TSMAD28/DIPWG6 11.3A

Paper for Consideration by TSMAD28/DIPWG6

[Use of SVG for S-100 Portrayal]

Submitted by:	CARIS
Executive Summary:	Discussion on use of SVG for S-100 Part 9 symbol definition
Related Documents:	S-100 Part 9, SVG specifications http://www.w3.org/TR/SVGTiny12/index.html S-52 Annex A of Appendix 2 Preslib Preslib_3.3Addendum.doc
Related Projects:	S-100 Part 9 and S-101 Portrayal Catalogue

Introduction / Background

In S-100 Part 9 SVG is identified as being used in the definition and exchange of symbols, linestyles and area patterns. This document describes concepts and suggestions regarding details related to the use of SVG and issues that may need further consideration.

It is considered that SVG be mainly a vehicle for transferring symbol information to the end display system and that the final implementation may be converting the SVG into an equivalent internal representation. It would be advantageous for implementers if a well-defined subset of SVG capabilities be utilized.

Analysis/Discussion

Units

In S-52 Preslib the units for symbols is defined as factors of 0.01mm on the display.
Line thickness of pen width are given in units of 0.32mm which came from a typical pixel size on a high res graphics screen at the time.
These units can be converted to mm upon converting symbols from S-52 into SVG.

SVG units for width, height and shape coordinates default to pixels but they can be set to other units such as Millimeters (mm) by placing an “mm” after each value.
The viewBox attribute of the svg element provides a way to make all the units behave as mm by setting the viewbox origin to 0 0 the max x to the width in mm and the maxy to the height in mm.
E,g, width="5.14mm" height="5.06mm" viewBox="0 0 5.14 5.06"

Origin

S-52 symbol definitions and SVG use the upper left corner as the origin with Y going down.

Pivot Point

In S-52 symbol definitions an X and Y value were assigned to identify how the symbol would be placed and around which position to rotate it. SVG does have functions to transform or rotate elements within an SVG graphic but what is needed is a way for the display system to know where to place the SVG symbol onto the screen with respect to the location on the screen where a geographic location is mapped to. For example a Lat, Long position would be mapped to a specific pixel on the screen and then the symbol is placed there. Another term for this is the anchor point of the symbol. The definition of what position within the symbol is mapped to the destination coordinate where the symbol is being placed.

It is proposed that the centre of the SVG element be considered to be the “Pivot Point” or reference point for the symbol. The Pivot Point X is the SVG width/2 and the Pivot Point Y is the SVG height/2.

The S-52 symbols can be translated into SVG by shifting the coordinates such that the pivot point is in the centre and if necessary the cover of the symbol (width, height) be adjusted accordingly.

Colours

The default colour specification used by SVG is sRGB. This is not to be confused with RGB as it is expected to be used by a calibrated monitor.

A requirement for portrayal is to be able to swap in/out different colour tables or palettes based on display conditions (Day, Dusk, Night) etc. We don't want to have to create new symbols for every set of colours so colour tokens are used to specify a colour and these are converted at draw time to apply the appropriate colour from the active palette.

S-52 has colour definitions using colour tokens and CIE managed colour values.

The SVG spec does describe a way to define colours using an icc-color function.
First a statement is needed to define the colour profile such as:

```
<color-profile name="S100Colors" xlink:href="http://www.iho.int/s100colorProfile"/>
```

Then colour can be assigned to a graphic element such as

```
<circle stroke="#8D642Eicc-color(S100Colors, LANDF)" fill="none" cx="2.5mm" cy="2.5mm"
```

The icc-color statements are supposed to operate such that it looks up the named colour (token) in the icc profile and if not found it uses a given sRGB default value. In the above example a colour named 'LANDF' looked up from the associated colour profile called "S100Colors". If the colour is not found in the referenced 'S100Colors' icc profile file or the system does not support icc-colors then the given sRGB value is used such as #8D642E.

The icc-color profile file is defined as being encoded according to ICC 42. There are few examples of this format available and the spec is rather crptic. It appears to be a binary tag value encoding.

> *International Color Consortium. Specification ICC.1:2004-10 (Profile version 4.2.0.0) Image technology colour management — Architecture, profile format, and data structure.*
> Available at http://www.color.org/ICC1V42.pdf

This icc-color statement seems to work in inkscape but not in Explorer, firefox or open source SVG drawing libraries such as cairo. Due to the lack of icc-color support and the complexity perhaps a low tek option should be considered.

