TCCS SD1 - Data Model - Train Protection

SPT2TS-127384 - Disclaimer: The data model defined here is a DRAFT version, developed from bottom up inputs as per approaches defined in previous European projects, and from ongoing implementations in Innovation Pillar FPs. The content defined here shall not be considered as 'finalized' and is still a work in progress with the respective system pillar domains. **[** Content to be approved **]**

1 Table of Contents

2 Package "Train Protection"

2.1 Package Header

SPT2TS-122298 - Package header

```
{
  "$schema": "ERJU meta-model.json",
  "isDefinedBy": "http://ERJU/datamodel/0.4/trainprotection",
  "name": "trainprotection",
  "containerStruct": "TrainProtection",
  "prefix": "tp",
  "intId": 4,
  "version": "1.0",
  "info": "Data model for Train Protection use case",
    "enums": [],     "structs": []
}
```

2.2 Drive Protection Section (Group)

SPT2TS-100761 - A Drive Protection Section (DPS) represents a track section that can be used to set different states to ensure the movement of trains for a particular route. In general, it is an abstraction of track sections on the railway network that might adapt different states corresponding to the position of controllable trackside assets (e.g., points or level crossings).
A Drive Protection Section Group (DPS Group) groups the DPSs. The DPSs within a DPS Group can only change their state depending on each other, meaning that a request for a different trafficability state on one DPS will affect any other DPS(s) in the group.

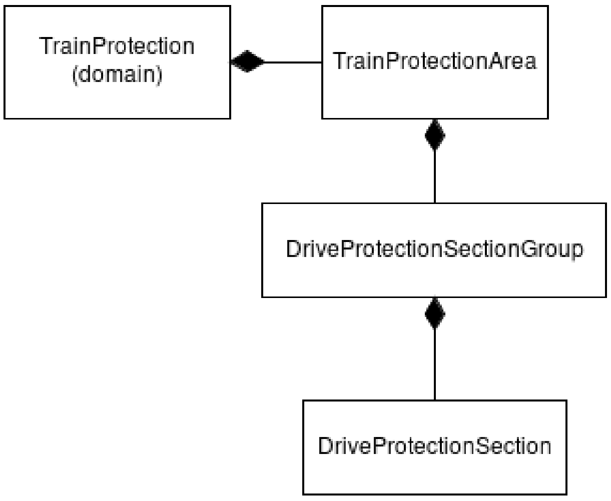


Figure 1 Class Diagram of Drive Protection Section

Figure 2 Illustration of Drive Protection Section

The following equivalent classes from other models can be referred to:

- Drive Protection Section and Drive Protection Section Group in SPT2TS-7 - RCA Digital Map – MAP Object Catalogue [RCA.Doc.69]

The attribute 'dependencies' provides the specific inter dependencies between the drive protection sections within a drive protection section group. These inter dependencies are defined either generic as in EXCLUSIVE or EQUIVALENT relations or SPECIFIC in the form of a matrix which can be then built up as arrays in the formal specification. EXCLUSIVE means only one DPS within a DPSG containing 2 DPS is usable simultaneously. If a DPSG contains more than 2 DPS (i.e., 4 in case of Double Slip Crossing) then a SPECIFIC interdependency is defined (see example below). EQUIVALENT means all DPS within a DPSG are either usable or not usable simultaneously. As example the inter dependencies for level crossing, simple point, and slip crossing, crossing, and derailer are provided.

