The result of this learning is demonstrated in the experiment *E3* in Section 5.5.2. The weights show that the function P^{CF} and P^{CS} are importance for this dataset, while the function P^{BC} becomes a key player when working with nodes having low degree.

5.4.7. Hybrid Recommender System for Link Prediction (P^{II})

To introduce the recommender model, we intend to combine each scoring function on the basis of the following assumptions.

- Including the behavior of data together with the notions of experts.
- Combining different types of graphs (based on linked data).
- Analyzing both sides of fungi and hosts.
- Demonstrating the importance of each graph against the characteristics of data. For example, the collaborative filtering will be weaker when items have small links.

Thus, the proposed link prediction function $P^{II}(f,h)$ is created using the weighed hybrid recommender system. Let P^{CF} be the scoring function based on the collaborative filtering approach, P^{CS} be the scoring function based on the frequent pattern of interactions under the same community structure, P^{BC} be the scoring function based on the frequent pattern of interactions under the same biological classification, f be given fungus, h be a given host, $\mu_?$ be

the weight of each scoring function (in this study, it is adjusted by the perceptron); the link prediction model for interspecies interaction P^{II} is defined by

$$P^{II}(f,h) = \mu_1 \cdot P^{CF}(f,h) + \mu_2 \cdot P^{CS}(f,h) + \mu_3 \cdot P^{BC}(f,h) + \mu_4 \cdot P^{CF}(h,f) + \mu_5 \cdot P^{CS}(h,f) + \mu_6 \cdot P^{BC}(h,f) \quad 5.8$$

5.5. Evaluation

To evaluate the hybrid recommender model for link prediction of interspecies interaction (P^{II}), this section demonstrates experiments, results, result explanation, and observation.

5.5.1. Proposed Experiments

We did some experiments for finding the well combination of the scoring functions for this dataset. There are five experiments. Each experiment is commonly done by cross-validation of the following steps.

1) Split the existent links (L^{Exist}) into the training set (L^{Train}) 90% and the validation set (L^{Test}) 10%
2) Use any method to learn L^{Train} and to generate the list of missing links (L^{Miss}) with predicted scores
3) Evaluate the top-*100* Precision and/or AUC of the L^{Miss} against the L^{Test}

E1: To find out a proper similarity index

In this experiment, we aim to find a proper similarity index from CN [83], Jaccard [49], Sørensen [117], HDI [83], and RA [131] for acting as the function $w(f,f_i)$ that is used for the scoring function $P^{CF}(f,h)$ (equation 5.3). Thus, we conducted the cross-validation experiments for testing the Precision value of the scoring function $P^{CF}(f,h)$ with different similarity indices against the whole dataset.

E2: To find out a proper community detection method

After having a proper similarity index, the projection of fungi $G_{\perp}{}^{Fungi}$ is constructed. Then, the cluster of fungi can be detected using a community detection method. This experiment aims to find a proper community detection method. It can be done by conducting the cross-validation experiments for testing the Precision value of the scoring function $P^{CS}(f,h)$ with every community detection methods: Walktrap [98], Fast Greedy [98], Edge Betweenness [98], InfoMap [104].

E3: To find out the importance of each scoring function

At the moment, we have the scoring function for collaborative filtering P^{CF} with a proper similarity index, the scoring function for community structure P^{CS} with a proper community detection method, and the scoring function for biological classification P^{BC}. Next, we find the importance of all scoring functions for each fungus-host interaction: $P^{CF}(f,h)$, $P^{CS}(f,h)$, and $P^{BC}(f,h)$; and for each host-fungus interaction: $P^{CF}(h,f)$, $P^{CS}(h,f)$, and $P^{BC}(h,f)$. In this case, the cross-validation experiment was conducted using the single-layer perceptron, and then the weight of each scoring function was reported. In addition to working with the whole dataset, we split the dataset into two datasets: the first part contains node degree being greater than the

mean and the second part contains the remaining data. Then, we recorded a proper weight of each scoring function for both split datasets.

E4: To evaluate the prediction model with the whole dataset

After we have the proper weight of each scoring function, the weighted hybrid model for link prediction is formed. In this experiment, we made the cross-validation experiment to find the Precision and AUC values of each scoring function alone, the hybrid model with equal coefficients, and the hybrid model with the weights from the previous experiment. In this case, the scoring function $P^{CF}(f , h)$ is used as a baseline.

E5: To evaluate the prediction model with different node degrees

In addition, we made the similar experiment with the previous experiment, but in this step, we split the dataset to be a dataset having high node degree (greater than mean) and a dataset having low node degree (lower than mean).

5.5.2. Results of the Experiments

The results the experiments *E1 – E5* are reported in the following topics one by one.

E1: A proper similarity index

The result of the first experiment is reported in Table 5.1. Based on this report, the Jaccard index gives the highest score, so the Jaccard index is selected to be used by the next experiment.

Table 5.1: The evaluation of similarity indices.
(Note: The highest value is written in bold.)

Similarity Index	Precision
CN	0.172
Jaccard	**0.297**
Sørensen	0.280
HDI	0.282
RA	0.274

E2: A proper community detection method

Next, we used the Jaccard index to build the projection of fungi ($G_{\perp}^{Fungi}$) and did community detection. The result of this experiment is reported in Table 5.2. Due to the score, we select the Walktrap for the incoming experiment.

Table 5.2: The evaluation of community detection methods
(Note: The highest value is written in bold.)

Community Detection Method	Precision
Walktrap	**0.442**
Fast Greedy	0.429
Edge Betweeness	0.317
InfoMap	0.226

E3: The importance of each scoring function

Based on two previous experiments, the scoring function P^{CF} employs the Jaccard to be a similarity index and the scoring function P^{CS} employs the Walktrap to be a community detection method. This experiment use the perceptron to find proper weights of the six scoring functions:

$P^{CF}(f, h)$, $P^{CS}(f, h)$, $P^{BC}(f, h)$, $P^{CF}(h, f)$, $P^{CS}(h, f)$, and $P^{BC}(h, f)$. Proper weights for all scoring functions categorized by the different conditions of the dataset are reported in Table 5.3. The result reports that the P^{CF} and the P^{CS} are important for the whole dataset and the high fungus degree; whereas for the low fungus degree, the P^{BC} becomes more important than P^{CF}.

Table 5.3: Weights of scoring functions
(Note: The mean of node degree is 3)

Scoring Function (Feature)	Weight Symbol	The Value of Weight		
		Whole Dataset	High Fungus Degree	Low Fungus Degree
Fungi side				
$P^{CF}(f, h)$	μ_1	0.763	0.914	0.177
$P^{CS}(f, h)$	μ_2	0.543	0.586	0.529
$P^{BC}(f, h)$	μ_3	0.101	0.211	0.453
Hosts side				
$P^{CF}(h, f)$	μ_4	0.428	0.604	0.223
$P^{CS}(h, f)$	μ_5	0.584	0.447	0.518
$P^{BC}(h, f)$	μ_6	0.025	0.046	0.454

E4: The result of P^{II} with the whole dataset

This experiment gave comparison among individual scoring functions, a hybrid model, and a weighted hybrid model. Since the experiment *E3* shows that the scoring function P^{BC} is not much important for the whole dataset, the weighted hybrid model without the function P^{BC} was also tested. The values of the Precision and AUC of each combination are reported in Table 5.4. The result can be interpreted that the hybrid model provides more accurate result than the individual scoring function, the weighted hybrid model gives the best result, and the hybrid model without the P^{BC} gives a slightly lower Precision and AUC.

Table 5.4: The evaluation of the whole interaction dataset.
(Note: The highest value of each column is written in bold.)

Scoring Function	Precision	AUC
Whole Dataset		
Individual		
$P^{CF}(f, h)$	0.303	0.855
$P^{CS}(f, h)$	0.448	0.776
$P^{BC}(f, h)$	0.385	0.816
$P^{CF}(h, f)$	0.059	0.760
$P^{CS}(h, f)$	0.342	0.684
$P^{BC}(h, f)$	0.206	0.708
Hybrid		
$P^{CF}(f, h) + P^{CS}(f, h) + P^{BC}(f, h) +$ $P^{CF}(h, f) + P^{CS}(h, f) + P^{BC}(h, f)$	0.537	0.904
Weighted Hybrid		
$0.763 \cdot P^{CF}(f, h) + 0.543 \cdot P^{CS}(f, h) + 0.101 \cdot P^{BC}(f, h) +$ $0.428 \cdot P^{CF}(h, f) + 0.584 \cdot P^{CS}(h, f) + 0.025 \cdot P^{BC}(h, f)$	**0.577**	**0.906**

Scoring Function	Precision	AUC
Weighted Hybrid without P^{BC}		
$0.763 \cdot P^{CF}(f, h) + 0.543 \cdot P^{CS}(f, h) + 0.428 \cdot P^{CF}(h, f) + 0.584 \cdot P^{CS}(h, f)$	0.575	0.869

E5: The result of P^{II} with different node degree

After splitting the dataset into two datasets: one containing high degree nodes and the other one containing low degree nodes. As show in Table 5.5, the Precision and AUC of their weighted hybrid models are better than their hybrid model. In addition, the prediction model of the former dataset without the scoring function P^{BC} and the prediction model of the latter dataset without the scoring function P^{CF} gave a slightly lower Precision and AUC.

Table 5.5: The evaluation of some conditions of the dataset.
(Note: The highest value of each column is written in bold.)

Scoring Function	Precision	AUC
High degree nodes		
Hybrid		
$P^{CF}(f, h) + P^{CS}(f, h) + P^{BC}(f, h) + P^{CF}(h, f) + P^{BC}(h, f) + P^{BC}(h, f)$	0.664	0.869
Weighted Hybrid		
$0.914 \cdot P^{CF}(f, h) + 0.586 \cdot P^{CS}(f, h) + 0.211 \cdot P^{BC}(f, h) + 0.604 \cdot P^{CF}(h, f) + 0.447 \cdot P^{CS}(h, f) + 0.046 \cdot P^{BC}(h, f)$	**0.683**	**0.875**
Weighted Hybrid without P^{BC}		
$0.914 \cdot P^{CF}(f, h) + 0.586 \cdot P^{CS}(f, h) + 0.604 \cdot P^{CF}(h, f) + 0.447 \cdot P^{CS}(h, f)$	0.676	0.861
Low degree nodes		
Hybrid		
$P^{CF}(f, h) + P^{CS}(f, h) + P^{BC}(f, h) + P^{CF}(h, f) + P^{CS}(h, f) + P^{BC}(h, f)$	0.436	0.856
Weighted Hybrid		
$0.177 \cdot P^{CF}(f, h) + 0.529 \cdot P^{CS}(f, h) + 0.453 \cdot P^{BC}(f, h) + 0.223 \cdot P^{CF}(h, f) + 0.518 \cdot P^{CS}(h, f) + 0.454 \cdot P^{BC}(h, f)$	**0.461**	**0.868**
Weighted Hybrid without P^{CF}		
$0.529 \cdot P^{CS}(f, h) + 0.453 \cdot P^{BC}(f, h) + 0.518 \cdot P^{CS}(h, f) + 0.454 \cdot P^{BC}(h, f)$	0.455	0.802

5.5.3. Explanation of the Results of Experiments

According to these experiments, for the whole dataset, the link prediction model using the neighborhood similarity (P^{CF}) and the frequent pattern under the same cluster (P^{CS}) are necessary. However, the frequent pattern under the same biological classification (P^{BC}) is not much important for the whole dataset. Thus, the scoring function P^{BC} is possible to be excluded from the prediction model P^{II} because it does not much impact the overall result. Thus, the experiment *E4* helps to confirm that the scoring functions P^{CF} and P^{CS} are key players for the link prediction model (P^{II}) but the scoring function P^{BC} is not necessary for the whole dataset.

In addition, we found much more interesting details about the characteristic of each scoring function from the experiment *E5*. If the fungi have high node degree, only the features of the neighborhood similarity (P^{CF}) and the frequent pattern under the same cluster (P^{CS}) are proper for making prediction because the dataset with a lot of high degree nodes is dense enough. On the other hands, when dealing with any fungi having low node degree, the scoring function P^{CF} does not work well because the data are not enough to construct a suitable pattern for making prediction. In this case, the prediction based on the frequent pattern under the same biological classification (P^{BC}) becomes a key player together with the frequent pattern under the same community (P^{CS}).

Thus, in the beginning phase of collecting interspecies interaction, when the dataset is not dense, making prediction using the scoring function P^{CS} and P^{BC} is recommended. However, when there are much more links and the dataset is dense enough, the scoring function P^{BC} becomes less important and the scoring function P^{CF} is suggested to be used.

5.5.4. Observation

In addition to the experiments against the existing test set, the observation from outside laboratory environment is done against the list of fungus-host interactions that KAHAKU [144] has updated in May 2015. Some unknown interactions with high scores and some top-*10* interactions under the same fungus are confirmed by the new discovery of experts and some external datasets. Table 5.6 shows that about eleven interactions are found from the new list of KAHAKU (35 new interactions are updated in February 2016), and about nine interactions are confirmed by other literatures in the Internets.

Table 5.6: The newly found fungus-host interactions.
(Note: This list is collected from the new list of KAHAKU and external literatures.
Rank is the position of an interaction under the same fungus.)

Fungi	Animals (A) or Plants (P)	P^{II} Score	Rank
From the new discovery of KAHAKU			
Amanita kotohiraensis	(A) *Megaselia gotoi*	0.826	2nd
Amanita kotohiraensis	(A) *Megaselia flava*	0.823	4th
Amanita kotohiraensis	(A) *Lonchaea sylvatica*	0.774	9th
Amanita pantherina	(A) *Megaselia gotoi*	0.825	3rd
Amanita pantherina	(A) *Megaselia salteri*	0.781	8th
Amanita pseudoporphyria	(A) *Megaselia salteri*	0.820	1st
Russula alboareolata	(A) *Drosophila brachynephros*	0.822	1st
Russula alboareolata	(A) *Drosophila bizonata*	0.808	4th
Russula alboareolata	(A) *Drosophila angularis*	0.801	5th
Tylopilus ballouii	(A) *Megaselia flava*	0.798	2nd
Tylopilus ballouii	(A) *Tricimba japonica*	0.789	3rd
From external literatures available in the Internet			
Chrysomyxa abietis	(P) *Picea abies*	0.611	2nd
Chrysomyxa abietis	(P) *Picea jezoensis var. jezoensis*	0.321	3rd
Coleosporium asterum	(P) *Pinus nigra*	0.312	8th
Coleosporium asterum	(P) *Pinus thunbergii*	0.306	9th
Cronartium flaccidum	(P) *Pinus koraiensis*	0.272	4th
Gymnosporangium asiaticum	(P) *Pyrus communis*	0.430	3rd
Puccinia infra-aequatorialis	(P) *Cirsium kamtschaticum*	0.572	1st
Puccinia kusanoi	(P) *Sasa senanensis*	0.744	1st
Pucciniastrum tiliae	(P) *Abies sachalinensis*	0.423	4th

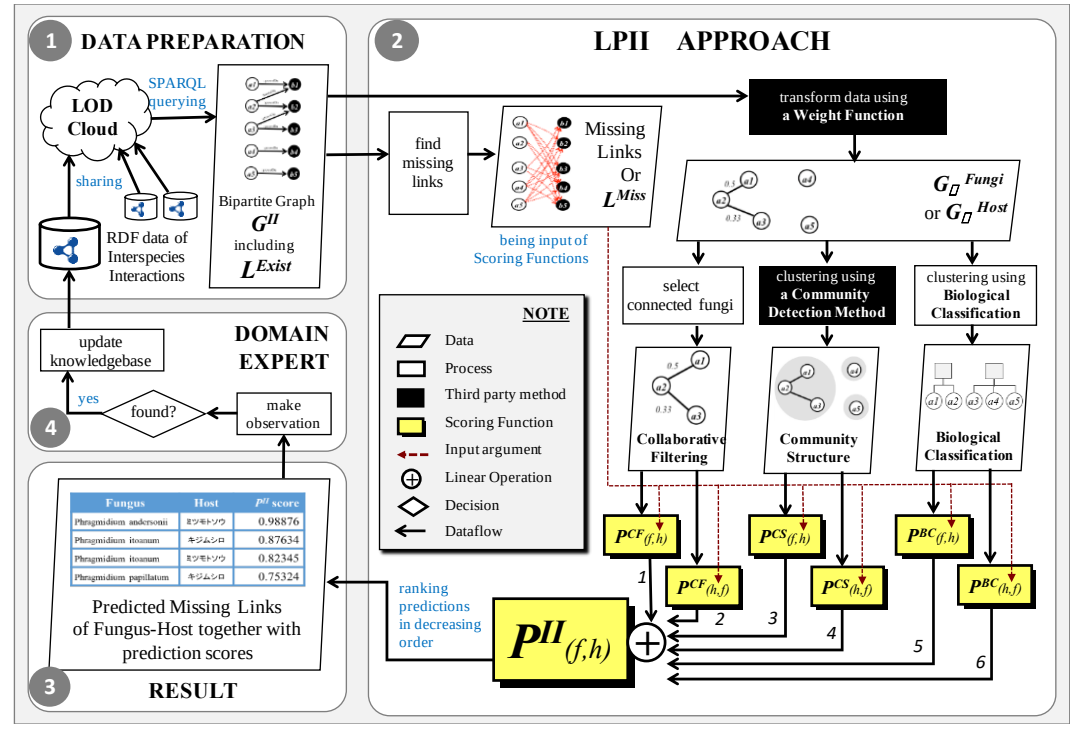


Fig. 5.3: Workflow diagram describing the summary of LPII approach
(1) Data preparation
(2) LPII approach
(3) Predicted result
(4) activity of domain experts

5.6. Summary

The LPII project is an attempt to address the issue of link prediction for the sparse network of linked data, so the hybrid recommendation approach to link prediction that uses the feature of knowledge graph and LOD is introduced. The research can be summarized into four steps as demonstrated in Fig. 5.3.

Step 1: A bipartite graph of fungus-host associations is generated from linked data of interspecies interaction using SPARQL query.

Step 2: This bipartite graph is executed by our prediction model that combines three scoring functions based on different perspectives: collaborative filtering (P^{CF}), community structure (P^{CS}), and biological classification (P^{BC}). The scoring function P^{CF} is evaluated based on the neighborhood similarity of species. The scoring function P^{CS} is evaluated based on the frequent interaction pattern of species under the same cluster created by the community detection method on the similarity network. The scoring function P^{CS} is evaluated based on the frequent interaction pattern of species under the same biological classification retrieved from any taxonomic database. The link prediction model P^{II} was evaluated by Precision and AUC. It has been found that the combination of the scoring functions P^{CF} and P^{CS} are optimal for this dataset. However, the P^{BC} becomes a key player when dealing with a dataset having low node degree.

Step 3: After the calculation of the project LPII, the list of missing links together with predictive scores is provided.

Step 4: Biologists make an observation over the prediction result of fungi and hosts. An observed result will be preserved in an RDF repository and published to LOD cloud in order to be a knowledge base for a prediction in the next time.

The effort of this chapter helps to confirm that it is possible and feasible to use LOD in terms of the proper structure of a knowledge graph and query for discovering more biodiversity knowledge.

.....

CHAPTER **6**

BIODIVERSITY KNOWLEDGE PRESENTATION

"One picture is worth ten thousand words."

- Chinese proverbs

The last KM activity studied by this thesis is the Knowledge Presentation. It is known that the proper format of a knowledge graph can be understood by machines and can be used into various applications, but it is not suitable for reading by normal users due to technical skills related to Semantic Web required. In biodiversity domain, most users are not expert in Semantic Web and the RDF expression itself becomes unfriendly for users, so it becomes a big barrier between domain knowledge and LOD. Fortunately, we learned that a concept-map or a node-link diagram can enhance the learning ability of learners from beginner to advanced user level especially in the biology domain, so an RDF graph visualization can be a suitable tool for making users be familiar with any knowledge graph. However, an RDF graph retrieved from a query result is not proper for reading, because the graph is highly connected like a hairball and less organized. Therefore, this chapter describes how to create a nice-looking RDF graph visualization using the combination of three main functions: graph simplification, triple ranking, and property selection. These functions are mostly initiated based on the knowledge structure under RDF data as knowledge units together with statistical analysis in order to deliver an easily-readable graph to users.

6.1. Overview

Due to the power of Semantic Web and LOD, enabling a big knowledge graph in the global knowledge space is possible to do [16, 50, 116]. Many pieces of Semantic Web research were commonly working with RDF at the data tier especially in the improvement of searching ability, because the advantages of knowledge representation and knowledge reasoning can construct rich machine-readable data in the form of a knowledge graph [54]. A large amount of connected data is required; however, RDF data are mostly provided by tech users [28] (ones who know Semantic Web well). In contrast, encouraging lay users [28] (ones who have less knowledge about Semantic Web) to contribute RDF data is very challenging, because they never realize how linked data work and RDF syntax itself is not user-friendly [15, 129]. It is resulted in a big barrier between human and linked data.

In addition to the knowledge graph in terms of machine readability, the knowledge graph should be learned by learners in a proper way. For this reason, RDF data should be located not only at the data tier but also at the presentation tier in order to have users be familiar with Semantic Web. In this case, we question, "How users can access linked data in a suitable way?" Since a concept-map or a node-link diagram can enhance the learning ability from beginner to professional level, the RDF graph visualization becomes an appropriate way for enabling users to learn knowledge described in the RDF format and making them appreciate the role of LOD in KM systems [35, 80, 110].

