Guidelines for STM

1. General Comments

The base RTZ format was contemplated as flexible and easily expandable. That is why the minimum requirements set for it are not burdensome: it is only necessary to provide a route with the route name (**routeName** in the **routeInfo**) and a set of (at least two) waypoints (in the **waypoints** container element). Each waypoint should necessarily be provided with at least geographic coordinates (**position**). The rest is optional in the RTZ.

For STM validation we have extra requirements and following fields became mandatory:

- **vesselVoyage** in the **routeInfo** element
- **routeStatusEnum** in the **routeInfoEx** extension of the **routeInfo** element.

It means that **routeInfoEx** extension element became mandatory for STM usage as well.

The vessel voyage ID is the Unique Identity of a Voyage and should be used exactly specified in the STM MRN format in the RTZ **vesselVoyage** field, for example for the org name 'acme':
vesselVoyage="urn:mrn:stm:voyage:id:acme:b6d7b492-ab3c-42f2-8afd-116c3d872f0c"

Service providers shall not change UVID of each route received from a ship.
IF it is brand new route that Service Provider is going to send to a ship, then UVID field must be filled in, because the UVID field CANNOT be empty.

2. Route Geometry

Each route leg is formed as the shortest of all the possible segments within the framework of the set geometry (Great Circle or Rhumb Line) on the terrestrial ellipsoid. The following multiplicities may arise in the process:

1. The GC or RL geometry type, and the ends of the segment are on the opposite poles. The shortest segment is then any meridian. To avoid this multiplicity, the zero meridian only will be selected.
2. The GC geometry type, and the segment ends are diametrically opposite, but are not poles. There may be two solutions in this case: a segment passing through the north pole and a segment passing through the south pole. To avoid the multiplicity, the one passing through the north pole is selected.
3. The RL geometry type, and the segment ends are on the diametrically opposite meridians, none of them the zero meridian. There may be two solutions in this case: the segment crossing the zero meridian and the segment crossing the 180th meridian. The first is always chosen out of the two.
4. The RL geometry type, and one of the segment ends is on the zero meridian, whereas the second on is on the 180th. There may be two solutions: the segment fully within the east hemisphere, and the segment fully within the west hemisphere. Of the two possible options, the first is always chosen.

If in the course of creating a route it becomes necessary to construct a leg using one of the aforementioned options (and it is of no importance if you are satisfied or not with the multiplicity

resolution method selected above), an intermediate point should be set on the leg, which will permit an absolutely identical route line to be drawn, but without any uncertainties of the kind.

Whenever the minimum accuracy is stipulated for any attribute, this means that this parameter can be recorded and read with a greater accuracy. But it should be kept in mind that there is no warranty whatsoever that an extraneous application will be able to read, process and record this attribute in the RTZ with this kind of accuracy.

3. Comments on 'routeInfo' Element

It has been mentioned before that the **routeName** field should necessarily be available and cannot be empty of consist of space symbols only. In the **routeAuthor** field, it is worthwhile to specify not only the name with initials, but also the position if possible. The **vesselVoyage** field is mandatory, according to the STM requirements. The XML **routeChangesHistory** element may be absent (being optional), or empty.

If any of the **routeInfo** fields is absent, it means that there is no data on this field.

The **vesselGM** and **vesselMaxWave** field values should be accurate to at least one decimal place.

The **vesselMaxWind**, **vesselSpeedMax**, **vesselServiceMin**, **vesselServiceMax** field values should be in knots and accurate to at least one decimal place.

4. Comments on 'routeInfoEx' extension element

routeStatusEnum attribute is integer and should correspond to the STM requirements for this field. Currently it should be in range from 1 to 8.

The **routeVersion** version serves for differentiating not only versions of the same route, but also different routes, which is easily enabled by the field type (GUID is recommended).

