INTEGRATION OF RAILWAY INFRASTRUCTURE TOPOLOGICAL DESCRIPTION ELEMENTS FROM THE MICRO$_{\mathrm{L_2}}$ TO THE MACRO$_{\mathrm{N_0,L_0}}$ LEVEL OF DETAIL

A. Hlubuček[*]

Abstract: The paper presents the method of integration, which is supposed to be applied to the structure of the railway infrastructure topological description system expressed at the level of detail designated as $\texttt{micro}_{\mathrm{L_2}}$ in order to transform it into the structure expressed at the level of detail designated as $\texttt{macro}_{\mathrm{N_0,L_0}}$. The $\texttt{micro}_{\mathrm{L_2}}$ level is the level of detail at which individual tracks in the structural sense and turnout branches are recognized, while the $\texttt{macro}_{\mathrm{N_0,L_0}}$ level is the level of individual operational points and line sections. The proposed integration algorithm takes into account both the parameter values of the individual elements appearing at the reference level of detail $\texttt{micro}_{\mathrm{L_2}}$ and their topological interconnectedness. Based on these aspects, these elements are integrated into the elements of the derived level of detail $\texttt{macro}_{\mathrm{N_0,L_0}}$ that can be described by the transformed parameter values. The relations between the respective elements are also transformed accordingly. While describing the transformation algorithm, the terminology and principles of the UIC RailTopoModel are used.

Key words: *integration of elements, topological description, railway infrastructure, RailTopoModel*

Received: November 18, 2022
Revised and accepted: February 28, 2023

DOI: 10.14311/NNW.2023.33.002

1. Introduction

The data description of the railway infrastructure is a key source for many currently existing and developed software applications aimed at supporting the processes taking place on the railway. For each application, a description of the infrastructure at a certain level of detail is appropriate. Each of the applications is focused on its range of attribute description of the railway infrastructure devices and properties, the data of which it uses for its purpose [3]. Although the focus area of individual applications varies, the underlying topological layer of own transport routes, relative to which individual railway devices and properties can be localized, remains

*Adam Hlubuček; Czech Technical University in Prague, Faculty of Transportation Sciences, Konviktská 20, CZ-110 00 Praha 1, Czech Republic, E-mail: hlubuada@fd.cvut.cz

the same. The way of its description can only differ if different methods are used and depending on the level of detail at which the structure of this layer is expressed.

For the reasons mentioned, there is a need to be able to automatically transform the structure of the railway infrastructure topological description system (which will be referred to as the topological layer in the following text) expressed at a certain level of detail into the structure of another level of detail. Let the source structure be referred to as the reference structure $\mathcal{S}_{\mathrm{R}}$ and the level of detail at which it is expressed as the reference level of detail l_{R}. Let the target structure be referred to as the derived structure $\mathcal{S}_{\mathrm{D}}$ and the level of detail at which it is expressed as the derived level of detail l_{D}. Carrying out the transformation of the topological layer structure from the reference level of detail l_{R} to the derived level of detail l_{D} is a prerequisite for achieving the portability of data describing the railway network across the relevant levels of detail.

The aim of this paper is to present such a transformation method that allows the structure of the topological layer expressed at a more detailed level at which the elements represent the individual tracks in the structural sense and branches of turnouts (let this level of detail in the role of the reference level of detail l_{R} be designated as $\texttt{micro}_{\texttt{L}_2}$) to be transformed into the form of a less detailed level at which the elements represent the individual operational points and line sections (let this level of detail in the role of the derived level of detail l_{D} be designated as $\texttt{macro}_{\texttt{N}_0,\texttt{L}_0}$). Since this is a transformation of the structure from a more detailed level to a less detailed level, it is supposed to be based on integrating the elements of the more detailed $\texttt{micro}_{\texttt{L}_2}$ level structure in order to obtain the elements of the less detailed $\texttt{macro}_{\texttt{N}_0,\texttt{L}_0}$ level structure.

2. The RailTopoModel principles

One of the most important recently widely applied methods of railway infrastructure systematic description that allows to define a view of the railway infrastructure at several levels of detail is the UML-based UIC RailTopoModel solution. The RailTopoModel version 1.0 was released in 2016 as IRS 30100 [4]. Since then, versions 1.1, 1.2 and 1.4 have been published. The two latest versions were created in collaboration with the development of raiML 3, the railway markup language interchange format, which is a significant use case of this generic model [5, 7].

The RailTopoModel makes it possible to describe a railway infrastructure network at different levels of detail, which can be identified with instances of its `LevelNetwork` class. To some extent, the level of detail can be expressed by the value of its attribute `descriptionLevel`. According to the RailTopoModel, the attribute can take the values `micro`, `meso` and `macro`, which are the recommended description levels [1, 4].

In terms of the network topology system, the RailTopoModel assumes the topological layer at a certain level of detail to be described by net elements and net relations. Net elements and net relations can be assigned to any of the defined description levels, while the structure of their description and interconnection remains the same. It uses two classes of net elements to distinguish between non-linear net elements and linear net elements. Non-linear net elements can represent real infrastructural objects of a point or area character, while linear net elements represent such objects of a linear character and have their orientation defined. The RailTopoModel also allow us to express which net elements of a more detailed level are aggregated into which net elements of a less detailed level. The element part collections used for this can be either unordered or ordere. [4, 6].

Originally, the RailTopoModel considered net elements to be dimensionless, allowing their intrinsic coordinates to be expressed only using a relative value on the scale between the limit values `0` and `1`. Nevertheless, the version 1.4 introduces the `length` attribute of the `NetElement` class, which expresses the length of the respective net element. Net elements can be connected by net relations of the `PositionedRelation` class, which makes it possible to distinguish whether the respective net relation is anchored at the beginning or at the end of the respective net element (which is expressed using the value `0` or `1` of the intrinsic coordinate). In addition, these relations are assumed to be described by the value of the `navigability` attribute, which determines their traversability with respect to the direction in which the relation is created. It can take the values `both`, `none`, `ab` or `ba` [4, 6].

With the use of the RailTopoModel it is also possible to define any number of linear and geometric coordinate systems. These coordinate systems, together with the concept of location relative to the net elements using intrinsic coordinates, can be used to locate net entities representing individual devices and properties within the respective network. The generic RailTopoModel only defines the general net entity concept and does not concretize the classes of the specific net entities. This is a task for its individual use cases such as railML 3 [4–6].

Although the RailTopoModel introduces the mechanism allowing net elements to be aggregated across different levels of detail, it does not define any rules on the basis of which this aggregation should be performed. For this reason, it is necessary to look for an appropriate method of transformation for each pair of levels of detail that are to be interconnected in this way. Of course, it is also necessary to take into account which of these levels plays the role of the reference level of detail l_{R} (i.e., at which the topological network description was originally created) and which plays the role of derived level of detail l_{D} (i.e., at which the topological network description is to be created).

