

Unveiling and Conceptual-Logical Modeling of Phase Sequences in Data Engineering

Aleksandr Rodionov^a and Georgiy Tsoy^a

^a Computer center of far eastern branch of the Russian academy of sciences, Khabarovsk, 68000, Russia

Abstract

The acceptance of the concept, according to which object types of the conceptual level are differed by functional heterogeneity, leads to the realization of the fact that there are no entity types that wouldn't exchange their instances and wouldn't form numerous sequences in their domains-phase sequences. Formally, such sequences are the directed graphs, in nodes of which are placed entity types, and arcs set sources and receivers of moving instances. The article aims at developing the method for revealing phase sequences at conceptual schemas by means of using a number of types of adjunct categories and subsequent representation of found sequences in databases logical structures. Along with simplest sequences with disjoint types the paper concerns the issuers of formalization and logical modeling of sequences, whose items can simultaneously be present in several types.

Keywords 1

Phase types, phase sequences, direct and reverse type conversion, categories of types, markers of phase sequences

1. Introduction

One of the key tasks that are addressed during conceptual data modeling is to establish a set of classes of interactions $C = \{c_j\}_1^J$ occurring in the domain, as well as constraints on these interactions. An arbitrary interaction class c_j always contains some set of types – $\{T1,..Tt\}$, whose objects participate and (or) can participate in the j-th interaction (figure 1).

No less relevant are the "dynamic " relationships that arise between classes and are associated with the movement of objects from one type to another, including those belonging to other classes. Such types in domains constitute sequences (phase sequences – PS), elements of which are termed phase types as, for example, in [8]. The same article points out that phase types appear when the source type are partitioned. It follows the types of PS aren't overlapped. References to phase types as components of evolutionary, circulating, incremental, loop, and networked object lifespans, can also be found in [6].

Meanwhile, phase sequences with intersecting types are often revealed in domains. For example, in the "Academic degrees" PS, a candidate of sciences in one realm remains a candidate of science even if he received a doctor of sciences degree in another field. Or, the title of master of sports does not cancel the title of candidate for master of sports, which can be observed in the "Sports titles" phase sequence


In addition to identifying such sequences, which is the subject of yet another problem of conceptual data modeling, the mechanisms of PS formation are also of interest. Since a current work is exclusively addressed to the information modeling issues, the certain groups of specific objects and interactions resulting to deriving phase sequences elements will be implied under the modeling

VI International Conference Information Technologies and High-Performance Computing (ITHPC-2021),


September 14–16, 2021, Khabarovsk, Russia

EMAIL: ran@newmail.ru(A. 1)

ORCID: 0000-0003-43N-8562 (A. 1)

 © 2021 Copyright for this paper by its authors.

Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

mechanisms. Some classes (categories) of similar objects were found and elaborated earlier in [1]. It's is the *rank*- and *profile*-classes. In the paper, the pointed list will be expanded due to addition to it types of *stage* and *status* categories.

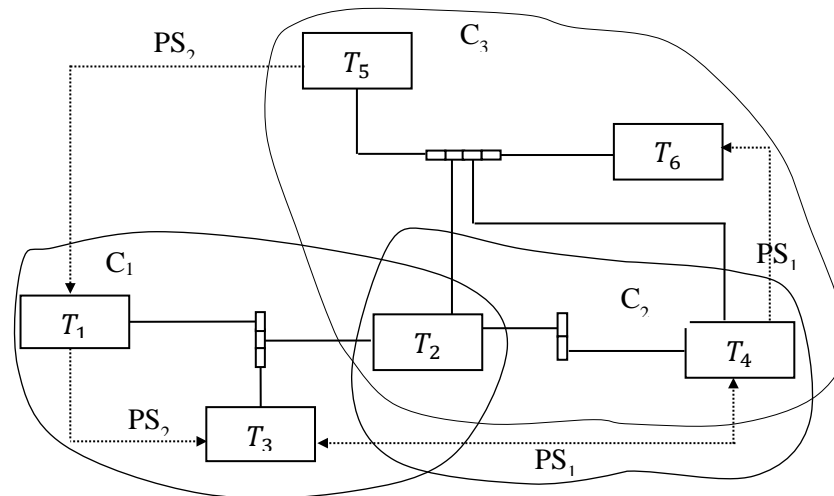


Figure 1: Interaction classes in the style of Object-Role Modeling (ORM) notation

The concept of the phase type considered in the article also applies to the types of the role category [5, 7], allowing us to interpret the latter as phase types with a nominally constant composition.