However, transforming RDF data into an easily-readable graph visualization is challenging due to a lot of issues caused by the behaviors of RDF data together with the reasoning results. As we analyzed the data closely, we have found that there are three significant issues.

- A visualized RDF graph is too complex to read by learners because a lot of inferred triples generate the graph to be highly connected like a hairball.
- There is lacking of the flow of reading in a graph because there is no ordering to triples in any RDF graph. Consequently, learners are not convenient to find which parts of the graph that they should be focused at first and hereafter.
- Learners need to find some links related to their interest among a large number of data presented.

This research aims to offer an approach to the presentation of RDF graph visualization as a learning tool by interpreting RDF data as knowledge structures. The following features are initiated to address the mentioned problems.

- **Graph Simplification**: To simplify a graph by removing some redundant triples that are resulted from ontological reasoning processes.
- **Triple Ranking**: To give a ranking score to each triple from common information (background content) to topic-specific information (main content), and to allow users to filter a graph based on this score.
- **Property Selection**: To allow users to filter a graph by selecting some properties in order to display or hide some triples.
- **User Interaction**: To control the above operations according to user demand.

6.2. Related Work

For the topic about network visualization, there are pieces of research that worked on this issue, and they aimed to operate a complex network in any visualization canvas to be friendly for general users.

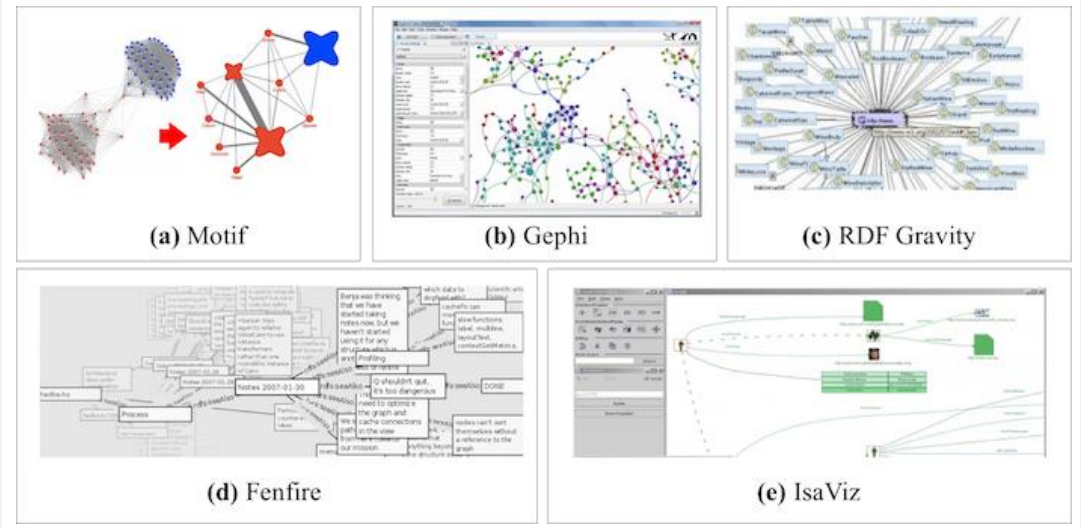


Fig. 6.1: Network Visualization Tools
(a) Gephi Open Viz Platform, (b) Motif Simplification,
(c) RDF Gravity, (d) Fenfire, and (e) IsaViz

We first reviewed some network visualization tools. Motif Simplification [33] considered some topologies of subgraphs, and replaced them with basic shapes such as diamonds, crescents, and tapered diamonds. It intended to give a big picture of a network rather than the detail of node-link as shown in Fig. 6.1(a). Gephi Open Viz Platform [85], which is shown in Fig. 6.1(b), is a powerful visualization tool that generated a well-shaped layout of network, allowed users to filter nodes and links, and had an option to set colors according to user preference. Both tools are suitable for general networks, but they are not designed for dealing with RDF data.

One important issue of RDF data is a large number of inferred links creating a hairball-like graph, so visualization tools should consider some proper ways to simplify a highly-dense graph into a sparse one. RDF Gravity [46] is an RDF visualization tool that provides an interactive view as shown in Fig. 6.1(c). Users can zoom in or zoom out a graph to see much more details, and they can read the information of some nodes in the focused area using text overlay. Next, Fenfire [119] is a good visualization tool that gives an alternative view of an RDF graph as shown in Fig. 6.1(d). It displays the full details of the focused node and its immediate neighbors, but the other links are faded away by considering the distance from the focus node. As we reviewed, both RDF Gravity and Fenfire offers well-organized displays, but they do not point out the how to sparsify a complex graph by concerning the issue of inferred triples. Moreover, IsaViz [99] is an interactive RDF graph browser that uses graph style sheets to draw a graph as shown in Fig. 6.1(e). The advantage is that it provides meaningful icons describing the type of each node such as *foaf:Person*, and groups its metadata into a table in order to reduce highly interlinked data. It also allows users to filter some nodes or properties for the specific purpose of users and simplify a graph, but this task requires much more human effort to select or deselect some preferred URIs one by one.

In addition to the issue about the complexity of a graph, the other issue is about the ability to read RDF data, because RDF data are not well arranged for reading from the introduction part to the main part. Some works target to rank query triples. Several approaches used Term Frequency- Inverse Document Frequency (TF-IDF) to extract keywords from a content [71, 73]. PageRank [18] gave a score to each page by calculating the number of links with the quality of neighbors. TripleRank [42] ranked query result by applying the decomposition of a three-dimensional tensor that is originated by HITS [69] in order to find some relevant resources and predicates. Ichinose [59] employed the idea of TF-IDF to identify how important of resources and predicates of each subject under the same classification for ranking the query result. Nevertheless, they did not discuss about how to order triples for supporting the readability of users by giving separate views of graph from the common information to the main content.

6.3. Data Analysis

This research views that, besides storing RDF data at the data tier, the RDF data should be presented at the visualization tier in order to have users to realize how importance of linked data in KM systems. Using graph visualization for presenting knowledge is a suitable way for users to read and understand Semantic Web data [80, 110].

The well-displayed graph visualization should be simple and sparse [94]. In other words, it should be similar to the original RDF data because they came from the original intention of data providers; however, a query result contains both raw RDF data and inferred data due to of the manner of a SPARQL engine. In this case, querying a graph by accessing the whole neighborhood of a given node within two hops is recommended to be a general input for this research due to the following scenario. If raw RDF data are

```
:Dog      skos:broaderTransitive  :Mammal .
:Mammal   skos:broaderTransitive  :Animal .
:Animal   skos:broaderTransitive  :LivingThing .
```

The inferred triples can be

```
:Dog      skos:broaderTransitive  :Animal .
:Dog      skos:broaderTransitive  :LivingThing .
:Mammal   skos:broaderTransitive  :LivingThing .
```

Then, the query graph becomes

```
:Dog      skos:broaderTransitive  :Mammal .
:Mammal   skos:broaderTransitive  :Animal .
:Animal   skos:broaderTransitive  :LivingThing .
:Dog      skos:broaderTransitive  :Animal .
:Dog      skos:broaderTransitive  :LivingThing .
:Mammal   skos:broaderTransitive  :LivingThing .
```

The well-displayed graph should be similar to Fig. 6.2(a) because it is easy to read and understand by humans, but in practice, it is hardly possible to obtain this kind of the result directly due to the reasoning mechanism of an RDF repository and a SPARQL service. Querying the information of the given node within one hop does not provide enough triples for constructing an informative structure of a graph, because some original triples are missing as

shown in Fig. 6.2(b). In contrast, querying within two hops can maintain the mostly complete structure of the raw data as shown in Fig. 6.2(c), so it has an opportunity to be transformed into a simple graph by removing some inferred triples out of the query graph. The following expression can be used as a guideline to query the whole neighborhood of a given node *(uri)* within two hops.

```
CONSTRUCT { ?s ?p ?o.  ?o ?p1 ?o1. }
WHERE     { ?s ?p ?o.  ?o ?p1 ?o1.   FILTER(?s = <uri>) }.
```

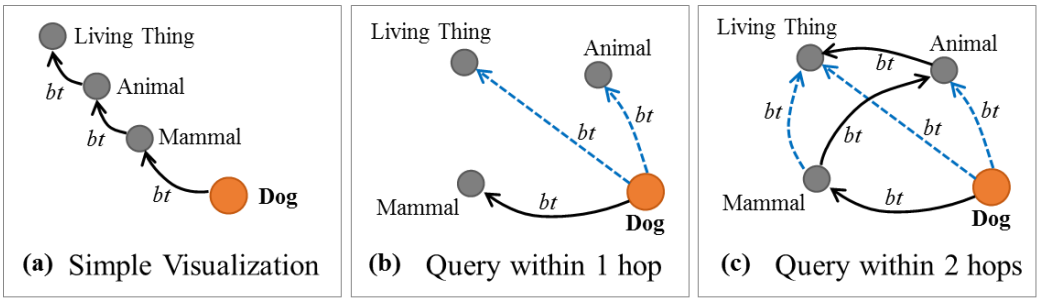


Fig. 6.2: Example query result of the given term.
(Note: *"bt"* denotes *skos:broaderTransitive*, a black solid line indicates an original triple, a blue dashed line specifies an inferred triple, and a big yellow node represents the given node.)

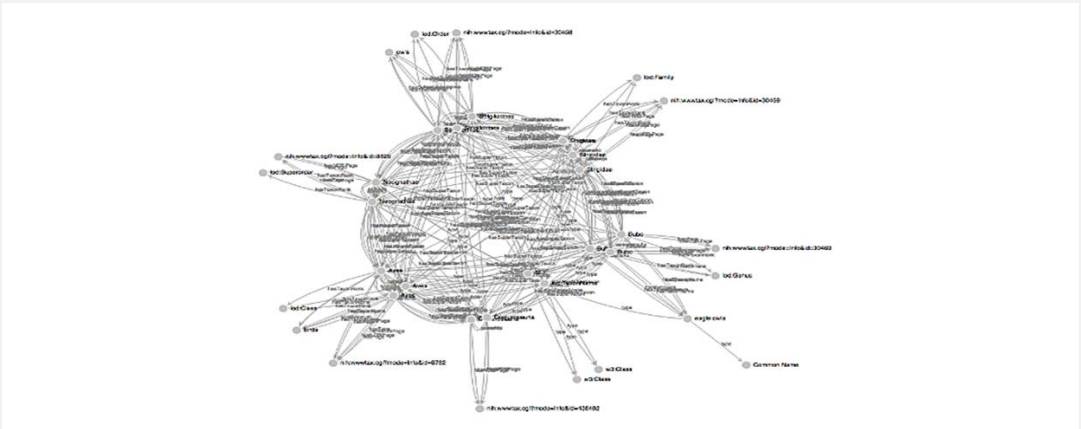


Fig. 6.3: Original RDF graph visualization from whole query result

Due to the common nature of RDF data together with reasoning output, the according SPARQL statement usually creates some giant components in a graph. A hairball-like graph, as shown in Fig. 6.3, gives a bad experience to users because it is difficult to read and learners may not satisfy the way of learning and teaching using linked data. As we analyzed the query data from DBpedia [72] and LODAC [88] databases, we found two major issues that are data redundancy and low readability.

Data redundancy

We have discussed about this issue in the beginning of this section. As we look at the data closely, it indicates that most inferred triples make a graph be highly complex. More than half of query triples are mainly formed by the reasoning results of *owl:sameAs*, *rdf:type* together with *rdfs:subClassOf*, and transitive properties. This behavior increases the average degree of the network and leads to have giant components, which produce a hairball-like graph.

Low readability

In general, most well-organized articles such as academic papers prepare background knowledge of some essential concepts before bringing readers to the main content. Thanks to a well-outlined paper, beginners can understand it by reading from the beginning part to the end part, while experts of its domain can skip the introduction part and go to the main content directly. However, it is hardly possible to do with RDF data, because triples have no ordering. Thus, to give a ranking score to each triple is necessary for any visualization tools.

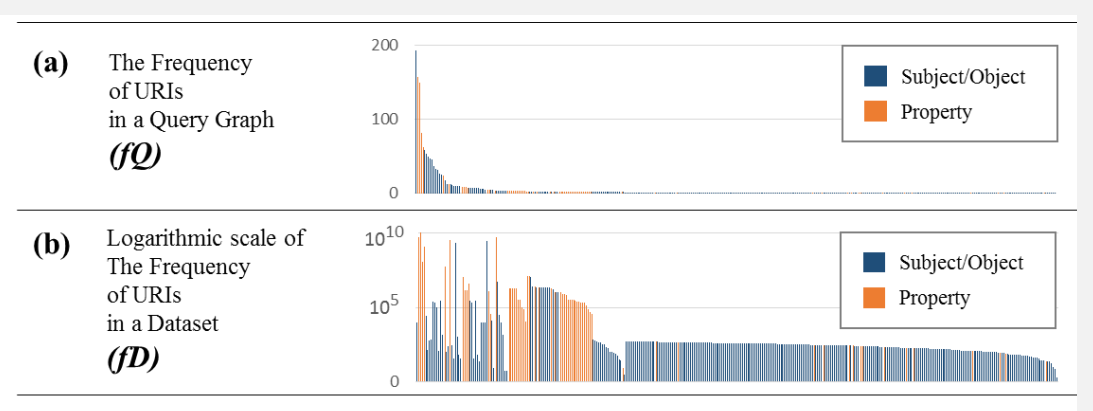


Fig. 6.4: Statistical analysis of URIs in the query result from DBpedia.

In this case, we observed the distribution of URIs. We found the distribution of the frequency of each URI in a query result *(fQ)* as shown in Fig. 6.4(a), where the horizontal axis shows individual URIs and the vertical axis shows the frequency of them. Several URIs have high degree. As we analyzed the high-degree URIs in each query, most of them are important to display in a graph as key concepts. For example, if we query a term *dbpedia:Tokyo*, the high-degree URIs becomes *dbpedia:Tokyo*, *dbpedia:Japan*, *dbpedia:Honchu* (The island where Tokyo located), *rdf:type*, *owl:sameAs*, *dc:subject*, etc. The *dbpedia:Tokyo*, *dbpedia:Japan*, and *dbpedia:Honchu* are remarkable because they are key concepts of "*Tokyo*" in our sense, whereas the *rdf:type*, *owl:sameAs*, and *dc:subject* are not much important for domain experts. Thus, we learned that using the frequency of each term in a query result alone is not enough. Next, we analyzed the frequency of every URI in a dataset *(fD)*, and compared each to the *fQ* chart one-by-one as shown in Fig. 6.4(b). This chart was drawn on a logarithmic scale because its distribution is extremely high variance. As *fD* of every URI found in the query result are estimated, a lot of high frequent ones are common properties such as *rdf:type*, *owl:sameAs*, *lodac:hasSuperTaxon*, etc. while the degrees of *rdf:type*, *owl:sameAs*, *dc:subject*, etc. while *dbpedia:Tokyo*, *dbpedia:Japan*, and *dbpedia:Honchu* are not much high.

This characteristic of the data is expressive. As query results are carefully analyzed, we found that URIs having high *fQ* can be treated as key concepts in the graph, while URIs having high *fD* indicate common information of the key concepts. This fundamental analysis will be utilized for ranking triples in the next section.

6.4. RDF Graph Diagram for Users (RDF4U)

As we discussed, learners have to be familiar with the knowledge representation of linked data in order to motivate them to consume and contribute RDF data. In this case, a node-link diagram is a suitable way to reduce a gap between human and Semantic Web. Understanding knowledge from a graph is quite challenging, because a graph is just a mathematical graph containing a set of nodes and edges. In order to deliver graph-based knowledge to readers, an application should interpret all nodes and links as knowledge structures and make decision to maintain or eliminate some triples. To achieve this goal, we have to address the issues that are mostly discussed in the previous section. Thus, this work is initiated to serve the following purposes.

- To simplify a complex graph by removing some redundant triples which are resulted from ontological reasoning.
- To serve different subgraphs on the basis of reading levels from common to topic-specific information.
- To filter a graph based on user preference.

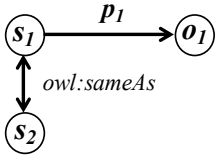
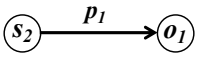
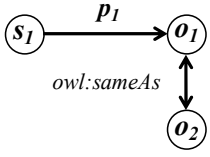
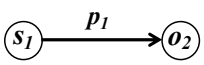
6.4.1. Graph Simplification

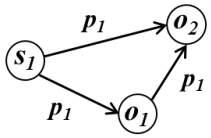
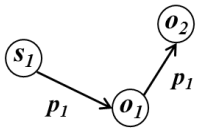
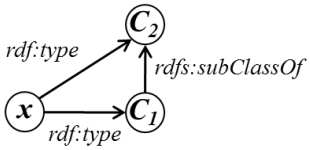
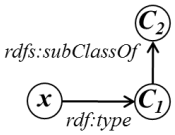
It is known that some well-prepared RDF repositories did reasoning on ontologies in order to support a SPARQL service, however, the inferred triples resulted in having giant components in a graph. As we investigate, equivalent or same-as instances (*owl:sameAs*), transitive properties (e.g. *skos:broaderTransitive*), and hierarchical classification (*rdf:type* together with *rdfs:subClassOf*) are commonly found in any query RDF graphs. Thus, this method aims to remove some redundant triples automatically by using rules that are defined in Table 6.1 and some descriptions as follows:

- *R.1*: To merge two same-as nodes at the subject side.
- *R.2*: To merge two same-as nodes at the object side.
- *R.3*: To remove implicit links that resulted by the chain of transitive links.
- *R.4*: To remove inferred links that caused by hierarchical classification.

Several rules use the occurrence number of a URI counted across data repositories in order to choose the most popular node from a same-as pair, because it has high opportunity to discover more knowledge in the next query.

Table 6.1: A set of rules used to simplify an RDF graph.
(Note: The term *fD(uri)* is a frequency of a URI occurred in datasets.)

Rule	Triples	Condition	Display only
To merge nodes			
R.1		$fD(s_2) > fD(s_1)$	
R.2		$fD(o_2) > fD(o_1)$	

Rule	Triples	Condition	Display only
To remove links			
R.3		*:p1 rdf:type owl:TransitiveProperty .*	
R.4			

If the query graph contains the following triples

```
spe:0644 owl:sameAs spe:Bubo_virginianus .
ge:1713 owl:sameAs ge:Bubo .
spe:Bubo_virginianus lodac:hasCommonName spe:SnowyOwl .
spe:Bubo_virginianus ltk:higherTaxon ge:Bubo .
spe:Bubo_virginianus ltk:higherTaxon fam:Strigdae .
spe:Bubo_virginianus ltk:higherTaxon odr:Strigiformes .
ge:Bubo ltk:higherTaxon fam:Strigdae .
ge:Bubo ltk:higherTaxon odr:Strigiformes .
fam:Strigdae ltk:higherTaxon odr:Strigiformes .
spe:Bubo_virginianus ltk:higherTaxon ge:1713 .
ge:1713 ltk:higherTaxon fam:Strigdae .
ge:1713 ltk:higherTaxon odr:Strigiformes .
spe:0644 ltk:higherTaxon ge:Bubo .
spe:0644 ltk:higherTaxon fam:Strigdae .
spe:Bubo rdf:type ltk:SimpleNorminalEntity .
spe:Bubo rdf:type ltk:NorminalEntity .
spe:Bubo rdf:type owl:Thing .
ltk:SimpleNorminalEntity
    rdfs:subClassOf ltk:NorminalEntity .
spe:SimpleNorminalEntity rdfs:subClassOf owl:Thing .
spe:NorminalEntity rdfs:subClassOf owl:Thing .
```

the LTK ontology informs that

```
ltk:higherTaxon rdf:type owl:TransitiveProperty .
rdfs:subClassOf rdf:type owl:TransitiveProperty .
```

and some example numbers of a URI in a whole dataset are listed as follows:

- *fD(spe:Bubo_virginianus)* = 100
- *fD(spe:0644)* = 90
- *fD(ge:Bubo)* = 230
- *fD(ge:1713)* = 180

Thus, the original graph can be simplified into the following simplified graph.