3. Allowable structure based on levels of detail

In order to design a transformation method applicable to the topological layer structure expressed at a certain reference level of detail l_{R}, which is to be transformed to the structure expressed at a certain derived level of detail l_{D}, it is necessary to define which rules are to be applied to the its structure expressed at these levels of detail. We are able to advantageously express the structure of the topological layer at different levels of detail using the RailTopoModel tools, i.e., net elements and net relations. When defining the rules for the allowable structure of a specific level of detail l, we can decide whether to combine non-linear and linear net elements or not. For the needs of connecting net elements, since this is sufficient, we will only use the net relations with the `both` or `none` value of the `navigability` attribute.

The enumeration of the `descriptionLevel` attribute allowable values, as introduced by the RailTopoModel, is not sufficient to express the rules regarding the allowable structure of the topological layer at a specific level of detail l. E.g., it does not take into account the net elements of which specific classes can be contained in the structure. Some applications also work with completely different levels of detail, e.g., the nano level [2]. For these purposes, it is necessary to introduce more precise denotations of the relevant levels of detail. The transformation method presented in this paper is intended to perform the transformation of the $\mathtt{micro}_{\mathtt{L}_2}$ structure into the $\mathtt{macro}_{\mathtt{N}_0,\mathtt{L}_0}$ structure, while the said designations will be further explained in the following subsections.

3.1 Reference level of detail

For the purposes of the transformation method presented in this paper, the role of the reference level of detail l_{R} is played by a relatively detailed level, at which the individual net elements represent tracks in the structural sense and turnout branches. A track in the structural sense means a continuous section of a track between branches of two turnouts or between a branch of a turnout and a differently defined end of the track, which can be, e.g., an administratively determined border or a buffer stop (which is not expected to be explicitly represented by a separate net element). A turnout branch is a part of a turnout that allows passage from the beginning to the end of the turnout or vice versa in a specified direction, depending on the status of the turnout switch. In the case of track crossings (although they are not turnouts), single and double slip crossovers and complex turnout structures, we also allow shorter sections that meet at the nodal points of such a structure, e.g., in the centre of the crossover, to be considered as turnout branches for the purpose of defining the structure of the reference level of detail.

To a certain extent, this way of description corresponds to the commonly understood meaning of the `micro` value of the `descriptionLevel` attribute that the RailTopoModel implements for the `LevelNetwork` class. It is rather even more detailed, as track routes are divided into sub-sections for the purposes of what physical infrastructure objects are taken into account, in this concept [4]. Both the tracks in the structural sense and the turnout branches are represented by linear net elements. For the transformation method, it is not even necessary to distinguish which linear net elements represent the tracks in the structural sense and which represent the turnout branches. Since the structure of the topological layer at this level of detail consists only of linear net elements (and positioned relations between them) that are divided in this way (in order for a simple turnout to consist of two linear net elements), the appropriate level of detail will be designated as $\mathtt{micro}_{\mathtt{L}_2}$.

An example of the topological layer structure expressed at the $\mathtt{micro}_{\mathtt{L}_2}$ level of detail is shown in the Fig. 1. The given example represents a circular railway line consisting of a single-track line section with three simple railway stations and a double-track line section between two junctions where the single-track line section changes to the double-track line section and vice versa. Individual linear net elements represented by one-way arrows are named with Arabic numbers. Navigable positioned relations are represented by bidirectional arrows, while non-navigable positioned relations are represented by line segments without arrows.

As for the description of the net elements at the $\mathtt{micro}_{\mathtt{L}_2}$ level of detail, this can also be done independently of which type of real object the corresponding linear

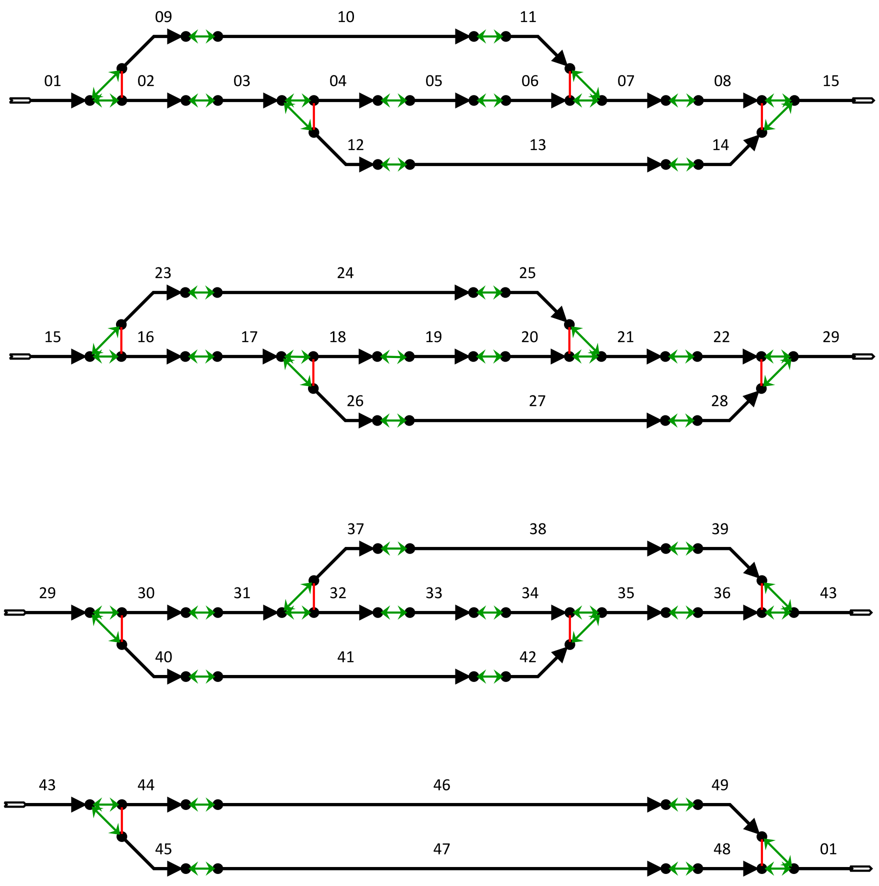

Fig. 1 *An example of the topological layer structure at the* $\mathtt{micro}_{\mathrm{L}_2}$ *level of detail.*

net element represents. This description can be implemented using net element attributes. The RailTopoModel class `NetElement` has the `length` attribute defined [6]. On the basis of the value of this attribute, a classification of the net elements can be suitably carried out for the purpose of their subsequent integration. Further description of the net elements depends on specific use cases. Additional descriptive data can also be attached to the net elements in the form of the net entities localized relative to them. In this case, it can be the class of the specific located net entity, values of its attributes, and in which position or range of positions the net entity is located (e.g., whether it covers the entire net element).

