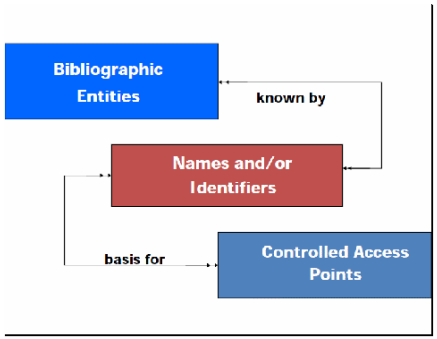
folio
future of libraries is open

Quintuplet as a Base for Flexible Data Formats Hungarian National Library Platform

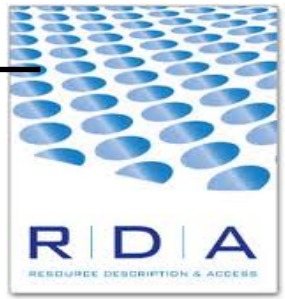
Miklós Lendvay
Hungarian National Széchényi Library

3rd EODOPEN Project Meeting 5th – 6th November 2020, online

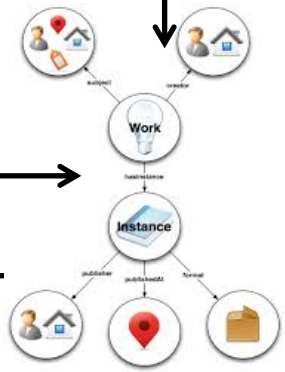
Identifying and Linking of Data



Functional Requirements for Authority Data

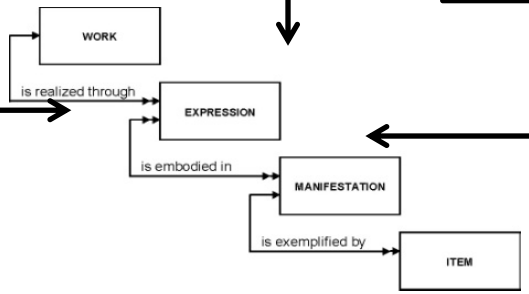


Resource Description and Access

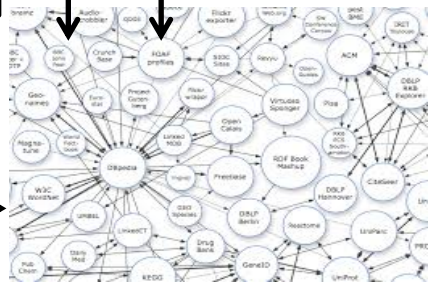