The S-52 CIE color values are defined using 'xyL' or 'xyY' which are not commonly used for display. Apparently though they are more commonly used by calibration measuring equipment.
The 'xyL' values can be mathematically converted into CIE 'XYZ' or CIE 'Lab' values.

We can use sRGB or CIE LAB directly in SVG drawing instructions however this would not provide an easy way to substitute different colours based on the active colour palette.

Performing a mapping based on the colour value is problematic because in some colour palettes there could be two colour tokens with the same colour value but in another palette the tokens have distinct colours. The mapping /conversion needs to use the colour token as the key.

One solution, the one proposed herein is to take advantage of how SVG supports CSS. With CSS a class or style name can be used in a graphic element and the definition of this style can exist in a separate CSS file.

A graphic element can be defined such as this

```
<circle class="fLANDF" cx="2.5mm" cy="2.5mm" r="0.5mm"/>
```

fLANDF is used as a key to find the style information from the associated CSS file. The CSS entry would look something like this where the key translates to a fill style using a given sRGB value.

```
.fLANDF {fill: #8D642E }
.sLANDF {stroke:#8D642E}
```

With the CSS definitions in a separate file, swapping colour palettes is as easy as switching the CSS file. If a pattern for the naming conventions of the style class id is defined then it could include the colour token as part of the pattern. In the above example the "f" means it is a fill color style and the rest of the id is the recognizable colour token. The sLANDF translates to a stroke or pen colour for line drawing.

The definition of the color in the CSS file could also be done using the CIE Lab color such as

```
. fLANDF {fill: #8D642E cielab(63.396275, 32.890056, 53.3213007) }
.sLANDF {stroke:#8D642E cielab(63.396275, 32.890056, 53.3213007) }
```

One potential issue with this is if we are looking for strict compliance the SVG Tiny profile may not include support for CSS, especially external CSS.
From http://www.w3.org/TR/SVGTiny12/styling.html
"SVG Tiny 1.2 does not require support for CSS selectors applied to SVG content. Authors must not rely on external, author stylesheets to style documents that are intended to be used with SVG Tiny 1.2 user agents."

It seems that we could not expect an SVG Tiny 1.2 viewer to support external CSS but we could allow it in an S-100 SVG profile.

The use of a CSS file can also be used to define commonly used style settings such as the line cap and mitre style. E.g.

```
.sl {stroke-linecap:round;stroke-linejoin:round}
.f0 {fill:none}
```

Title and Description
SVG has elements for title "<title>" and description "<desc>".

```
<title>POSGEN01</title>
<desc>position of a point feature</desc>
```

SVG Drawing Command elements

From S-52 we need to support circles, filled polygons and drawing paths (moveto, drawto). SVG has all of these. S-52 also uses symbol references, where another symbol file is referenced and included into the definition. This is also possible in SVG with the 'use' element.
To convert the S-52 Point symbols into SVG the following SVG drawing elements were used:

<path>
The <path> element is used to define an open or closed shape using a sequence of positions. The 'd' attribute is used to describe the path. The 'd' attribute is a string which contains a series of sub commands. The following subcommands were used:
Moveto – uppercase 'M' which defines an absolute X,Y position in mm.
Lineto – uppercase 'L' which defines drawing a line from the current location to the absolute X,Y position given with the 'L' command. This draws a straight line segment from the last Moveto or Lineto location to the location given.
ClosePath – 'Z' closes the path with a straight line from the last Moveto or Lineto position to the first position in the path.

The 'class' attribute is used to select a CSS style to assign styling attributes. Multiple class identifiers can be assigned in a space separated list.

The 'stroke' and 'fill' styles are applied to the path element as attributes or coming from values specified in the associated CSS file found by looking up the Class id.

<circle>
The circle element is used to make filled or unfilled circles by specifying a centre and radius and applicable styles.

The ‘cx’ and ‘cy’ attributes defines the location of the centre of the circle in mm.
The ‘r’ attribute defines the radius also in mm.

The ‘class’ attribute is used to select a CSS style to assign styling attributes. Multiple class identifiers can be assigned in a space separated list.

The ‘stroke’ and ‘fill’ styles are applied to the path element as attributes or coming from values specified in the associated CSS file found by looking up the Class id.

<rect>

The ‘rect’ element is used to make a simple rectangle shape. The attributes used were:
‘x’ and ‘y’ to define the upper left location of the rectangle
‘width’ to define the width of the rectangle.
‘height’ to define the height of the rectangle.

