ECDIS S-411 for Antarctica

Dr. Jürgen Holfort

Head of German ice service at the Bundesamt für Seeschifffahrt und Hydrographie
Member of the JCOMM Expert team on sea ice

May 2016

"S-411 Ice Information Product Specification" summary

Development was led by the BSH as a part of JCOMM/ETSI. It was adopted as version 1.0 by JCOMM ETSI-5, March 2014 with the latest version 1.1.0 (June 2014)

The S-411 is fully based on the IHO S-100 framework specification, Geography Markup Language (GML), Encoding Standard and the ISO 19100 series of standards.

It is a vector product specification that is primarily intended for encoding the extent and nature of Sea Ice for navigational purpose.

The application schema of ice information product contains 28 feature types with their attributes, enumerations, is based on the Ice Objects Catalogue (Version 5.1) and can be found in the ICE domain of the IHO Registry. The full schema as XML Schema File included in Annex B – Data Product format (encoding).

Ice information datasets use S-100 Level 3a geometry which supports 0-, 1-, and 2-dimensional objects (points, line strings, polygons)

The portrayal specification is based on Styled Layer Descriptors (SLD), follows OGC standards and supports 3 polygon portrayals, one according to the vessels ice capabilities, the second and third one being the WMO ice concentration/stages of development colour codes. Portrayals for line and point objects follow the WMO symbology and are implemented using SVG-graphics.

Besides future ECDIS SDKs, the S-411 data could actually be read and presented using open source GIS software (e.g. QGis) with python scripts from the national ice services available to convert shapefiles, specially SIGRID-3 files, into the S-411 format.

S-411

Charts in S-411 format are available in the JCOMM Ice Logistics Portal http://www.bsis-ice.de/IcePortal/.

there are also available over ftp at ftp://ftp.bsh.de/outgoing/Eisbericht/S411/

INTERNATIONAL HYDROGRAPHIC ORGANIZATION

JCOMM

Ice Information Product Specification

Edition 1.1.0, June 2014

Special Publication JCOMM S-411

Published by the
International Hydrographic Bureau
4, Quai Antoine 1er
B.P. 445 - MC 98011 MONACO Cedex
Principauté de Monaco
Telefax: (377) 93 10 81 40
E-mail: info@ihb.mc
Web: www.iho.int

S-411 Production 1: setup

```
#
# This the main program part to for the Antarctic data
import struct
import shutil
import glob
import tarfile
import zipfile
from ftplib import FTP
import os
import datetime, time
from xml.etree import ElementTree as ET
import arcpy
import geometry
import s411
import s411metadata
import s411objects
from sigrid3 import sigrid3ToS411Objects
from s411 import createS411DataSet
from s411 import createS411ExchangeSet
from s411 import s411ObjectsToGml
from s411 import zipdir

# path for input files and S411 output files
PRODUKT='Ant'
ekPath = u'X:\S411_Specification\production\\'+PRODUKT+'Input'
OutPath= u'X:\\S411_Specification\production\\'+PRODUKT+'S411'
InName= u'*_ant*pl*'
```

S-411 Production 2: getting data

```
#
# Getting the newest zip files from ftp
INPUT_ftp= FTP('ice.aari.aq','USER','PASSWORD')
RemoteDir=['/latest/aari/','/latest/nic/','/latest/nis/']
for R_dir in RemoteDir:
        INPUT_ftp.cwd(R_dir)
        F_dir=INPUT_ftp.nlst()
        for F_tar in F_dir:
                fileName, fileExtension = os.path.splitext(F_tar)
                if fileExtension != '.zip' : continue
                if os.path.isfile(os.path.join(ekPath,F_tar)): continue
                # we could also check if the S411 file is already present, so we could          # delete tarfiles and shapefiles
                T_file=open(os.path.join(ekPath,F_tar),'wb')
                INPUT_ftp.retrbinary('RETR '+F_tar,T_file.write)
                T_file.close()
# the S411 files will be put back in the subdirectory S411
INPUT_ftp.cwd('/S411')

^
```

```
# Search for Zip files and decompres files starting with InName
# as long as S411-Zipfile is not present
Alltar=glob.glob(os.path.join(ekPath,InName+'.zip'))
for Datei in Alltar:
        filename = os.path.basename(Datei)
        ICEshape = filename.split(".")[0]
        N_411='S411_'+ICEshape
        if glob.glob(os.path.join(OutPath,N_411+'.zip')): continue
        zfile = zipfile.ZipFile(Datei)
        for name in zfile.namelist():
                (dirname, zfilename) = os.path.split(name)
                zfile.extract(name, ekPath)
```

```
# Search for Shape files starting with Inputname
# and start S411 production
# as long as S411-Zipfile is not present
Allshape=glob.glob(os.path.join(ekPath,InName+'*.shp'))
for Datei in Allshape:
        filename = os.path.basename(Datei)
        ICEshape = filename.split(".")[0]
        N_411='S411_'+ICEshape
        if glob.glob(os.path.join(OutPath,N_411+'.zip')): continue
        print 'Starting S411 chart production:'+N_411

        dumy=ICEshape.find('_20')
        JDATUM=ICEshape[dumy+1:dumy+9]
        datum = datetime.datetime.strptime(JDATUM,"%Y%m%d")
        dateStamp0=datum.strftime("%Y%m%d")
        dateStampS=datum.strftime("%Y%m%d")
        dateStampE=(datum + datetime.timedelta(days=5)).strftime("%Y%m%d")
        extent = [0,0,0,0]
```

```
# For exchange of ice data WGS84 (EPSG: 4326) must be used.
# set the workspace environment
arcpy.env.workspace = ekPath
# create a spatial reference object to be used as output coordinate system
out_sr = arcpy.CreateSpatialReference_management(4326)
# use the output of CreateSpatialReference as input to Project tool
# to reproject the shapefile
spatialRef = arcpy.Describe(os.path.join(ekPath,ICEshape+'.shp')).spatialReference
if ICEshape.find('aari')>=0:
        # special test for AARI Antarctic
        out_sr2 = arcpy.CreateSpatialReference_management(3031)
        # use the output of CreateSpatialReference as input to Project tool
        arcpy.Project_management(os.path.join(ekPath,ICEshape+'.shp'), os.path.join(ekPath,'DUMY'+ICEshape), out_sr2)
        ICEshape='DUMY'+ICEshape
        spatialRef = arcpy.Describe(os.path.join(ekPath,ICEshape+'.shp')).spatialReference
if spatialRef.factoryCode != 4326 :
        arcpy.Project_management(os.path.join(ekPath,ICEshape+'.shp'), os.path.join(ekPath,'WGS'+ICEshape), out_sr)
        ICEshape='WGS'+ICEshape
```