BIBFRAME

International Cataloguing Principles

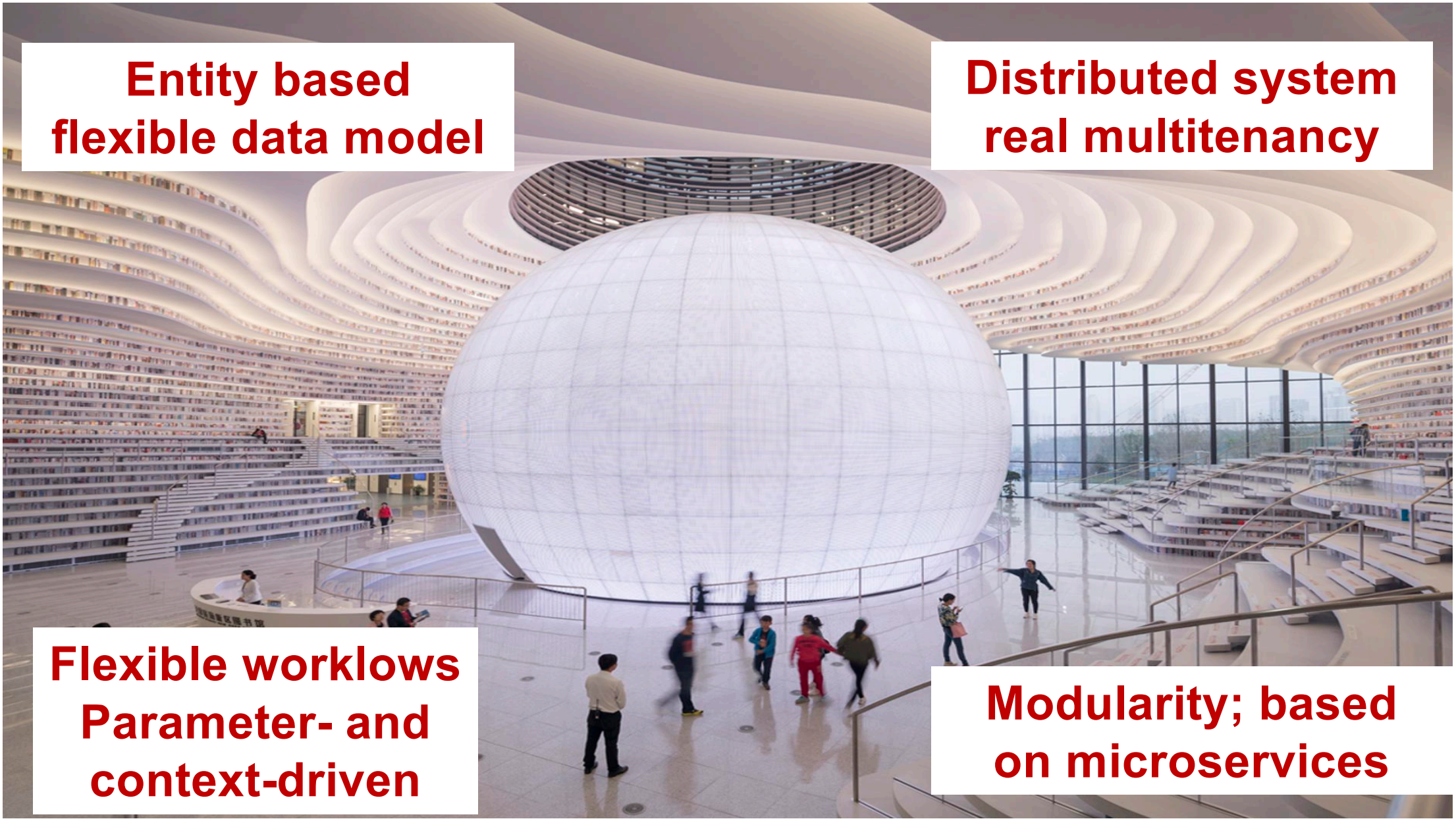


Functional Requirements for Bibliographic Records

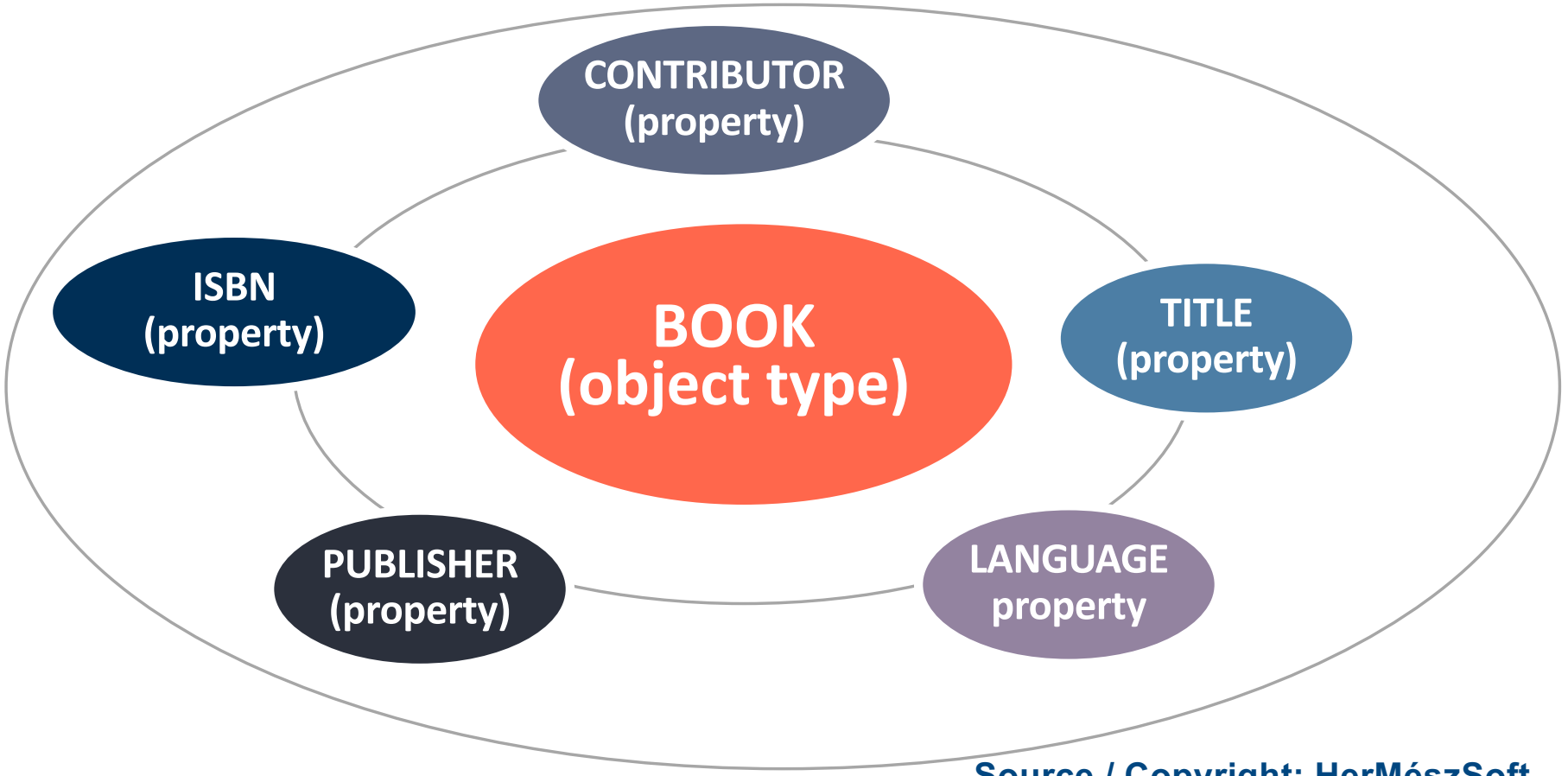


Semantic web / Linked data

Source / Copyright: Tiziana Possemato, @cult


Entity based flexible data model
Distributed system real multitenancy
Flexible worklows Parameter- and context-driven
Modularity; based on microservices

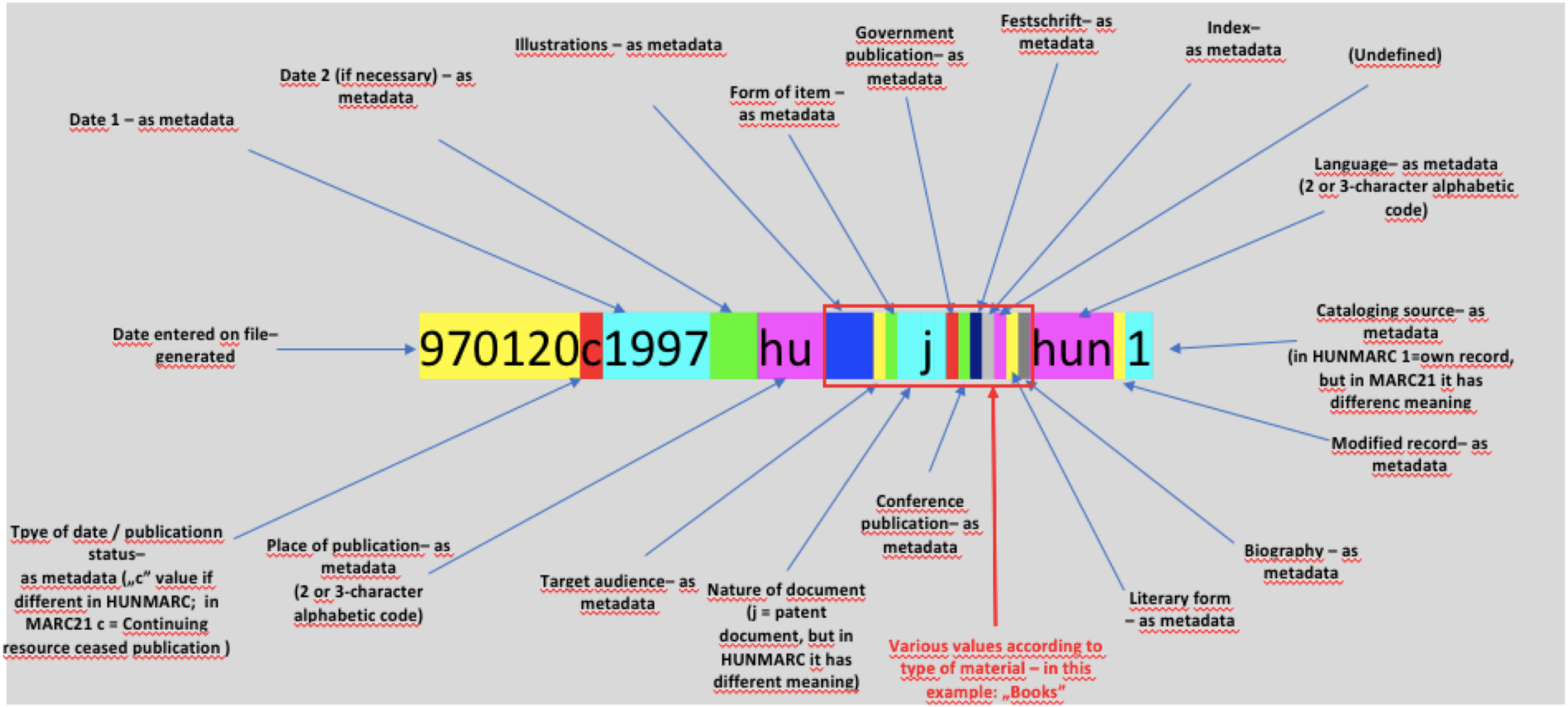
The Base for a Flexible Cataloging: the Schema Definition