Examples for EQUIVALENT and EXCLUSIVE:

Table 1 Inter dependencies for DPSs within a DPSG of specific protected infrastructure elements

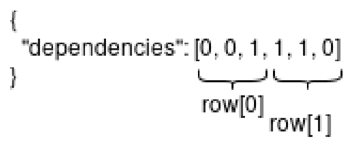
ProtectedInfraElement	Drive Protection Section Interdependency
Simple Point	EXCLUSIVE with 2 Drive Protection Sections
derailer	EXCLUSIVE with 1 Drive Protection Section
levelCrossing	EQUIVALENT with n Drive Protection Sections
crossing	EXCLUSIVE with 2 Drive Protection Sections

Examples for SPECIFIC:

Table 2 Inter dependency matrix for 3 DPSs within a DPSG

driveProtectionSections[0]	driveProtectionSections[1]	driveProtectionSections[2]
0	0	1
1	1	0

Example:

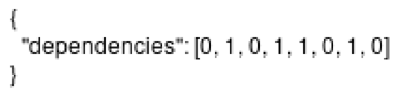


```
{
  "dependencies": [0, 0, 1, 1, 1, 0]
}
```

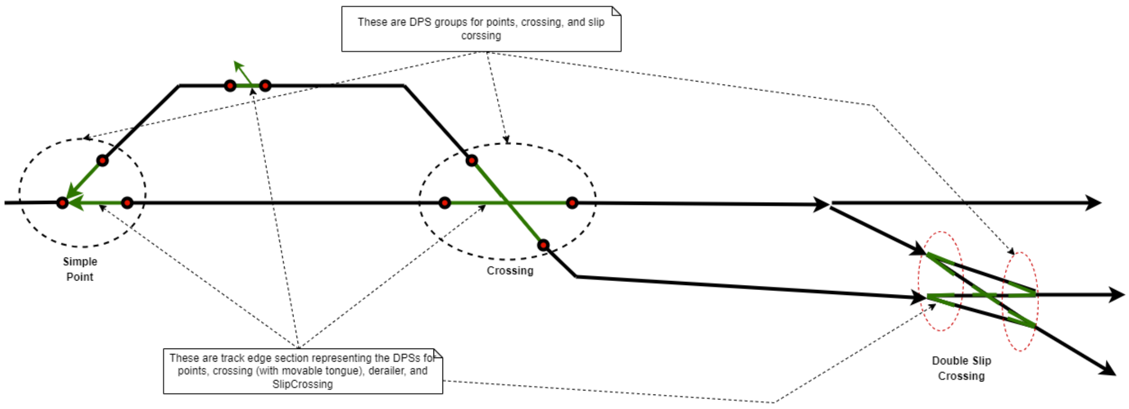
Table 3 Inter dependency matrix for 4 DPSs within a DPSG for a Double Slip Crossing and Single slip crossing with two simple points

driveProtectionSections[0]	driveProtectionSections[1]	driveProtectionSections[2]	driveProtectionSections[3]
0	1	0	1
1	0	1	0

Example:



```
{
  "dependencies": [0, 1, 0, 1, 1, 0, 1, 0]
}
```


These are DPS groups for points, crossing, and slip corssing
Simple Point
Crossing
Double Slip Crossing
These are track edge section representing the DPSs for points, crossing (with movable tongue), derailer, and SlipCrossing

[Content to be approved]