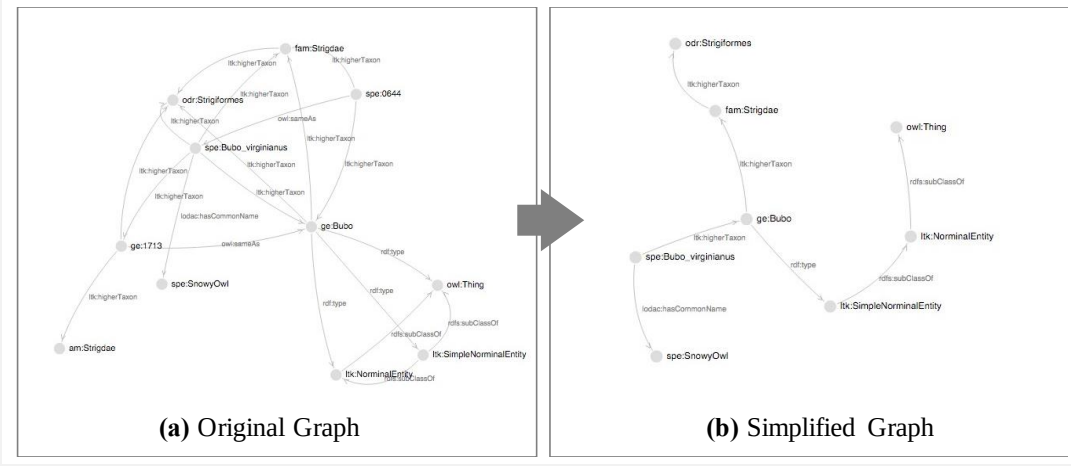
```
spe:Bubo_virginianus lodac:hasCommonName spe:SnowyOwl .
spe:Bubo_virginianus ltk:higherTaxon ge:Bubo .
```

```
 7 ge:Bubo ltk:higherTaxon fam:Strigdae .
 9 fam:Strigdae ltk:higherTaxon odr:Strigiformes .
15 spe:Bubo rdf:type ltk:SimpleNorminalEntity .
18 ltk:SimpleNorminalEntity
       rdfs:subClassOf ltk:NorminalEntity .
20 spe:NorminalEntity rdfs:subClassOf owl:Thing .
```

This scenario can be explained by the following list. Let *R.x* be the rule number *x*, and *n* be a line number *n* of the original graph,

- *R.1* excuses *1* and *13*, and then it results in the same triple as *4*, so *1* and *13* are ignored.
- *R.1* excuses *1* and *14*, and then it results in the same triple as *5*, so *1* and *14* are ignored.
- *R.1* excuses *2* and *11*, and then it results in the same triple as *7*, so *1* and *11* are ignored.
- *R.1* excuses *2* and *12*, and then it results in the same triple as *8*, so *2* and *12* are ignored.
- *R.2* excuses *2* and *10*, and then it results in the same triple as *4*, so *2* and *10* are ignored.
- *R.3* excuses *5* and *7*, and then it results in the same triple as *7*, so *5* is ignored.
- *R.3* excuses *6* and *8*, and then it results in the same triple as *8*, so *6* is ignored.
- *R.3* excuses *8* and *9*, and then it results in the same triple as *9*, so *8* is ignored.
- *R.3* excuses *18, 19* and *20*, and then it results in the same triples as *18 and 20*, so *19* is ignored.
- *R.4* excuses *16, 17* and *20*, and then it results in the same triples as *16 and 20*, so *17* is ignored.
- *R.4* excuses *15, 16* and *18*, and then it results in the same triples as *15 and 18*, so *16* is ignored.

Consequently, the simplified graph is sparser and simpler to read as shown in Fig. 6.5. In this case, about 35% of triples in the original graph are maintained.



(a) Original Graph **(b)** Simplified Graph

Fig. 6.5: Graph diagrams before and after executing the simplification rules

6.4.2. Triple Ranking

Besides the graph simplification, the ordering of triples in a graph is also important. The section of data analysis mentioned that the arrangement of any content is necessary for readers by preparing background knowledge in the beginning part in order to understand the main content well. In other words, some well-organized articles provide common information at the

beginning, and then bring readers to the topic-specific information. As we reviewed, existing works focused on seeking relevant data according to a query expression, but they less mentioned about how to order them according to readability. Thus, this research introduces a simple method to sort triples on the basis of different levels of knowledge structure. There are a concept level and an information level.

Concept Level

A general article contains different roles of concepts. In terms of RDF, concepts are resources (including subjects and objects) and properties. In this study, we propose two types of concepts that are general concepts and key concepts.

General Concepts

General concepts are terms that are commonly known such as “animal”, “tree”, “air”, “water”, etc., and they are also commonly found in a corpus. Since these terms are always found, normal readers potentially understand them by their background knowledge.

Key Concepts

Key concepts are important terms that are always found in the given article but rarely found in a dataset. The key concepts are more relevance to the given article rather than the general ones. For example, a chemical book contains a lot of exclusive words such as “isotope”, “isotone”, “isobar”, “precipitate”, etc., and these terms are not generally found in other kinds of books. If some widely-distributed articles such as newspapers have to talk about those terms, they have to give brief background knowledge of those terms certainly.

Moreover, the key concepts always present thorough the article, while general concepts are used as composition information for giving background knowledge of the key concepts as shown in Fig. 6.6(a).

Information Level

In addition, different levels of information are defined.

Common Information

Common information explains background knowledge that supports readers to understand the main content. It means that in one sentence, there are a lot of common terms rather than a technical or key terms. It generally gives introduction of key concepts by using general terms. It means that triples being common information consist of general concepts rather than key concepts as shown in Fig. 6.6(b).

Topic-Specific Information

Topic-specific information contains specific terms that are highly relevance to the article. Thus, some triples acting as topic-specific information comprise of key concepts rather than general concepts as shown in Fig. 6.6(c).

The level of each concept is valued according to a query result, so some concepts may be or may not be key concepts if query graphs are different. As we analyzed, the key concepts are commonly found in the query result but they are rarely found in the dataset, while the general concepts are frequently appeared in the dataset. This manner is consistent with the TF-IDF method, however an RDF dataset contains only separated triples but not documents of many words, so this method has to be adapted for RDF data.

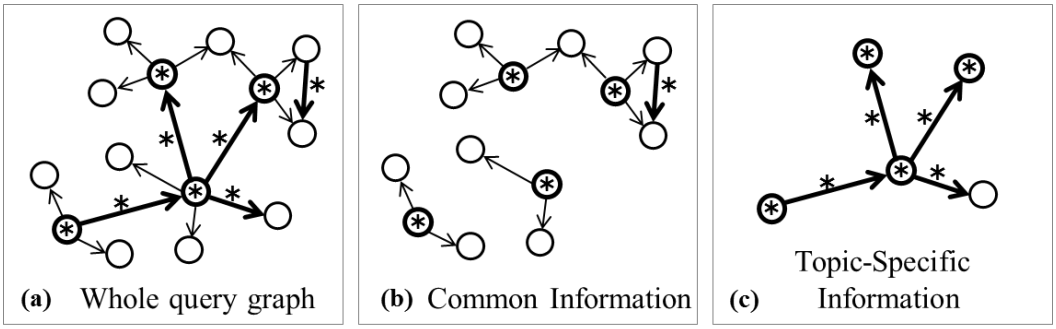


Fig. 6.6: The idea of common information and topic-specific information. (Nodes and links with stars (∗) indicate key concepts, whereas the others are general concepts.)

In this research work, we intend to define that a key concept has higher score than a general concept, so the scoring function of a URI *(w(uri))* is the occurrence number of a URI in a query result *(fQ(uri))* weighted by the its occurrence number found in datasets *(fD(uri))*. Since the data analysis informed that the variance of *fD(uri)* is extremely high, the logarithm is taken for this term. The function *w* is defined by the equation 6.1.

$$w(uri) = \frac{fQ(uri)}{log(fD(uri) + 1)} \qquad 6.1$$

The example scores calculated by this equation is demonstrated in Table 6.2.

Table 6.2: The values of *fQ*, *fD*, and *w* of each URI in a query result.
(Note: This is a part of the query result of "*lodac:Bubo_virginiaus*" from LODAC.
This table is ordered by the column *"w"*.
In the column "Type", *"IR"* is a resource and *"IP"* is a property.)

URI	Type	*fQ*	*fD*	*w*
http://lod.ac/species/Bubo_virginianus	*IR*	93	93	20.52
http://lod.ac/ns/species#hasSuperTaxon	*IP*	241	1,769,381	16.75
http://lod.ac/species/Bubo	*IR*	66	135	13.46
http://lod.ac/species/Great_Horned_Owl	*IR*	30	30	8.82
http://lod.ac/species/Strigidae	*IR*	40	657	6.17
http://www.w3.org/2002/07/owl#sameAs	*IP*	104	34,790,506	5.99
http://lod.ac/species/eagle_owls	*IR*	12	12	4.83
http://lod.ac/ns/species#hasTaxonName	*IP*	67	1,769,381	4.66
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	*IP*	84	23,7414,691	4.36
http://lod.ac/ns/species#hasParentTaxon	*IP*	57	801,028	4.19
http://lod.ac/species/great_horned_owl	*IR*	12	20	4.01
http://lod.ac/ns/species#TaxonName	*IR*	48	2,429,546	3.27
http://lod.ac/species/Coelurosauria	*IR*	32	43,056	3.00
http://lod.ac/ns/species#hasTaxonRank	*IP*	38	2,196,039	2.60
http://lod.ac/ns/species#ScientificName	*IR*	28	2,132,285	1.92
http://lod.ac/species/Animalia	*IR*	22	114,714	1.89

URI	Type	*fQ*	*fD*	*w*
http://lod.ac/ns/species#hasPage	*IP*	22	198,345	1.80
http://lod.ac/ns/species#hasCommonName	*IP*	24	1,299,842	1.71
http://lod.ac/ns/species#hasScientificName	*IP*	24	1,299,842	1.71
http://lod.ac/ns/species#hasNCBIPage	*IP*	22	906,800	1.60
http://lod.ac/ns/species#hasSynonym	*IP*	19	352,580	1.49
http://lod.ac/species/Kingdom	*IR*	4	25	1.24
http://lod.ac/ns/species#CommonName	*IR*	14	242,695	1.13
http://www.w3.org/2004/02/skos/core#closeMatch	*IP*	4	8,351	0.44
http://xmlns.com/foaf/0.1/page	*IP*	4	13,747	0.42
http://www.w3.org/2000/01/rdf-schema#subClassOf	*IP*	3	677,040	0.22

Next, a function named Visualization-Weight (*vw*) is defined to measure that a triple (⟨*s*,*p*,*o*⟩) should be in the direction of common or topic-specific information. It is the summary of weighting scores of subject (*s*), predicate (*p*), and object (*o*) of a triple as presented by the equation 6.2.

$$vw(\langle s,p,o\rangle) = \frac{\alpha \cdot w(s) \ + \ \beta \cdot w(p) \ + \ \gamma \cdot w(o)}{\alpha + \beta + \gamma} \qquad 6.2$$

The coefficients (α, β, and γ) of these terms are 1.0 by default; however, they can be adjusted if some domains place important to each term differently.

Some example scores calculated by the equation 6.2 where α, β, and γ are 1.0 are shown in Table 6.3. It shows that the triples having high *vw* score are more likely to be topic-specific information than the lower scores.

Table 6.3: The *vw* score of each triple in a query result.
(Note: This is a part of the query result of "*lodac:Bubo_virginiaus*" from LODAC.)

Triple	*vw*
<http://lod.ac/species/Bubo_virginianus> <http://lod.ac/ns/species#hasSuperTaxon> <http://lod.ac/species/Bubo> .	16.89
<http://lod.ac/species/Bubo_virginianus> <http://lod.ac/ns/species#hasSuperTaxon> <http://lod.ac/species/Strigidae> .	14.46
<http://lod.ac/species/Bubo_virginianus> <http://lod.ac/ns/species#hasParentTaxon> <http://lod.ac/species/Bubo> .	12.7
<http://lod.ac/species/Bubo_virginianus> <http://lod.ac/ns/species#hasTaxonName> <http://lod.ac/species/Great_Horned_Owl> .	11.29
<http://lod.ac/species/Bubo_virginianus> <http://lod.ac/ns/species#hasCommonName> <http://lod.ac/species/アメリカワシミミズク> .	10.58
<http://lod.ac/species/Bubo_virginianus> <http://lod.ac/ns/species#hasCommonName> <http://lod.ac/bdls/species/Great_Horned_Owl> .	10.30

Triple	***vw***
<http://lod.ac/species/Bubo_virginianus> <http://lod.ac/ns/species#hasTaxonName> <http://lod.ac/species/great_horned_owl> .	9.69
<http://lod.ac/bdls/species/Strigidae> <http://lod.ac/ns/species#hasSuperTaxon> <http://lod.ac/species/Strigiformes> .	9.58
<http://lod.ac/species/Bubo_virginianus> <http://lod.ac/ns/species#hasSynonym> <http://lod.ac/species/great_horned_owl> .	8.63
<http://lod.ac/species/Bubo> <http://lod.ac/ns/species#hasSynonym> <http://lod.ac/species/eagle_owls> .	6.53
<http://lod.ac/bdls/species/Bubo> <http://lod.ac/ns/species#hasTaxonRank> <http://lod.ac/species/Genus> .	5.51
<http://lod.ac/species/Strigidae> <http://lod.ac/ns/species#hasParentTaxon> <http://lod.ac/species/Strigiformes> .	5.39
<http://lod.ac/species/Great_Horned_Owl> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://lod.ac/ns/species#CommonName> .	4.74
<http://lod.ac/bdls/species/Aves> <http://lod.ac/ns/species#hasTaxonName> <http://lod.ac/species/birds> .	3.58
<http://lod.ac/species/Coelurosauria> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://lod.ac/ns/species#TaxonName> .	3.54
<http://lod.ac/species/Strigiformes> <http://lod.ac/ns/species#hasSynonym> <http://lod.ac/species/owls> .	3.30
<http://lod.ac/species/great_horned_owl> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://lod.ac/ns/species#CommonName> .	3.14
<http://lod.ac/species/great_horned_owl> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://lod.ac/ns/species#CommonName> .	3.14
<http://lod.ac/species/Aves> <http://lod.ac/ns/species#hasSynonym> <http://lod.ac/species/birds> .	2.52
<http://lod.ac/ns/species#ScientificName> <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://lod.ac/ns/species#TaxonName> .	1.80
<http://lod.ac/ns/species#ScientificName> <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://lod.ac/ns/species#ScientificName> .	1.36
<http://lod.ac/species/Animalia> <http://www.w3.org/2004/02/skos/core#closeMatch> <http://dbpedia.org/resource/Animal> .	0.83

6.4.3. Property Selection

In addition, although the problems discussed in the previous parts can be addressed, there are much more triples remained in the visualization and some of them are not interesting for readers. Since users have their own expectation to view a graph, they should customize the graph based on their interest by themselves. They always prefer to filter a graph by selecting only properties that they are interested.

This additional method named "Property Selection" is lastly described in this project. The method helps users to focus on information that they desire to view by selecting or deselecting some properties names to filter a graph. It is a simple technique that is always found in any visualization tool. In addition, we learn that most triples related to RDF, RDFS, and OWL are sometimes not needed by readers as shown in the following example triples.

- ⟨*foaf:Person, rdf:type, rdfs:Class*⟩
- ⟨*foaf:Person, rdfs:subClassOf, foaf:Agent*⟩
- ⟨*foaf:Person, owl:disjointWith, foaf:Organization*⟩

In this case, filter out some of these properties and resources one-by-one consume much user effort. Thus, this task allows to remove some triples containing some vocabularies from RDF, RDFS, and OWL from a graph by considering the namespaces of subjects, predicates, and objects. There are two options: (1) checking the namespaces of either subject or object, and (2) checking the namespaces of a predicate. The first option can remove ⟨*foaf:Person, rdf:type, rdfs:Class*⟩ because *rdfs:Class* is located at the object side, where the second option can remove ⟨*foaf:Person, rdfs:subClassOf, foaf:Agent*⟩ and ⟨*foaf:Person, owl:disjointWith, foaf:Organization*⟩ because of *rdfs:subClassOf* and *owl:disjointWith* are located at the predicate side.

6.5. Prototype

The proposed approach originates an idea to organize RDF data for a graph visualization. In order to verify the suitability and the feasibility of the proposed methods, a prototype has been developed.

6.5.1. User Requirement

Apart from the data analysis, we have gathered requirements from different users who have different levels of experience with Semantic Web and domain knowledge. In this part, the requirements from users are summarized into the following topics.

General Requirements

- An application should provide different input interfaces for different types of users. A simple interface allows users to enter a single URI, and then the system queries a graph automatically. Besides, tech users are allowed to input a SPARQL expression with the command "CONSTRUCT" for the advanced query.
- It is known that URIs are fundamental components in Semantic Web, and they are used as identifiers for machine-readable data on the web. However, most of them are difficult to be read by lay users. Thus, in a visualization, it should display human-readable labels in a graph for general users by default, and also provide an option to display URIs for tech users.
- Users are able to move any node in the graph diagram.

- To the principle of Semantic Web, a subject and a property must be URIs, and object can be either URI or literal. Pairs of a datatype property and a literal node are commonly used as metadata of a single resource. Some literal nodes may contains long strings such as the value of *dc:description*, *rdfs:comment*, *dbpedia:abstract*, etc. so long texts are not suitable to display in the limit area of a node-link diagram. Since these data are somehow useful for readers because they explain something in a human language, they should be displayed in another panel that users can access conveniently.

Graph Simplification

- Users can simplify a graph by merging same-as nodes, removing transitive links, and eliminating inferred hierarchical classifications.

Triple Ranking

- Since users have different background knowledge in a specific topic, beginners may be interested in reading common information before getting to topic-specific information, while experts may prefer to read only topic-specific information. Thus, the application should dynamically alter a graph according to the level of knowledge that users can customize and access on demand.

Property Selection

- Users can select only properties that they prefer to view.
- Some triples containing vocabularies from RDF, RDFS, and OWL can be ignored.

6.5.2. Implementation

According to the proposed approach and the user requirements, we implement a prototype of the knowledge presentation on the basis of the following features.

- Graph Simplification: To simplify a graph by removing redundant triples.
- Triple Ranking: To give ranking scores to triples based on common and topic-specific information.
- Property Selection: To filter a graph by selecting preferred properties.
- User Interaction: To control a graph according to user demand.

For this prototype, the functional diagram that described user actions and system workflows is depicted in Fig. 6.7, the user interface is demonstrated in Fig. 6.8, and example graphs that are resulted from user actions are displayed in Fig. 6.9. The prototype is accessible at the following URL.

http://rc.lodac.nii.ac.jp/rdf4u/

This prototype is a web application that is primarily developed using the force layout of the D3 JavaScript library [135]. The main user scenarios are described in the following topics. Each topic refers to steps displayed in Fig. 6.7, Fig. 6.8, and Fig. 6.9. In the description, the label "Fig. *n* (*s*)" denotes that it is the step number *s* of figure number *n*.

General Requirements

For general requirements, there are querying, displaying, interacting with a graph.

Querying a Graph

The main flow visualization is the query of graph as shown in Fig. 6.7(1). For this prototype, users have two options to get a graph as shown in the panel in Fig. 6.8(1). First, a

simple option is to give a single URI, then press the button "Query". Second, an advanced option in requires a "SPARQL CONSTRUCT" query, so users have to understand the SPARQL syntax. After that, the module "Query Service" in Fig. 6.7(2) forwards the query statement to a SPARQL endpoint for receiving a graph, counting the number of each URI, and inquiring the label of each URI.

Viewing a Graph

After the query sent to a SPARQL endpoint, a result is returned to the "Query Service" at the step Fig. 6.7(3). Then, the graph is forwarded to the module "Visualization Builder" at the step Fig. 6.7(4). As a result, the Visualization Buidler generates a graph visaualization to users as shown in Fig. 6.8(5) and Fig. 6.9(5). Since inferred triples are also retrieved, the original graph is highly complicated as shown in Fig. 6.3.

Interacting with a Graph

In addition, users can click and drag every node in a graph, all of labels are human readable, a URI is shown when a user moves a pointer over a node or a link. When a node is double-clicked, the literal information of a node is shown in the panel "Metadata" in Fig. 6.8. Moreover, every displayed triple is sent back to the Query Service again in order to be input data for other modules in next user actions as drawn in the step in Fig. 6.7(6).

Graph Simplification

For the graph simplification, users are allowed to select some preferred simplification rules at the steps Fig. 6.7(a1) and Fig. 6.8(a1). When users click on any options, the module "Graph Simplification" executes some related rules and forwards result triples to the "Visualization Builder" as can be seen at the steps Fig. 6.7(a2-a3). There are two options: the option "Merge same-as nodes" executes the rule *R.1* and *R.2*, and the option "Remove transitive links" executes the rule *R.3* and *R.4*. As a result, the graph visualization in Fig. 6.9(a) shows that the simplified graph is more readable than the original one. In the experiment, some redundant triples that are about 50-70% of the original query graph are eliminated during this process.

Triple Ranking

For the triple ranking, users can select the range of visualization ranking at the step Fig. 6.7(b1) by sliding the bars of a two-way slider bar or clicking on either the button "Common Information" or the button "Topic-Specific Information" at Fig. 6.8(b1). The former button displays triples having the lower *vw* score, while the latter one displays triples having the higher *vw* score.

In addition, the *vw* score calculated by the equation 6.2 is a floating number, so it is not proper for showing in the user interface. In order to communicate to users in a suitable way, the visualization tier uses the percentile of *vw* score and shows user as a visualization level from 0 to 100. Then, the module "Triple Ranking" computes and returns the triples that satisfy user input at steps Fig. 6.7(b2-b3).

The result of this action together with the graph simplification is shown in Fig. 6.9(b). This figure displays only common information that contains some key concepts and some general concepts, and the graph shows the background knowledge of the key concepts.

Property Selection

For the additional feature named "Property Selection", this feature is created to serve the need of users that prefer to view a graph depended on their interesting properties. Users can select or deselect some properties at the step Fig. 6.7(c1). For reducing these routine tasks, the

user interface allows users to hide resources and predicates that are vocabularies of RDF, RDFS, and OWL; and to show triples containing selected properties at the step Fig. 6.8(c1). Then, the module "Property Selection" filters the triples according to the user input, and forwards the result to the Visualization Builder at the steps Fig. 6.7(c2-c3). An example result of this scenario together with the graph simplification is shown in Fig. 6.9(c).