From the point of view of the way of description, the linear net elements expressing real objects of the stated two types are equivalent. Nevertheless, the interface between them must be clearly defined in order to they can be properly clustered and assigned to the emerging net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$. If these interfaces are not recognisable, e.g., if the linear net elements of the reference structure $\mathcal{S}_{\mathrm{R}}$ represent entire track routes between switch points (which then does not meet the definition of the $\texttt{micro}_{\mathrm{L}_2}$ level of detail), the structure must first be modified to meet the defined requirements. In the case of the given example, this could be performed, e.g., using a special case of series linear net element disintegration. For the same reason, two linear net elements representing a track in the structural sense cannot be directly connected to each other. If this situation occurs, they must be replaced by one linear net element to meet the required conditions. This can be performed as series linear net element integration. In all the above cases of transformation of the topological layer structure, a recalculation of the description of the linear net elements must also be performed (this may include, e.g., the recalculation of the `length` attribute values of the individual linear net elements).

3.2 Derived level of detail

As regards the derived level of detail l_{D}, it is less detailed in the case of the method presented in this paper. At this level of detail, the individual net elements represent either operational points or line sections. An operational point is understood as a rail transport control point with track branching, which may include, e.g., stations, junctions and shunting yards. A line section in the context of this usage means the section of railway line connecting two operation points. A line section can be both single-track and multi-track, which can only be expressed by attribute values and not discernible based on the network topology at this level of detail.

This way of description corresponds to the commonly understood meaning of the `macro` value of the `descriptionLevel` attribute that the RailTopoModel implements for the `LevelNetwork` class [4]. The said facts mean that at this level of detail the structure of the topological layer is expressed using net elements that represent real objects of largely different types. Real objects of each of these types should be represented using net elements of different classes. It is advisable to express the operating points as non-linear net elements and the line sections as linear net elements. Since the structure of the topological layer at this level of detail consists of both linear and non-linear net elements, while only net elements of different classes can be interconnected by positioned relations, the appropriate level of detail will be designated as $\texttt{macro}_{\mathrm{N}_0,\mathrm{L}_0}$.

An example of the topological layer structure expressed at the $\texttt{macro}_{\mathrm{N}_0,\mathrm{L}_0}$ level of detail is shown in the Fig. 2. The given example represents the same circular railway line as the Fig. 1, only expressed at the less detailed level. Individual non-linear net elements represented by circles and individual linear net elements represented

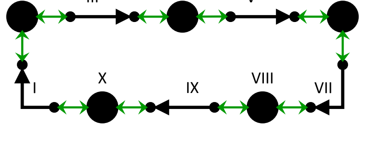

Fig. 2 *An example of the topological layer structure at the* $\texttt{macro}_{\mathrm{N}_0,\mathrm{L}_0}$ *level of detail.*

by one-way arrows are named with Roman numbers. Navigable positioned relations are represented by bidirectional arrows. The topological description at the `macro` level (unlike the `meso` level) does not even show the number of tracks of the individual line sections. This may only be apparent from the attribute description of the individual linear net elements or the net entities localized to them, if such a description is introduced.

Based on the specific class of net elements (and also on the type of real objects they express), the structuring of their description may vary. In general, it is possible to use the values of the dedicated attributes of the net element class as well as the classes and values of the attributes of the net entities located in relation to them to describe these net elements, as mentioned above. At this level of detail, the attributes have a more general meaning. Their values can be determined, e.g., when integrating the net elements of a more detailed level, i.e., in our case, e.g., the $\mathtt{micro}_{\mathrm{L}_2}$ level in the role of the reference level of detail l_{R}.

4. Transformation method proposal

In order to transform the topological layer structure from the $\mathtt{micro}_{\mathrm{L}_2}$ level of detail to the $\mathtt{macro}_{\mathrm{N}_0,\mathrm{L}_0}$ level of detail, a transformation algorithm is proposed. For the sake of clarity, this algorithm is divided into three consecutive sub-algorithms. These sub-algorithms are the Classification of Linear Net Elements Algorithm 1, the Integration into Non-Linear Net Elements Algorithm 2 and the Integration into Linear Net Elements and Creation of Positioned Relations Algorithm 3.

The Classification of Linear Net Elements Algorithm 1 ensures the sorting of the linear net elements of the reference structure $\mathcal{S}_{\mathrm{R}}$ into respective sets according to net elements of which class they are to be subsequently integrated in order to create the derived structure $\mathcal{S}_{\mathrm{D}}$. Based on this class, the sorted linear net elements are further processed either by the Integration into Non-Linear Net Elements Algorithm 2 or by the Integration into Linear Net Elements and Creation of Positioned Relations Algorithm 3.

The initial assumption for the use of the transformation algorithm is the knowledge of the structure $\mathcal{S}_{\mathrm{R}}$ of the topological layer, described at the reference level of detail l_{R}, which corresponds to the $\mathtt{micro}_{\mathrm{L}_2}$ level of detail. Since we assume the description using the RailTopoModel solution, the reference structure $\mathcal{S}_{\mathrm{R}}$ includes a set of net elements $\mathcal{A}_{\mathrm{R}}$ (in this case these are linear net elements) and a set of positioned relations $\mathcal{R}_{\mathrm{R}}$. This fact can be expressed as

$$\mathcal{S}_{\mathrm{R}} = (\mathcal{A}_{\mathrm{R}}, \mathcal{R}_{\mathrm{R}})\,.$$

By applying the algorithm, we obtain the structure $\mathcal{S}_{\mathrm{D}}$ of the topological layer described at the derived level of detail l_{D}, which corresponds to the $\mathtt{macro}_{\mathrm{N}_0,\mathrm{L}_0}$ level of detail. Similarly to the previous case, the structure $\mathcal{S}_{\mathrm{D}}$ includes a set of net elements $\mathcal{A}_{\mathrm{D}}$ (in this case these are both non-linear and linear net elements) and a set of positioned relations $\mathcal{R}_{\mathrm{D}}$. This fact can be expressed as

$$\mathcal{S}_{\mathrm{D}} = (\mathcal{A}_{\mathrm{D}}, \mathcal{R}_{\mathrm{D}})\,.$$

In addition, we also obtain a set $\mathcal{U}$, which is the set of objects representing unordered collections of the linear net elements of the set $\mathcal{A}_{\mathrm{R}}$ expressing their grouping into the net elements of the set $\mathcal{A}_{\mathrm{D}}$.

As regards the individual elements of the already mentioned sets, as well as of the sets mentioned below, they can be also understood as unique identifiers of the corresponding instances of the respective model classes.