SPT2TS-100760 - DriveProtectionSectionGroup

```
{
"structs" : [
{
  "name": "ProtectedInfraElement",
  "union": true,
  "attrs": [
    {"intId": 1, "name": "simplePoint", "reference": "infra.SimplePoint"},
    {"intId": 2, "name": "slipCrossing", "reference": "infra.SlipCrossing"},
    {"intId": 3, "name": "crossing", "reference": "infra.Crossing"},
    {"intId": 4, "name": "derailer", "reference": "infra.Derailer"},
    {"intId": 5, "name": "levelCrossing", "reference": "infra.LevelCrossing"}
  ]
},
{
"name":"DriveProtectionSectionGroup",
"info": "Defines the configuration of a DPS Group",
"attrs":[
  {"intId":1,"name":"id","dataType":"string","key": "global", "info": "Identity of the object; used for referencing"},
  {"intId":2,"name":"name","dataType":"string"},
  {"intId":3,"name":"driveProtectionSections","composition":"DriveProtectionSection", "multiplicity": "1..*"},
  {"intId":4,"name":"protectedInfraElements","composition":"ProtectedInfraElement"},
  {"intId":5,"name":"dependenciesType","enumType":"DPSDependencyType", "info": "defines dependencies between n drive protection sections"},
  {"intId":6,"name":"dpsgDependencies","dataType":"uint32", "multiplicity": "0..*", "info": "contains row-wise bit matrix with column = index of the element in DriveProtectionSectionGroup. 0 means not allowed to use ; 1 means allowed to use for rail traffic. Each line in the matrix contains a valid state of the
DriveProtectionSectionGroup ; Use only if DPSDependencyType == Specific"},
 {"intId": 7, "name": "runTime", "dataType": "uint32", "unit": "s", "exp": -1, "info": "Defines the total time
required for the DPS(s) in a DPS Group to switch from one position to the other"}
]
},
{
"name":"DriveProtectionSection",
"info": "Defines a DPS for a section of track",
"attrs":[
   {"intId":1,"name":"id","dataType":"string","key": "global", "info": "Identity of the object; used for
referencing"},
   {"intId":2,"name":"name","dataType":"string"},
   {"intId":3,"name":"trackEdgeSection","composition":"infra.TrackEdgeSection"},
   {"intId": 4, "name": "maxFlankProtectionSpeed", "dataType": "uint32", "unit": "km/h"}
 ]
}
],
"enums": [
       {
        "name": "DPSDependencyType",
        "info": "Defines the type of DPS dependency",
         "enumLiterals": [
            { "intId": 0, "name": "Exclusive" },
            { "intId": 1, "name": "Equivalent" },
            { "intId": 2, "name": "Specific" }
           ]
        }
   ]
}
```

2.3 Allocation Section

SPT2TS-100762 - Allocation Sections are defined where one or more clearance gauge conflicts between different tracks arise. The conflict arises when the clearance gauges of different tracks overlap each other. Out of this conflict, an exclusive and symmetric inter-dependency between two or more Allocation Sections can be deduced. The attribute 'isDependentOnAllocationSection' provides an exclusive dependency to other Allocation Sections. The dependency refers to the adjacent allocation sections, which might be affected by a moveable object or movement permission extent when the current allocation section is fully or partially occupied by a moveable object or movement permission extent.