The ‘class’ attribute is used to select a CSS style to assign styling attributes. Multiple class identifiers can be assigned in a space separated list.

The ‘stroke’ and ‘fill’ styles are applied to the path element as attributes or coming from values specified in the associated CSS file found by looking up the Class id.

The ‘use’ element was not used as it was only implemented in S-52 for the purpose of defining complex line patterns. The proposed S-100 Part 9 defines the complex line instructions as XML and not in SVG.

Stroke Attributes

The following stroke attributes can be used for drawing lines. These stroke attributes can be defined within a CSS class and applied in a common way.

‘stroke’ – used to define the color of a line. A value of ‘none’ means the line is not drawn. Color by default is defined using a hexadecimal encoding of an sRGB value.

‘stroke-width’ – used to define the line width or pen thickness when drawing lines. A value of 0 means the line is not drawn. The units will default to mm if the viewbox is configured as above.

‘stroke-opacity’ – used to control the opacity/transparency of a line. 0 is fully transparent 1 is fully opaque. 0.5 is 50% transparent.

‘stroke-linecap’ – the shape of the end of a line. Value choices are: butt | round | square | inherit.

‘stroke-linejoin’ – used on corners and where lines come together. Value choices are: miter | round | bevel | inherit.

Fill attributes

The following fill attributes can be applied to individual elements or can be assigned to a class if within a CSS file and assigned to all drawing objects referencing that class.

‘fill’ – used to define the color for filling a closed shape e.g. circle, rect or closed path. Color by default is defined using a hexadecimal encoding of an sRGB value. A value of ‘none’ means no fill.

‘fill-opacity’ – used to control the opacity/transparency of a color fill. 0 is fully transparent 1 is fully opaque. 0.5 is 50% transparent.

Specification of the overall symbol bounding box, symbol covering rectangle and pivot point.
The following elements were added to each SVG symbol to support parsing and browsing of the symbols. They would normally be defined in the CSS with a stroke of ‘none’ meaning that they are not to be drawn.

svgBox
A rectangle element being identified with a class identifier ‘svgBox’ defined as going from 0,0 to the full width and height of the SVG symbol. This can be used to draw the full cover of the SVG by assigning a stroke style to this class in the CSS file other than ‘none’.

symbolBox
A rectangle element being identified with a class identifier ‘symbolBox’ defined as being the minimum rectangle around the actual drawn portion of the symbol. This can be used to draw the cover of the symbol by assigning a stroke style to this class in the CSS file other than ‘none’.

pivotPoint
A small circle being identified with a class identifier ‘pivotPoint’ centred on the width/2 and height/2 of the SVG element and drawn with a 1mm radius that can be used to show the pivot point when browsing the symbol when the CSS class is assigned stroke style other than ‘none’.

Point Symbols

With the definitions above it is possible to define SVG symbols that are equivalent to S-52.

Line styles

In S-100 Part 9 it is proposed that linestyles be defined in XML as part of or referenced by the defined drawing instructions. The defined linestyles may refer to point symbols that will be included as part of the linestyle.

Pixmaps

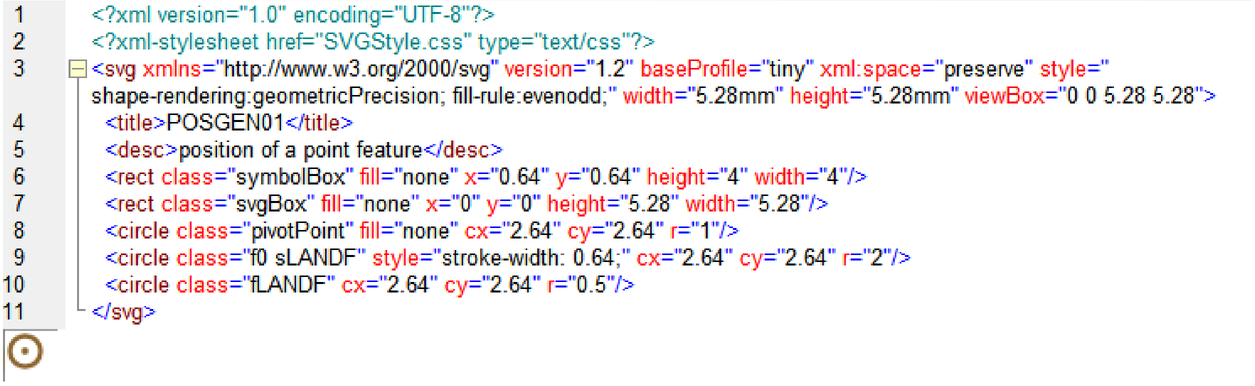
Pixmaps can be embedded into an SVG using base64 encoding or a reference can be made to an external pixmap defined as Tiff or PNG etc. The current proposal for S-100 Part 9 is to allow a simple pixmap defined in XML similar to how it was done in S-52 and also similar to the XPM format. One of the main reasons for not adopting another pixmap format is the ability to control the colours used through colour tokens.