In summary of this section, the prototype demonstrates that our approach is possible and suitable to implement with the actual data especially in DBpedia [72] and LODAC [88]. The features that we provide satisfy all requirements that we have previously reviewed.

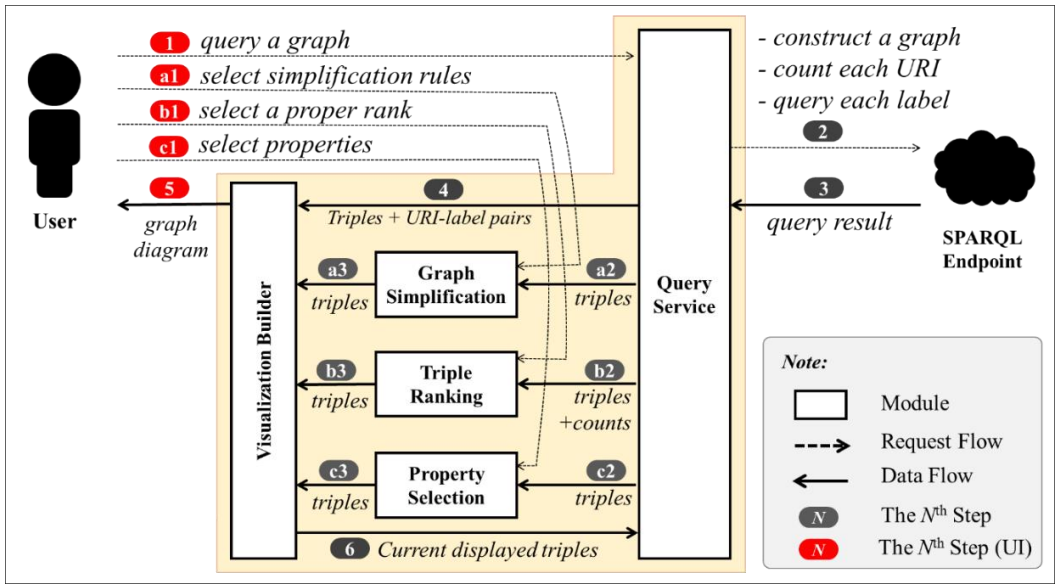


Fig. 6.7: BViz: Functional Diagram

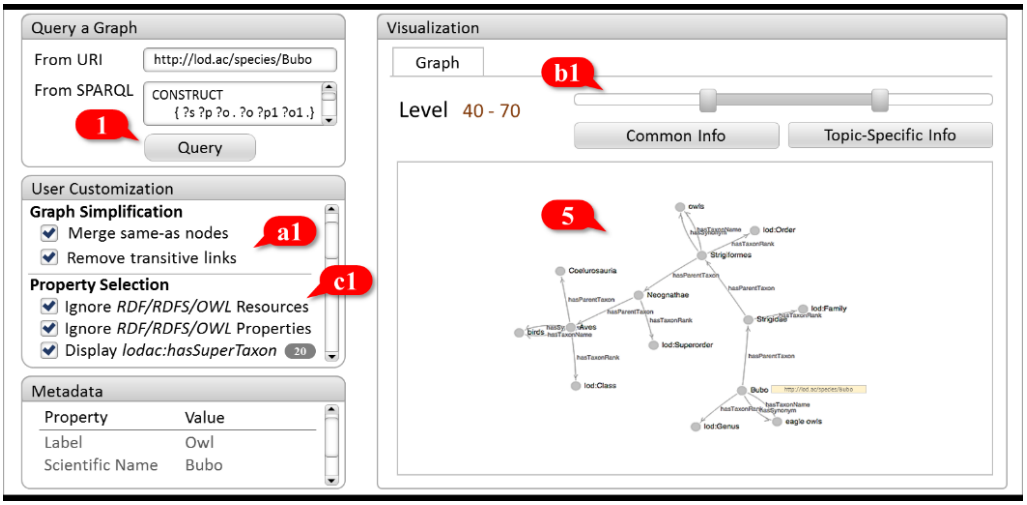


Fig. 6.8: BViz: User Interface

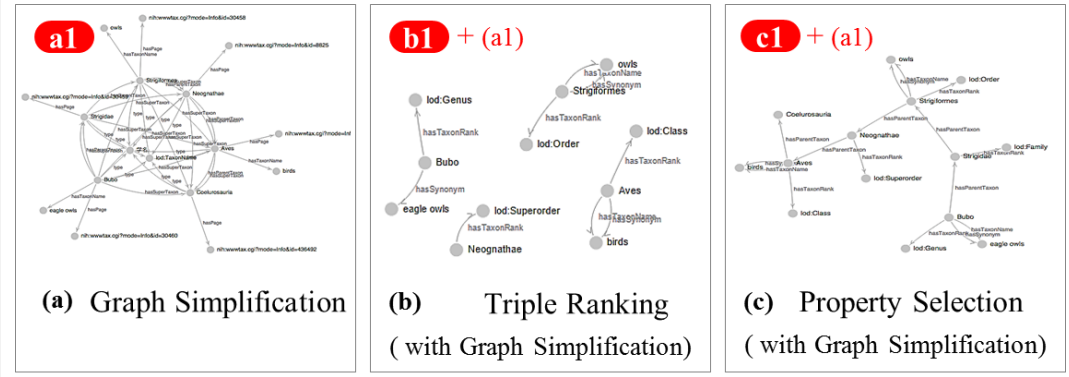


Fig. 6.9: BViz: Output from the prototype

6.6. Evaluation

The intention of the evaluation of this work is to show the practicability of our approach. The implementation described in the previous section shows that our proposed techniques can be developed to be an application using today's programming tools and software environment. The prototype is a key point that demonstrates the practicability of our introduced methods.

Next, for further evaluations supporting the practicability evaluation, the graph simplification and the triple ranking are measured against existent data, while the evaluation of the property selection is excluded because it is an additional feature having a simple mechanism. In order to verify the idea of both methods, the evaluations aim to express that the graph simplification can sparsify a query graph, and the triple ranking can determine the score of common information being less than the topic-specific information of the same article.

6.6.1. Graph Simplification

For the graph simplification, we mainly evaluate this feature by functional comparison against the reviewed tools. There are several functions support this feature but not the same strategies. Motif [33] replaces a dense component by an abstract shape, so a graph becomes easy to view, but its detail is omitted because it shows only abstract level. Gephi [85] allows users to filter a graph, but it does not provide an automatic task to reduce some inferred triples. FenFire [85] can show the focused node and its neighbors but it does not remove some links that create giant components being highly dense parts in a graph. Next, RDF Gravity [85] and IsaViz [99] can simplify a graph but they do not concern about the issue of same-as nodes, transitive links, and hierarchical classifications, so the dense parts in a graph still be found. According to the issues we raise, our simplification method has advantage over the reviewed tools in terms of eliminating redundant triples automatically by using the knowledge structure of Semantic technology.

In addition, we found that the simplification method can remove some inferred triples about 70.21% from the LODAC database, and about 34.78% from the DBpedia database (from 100 examples). This method is good for the LODAC data because most URIs in the LODAC are expressive, and class hierarchies and property hierarchies are clearly defined. However, for the DBpedia data, there are a lot of resources and properties that do not express the clearly meaning such as *http://www.wikidata.org/entity/P508s*, *http://wikidata.dbpedia.org/resource/Q43284*, etc., and one instance is the type of so many classes where most of classes are not categorized

in the well-shaped structure. In this case, experts from biodiversity domain also agreed that this method is appropriate for reducing the complexity of a messy graph and it is good for learning biodiversity from a knowledge graph.

6.6.2. Triple Ranking

For the triple ranking, since this feature is firstly invented here, it is hardly possible to compare to other tools. The way to evaluate the accuracy of the proposed algorithm is also difficult to do directly, because there are no any supervised data notify that which triples are common information or topic-specific information. In this case, another indirect evaluation method is invented.

For this indirect evaluation, we analyze many abstracts of publications from BioMed Central [134] that is an open access publisher. The advantage of the abstract of BioMed Central is that the abstract clearly contains the parts of background and method. We used the abstract because it is strengthening and does not contain much diffuse sentences. This behavior is very close to the manner of publishing RDF data. In this case, we assume that the part titled "Background" acts as the common information because the background contains a lot of general terms, and the part titled "Method" acts as the topic specific information because the method always contains many technical terms. Another advantage of using the abstract is that the numbers of terms in both the background and the method in a single abstract are just about the same.

There are about 2,142 abstracts and all of them are cleaned by removing some stop words and being lemmatized. A single abstract is viewed as a query graph. For the calculation of the *w(term)*, the *fQ(term)* is done by counting the given *term* under a single abstract and the *fD(term)* is done by counting the given *term* under all abstracts. The visualization-weight is calculated using all terms in the background part and all terms in the method part, and then the scores of *vw(backgournd)* and *vw(method)* are determined. In order to prove the usefulness of the proposed algorithm, the score of *vw(topic-specific information)* should be greater than the score of *vw(common information)*. Thus, the result from this experiment has to show that the score of *vw(method)* is greater than the score of *vw(backgournd)* of the same abstract. After making experiment, it is found that 71.53% of 2,142 abstracts are consistent with our hypothesis.

This result helps to verify that it is possible to rearrange pieces of knowledge from the common information to the topic-specific information using the popularity of terms. However, this evaluation experiment is not a direct way. It just helps to confirm the possibility to use the term frequency for rearranging data in a knowledge graph but it cannot perfectly prove the concept of our triple ranking method, because the RDF is structured data and itself contains knowledge structure in terms of ontology and reasoning which do not present in any unstructured natural language texts.

6.7. Summary

Using an RDF graph diagram in knowledge presentation is very challenging due two main issues. One is the issue of a large number of inferred triples creating a highly-dense graph like a hairball, and the other one is the issue about how to enable the flow of reading a graph diagram from common information to topic-specific information. For this reason, we initiated two main methods and an additional method to support readers. First, Graph Simplification executes the proposed Semantic Web rules for removing some inferred data. Second, Triple Ranking

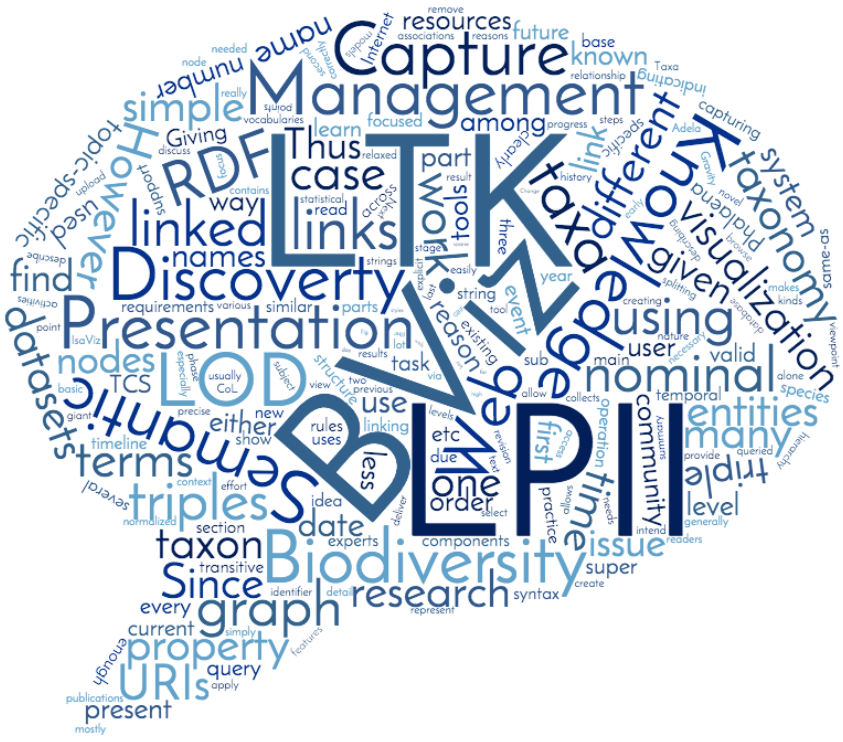
prepares different sections of a graph from common to topic-specific information for different levels of users by adapting TF-IDF algorithm for an RDF graph. Last, Property Selection is additionally developed to allow users to display or hide some triples by selecting some properties, and to help users to filter some triples containing some vocabularies from RDF, RDFS, and OWL. These methods mostly use the statistical analysis of a RDF graph together with the interpretation of RDF data as knowledge structures in order to produce an easily-readable node-link diagram for readers. The prototype is implemented by including an interactive RDF visualization in order to verify the suitability and the feasibility of this approach. It proves that these methods can be developed on the basis of today's technologies.

In conclude, the role of LOD in knowledge presentation can be demonstrated using the knowledge structure from ontologies. It can create a better node-link diagram for end-users, and it also enables users to realize the power of Semantic Web and LOD for enhancing the ability of KM systems. The effort of this chapter helps to confirm that it is possible and feasible to use LOD in terms of knowledge graph, schema, reasoning, and query for presenting biodiversity knowledge as well.

.....

CHAPTER **7**

DISCUSSION



The proposed projects have been developed in order to express the role of LOD in biodiversity KM process. The LTK project takes care the knowledge capture and knowledge exchange, the LPII project enables knowledge discovery, and the BViz takes into account the knowledge presentation. In this chapter, the various perspectives of each project and each KM activity are discussed. Eventually, the outcome of the integration of these projects are also demonstrated.

7.1. Biodiversity Knowledge Capture and Exchange

Since the LTK project accounts to knowledge capture and knowledge exchange, these KM activities are discussed in the same topic. As we reviewed, many approaches [13, 96, 63, 106, 109] always take care about how to storing up-to-date taxonomic data. In practice, keeping up-to-date data alone is not enough for comprehensively studying biodiversity, so the capturing the change in taxonomic knowledge becomes necessary for education. Several previous pieces of work on taxonomic databases mostly focused on the collection of name strings with proper identifiers at the first step of the integration of taxonomic information, but the history describing changes in taxonomic knowledge is less discussed. To address this issue, the LTK project provides a framework for capturing and exchanging the change in taxonomic knowledge using LOD. We introduce operations for capturing the changes, such as merging, splitting, replacing, changing a higher taxon, etc., as shown in Appendix. We discuss the values of our approach from four perspectives: knowledge representation, user engagement, and system integration, and challenge.

7.1.1. Knowledge Representation

In term of knowledge representation, this project mentions on different viewpoints of the change in taxonomic knowledge in order to have better understanding of biodiversity.

Chronological Change in Taxa

Browsing chains of changes in taxa is a feature with which learners can observe the historical changes in a given taxon. LTK provides properties indicating dynamic changes in taxa for this feature. Discussed in other pieces of work, the Taxonomic Concept Schema (TCS) [154] is one of the well-known approaches to describing a taxon concept in an informatics way. This approach was used to describe a taxon concept expressed as RDF in a piece of work titled "Describing Taxon Concept as RDF" [156]. The TCS regarded each concept as more static and organized operations of change into appropriate categories, so most activities do not focus on any aspect about historical information unlike LTK. In terms of using properties to represent any changes in the conception of taxa, our work introduced the hierarchy and configuration of the properties in Appendix.

To find chronological changes in taxa, learners just simply use the properties *ltk:mergedInto*, *ltk:splitInto*, and *ltk:replacedTo* in query statements. In addition, show that their subject and object are dominant in the change, LTK can also present the main concepts in the timeline by using the properties *ltk:majorMergedInto* and *ltk:majorSplitInto*, which are sub properties of *ltk:mergedInto* and *ltk:splitInto*, respectively, so the concepts connected by these properties have a stronger relationship than those linked by *ltk:mergedInto* and *ltk:splitInto*.

Query statements with the according properties result in only directed-adjacent nodes of a given concept, because there are asymmetric and non-transitive object properties. To reach all concepts having the same history can be queried by using the properties *cka:serialLinkTo* and *cka:semanticLink*. The former, *cka:serialLinkTo*, is a transitive and asymmetric object property, so all concepts in only one direction in a timeline occurring before or after the change in the given concepts can be queried. In addition, if it needs to find out all concepts in the same history, the query expression should include the property *cka:semanticLink*, which is a transitive and symmetric property and also a super property of *cka:serialLinkTo*.

Using LOD and Semantic Web reasoning with different configurations of properties can apply many styles of queries in order to support many styles of questions and answers.

Temporal Information of Taxa

In addition to the chronological changes in taxa, the use of temporal information of taxa enables learners to learn of the change in taxonomic knowledge in terms of the change in triples, for example the changes in classification, membership, metadata, etc. Browsing triples before and after the change makes learners understand the movement of taxonomic knowledge easily.

A change in a triple are expressed by a single operation of the change in relation between taxa, and some operations that share the same context (e.g. same publication or event) are grouped into a single event-centric model whose aspects about time and provenance are assigned. Each operation assured by the event entity can be transformed into an accepted triple that is happening during the begin and end time points. However, this kind of triples is not directly stored in the database, so a client needs to use SPARQL to build a snapshot model of a given concept at a given time point. In the case a concept is given without a time point, the system assigns a current time by default.

Although the event-centric model consumes many triples, the performance analysis from the previous section confirms that this is not an issue for current SPARQL engines. Thus, users do not only learn the association between data but also understand the precise context of the linked data by temporal information and references. They also recognize triples added or removed at different times, so they can learn the progress of biodiversity knowledge along with time.

Background Knowledge of Change

As we discussed, monitoring the change in knowledge is really nesessary, in addition, browsing reasons behind and results after a change is also important for learners. In terms of managing the changes and linked data, our approach has similar objectives as TaxMeOn [118], but both pieces of work are technically different due to specific purposes due to the issue of linking with background knowledge. TaxMeOn regularly models a change by using one triple containing an old taxon concept, a property indicating taxonomic change, and a new taxon concept. It sometimes uses an individual for indicating a change such as lumping and splitting. Thus, data model gives a simple and easily-understandable timeline of the changes in taxon concepts.

However, in the case of using only a single triple for representing a change, it is limited to demonstrate a link between changes, so associations between background knowledge cannot be implemented directly. In this case, the event-centric model becomes more advantageous for meeting this requirement because one operation can also be regarded as background knowledge, so the link between operations allows users to trace back to what reason behind or what effect after the change is. For this task, the properties *cka:cause* and *cka:effect* are used in a query string to find the reason and the result of a particular change, respectively. Our prototype demonstrates how two concepts are related by finding operations that are the background knowledge of a link between the given subject and object.

Ability to Publish Linked Data

The LTK project publishes different views of data models that are the event centric model, the transition model, and the snapshot model. The event-centric model represents the change in taxonomic knowledge base on the grounds of Semantic Web and the underlying community knowledge [22, 40]. There are operations of changes, context information, and relations between operations. To make the global access of data, the LTK uses the property *dct:isVersionOf* to refer to data from GBIF [140], CoL [63], uBio [106], and LODAC [146].

In addition, the role of the LTK for linked data is presented in Fig. 4.3. It shows that the LTK acts as the collection of change in taxonomy among taxonomic databases. It consists of external links for representative concepts and links to external datasets; the transition model and snapshot model presenting the change in taxonomy with a simple expression; and the event-centric model acts as the background knowledge of change. Making URI be dereferenceable, having a SPARQL endpoint, and integrating data with well-known ontologies allows the LTK can publish data to the LOD Cloud [50].

All knowledge graphs of LTK project in the figure are globally accessible. We recommend learners to find any taxon in the External Links part at first, because the simple nominal entities are close to scientific names and they are mostly linked with other external datasets. Next, learners are easy to browse the chronological and temporal information of contextual nominal entities from the transition model and the snapshot model. After that, they can browse the background knowledge and reference about any change in taxonomy from the event-centric model.

7.1.2. User Engagement

Another important task of building a knowledge graph for biodiversity is to encourage users such as taxonomists, ecologists, and molecular biologists to participate in providing and consuming a knowledge graph. However, many of them are non-computer-expert users. Since the RDF syntax is a requirement for using the power of Semantic Web and LOD, we recommend users understand basic RDF syntax in order to benefit from linked data. For this project, we intend to keep taxonomic knowledge representation as simple as possible under the boundary of the RDF framework.

Human Readability

Since the event-centric model is considered to represent data in various dimensions, the data model represented by RDF format is complicated by designed. However, the simplicity of the model can be improved by using simple identifiers, making the transition model, and querying the snapshot model, so the uses of the LTK become consequently simpler.

In terms of human readability, the uses of the contextual nominal entity and simple nominal entity are consistent with the idea of GNUB, which describes the usage of a name, and GNI, which collects name strings, respectively [97]. Thus, normalized and valid readable names are tied to a checklist such as CoL [63]. In another viewpoint, GBIF [26, 103] suggested that the persistent identifiers of taxa should be unfriendly to read, and a taxon concept and name should be presented separately so that the identifiers still endure, while the names change. This idea is basically consistent with the normalized database design that eliminates the difficulty of updating data, but the data model is much more complex for accessing. For capturing the change, we more focus on accessing linked data, but updating is less emphasized because the change in knowledge is recorded by appending a new revision. Working with a revision of knowledge, an identifier is does not necessarily have to be viewed as a persistent thing. This viewpoint leads to the idea that designing a data model is more relaxed than the use of persistent identifiers. Thus, it is possible and reasonable to represent a taxon concept by a URI containing a human-readable string of a valid name, instead of a non-human-readable identifier.