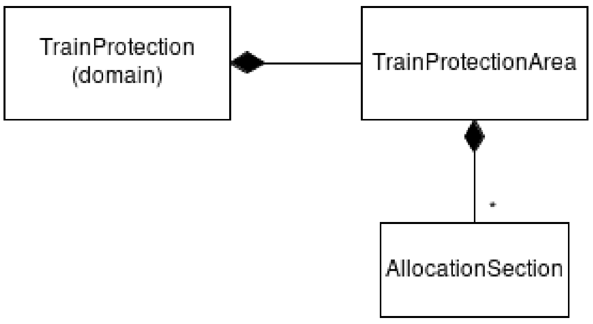


Figure 3 Class Diagram of Allocation Section

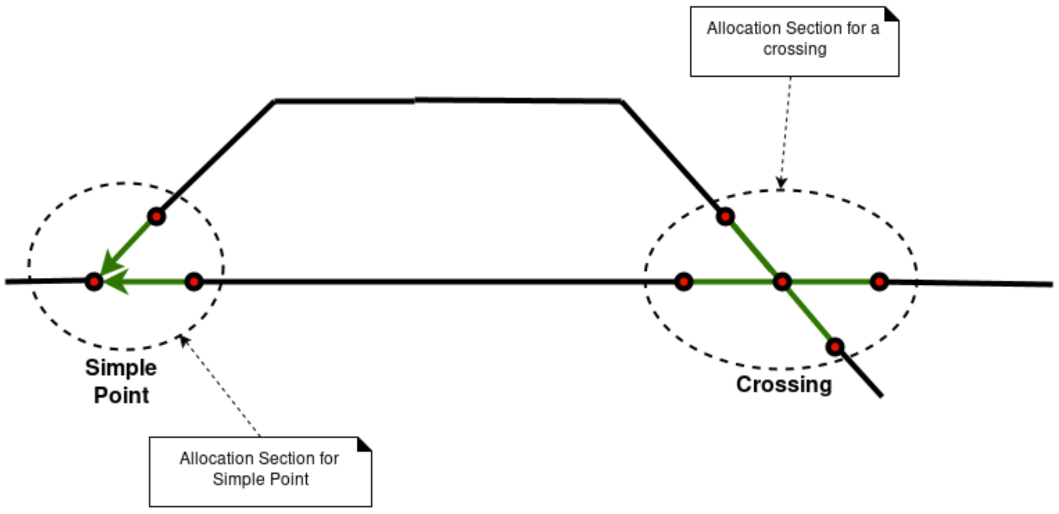

Allocation Section for a crossing
Simple Point
Crossing
Allocation Section for Simple Point

Figure 4 Illustration for Allocation Section

The following equivalent classes from other models can be referred to:

- Allocation Section in SPT2TS-7 - RCA Digital Map – MAP Object Catalogue [RCA.Doc.69]

[Content to be approved **]**

SPT2TS-48922 - AllocationSection

{
  "structs": [
  {
    "name":"AllocationSection",
    "info": "Defines an allocation section for a section of track",
    "attrs":[
      {"intId":1,"name":"id","dataType":"string","key": "global", "info": "Identity of the object; used for
referencing"},
      {"intId":2,"name":"name","dataType":"string"},
      {"intId":3,"name":"linearLocation","composition":"infra.LinkedPath"},
      {"intId":4,"name":"dependencies","reference":"AllocationSection", "multiplicity": "1..*", "info": "is

dependent on the allocation sections"}
    ]
  }]
}

2.4 Field Object Controller

SPT2TS-124028 - Field Object Controller is a device that exchanges commands, information with one or a few individual field objects like Level Crossing, point, etc.

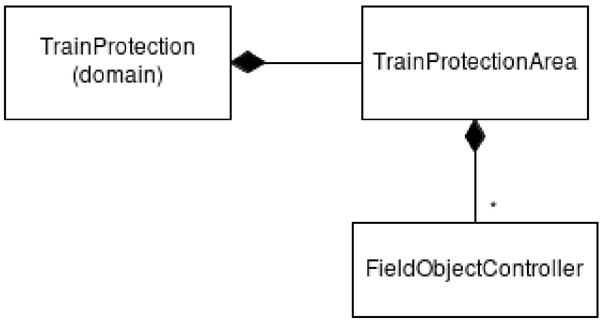


Figure 5 Class diagram for Field Object Controller

The following equivalent classes from other models can be referred to:

- Field Object Controller in https://eulynx.eu/index.php/dataprep

[Content to be approved **]**

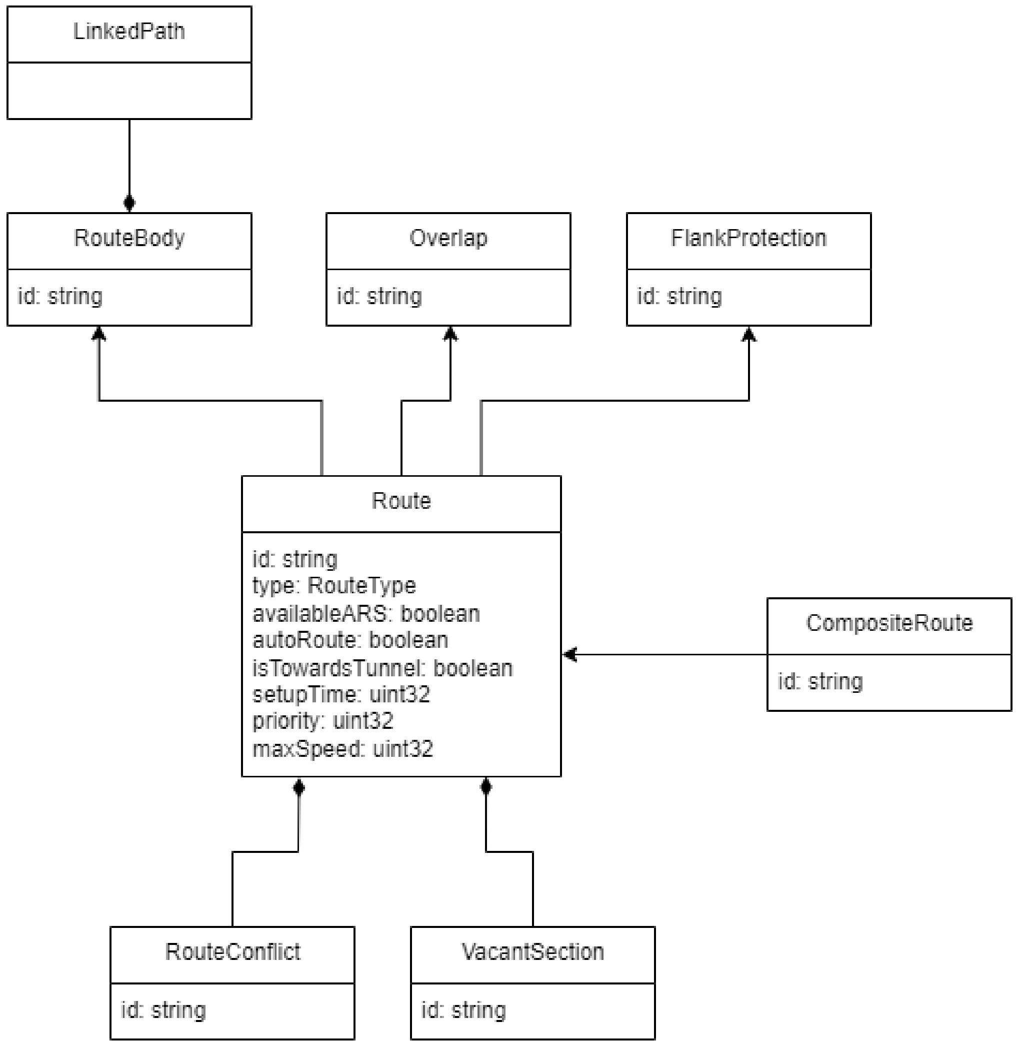
SPT2TS-124020 - FieldObjectController

{
  "structs": [
{
  "name": "ControlledInfraElement",
  "union": true,
  "attrs": [
    {"intId": 1, "name": "dpsGroup", "reference": "DriveProtectionSectionGroup"},
    {"intId": 2, "name": "tvpSection", "reference": "infra.TvpSection"}
  ]
},
    {
      "name": "FieldObjectController",
      "info": "Defines a Field Object Controller",
      "attrs": [
        {"intId": 1, "name": "id", "dataType": "string", "key": "global", "info": "Identity of the object; used for

referencing ; structure refers to EULYNX EU.SAS.77"},
{"intId": 2,"name":"controlledInfraElements","composition":"ControlledInfraElement", "multiplicity": "1..*"},
{"intId": 3, "name": "rastaServerId", "dataType": "string", "info": "defines EULYNX RaSTA Server identifier for the field object controller"},
{"intId": 4, "name": "rastaOCId", "dataType": "string", "info": "defines EULYNX RaSTA OC identifier for the field object controller"},
{"intId": 5, "name": "version", "dataType": "uint32", "info": "defines the EULYNX field object controller version"}
]
}
]
}