The XML **routeChanges** element is an XML container of **historyItem** elements and may be absent (being optional), or empty. But each **historyItem** available in it, is required to be filled, since all of its fields (**dateTime**, **author**) are mandatory, excepting **reason** field. The **author** field cannot be empty or consist exclusively of space symbols. Since there is no dedicated route creation date or time field in the **routeInfo**, the route creation can be interpreted as its revision.

The filling in of the **routeChanges** element is done in the following manner, in case of the human interaction:

1) The user creates or opens a route for editing.

2) He/she performs a certain set of operations aimed at its revision.

3) When the user considers the route editing to be completed, he/she should complete the "Change reason" and "Change author" fields in the route editor and press the "Log to history" button.

4) Should the user fail to fill in the fields after the editing, and try to exclude the route from the list of edited one, or close the route editor, the system will not permit him/her to do this until these fields are filled in.

If the user makes some changes in the route after filling in the “Change reason” and “Change author” fields and pressing the “Log to history” button, the “Change reason” and “Change author” fields are reset. After completing a new series of revisions, the user is required to fill in these fields and press the “Log to history” button again.

For M2M interaction those fields could be filled automatically by the corresponding systems and solutions.

depPortCallId and **arrPortCallId** fields should correspond to the **Port Call Identifier (UPCID)** format (pattern) and do not exceed 120 symbols:

<xs:pattern

value="**urn:mrn:stm:portcdm:port_call:[A-Z]{5}:[A-Za-z0-9()+,\-.:=@;$_!*'%/?#]+**"/>

Example of the **Port Call Identifier (UPCID)**:

urn:mrn:stm:portcdm:port_call:SEGOT:C44928d8-0e93-46Be-baf9-b824e0fdbe90

Description for the **startSeaPassage** and **endSeaPassage**. These fields should correspond to the **Port Call Identifier (UPCID)** format according to **arrivalLocation** or **departureLocation** format. Where, **LogicalLocation** should be specified according to the PCM format, and additionally it is recommended to specify the WP ID for this **LogicalLocation** after the colon symbol. These fields in RTZ will help for the following M2M ‘understanding’:

- which WP is specified for the Arrival / Departure PCM reports from the ship;
- what is the PCM Logical Location for these points on the route (BERTH, PILOT_BOARDING_...)

Examples:

Example 1: **startSeaPassage="BERTH:WP2"/>**

Example 2: **endSeaPassage="LOC:WP311"/>**

Example 3: **endSeaPassage="PILOT_BOARDING_ AREA:WP56"/>**

Logical Location LOC shall be used for the any specified Waypoint by default.

5. Comments on ‘defaultWaypoint’ Element

It is important to keep in mind that **defaultWaypoint** cannot contain fields: **name** (since the uniqueness is also implied here), revision (since the revision number for a newly created point always begins with one and is individual for each of the waypoints)**, position** (since the concept of a position is by default meaningless for the routes). As far as the **id** field is concerned, its availability may be justified and considered as an identifier of the next waypoint which will be created. In the **defaultWaypoint** element, it is preferable to set the parameters that are used by default in creating a new waypoint. Such approach may be useful when, e.g., it is necessary to send a route from the ship to a third party for some minor adjustments. Then such party can only set geographic coordinates of new waypoints without specifying any other attributes requiring additional knowledge.

And after such addition, the shipboard personnel, with a high degree of probability, will not have to set the necessary additional parameters for each so added waypoint, since we may be quite satisfied with what is available in the **defaultWaypoint**. If, however, the geometry of the obtained route proves to be incorrect, this is of no concern for the other party, since the obtained route should anyway be checked on the ship and corrected if required, as the ship personnel is as a rule better aware of their ship parameters and its current condition.