S-411 Production 6: main conversion

```
fileList = []
fileList.append(os.path.join(ekPath,ICEshape+'.shp'))
extent = geometry.getExtent(fileList)

s411FeatureList = sigrid3ToS411Objects(os.path.join(ekPath,ICEshape+'.shp'))
s411FeatureElementsList = s411ObjectsToGml(s411FeatureList)
datasetTree = createS411DataSet(s411FeatureElementsList)
metadataTree = s411metadata.getS411MetadataFromTemplate("mdTemplate"+PRODUKT+".xml", N_411, dateStamp0,
            extent, dateStampS, dateStampE)
# the S411 directory structure is written to the subdirectory with the icearea
createS411ExchangeSet(OutPath, N_411, datasetTree, metadataTree)
exchangeCatalogueTree = s411metadata.getS411ExchangeCatalogueFromTemplate("ecTemplate"+PRODUKT+".xml",
            metadataTree, N_411, N_411 + ".gml", extent, OutPath + os.path.sep + N_411 + os.path.sep + "data" +
            os.path.sep +  N_411 + ".gml")
exchangeCatalogueTree.write(OutPath + os.path.sep + N_411 + os.path.sep + "catalogue.xml", encoding ='utf-8',
            xml_declaration=True)
```

S-411 Production 7: zip, write ftp, clean

```python
# a S411-zipfile is written to the chartdirectory =dirPath
dumy=os.path.join(OutPath, N_411)
zf = zipfile.ZipFile(dumy+'.zip', mode='w')
zipdir(dumy,zf)
zf.close()
INPUT_ftp.storbinary('STOR '+N_411+'.zip',open(dumy+'.zip','rb'))

# remove the shapefiles with WGS datum
for dumy in glob.glob(os.path.join(ekPath,'WGS'+'*')):
        os.remove(dumy)
if ICEshape.find('aari')>=0:
        for dumy in glob.glob(os.path.join(ekPath,'DUMY'+'*')):
                os.remove(dumy)


#end of S411 production
INPUT_ftp.quit()
```

Data transmission

Data transmission to ship far up north or south is not possible with Inmarsat.
It is possible using Iridium, but transmission is slow and more expensive.
Iridium claims 10 kbit/sec but actual rates for compressed files are more like
2300 to 2400 bit/sec.
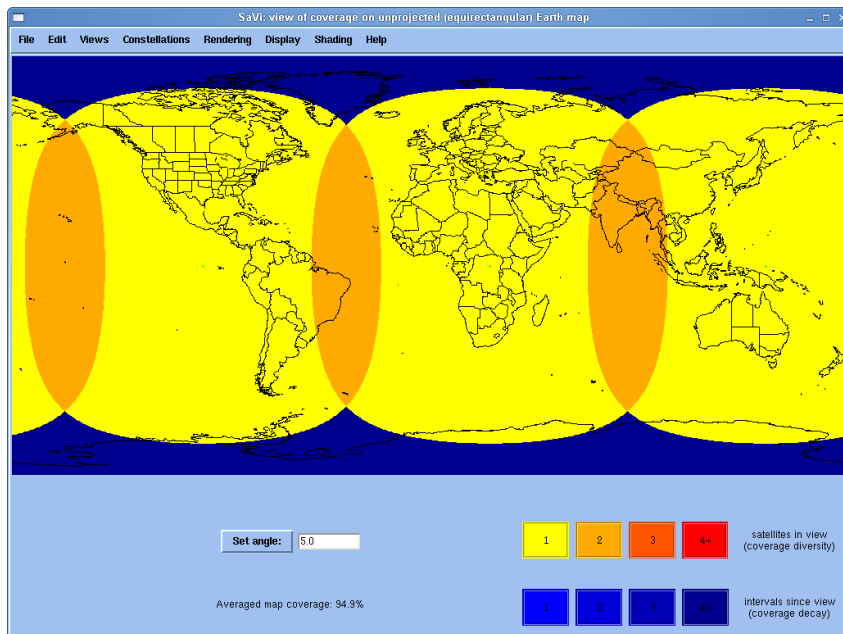
Inmarsat coverage (Wikipedia)
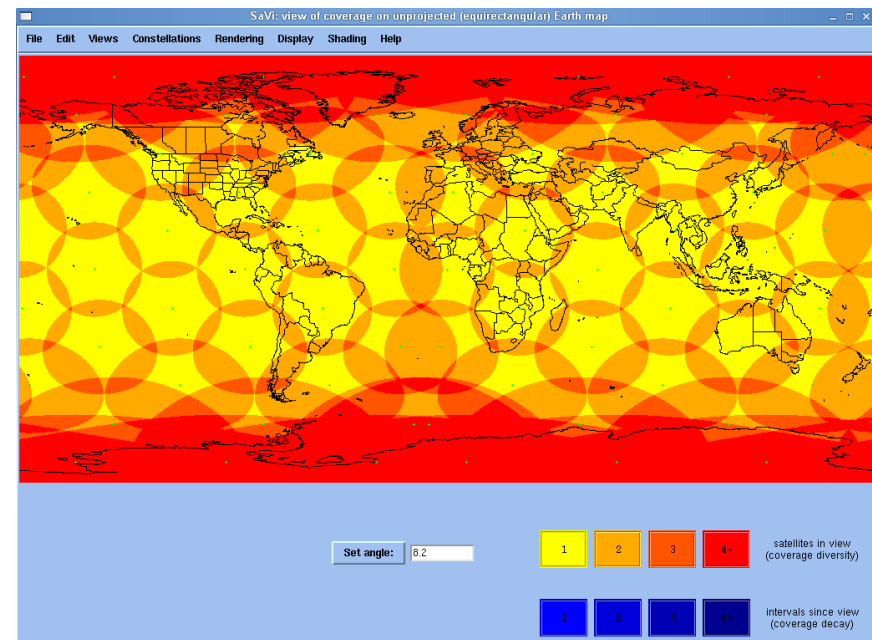
Iridium coverage (Wikipedia)