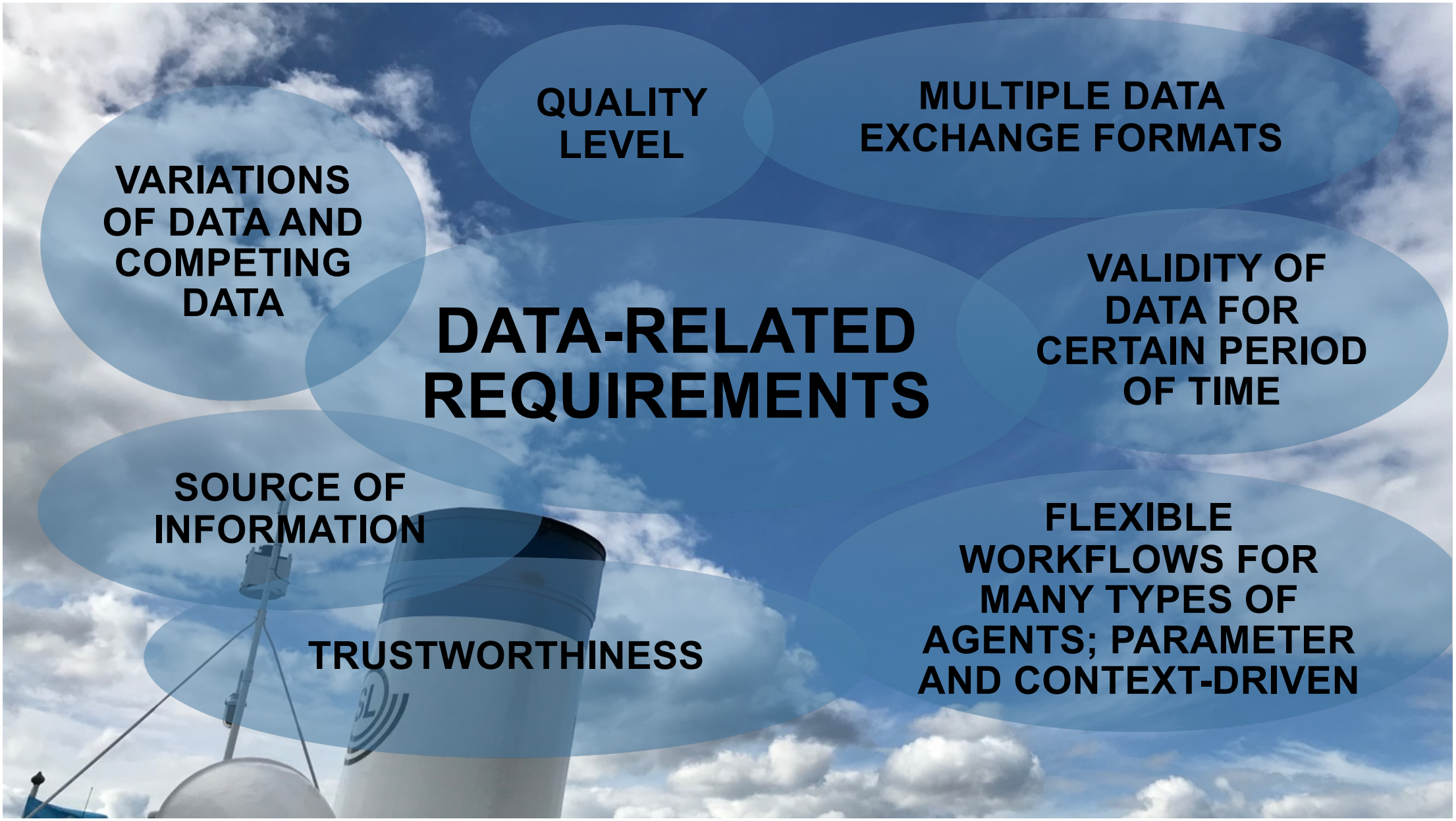


Source / Copyright: HerMészSoft

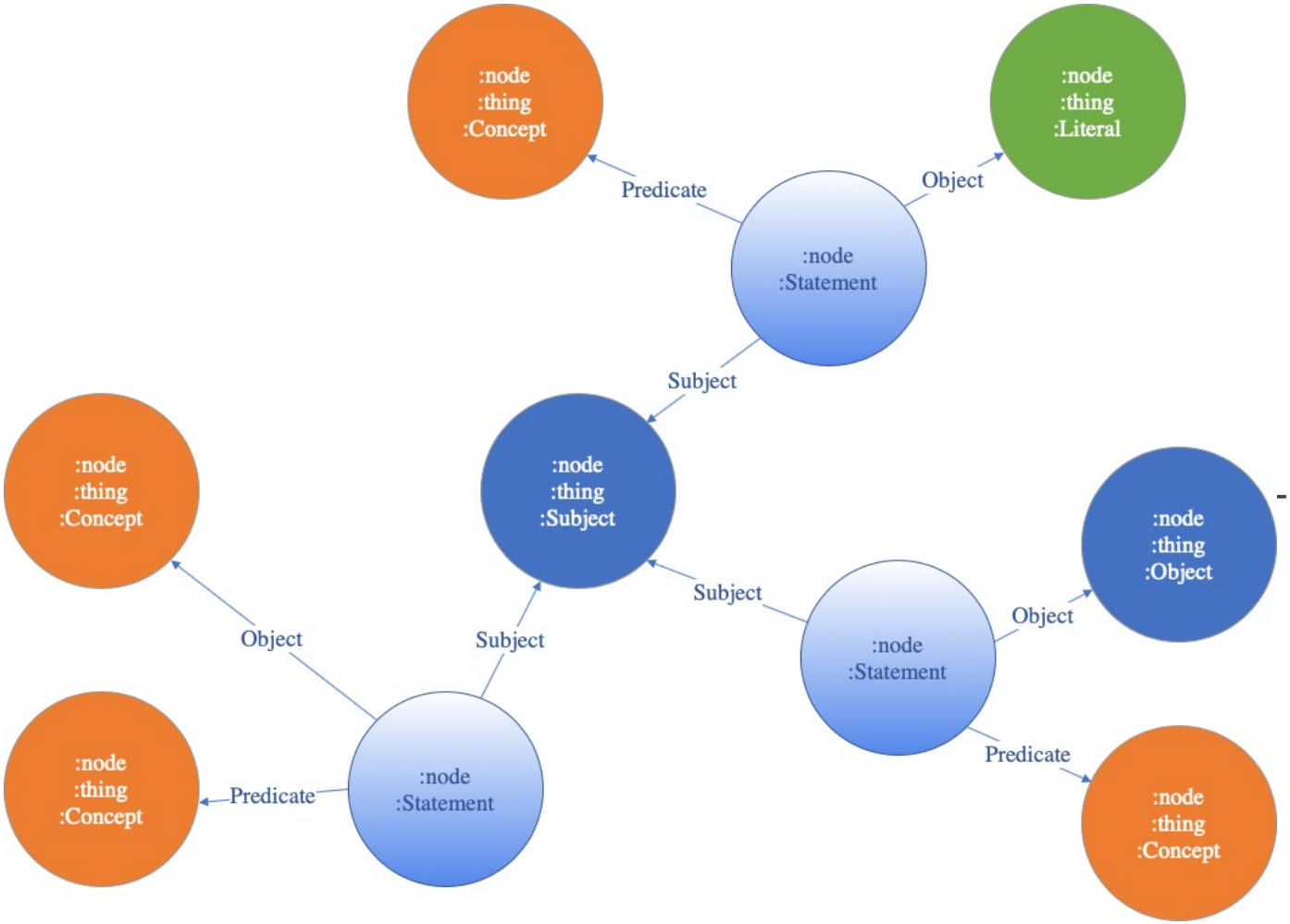
MARC21 008 – Fixed Lenght Data Elements

Example: mek.oszk.hu – Jókai Mór: Az arany ember MARC21 record (URL: http://mek.oszk.hu/00600/00688/usmarc.html)