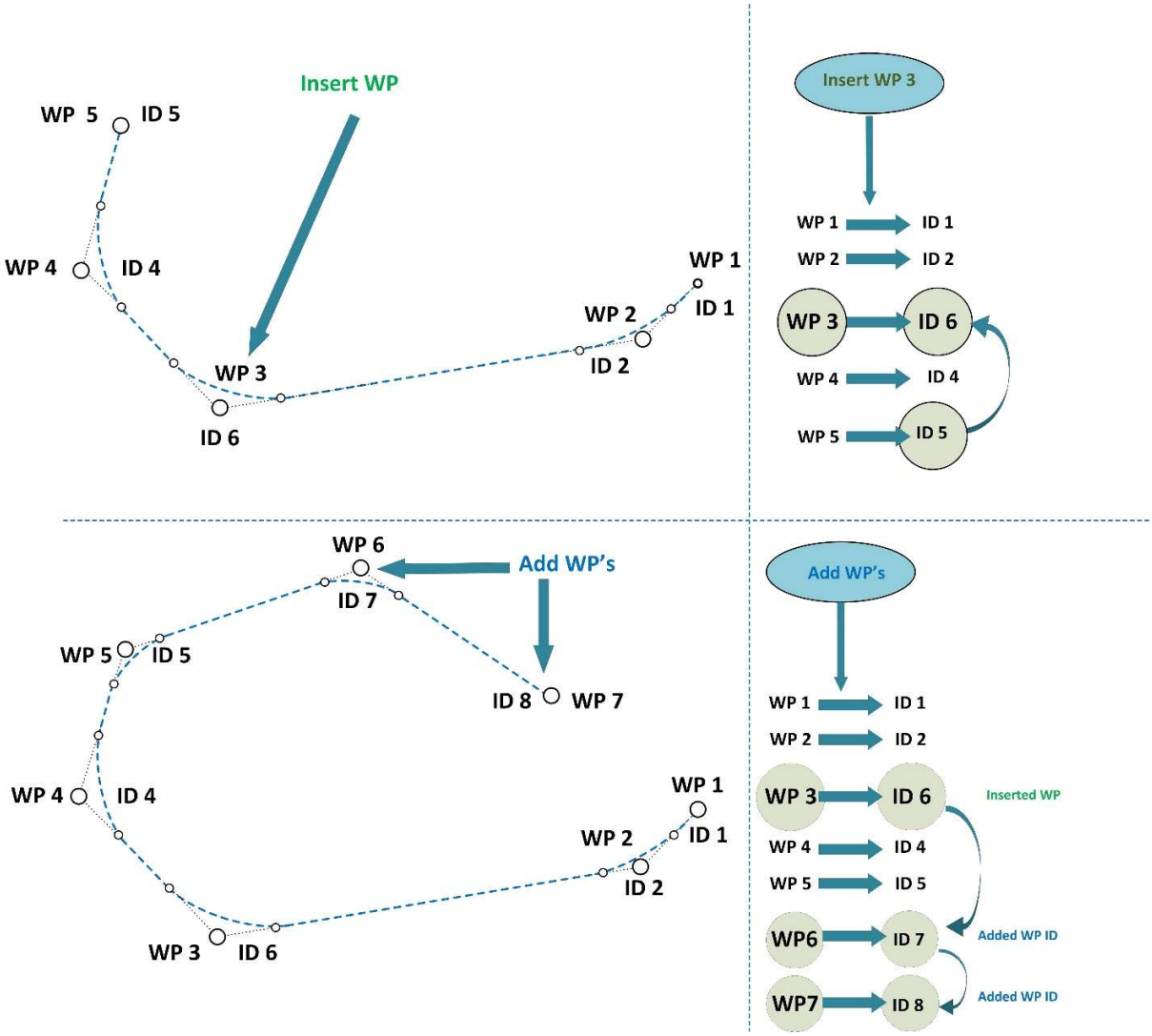
When it comes to service provider or shore center, e.g., when they recommend some changes to ships' RTZ, but they should not change the content of **defaultWaypoint** element or delete it [if it is available in ship's RTZ].