Ice charts

Bundesamt für Seeschifffahrt und Hydrographie

EISKARTE vom 19.03.2014
Spitzbergen / Barentssee
Total concentration

Ice Concentration

< 1/10 Open water
1-3/10 Very open ice
4-6/10 Open ice
7-8/10 Close ice
9-10/10 Very close ice
no data
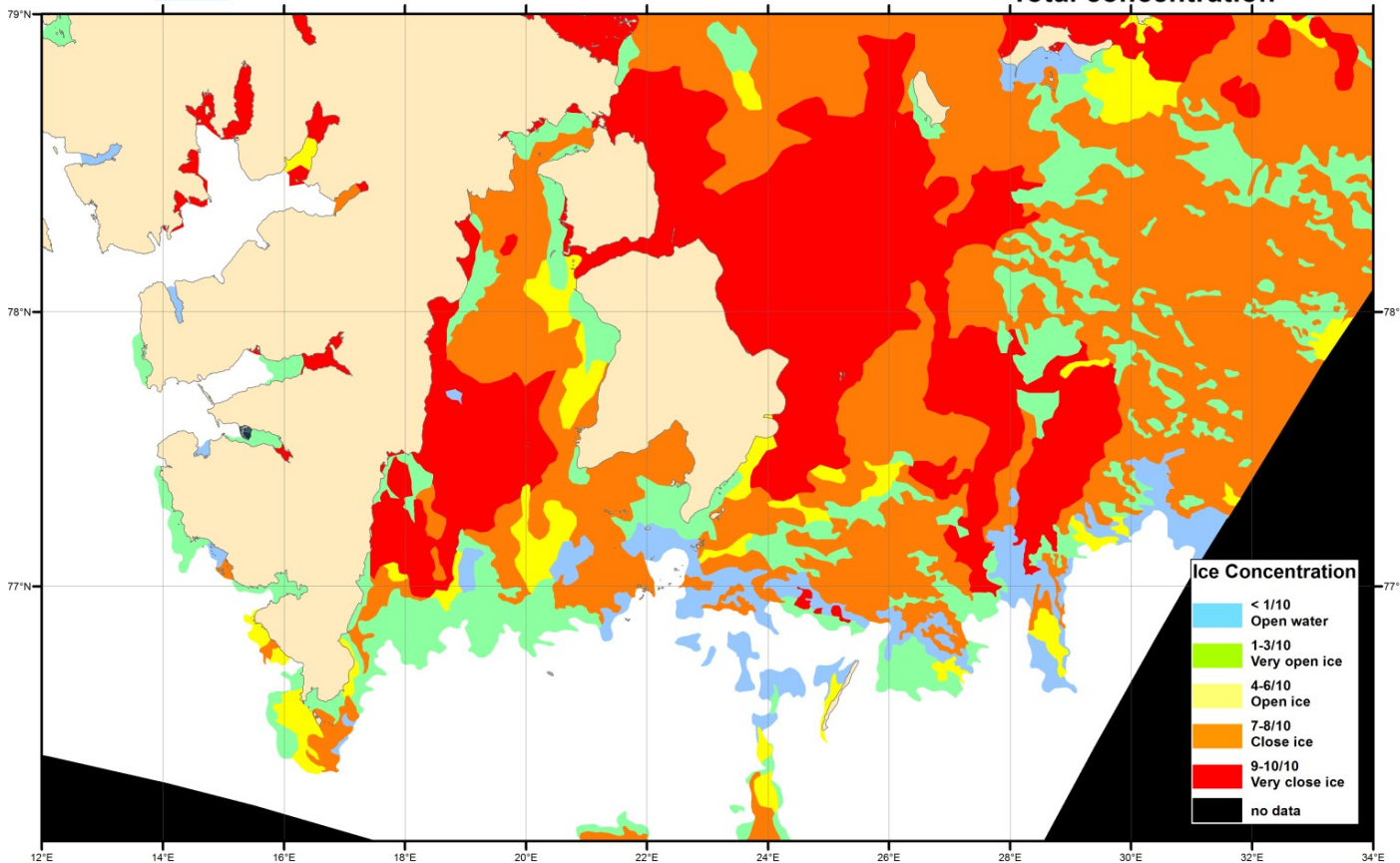
Size:
200 to 900 KB as vector PDF

About 100 to 200kb as JPEG

S-411
starting at ~20kb, typical ~60kb

Model Ice charts

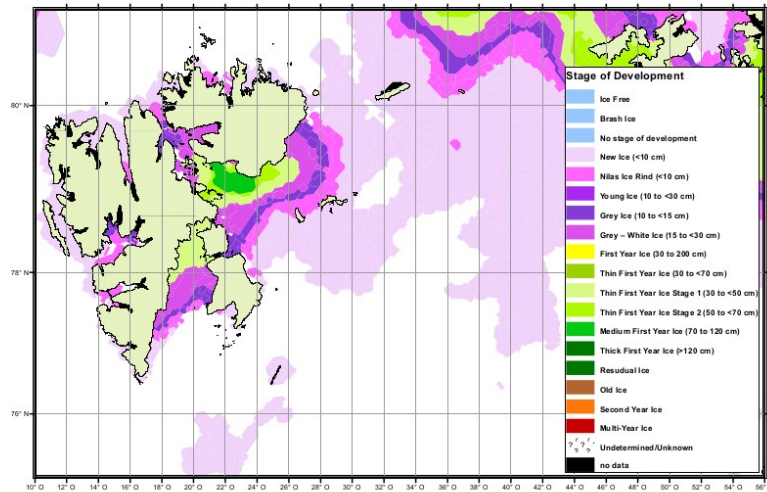
Bundesamt für Seeschifffahrt und Hydrographie