QUALITY LEVEL
MULTIPLE DATA EXCHANGE FORMATS
VARIATIONS OF DATA AND COMPETING DATA
DATA-RELATED REQUIREMENTS
VALIDITY OF DATA FOR CERTAIN PERIOD OF TIME
SOURCE OF INFORMATION
TRUSTWORTHINESS
FLEXIBLE WORKFLOWS FOR MANY TYPES OF AGENTS; PARAMETER AND CONTEXT-DRIVEN

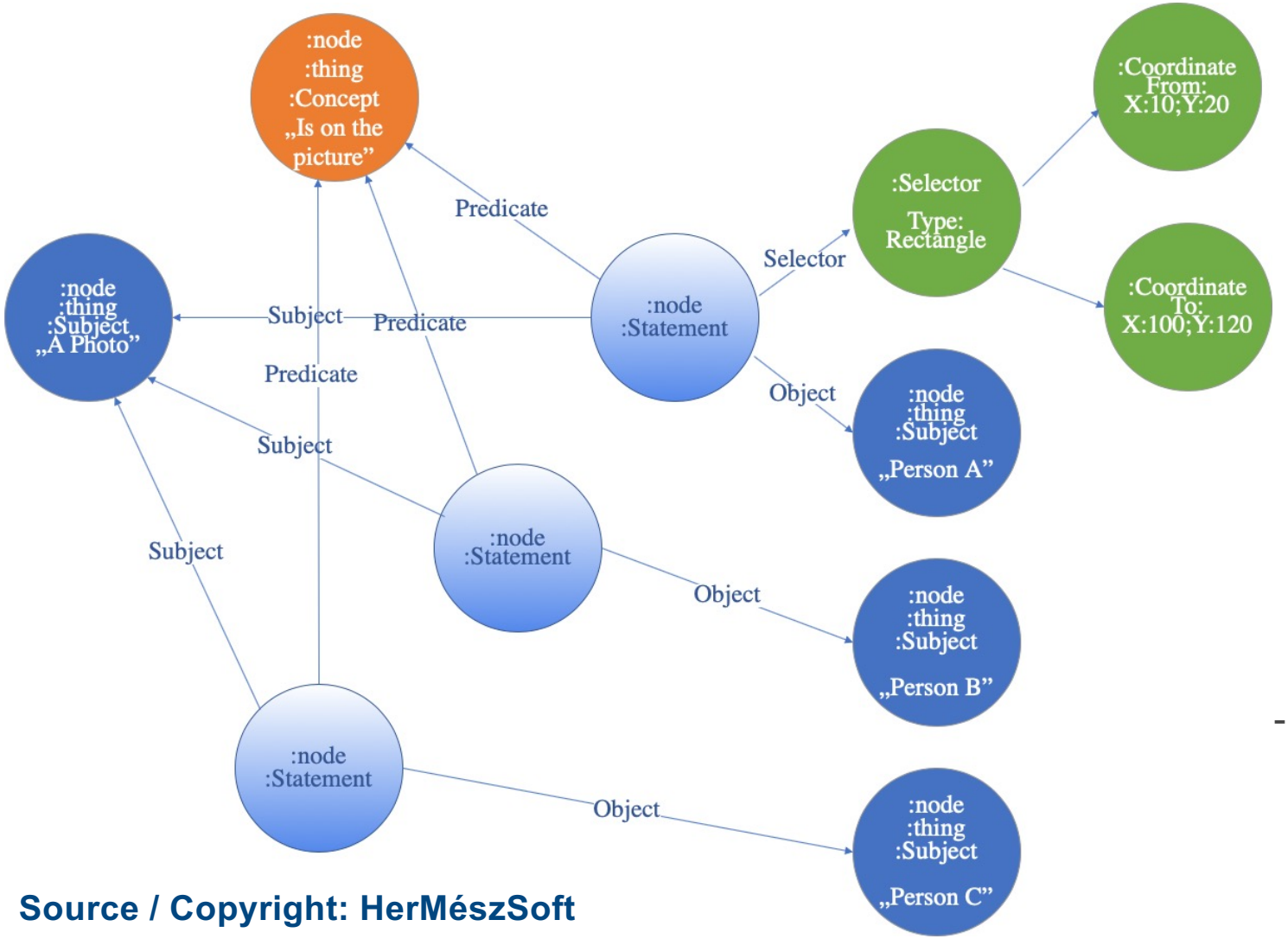
Generic ‘dynamically expandable value set’ knowledge graph



- When creating a triplet, the predicate is not stored as the quality of the relationship between the records, instead the predicate is built into the relationship chain as a record.
- The common point of relationships is the statement that is able to make a piece of elementary statement about a given subject.
- The object of the statement may be another object, literal value, ‘itemized’ literal value.

Source / Copyright: HerMészSoft

The Stucture of General Statements

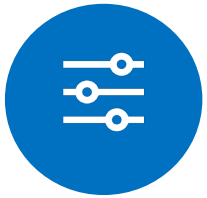


- The “triplet” is used to define elementary statements
- To add more specific data, statements must be made about a statement
- All statements are equally true until we make a “false” statement about that statement
- The statement “tree” can be branched to infinity
- The framework does not provide guidance on how to deal with competing statements