4.1 Classification of linear net elements

The Classification of Linear Net Elements Algorithm 1 provides sorting the linear net elements of the reference structure $\mathcal{S}_{\mathrm{R}}$ from the set $\mathcal{A}_{\mathrm{R}}$ into sets $\mathcal{B}$ and $\mathcal{C}$ based on a classification criterion. The set $\mathcal{B}$ is supposed to include the elements of the set $\mathcal{A}_{\mathrm{R}}$ that are intended to be integrated into the non-linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$. The set $\mathcal{C}$ is supposed to include the linear net elements of the set $\mathcal{A}_{\mathrm{R}}$ that are intended to be integrated into the linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$

Algorithm 1 Clasification of Linear Net Elements.

```
Require: $\mathcal{A}_{\mathrm{R}} \in \mathcal{S}_{\mathrm{R}}$
  $\mathcal{A} \leftarrow \mathcal{A}_{\mathrm{R}}$
  $\mathcal{B} \leftarrow \emptyset$
  $\mathcal{C} \leftarrow \emptyset$
  for all $a_{\mathrm{R}}^{\mathrm{L}} \in \mathcal{A}$ do
    $\kappa \leftarrow \mathsf{MeetsCriterion}\left(a_{\mathrm{R}}^{\mathrm{L}}\right)$
    if $\kappa = 1$ then
      $\mathcal{B} \leftarrow \mathcal{B} \cup \left\{a_{\mathrm{R}}^{\mathrm{L}}\right\}$
    else
      $\mathcal{C} \leftarrow \mathcal{C} \cup \left\{a_{\mathrm{R}}^{\mathrm{L}}\right\}$
    end if
  end for
  return $\mathcal{B}, \mathcal{C}$
```

The input of the Algorithm 1 is the set $\mathcal{A}_{\mathrm{R}}$, hereafter internally referred to as the set $\mathcal{A}$. It is the set of the linear net elements of the reference structure $\mathcal{S}_{\mathrm{R}}$. For these net elements, the values of the parameters required by the classification criterion must be known (we assume that these values are part of the description of the individual net elements). What is next, the classification criterion must be defined.

In the notation of the Algorithm 1, the classification criterion is built in the MeetsCriterion function. The function is used to decide whether a classification criterion is met for a given input net element. In this case, it is supposed to be applied to each linear net element $a_{\mathrm{R}}^{\mathrm{L}}$ from the set $\mathcal{A}$. As output, the function returns the value `1` if the classification criterion is met and the value `0` if it is not met. The classification criterion itself needs to be defined for each network depending on its typical characteristics. In general, it can work with the values of various parameters related to the net elements of the set $\mathcal{A}$. These parameters can be, e.g., directly selected attributes of individual net elements, or they can express certain facts regarding the net entities localized to them. Depending on the specific application case, the classification criterion can be simple but also more complex, working with more than one such parameter.

All the linear net elements of the set $\mathcal{A}$ for which the classification criterion is met are classified as to be integrated into the non-linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$ and to be included in the set $\mathcal{B}$, while the other linear net elements of the set $\mathcal{B}$ are classified as to be integrated into the linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$ and to be included in the set $\mathcal{C}$.

The outputs of the Algorithm 1 are the sets $\mathcal{B}$ and $\mathcal{C}$.

An example of a simple classification criterion is the criterion which takes into account whether the value of the `length` attribute of these linear net elements is smaller than a certain determined value d or not. Assuming that we introduce the function Length, which returns the value of the attribute `length` of the specific linear net element $a^{\mathrm{L}}_{\mathrm{R}}$, which is its only input, as an output, we can define the classification criterion in the following way:

$$d < \mathit{Length}\left(a^{\mathrm{L}}_{\mathrm{R}}\right).$$

If we consider the structure $\mathcal{S}_{\mathrm{R}}$ as expressed in the Fig. 1, where the only linear net elements for which the value of the `length` attribute is greater than d are the net elements `01`, `15`, `29`, `43`, `46` and `47`, the Algorithm 1 will return the following sets $\mathcal{B}$ and $\mathcal{C}$:

$$\begin{aligned}\mathcal{B} = \{&\mathtt{02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12, 13, 14, 16, 17, 18, 19, 20, 21, 22, 23,}\\ &\mathtt{24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 44, 45, 48, 49}\},\end{aligned}$$

$$\mathcal{C} = \{\mathtt{01, 15, 29, 43, 46, 47}\}.$$

4.2 Integration into non-linear net elements

Further processing of the linear net elements of the set $\mathcal{B}$ is performed by the Integration into Non-Linear Net Elements Algorithm 2. It provides sorting the linear net elements from the set $\mathcal{B}$ into sets $\mathcal{D}_i$ using topological clustering and creating sets $\mathcal{A}^{\mathrm{N}}_{\mathrm{D}}$ and $\mathcal{U}^{\mathrm{N}}$ whose elements are based on them.

Each of the sets $\mathcal{D}_i$ is supposed to include the linear net elements of the set $\mathcal{B}$ of the reference structure $\mathcal{S}_{\mathrm{R}}$ that are intended to be integrated into one specific non-linear net element of the derived structure $\mathcal{S}_{\mathrm{D}}$. The set $\mathcal{A}^{\mathrm{N}}_{\mathrm{D}}$ is the set of all such non-linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$ created on the basis of the individual sets $\mathcal{D}_i$. The set $\mathcal{U}^{\mathrm{N}}$ is the set of unordered collections assigning each element of the set $\mathcal{B}$ to one specific element of the set $\mathcal{A}^{\mathrm{N}}_{\mathrm{D}}$ as part of it.

The input of the Algorithm 2 are the sets $\mathcal{B}$ and $\mathcal{R}_{\mathrm{R}}$. The set $\mathcal{B}$, obtained as the output of the Classification of Linear Net Elements Algorithm 1, is the set of linear net elements of the reference structure $\mathcal{S}_{\mathrm{R}}$ which are intended to be integrated into the non-linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$. The set $\mathcal{R}_{\mathrm{R}}$ is the set of positioned relations of the reference structure $\mathcal{S}_{\mathrm{R}}$.