Metadata in SVG

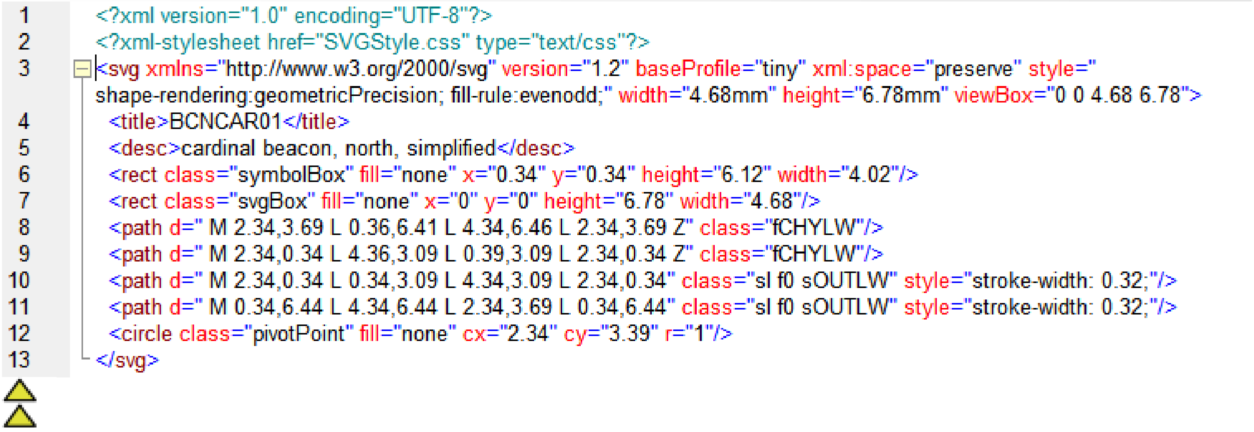
SVG includes a ‘metadata’ element as well as some extensible metadata attributes.
The ‘metadata’ element can be populated by first providing the appropriate namespace information and then the metadata content in xml form compliant with the identified namespace.

http://www.w3.org/TR/SVGTiny12/metadata.html#MetadataAttributes

Example symbols in SVG:

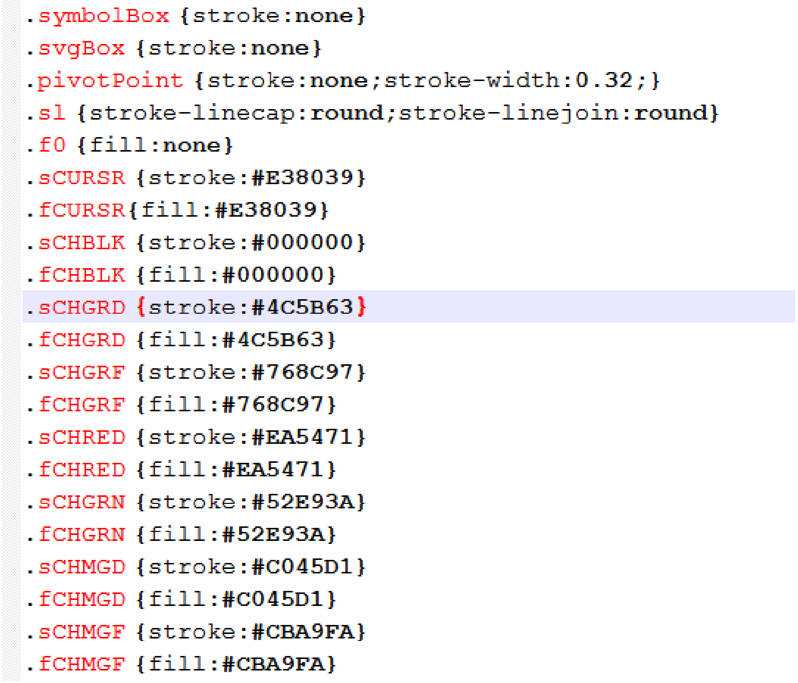


```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="SVGStyle.css" type="text/css"?>
<svg xmlns="http://www.w3.org/2000/svg" version="1.2" baseProfile="tiny" xml:space="preserve" style="
shape-rendering:geometricPrecision; fill-rule:evenodd;" width="5.28mm" height="5.28mm" viewBox="0 0 5.28 5.28">
  <title>POSGEN01</title>
  <desc>position of a point feature</desc>
  <rect class="symbolBox" fill="none" x="0.64" y="0.64" height="4" width="4"/>
  <rect class="svgBox" fill="none" x="0" y="0" height="5.28" width="5.28"/>
  <circle class="pivotPoint" fill="none" cx="2.64" cy="2.64" r="1"/>
  <circle class="f0 sLANDF" style="stroke-width: 0.64;" cx="2.64" cy="2.64" r="2"/>
  <circle class="fLANDF" cx="2.64" cy="2.64" r="0.5"/>
</svg>
```



