TSM7-5.4

Paper for Consideration by the S-100 TSM

Addressing Capability Gaps in the S-100 Portrayal Model

Submitted by:	NIWC Atlantic
Executive Summary:	Identifies capability gaps and provides recommendations for addressing them.
Related Documents:	S-100 Part 9 and 9a
Related Projects:	Development of S-100 and S-1XX product specifications

1 Introduction / Background

S-100 provides a portrayal framework defining capabilities for use by products. The framework ensures product presentation, use, and implementation in a standard, consistent manner.

In order to smooth the transition from legacy standards the framework must provide portrayal functionality to meet ECDIS requirements. Several capability gaps have been identified during testbed development; they are presented in this paper along with recommendations for eliminating those gaps.

2 Identifying the Gaps

Most of the portrayal gaps involve some form of “dynamic portrayal”. We define dynamic portrayal as the requirement for repeated re-rendering based on internal or external inputs. Note that re-rendering does not necessarily imply re-generation of the portrayal drawing instructions.

The portrayal gaps can be sub-divided into those involving the implementation of ECDIS requirements, and those that are not ECDIS related.

The following ECDIS portrayal gaps have been identified:

- Alerts and Indications (when implemented as part of portrayal)
  - Requires re-rendering of alert highlights based on ownship movement
- Temporary Overlays (Hover)
  - Requires re-rendering of temporary overlays based on cursor movement
- Dependent Symbol Visibility
  - Requires filtering of drawing instructions based on visibility of other drawing instructions
- Date Dependent Symbols
  - Requires filtering of symbols based a date or date range

The following non-ECDIS dynamic portrayal gaps exist:

- Animation playback
  - Requires re-rendering based on an automated or manual frame counter
- Time Series portrayal
  - Requires re-rendering based on an automated or manual sequencing of time instances

2.1 Alerts and Indications

There is currently no alerts capability provided in S-100. ECDIS requires the ability to generate alerts based on the evaluation of dataset features.

TSM6 recommended implementation of Alerts and Indications functionality as part of the portrayal. A separate paper provides the full details of a proposed model for Alerts and Indications; a summary of the proposed implementation is included in this paper to highlight similarities with the approach used to address other portrayal gaps.

2.2 Temporary Overlays / Hover

S-52 PresLib 10.8.5 specifies optional OEM implementation of hover functionality. This functionality is applicable to the following:

- SY(INFORM01)
- SY(CHDATD01)
- LIGHTS
- LNDMRK
- Buoys
- Beacons

In order to implement this optional functionality in S-101, OEM's must code to specific versions of feature and portrayal catalogues. As these catalogs change due to introduction of new features, changes to existing features, or portrayal modifications, the OEM may need to respond with software changes.

2.3 Dependent Symbol Visibility

The visibility of certain symbols is dependent on the visibility of a "base" symbol. Examples include:

- Labels
  - Text drawn with symbols
  - Contour labels
- Highlights
  - SY(INFORM01)
  - SY(CHDATD01)

Several factors affect a "base" symbols visibility:

- Viewing Group / Viewing Group Layers / Display Mode settings
- Scale (ScaleMinimum / ScaleMaximum)
- Line Suppression
- Date Dependent

Currently there is no portrayal mechanism to express a dependency between symbols.

2.4 Date Dependent Symbols

S-52 PresLib 10.4.1 specifies requirements for date dependent symbols. It requires the ability to specify a viewing date (or date range) against which feature attributes are compared to determine symbol visibility. It also requires use of highlight symbol SY(CHDATD01) for identifying date dependent features. OEM implementation of hover functionality on the highlight symbol is optional.

There are several gaps preventing the implementation of date dependent symbols:

- How to implement using context parameter type *Date* is unclear
  - How should a date range be specified
  - How to select between a date and date range
  - XSLT does not have date comparison operators
  - Lua date comparison requires embedding a date library into the portrayal catalog
- SY(CHDATD01) is dependent on the visibility of the base feature
  - an instance of 2.3 Dependent Symbol Visibility
- The S-52 implementation of highlight date dependent is inconsistent with SY(INFORM01)

    - SY(CHDATD01) uses an OEM provided selector to control visibility
    - SY(INFORM01) uses viewing group 31030 to control visibility

2.5 Dynamic Portrayal

Use cases for dynamic portrayal include the display of weather, tides, or historical imagery. Point sets or coverages are most often used to encode datasets intended for dynamic portrayal.

S-100 does not currently support dynamic portrayal. At least two types of dynamic portrayal could be provided:

- Animation – the sequenced display of frames based on an automated or manual frame counter.
- Time series – the automated or manual selection or sequencing of frames based on an absolute time or timespan.

Prior to implementing dynamic portrayal, the following gaps must be addressed:

- Clarification on GML encoding of time series data
    - see OGC 15-043r3 Time series Profile
- Collect product requirements
    - Which products need dynamic display?
    - How to synchronize the portrayal time?
        - For multiple datasets from a single product
        - Across multiple products
        - With date dependent features
    - Are indications / overlays needed to show the portrayal date?
- Recommendations for efficient implementation of the XSLT input schema for use with dynamic portrayal

It may be possible to leverage the date dependent functionality proposed in this paper to support portrayal of time series. Due to the unknown requirements, further analysis and recommendations for dynamic portrayal are not provided in this paper.