EISKARTE für den 19.03.2014, 12 Uhr
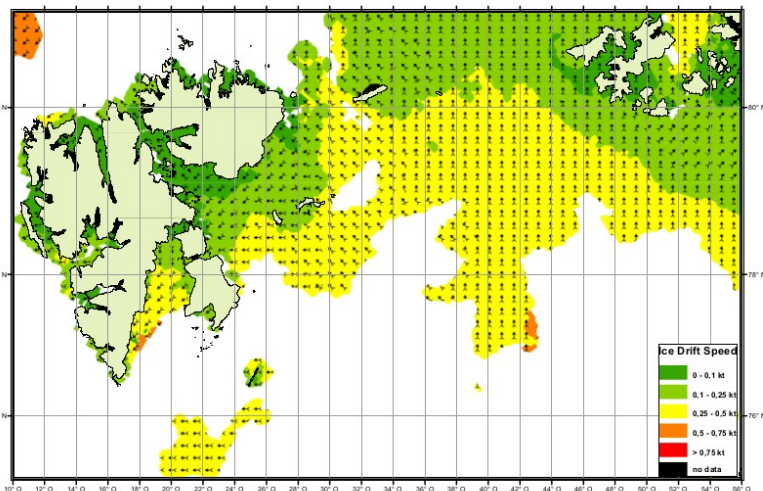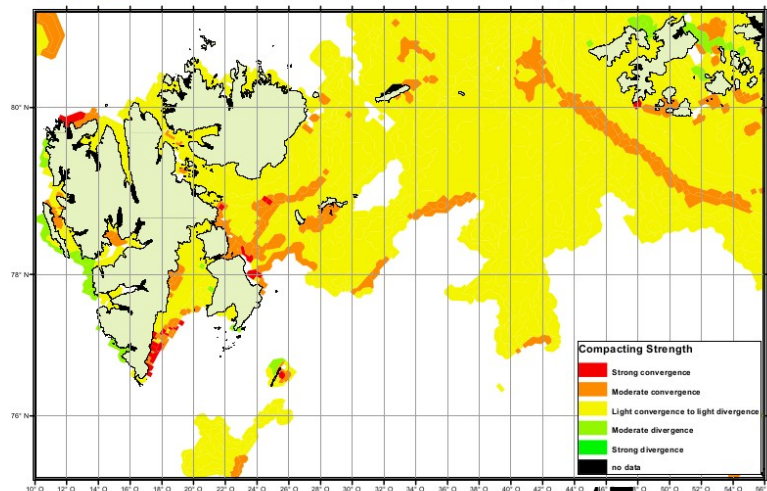Hammermodel vom 18.03.2014 12Uhr

Spitzbergen / Barentssee

Stage of Development

- Ice Free
- Brash Ice
- No stage of development
- New Ice (<10 cm)
- Nilas Ice Rind (<10 cm)
- Young Ice (10 to <30 cm)
- Grey Ice (10 to <15 cm)
- Grey – White Ice (15 to <30 cm)
- First Year Ice (30 to 200 cm)
- Thin First Year Ice (30 to <70 cm)
- Thin First Year Ice Stage 1 (30 to <50 cm)
- Thin First Year Ice Stage 2 (50 to <70 cm)
- Medium First Year Ice (70 to 120 cm)
- Thick First Year Ice (>120 cm)
- Resudual Ice
- Old Ice
- Second Year Ice
- Multi-Year Ice
- Undetermined/Unknown
- no data
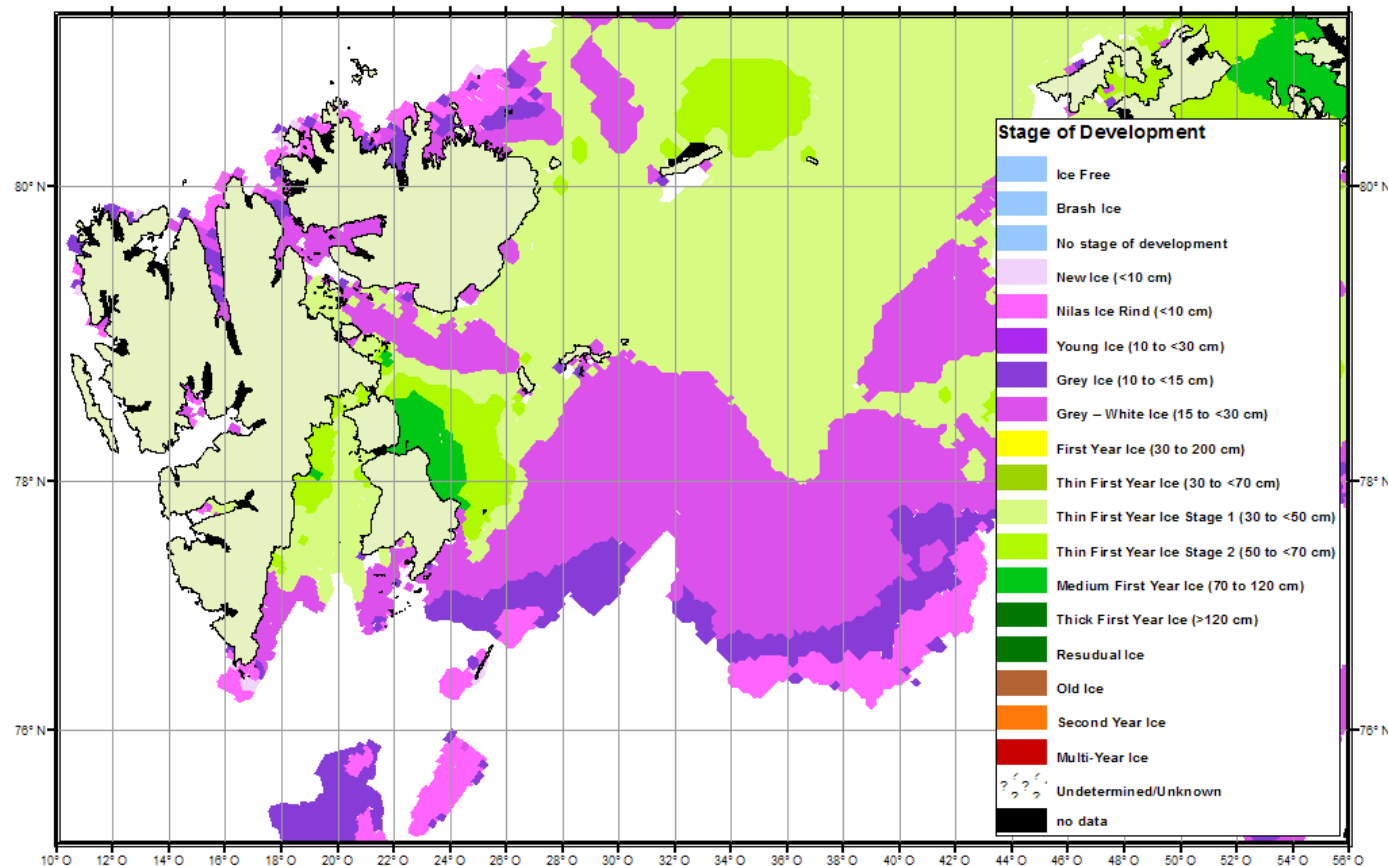
Size:
200 to 900 KB as vector PDF