This simple representation comes with several advantages: lightweight data, recognizable URIs, and understandable linked data. Although it results in a slight decrease of information granularity, it improves user satisfaction in contributing and consuming data. However, this model does not restrict the use of URIs; either separating a taxon concept and name or using unreadable URIs is possible to implement.

Data Preparation

For biodiversity domain, data are usually contributed by domain experts, especially taxonomists. We have implemented a form-based web application with text fields for user input. It is proper for a small number of data in practice. However, when dealing with a large number of data, we recommend users upload a text file containing the event-centric model. Since this project does not put effort on user-experience design, in this phase, we encourage users to understand the basic syntax of RDF/Turtle. The data preparation steps are simply demonstrated as the following steps.

1) Giving contextual nominal entities for every taxon with every change. For example,

```
- genus:Bubo_1805
- genus:Nyctea_1826
- genus:Bubo_1999
```

2) Creating an event entity with a time interval and references. For example,

```
ex:event1999
     dct:source       pub:5224773 ;
     cka:interval     [tl:beginsAtDateTime "1999"] .
```

3) Creating instances of proper operations for every change. For example,

```
ex:mg1     rdf:type   ltk:TaxonMerger .
ex:rp1     rdf:type   ltk:TaxonReplacement .
```

4) Assigning contextual nominal entities before and after a change. For example,

```
ex:mg1     ltk:majorTaxonBefore  genus:Bubo_1805 ;
           ltk:taxonBefore       genus:Nyctea_1826 ;
           ltk:taxonAfter        genus:Bubo_1999 .
```

5) Assigning each operation to the event entity. For example,

```
ex:event1999    cka:assures     ex:mg1, ex:rp1 .
```

6) Giving links for causes and effects between operations.

```
ex:mg1  cka:effect  ex:rp1 .
```

7) Creating simple nominal entities to be representatives of external URIs for all taxa.

```
- genus:Bubo
- genus:Nyctea
```

8) Giving links between contextual nominal entities and representatives of external nominal entities.

```
genus:Bubo_1805       dct:isVersionOf genus:Bubo .
genus:Bubo_1999.      dct:isVersionOf genus:Bubo .
genus:Nyctea_1826     dct:isVersionOf genus:Nyctea .
```

9) Searching for external URIs from the Internet.

```
- lodac:Bubo
- lodac:Nyctea
```

10) Giving links between representatives and external URIs.

```
genus:Bubo      owl:sameAs lodac:Bubo .
genus:Nyctea    owl:sameAs lodac:Nyctea .
```

Since all operations are used in similar ways and URIs are human-readable, non-computer-expert users can create data and import them into the system. However, we learned that finding available URIs from known online datasets requires a lot of effort. In the future, we will find proper solutions to support this task and create a spreadsheet template for bulk upload.

7.1.3. System Integration

For the design of the data model, apart from satisfying the present requirements, the viewpoints of framework enhancement and data exchange are discussed.

Extensibility

Vocabularies describing the change in taxonomy are not limited by our LTK framework. There are metadata schemas for describing species such as comprehensive relationships documented by TCS [154]. Some of them provide relationships between names and concepts, but these relationships are usually summarized as valid (accepted), invalid (not valid but correctly proposed), and unavailable (neither valid nor correctly proposed). Some of the properties collected by TCS [154] and Franz [41] are is-homotypic-synonym-of, is-later-homonym-of, is-validation-of, is-vernacular-for, has-conserved-name, is-second-parent-for, and is-hybrid-child-of. However, our present work is focused mainly on the changes in taxonomic knowledge with simple situations, and the introduction of more terms is a future challenge. In this case, our framework allows increasing the capability of a system with other vocabularies by creating operations under either the classes of the change in conception (*cka:ConceptEvolution*) or the change in triple (*cka:RelationEvolution*) and reusing or adapting the Semantic Web rules.

For example, we can create an operation named "*ex:LinkVernacular*" for linking the common name or vernacular by using the properties name *ex:isVernacularFor*. This operation indicated a link between concepts, so it becomes the extension of *cka:RelationEvolution*. In this case, the new operation can be expressed as follows:

```
ex:LinkVernacular
      rdfs:subClassOf  cka:RelationEvolution ;
      cka:relation     ex:isVernacularFor .
```

The extensibility of our framework allows to apply for other domain rather than biodiversity informatics. For example, in business domain, two big oil companies *Exxon* and *Mobil* signed an agreement to merge and form a new company named *ExxonMobil* in 1999. In this case, we can create a new operation named "*ex:BusinessMerger*" for merging companies. Since this operation is the description of the change in concepts, it is considered to be the extension of *cka:ConceptEvolution*. When the operation is described, the merging between these two companies can be expressed as follows:

```
ex:mg
      rdf:type                ex:BusinessMerger ;
      cka:conceptBefore       ex:Exxon_1973 , ex:Mobil_1911 ;
      cka:conceptAfter        ex:ExxonMobil_1999 .
```

Interoperability

Thanks to the progress of Semantic Web technology, current RDF repositories can maintain billions of pieces of data. However, in reality, the technology does not rely on a single data source. The integration among taxonomic information systems is able to be done via the Internet by using either web services or SPARQL endpoints together with commonly accepted data models.

Challenge

For the LTK project, we assume that every change in taxonomy is clearly described. The representations of any changes are based on explicit evidence such as publication. In our experiment, before creating RDF data presenting the changes, a domain expert has to analyze the difference between several checklists, finding how names are different, and summarize them into operations of changes. For this reason, the precision of the RDF data relies on the completeness and the correctness of collected data. However, even existing references such as books and publications contain only insufficient information. For example, a synonymic catalogue, also called a "synonym list," is a standard way in taxonomy to present a historical summary of taxonomic studies on each species, including unaccepted names, misidentifications, references, etc. The following statement is an example from the synonymic catalogue [60].

> **Adela** Latreille, 1796
> 35. *reaumurella* (Linnaeus, 1758),
> Syst. Nat. (Edn 10) **1**:540 (*Phalaena*).
> *viridella* (Scopoli, 1763), Ent. Carniolica: 250 (*Phalaena*).

It is interpreted that the species *Phalaena viridella* is a synonym of the accepted name of the species *Adela reaumurella*, but the reason behind this synonym is not available. There are many possible reasons for why when the either the genus *Phalaena* or the species *P. viridella* was rejected, while our model preferred only explicit facts to be recorded. In other words, our present approach is not designed for dealing with any incomplete and inconsistent data. Although our data model can document these kinds of data by using contextual nominal entities as fragments of historical data, it cannot guarantee the precise interpretation of taxonomy if some of the linked fragments are disconnected or mistakenly connected. The interpretation cannot be uniquely and automatically determined and varied among taxonomists. Taxonomy more or less has objective aspects. In this case, a relaxed data model is needed to handle any implicit taxonomic knowledge and inspect correct knowledge from fuzzy explanation.

In practice, a publication sometimes does not describe an exact date of a particular change clearly, so a published date of the earliest publication that announced the change can be used to assign in the knowledge base as a workaround. A published date is generally written only with a year, but due to the constraint of the datatype *xsd:dateTime*, which is the range of the property *tl:interval* of the Timeline ontology [157], other components such as a day and a month are also required. In this manner, regarding the determination of date recommended by International Code of Zoological Nomenclature (ICZN) [141], if a date is not completely specified but either a month-year or a year is known, the last day of the known period should be entered in a knowledge base. In case a developer considers that this format shows too much

detail to users, an application can select a suitable part of the date and time string such as a month-year or a year number for interacting with users.

For the other remaining issue, there is no single globally-accepted taxonomy, so it becomes a great challenge at the moment. There are multiple branches of taxonomies and each of them is agreed by different communities of taxonomists. For example, GBIF taxonomy [140], which is used in GBIF database, is a candidate for the de fact standard of taxonomy in biodiversity informatics fields, while NCBI taxonomy [159] is a global standard taxonomy for bioinformatics field. Since the change in taxonomic knowledge across multiple accepted taxonomies is not normally found, the management on historical changes within a single accepted taxonomy is still in our scope. For this issue, it is recommended that the administration of multiple accepted taxonomies is possible to be performed by using some separated installations of taxonomic information systems and linking some Internet resources of the same taxa across all data repositories.

7.2. Biodiversity Knowledge Discovery

The aim of LPII project is to create a link prediction model for finding some potential interspecies interactions by analyzing the pattern of the existing dataset. Due to the sparsity of the dataset, the LPII project considers to use the appropriate structure of knowledge graphs together with linked data to build a model. For this reason, this project gives advantages to both informatics and biodiversity communities.

7.2.1. Value for Informatics

Dealing with sparse data is always unavoidable in the beginning phase of data collection, especially the early stage of linked data. The associations between resources are not dense, so it is really difficult to predict the undiscovered link. The prediction based on collaborative filtering is suitable when the data is dense enough. Then, prediction based on community structure becomes important for data that is less dense but highly clustered. When data is less connected, the projection of a bipartite graph and the community detection are hardly possible to be implemented, so the link prediction based on clustering defined by background information of resources becomes a key player. In case of using LOD, taxonomy of resources identified by well-known predicates such as *rdf:type*, *rdf:subClassOf*, *skos:broader*, *skos:narrower*, *lodac:higherTaxon*, etc., are evaluated.

The prediction model is also adaptable because of the nature and our knowledge of fungi-host interactions. When the neighborhood or collaborative similarity is used, some fungi having similar characteristics should be found at similar hosts because the hosts offer the similar environmental factors such as chemical composition, biological responses, environmental temperature, etc. In contrast, when many fungi share similar hosts, more complex ecosystem with a network between fungi and hosts is formed, and the prediction based on the community structure of fungi becomes meaningful in the link prediction model as well.

In addition, biological classification may be another dataset to assist the link prediction because the taxonomy of organisms is based on the similar characteristics of organisms observed by taxonomists and the taxonomy can be considered as background knowledge for making prediction. However, the biological classification is not always stable. The taxonomy changes based on new discovery of organisms, new taxonomic criteria, and new phylogenetic relationship. Thus, using the biological classification in the prediction model should be carefully taken into consideration.

For the reason above, the link prediction based on the dimensions of similarity between collaborators, probability to find links among community members, and probability to find links among nodes having similar background knowledge are always necessary for early stage of linked data.

7.2.2. Value for Biodiversity

For the advantage to the biodiversity domain, the prediction model is exploitable to represent how domain experts know the pattern of interspecies relationship and how they plan to do further observation. This method does not only improve the accuracy against the existing data but also helps to discover more fungus-host interactions. By observing the high-score and top-rank interactions in the predicted list, about twenty fungus-host interactions were found from the new discovery of KAHAKU [144] and other online literatures as shown in Table 5.6. Many high-score unknown interactions, which are interesting for experts, are also in the waiting list for future observation. This results provide confirmatory evidence that the LPII model is practical use for the real-world problem. This discovery is the great evidence to support the practicability of our hybrid recommender model.

The combination of the introduced scoring functions that capture the pattern of data and the notion of experts is in line with the practical observation process of interspecies interaction. In the beginning phase of biological observation, a dataset is extremely sparse, so most experts use the biological classification to be a guideline for making observation. In this case, the scoring function based on biological classification is one practical activity that transforms the tacit knowledge from mycologists into explicit knowledge in form of the prediction model. From the knowledge of well-experienced experts, the prediction of the relationship would be easy, but many young researchers in the field do not always have enough experience. Thus, this model together with in-hand data can support researchers with limited knowledge about biodiversity to predict and increase opportunities to observe any potential interspecies relationships. The significance of the present approach in the prediction based on the neighborhood similarity and the network structure lies in the demonstration of the power of LOD and data science for biodiversity knowledge. For this reason, the contribution of this project is not only offering the prediction model and the recommended list for less-experienced biologists but also giving feedback to experts about the roles of knowledge graph and LOD as meaningful features for making the prediction on interspecies interactions.

7.3. Biodiversity Knowledge Presentation

The BViz project aims to provide a suitable knowledge graph visualization that learners are easily to consume knowledge by learning from relationship among concepts. Thus, the three main methods: graph simplification, triple ranking, and property selection, are proposed to deliver an easily-readable knowledge graph to readers. The first and the second methods are major contribution, while the last one is an additional method used for fulfilling some minor requirements. In this project, we intend to introduce the according methods rather than a new fully-functioned visualization tool. Thus, this section points to the discussion about the usefulness, uniqueness, novelty, and prospect of this research.

7.3.1. Usefulness

Since a knowledge graph generated by RDF data is complicated by nature, learners are not convenient to read and understand knowledge from a graph directly. The analysis of

mathematical features of a graph alone is not enough for simplifying the complexity of an RDF graph, because the RDF graph has semantic relationships that should be interpreted as knowledge structures. We carefully examined the actual behavior of RDF datasets, and found that the semantic structure of the datasets is meaningful in terms of knowledge representation, and it is useful for our research. The observation includes data redundancy such as same-as nodes and inferred relations. When same-as nodes are merged and some inferred triples are filtered out by the simplification rules, some giant components in a network are eliminated, so the interactive graph on two-dimensional canvas becomes sparser and convenient for users to control and read.

In addition, the degree of importance of triples such as distinction between common and topic-specific information was also investigated. For this reason, we have to realize the importance of triples depended on the expertise level of users. For domain experts, only topic-specific information is needed to show, while common information should be more emphasized for beginners. A case of multiple links between two nodes caused by the hierarchy of property demonstrates how this method is suitable for arranging data for readers. In general, a super property in an upper ontology is labeled by a common vocabulary describing the broader meaning, while a sub property is used by a specific domain. After reasoning, the number of a super property is certainly greater than the number of a sub property, so the super property trends to be displayed at the common level while the sup property often appears at the topic-specific level.

7.3.2. Uniqueness

The uniqueness of this project is discussed by functional comparison. The functionality of some visualization tools: Motif [33], Gephi [85], RDF Gravity [46], Fenfire [119], and IsaViz [99]; are studied according to the key methods of this research.

Graph Simplification

There are several works support this feature but the strategies are different. Motif replaces a dense component by an abstract shape, so a graph seems simple, but its detail is omitted. Gephi uses mathematical characteristics of a graph such as a node degree and a weight on edge, but it does not employ the knowledge structure of Semantic Web to reduce some redundant links. FenFire fades away some far nodes, but the subgraph including the focused node and its neighbors can produce giant components. Next, RDF Gravity and IsaViz can simplify a graph by having users to query inside the graph or select some URIs to be visible or hidden. However, they less discuss about options to merge same-as nodes and remove transitive links, which are the main issues of having dense parts in a graph. Unlike these existing tools, our approach adopts Semantic Web rules to interpret data and eliminate this issue automatically.

Triple Ranking

The according visualization tools do not mention about a way to arrange contents in a graph for serving different levels of knowledge to different learners. A workaround is to filter some resources or properties based on user interest, but users have to put their effort to learn what they want to view and how to filter data by themselves. Thus, in this case, our work provides a smart way to solve this issue by analyzing the statistical feature of data and then it rearranging a knowledge graph from common to topic-specific information automatically.

Property Selection

Filtering a graph by selecting preferred properties is a common feature that most visualization tools provide. Our work was implemented in the same way. In addition, we added

an option to show or hide triples containing some vocabularies from RDF, RDFS, and OWL automatically, so users do not have to remove them one by one.

In summary, considering these three features, our solution has advantage over the existing visualization tools because our approach does not only allow users to customize a graph but also automatically deliver an easy-readable graph based on the knowledge interpretation and the statistical analysis of Semantic Web data.

7.3.3. Novelty

Due to the contradictory requirements from different types of learners: beginners and domain experts, we adapted TF-IDF method for ordering triple from common to topic-specific levels. The degree of commonness versus specificity is calculated by evaluating the nature of the dataset with the algorithm. After that, the RDF visualization application is designed to allow users to choose the level of information (from common to domain-specific information) that they need by clicking a button or controlling a two-way slider bar. The prototype was demonstrated and it got positive impression from users. Moreover, it can be considered that this work is a novel approach because it operates a graph at the knowledge level by concerning domain independent, so this approach is applicable to any domains.

7.3.4. Prospect

Since the arrangement of triples for reading is a novel approach, it has opportunity to be value-added by the community of Semantic Web researchers. This approach can be extended by applying various algorithms in order to satisfy diverse characteristics of data from other domains. We are going to apply this system as a learning and teaching tool for a specific domain rather than the biodiversity domain [20, 21]. Since a knowledge graph diagram can enhance the learning of biology [80], it should be apply to other fields as well. In future, some methods for identifying the level of learners from beginners (e.g. high school students) to experts (i.e. researchers) are considered.

Moreover, as our observation, although this RDF graph visualization application does not give technical knowledge of RDF to lay users directly, it makes them appreciate and understand the role of linked data for the future of KM. This one important task that attempts to break a barrier between humans and Semantic Web, and motivate them to contribute much more knowledge graphs.

7.4. Overall Outcome of this Study

All projects have been individually discussed and demonstrated the possibility and the feasibility to use LOD as a key role in the biodiversity KM process. In this section, we would like to discuss about the potential positions of these KM activities with the LOD cloud, to present the scenario about the flow of these KM activities and the overall KM process for managing and creating biodiversity knowledge, and to inform the capability of these KM activities for the biodiversity domain and others.

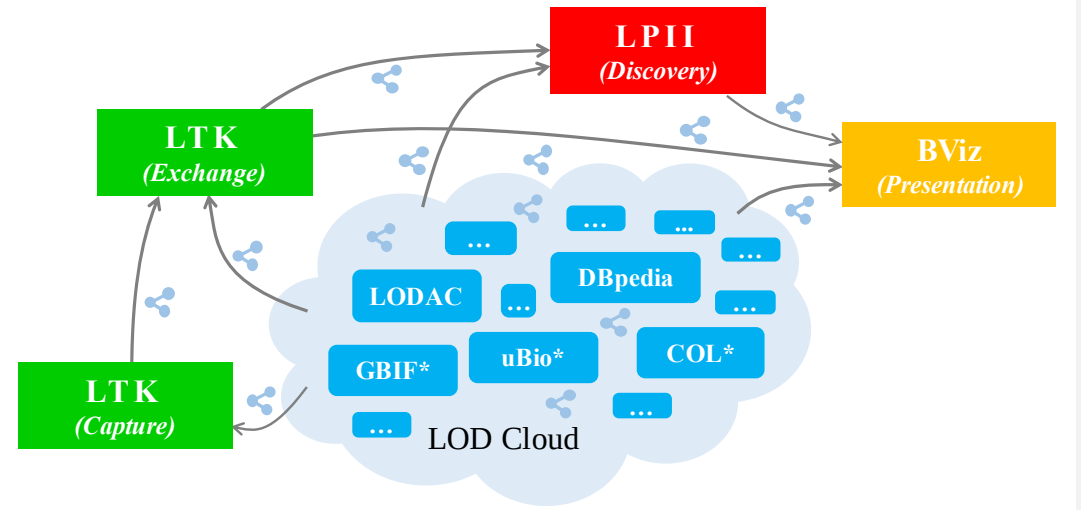


Fig. 7.1: The positions of LTK, LPII, and BViz with LOD cloud
(Note: All arrows are the flow of RDF data. The star (∗) indicated that linked data version of each online dataset.)

7.4.1. Exchanging Knowledge with LOD Cloud

We demonstrate an appropriate way to put each project into a particular position with LOD cloud as shown in Fig. 7.1. The LTK project, whose position is clearly assigned as shown in Fig. 4.3, is split into two parts: one is for knowledge capture and the other one for knowledge exchange. The LTK(capture) contains the event-centric model that captures changes in taxonomic knowledge across knowledge repositories. Since the data model is quite complicated but expressive, it has to be transformed into the simple models: the transition model and the snapshot model that are located in the LTK(exchange). Due to the simplicity of the models expressed in LTK(exchange), this module is selected to be another access point for providing and exchanging knowledge graphs through LOD cloud. As a sequence, the LPII project has proper knowledge graphs for building prediction models for discovering more information. The future of this unit is planned to be decision support systems for researchers in the biodiversity domain. After that, knowledge graphs from LOD, LTK(exchange), and LPII can be visualized as a node link-diagram for learners using the BViz project. The potential of BViz project will not limited at the node-link diagram visualization. It also aims to provide any suitable presentations of knowledge to learners according to the characteristics of data and the purposes of users. In the future, BViz will be a part of E-Learning service as well.

7.4.2. The Scenario of LOD-based KM Process for Biodiversity

To have a clear picture of the LOD-based KM Process, the scenario of the integration of these KM activities is summarized here step by step. This scenario simply explains the application of learning and exploring knowledge in the mycology domain, which is the branch of biology concerned with the study of fungi.

1) In the nature, fungi are growing by parasitizing some specific hosts that provide proper environment for fungi.

2) Biologists observe the nature and/or make experiments in the laboratory for studying the features, characteristics, behaviors, interactions, etc. of fungi.