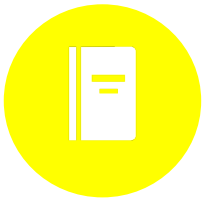
Source / Copyright: HerMészSoft

Anatomy of Statements:
The Structure of a Quintuplet

SUBJECT:
THE SUBJECT IS THE DOCUMENT TO WHICH THE STATEMENT APPLIES



SELECTOR:
THE POSITION OF THE STATEMENT ALONGSIDE THE DIMENSIONS OF THE DOCUMENT TYPE OF THE SUBJECT

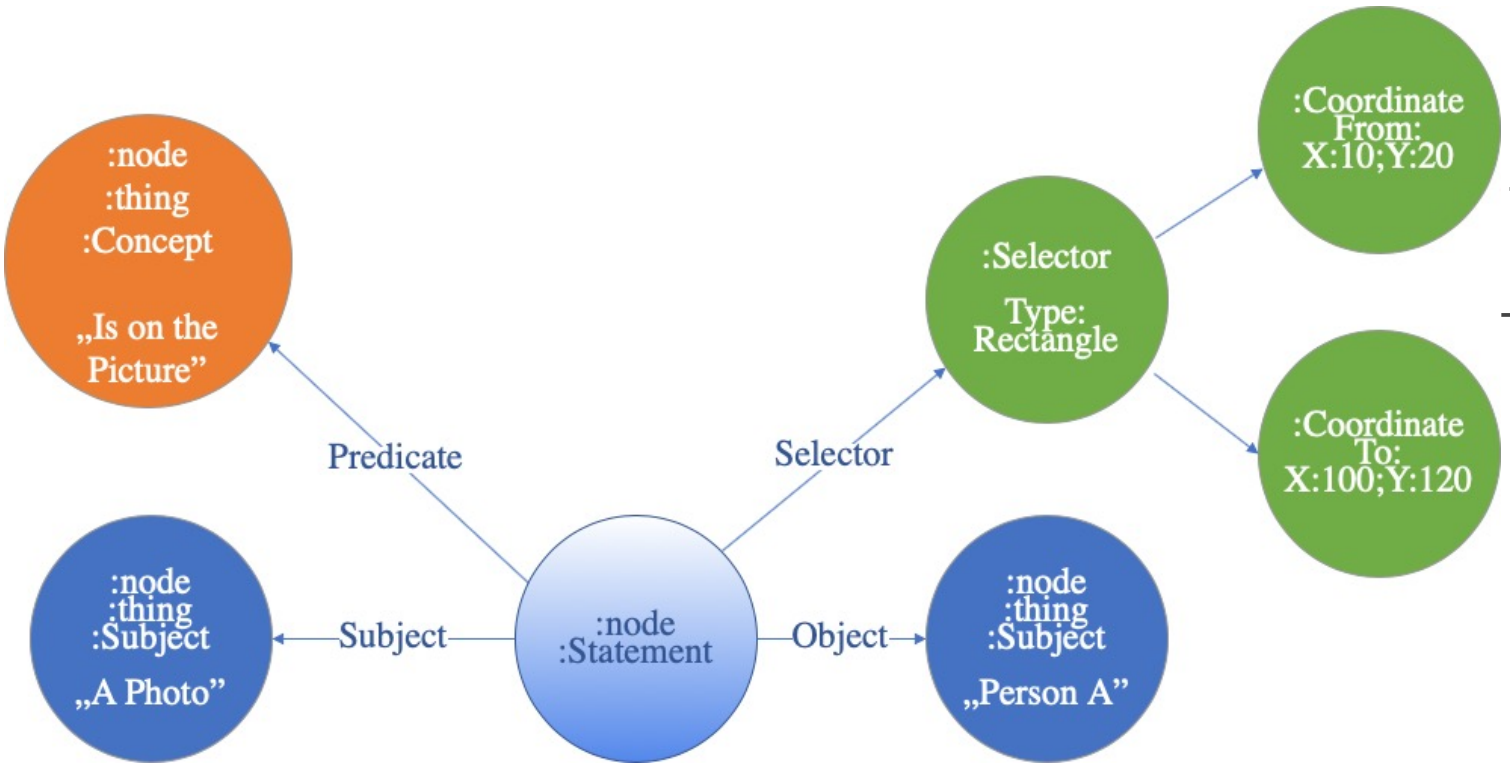


PREDICATE:
THE PREDICATE IS A VOCABULARY ELEMENT TIPIFYING THE STATEMENT, WITH AN EXTENDABLE VALUE SET

OBJECT:
THE OBJECT IS THE BODY OF THE STATEMENT THAT CAN STORE A LITERAL VALUE, POINT TO AN ENTITY AVAILABLE IN ANOTHER SYSTEM

LIFECYCLE:
THE LIFE CYCLE OF A STATEMENT CARRIES, AMONG OTHER THINGS, THE TIME OF CREATION, THE CREATING AGENT, AS WELL AS THE BEGINNING AND THE END OF THE VALIDITY PERIOD OF THE STATEMENT, AND THE “CERTAINTY” CLASSIFICATION OF THE STATEMENT.

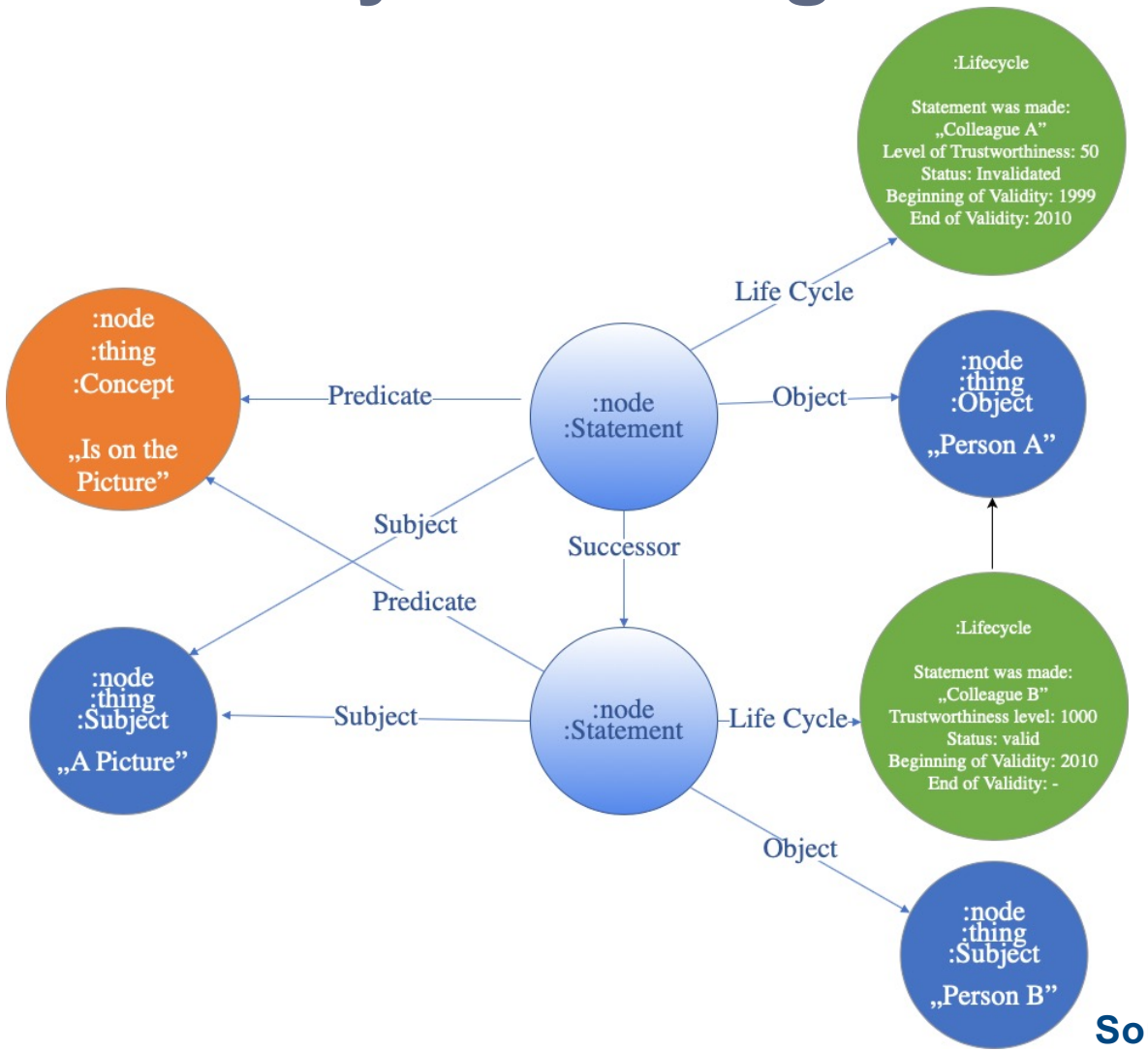
IIIF – localisation of abstract statements



- The framework specializes in displaying / visualising metadata
- The statements are placed on a virtual canvas
- At the visualisaton of an image the given annotations, metadata can be placed in the viewer in an exact manner
- The abstraction formulated in the framework can be extended to all types of media content, by defining the appropriate coordinate system

Source / Copyright: HerMészSoft

Normalized Life Cycle Management of Statements

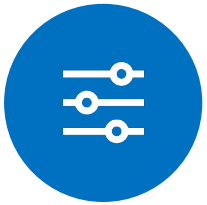


- An illustration of a hierarchy of conflicting statements
- Easy to select statements currently accepted
- Preserving the history of statements
- Statement protection: "Immutable" data

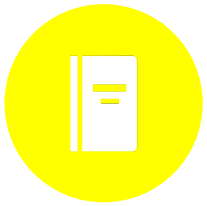
Source / Copyright: HerMészSoft

Cataloguing Module – Local Namespace

ENTITY TYPE:
THE DEFINITION OF THE POSSIBLE REPRESENTATIONS OF THE ENTITIES MANAGED IN THE SYSTEM



AVAILABLE PROPERTY:
THE DEFINITION OF NAMESPACE ELEMENTS CREATED FOR TYPES.