2.5 Route

This object is required for safety logic use case defined in IP FA1



SPT2TS-126716 - Route

```
{
   "structs": [
      {
         "name": "Route",
         "info" : "A route that starts and ends at specific pos of TrackEdges on which train can run",
         "attrs": [
            {"intId": 1, "name": "id", "dataType": "string", "key": "global" ,"info": "Identity of the object; used
for referencing"},
            {"intId": 2, "name": "type", "enumType": "RouteType"},
            {"intId": 3, "name": "availableARS", "dataType": "boolean",
"info":"ARS=AutomaticRouteSetting"},
            {"intId": 4, "name": "autoRoute", "dataType": "boolean", "info": "see Eulynx.dataprep"},
            {"intId": 5, "name": "isTowardsTunnel", "dataType": "boolean"},
            {"intId": 6, "name": "setupTime", "dataType": "uint32", "unit": "s", "exp": -3, "info": "use 0 if not
defined"},
            {"intId": 7, "name": "priority", "dataType": "uint32", "info": "use 0 if not defined"},
            {"intId": 8, "name": "maxSpeed", "dataType": "uint32", "unit": "km/h", "info": "use 0 if not
defined"},
            {"intId": 10, "name": "overlaps", "reference": "Overlap", "multiplicity": "*"},
            {"intId": 11, "name": "flankProtections", "reference": "FlankProtection", "multiplicity": "*"},
            {"intId": 12, "name": "flankProtectionVacantSections", "composition": "VacantSection",
"multiplicity": "*"},
            {"intId": 13, "name": "conflicts", "composition": "RouteConflict", "multiplicity": "*"}
           ]
      },
      {
        "name": "RouteBody",
        "info": "part the train uses directly. It is reused by several routes, which differ in restrictions etc. To
be extended",
        "attrs": [
          {"intId": 1, "name": "id", "dataType": "string", "key": "global"},
          {"intId": 2, "name": "path", "composition": "infra.LinkedPath"}
        ]
      },
      {
        "name": "Overlap",
        "info": "to be extended",
        "attrs": [
          {"intId": 1, "name": "id", "dataType": "string", "key": "global"},
          {"intId": 2, "name": "path", "composition": "infra.LinkedPath"}
        ]
      },
      {
        "name": "FlankProtection",
        "info": "to be extended",
        "attrs": [
          {"intId": 1, "name": "id", "dataType": "string", "key": "global"},
          {"intId": 2, "name": "simplePoint", "reference": "infra.SimplePoint"},
          {"intId": 3, "name": "isLeftPosition", "dataType": "boolean", "info": "if false - rightPosition"}
        ]
      },
      {
        "name": "RouteConflict"
      },
      {
        "name": "VacantSection",
        "attrs": [
          {"intId": 1, "name": "section", "reference": "infra.TvpSection"},
          {"intId": 2, "name": "proving", "enumType": "ProvingType"}
        ]
      }
   ],
  "enums": [
 {
   "name": "RouteType",
   "info": "Eulynx.Taxonomy, https://dataprep.eulynx.eu/2021-05/EARoot/EA2/EA4/EA3021.htm",
   "enumLiterals": [
      {"intId": 0, "name": "mainRoute"},
      {"intId": 1, "name": "onsightRoute"},
      {"intId": 2, "name": "shuntingRoute"}
   ]
 },
 {
   "name": "ProvingType",
   "info": "Eulynx.ProvingTypes, https://dataprep.eulynx.eu/2021-05/EARoot/EA2/EA3/EA2942.htm",
   "enumLiterals": [
      {"intId": 0, "name": "continuously"},
      {"intId": 1, "name": "oneOff"},
      {"intId": 2, "name": "staffAcknowledged"},
      {"intId": 3, "name": "none"}
    ]
  }
]
}
```