3 Analysis

As a first step, the visualization process was analyzed to determine how it could be modified to address the capability gaps. The visualization process is summarized as:

1. Use portrayal rules and context parameter values to transform dataset features into drawing instructions
2. Sort instructions by display plane
3. Sort the instructions within each display plane by drawing priority
4. Style using selected palette and CSS rules
5. Filter by:
    a. Viewing groups
    b. ScaleMinimum / ScaleMaximum
    c. Line suppression
6. Transform using display scale, projection, and orientation
7. Clip
8. Determine location of centered area symbols
9. Render

At a high level, the visualization process consists of two parts:

- Portrayal (steps 1-3)
    - Generation of drawing instructions and one-time processes applied to the drawing instructions
- Rendering (steps 4-9)

- Dynamic processes applied to the drawing instructions

The capability gaps can be addressed by modifying the portrayal, the rendering, or both.

4 Recommendations

We recommend extending the drawing instructions to address the capability gaps. This involves modifications to the rendering process, which has the advantage of minimizing latency (the portrayal can be used as-is, rather than regenerated), enabling OEMs to minimize the number of layers which need to be redrawn.

The recommended changes to the visualization process are highlighted below:

- Portrayal
    1. Use portrayal rules and context parameter values to transform dataset features into drawing instructions
    2. Sort by display plane
    3. Sort by drawing priority
- Rendering
    4. Style using selected palette and CSS rules
    5. Filter by:
        a. Viewing groups
        b. ScaleMinimum / ScaleMaximum
        c. Line suppression
        d. Date dependent
        e. Hover
        f. Symbol dependency
    6. Transform using display scale, projection, and orientation
    7. Clip
    8. Determine location of centered area symbols
    9. Render

Implementing these changes in Part 9 and Part 9a addresses the capability gaps in the S-100 portrayal model related to ECDIS requirements.

4.1 Changes to Part 9 Portrayal

Part 9 redlines are provided separately and are summarized here.

Role Name	Name	Description	Mult.	Type
Class	DrawingInstruction	Abstract base class for all drawing instructions	-	-
Attribute	id	An identifier for the drawing instruction	0..1	string
Attribute	parentId	Visibility is conditional on parent drawing instruction(s) visibility	0..1	string
Attribute	hover	The drawing instruction will be shown only on hover-over OEM support for this feature is optional	0..1	boolean
…	…	…	…	…
Association	alertReference	A reference to an alert catalog entry	0..1	AlertReference *Described in paper TSM7-5.2*
Association	timeValid	The drawing instruction is valid during the specified time interval(s)	0..*	TimeInterval

Role Name	Name	Description	Mult.	Type
Class	TimeInterval	A time or time period Single value intervals are encoded with *upper* = *lower* and *closure* = *closedInterval* (or omitted)	-	-
Attribute	closure	Specifies an *S100_IntervalType* Default is *closedInterval*	0..1	string
Association	lower	Start of the interval	0..1	Instant
Association	upper	End of the interval	0..1	Instant

Role Name	Name	Description	Mult.	Type
Class	Instant	A point in time. Multiple points may be described via truncated dates used to represent recurring instants.	-	-
Attribute	date	An *S100_TruncatedDate* (see table 1-2)	0..1	string
Attribute	time	A *Time* (see table 1-2)	0..1	string
Attribute	dateTime	A *DateTime* (see table 1-2)	0..1	string

4.1.1 Sample drawing instruction

```
<textInstruction parentId=“Foo” hover=“true”>
    <timeValid closure=“leSemiInterval”>
        <upper date=“20190101”/>
    </timeValid>
    …
    <text>bar</text>
    …
</textInstruction>
```

4.2 Changes to Part 9a Portrayal

Part 9a redlines are provided separately and are summarized here.

- Add new 9a-11.2.2.1 visibility commands
    - Id*[:id]*
        - Sets an identifier for subsequent drawing commands
        - If *id* is not present, resets to the default state of no identifier
    - Parent*[:id]*
        - Visibility of subsequent drawing commands is dependent on the visibility of command(s) with the specified *id*
        - If *id* is not present, resets to the default state of no parent dependency
    - Hover:*hover*
        - Specifies whether subsequent drawing commands are dependent on hover-over
        - OEM implementation of support for this feature is optional
- Add new 9a-11.2.2 state command type: Time
    - Associates time intervals with drawing commands
    - A time interval is described by a lower and upper value, each containing a date and time
    - Time commands can be intermixed
        - Date:*20190101*;Time:*183059Z*;DateTime:,*20190101T200000Z*
- Add new 9a-11.2.2.8 Time commands
    - Date:*[lower][,upper]*
        - Sets the lower and / or upper date
        - *lower* and *upper* are *S100_TruncatedDate*(s) (see S-100 Table 1-2)

- Time:*[lower][,upper]*
  - Sets the lower and / or upper time
  - *lower* and *upper* are *Time*(s) (see S-100 Table 1-2)
- DateTime:*[lower][,upper]*
  - Sets the lower and / or upper date and time
  - *lower* and *upper* are *DateTime*(s) (see S-100 Table 1-2)
- TimeValid*[:closure]*
  - Creates a time interval which applies to subsequent drawing commands
  - Interval described by preceding *Date* / *Time* / *DateTime* commands
  - Intervals accumulate until cleared with a *ClearTime* command
  - *closure* specifies a *S100_IntervalType* (see S-100 Table 1-3), default is *closedInterval*
  - Subsequent drawing commands and their associated information (e.g. alerts) are valid if any time interval coincides with the viewing date of the portrayal (or other appropriate selector)
- ClearTime
  - Clears all accumulated time intervals
  - Subsequent drawing commands are not time dependent

4.2.1 Sample drawing instruction

```
Parent:Foo
Hover:true
Date:,20190101
TimeValid:leSemiInterval
…
TextInstruction:bar
```

5 Action Required

The group is invited to:

a. Note the issues presented

b. Determine next actions for addressing dynamic portrayal

c. Provide feedback on the recommended changes