6. Comments on 'Waypoint' Element

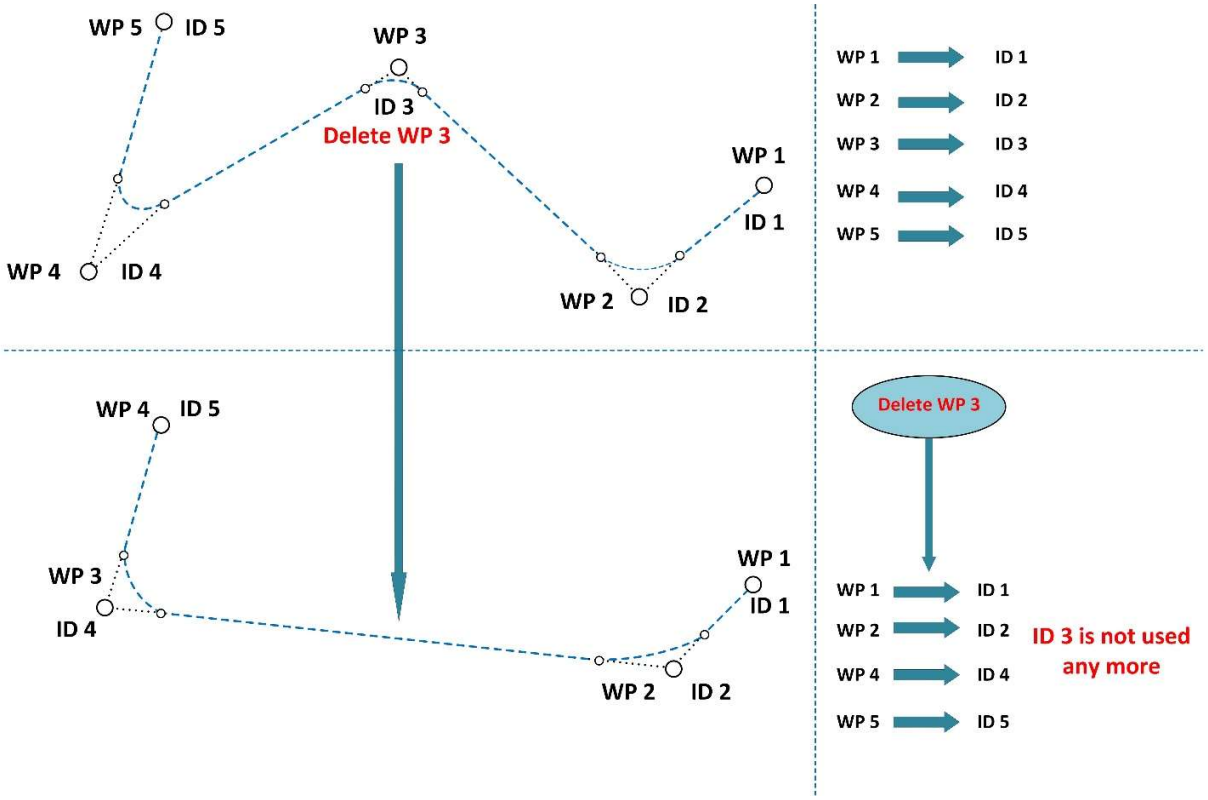
For the sake of uniqueness of the sequence number, the first point in the route should be numbered 1 (rather than 0). The **id** field value for a waypoint should not be necessarily the same as its index in a route. The waypoints succession order is determined exclusively by their position in the **waypoints** container, so that the adjacent point identifiers may not correlate at all. The essential thing for an id is its uniqueness within the route, i.e., there should be no waypoints with the same id within one route. The waypoint identifier mechanism has been specifically devised to ensure an ability to easily understand which points remained unchanged, which were subjected to changes (by using the revision field), which were deleted and which of them appeared after arbitrary changes in the route geometry. In addition, the waypoint identifiers permit references to the geometry (rather that the succession order) from other RTZ elements, e.g., from the **scheduleElement**. The easiest way is to number waypoints in the order of their appearance in the process of creating and revising a route.