Knowing the set $\mathcal{R}_{\mathrm{R}}$ is important for the correct functioning of the Adjacent-NetElements function. The function is used to find the net elements adjacent to the

Algorithm 2 Integration into Non-Linear Net Elements.

```
Require: $\mathcal{B}, \mathcal{R}_{\mathrm{R}} \in \mathcal{S}_{\mathrm{R}}$
  $i \leftarrow 0$
  $\mathcal{E} \leftarrow \emptyset$
  while $\mathcal{B} \neq \emptyset$ do
    $i \leftarrow i + 1$
    $\mathcal{D}_i \leftarrow \emptyset$
    $a_{\mathrm{R}}^{\mathrm{L}} \leftarrow a \in \mathcal{B}$
    repeat
      if $\mathcal{D}_i \neq \emptyset$ then
        $a_{\mathrm{R}}^{\mathrm{L}} \leftarrow a \in \mathcal{E}$
        $\mathcal{E} \leftarrow \mathcal{E} \setminus \left\{a_{\mathrm{R}}^{\mathrm{L}}\right\}$
      end if
      $\mathcal{D}_i \leftarrow \mathcal{D}_i \cup \left\{a_{\mathrm{R}}^{\mathrm{L}}\right\}$
      $\mathcal{B} \leftarrow \mathcal{B} \setminus \left\{a_{\mathrm{R}}^{\mathrm{L}}\right\}$
      $\mathcal{E} \leftarrow \mathcal{E} \cup \left(\mathsf{AdjacentNetElements}\left(a_{\mathrm{R}}^{\mathrm{L}}, \{\mathtt{0}, \mathtt{1}\}, \{\mathtt{both}, \mathtt{none}\}, \mathcal{R}_{\mathrm{R}}\right) \cap \mathcal{B}\right)$
    until $\mathcal{E} = \emptyset$
  end while
  $k \leftarrow i$
  $\mathcal{A}_{\mathrm{D}}^{\mathrm{N}} \leftarrow \emptyset$
  $\mathcal{U}^{\mathrm{N}} \leftarrow \emptyset$
  for $i \leftarrow 1$ to $k$ do
    $a_{\mathrm{D}}^{\mathrm{N}} \leftarrow \mathsf{CreateNonLinearElementOfDerivedStructure}$
    $\mathcal{A}_{\mathrm{D}}^{\mathrm{N}} \leftarrow \mathcal{A}_{\mathrm{D}}^{\mathrm{N}} \cup \left\{a_{\mathrm{D}}^{\mathrm{N}}\right\}$
    $u^{\mathrm{N}} \leftarrow \mathsf{CreateUnorderedCollection}\left(a_{\mathrm{D}}^{\mathrm{N}}, \mathcal{D}_i\right)$
    $\mathcal{U}^{\mathrm{N}} \leftarrow \mathcal{U}^{\mathrm{N}} \cup \left\{u^{\mathrm{N}}\right\}$
  end for
  return $\mathcal{A}_{\mathrm{D}}^{\mathrm{N}}, \mathcal{U}^{\mathrm{N}}$
```

input net element. In this case of using the function, it is repeatedly applied to the input linear net elements denoted as $a_{\mathrm{R}}^{\mathrm{L}}$. To do this, the function needs to know the individual relevant net relations. This function is designed to only consider the input positioned relations of required attribute values. The inputs of the function are the input net element the adjacent net elements of which are to be found, list of permissible values of the binding positions on the input net element to which the searched adjacent net elements can be bound, list of permissible values of the `navigability` attribute and the set of positioned relations to be concerned. The output of the function is the set of the found adjacent net elements.

Since in this case of using the function the adjacent net elements can be bound to the input element $a_{\mathrm{R}}^{\mathrm{L}}$ both at the beginning and at the end, we enter the values `0` and `1` as the permissible values of the binding positions. Since we are concerned with the physical connections (not only the functional ones) when searching for the adjacent elements and we consider values of the `navigability` attribute `both` or `none`, we enter both these values as the permissible values of this attribute, in this case.

By gradually expanding clusters of interconnected linear net elements originating from the set $\mathcal{B}$, the sets $\mathcal{D}_i$ are created. The index i takes the value of a natural number from 1 to k, where k is the total number of clusters that are created by this procedure using all the linear net elements of the set $\mathcal{B}$. Each of the sets $\mathcal{D}_i$ then contains the linear net elements of the reference structure $\mathcal{S}_\mathrm{R}$ the integration of which is supposed to result in one specific non-linear net element of the derived structure $\mathcal{S}_\mathrm{D}$.

Based on each of the sets $\mathcal{D}_i$, one non-linear net element a_D^N of the derived structure $\mathcal{S}_\mathrm{D}$ is created, subsequently. This is provided by the CreateNonLinearElementOfDerivedStructure function. As used here, this function has no inputs and creates the non-linear net element a_D^N that the function returns as its output. If desired, this function can be modified to assign the values of certain parameters to the created non-linear net element. Such parameters can express, e.g., the number of the linear net elements of the reference structure $\mathcal{S}_\mathrm{R}$, by the integration of which the non-linear net element of the derived structure $\mathcal{S}_\mathrm{D}$ is created (similarly to the Integration into Linear Net Elements and Creation of Positioned Relations Algorithm 3), the length of the longest of them, etc. Each non-linear net element a_D^N created in this way is supposed to be added to the set $\mathcal{A}_\mathrm{D}^\mathrm{N}$, which is the set of non-linear net elements of the derived structure $\mathcal{S}_\mathrm{D}$.

In addition to the non-linear net elements of the derived structure $\mathcal{S}_\mathrm{D}$, it is necessary to create the unordered collections based on each of the sets $\mathcal{D}_i$. These make it possible to express the interconnectedness between the net elements of the reference structure $\mathcal{S}_\mathrm{R}$ and the net elements of the derived structure $\mathcal{S}_\mathrm{D}$. Their creation is provided by the CreateUnorderedCollection function. The input of the function is the net element in the role of composition net element and a set of net elements in the role of parts of which the composition element consists. The output of the function is the created unordered collection. In this case of using the function, it requires to know the non-linear net element a_D^N and the particular set $\mathcal{D}_i$ on the basis of which this net element was created. Each unordered collection u^N created in this way is supposed to be added to the set $\mathcal{U}^\mathrm{N}$.

The outputs of the Algorithm 2 are the sets $\mathcal{A}_\mathrm{D}^\mathrm{N}$ and $\mathcal{U}^\mathrm{N}$.

If we apply the Algorithm 2 to the considered example with the reference structure $\mathcal{S}_\mathrm{R}$ expressed in the Fig. 1, we obtain the following $\mathcal{D}_i$ sets:

$$\mathcal{D}_1 = \{\mathsf{02},\mathsf{03},\mathsf{04},\mathsf{05},\mathsf{06},\mathsf{07},\mathsf{08},\mathsf{09},\mathsf{10},\mathsf{11},\mathsf{12},\mathsf{13},\mathsf{14}\},$$

$$\mathcal{D}_2 = \{\mathsf{16},\mathsf{17},\mathsf{18},\mathsf{19},\mathsf{20},\mathsf{21},\mathsf{22},\mathsf{23},\mathsf{24},\mathsf{25},\mathsf{26},\mathsf{27},\mathsf{28}\},$$

$$\mathcal{D}_3 = \{\mathsf{30},\mathsf{31},\mathsf{32},\mathsf{33},\mathsf{34},\mathsf{35},\mathsf{36},\mathsf{37},\mathsf{38},\mathsf{39},\mathsf{40},\mathsf{41},\mathsf{42}\},$$

$$\mathcal{D}_4 = \{\mathsf{44},\mathsf{45}\},$$

$$\mathcal{D}_5 = \{\mathsf{48},\mathsf{49}\}.$$

Based on each of the $\mathcal{D}_i$ sets, the Algorithm 2 further creates a non-linear net element of the derived structure $\mathcal{S}_\mathrm{D}$. If we follow the naming of the corresponding net elements in the Fig. 2, the output set $\mathcal{A}_\mathrm{D}^\mathrm{N}$ will be written as follows:

$$\mathcal{A}_\mathrm{D}^\mathrm{N} = \{\mathtt{II},\mathtt{IV},\mathtt{VI},\mathtt{VIII},\mathtt{X}\}.$$