2.6 Simple point

This object is required for safety logic use case defined in IP FA1

SPT2TS-126714 - SimplePoint

```
{
   "structs": [
      {
         "name": "SimplePoint",
         "info" : "An object to extend infra.SimplePoint",
         "attrs": [
            {"intId": 1, "name": "id", "dataType": "string",  "sameKeyAs": "infra.SimplePoint", "info": "Identity
of the object; used for referencing"},
            {"intId": 2, "name": "routeRequiresPointRight", "reference": "Route", "multiplicity": "0..*",
"info":"routes require this SimplePoint to allow travelling through pointRight"},
            {"intId": 3, "name": "routeRequiresPointLeft", "reference": "Route", "multiplicity": "0..*",
"info":"routes require this SimplePoint to allow travelling through pointLeft"},
            {"intId": 4, "name": "transitLockingPointRight", "reference": "Transit", "multiplicity": "0..*",
"info":"Transit that is locking the SimplePoint to right"},
            {"intId": 5, "name": "transitLockingPointLeft", "reference": "Transit", "multiplicity": "0..*",
"info":"Transit that is locking the SimplePoint to left"},
             {"intId": 6, "name": "zoneLockingPointRight", "reference": "infra.TvpSection", "multiplicity":
"0..*", "info":"Zone (TvpSection of a single track) that is locking the SimplePoint to right"},
             {"intId": 7, "name": "zoneLockingPointLeft", "reference": "infra.TvpSection", "multiplicity": "0..*",
"info":"Zone that is locking the SimplePoint to left"}
           ]
      }
   ]
}
```

2.7 Transit

This object is required for safety logic use case defined in IP FA1

SPT2TS-126921 - Transit

```
{
   "structs":[
      {
         "name":"Transit",
         "info":"A Transit represents a directed movement across a single Zone. Each route activates one
or more transits.",
         "attrs":[
            {"intId": 1, "name": "id", "dataType": "string", "key": "global", "info": "Identity of the object; used
for referencing"},
            {"intId": 2, "name": "upstreamTransits", "dataType": "string"},
            {"intId": 3, "name": "transitZone", "reference": "infra.TvpSection", "multiplicity":"0..*"},
            {"intId": 4, "name": "routeRequestingTransit", "reference": "Route", "multiplicity":"0..*"}
         ]
      }
   ]
}
```

2.8 Container for Train Protection Area

SPT2TS-127377 - Assumption: Train Protection Area is identical with TopoArea, even if several interlockings are located inside of one Train Protection Area. Assignment of dpsGroups and allocationSections to interlockings will be done in a dedicated data structure for interlocking configuration. **[** Content to be approved **]**

SPT2TS-122302 - TrainProtectionArea

```
{
  "structs": [
    {
      "name": "TrainProtectionArea",
      "attrs": [
        {"intId": 1, "name": "id", "dataType": "string", "key": "global", "sameKeyAs": "infra.TopoArea"},
        {"intId": 2, "name": "dpsGroups", "composition": "DriveProtectionSectionGroup", "multiplicity": "*",
"sortedByKey": true},
        {"intId": 3, "name": "allocationSections", "composition": "AllocationSection", "multiplicity": "*",
"sortedByKey": true},
        {"intId": 4, "name": "fieldObjectControllers", "composition": "FieldObjectController", "multiplicity": "*",
"sortedByKey": true},
        {"intId": 5, "name": "routes", "composition": "Route", "multiplicity": "*", "sortedByKey": true},
        {"intId": 6, "name": "simplePoints", "composition": "SimplePoint", "multiplicity": "*", "sortedByKey":
true},
        {"intId": 7, "name": "transits", "composition": "Transit", "multiplicity": "*", "sortedByKey": true}
      ]
  }
  ]
}
```

2.9 Container for Train Protection package

SPT2TS-122309 - Formal Specification "Train Protection" as a container for Train Protection Areas:

The user instructions and use-cases can be found at SPT2TS-122470
[Content to be approved **]**

SPT2TS-125487 - TrainProtection

```
{
  "structs": [
    {
      "name": "TrainProtection",
      "attrs": [
        {"intId": 1, "name": "tpArea", "composition": "TrainProtectionArea", "multiplicity": "*", "sortedByKey":
true}
      ]
    }
  ]
}
```