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="SVGStyle.css" type="text/css"?>
<svg xmlns="http://www.w3.org/2000/svg" version="1.2" baseProfile="tiny" xml:space="preserve" style="
shape-rendering:geometricPrecision; fill-rule:evenodd;" width="4.68mm" height="6.78mm" viewBox="0 0 4.68 6.78">
  <title>BCNCAR01</title>
  <desc>cardinal beacon, north, simplified</desc>
  <rect class="symbolBox" fill="none" x="0.34" y="0.34" height="6.12" width="4.02"/>
  <rect class="svgBox" fill="none" x="0" y="0" height="6.78" width="4.68"/>
  <path d=" M 2.34,3.69 L 0.36,6.41 L 4.34,6.46 L 2.34,3.69 Z" class="fCHYLW"/>
  <path d=" M 2.34,0.34 L 4.36,3.09 L 0.39,3.09 L 2.34,0.34 Z" class="fCHYLW"/>
  <path d=" M 2.34,0.34 L 0.34,3.09 L 4.34,3.09 L 2.34,0.34" class="sl f0 sOUTLW" style="stroke-width: 0.32;"/>
  <path d=" M 0.34,6.44 L 4.34,6.44 L 2.34,3.69 L 0.34,6.44" class="sl f0 sOUTLW" style="stroke-width: 0.32;"/>
  <circle class="pivotPoint" fill="none" cx="2.34" cy="3.39" r="1"/>
</svg>
```

Example CSS file extract