3) Biologists analyze, discuss, and summarize what they know; and capture their tacit knowledge into a knowledge graph in a KM system using LTK (Knowledge Capture). Thus, at the moment, the KM system has a knowledge graph of the description, taxonomy, and interactions of fungi.
4) When biologists find out more evidence (such as genetic information and newly-discovered interactions) and improve the naming and classification systems, the changes in biodiversity knowledge are recorded [126].
5) At this state, some new pieces of and the changes in biodiversity knowledge are captured again using the event-centric model of LTK (Knowledge Capture).
6) After that, the event-centric model is transformed into the transition model and the snapshot model using LTK (Knowledge Exchange) for exchanging data with other repositories through the Internet. Thus, knowledge about the description, taxonomy, and interaction of fungi is increasing.
7) Making prediction on fungus-host interactions to be a guideline for the next observation can be done using LPII (Knowledge Discovery). However, some fungus-host interaction data that have been collected for a long time may contain some obsolete names and taxonomies. Thus, all names and taxonomies have to be refreshed using the up-to-date data from LTK (Knowledge Capture and Exchange).
8) Biologists use the potential missing fungus-host interactions from LPII (Knowledge Discovery) to do observation again. When new interactions are discovered, they are captured into the knowledge graph again using the ability of LTK (Knowledge Capture and Exchange).
9) Biodiversity knowledge from the knowledge graph can be delivered to any learners, who are interested in this topic, by many ways. One straightforward way is to create a concept-map or a node-link diagram to present biodiversity knowledge using BViz (Knowledge Presentation). Thus, learners have opportunity to perceive various elements of biodiversity knowledge such as the up-to-date or the temporal description of, the classification of, the chronological changes in knowledge of, and the existing or the potential interactions of the organismal groups of fungi from multiple sources.
10) When learners obtain much more knowledge; they may have enough guideline and motivation to observe the nature, study more about fungi, and update the biodiversity knowledge continuously.

7.4.3. Capacity and Opportunity of this Thesis

This thesis mainly uses biodiversity as domain knowledge and solves some issues based on the biodiversity. However, the ultimate goal beyond this thesis is to have LOD-based KM System for any other domains. This part describes some contributions that have potential for applying our approach to other domains.

Knowledge Capture

The idea of the Event-Centric Model can be applied for any other domains as well, because the change in knowledge is commonly found in our daily life. For example, the changes in classifications and names of space objects (Astronomy), the changes in regions and names of states (History and Political Science), the merging and renaming of firms (Business and Economics), the change in the meaning of vocabularies and language structures (Linguistics), etc. However, since the ontology introduced by LTK are mainly focusing on biodiversity

domain, developers of another particular domain have to adjust the vocabularies and ontologies for making them be consistent with their domain.

Knowledge Exchange

The terms Nominal Entity, Simple Nominal Entity, and Contextual Nominal Entity are introduced for the biodiversity only. However, the idea of using human-readable and/or context-included identifiers can be applied to other domains as well. In addition, the structures of the Transition Model and the Snapshot Model can be reused directly for having simple and lightweight RDF expressions, but the vocabularies and the transformation rules have to be adjusted according to the needs of a particular domain.

Knowledge Discovery

The strategy of LPII that uses the combination of scoring functions based on a bipartite graph, a projection network, and a taxonomy for making a prediction of any interaction data can be applied to any other problems about recommendation systems such as the prediction of co-authorship of academic papers. In this case, the developers have to realize that the interaction data is sparse, and the interaction pattern of data together with the taxonomy of items are conductive to the prediction result under the focusing domain. In addition, some more features can be included if that domain requires, and a proper similarity index and a community detection method should be carefully selected.

Knowledge Presentation

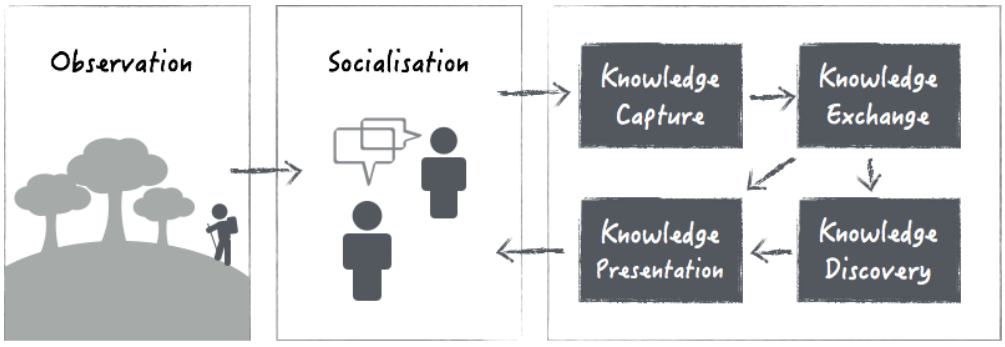
The BViz project has been designed to be a domain-independent methodology. However, the weighs of the subject, a predicate, and an object of a triple in the visualization-weight (*vw*) function can be fine-tuned for achieving the characteristic of an individual domain. Moreover, the developers have to understand their users in order to deliver a great user-experience graph visualization tool to learners.

In conclude, the position of each KM activity, the flow of knowledge graph, the scenario, and capability of this study are clearly depicted. The flow of knowledge graphs is not ended at the BViz project, because they are also delivered to humans including learners and researchers. When humans can discover more knowledge by making observation and experiment or having social activities, they will transfer their knowledge into LTK(capture) again. Therefore, the knowledge graph circulates continuously through machines and humans. Since it is consistent with the spiral of knowledge creation [93], the knowledge graph-based management has much more opportunity to support the creation of innovation someday.

……

CHAPTER 8

SUMMARY



This thesis introduced the research problems and methods to study the role of LOD in KM Process by using the practical cases under biodiversity domain. Essential KM activities: knowledge capture, knowledge exchange, knowledge discovery, and knowledge presentation are studied by the proposed projects: LTK, LPII, and BViz. The suitability and feasibility are already demonstrated in previous chapters. In the last chapter, we would like to summarize all of our study and recommend future work.

8.1. Thesis Summary

This thesis aims to study the roles of LOD in KM process by working with the biodiversity domain. In this study, we investigate four KM activities that are commonly used in any KM processes. They are knowledge capture, knowledge exchange, knowledge discovery, and knowledge presentation. To achieve this goal, three projects are introduced to analyze the roles of LOD in these KM activities. The Linked Taxonomic Knowledge (LTK) mainly takes responsibility for the knowledge capture and knowledge exchange, the Link Prediction on Interspecies Interaction (LPII) primarily accounts for the knowledge discovery, and the Biodiversity Knowledge Graph Visualization (BViz) is in charge of the knowledge presentation.

LTK demonstrates that we can use a knowledge graph and schema to capture the change in taxonomic knowledge in RDF format and the knowledge graph can be publicly exchanged using Semantic Web reasoning and SPARQL. Three kinds of knowledge graph: the event-centric model, the transition model, and the snapshot model, are designed for specific purposes. First, the even-centric model that is used to capture the change in biodiversity knowledge includes operations of change, relations between background knowledge of the changes, aspects of time, and references, so it is complex by design but it is flexible for the purpose of different applications. Second, the transition model is used to present the chronological change between taxa, and the model is simple and consistent with general triples in LOD cloud. Last, the snapshot model presents a triple with a time interval, so learners have to add a condition of time when querying data from this model. Both transition model and snapshot model are proper for knowledge exchange while the even-centric model is appropriate for knowledge capture. This project also introduces the simple nominal entity for encapsulating a taxon concept and a scientific name within a single URI, and the contextual nominal entity for representing a taxon concept, a scientific name, and an aspect of time within a single URI. Although it reduces the information granularity but the data are more friendly for learners. This project also provides a web interface, and it results in the possibility to capture the change in taxonomy of moths and make them publicly exchange through the LOD cloud.

Second, LPII verifies that the proper structure of a knowledge graph can support the discovery of new knowledge. This project uses three different knowledge graphs for making prediction. We start the construction of the interaction between fungi and hosts by a bipartite graph, and then compute the scores of missing links based on the collaborative filtering method. We then transform the bipartite graph into projection networks of fungi and hosts, do community detection for creating clusters of fungi and hosts, and estimate the scores of missing links using the probability to find those links in the generated cluster. Finally, we retrieve the biological classification of fungi and hosts using LOD from the LODAC database, and calculate the scores of missing links based on the biological groups of fungi and hosts. We combine these three scoring functions to be the weighted hybrid recommender system for finding potential missing links between species. It has been found that the linear combination of three scoring functions is more accurate than other combinations. All perspectives are statistically significant and play different roles in different characteristics of data. The collaborative filtering and the community structure are highly significant when fungal degree is not low, whereas biological classification becomes highly important when node degree and community size are not high.

Third, BViz demonstrates that it is possible to use a knowledge graph for visualizing a node-link diagram for learners. However, due to the behavior of RDF data and the nature of a graph diagram, the query graph is extremely complicated like a hairball, and learners cannot rearrange a graph for proper reading flow. For these reasons, we propose three methods: the graph simplification, the triple ranking, and the property selection. First, the graph

simplification introduces new rules to eliminate some inferred links caused by same-as nodes, transitive properties, and the chain of class membership. It is found that about half of triples are removed from the complicated graph using the introduced rules. Second, the triple ranking adopts TF-IDF to give scores to all triples and rearrange them from common information to topic-specific information. Thus, learners are able to read a node-link diagram in a proper way. Last, the property selection is a minor method that helps learners to display or hide some triples including some selected properties. A feature for hiding some resources under the specific namespaces is also added. These three methods can be controlled by users in the interactive user interface, so learners can query a knowledge graph, simplify the graph, rearrange the graph, and select some parts of the graph. Thus, this project can support learners to learn biodiversity knowledge from a node-link diagram.

These three projects assure that the features of LOD play important roles in any KM process based on the study on major KM activities. First, for the knowledge capture, the LTK project shows that it is possible to use a knowledge graph and schema to preserve the change in biodiversity knowledge with context as an event of changes. Second, for the knowledge exchange, the LTK project helps to ensure that a knowledge graph, schema, reasoning, and query can improve the interoperability among knowledge management systems. Third, for the knowledge discovery, the LPII project points out that the proper structure of a knowledge graph and query can help biologists to analyze, calculate, and recommend some potential missing interactions of species. Last, for the knowledge presentation, the BViz project expresses that knowledge graph, schema, reasoning, and query can create an appropriate node-link diagram visualization for learners to learn biodiversity from a knowledge graph.

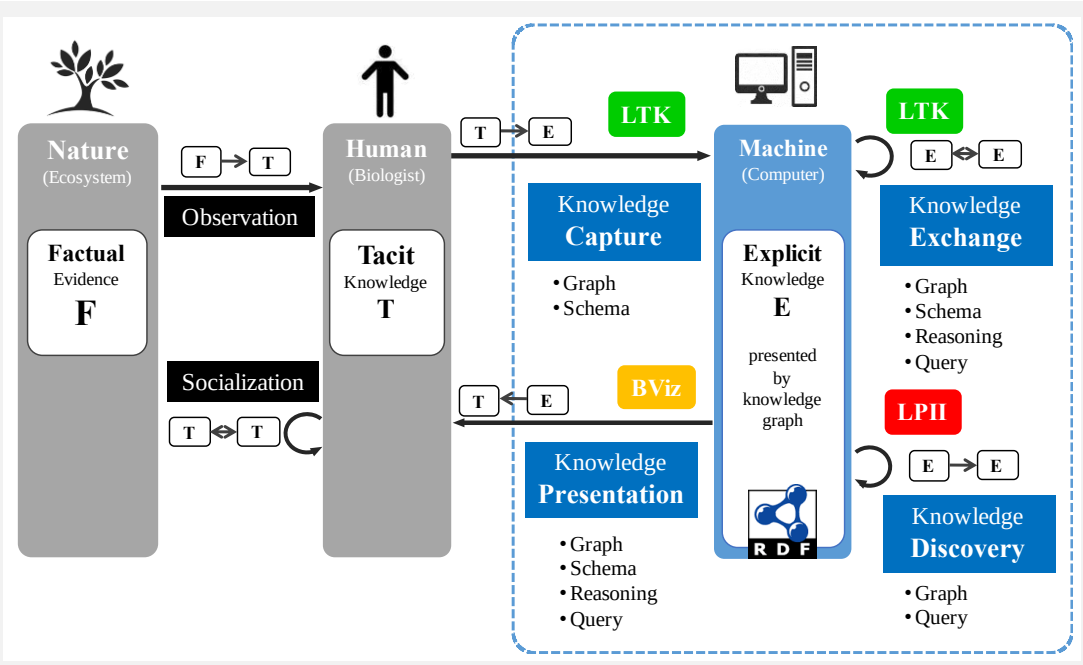


Fig. 8.1: The summary of this thesis.
– **F** is Factual evidence that is occurred in the nature or ecosystem.
– **T** is Tacit knowledge that human has a justified true belief about the nature.
– **E** is Explicit knowledge that is presented by a knowledge graph.
– A blue dash line encircles the scope of this thesis.

In the overall picture, it can be summarized into the big picture of KM process such as the knowledge creation process as demonstrated in Fig. 8.1. Researchers and learners acquire

biodiversity knowledge by observing factual evidence. They use social activities to summarize what they found into their tacit knowledge. They can capture their tacit knowledge into explicit knowledge using a knowledge graph in a KM system. The knowledge graph can be exchanged with other KM systems using the power of LOD. Then, a computer system can use knowledge graphs from different sources for discovering new knowledge. Knowledge graphs can be presented to a learner in order to add much more tacit knowledge. Researchers and learners can have social discussions again and observe to find more factual evidences by following a guideline from the KM system. The flow of KM activities is not limited to the according statements. It can be adapted to be another KM process in order to satisfy the strategy and requirement of an organization as well.

At last, this thesis researches that LOD is feasible to enhance KM systems. The study of the four main KM activities (knowledge capture, knowledge exchange, knowledge discovery, and knowledge presentation) using the three projects (LTK, LPII, and BViz) helps to demonstrate the role of LOD in KM process. Although this study is done under the biodiversity domain, problems and solutions that are faced during the study are common issues for other domains. Therefore, the proposed methods can be applied for others domains in order to manage knowledge graphs for global uses as well.

8.2. Future Work

The intention of research in the doctoral course is to study the role of LOD in KM process using the case study of biodiversity domain. Beyond the goal of this thesis is the real implementation of a full-functioned knowledge graph management system for any other domains. In this case, much more effort, budget, technologies, tools, as well as supports from other agencies are needed. To achieve this plan, some further points have to be considered.

- To have high volume and high quality of RDF data, the contribution from various providers is needed [16]. Thus, to encourage non-computer-expert users to get involved with the system, an application should have rich user-experience design.
- It is known that data around the world are mostly unstructured data. Writing natural language statements together with RDF statements is a high-cost task and requires technical knowledge, so either organizations or individuals are not willing to do. An automatic data converter that can migrate other legacy datasets into a well-shaped human-readable knowledge graph should be developed and customized for specific domains [61].
- One significant task of creating RDF data is to give identifiers to every resources. Generating local URIs from structured data is unproblematic, however, reusing URIs from or mapping them to some well-known datasets and make them be the five-star linked data consumes much more effort [67]. Automatic instance and ontology matching is needed to help data providers to reduce human effort to search and collect external URIs and to make links to the LOD Cloud by oneself.
- The fruitful product of our study and the investment of knowledge management is to have a powerful KM system. In this case, the system needs high-quality functions for authentication, authorization, and administration that can manage user privileges and access controls over a data layer [56]. Moreover, the licensing of knowledge graph must be properly declared.

As can be seen, we and new generations will have opportunity to have a full-functioned KM system for open knowledge graphs, although it seems far away. We, contributors, wish that our study together with other pieces of research in the past and future will be a part of a collaborative effort to drive the global LOD-based KM systems.

.....

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions.," *Knowledge and Data Engineering, IEEE Transactions on,* vol. 17, no. 6, pp. 734-749, 2005.

[2] G. N. Agrios, Plant pathology Fifth edition, Elsevier, 2005, p. 922.

[3] C. Alexopoulos, C. Mims and M. Blackwell, Introductory mycology, 4 ed., John Wiley & Sons, 1996, p. 868.

[4] K. M. Andersson, D. Kumar, J. Bentzer and E. Friman, "Interspecific and host-related gene expression patterns in nematode-trapping fungi," *BioMed Central Genetics,* vol. 15, no. 1, p. 968, 2014.

[5] D. Artz and Y. Gil, "A survey of trust in computer science and the semantic web," *Web Semantics: Science, Services and Agents on the World Wide Web,* vol. 5, no. 2, pp. 58-71, 2007.

[6] M. Balabanovic and Y. Shoham, "Fab: Content-Based Collaborative Recommendation," *Communications of the ACM,* vol. 40, no. 3, pp. 66-72, 1997.

[7] R. C. Banks, C. Cicero, J. L. Dunn and et al., "Forty-fourth Supplement to The American Ornithologists'union Check-list of North American Birds," *The Auk,* vol. 120, no. 3, pp. 923-931, 2003.

[8] S. Barker, D. Slingsby and S. Tilling, "Teaching biology outside the classroom," *Is it heading for extinction,* pp. 14-19, 2012.

[9] J. Bascompte and P. Jordano, "Mutualistic Networks. Monographs," *Population Biology,* pp. 1-206, 2014.

[10] C. Basu, H. Hirsh and W. Cohen, "Recommendation as Classification: Using Social and Content-Based Information in Recommendation," in *The 10th Innovative Applications of Artificial Intelligence Conference on Artificial Intelligence (AAAI/IAAI-98)*, 1998.

[11] I. Becerra-Fernandez and R. Sabherwal, Knowledge management: systems and processes, Routledge, 2014.

[12] T. Beckman, "A Methodology for Knowledge Managment," *International Association of Science and Technology for Development (IASTED) AI and Soft Computing Conference,* 1997.

[13] W. G. Berendsohn, "A taxonomic information model for botanical databases: the IOPI model," *Taxon*, pp. 283-309, 1997.

[14] E. Berlow, J. A. Dunne, N. D. Martinez, P. B. Stark and R. J. B. U. Williams, "Simple prediction of interaction strengths in complex food webs," *Proceedings of the National Academy of Sciences*, vol. 106, no. 1, pp. 187-191, 2009.

[15] C. Bezerra, F. Freitas and F. Santana, "Evaluating ontologies with competency questions," in *Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, 2013.

[16] C. Bizer, T. Heath and T. Berners-Lee, "Linked data-the story so far," in *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, 2009.

[17] D. Brickley and R. V. Guha, "RDF Schema 1.1," 2014. [Online]. Available: https://www.w3.org/TR/rdf-schema/.

[18] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," in *The 7th International World-Wide Web Conference (WWW1998)*, 1998.

[19] R. Burke, "Hybrid recommender systems: Survey and experiments," *User modeling and user-adapted interaction*, vol. 12, no. 4, pp. 331-370, 2002.

[20] R. Chawuthai, H. Takeda and T. Hosoya, "Link Prediction in Linked Data of Interspecies Interactions Using Hybrid Recommendation Approach," in *The 4th Joint International Conference (JIST2014)*, Chiang Mai, Thailand, 2015.

[21] R. Chawuthai, H. Takeda, V. Wuwongse and U. Jinbo, "A logical model for taxonomic concepts for expanding knowledge using Linked Open Data," in *Workshop on Semantics for Biodiversity (S4BioDiv 2013)*, 2013.

[22] R. Chawuthai, V. Wuwongse and H. Takeda, "A Formal Approach to the Modelling of Digital Archives," in *The Outreach of Digital Libraries: A Globalized Resource Network*, Springer, 2012, pp. 179-188.

[23] F. Cheng, W. Lu, W. Li and et al., "Prediction of drug-target interactions and drug repositioning via network-based inference," *PLOS Computational Biology*, vol. 8, no. 5, p. e1002503, 2012.

[24] M. C. Cobanoglu and et al., "Home Browse the Journal Articles ASAP Current Issue Submission & Review Subscribe About the Journal Article Previous Article Next Article Table of Contents Predicting Drug–Target Interactions Using Probabilistic Matrix Factorization," *Journal of proteome research*, vol. 5, no. 11, pp. 2909-2918, 2013.

[25] P. W. Crous, W. Gams, J. Stalpers and et al., "MycoBank: an online initiative to launch mycology into the 21st century," *Studies in Mycology*, vol. 50, no. 1, pp. 19-22, 2004.

[26] P. Cryer, R. Hyam, C. Miller and et al., "Adoption of persistent identifiers for biodiversity informatics: Recommendations of the GBIF LSID GUID task group, 6. November 2009," *Global Biodiversity Information Facility (GBIF), Copenhagen, Denmark (version 1.1, last updated 21 Jan 2010)*, vol. 62, 2010.

[27] G. C. Cummins and Y. Hiratsuka, Illustrated Genera of Rust Fungi, 3 ed., APS Press, 2003.

[28] A.-S. Dadzie and M. Rowe, "Approaches to visualising Linked Data: A survey.," *Semantic Web Journal*, pp. 89-124, 2011.

[29] M. Dean and G. Schreiber, "OWL Web Ontology Language Reference," 2004. [Online]. Available: https://www.w3.org/TR/owl-ref/.

[30] M. Deng, "Prediction of protein function using protein-protein interaction data," *Computational Biology*, vol. 10, no. 6, pp. 947-960, 2003.

[31] P. F. Drucker, "The age of social transformation," *The Atlantic Monthly*, November 1994.

[32] D. Duma and E. Klein, "Generating natural language from Linked Data: Unsupervised template extraction," *Association for Computational Linguistics, Potsdam, Germany*, pp. 83-94, 2013.

[33] C. Dunne and B. Shneiderman, "Motif simplification: improving network visualization readability with fan, connector, and clique glyphs.," in *SIGCHI Conference on Human Factors in Computing Systems*, 2013.