In case we express an unordered collection as a tuple of a net element of the derived structure $\mathcal{S}_{\mathrm{D}}$ and the set of corresponding net elements of the reference structure $\mathcal{S}_{\mathrm{R}}$, we can write the output set $\mathcal{U}^{\mathrm{N}}$ as follows:

$$\mathcal{U}^{\mathrm{N}} = \{(\mathtt{II}, \mathcal{D}_1), (\mathtt{IV}, \mathcal{D}_2), (\mathtt{VI}, \mathcal{D}_3), (\mathtt{VIII}, \mathcal{D}_4), (\mathtt{X}, \mathcal{D}_5)\}.$$

4.3 Integration into linear net elements and creation of positioned relations

Further processing of the linear net elements of the set $\mathcal{C}$ is performed by the integration into linear net elements and creation of positioned relations Algorithm 3. It provides sorting the linear net elements from the set $\mathcal{C}$ into sets $\mathcal{I}_i$ in such way that respects their connection with the linear net elements integrated into the same non-linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$ and creating sets $\mathcal{A}^{\mathrm{L}}_{\mathrm{D}}$, $\mathcal{U}^{\mathrm{L}}$ and $\mathcal{R}^{\mathrm{D}}$ the elements of which are based on them.

Each of the sets $\mathcal{I}_i$ is supposed to include the linear net elements of the set $\mathcal{C}$ of the reference structure $\mathcal{S}_{\mathrm{R}}$ that are intended to be integrated into one specific linear net element of the derived structure $\mathcal{S}_{\mathrm{D}}$. The set $\mathcal{A}^{\mathrm{L}}_{\mathrm{D}}$ is the set of all such linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$ created on the basis of the individual sets $\mathcal{I}_i$. The set $\mathcal{U}^{\mathrm{L}}$ is the set of unordered collections assigning each element of the set $\mathcal{C}$ to one specific element of the set $\mathcal{A}^{\mathrm{L}}_{\mathrm{D}}$ as part of it. The set $\mathcal{R}^{\mathrm{D}}$ is the set of the positioning relations of the derived structure $\mathcal{S}_{\mathrm{D}}$ created within this Algorithm 3 in order to connect the net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$ accordingly.

The input of the Algorithm 3 are the sets $\mathcal{C}$, $\mathcal{R}_{\mathrm{R}}$ and $\mathcal{U}^{\mathrm{N}}$. The set $\mathcal{C}$, obtained as the output of the Classification of Linear Net Elements Algorithm 1, is the set of the linear net elements of the reference structure $\mathcal{S}_{\mathrm{R}}$ which are intended to be integrated into the linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$. The set $\mathcal{R}_{\mathrm{R}}$ is the set of the positioned relations of the reference structure $\mathcal{S}_{\mathrm{R}}$.

Knowing the set $\mathcal{R}_{\mathrm{R}}$ is important for the correct functioning of the AdjacentNetElements function, again. This time we use it repeatedly to search separately for the net elements adjacent to the beginning and to the end of the input linear net elements denoted as $a^{\mathrm{L}}_{\mathrm{R}}$. In the case of using the function, when the searched net elements are required to be connected to the beginning of the input linear net element, we enter the value `0` as the permissible value of the binding position. In the case of using the function, when the searched net elements are required to be connected to the end of the input linear net element, we enter the value `1` as the permissible value of the binding position. Since we are concerned with the functional connection of net elements of net elements when searching for the adjacent net elements, we enter only the value `both` as the permissible value of the `navigability` attribute in both of these cases.

Knowing the set $\mathcal{U}^{\mathrm{N}}$ is required by the SetOfCompositionNetElements function. The function is used to find the net elements the parts of which are the elements of the input set of net elements. The inputs of the function are the set of input net elements in the role of parts and the set of element part collections to be concerned. The output of this function is the set of the found net elements in the role of composition net elements.

We gradually apply this function to the set of net elements functionally adjacent to the beginning and the end of each linear net element of the set $\mathcal{C}$. With a