```
.symbolBox {stroke:none}
.svgBox {stroke:none}
.pivotPoint {stroke:none;stroke-width:0.32;}
.sl {stroke-linecap:round;stroke-linejoin:round}
.f0 {fill:none}
.sCURSR {stroke:#E38039}
.fCURSR{fill:#E38039}
.sCHBLK {stroke:#000000}
.fCHBLK {fill:#000000}
.sCHGRD {stroke:#4C5B63}
.fCHGRD {fill:#4C5B63}
.sCHGRF {stroke:#768C97}
.fCHGRF {fill:#768C97}
.sCHRED {stroke:#EA5471}
.fCHRED {fill:#EA5471}
.sCHGRN {stroke:#52E93A}
.fCHGRN {fill:#52E93A}
.sCHMGD {stroke:#C045D1}
.fCHMGD {fill:#C045D1}
.sCHMGF {stroke:#CBA9FA}
.fCHMGF {fill:#CBA9FA}
```

Symbol Conversion to SVG and comparison to Preslib_3.3Addendum.doc

The symbol aissel01.sym seems to have an error.
The height should be 01450 instead of 14500 which is in the header. Corrected in SVG.

BCNSAW13, BCNSAW 21, BCNSPP13, BCNSPP 21,BOYDEF03,BOYLAT13, BOYLAT14, BOYLAT23, BOYLAT24, BOYSAW12, BOYSPP11, BOYSPP15, BOYSPP25, BOYSPP35, BCNDEF13,BCNLAT15,BCNLAT16, BCNLAT21, BCNLAT22, seem to have centre point circle with radius .15mm but should be .3mm

BOYCAR01, BOYCAR02, BOYCAR03,BOYCAR04 lineweight is 0.3 but addendum says 0.6. 0.3 looks better.

BOYMOR11 point radius is 0.71 but comments says 1.01mm

BUIREL05 diameter of point should be 0.7 radius .35
DNGHILIT, DWRUTE51,LIGHTDEF, all light flares boundary transparent? Set to not.
TSLDEF51, TSSLPT51 boundary not transparent.

Symbols converted into SVG:

achare02.svg
achare51.svg
achbrt07.svg
achres51.svg
achres61.svg
achres71.svg
airare02.svg
aisdef01.svg
aisdgr01.svg
aislst01.svg
aissel01.svg
aisslp01.svg
aistrn01.svg
aistrn02.svg
aisves01.svg
arpatg01.svg
arpone01.svg
arpsix01.svg
bcncar01.svg
bcncar02.svg
bcncar03.svg
bcncar04.svg
bcndef13.svg
bcngen01.svg
bcngen03.svg
bcnisd21.svg
bcnlat15.svg
bcnlat16.svg
bcnlat21.svg
bcnlat22.svg
bcnltc01.svg
bcnsaw13.svg
bcnsaw21.svg
bcnspp13.svg
bcnspp21.svg
bcnstk02.svg
bcntow01.svg
blkadj01.svg
boybar01.svg
boycan01.svg
boycar01.svg
boycar02.svg
boycar03.svg
boycar04.svg
boycon01.svg
boydef03.svg
boygen03.svg
boyinb01.svg
boyisd12.svg
boylat13.svg
boylat14.svg
boylat23.svg
boylat24.svg
boymor01.svg
boymor03.svg
boymor11.svg
boypil01.svg
boysaw12.svg
boysph01.svg
boyspp11.svg
boyspp15.svg
boyspp25.svg
boyspp35.svg
boyspr01.svg
boysup01.svg
boysup02.svg
boysup03.svg
bridge01.svg
brthno01.svg
buaare02.svg
buirel01.svg
buirel04.svg
buirel05.svg
buirel13.svg
buirel14.svg
buirel15.svg
buisgl01.svg
buisgl11.svg
cairns01.svg
cairns11.svg
cblare51.svg
cgusta02.svg
chcrdel1.svg
chcrid01.svg
chimny01.svg
chimny11.svg
chinfo06.svg
chinfo07.svg
chinfo08.svg
chinfo09.svg
chinfo10.svg
chinfo11.svg
chksym01.svg
clrlin01.svg
cranes01.svg
ctnare51.svg
ctyare51.svg
ctyare71.svg
curdef01.svg
curent01.svg
cursra01.svg
cursrb01.svg
danger01.svg
danger02.svg
danger03.svg
daysqr01.svg
daysqr21.svg
daytri01.svg
daytri05.svg
daytri21.svg
daytri25.svg
deleatur2.svg
dirboy01.svg
dirboya1.svg
dirboyb1.svg
dismar03.svg
dismar04.svg
dismar07.svg
dnghilit.svg
domes001.svg
domes011.svg
dshaer01.svg
dshaer11.svg
dwrtpt51.svg
dwrute51.svg
ebbstr01.svg
eblvrm11.svg
emachar1.svg
emachre1.svg
emachre2.svg
emaregr1.svg
emaremg1.svg
emcblar1.svg
emcblsu1.svg
emctnar1.svg
emdrgre1.svg
emdrgre2.svg
emdwrtc1.svg
emdwrtc2.svg
emdwrut1.svg
emdwrut2.svg
ementre1.svg
emfeyrt1.svg
emfeyrt2.svg
emfshfa1.svg