This work is aimed at developing methods for identifying phase sequences in conceptual schemas, their subsequent representation in logical database structures, and reverse transformation of logical model types into the phase types of the conceptual model.

2. Procedures for conceptual-logical and logical-conceptual data transformation

At first glance, it may seem that mapping classes of conceptual level interactions into logical structures (the direct conversion) is not a difficult task, unless one circumstance is taken into account. Conceptual level types can overlap. Intersections of logical level types are forbidden. This is not the only requirement for the organization of logical data models. The designed data structures should also be compact and lack of redundancy. If the suite of conceptual types didn't systemize in advance – type's categories aren't determined, the number of potential logic schemas will be excessively large. Given the preliminary carried out categorization of types, the conversion procedure may be come to the form shown in Figure 2. (The purpose and functionality of all categories listed in Fig.2 will be elaborated on below as they are arisen in model constructions.)

One should notice that a trivial biunique correspondence are ascertained between most conceptual and logical types and therefore direct conversion doesn't represent any hurdles. The exception is the types of entity categories, for which variant ambiguity is preserved, as the "intersection" issue is relevant only for the entity types.

The task becomes even more complicated if we take into account that entities in domains are present in two kinds - in Prototype and in Sample/Instance – formats (kinds) [2]. (The same is true for processes, with the exception of the Sample format because types of process category can't be in the Sample format.) Prototype-instances compose a Prototype-kind of an entity type and represent prototypes, models of real entities. Instances of other format, being real entities, are generally dispersed among one Sample- and several Instance-kinds, but can form either Sample - or Instance-kinds separately. Entities that participate in interactions can belong to different kinds. Prototype-kinds are associated with Sample/Instance-kinds by associations that have cardinality 1:M. Entity types of the conceptual level can only subsume into one of kinds listed the above, and the correct identification

of kinds is no less important than the categorization of types themselves, since both delineate predefined classes of interactions that instances of the individual entity “types/kinds” can engage into.