Algorithm 3 Integration into Linear Net Elements and Creation of Positioned Relations.

```
Require: $\mathcal{C}, \mathcal{R}_{\mathrm{R}} \in \mathcal{S}_{\mathrm{R}}, \mathcal{U}^{\mathrm{N}}$
  $c \leftarrow 1$
  $j \leftarrow 1$
  for all $a^{\mathrm{L}}_{\mathrm{R}} \in \mathcal{C}$ do
    $\mathcal{H}^0_j \leftarrow \emptyset$
    $\mathcal{H}^1_j \leftarrow \emptyset$
    $\mathcal{F}^0 \leftarrow \mathsf{AdjacentNetElements}\left(a^{\mathrm{L}}_{\mathrm{R}}, \{\mathtt{0}\}, \{\mathtt{both}\}, \mathcal{R}_{\mathrm{R}}\right)$
    $\mathcal{F}^1 \leftarrow \mathsf{AdjacentNetElements}\left(a^{\mathrm{L}}_{\mathrm{R}}, \{\mathtt{1}\}, \{\mathtt{both}\}, \mathcal{R}_{\mathrm{R}}\right)$
    $\mathcal{G}^0 \leftarrow \mathsf{SetOfCompositionNetElements}\left(\mathcal{F}^0, \mathcal{U}^{\mathrm{N}}\right)$
    $\mathcal{G}^1 \leftarrow \mathsf{SetOfCompositionNetElements}\left(\mathcal{F}^1, \mathcal{U}^{\mathrm{N}}\right)$
    $i \leftarrow 0$
    repeat
      $i \leftarrow i + 1$
    until $i = j$ or $\mathcal{G}^0 \cup \mathcal{G}^1 = \mathcal{H}^0_i \cup \mathcal{H}^1_i$
    if i = j then
      $\mathcal{H}^0_i \leftarrow \mathcal{G}^0$
      $\mathcal{H}^1_i \leftarrow \mathcal{G}^1$
      $\mathcal{I}_i \leftarrow \left\{a^{\mathrm{L}}_{\mathrm{R}}\right\}$
      $t_i \leftarrow 1$
      $j \leftarrow j + 1$
    else
      $\mathcal{I}_i \leftarrow \mathcal{I}_i \cup \left\{a^{\mathrm{L}}_{\mathrm{R}}\right\}$
      $t_i = t_i + 1$
    end if
  end for
  $k \leftarrow j - 1$
  $\mathcal{A}^{\mathrm{L}}_{\mathrm{D}} \leftarrow \emptyset$
  $\mathcal{U}^{\mathrm{L}} \leftarrow \emptyset$
  $\mathcal{R}_{\mathrm{D}} \leftarrow \emptyset$
  for $i \leftarrow 1$ to $k$ do
    $a^{\mathrm{L}}_{\mathrm{D}} \leftarrow \mathsf{CreateLinearElementOfDerivedStructure}\left(t_i\right)$
    $\mathcal{A}^{\mathrm{L}}_{\mathrm{D}} \leftarrow \mathcal{A}^{\mathrm{L}}_{\mathrm{D}} \cup \left\{a^{\mathrm{L}}_{\mathrm{D}}\right\}$
    $u^{\mathrm{L}} \leftarrow \mathsf{CreateUnorderedCollection}\left(a^{\mathrm{L}}_{\mathrm{D}}, \mathcal{I}_i\right)$
    $\mathcal{U}^{\mathrm{L}} \leftarrow \mathcal{U}^{\mathrm{L}} \cup \left\{u^{\mathrm{L}}\right\}$
    for all $a^{\mathrm{N}}_{\mathrm{D}} \in \mathcal{H}^0_i$ do
      $r_{\mathrm{D}} \leftarrow \mathsf{CreatePositionedRelationOfDerivedStructure}\left(a^{\mathrm{L}}_{\mathrm{D}}, a^{\mathrm{N}}_{\mathrm{D}}, \mathtt{0}, \mathtt{0}, \mathtt{both}\right)$
      $\mathcal{R}_{\mathrm{D}} \leftarrow \mathcal{R}_{\mathrm{D}} \cup \{r_{\mathrm{D}}\}$
    end for
    for all $a^{\mathrm{N}}_{\mathrm{D}} \in \mathcal{H}^1_i$ do
      $r_{\mathrm{D}} \leftarrow \mathsf{CreatePositionedRelationOfDerivedStructure}\left(a^{\mathrm{L}}_{\mathrm{D}}, a^{\mathrm{N}}_{\mathrm{D}}, \mathtt{1}, \mathtt{0}, \mathtt{both}\right)$
      $\mathcal{R}_{\mathrm{D}} \leftarrow \mathcal{R}_{\mathrm{D}} \cup \{r_{\mathrm{D}}\}$
    end for
  end for
  return $\mathcal{A}^{\mathrm{L}}_{\mathrm{D}}, \mathcal{U}^{\mathrm{L}}, \mathcal{R}_{\mathrm{D}}$
```

correctly designed reference structure $\mathcal{S}_{\mathrm{R}}$, it is possible to assume that each of the sets obtained in this way has at most one element. Under this assumption, the union of the sets obtained for the beginning and for the end of a specific linear net element $a^{\mathrm{L}}_{\mathrm{R}}$ has at most two elements.

By gradually expanding clusters of the linear net elements originating from the set $\mathcal{C}$ for which the unions are equal, the sets $\mathcal{I}_i$ are created. The index i takes the value of a natural number from 1 to k, where k is the total number of clusters that are created by this procedure using all the linear net elements of the set $\mathcal{C}$. Each of the sets $\mathcal{I}_i$ then contains the linear net elements of the reference structure $\mathcal{S}_{\mathrm{R}}$ the integration of which is supposed to result in one specific linear net element of the derived structure $\mathcal{S}_{\mathrm{D}}$. In practical examples, it is not assumed that there are many elements in the individual sets $\mathcal{I}_i$, as they express the individual tracks of the respective line sections. Their number is continuously calculated for each of the sets $\mathcal{I}_i$ and is expressed as the resulting value of the respective parameter t_i.

Based on each of the sets $\mathcal{I}_i$, one linear net element $a^{\mathrm{L}}_{\mathrm{D}}$ of the derived structure $\mathcal{S}_{\mathrm{D}}$ is created, subsequently. This is provided by the CreateLinearElementOfDerived-Structure function. As used here, the only input of this function is the t_i parameter the value of which is to be assigned to the created linear net element $a^{\mathrm{N}}_{\mathrm{D}}$ that the function returns as its output. If desired, this function can be modified to assign no parameter value (similarly to the Integration into Non-Linear Net Elements Algorithm 2) or to assign other parameter values to the created linear net element (similarly to what is stated in the description of the Algorithm 2). Each linear net element $a^{\mathrm{L}}_{\mathrm{D}}$ created in this way is supposed to be added to the set $\mathcal{A}^{\mathrm{L}}_{\mathrm{D}}$, which is the set of the linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$.

In addition to the linear net elements of the derived structure $\mathcal{S}_{\mathrm{D}}$, it is necessary to create unordered collections based on each of the sets $\mathcal{I}_i$. The CreateUnordered-Collection function is again used to do this. In this case of using the function, it requires to know the linear net element $a^{\mathrm{L}}_{\mathrm{D}}$ and the particular set $\mathcal{I}_i$ on the basis of which this net element was created. Each unordered collection u^{L} created in this way is supposed to be added to the set $\mathcal{U}^{\mathrm{L}}$.

In order to complete the description of the derived structure $\mathcal{S}_{\mathrm{D}}$, it is also necessary to create the positioned relations of this structure. This is provided by the CreatePositionedRelationOfDerivedStructure function. The inputs of the function are the net element of the derived structure $\mathcal{S}_{\mathrm{D}}$ in the role A, the net element of the derived structure $\mathcal{S}_{\mathrm{D}}$ in the role B, binding position on the net element in the role A, binding position on the net element in the role B and navigability of the created positioned relation. The output of the function is the created positioned relation. The function is applied to each linear net element $a^{\mathrm{L}}_{\mathrm{D}}$ of the structure $\mathcal{S}_{\mathrm{D}}$ in order to connect its beginning and its end with the corresponding non-linear net elements $a^{\mathrm{N}}_{\mathrm{D}}$ of the structure $\mathcal{S}_{\mathrm{D}}$. If the net element $a^{\mathrm{L}}_{\mathrm{D}}$ appears in the role A and the net element $a^{\mathrm{L}}_{\mathrm{D}}$ in the role B, the value of the binding position on the net element in the role A can be `0` or `1` depending on the case of using the function (it depends on whether the created positioned relation is to be bound to the beginning or the end of the linear net element $a^{\mathrm{L}}_{\mathrm{D}}$), while the value of the binding position on the net element in the role B is `0` (we do not expect to distinguish any other position value on non-linear net element) and the value of the `navigability` attribute is `both` (to create a navigable positioned relation). Each positioned relation r_{D} created in this way is supposed to be added to the set $\mathcal{R}^{\mathrm{R}}$.