Please refer to the pictures below clarifying the text above:

Inserting and modification of a waypoint:



Deleting of a waypoint:



The **revision** field is required to be updated each time some changes are applied to the relevant **waypoint** element: changes of a position or some other attribute value (including the extension for a **waypoint**, the **leg** element and its extensions), addition or deleting of some attribute (with the same type of reservation). Whenever a waypoint position is changed by using the "drag and drop" technique, it is worthwhile not to update the **revision** field until the "drop" part is performed.

To avoid mixing up the deleted points with recently added ones, it is worthwhile to store separately the next identifier value, so that each newly created waypoint is assigned with this identifier which should after this be incremented for the subsequent use (maybe it should be stored in the **id** field for the **defaultWaypoint**).

The **leg** element is not mandatory for a waypoint. This element can nonetheless be meaningful for the first waypoint, although no leg can correspond to it. The matter is that when a point is added to the beginning of the route, it is sometimes convenient to have a ready-for-use set of parameters for a newly formed leg. This is also applicable to the deleting of the first waypoint: there is no need to immediately thereupon clear the **leg** for the point which was previously the second one and became the first. Another use of the **leg** element for the first **waypoint** is the merging of two routes, i.e., leg parameters for the first point that turned up in the middle of the joint route, will be referred to the leg connecting the two routes.

The **radius** field should contain a non-negative number smaller or equal to 5 NM, since a larger value of this parameter is not practical, whereas the upper limit provides a higher "junk" detection probability.

This field should be accurate to at least two decimal places. This field presentation precision may vary from 0.1 to 0.001 NM.

If the **radius** field is not available in both **waypoint** element and in the **defaultWaypoint**, for the waypoint in question this parameter is understood as equal to zero, since it is essential that this route should have a strictly defined geometry, and the turn radius, in its turn, has a considerable effect on this geometry, and, therefore, cannot be indefinite.

The **position** element should necessarily contain the latitude and longitude values expressed in degrees and tenths of a degree. Coordinates should be presented with an accuracy to at least five decimal places. These values presentation precisions may vary from 0.00001^0 (with a precision in the order of 1 metre) to 0.0000001^0 (with a precision in the order of 1 cm) depending on the application area. It has been mentioned that each route leg is formed as the shortest of all the possible segments within the framework of the set geometry (Great Circle or Rhumb Line) on the terrestrial ellipsoid. It follows that there is no need to permit entry of arbitrary longitude value, and it would seem reasonable to have the $-180^0 \leq lon < 180^0$ restriction. With this kind of restriction each point on the terrestrial surface (except for the poles, which can be corrected by setting zero longitudes only for them) can be uniquely correlated with a lat/ lon pair. Moreover, introduction of this restriction permits, just like in the case of the turn radius, quite easy detection of the “junk” longitude values.

Values in the **starboardXTD** and **portsideXTD** fields are required to be presented with an accuracy to at least two decimal places. The upper limit of 10 NM for these fields serves the same goal as the limitation for a **radius**. If any of these fields are not set for a certain **leg**, and they are not available in the **leg** for the **defaultWaypoint**, this means that such field is considered to be equal to zero, since the **starboardXTD** and **portsideXTD** essentially define the route geometry, and, as mentioned above, the route geometry cannot be undefined.

Values of the **safetyContour**, **safetyDepth**, **draughtForward**, **draughtAft**, **staticUKC**, **dynamicUKC** and **masthead** fields should be accurate to at least one decimal place.

Values of the **speedMin** and **speedMax** fields should be accurate to at least one decimal place.

It follows from the table for the **leg** element that the **legReport** field should contain part of the annotated route planning. Information presentation format is arbitrary.

According to the table for the **leg** element, the **legInfo** field should contain phones and e-mail addresses of some services, which may be useful during the approach to a port or VTS. Information presentation format is arbitrary.

The **legNote1** field should contain some note concerning the arrival and departure time. This field filling format is also arbitrary.