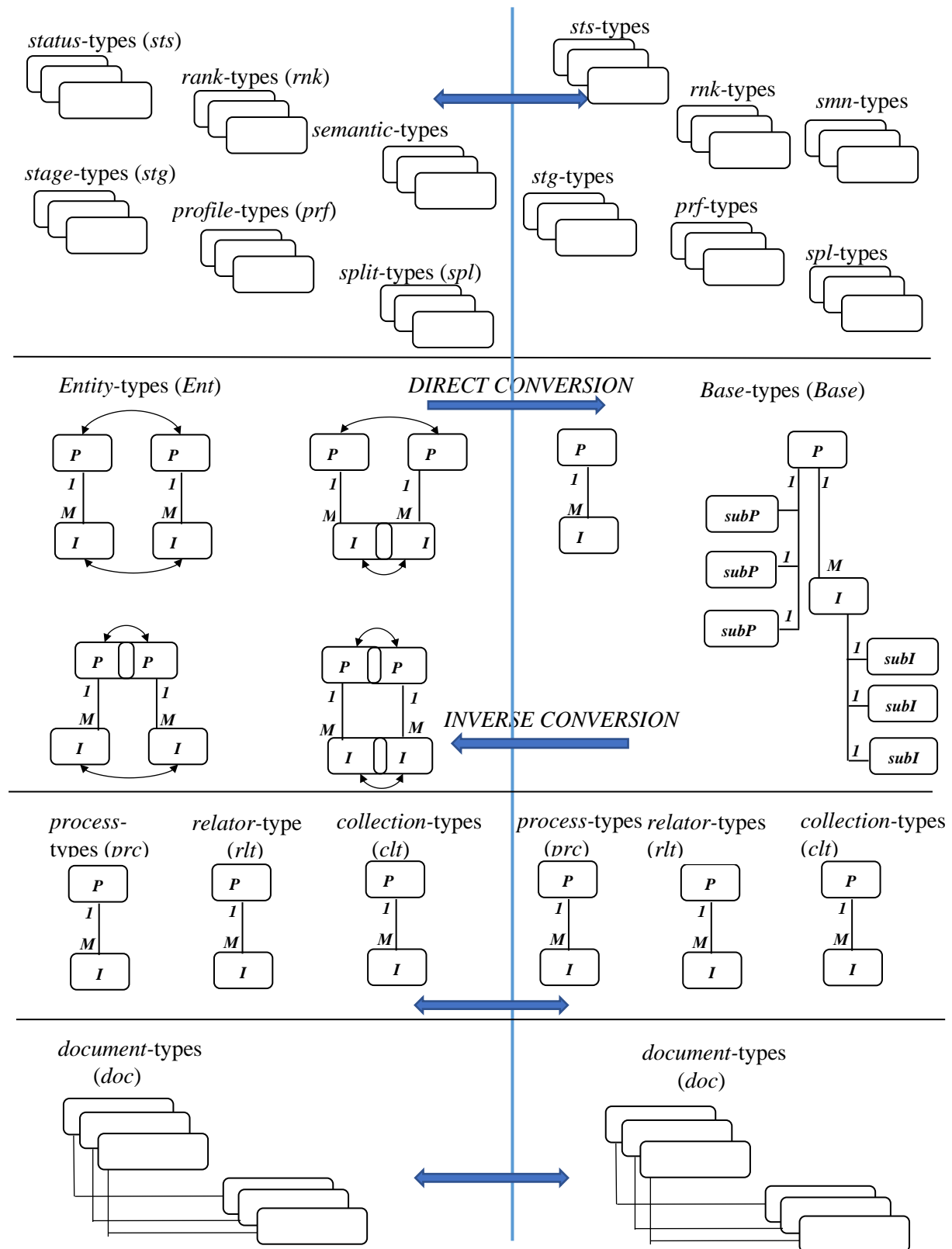


Figure 2: Direct and reverse conversions of categorized types of the conceptual and logical levels

Another important point is that entity instances can migrate from one type to another. Options for potential migrations are signified in Figure 1 with a dashed line. In addition, subsets that exchange their elements may appear within individual types. In order not to complicate the principle diagram of conceptual-logical modeling, such subsets are not included in the scheme.

Note that the abstract conceptual construction in Figure 2 is already a transformed initial conceptual model, in which all classes of interactions in the modeled domain were to be represented.

Regardless of how entity types are structured in the logical scheme, in which their suites are named as base types, the task of inverse transformation of base types into dynamic types remains key for the functional component of any information system with a database.

3. Localization of phase sequences in the conceptual modeling stage. Phase sequence markers

The objects of the world around us are constantly in motion. For specific modeled domains, the vast majority of all possible “motion” kinds are not of any interest. Only a small part of them, which is important in the context of a particular universe of discourse, is the subject to fixing. Transitions can be orthogonal, “natural” such as, for example, changes in the age or weight of an object or “programmed”, motivated by certain circumstances. In the first case, the influence of some objects on dynamic of other objects is either absent, limited, or insignificant, in the second - it matters. Many methods and models have been developed for setting and consequent tracking motion trajectories. With respect to information systems, workflows are the most widespread [3, 6]. In the scope of data, workflows can be represented in the format of phase sequence models. Formally, the latter are directed graphs with the set of states and transitions between them. One of the information modeling tasks is to reveal and represent, first at the conceptual and then logical level, an object type, the instances of which will transfer between states, forming dynamic types associated with these states. The formalization of ways of forming dynamic types is especially important if we take into account that a particular such type also defines the classes of interactions, in which instances of this type can engage.

Detecting of phase sequences among numerous types of conceptual level without using auxiliary types, mostly non-entity ones, is a rather serious issue. Here is a list of types that indirectly indicate the presence of phase sequences in the simulated space:

- age groups by sport – stg-+spl-/prc-types;
- weight categories – stg-type;
- semesters at universities – sts-type;
- track scientific paper in a journal – sts-type + adjacency matrix;
- academic degrees and ranks – rnk-type;
- military ranks of Russian Navi – rnk- type+ prf-type;

All types marked with underscores, if they are linked to one or more entity types, are able to specify phase sequences. Moreover, the very presence of the listed categories types in domains makes sense only when they are combined with the entity types. The types marked as rnk are the rank types, which are ordinary ordered lists. An instance of any similar type is nothing more than the name of one of a phase sequence item. The rank, being by definition the position of an element in relation to other elements, can be set either explicitly or implicitly. Implicitly-as the ordinal number of the item in the list. Explicitly-in the form of some kind of a rating scale.