About 100 to 200kb as JPEG

S-411
starting at ~20kb, typical ~60kb

A simple way to read and portray S-411.

A short, 4 page long document describes how to read and portray S-411 ice information in Quantum GIS (Version 2.0 or newer).

Qgis

QGIS is an Open-Source Geographic Information System available at no cost for Linux, Windows, Mac OsX, BSD and Android at http://qgis.org/de/site/index.html.
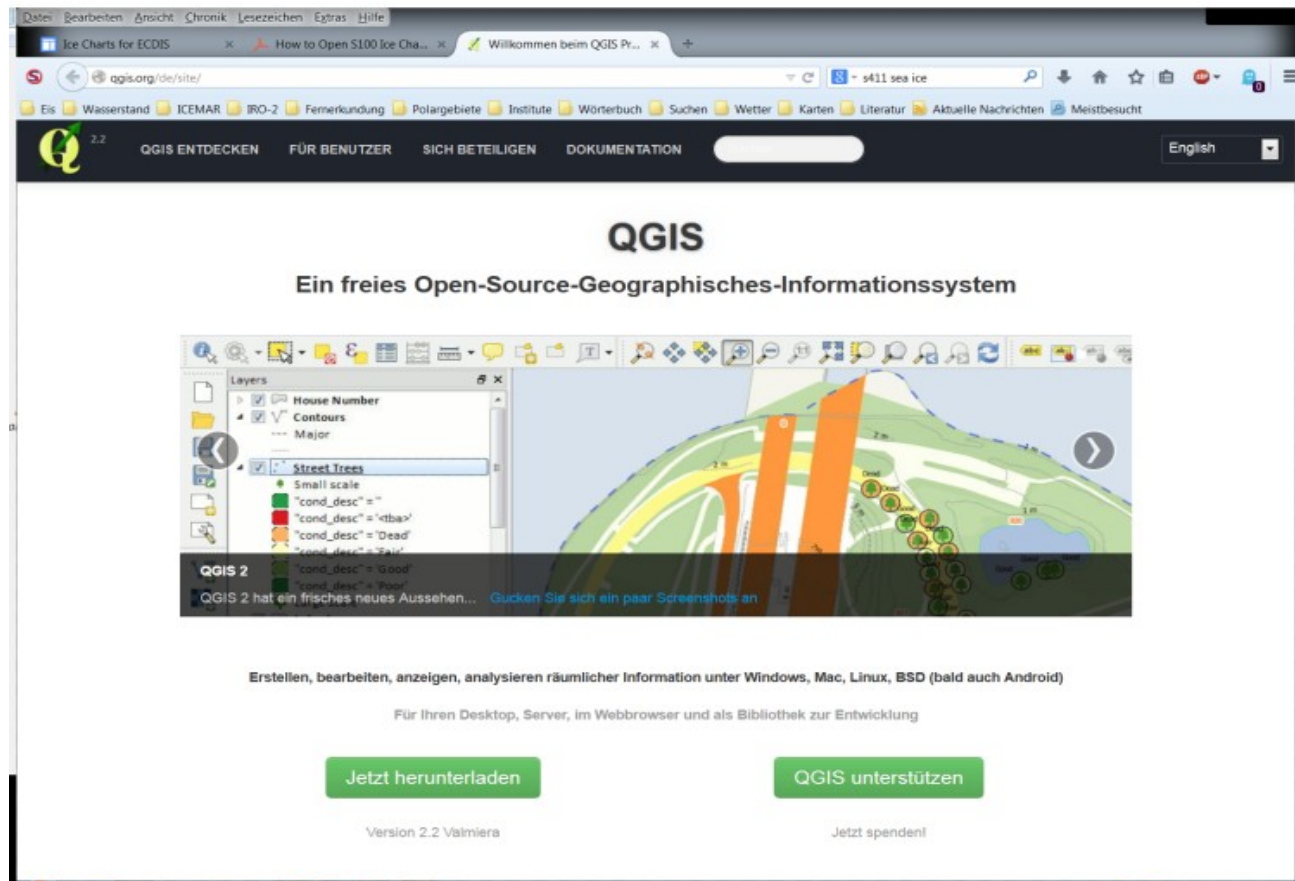
An example of S-411 data in Qgis

Example of an S-411 ice chart represented in Qgis using the respective S-411

In the upper right the total ice concentration, in the lower right the stage of development.