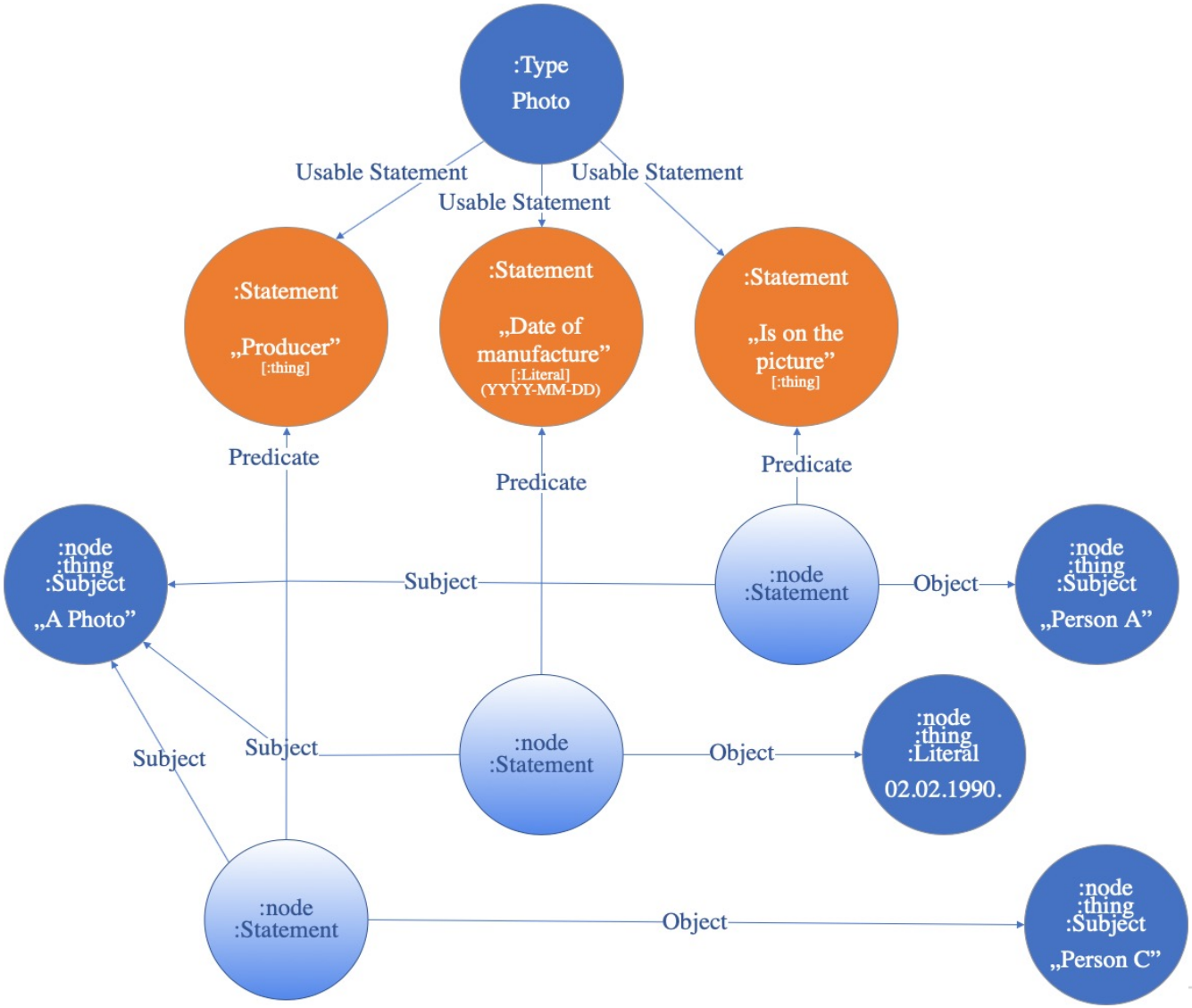
ENTITY:
ENTITIES AND RECORDS MANAGED IN THE SYSTEM

PROPERTY:
STATEMENTS MANAGED IN THE SYSTEM

EVENT:
RECORD OF THE CHANGES IN THE SYSTEM

Source / Copyright: HerMészSoft

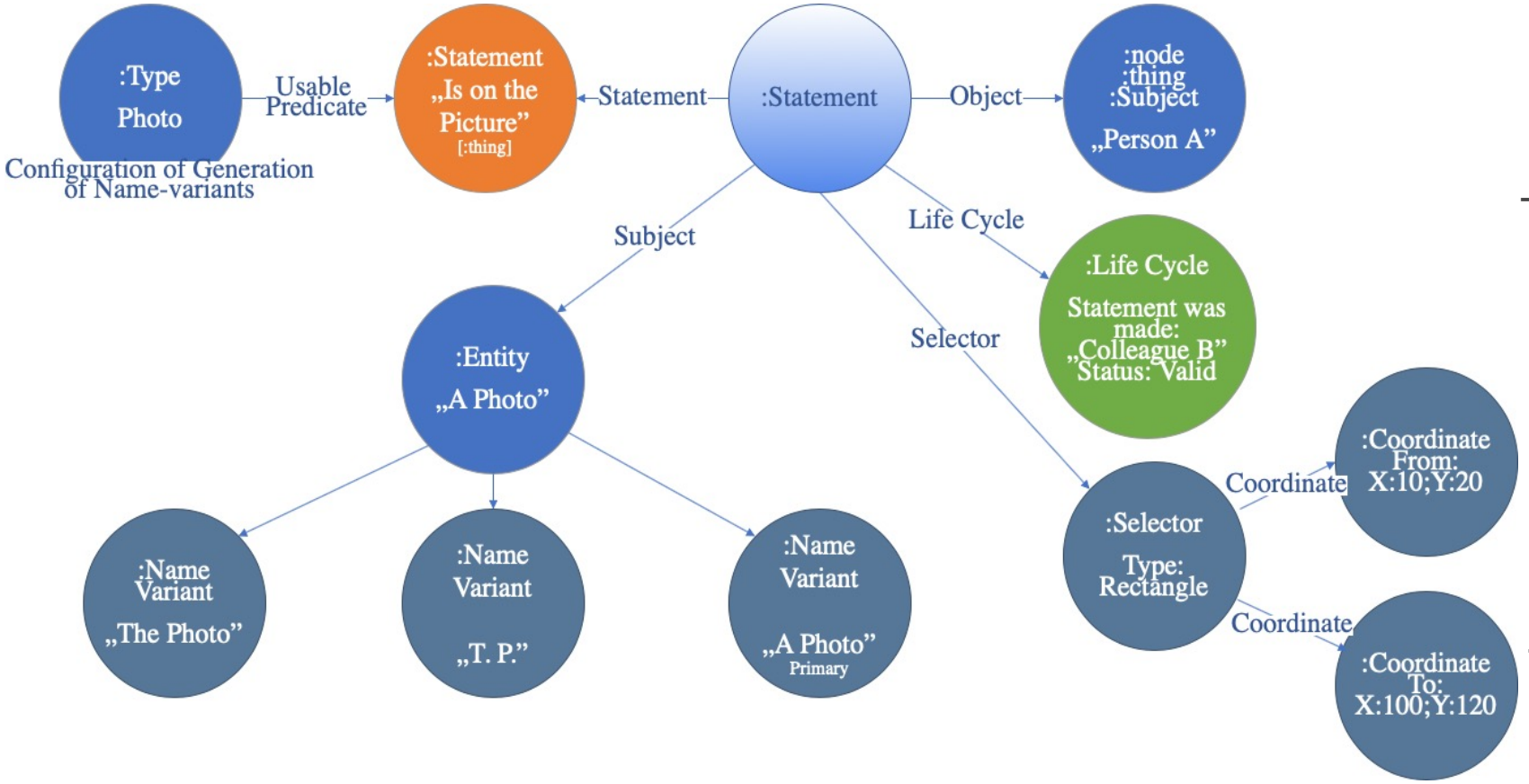
Customizable Set of Values for Record Types