Another category of types, by which it is often customary to describe phase sequences, we have designated as status. Statuses can be either nodes, or transitions between process nodes, or both, depending on what meaning is attached to these elements in the domain. The objects involved in the process move from one state (status) to another, forming both phrases (statuses) and phase sequences. In general, if the conceptual model of a domain contains prc-types, it is worth to look for phase sequences that correspond to this types. In the role of typical rank types are served, for example, "Military Ranks" and "Scientific Positions".

If we exclude the presence of order in the types that are able to set transitions, we get ordinary lists, such as the type - "Role of players in team sports".

Types subsumed into the stage category also model dimensions of processes, but only the processes, in which transitions occur the natural way, as in the case with age groups or weight categories. Here, too, we can find traces of processes. Stage category is introduced only in order to somehow differ “natural” and “programmed” processes. Since stage-, status -, and rank-type objects are involved in setting phase sequences, it makes sense to reduce them all into the general class (including for the subsequent references to them), and to term this class, for example as the chain (ch) class.

Regardless of which type of the listed categories is involved in specifying the phase sequence, any phase type will represent one of the variants of the entity collections classified in [9]. Depending on the presence or absence of duplicates, as well as the significance or insignificance of the order, the authors of this work suggest to distinguish four kinds of collections: list, set, bag, and ranking.

Phase sequences in domains not often are revealed at the first attempt. First, some phase sequences can be elementary omitted during the conceptual modeling. Secondly, some transitions between individual types may not turn out as obvious as they actually are. For example, this applies to such a set of types as: "Bachelor", "Master" and "Doctoral". And, thirdly, there is simply no need to keep track any transitions. In some measure, the severity of this problem is reduced while using the markers of phase sequences represented by the same ch-types. But wholly, any entity type need consider also the phase type. And in general, any single entity type should be considered as a phase type - the only one in the corresponding phase sequence. At least, in virtue of the fact that there always exist sources and receivers of entity instances.

4. Modeling phase sequences at the logical level

Earlier, it was noticed that ordinary adhering types of chain classes such as *rnk* or *prs* to base types sets the phase sequences. The formal schemas covering all possible ways for specifying the phase sequences (resulting ultimately from the permissible permutation of logical scheme tools) are depicted in Fig. 3. Their number is limited.

Let us proceed from abstract schemas (fig. 3a) to practice situations that better illustrate the meaning and content the issues under consideration. Two *rnk*-types in fig. 3b are connected with the base type “Persons”. In both cases, the cardinality of the connections is M:M. This is a redundant cardinality, since it would be possible to limit it to 1:M, which would reflect and at the same time implement the restriction of the disjointness for individual types in the two phase sequences: “Military ranks” and “Academic degrees”. The M:M cardinality indicates that it is necessary to track the history of transitions of elements between phase types. To fix transitional tracks the weak entity corresponding to M:M must content the mandatory "temporary" attribute as one of the components of its composite primary key.