The **legNote2** field is intended for local notes, i.e., it is not only the format, but also the contents of this field, which is arbitrary. It is advisable not to fill this field with what can be presented in some other **leg** element field.

7. Comments on 'Schedules' Element

The route weather and currents optimizing should occur concurrently with the schedule calculations, since this optimizing is essentially time-dependent.

A unique **id** should be set for each schedule element. At the same time, the **name** attribute is not mandatory, and there may be several schedules with the same name within the framework of one route (although this could hardly be recommended). Therefore, the **name** attribute is intended mainly for making it easier for a person handling several schedules for one and the same route, to find the necessary schedule and distinguish the schedules one from another.

Like in the case of an **id** for a **waypoint**, identifier for a **schedule** should be non-negative integer.

If the route geometry changes, all the calculated schedules should be reset, as the calculation results basically depend on the geometry. This means that every change in geometry leads to deleting of **calculated** elements of all schedules in the route.

If there are several schedules in the RTZ file, the **one which comes first** in the **schedules** element container is considered to be active. So, if the active schedule has been changed, the new active schedule should be moved to the beginning of the **schedules** section.

The **calculated** element should contain both parameters consistent with the manually set ones (presented in the **manual**), and the calculated values. Consistency means equal value intervals for the **eta**, **etd** and **speed** (with some reservations for this parameter which will be referred to below) and equality of the rest of parameters except for the **stay** parameter, as it is essentially dependent on the specific **eta**, **etd** values rather than on their intervals. The matter is that in the **scheduleElement** it is possible to set intervals for changing the **eta**, **etd** and **speed** in the course of optimizing by filling in the **etdWindowBefore**, **etdWindowAfter**, **etaWindowBefore**, **etaWindowAfter**, **speedWindow** parameters. This is why, if after the optimizing, the manually set **eta**, **etd** or **speed** values have changed, the appropriate changes should be made to the **etdWindowBefore**, **etdWindowAfter**, **etaWindowBefore**, **etaWindowAfter, speedWindow** parameters.

If some of crucial manual parameters (speed, stay, ETA or ETD) in a schedule is changed, all calculated data of the schedule shall be deleted.

STM service providers ***shall*** not change any manual schedule parameters in routes having been received from ships. If a STM service provider calculates for such routes some schedule parameters, these parameters shall be put in **calculated** elements only.

Excepting the case when Service Provider or Shore Centre wants to propose a new arrival time, then those service provides **MAY CHANGE** the manual section of schedule.

The schedule is considered to be complete if:

1) The **manual** element is not in contradiction to the **calculated** element, and the data for **manual** is sufficient for recalculating the schedule.

2) Its **calculated** element contains **etd** parameter values for all the waypoint except the last one, and the **eta** and **speed** parameters exist for all waypoints, except the first one.

It is only a complete schedule that can be selected for voyage plan monitoring.

It is important to bear in mind that all the fields describing a certain time interval (**etdWindowBefore**, **etdWindowAfter**, **etaWindowBefore**, **etaWindowAfter**, **stay**) do not comply with the "HH.MM" format, e.g., "15.05". This interval should have the form of PT15H5M instead. If the interval length is more than 24 hours, and comes to, e.g., 37 hours 27 minutes, this should be expressed as follows: "P1DT13H27M" in full compliance with requirements for the xds duration type.

For all the fields in the **scheduleElement,** there is a capability to set them manually, but in the vast majority of cases, parameters like **windSpeed**, **windDirection**, **currentSpeed**, **currentDirection**, **windLoss**, **waveLoss**, **totalLoss**, **rpm**, **pitch**, **fuel**, **relFuelSave**, **absFuelSave** will be obtained as the result of the schedule calculations or route optimizing.

Fields **speed**, **speedWindow**, **windSpeed**, **currentSpeed**, **windLoss**, **waveLoss**, **totalLoss** are required to be accurate to at least one decimal place.