- Entry types exist as part of the data model
- Possible statements (vocabulary) handled by a particular type are freely expandable
- At the statement level, the type of data, the precision of the data, the position of the data on the “canvas” defined by the statement can be defined
- Statement types protection: "Immutable" data

folio

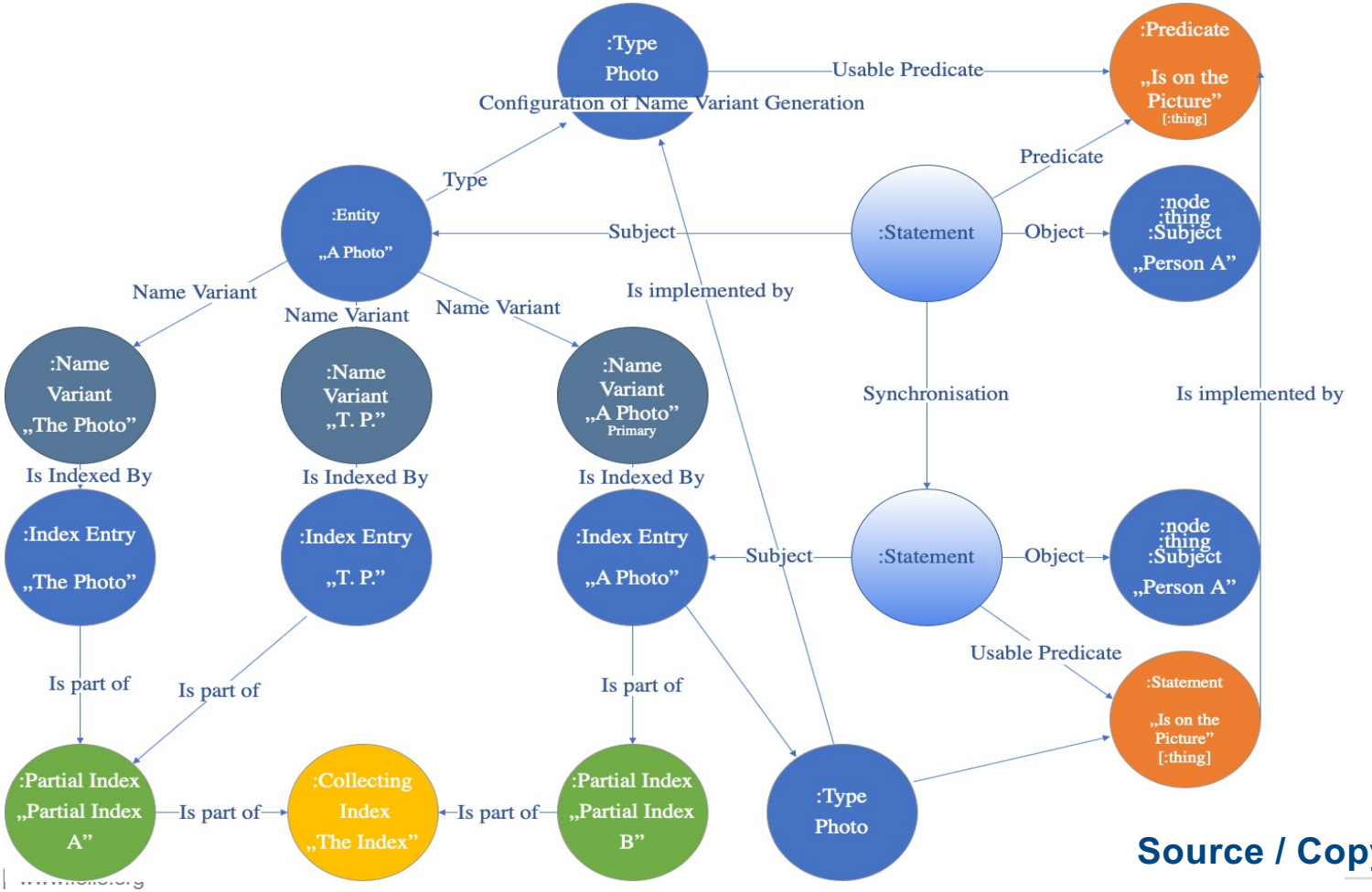
Authority Record simplified Graph Representation



- Individually configurable vocabulary set
- Elemental, individually positionable statements
- Normalized handling of complex data
- "Immutable" statements
- Historical managing
- Automatically derived name variants based on statements