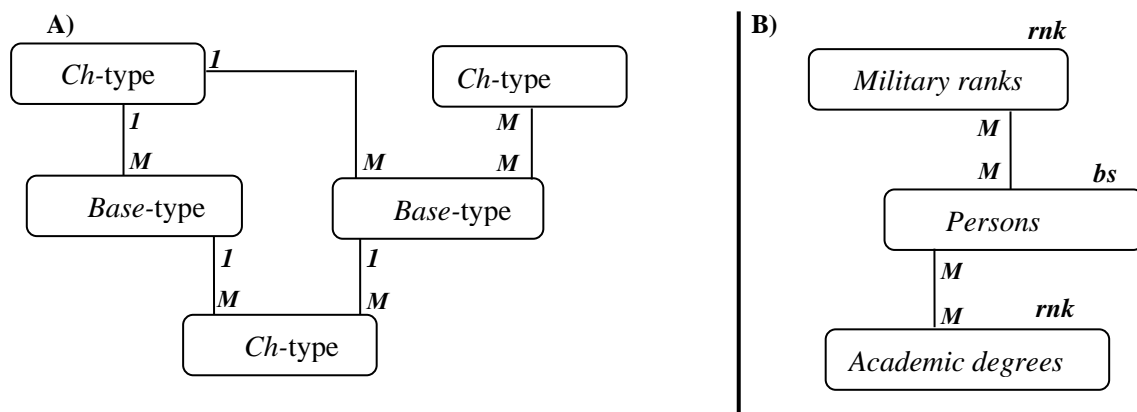


Figure 3: Variants and examples of modeling phase sequences in logic schemas

Rank-types, as a rule, rarely act as independent units. They are usually consolidated with types of other categories, for example, with profile-, split- or base-types, and only then are connected with those base-types, whose instances participate in phase sequences. Similar concatenations are

particular true while modeling phase sequences with overlapping types (PAOT). Moreover, the very existence of PAOT depends on the presence of consolidated types, one of which must pertain to the rank category.

To reveal the concatenation mechanism, we again resort to the "Academic degrees". The specified type is always primarily attached to the "Academic degrees", and then, attracting an intermediary – the "Thesis committees", is connected with the "Persons". The evolution of the transformations for the logical subschema containing all relevant situations for the concrete domain looks like this, as shown in Figure 4.