The outputs of the Algorithm 3 are the sets $\mathcal{A}_{\mathrm{D}}^{\mathrm{L}}$, $\mathcal{U}^{\mathrm{L}}$ and $\mathcal{R}^{\mathrm{R}}$.

If we apply the Algorithm 3 to the considered example with the reference structure $\mathcal{S}_{\mathrm{R}}$ expressed in the Fig. 1, we obtain the following $\mathcal{I}_i$ sets with the respective values of the corresponding t_i parameters:

$$\mathcal{I}_1 = \{\mathtt{01}\},\ \mathcal{I}_2 = \{\mathtt{15}\},\ \mathcal{I}_3 = \{\mathtt{29}\},\ \mathcal{I}_4 = \{\mathtt{43}\},\ \mathcal{I}_5 = \{\mathtt{46}, \mathtt{47}\},$$

$$t_1 = 1,\ t_2 = 1,\ t_3 = 1,\ t_4 = 1,\ t_5 = 2.$$

Based on each of the $\mathcal{I}_i$ sets, the Algorithm 3 further creates a linear net element of the derived structure $\mathcal{S}_{\mathrm{D}}$. For the already considered approach and a positioned relation expressed as a set of the respective interconnected net elements, the corresponding output sets $\mathcal{A}_{\mathrm{D}}^{\mathrm{L}}$, $\mathcal{U}^{\mathrm{L}}$ and $\mathcal{R}_{\mathrm{D}}$ can be written in the following way:

$$\mathcal{A}_{\mathrm{D}}^{\mathrm{L}} = \{\mathtt{I}, \mathtt{III}, \mathtt{V}, \mathtt{VII}, \mathtt{IX}\},$$

$$\mathcal{U}^{\mathrm{L}} = \{(\mathtt{I}, \mathcal{I}_1), (\mathtt{III}, \mathcal{I}_2), (\mathtt{V}, \mathcal{I}_3), (\mathtt{VII}, \mathcal{I}_4), (\mathtt{IX}, \mathcal{I}_5)\},$$

$$\begin{aligned}\mathcal{R}_{\mathrm{D}} = \{&(\mathtt{I}, \mathtt{X}), (\mathtt{I}, \mathtt{II}), (\mathtt{III}, \mathtt{II}), (\mathtt{III}, \mathtt{IV}), (\mathtt{V}, \mathtt{IV}),\\ &(\mathtt{V}, \mathtt{VI}), (\mathtt{VII}, \mathtt{VI}), (\mathtt{VII}, \mathtt{VIII}), (\mathtt{IX}, \mathtt{VIII}), (\mathtt{IX}, \mathtt{X})\}.\end{aligned}$$

4.4 Summary of outputs

Using the outputs of individual sub-algorithms, we can express the final outputs in the following way:

$$\mathcal{S}_{\mathrm{D}} = (\mathcal{A}_{\mathrm{D}}, \mathcal{R}_{\mathrm{D}}) = \left(\mathcal{A}_{\mathrm{D}}^{\mathrm{N}} \cup \mathcal{A}_{\mathrm{D}}^{\mathrm{L}}, \mathcal{R}_{\mathrm{D}}\right),$$

where the set $\mathcal{A}_{\mathrm{D}}^{\mathrm{N}}$ comes from the Algorithm 2 and the sets $\mathcal{A}_{\mathrm{D}}^{\mathrm{L}}$ and $\mathcal{R}_{\mathrm{D}}$ come from the Algorithm 3.

$$\mathcal{U} = \mathcal{U}^{\mathrm{L}} \cup \mathcal{U}^{\mathrm{N}},$$

where the set $\mathcal{U}^{\mathrm{N}}$ comes from the Algorithm 2 and the set $\mathcal{U}^{\mathrm{L}}$ comes from the Algorithm 3.

5. Conclusions

The paper presented the proposal of the transformation algorithm used to transform the topological layer structure expressed at the $\mathtt{micro}_{\mathtt{L}_2}$ level of detail into the topological layer structure expressed at the $\mathtt{macro}_{\mathtt{N}_0,\mathtt{L}_0}$ level of detail. For these needs it also summarized the expected features of these structures and related requirements. Applying the algorithm to structures of the type shown in the Fig. 1 shows that the transformation algorithm yields the expected results.

An undesirable result could occur if the algorithm were applied to such a network in which two operational points are directly connected by several different line sections or if several line sections are not terminated by operational points on both sides. In such cases, the respective tracks could be integrated into line sections in an unsatisfactory manner. Taking these specific cases into account can be a topic for further development of the transformation algorithm.

Of course, the correctness of the result also depends on the values of the parameters used by the classification criterion and their assignment to individual net elements of the reference structure $\mathcal{S}_{\mathrm{R}}$, as well as on the design of the classification criterion itself. Another topic for automation in this area can be the development of a machine learning algorithm used to design a classification criterion for a certain railway network based on a training data set, where the individual selected net elements of the reference structure $\mathcal{S}_{\mathrm{R}}$ are classified in advance based on expert knowledge.

References

[1] BISCHOF S., SHENNER G. Rail Topology Ontology: A Rail Infrastructure Base Ontology. In: *The Semantic Web – ISWC 2021: 20th International Semantic Web Conference, ISWC 2021, Virtual Event, October 24–28, 2021, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2021, pp. 597–612, doi: 10.1007/978-3-030-88361-4_35.

[2] BOLLIG Y.C. *Geometrical and Topological Linking of Railway Systems* [online]. München, 2020. Master's thesis, Ingenieurfakult at Bau Geo Umwelt [viewed 2022-11-14]. Available from: https://publications.cms.bgu.tum.de/theses/2020_Bollig_Esser.pdf

[3] HLUBUČEK A. Význam popisu infrastruktury pro inteligentní dopravní systémy na železnici. *Vědeckotechnický sborník ČD*. 2016, 42/2016, pp. 1–9, ISSN: 1214-9047.

[4] INTERNATILONAL UNION OF RAILWAYS (UIC). *IRS 30100: RailTopoModel – Railway Infrastructure Topological Model*. 1st edition. Paris: International Union of Railways (UIC), 2016. ISBN: 978-2-7461-2513-1.

[5] RAILML.ORG. *railML 3.2* [online]. railML.org, 2022-04-26 [viewed 2022-11-14]. Available from: https://www.railml.org/en/download/schemes.html

[6] RAILML.ORG. *RailTopoModel v1.4* [online]. railML.org, 2022-04-29 [viewed 2022-09-19]. Available from: https://www.railtopomodel.org/download-rtm.html

[7] RAILTOPOMODEL. State of Development. In: *RailTopoModel* [online]. RailTopoModel, ©2022 [viewed 2022-11-14]. Available from: https://www.railtopomodel.org/state-of-development.html