Source / Copyright: HerMészSoft

Relationship between Authority Record and Index Items Indexing Records

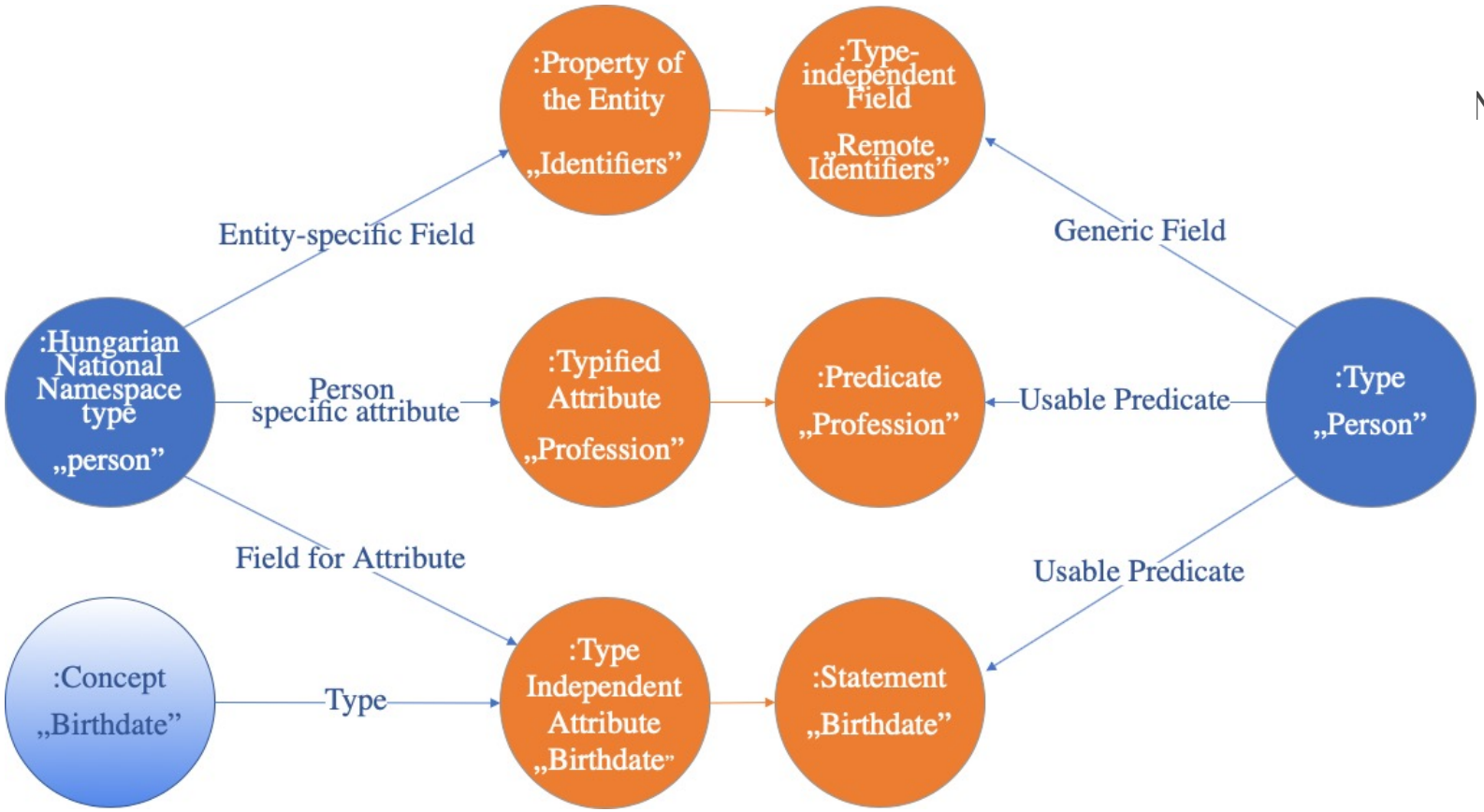


- Entity name variants are formed automatically based on the statements on the entity and the specified configuration
- Each name variant is represented by a separate Index item
- For namespace entities, index items are populated.
- The item is constantly synchronized to changes in the entity.
- "Immutable" data
- Historical management

Source / Copyright: HerMészSoft

folio

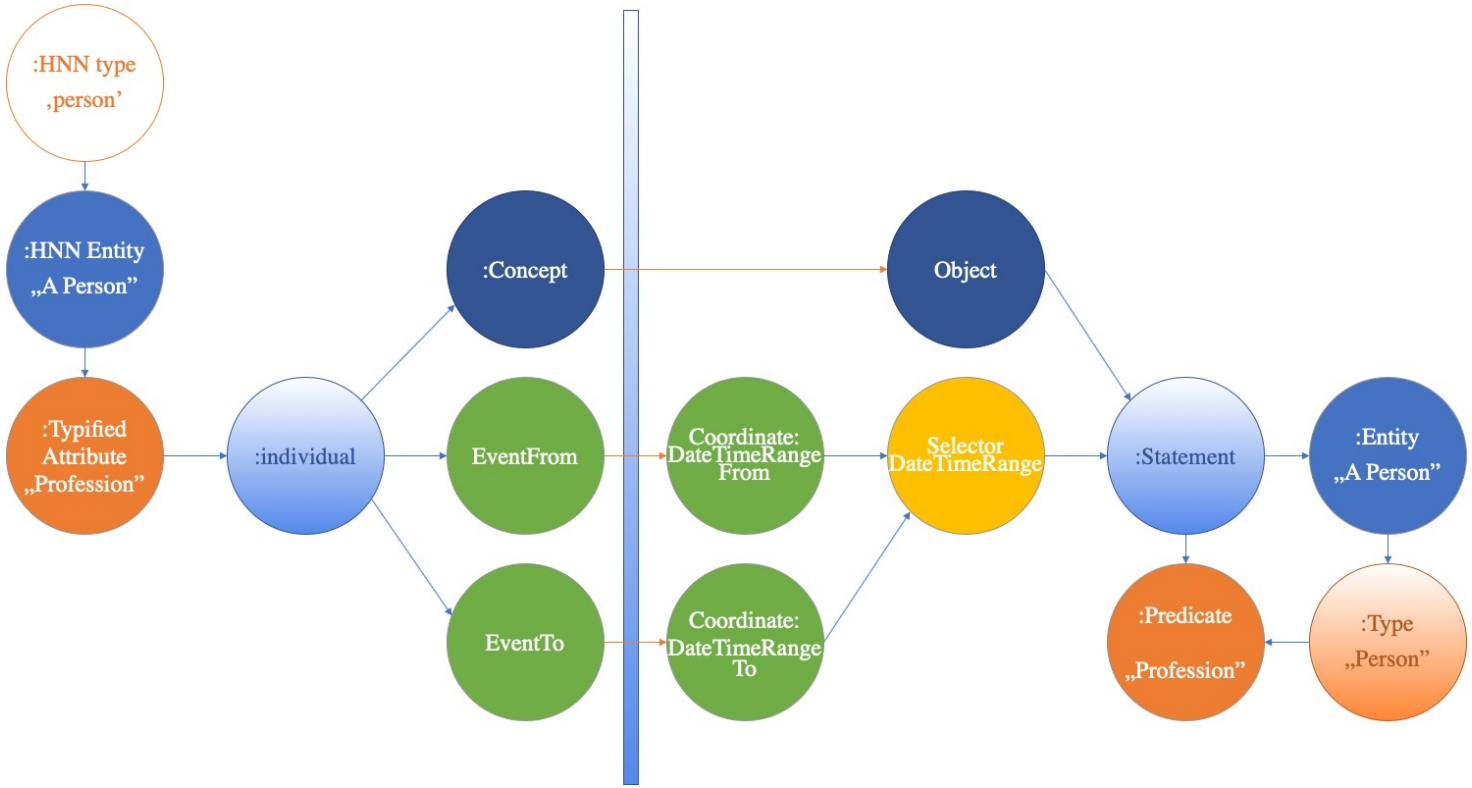
Importing Entities from the Hungarian National Namespace



- The Hungarian National Namespace uses a concept-based dynamically expandable vocabulary to describe Entities.
- Each entity type has specific properties due to the nature of the particular Authority type implemented.
- The predicate of the typizied fields is carried by the field name
- Type-independent fields can be evaluated based on a "concept".

Source / Copyright: HerMészSoft

Hungarian National Namespace Mapping for Import



In the case of the type-specific fields of the Hungarian National Namespace, it can be clearly determined, which fields of the "individual" type of the obtained data structure should be included in which fields of the "statement".

For type-independent fields, we determine which "Statement" is needed to store the value based on the "Concept" ID.

The expression types that carry a value in the HNN are: Individual (Entity, Concept, Event) and Literal.

Data fields for each expression type can be matched 1:1 to the local type (customizable) statement set

Source / Copyright: HerMészSoft

folio

future of libraries is open

More information about the projects:
http://hnlp.oszk.hu
https://www.folio.org

Miklós Lendvay, HNSZL, lendvay.miklos@oszk.hu