In the lower left is a concentration chart with additional symbols for pressure ridges, cracks, and rafted ice..
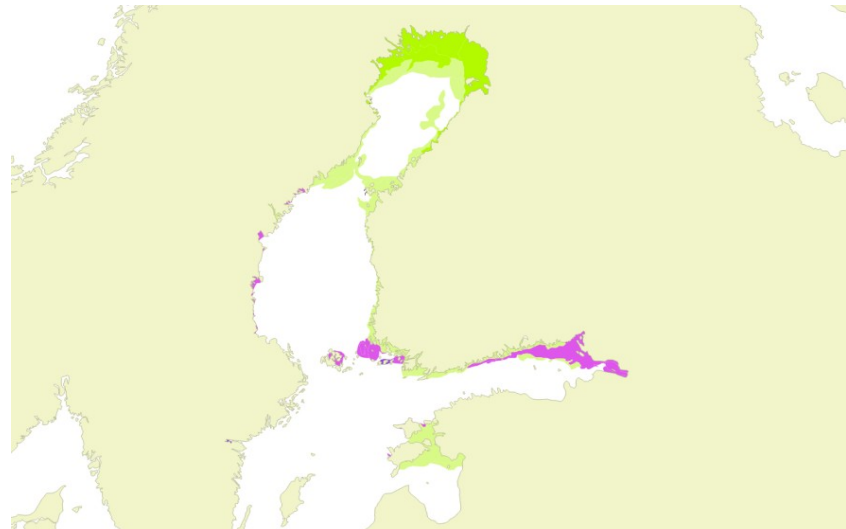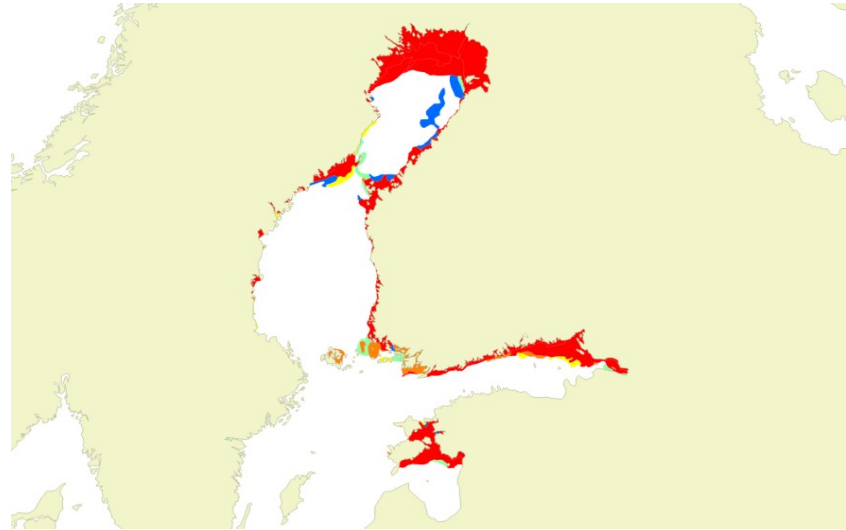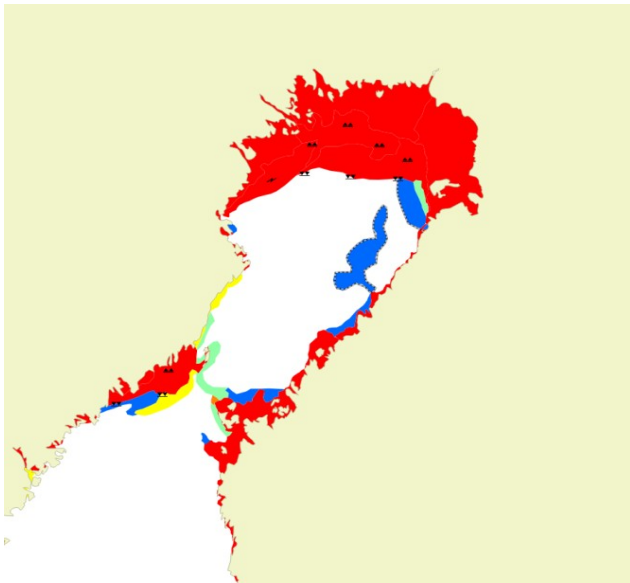
Different presentation of one S-411 dataset

WMO/IHO standard color scheme:
* Concentration
* Stage of development
* (Ship capability)

User defined color scheme:
* Drift velocity
* Divergence and convergence
* (any other)

Further development of S-411

A) better iceberg objects (points lines and polygones?; error area?)