[34] J. Dunne, R. Williams and N. Martinez, "Food-web structure and network theory: The role of connectance and size," *Proceedings of the National Academy of Sciences (PNAS)*, vol. 99, no. 20, pp. 12917-12922, 2002.

[35] D. C. Edelson and D. Gordin, "Visualization for learners: a framework for adapting scientists' tools," *Computers & Geosciences*, pp. 607-616, 1998.

[36] D. Emig, A. Ivliev and O. Pustovalova, "Drug Target Prediction and Repositioning Using an Integrated Network-Based Approach," *PLOS One*, vol. 8, no. 4, p. e60618, 2013.

[37] S. Fakhraei, L. Raschid and L. Getoor, "Drug-target interaction prediction for drug repurposing with probabilistic similarity logic," in *Proceedings of the 12th International Workshop on Data Mining in Bioinformatics*, 2013.

[38] D. F. Farr and A. Y. Rossman, "Fungal Databases, Systematic Mycology and Microbiology Laboratory, ARS, USDA".

[39] X. Feng, J. Zhao and K. Xu, "Link prediction in complex networks: a clustering perspective," *The European Physical Journal B*, vol. 85, no. 1, pp. 1-9, 2012.

[40] G. Flouris and C. Meghini, "Terminology and Wish List for a Formal Theory of Preservation," *The PV 2007 International Conference*, 2007.

[41] N. Franz and R. Peet, "Perspectives: towards a language for mapping relationships among taxonomic concepts," *Systematics and Biodiversity,* vol. 7, no. 1, pp. 5-20, 2009.

[42] T. Franz, A. Schultz, S. Sizov and S. Staab, "Triplerank: Ranking semantic web data by tensor decomposition," in *The 8th International Semantic Web Conference (ISWC2009)*, 2009.

[43] L. Freeman, The development of social network analysis: A study in the sociology of science, Empirical Press, 2004, p. 205.

[44] S. Freeman and R. M. Zink, "A phylogenetic study of the blackbirds based on variation in mitochondrial DNA restriction sites," *Systematic Biology,* vol. 44, no. 3, pp. 409-420, 1995.

[45] K. J. Gaston and J. I. Spicer, Biodiversity: an introduction, John Wiley & Sons, 2013.

[46] S. Goyal and R. Westenthaler, "Rdf gravity (rdf graph visualization tool)," Austria, 2004.

[47] C. Gutierrez, C. Hurtado and A. Vaisman, "Temporal rdf," in *The Semantic Web: Research and Applications*, Springer, 2005, pp. 93-107.

[48] M. Göksedef and Ş. Gündüz-Öğüdücü, "Combination of Web page recommender systems," *Expert Systems with Applications,* vol. 37, no. 4, pp. 2911-2922, 2010.

[49] L. Hamers, Y. Hemeryck and et al., "Similarity measures in scientometric research: the Jaccard index versus Salton's cosine formula.," *Information Processing & Management,* vol. 25, no. 3, pp. 315-318, 1989.

[50] T. Heath and B. Christian, "Linked data: Evolving the web into a global data space," in *Synthesis lectures on the semantic web: theory and technology*, 2011.

[51] J. Hebeler, M. Fisher, R. Blace and A. Perez-Lopez, Semantic web programming, John Wiley & Sons, 2011.

[52] G. Heijden, F. Martin, M. Selosse and I. Sanders, "Mycorrhizal ecology and evolution: the past, the present, and the future.," *New Phytologist*, vol. 205, pp. 1406-1423, 2015.

[53] S. Hellmann, J. Lehmann, S. Auer and M. Brümmer, "Integrating NLP using linked data," in *The 12th International Semantic Web Conference*, 2013.

[54] P. Hitzler, M. Krotzsch and S. Rudolph, Foundations of semantic web technologies, CRC Press, 2009.

[55] D. Hobern, A. Apostolico, E. Arnaud and et al., "Global Biodiversity Informatics Outlook," in *Global Biodiversity Information Facility Secretariat*, 2012.

[56] C. Holsapple, Handbook on knowledge management 1: Knowledge matters, vol. 1, Springer Science & Business Media, 2013.

[57] C. Holsapple and K. Joshi, "Knowledge management: A threefold framework," *The Information Society,* vol. 18, no. 1, pp. 47-64, 2002.

[58] Z. Huang, X. Li and H. Chen, "Link prediction approach to collaborative filtering," in *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, 2005.

[59] S. Ichinose, I. Kobayashi, M. Iwazume and K. Tanaka, "Ranking the Results of DBpedia Retrieval with SPARQL Query," in *The 3rd Joint International Semantic Technology Conference (JIST2013)*, 2013.

[60] H. Inoue, S. Sugi, H. Kuroko and et al., Moths of Japan, vol. 2. Plates and synonymic catalogue, vol. 2, Tokyo: Kodansha Tokyo, 1982.

[61] A. Isabelle, P. Sebastian and R. Sebastian, "Lodifier: Generating linked data from unstructured text," *The 9th Extended Semantic Web Conference (ESWC2012)*, pp. 210-224, 2012.

[62] U. Jinbo, "List-MJ: A checklist of Japanese moths 2004-2008," 2008. [Online]. Available: http://listmj.mothprog.com.

[63] A. C. Jones, R. J. White and E. R. Orme, "Identifying and relating biological concepts in the Catalogue of Life," *Journal of Biomedical Semantics,* vol. 2, no. 1, 2011.

[64] K. Katumoto, "List of fungi recorded in Japan," 2010.

[65] G. Kemmitt, "Early blight of potato and tomato," *The Plant Health Instructor,* pp. 182-203, 2002.

[66] J. B. Kennedy, R. Kukla and T. Paterson, "Scientific names are ambiguous as identifiers for biological taxa: their context and definition are required for accurate data integration," in *Data integration in the life sciences*, Springer, 2005, pp. 80-95.

[67] N. Kertkeidkachorn, R. Ichise, A. Suchato and P. Punyabukkana, "An Automatic Instance Expansion Framework for Mapping Instances to Linked Data Resources," in *Joint International Semantic Technology Conference*, 2013.

[68] Y. Kishida, The Standard of Moths in Japan II, vol. 2, Tokyo: Tokyo: Gakken, 2011.

[69] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of Association for Computing Machinery (JACM),* vol. 46, no. 5, pp. 604-632, 1999.

[70] N. Laurenne, J. Tuominen, H. Saarenmaa and E. Hyvonen, "Making species checklists understandable to machines--a shift from relational databases to ontologies," *Journal of Biomedical Semantics,* vol. 5, no. 1, 2014.

[71] S. Lee and H.-j. Kim, "News keyword extraction for topic tracking," in *The 4th International Conference on Networked Computing and Advanced Information Management (NCM'08)*, 2008.

[72] J. Lehmann, R. Isele, M. Jakob and et al., "DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia," *Semantic Web Journal*, vol. 6, no. 2, pp. 167-195, 2015.

[73] J. Li and K. Zhang, "Keyword extraction based on tf/idf for Chinese news document," in *Wuhan University Journal of Natural Sciences* , 2007.

[74] R. Lichtnwalter and N. V. Chawla, "Link prediction: fair and effective evaluation.," in *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2012.

[75] J. Liebowitz, Knowledge management handbook, CRC Press, 1999.

[76] C. Lin, D. Jiang and A. Zhang, "Prediction of protein function using common-neighbors in protein-protein interaction networks," in *BioInformatics and BioEngineering*, 2006.

[77] G. Linden, B. Smith and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *Internet Computing*, vol. 7, no. 1, pp. 76-80, 2003.

[78] C. Linnaeus, Systema Naturae, 10 ed., vol. 1, Salvii, Holmiae, 1758, p. 824.

[79] H. Liu and H. Motoda, Feature selection for knowledge discovery and data mining, vol. 454, Springer Science & Business Media, 2012.

[80] S.-H. Liu and G.-G. Lee, "Using a concept map knowledge management system to enhance the learning of biology," in *Computers & Education*, 2013.

[81] D. Lowd and P. Domingos, "Naive Bayes models for probability estimation," in *Proceedings of the 22nd international conference on Machine learning*, 2005.

[82] L. Luoqing, "MPGraph: multi-view penalised graph clustering for predicting drug-target interactions.," *Systems Biology*, vol. 8, no. 2, pp. 67-73, 2014.

[83] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 6, pp. 1150-1170, 2011.

[84] J. Mallet, "Species, concepts of," *Encyclopedia of biodiversity*, vol. 5, pp. 427-440, 2001.

[85] B. Mathieu, H. Sebastien and J. Mathieu, "Gephi: an open source software for exploring and manipulating networks," in *The 3rd International AAAI Conference on Web and Social Media (ICWSM 8)*, 2009.

[86] A. K. Menon and C. Elkan, "Link prediction via matrix factorization," in *Machine Learning and Knowledge Discovery in Databases.*, Springer Berlin Heidelberg, 2011, pp. 437-452.

[87] L. Mills, M. Soule and D. Doak, "The Keystone-species concept in ecology and conservation," *BioScience*, vol. 43, pp. 219-224, 1993.

[88] Y. Minami, H. Takeda, F. Kato and et al, "Towards a Data Hub for Biodiversity with LOD," in *The 2nd Joint International Semantic Technology Conference*, 2013.

[89] M. Moslonka-Lefebvre and et al., "Networks in plant epidemiology: from genes to landscapes, countries, and continents," *Phytopathology*, vol. 101, pp. 219-224, 2011.

[90] G. Mueller, G. Bills and M. Foster, Biodiversity of Fungi: Inventory and monitoring methods, Academic Press, 2004, p. 777.

[91] M. E. Newman, "Detecting community structure in networks," *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 38, no. 2, pp. 321-330, 2004.

[92] M. E. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences (PNAS)*, vol. 103, no. 23, pp. 8577-8582, 2006.

[93] I. Nonaka and H. Takeuchi, The knowledge creating company: How Japanese companies create the dynamics of innovation, Oxford university press, 1995.

[94] J. D. Novak and A. J. Cañas, "The theory underlying concept maps and how to construct and use them," in *Florida Institute for Human and Machine Cognition*, 2006.

[95] E. P. Odum, Fundamentals of ecology, 2 ed., Philadelphia: W. B. Saunders, 1959.

[96] R. D. Page, "Taxonomic names, metadata, and the Semantic Web," *Biodiversity Informatics*, vol. 3, 2006.

[97] D. J. Patterson, J. Cooper, P. M. Kirk and et al., "Names are key to the big new biology," *Trends in ecology & evolution*, vol. 25, no. 12, pp. 686-691, 2010.

[98] P. Pons and M. Latapy, "Computing communities in large networks using random walks," *Graph Algorithms Appl*, vol. 10, no. 2, pp. 191-218, 2006.

[99] J. A. Pretorius, J. Jarke and W. Van, "What does the user want to see? What do the data want to be?," *Information Visualization*, vol. 8, no. 3, pp. 153-166, 2009.

[100] R. L. Pyle and E. Michel, "Zoobank: Developing a nomenclatural tool for unifying 250 years of biological information," *Zootaxa*, vol. 1950, pp. 39-50, 2008.

[101] D. Remsen, M. Doring and T. Robertson, "GBIF GNA Profile Reference Guide for Darwin Core Archives, version 1.2," Global Biodiversity Information Facility (GBIF), 2011.

[102] F. Ricci, L. Rokach and B. Shapira, Recommender Systems Handbook, Springer, 2011.

[103] K. Richards, R. White, N. Nicolson and R. Pyle, A beginner’s guide to persistent identifiers, GBIF, 2011.

[104] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal commu-nity structure," *Proceedings of the National Academy of Sciences*, vol. 105, no. 4, pp. 1118-1123, 2008.

[105] G. Salton and B. Christopher, "Term-weighting approaches in automatic text retrieval," *Information processing & management*, vol. 24, no. 5, pp. 513-523, 1988.

[106] I. N. Sarkar, "Biodiversity informatics: organizing and linking information across the spectrum of life," *Briefings in Bioinformatics*, vol. 8, no. 5, pp. 347-357, 2007.

[107] A. I. Schein and A. Popescul, "Methods and Metrics for Cold-Start Recommendations," in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2002.

[108] G. Schreiber and Y. Raimond, "RDF 1.1 Primer," 2014. [Online]. Available: https://www.w3.org/TR/rdf11-primer/.

[109] S. Schulz, H. Stenzhorn and M. Boeker, "The ontology of biological taxa," *Bioinformatics*, vol. 24, no. 13, pp. i313-i321, 2008.

[110] B. A. Schwendimann, "Concept maps as versatile tools to integrate complex ideas: From kindergarten to higher and professional education," *Knowledge Management & E-Learning. Vol. 7. No. 1*, pp. 73-99, 2015.

[111] D. Sedera, " Stakeholder View of Enterprise System Knowledge Management Process," *Proceedings Pacific Asian Conference on Information Systems*, 2007.

[112] U. Shah, T. Finin, A. Joshi, R. S. Cost and J. Matfield, "Information retrieval on the semantic web," *Proceedings of the eleventh international conference on Information and knowledge management*, pp. 461-468, 2002.

[113] P. Shvaiko and J. Euzenat, "Ontology matching: state of the art and future challenges," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 158-176, 2013.

[114] C. G. Sibley and L. L. Short, "Hybridization in the orioles of the Great Plains," *Condor*, pp. 130-150, 1964.

[115] J. Soberón and T. Peterson, "Biodiversity informatics: managing and applying primary biodiversity data," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 359, pp. 689-698, 2004.

[116] F. Suchanek and G. Weikum, "Knowledge harvesting in the big-data era," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, 2013.

[117] T. Sørensen, "A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons," *Journal of Biology,* vol. 5, pp. 1-34, 1948.

[118] J. Tuominen, N. Laurenne and E. Hyvonen, "Biological names and taxonomies on the semantic web: managing the change in scientific conception," in *The Semanic Web: Research and Applications*, Springer, 2011, pp. 255-269.

[119] H. Tuukka, R. Cyganiak and U. Bojars, "Browsing linked data with Fenfire," in *Linked Data on the Web (LDOW)*, 2008.

[120] S. Van Hooland and R. Verborgh, Linked Data for Libraries, Archives and Museums: How to clean, link and publish your metadata, Facet, 2014.

[121] M. Wink and P. Heidrich, "Molecular evolution and systematics of owls (Strigiformes)," *Owls A Guide to the Owls of the World*, pp. 39-57, 1999.

[122] J. E. Winston, Describing species: practical taxonomic procedure for biologists, Columbia University Press, 1999.

[123] J. Wisecaver, J. Slot and A. Rokas, "The evolution of fungal metabolic pathways," *PLOS Genetics,* vol. 10, no. 12, p. e1004816, 2014.

[124] A. K. Wollan, V. Bakkestuen, H. Kauserud, G. Gulden and R. Halvorsen, "Modelling and predicting fungal distribution patterns using herbarium data," *Journal of Biogeography,* vol. 35, no. 12, pp. 2298-2310, 2008.

[125] C. Wright, "Spationtemporal dynamics of prairie wetland networks: power-law scaling and implications for conservation planning," *Ecology,* vol. 91, pp. 1924-1930, 2010.

[126] C. K. Yoon, Naming nature: the clash between instinct and science, London: WW Norton & Company, 2010.

[127] T. Yoshikawa, Y. Isagi and K. Kikuzawa, "Relationships between bird-dispersed plants and avian fruit consumers with different feeding strategies in Japan," *Ecological research,* vol. 24, no. 6, pp. 1301-1311, 2009.

[128] N. Ytow, D. R. Morse and D. M. Roberts, "Nomencurator: a nomenclatural history model to handle multiple taxonomic views," *Biological journal of the Linnean Society,* vol. 73, no. 1, pp. 81-98, 2001.

[129] L. Zemmouchi-Ghomari and A. R. Ghomari, "Translating Natural Language Competency Questions into SPARQLQueries: A Case Study," in *The First International Conference on Building and Exploring Web Based Environments*, 2013.

[130] J. Zhang, Y. L. Liu and Y. Xiao, "Internet knowledge-sharing system based on Object-oriented," *Intelligent Information Technology Application,* vol. 1, 2008.

[131] T. Zhou, "Predicting missing links via local information," *The European Physical Journal B,* vol. 71, no. 4, pp. 623-630, 2009.

[132] International Commission on Zoological Nomenclature: International Code of Zoological Nomenclature, 4 ed., International Trust for Zoological Nomenclatu History Museum, 1999.

[133] "Bibliographic Ontology," [Online]. Available: http://bibliontology.com.

[134] "BioMed Central," [Online]. Available: https://www.biomedcentral.com.

[135] "Data-Driven Documents (D3)," [Online]. Available: http://d3js.org/.

[136] "Open Government Data in The United Kingdom," [Online]. Available: https://data.gov.uk/.

[137] "Dublin Core Metadata Initiative Terms," [Online]. Available: http://dublincore.org/documents/dcmiterms/.

[138] "Darwin Core," [Online]. Available: [http://rs.tdwg.org/dwc/terms/].

[139] "Friend of a Friend," [Online]. Available: http://xmlns.com/foaf/0.1/.

[140] "The Global Biodiversity Information Facility (GBIF)," [Online]. Available: http://www.gbif.org.

[141] "Article 21. Determination of date (online): In International Commission on Zoological Nomenclature (ICZN)," [Online]. Available: http://www.nhm.ac.uk/hosted-sites/iczn/code/index.jsp?article=21.

[142] "Interaction Web DataBase (IWDB)," [Online]. Available: https://www.nceas.ucsb.edu/interactionweb/.

[143] "Apache Jena," [Online]. Available: http://jena.apache.org/.

[144] "National Museum of Nature and Science," [Online]. Available: http://www.kahaku.go.jp/english.

[145] "Linked Open Data," [Online]. Available: http://linkeddata.org/.

[146] "Linked Open Data for ACademia," [Online]. Available: http://lod.ac.

[147] "Linked Open Data Initiative," [Online]. Available: http://linkedopendata.jp/.

[148] "Open Government Data in Thailand," [Online]. Available: http://data.go.th/.

[149] "Sesame Framework," [Online]. Available: http://rdf4j.org/.

[150] "Simple Knowledge Organization System," [Online]. Available: http://www.w3.org/TR/skos-primer/.

[151] "Semantically-Interlinked Online Communities Core Specification," [Online]. Available: http://www.w3.org/Submission/sioc-spec/.

[152] "SPARQL Query Language for RDF," 2008. [Online]. Available: https://www.w3.org/TR/rdf-sparql-query/.

[153] "Biodiversity Information Standards (TDWG)," [Online]. Available: http://www.tdwg.org.

[154] "Taxonomic Concept Schema Complementary Documentation for Draft Standard," [Online]. Available: http://wiki.tdwg.org/twiki/pub/TNC/EarlyDiscussionOnRelationshipTypes/tdwg_tcs.doc.

[155] "Taxonomic Names and Concepts Interest Group: Taxonomic concept transfer schema," 2005. [Online]. Available: http://www.tdwg.org/standards/117/.

[156] "Describing Taxon Concepts as RDF (draft)," [Online]. Available: http://rs.tdwg.org/dwc/terms/.

[157] "The Timeline Ontology," [Online]. Available: http://motools.sourceforge.net/timeline/.

[158] "Plant Trait Database (TRY)," [Online]. Available: https://www.try-db.org/TryWeb/Home.php.

[159] "National Center for Biotechnology Information (NCBI) - Taxonomy," [Online]. Available: http://www.ncbi.nlm.nih.gov/taxonomy.

.....

APPENDIX: LTK FRAMEWORK

This appendix describes the LTK framework in more detail by giving more information about namespaces, classes, properties, and the uses of operations.