emfshre1.svg
empipar1.svg
empipar2.svg
empipsl1.svg
empipsl2.svg
emprcar1.svg
emquesm1.svg
emquesm2.svg
emquesm3.svg
emrcrtc1.svg
emrcrtc2.svg
emrectr1.svg
emrectr2.svg
emresar1.svg
emtidin1.svg
entres51.svg
entres61.svg
entres71.svg
erbltik1.svg
essare01.svg
events02.svg
fairwy51.svg
fairwy52.svg
flastk01.svg
flastk11.svg
fldobs01.svg
fldstr01.svg
flgstf01.svg
flthaz01.svg
flthaz02.svg
fogsig01.svg
forstc01.svg
forstc11.svg
foulgnd1.svg
fryare51.svg
fryare52.svg
fshfac02.svg
fshfac03.svg
fshgrd01.svg
fshhav01.svg
fshres51.svg
fshres61.svg
fshres71.svg
gatcon03.svg
gatcon04.svg
hiltop01.svg
hiltop11.svg
hrbfac09.svg
hulkes01.svg
ihosyms.txt
infare51.svg
inform01.svg
isodgr01.svg
itzare51.svg
lightdef.svg
lights01.svg
lights02.svg
lights03.svg
lights11.svg
lights12.svg
lights13.svg
lights81.svg
lights82.svg
litdef11.svg
litflt01.svg
litflt02.svg
litves01.svg
litves02.svg
lndare01.svg
locmag01.svg
locmag51.svg
lowacc01.svg
magvar01.svg
magvar51.svg
marcul02.svg
monumt02.svg
monumt12.svg
morfac03.svg
morfac04.svg
mstcon04.svg
mstcon14.svg
northar1.svg
notbrd11.svg
obstrn01.svg
obstrn02.svg
obstrn03.svg
obstrn11.svg
ofsplf01.svg
ospone02.svg
ospsix02.svg
oversc01.svg
oversc11.svg
oversc12.svg
ownshp01.svg
ownshp05.svg
pastrk01.svg
pastrk02.svg
pilbop02.svg
pilpnt02.svg
plnpos01.svg
plnpos02.svg
plnspd03.svg
plnspd04.svg
posgen01.svg
posgen03.svg

posgen04.svg
positn02.svg
prcare12.svg
prcare51.svg
prdins02.svg
pricke03.svg
pricke04.svg
pssare01.svg
quapos01.svg
quarry01.svg
quesmrk1.svg
racnsp01.svg
radrfl03.svg
rascan01.svg
rascan11.svg
rcldef01.svg
rctlpt52.svg
rdocal02.svg
rdocal03.svg
rdosta02.svg
recdef51.svg
rectrc55.svg
rectrc56.svg
rectrc57.svg
rectrc58.svg
refpnt02.svg
retrfl01.svg
retrfl02.svg
rfnery01.svg
rfnery11.svg
rolrol01.svg
rscsta02.svg
rsrdef51.svg
rtldef51.svg
rtpbcn02.svg
safcon00.svg
safcon01.svg
safcon02.svg
safcon03.svg
safcon04.svg
safcon05.svg
safcon06.svg
safcon07.svg
safcon08.svg
safcon09.svg
safcon10.svg
safcon11.svg
safcon12.svg
safcon13.svg
safcon14.svg
safcon15.svg
safcon16.svg
safcon17.svg
safcon18.svg
safcon19.svg
safcon20.svg
safcon21.svg
safcon22.svg
safcon23.svg
safcon24.svg
safcon25.svg
safcon26.svg
safcon27.svg
safcon28.svg
safcon29.svg
safcon30.svg
safcon31.svg
safcon32.svg
safcon33.svg
safcon34.svg
safcon35.svg
safcon36.svg
safcon37.svg
safcon38.svg
safcon39.svg
safcon40.svg
safcon41.svg
safcon42.svg
safcon43.svg
safcon44.svg
safcon45.svg
safcon46.svg
safcon47.svg
safcon48.svg
safcon49.svg
safcon50.svg
safcon51.svg
safcon52.svg
safcon53.svg
safcon54.svg
safcon55.svg
safcon56.svg
safcon57.svg
safcon58.svg
safcon59.svg
safcon60.svg
safcon61.svg
safcon62.svg
safcon63.svg
safcon64.svg
safcon65.svg
safcon66.svg
safcon67.svg
safcon68.svg
safcon69.svg
safcon70.svg
safcon71.svg
safcon72.svg
safcon73.svg
safcon74.svg
safcon75.svg
safcon76.svg
safcon77.svg
safcon78.svg
safcon79.svg
safcon80.svg
safcon81.svg
safcon82.svg
safcon83.svg
safcon84.svg
safcon85.svg
safcon86.svg
safcon87.svg
safcon88.svg
safcon89.svg
safcon90.svg
safcon91.svg
safcon92.svg
safcon93.svg
safcon94.svg
safcon95.svg
safcon96.svg
safcon97.svg
safcon98.svg
safcon99.svg
scaleb10.svg
scaleb11.svg
shlbox01.svg
silbui01.svg
silbui11.svg
sistat02.svg
smcfac02.svg
sndwav02.svg
soundg00.svg
soundg01.svg