B) changing portrayal according to scale:
    -join/dissolve/delete smaller polygons
    -change colors to white-grey at larger scales
    -group together same symbols if the overlap
    -join different overlapping symbols (e.G. rafted and ridged ice into a
     symbol denoting „heavy" ice.
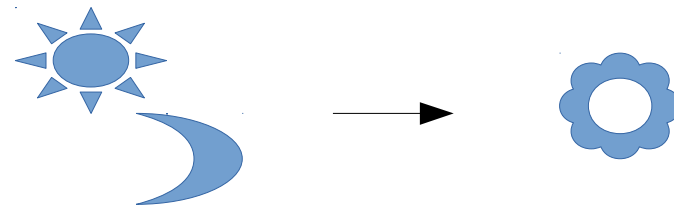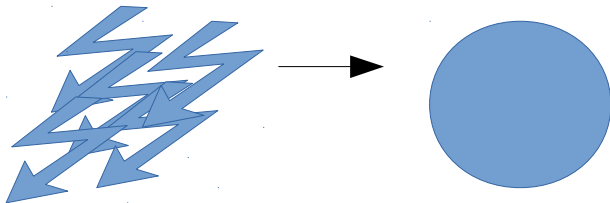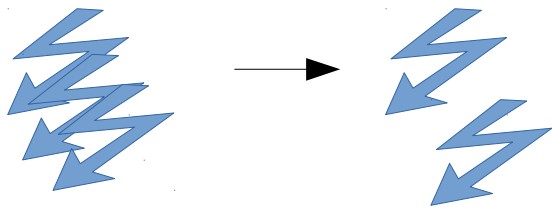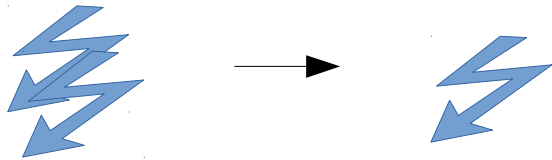   - smooth features (polygon-coverage/topology)