Namespaces

```
@prefix ltk:    <http://rc.lodac.nii.ac.jp/ns/ltk#> .
@prefix rdfs:   <http://www.w3.org/2000/01/rdf-schema#> .
@prefix lodac:  <http://lod.ac/ns/species#> .
@prefix dct:    <http://rs.tdwg.org/dwc/terms/> .
@prefix owl:    <http://www.w3.org/2002/07/owl#> .
@prefix xsd:    <http://www.w3.org/2001/XMLSchema#> .
@prefix rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix cka:    <http://www.cka.org/2012/01/cka-onto#> .
@prefix skos:   <http://www.w3.org/2004/02/skos/core#> .
@prefix tmo:    <http://www.yso.fi/onto/taxmeon/> .
```

Classes

Taxonomic Entities

Nominal Entity

```
ltk:NominalEntity     rdf:type   owl:Class .
```

Simple Nominal Entity

```
ltk:SimpleNominalEntity    rdf:type   owl:Class ;
     rdfs:subClassOf  ltk:NominalEntity .
```

Contextual Nominal Entity

```
ltk:ContextualNorminalEntity     rdf:type   owl:Class ;
     rdfs:subClassOf  ltk:NominalEntity .
```

Taxonomic Operations

```
ltk:TaxonOperation    rdf:type   owl:Class ;
cka:ConceptEvolution  rdf:type   ltk:TaxonOperation .
cka:RelationEvolution rdf:type   ltk:TaxonOperation .
```

Operation of Change in Conception

Taxon Replacement

```
ltk:TaxonReplacement
      rdfs:subClassOf            cka:ConceptEvolution ;
      ltk:linkingProperty        ltk:replacedTo ;

      rdfs:subClassOf [
            rdf:type             owl:Restriction ;
            owl:onProperty       ltk:taxonBefore ;
            owl:cardinality      1
      ] ;

      rdfs:subClassOf [
            rdf:type             owl:Restriction ;
            owl:onProperty       ltk:taxontAfter ;
            owl:cardinality      1
      ] .
```

Taxon Merger

```
ltk:TaxonMerger
      rdfs:subClassOf            cka:ConceptEvolution ;
      ltk:linkingProperty        ltk:mergedInto ;
      ltk:majorLink              ltk:majorMergedInto ;

      rdfs:subClassOf [
            rdf:type             owl:Restriction ;
            owl:onProperty       ltk:taxonBefore ;
            owl:minCardinality   2
      ] ;

      rdfs:subClassOf [
            rdf:type             owl:Restriction ;
            owl:onProperty       ltk:majorTaxonBefore ;
            owl:cardinality      1
      ] ;

      rdfs:subClassOf [
            rdf:type             owl:Restriction ;
            owl:onProperty       ltk:taxonAfter ;
            owl:cardinality      1
      ] .
```

Taxon Splitter

```
ltk:TaxonSplitter
      rdfs:subClassOf            cka:ConceptEvolution ;
      ltk:linkingProperty        ltk:splitInto ;
      ltk:majorLink              ltk:majorSplitInto ;

      rdfs:subClassOf [
            rdf:type             owl:Restriction ;
            owl:onProperty       ltk:taxonBefore ;
```

```
            owl:cardinality       1
        ] ;

      rdfs:subClassOf [
            rdf:type              owl:Restriction ;
            owl:onProperty        ltk:taxonAfter ;
            owl:minCardinality    2
        ] ;

      rdfs:subClassOf [
            rdf:type              owl:Restriction ;
            owl:onProperty        ltk:majorTaxonAfter ;
            owl:cardinality       1
      ] .
```

Circumscription Change

```
ltk:CircumscriptionChange
      rdfs:subClassOf       cka:ConceptEvolution ;
      ltk:linkingProperty         ltk:circChangedTo ;

      rdfs:subClassOf [
            rdf:type              owl:Restriction ;
            owl:onProperty        ltk:taxonBefore ;
            owl:cardinality       1
      ] ;

      rdfs:subClassOf [
            rdf:type              owl:Restriction ;
            owl:onProperty        ltk:taxontAfter ;
            owl:cardinality       1
      ] .
```

Taxon Complex Change

```
ltk:TaxonComplexChange
      rdfs:subClassOf             cka:ConceptEvolution ;
      ltk:linkingProperty         ltk:cpxChangedTo ;

      rdfs:subClassOf [
            rdf:type              owl:Restriction ;
            owl:onProperty        ltk:taxonBefore ;
            owl:minCardinality    2
      ] ;

      rdfs:subClassOf [
            rdf:type              owl:Restriction ;
            owl:onProperty        ltk:taxonAfter ;
            owl:minCardinality    2
      ] .
```

Related Properties

```
ltk:linkingProperty
      rdf:type              owl:ObjectProperty ;
      rdfs:domain           cka:ConceptEvolution ;
      rdfs:range            owl:ObjectProperty .

ltk:majorLink
      rdf:type              owl:ObjectProperty ;
      rdfs:domain           cka:ConceptEvolution ;
      rdfs:range            owl:ObjectProperty .

ltk:taxonBefore
      rdf:type              owl:ObjectProperty ;
      rdfs:subPropertyOf    cka:conceptAfter ;
      rdfs:domain           cka:ConceptEvolution ;
      rdfs:range            ltk:ContextualNominalEntity .

ltk:majorTaxonBefore
      rdf:type              owl:ObjectProperty ;
      rdfs:subPropertyOf    ltk:taxonBefore ;
      rdfs:domain           cka:ConceptEvolution ;
      rdfs:range            ltk:ContextualNominalEntity .

ltk:taxonAfter
      rdf:type              owl:ObjectProperty ;
      rdfs:subPropertyOf    cka:taxonAfter ;
      rdfs:domain           cka:ConceptEvolution ;
      rdfs:range            ltk:ContextualNominalEntity .

ltk:majorTaxonAfter
      rdf:type              owl:ObjectProperty ;
      rdfs:subPropertyOf    ltk:taxonAfter ;
      rdfs:domain           cka:ConceptEvolution ;
      rdfs:range            ltk:ContextualNominalEntity .
```

Operation of Change in Relation between Taxa

Change Higher Taxon

```
ltk:ChangeHigherTaxon
      rdfs:subClassOf       cka:RelationEvolution ;
      ltk:relation          ltk:synonym .
```

Subdivide Taxon

```
ltk:SubdivideTaxon
      rdfs:subClassOf       cka:RelationEvolution ;
      ltk:relation          ltk:subdividedInto .
```

Combine Taxa

```
ltk:CombineTaxa
      rdfs:subClassOf       cka:RelationEvolution ;
      ltk:relation          ltk:combinedInto .
```

Synonym Link

```
ltk:SynonymLink
      rdfs:subClassOf       cka:RelationEvolution ;
      ltk:relation          ltk:synonym .
```

Senior Synonym Link

```
ltk:SeniorSynonymLink
      rdfs:subClassOf       cka:RelationEvolution ;
      ltk:relation          ltk:seniorSynonym .
```

Directed Synonym Link

```
ltk:DirectedSynonymLink
      rdfs:subClassOf       cka:RelationEvolution ;
      ltk:relation          ltk:dSynonym .
```

Homonym Link

```
ltk:HomonymLink
      rdfs:subClassOf       cka:RelationEvolution ;
      ltk:relation          ltk:homonym .
```

Related Properties

```
ltk:subjectTaxon
      rdf:type              owl:ObjectProperty ;
      rdfs:subPropertyOf    cka:subject ;
      rdfs:domain           cka:ConceptEvolution ;
      rdfs:range            ltk:NominalEntity .

ltk:objectTaxonBefore
      rdf:type              owl:ObjectProperty ;
      rdfs:subPropertyOf    cka:objectBefore ;
      rdfs:domain           cka:ConceptEvolution ;
      rdfs:range            ltk:NominalEntity .

ltk:objectTaxonAfter
      rdf:type              owl:ObjectProperty ;
      rdfs:subPropertyOf    cka:objectAfter ;
      rdfs:domain           cka:ConceptEvolution ;
      rdfs:range            ltk:NominalEntity .

ltk:child
      owl:equivalentProperty     ltk:subjectTaxon  .

ltk:sourceTaxon
      owl:equivalentProperty     ltk:subjectTaxon  .

ltk:parentBefore
      owl:equivalentProperty     ltk: objectTaxonBefore .

ltk:higherTaxonBefore
      owl:equivalentProperty     ltk: objectTaxonBefore .
```

```
ltk:parentAfter
     owl:equivalentProperty     ltk: objectTaxonAfter .

ltk:higherTaxonAfter
     owl:equivalentProperty     ltk: objectTaxonAfter .

ltk:targetTaxon
     owl:equivalentProperty     ltk: objectTaxonAfter .
```

Properties

Event-Centric Model

```
cka:assures
     rdf:type              owl:ObjectProperty ;
     rdfs:domain           cka:CommunityKnowledge ;
     rdfs:range            ltk:TaxonOperation .

tl:interval
     rdf:type              owl:ObjectProperty ;
     rdfs:domain           cka:CommunityKnowledge ;
     rdfs:range            tl:Interval .

tl:beginsAtDateTime
     rdf:type              owl:DataTypeProperty ;
     rdfs:domain           tl:Interval ;
     rdfs:range            xsd:dateTime .

tl:endsAtDateTime
     rdf:type              owl:DataTypeProperty ;
     rdfs:domain           tl:Interval ;
     rdfs:range            xsd:dateTime .

bibo:performer
     rdf:type              owl:ObjectProperty ;
     rdfs:domain           cka:CommunityKnowledge ;
     rdfs:range            foaf:Person .

bibo:issuer
     rdf:type              owl:ObjectProperty ;
     rdfs:domain           cka:CommunityKnowledge ;
     rdfs:range            foaf:Person .

dct:source
     rdf:type              owl:ObjectProperty ;
     rdfs:domain           cka:CommunityKnowledge ;
     rdfs:range            foaf:Docuement .

ltk:effect
     rdf:type              owl:ObjectProperty ;
     rdfs:domain           ltk:TaxonOperation ;
```

```
      rdfs:range              ltk:TaxonOperation .

ltk:cause
      owl:inverseOf           ltk:effect .

ltk:detail
      rdf:type                owl:ObjectProperty ;
      rdfs:domain             ltk:TaxonOperation ;
      rdfs:range              ltk:TaxonOperation .
```

Transition Model

```
ltk:majorMergedInto
      rdfs:subPropertyOf      ltk:mergedInto ,
                              skos:closeMatch ;
      rdfs:domain             ltk:ContextualNominalEntity ;
      rdfs:range              ltk:ContextualNominalEntity .

ltk:majorSplitInto
      rdfs:subPropertyOf      ltk:splitInto ,
                              skos:closeMatch ;
      rdfs:domain             ltk:ContextualNominalEntity ;
      rdfs:range              ltk:ContextualNominalEntity .

ltk:mergedInto
      rdfs:subPropertyOf      cka:serialLinkTo ,
                              skos:broadMatch ;
      rdfs:domain             ltk:ContextualNominalEntity ;
      rdfs:range              ltk:ContextualNominalEntity .

ltk:splitInto
      rdfs:subPropertyOf      cka:serialLinkTo ,
                              skos:narrowMatch ;
      rdfs:domain             ltk:ContextualNominalEntity ;
      rdfs:range              ltk:ContextualNominalEntity .

ltk:replacedTo
      rdfs:subPropertyOf      cka:serialLinkTo ,
                              tmo:congruentWithTaxon ,
                              skos:exactMatch ;
      rdfs:domain             ltk:ContextualNominalEntity ;
      rdfs:range              ltk:ContextualNominalEntity .

ltk:circChangedTo
      rdfs:subPropertyOf      cka:serialLinkTo ,
                              skos:closeMatch ;
      rdfs:domain             ltk:ContextualNominalEntity ;
      rdfs:range              ltk:ContextualNominalEntity .

ltk:cpxChangedTo
      rdfs:subPropertyOf      cka:serialLinkTo ,
                              skos:relatedMatch ;
      rdfs:domain             ltk:ContextualNominalEntity ;
      rdfs:range              ltk:ContextualNominalEntity .
```

```
ltk:serialLinkTo
     rdf:type              owl:TransitiveProperty ;
     rdfs:subPropertyOf    cka:semanticLink ;

ltk:semanticLink
     rdf:type              owl:TransitiveProperty
                           owl:SymmetricProperty .
```

Snapshot Model

```
ltk:higherTaxon
     rdfs:subPropertyOf    skos:broaderTransitive ,
                           tmo:isPartOfHigherTaxon ,
                           lodac:hasSuperTaxon ;
     rdfs:domain           ltk:NominalEntity ;
     rdfs:range            ltk:NominalEntity .

ltk:lowerTaxon
     owl:inverseOf         ltk:higherTaxon .

ltk:subdividedInto
     rdfs:subPropertyOf    skos:narrowMatch ;
     rdfs:domain           ltk:NominalEntity ;
     rdfs:range            ltk:NominalEntity .

ltk:combinedInto
     rdfs:subPropertyOf    skos:broadMatch ;
     rdfs:domain           ltk:NominalEntity ;
     rdfs:range            ltk:NominalEntity .

ltk:dsynonym
     rdfs:subPropertyOf    skos:exactMatch ,
                           lodac:hasSynonym ;
     rdfs:domain           ltk:NominalEntity ;
     rdfs:range            ltk:NominalEntity .

ltk:synonym
     rdf:type              owl:SymmetricProperty ;
     rdfs:subPropertyOf    skos:exactMatch ,
                           ltk:dsynonym ,
                           lodac:hasSynonym ;
     rdfs:domain           ltk:NominalEntity ;
     rdfs:range            ltk:NominalEntity .

ltk:seniorSynonym
     rdfs:subPropertyOf    skos:exactMatch ,
                           ltk:synonym ,
                           lodac:hasSynonym ;
     rdfs:domain           ltk:NominalEntity ;
     rdfs:range            ltk:NominalEntity .

ltk:juniorSynonym
     owl:inverseOf         ltk:seniorSynonym.
```

```
ltk:homonym
      rdfs:domain           ltk:NominalEntity ;
      rdfs:range            ltk:NominalEntity .
```

The uses of LTK Operations

The following list declared operations and their parameters, which are provided by LTK ontology. An italic symbol in the parentheses of each parameter indicates its cardinality for every operation. The symbol "*(1)*" allows only one value, the symbol "*(2..*)*" expects at least two values required, and the symbol "*(0..1)*" presents one optional value.

Operation of Change in Conception

ltk:TaxonMerger

Description	For merging some taxa (before) into one taxon (after).
Parameters	- ltk:taxonBefore *(2..*)* - ltk:majorTaxonBefore *(0..1)* - ltk:taxonAfter *(1)*
Example input RDF	ex:opr rdf:type ltk:TaxonMerger . ex:opr ltk:taxonBefore ex:be1, ex:be2 ; ltk:majorTaxonBefore ex:mb0 ; ltk:taxonAfter ex:af1 .
Example result	ex:be1 ltk:mergedInto ex:af1 . ex:be2 ltk:mergedInto ex:af1 . ex:mb0 ltk:majorMergedInto ex:af1 .
Example entailment	ex:be1 skos:broadMatch ex:af1 . ex:be2 skos:broadMatch ex:af1 . ex:mb0 skos:closeMatch ex:af1 .

ltk:TaxonSplitter

Description	For splitting a taxon (before) into new taxa (after).
Parameters	- ltk:taxonBefore *(1)* - ltk:taxonAfter *(2..*)* - ltk:majorTaxonAfter *(0..1)*
Example input RDF	ex:opr rdf:type ltk:TaxonSplitter . ex:opr ltk:taxonBefore ex:be1 ; ltk:taxonAfter ex:af1, ex:af2 ; ltk:majorTaxonAfter ex:ma0 .
Example result	ex:be1 ltk:splitInto ex:af1 . ex:be1 ltk:splitInto ex:af2 . ex:be1 ltk:majorSplitInto ex:ma0 .
Example entailment	ex:be1 skos:narrowMatch ex:af1 . ex:be1 skos:narrowMatch ex:af2 . ex:be1 skos:closeMatch ex:ma0 .

ltk:TaxonReplacement

Description	For replacing one taxon (before) to another one taxon (after).
Parameters	- ltk:taxonBefore (1) - ltk:taxonAfter (1)
Example input RDF	ex:opr rdf:type ltk:TaxonReplacement . ex:opr ltk:taxonBefore ex:be1 ; ltk:taxonAfter ex:af1 .
Example result	ex:be1 ltk:replacedTo ex:af1 .
Example entailment	ex:be1 skos:exactMatch ex:af1 . ex:be1 tmo:congruentWithTaxon ex:af1 .

ltk:TaxonComplexChange

Description	For a complex case that many taxa (before) are merged and split into many other taxa (after).
Parameters	- ltk:taxonBefore (2..*) - ltk:taxonAfter (2..*)
Example input RDF	ex:opr rdf:type ltk:TaxonComplexChange . ex:opr ltk:taxonBefore ex:be1, ex:be2 ; ltk:taxonAfter ex:af1, ex:af2 .
Example result	ex:be1 ltk:cpxChangedTo ex:af1 . ex:be1 ltk:cpxChangedTo ex:af2 . ex:be2 ltk:cpxChangedTo ex:af1 . ex:be2 ltk:cpxChangedTo ex:af2 .
Example entailment	ex:be1 skos:relatedMatch ex:af1 . ex:be1 skos:relatedMatch ex:af2 . ex:be2 skos:relatedMatch ex:af1 . ex:be2 skos:relatedMatch ex:af2 .

ltk:CircumscriptionChange

Description	For changing circumscription of one taxon (before) to another one taxon (after).
Parameters	- ltk:taxonBefore (1) - ltk:taxonAfter (1)
Example input RDF	ex:opr rdf:type ltk:CircumscriptionChange. ex:opr ltk:taxonBefore ex:be1 ; ltk:taxonAfter ex:af1 .
Example result	ex:be1 ltk:circChangedTo ex:af1 .
Example entailment	ex:be1 skos:closeMatch ex:af1 .

Operation of Change in Relation between Taxa

ltk:ChangeHigherTaxon

Description	For reclassifying a lower taxon (child) by moving from a higher taxon (before) to another higher taxon (after).
Parameters	- `ltk:child` (1) - `ltk:parentBefore` (1) - `ltk:parentAfter` (1)

Example input RDF	`ex:opr rdf:type ltk:ChangeHigherTaxon .`
	`ex:opr ltk:child ex:c1 ;`
	`ltk:parentBefore ex:p1 ;`
	`ltk:parentAfter ex:p2 .`

Example result	`ex:c1 ltk:higherTaxon ex:p2 .` `ex:p2 ltk:lowerTaxon ex:c1 .`

Example entailment	`ex:c1 skos:broaderTransitive ex:p2 .`
	`ex:p2 skos:narrowerTransitive ex:c1 .`
	`ex:c1 lodac:hasSuperTaxon ex:p2 .`

ltk:SubdivideTaxon

Description	For subdividing a higher taxon (source) into some lower taxa (target).
Parameters	- `ltk:sourceTaxon` (1) - `ltk:targetTaxon` (2..*)

Example input RDF	`ex:opr rdf:type ltk:SubdivideTaxon .`
	`ex:opr ltk:sourceTaxon ex:h1 ;`
	`ltk:targetTaxon ex:c1, ex:c2 .`

Example result	`ex:h1 ltk:subdividedInto ex:c1 .` `ex:h1 ltk:subdividedInto ex:c2 .`
Example entailment	`ex:h1 skos:narrowMatch ex:c1 .` `ex:h1 skos:narrowMatch ex:c2 .`

ltk:CombineTaxa

Description	For combining lower taxa (source) into a higher taxon (target).
Parameters	- `ltk:sourceTaxon` (2..*) - `ltk:targetTaxon` (1)

Example input RDF	`ex:opr rdf:type ltk:CombineTaxa .`
	`ex:opr ltk:sourceTaxon ex:c1 , ex:c2 ;`
	`ltk:targetTaxon ex:h1 .`

Example result	`ex:c1 ltk:combindedInto ex:h1 .` `ex:c2 ltk:combindedInto ex:h1 .`
Example entailment	`ex:c1 skos:broadMatch ex:h1 .` `ex:c2 skos:broadMatch ex:h1 .`

ltk:HomonymLink

Description	For identifying a homonym (target) of a taxon (source).
Parameters	- ltk:sourceTaxon (1) - ltk:targetTaxon (1)
Example input RDF	ex:opr rdf:type ltk:HomonymLink . ex:opr ltk:sourceTaxon ex:c1 ; ltk:targetTaxon ex:c2 .
Example result	ex:c1 ltk:homonym ex:c2 . ex:c2 ltk:homonym ex:c1 .

ltk:DirectSynonymLink

Description	For identifying a synonym (target) of a taxon (source). It is a directional synonym, which is always used in botany.
Parameters	- ltk:sourceTaxon (1) - ltk:targetTaxon (1)
Example input RDF	ex:opr rdf:type ltk:DirectSynonymLink . ex:opr ltk:sourceTaxon ex:c1 ; ltk:targetTaxon ex:c2 .
Example result	ex:c1 ltk:dsynonym ex:c2 .
Example entailment	ex:c1 skos:exactMatch ex:c2 . ex:c2 skos:exactMatch ex:c1 . ex:c1 lodac:hasSynonym ex:c2 .

ltk:SynonymLink

Description	For identifying a synonym (target) of a taxon (source). It is a bidirectional synonym, which is generally used in many domains especially in zoology.
Parameters	- ltk:sourceTaxon (1) - ltk:targetTaxon (1)
Example input RDF	ex:opr rdf:type ltk:SynonymLink . ex:opr ltk:sourceTaxon ex:c1 ; ltk:targetTaxon ex:c2 .
Example result	ex:c1 ltk:synonym ex:c2 .
Example entailment	ex:c2 ltk:synonym ex:c1 . ex:c1 ltk:dsynonym ex:c2 . ex:c2 ltk:dsynonym ex:c1 . ex:c1 skos:exactMatch ex:c2 . ex:c2 skos:exactMatch ex:c1 . ex:c1 lodac:hasSynonym ex:c2 . ex:c2 lodac:hasSynonym ex:c1 .

ltk:SeniorSynonymLink

Description	For identifying a senior synonym (target) of a taxon (source).
Parameters	- ltk:sourceTaxon (1) - ltk:targetTaxon (1)
Example input RDF	ex:opr rdf:type ltk:SeniorSynonymLink. ex:opr ltk:sourceTaxon ex:c1 ; ltk:targetTaxon ex:c2 .
Example result	ex:c1 ltk:seniorSynonym ex:c2 . ex:c2 ltk:juniorSynonym ex:c1 .
Example entailment	ex:c1 ltk:synonym ex:c2 . ex:c2 ltk:synonym ex:c1 . ex:c1 skos:exactMatch ex:c2 . ex:c2 skos:exactMatch ex:c1 . ex:c1 lodac:hasSynonym ex:c2 . ex:c2 lodac:hasSynonym ex:c1 .

.....

wish our effort inspires your passion