soundg02.svg
soundg03.svg
soundg04.svg
soundg05.svg
soundg06.svg
soundg07.svg
soundg08.svg
soundg09.svg
soundg10.svg
soundg11.svg
soundg12.svg
soundg13.svg
soundg14.svg
soundg15.svg
soundg16.svg
soundg17.svg
soundg18.svg
soundg19.svg
soundg20.svg
soundg21.svg
soundg22.svg
soundg23.svg
soundg24.svg
soundg25.svg
soundg26.svg
soundg27.svg
soundg28.svg
soundg29.svg
soundg30.svg
soundg31.svg
soundg32.svg
soundg33.svg
soundg34.svg
soundg35.svg
soundg36.svg
soundg37.svg
soundg38.svg
soundg39.svg
soundg40.svg
soundg41.svg
soundg42.svg
soundg43.svg
soundg44.svg
soundg45.svg
soundg46.svg
soundg47.svg
soundg48.svg
soundg49.svg
soundg50.svg
soundg51.svg
soundg52.svg
soundg53.svg
soundg54.svg
soundg55.svg
soundg56.svg
soundg57.svg
soundg58.svg
soundg59.svg
soundgb1.svg
soundgc2.svg
sounds00.svg
sounds01.svg
sounds02.svg
sounds03.svg
sounds04.svg
sounds05.svg
sounds06.svg
sounds07.svg
sounds08.svg
sounds09.svg
sounds10.svg
sounds11.svg
sounds12.svg
sounds13.svg
sounds14.svg
sounds15.svg
sounds16.svg
sounds17.svg
sounds18.svg
sounds19.svg
sounds20.svg
sounds21.svg
sounds22.svg
sounds23.svg
sounds24.svg
sounds25.svg
sounds26.svg
sounds27.svg
sounds28.svg
sounds29.svg
sounds30.svg
sounds31.svg
sounds32.svg
sounds33.svg
sounds34.svg
sounds35.svg
sounds36.svg
sounds37.svg
sounds38.svg
sounds39.svg
sounds40.svg
sounds41.svg
sounds42.svg
sounds43.svg
sounds44.svg
sounds45.svg
sounds46.svg
sounds47.svg
sounds48.svg
sounds49.svg
sounds50.svg
sounds51.svg
sounds52.svg
sounds53.svg
sounds54.svg
sounds55.svg
sounds56.svg
sounds57.svg
sounds58.svg
sounds59.svg
soundsa1.svg
soundsb1.svg
soundsc2.svg
spring02.svg
svgStyle.css
swpare51.svg
tidcur01.svg
tidcur02.svg
tidcur03.svg
tideht01.svg
tidstr01.svg
tmardef1.svg
tmardef2.svg
tmbyrd01.svg
tnkcon02.svg
tnkcon12.svg
tnkfrm01.svg
tnkfrm11.svg
todga035.svg
todga055.svg
todzb035.svg
topmar02.svg
topmar04.svg
topmar05.svg
topmar06.svg
topmar07.svg
topmar08.svg
topmar10.svg
topmar12.svg
topmar13.svg
topmar14.svg
topmar16.svg
topmar17.svg
topmar18.svg
topmar22.svg
topmar24.svg
topmar25.svg
topmar26.svg
topmar27.svg
topmar28.svg
topmar30.svg
topmar32.svg
topmar33.svg
topmar34.svg
topmar36.svg
topmar65.svg
topmar85.svg
topmar86.svg
topmar87.svg
topmar88.svg
topmar89.svg
towers01.svg
towers02.svg
towers03.svg
towers05.svg
towers12.svg
towers15.svg
trepnt04.svg
trepnt05.svg
tsldef51.svg
tsscrs51.svg
tsslpt51.svg
tssron51.svg
twrdef51.svg
twrtpt52.svg
twrtpt53.svg
unitfth1.svg
unitmtr1.svg
uwtroc03.svg
uwtroc04.svg

vecgnd01.svg
vecgnd21.svg
vecwtr01.svg
vecwtr21.svg
wattur02.svg
waypnt01.svg
waypnt03.svg
waypnt11.svg
wedklp03.svg
wimcon01.svg
wimcon11.svg
wndfrm51.svg
wndfrm61.svg
wndmil02.svg
wndmil12.svg
wrecks01.svg
wrecks04.svg
wrecks05.svg

Conclusions

A first cut of point symbols has been converted by CARIS into simple SVG symbols.

Recommendations

Point symbols need to be reviewed for correctness.

Justification and Impacts

This gives an initial set of point symbols for S-101 testing phase.

Symbols in SVG are easy to examine, the SVG format is quite readable.

SVG symbols can be readily viewed by opening the symbols in a web browser or desktop app that supports SVG.

Action Required of DIPWG

The DIPWG is invited to:

a. Review the symbols for correctness and if any are missing.