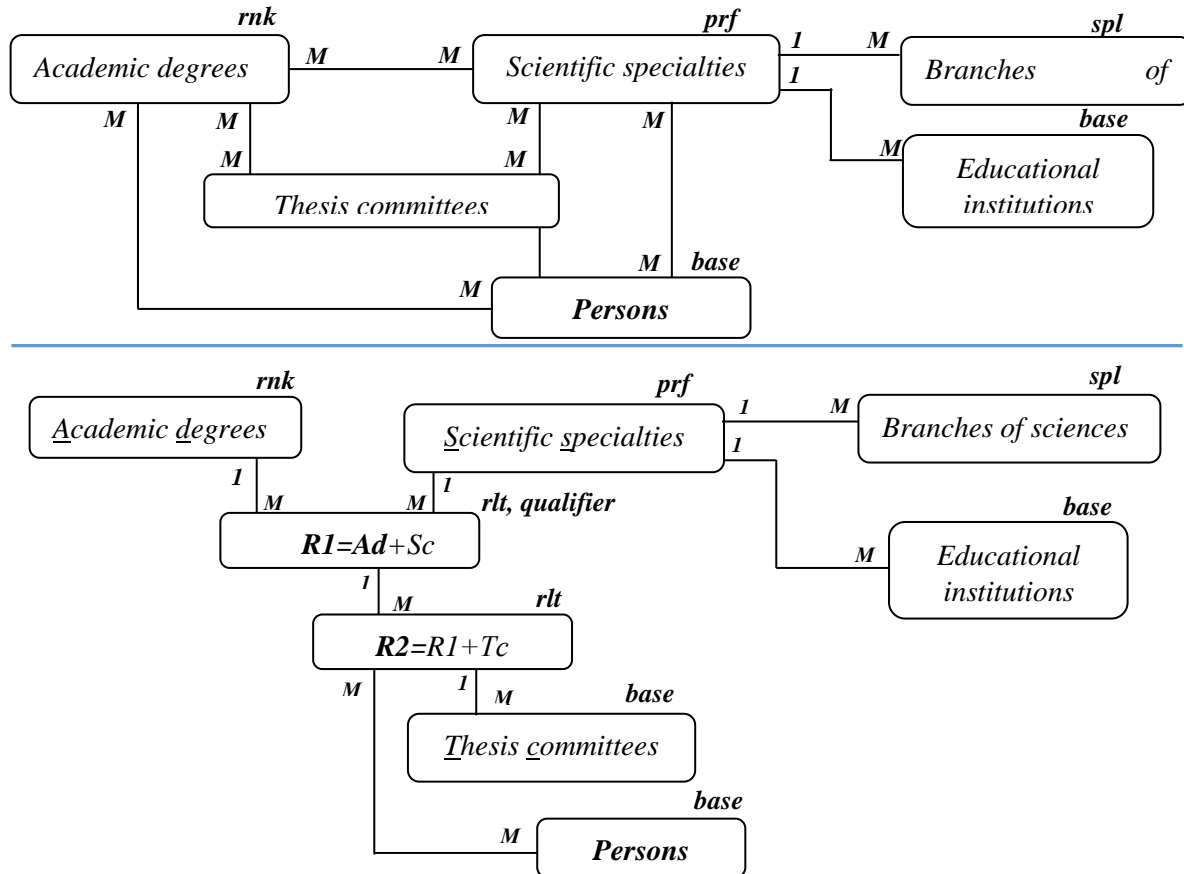


Figure 4: Evolution of logical construction

For the first time prf- and spl-types appeared on the schema. The first type specifies an abstract "scope" where the rnk-type is applicable. The second classifies this area by partitioning it into disjoint subsets. The pair of rnk-prf-types, as shown in [1], generates a new type concerning to the qualifier category, which formally is a "weak entity" in the format of a rlt-type-materialized relation [10]. In the diagram, it is designated as R1. The inclusion of another rlt-type R2, as well as the previous R1 in general, is caused by the fact that their instances manifest themselves as self-sufficient objects, i.e., they participate in interactions with "Tc" (R1) and "Persons" (R2).

The principle diagram for modeling PAOT at the logical level (fig. 4) is obvious. The latter must contain from one to several target base-types - suppliers of elements for phase types, rnk-type, and from one to several "auxiliary" types represented by base-types, which distinguish from target types, prf- and types of other categories.

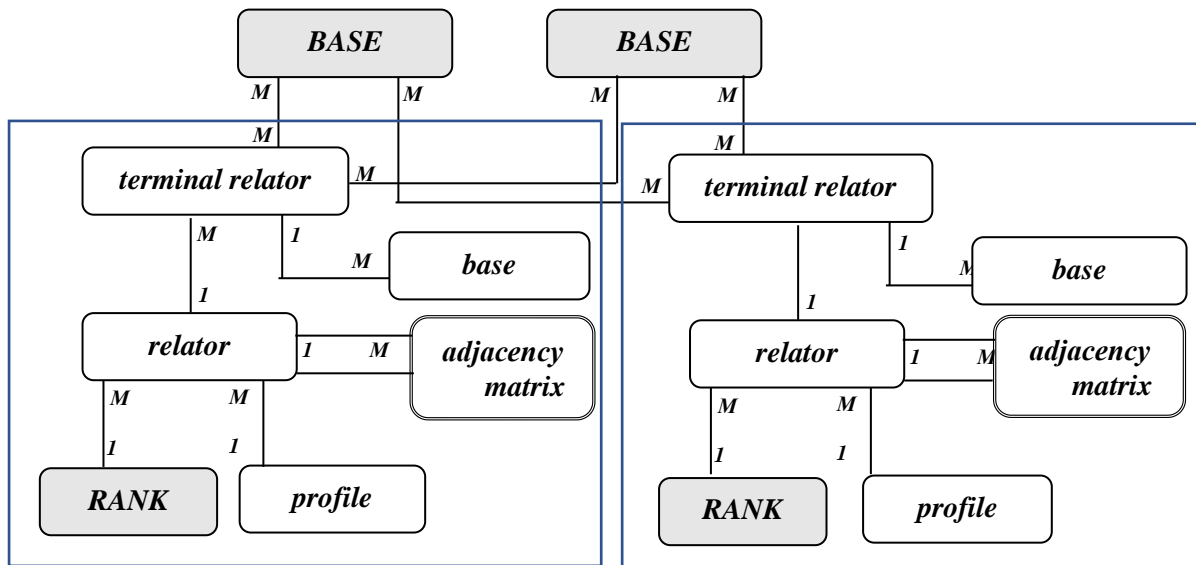


Figure 5: Elements of a logic subschema for modeling multiphase sequences with intersecting types

If there are transition matrixes (adjacency matrixes) that reflect the coherence of rlt-type instances, we can obtain several PAOT. For an example from the practice used in the article, it can be such PAOT as, for example, “Candidates of Sciences – Doctors of Sciences”, “Candidates of Sciences by branches of Sciences – Doctors of Sciences by branches of sciences”, etc.