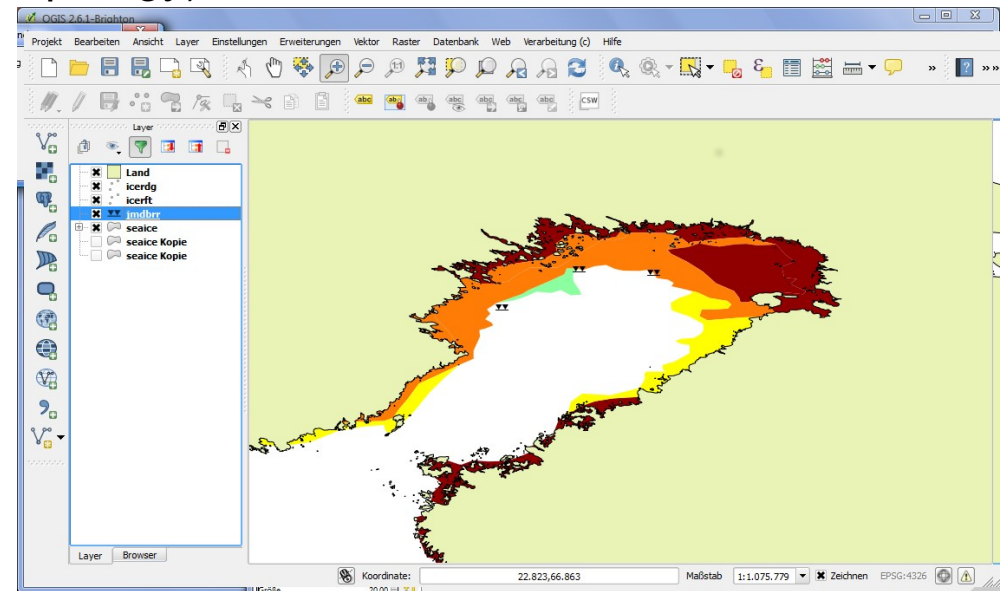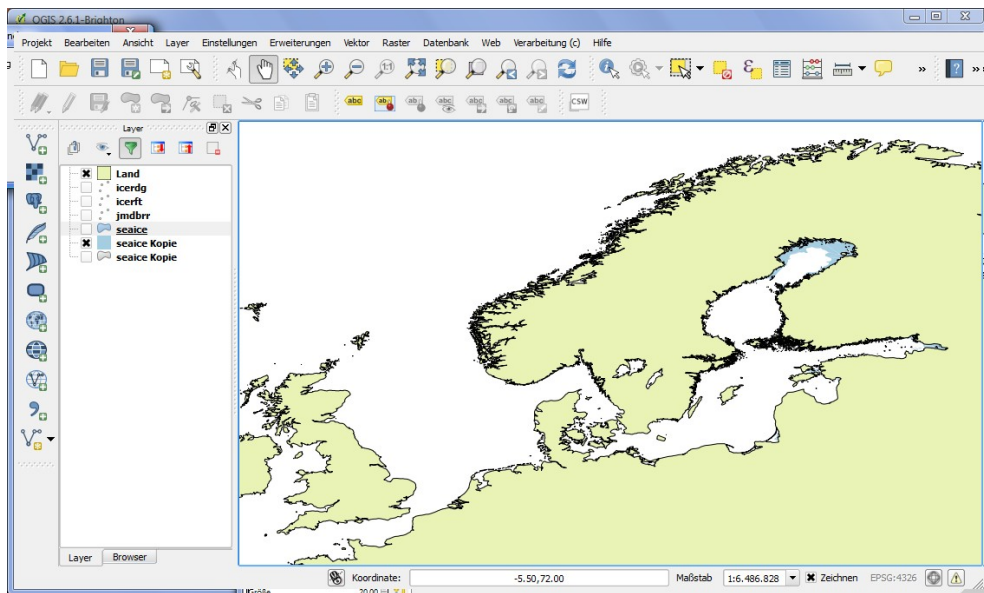
Simplification/generalization polygons
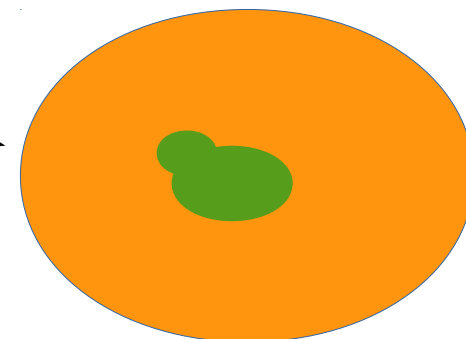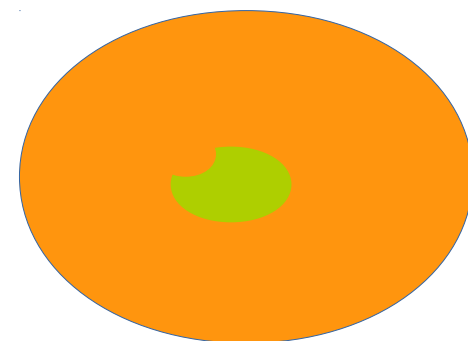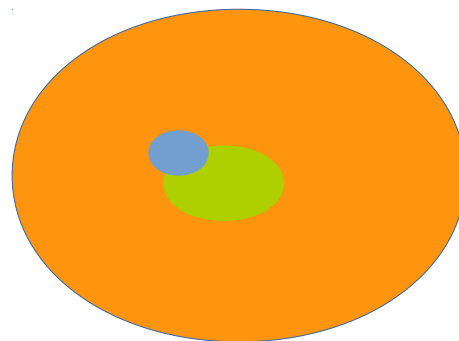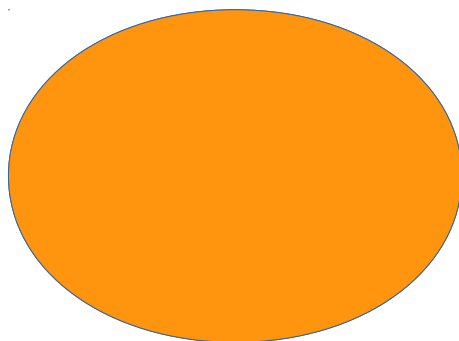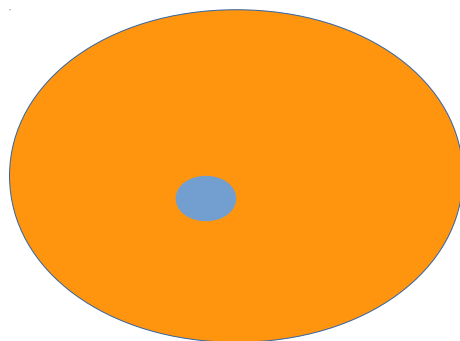
Simplification/generalization polygons
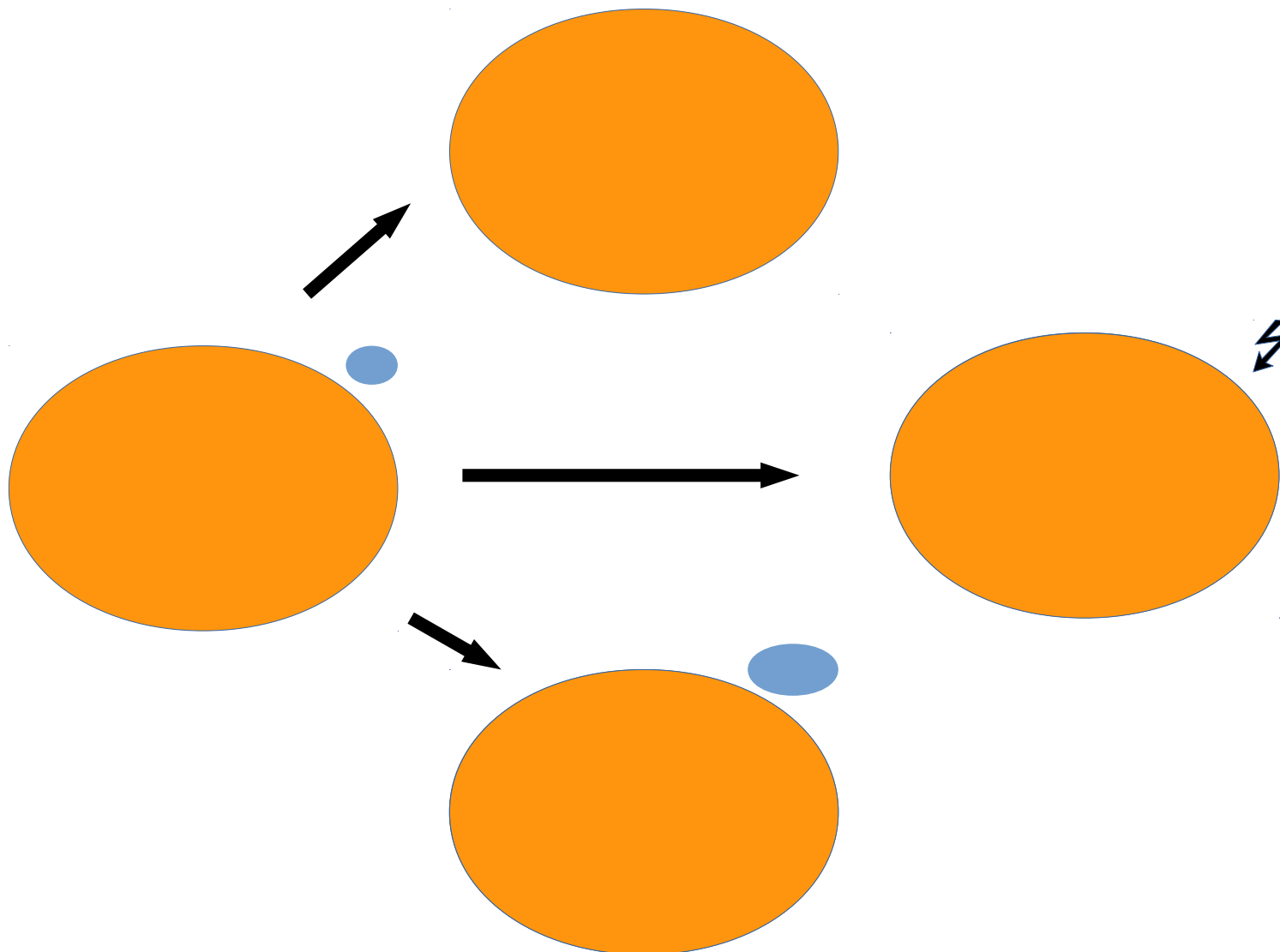
Simplification/generalization Symbols

Thanks for your attention