It may occur that the wind pushes the ship forward, and in this case the **windLoss** and **totalLoss** parameters may be negative.

The **relFuelSave** and **absFuelSave** parameters may also prove to be negative, since the optimizing goal may be other that the fuel saving, and consist, e.g., in minimizing the passage time on the route or minimizing some risks, so that as the result of such optimizing more fuel would be required than for a non-optimized route. presenting of the **relFuelSave** parameter should be accurate to at least one decimal place, whereas for the **absFuelSave, presentation** the accuracy to the integers would be sufficient.

The **note** field permits an arbitrary format and contents.

8. Comment on 'Extension' Element

It is desirable to place the extension in the element that this extension is best associated with, rather than at random in the RTZ file. E.g., additional fields for the STM that we would like to see in the **routeInfo** element were included in the extension for this element, rather than for the **route** element or some other. Another example is provided by reference points for the waypoints: it is natural to see this extension in the **waypoint** element rather than anywhere else.

9. Comment on 'Extension' / 'routeStatusEnum' element

The **routeStatusEnum** field should be used to inform about the status of the route in order to support the various scenarios, e.g. route optimization and crosschecking. Note that there is no requirement to use these statuses during voyage planning, but as a rule, the route must have a status. The intention with status 8, inactive, is to be able to inform subscribers to a voyage plan that the voyage has been completed or cancelled, e.g. if a voyage is cancelled possible previous subscribers need to be made aware about this whether or not they are given access rights to the voyage that "replaces" cancelled one.

Following table shall be used in advisory mode for the different STM stakeholders in relation to the **routeStatusEnum** field:

Route Status	Name	Description	Responsible System or Service
1	Original	Template or basic voyage planned and received from shore.	• **ECDIS:** Route template, which is created in the ECDIS route database on the ship (*Voyage Plan is not loaded into monitoring*) • **Service Provider**: Route template, prepared as a Pilotage Route or Winter Ice Navigation route by any appropriate service provider on the shore.
2	Planned for voyage	Route and schedule prepared by crew.	• **ECDIS:** Route with one or several schedules, calculated and stored in the ECDIS route database on the ship (*Voyage Plan is not loaded into monitoring*)
3	Optimized	Route and schedule optimized by 3rd party service provider.	• **Service provider:** Optimized Route with one or several schedules, calculated by any 3rd party Service provider (Weather optimization, Auto-Routing, etc.) and delivered to the Ship or Fleet operator with the **routeStatusEnum="3"**.
4	Cross Checked	Route verified by 3rd party	• **Service provider:** Cross-Checked route with one or several schedules, verified by any 3rd party Service provider (Fleet operator, Weather optimization, Auto-Routing, etc) and delivered to the Ship with the **routeStatusEnum="4"**.
5	Safety Checked	Safety check by ECDIS/crew	• **ECDIS:** Route with one or several schedules has been checked by the ECDIS Check-Route functionality, if applicable (*Voyage Plan is not loaded into monitoring*)
6	Approved	Approved by master	• **ECDIS:** Route with one or several schedules has been Approved / Protected in the on board ECDIS system, if applicable (*Voyage Plan is not loaded into monitoring*)

Route Status	Name	Description	Responsible System or Service
7	Used for monitoring	Loaded in ECDIS for monitoring	• **ECDIS:** Route **with or w/o schedule** is loaded into monitoring in the ECDIS system on board (Voyage Plan is loaded into monitoring and VIS instance is updated from the ship with RTZ **routeStatusEnum="7"**). It is recommended to use only one valid and active 'schedule' in the RTZ ship route with such Unique Voyage ID and **routeStatusEnum="7"**) during the voyage
8	Inactive	Voyage completed or cancelled	• **ECDIS:** Route / Voyage has been completed or stopped/cancelled in the ECDIS system on board (VIS instance is updated from the ship with RTZ with the previous UVID **routeStatusEnum="8"**, if applicable).