All adjacency matrixes should be attributed to the prototype-kind for an obvious reason, since they define the allowed tracks for instances of the target base-types. Real transitions are tracked by means of instances of weak entities that implement relationships between the “terminal relator” and target base-types.

If there are several rlt-types in a structural cluster, it is appropriate to articulate the issue of the relations between different terminal relators and the constraints on these relations. Suppose that the schema in the figure 5 contains two connected terminal relators. But the hierarchical nature of the links connecting the rank-type with the terminal relator suggests that the higher adjacency matrix should “absorb” all the lower-lying adjacency matrix. It follows that the number of independent rank-types concentrated in a structural cluster also determine the maximum number of adjacency matrix in it.

5. Conclusion

There are no entity types in domains that wouldn’t exchange their instances, and thus wouldn’t form phase sequences. Even if phase sequence consists of only one single type, there must exist (most likely outside the modeled domain) some “input” and “output” types that point to the sources and receivers of entity instances. Specific types in phase sequences are ordinary role types, in the configuration of which, as well as the phase sequences themselves, types of certain non-entity categories participate. First of all, this applies to the types of rank, stage or status categories, which are put together in a chain-group.

The appearance of types of listed categories at conceptual schemas immediately indicates the need for identification and consequent modeling of a particular phase sequence. The article describes in detail kinds, assignments and various options for the attachment of chain-types to the entity types, as well as logical constructions for modeling the composition of phase types and the movement of particular elements between them.

Separately, the issues of detecting and modeling phase sequences with overlapping types reflecting the fact that the same instance can simultaneously be in more than one phase type are discussed. We ascertained that in order to generate the corresponding sequences it needs a mandatory rank-type and from one to several concomitant types that related to types of profile, entity or split categories.

Types of corresponding categories, apart from the aforementioned role, bear also the basic load in ensuring transformation of base types (base types are the logical model types containing entities) into the entity types of a conceptual level. All data needed for transformation are concentrated in the associations that connect base and auxiliary types. The paper generally reveals the mechanism of direct and inverse transformations for types of conceptual and logical models, which allows us to form a holistic system view of the purpose, basic properties and interrelation of conceptual and logical structures.

6. Acknowledgements

The studies were carried out using the resources of the Center for Shared Use of Scientific Equipment "Center for Processing and Storage of Scientific Data of the Far Eastern Branch of the Russian Academy of Sciences", funded by the Russian Federation represented by the Ministry of Science and Higher Education of the Russian Federation under project No. 075-15-2021-663.

7. References

- [1] A. N. Rodionov, The abstract roles and the primitives of role modeling in the conceptual, logical, and physical data models system *Information technology*, 2019, №4, V. 25, pp. 451–466
- [2] A. N. Rodionov, Semantic identification, configuration and entities types modeling for the data model engineering *Vestnik NSU. Series: Information technologies*, 2014, V. 12, №.1, pp. 64–78.
- [3] N. Sidorova, C. Stahl, N. Trcka, Soundness verification for conceptual workflow nets with data: Early detection of errors with the most precision possible // *Information system – 2011. vol. 36. – P. 1026-1043.*
- [4] B. Thalheim, Component development and construction for database design // *Data and knowledge engineering*, 2005, V.54, pp. 77-95.
- [5] F. Steimann, On the representation of roles in object-oriented and conceptual modeling // *Data and knowledge engineering*. 2000. V.35. P.83-106.
- [6] W. van der Aalst, van Hee, *Workflow management: models and systems* The MIT press Cambridge, Massachusetts London, England 2002
- [7] T. Halpin, T. Morgan, *Information modeling and relational databases*. Morgan Kaufmann Publishers, 2008. 970 p.
- [8] G. Guizzardi, Ontological patterns, anti-patterns and pattern languages for next-generation conceptual modeling *ER 2014, LNCS 8824*, pp. 13-27, 2014
- [9] S. Hartmann, S. Link, Collection type constructors in entity-relationship modeling *ER 2007*, pp. 307-322.
- [10] G. Guizzardi, *Ontological foundations for structural conceptual models*, Ph.D., thesis, Center for Telematics and Information Technology, University of Twente, The Netherlands, 2005, 441 p.