

# READI

## ISO 15926-14

*Anders Gjerver*  
**18.02.2022**

```
mirror_mod = modifier_ob.modifiers.new("MIRROR_X")
class mirror object to mirror_ob
mirror_mod.mirror_object = mirror_ob

operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
operation == "MIRROR_Y":
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True

selection at the end -add back the deselection
mirror_ob.select= 1
mirror_ob.select=1
obj.context.scene.objects.active = modifier_ob
name "selected" + str(modifier_ob) # modifier
mirror_ob.select = 0
obj = bpy.context.selected_objects[0]
obj.data.objects[one.name].select = 1

print("please select exactly two objects.")

OPERATOR CLASSES

class MirrorOperator(bpy.types.Operator):
    bl_label = "Mirror"
    bl_idname = "mirror"
    bl_options = {'REGISTER', 'UNDO'}

    @classmethod
    def poll(cls, context):
        obj = context.object
        if obj is None:
            return False
        if obj.type != 'MESH':
            return False
        if len(context.selected_objects) != 2:
            return False
        return True

    def execute(self, context):
        obj = context.object
        mirror = context.selected_objects[0]
        mirror.mirror_mirror_x = True
        mirror.mirror_mirror_y = False
        mirror.mirror_mirror_z = False
        return 'FINISHED'
```



~~Simplify and reduce complexity!~~

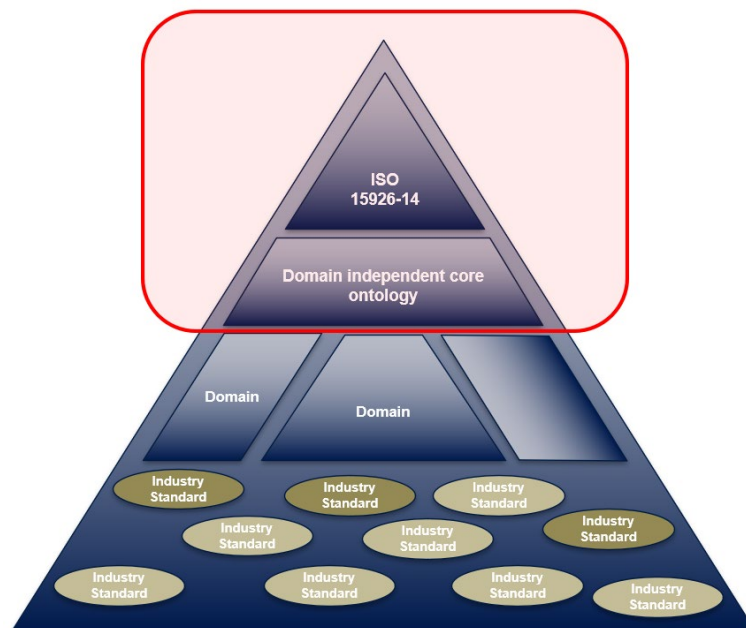
Only complexity can reduce complexity.

Capture rules and domain expertise in complex ontologies

# Core reference data

- Process activities
- Equipment types
- Units of measure and quantities
- Elements and molecules as building blocks for compound
- Relationship types
- System and functional blocks

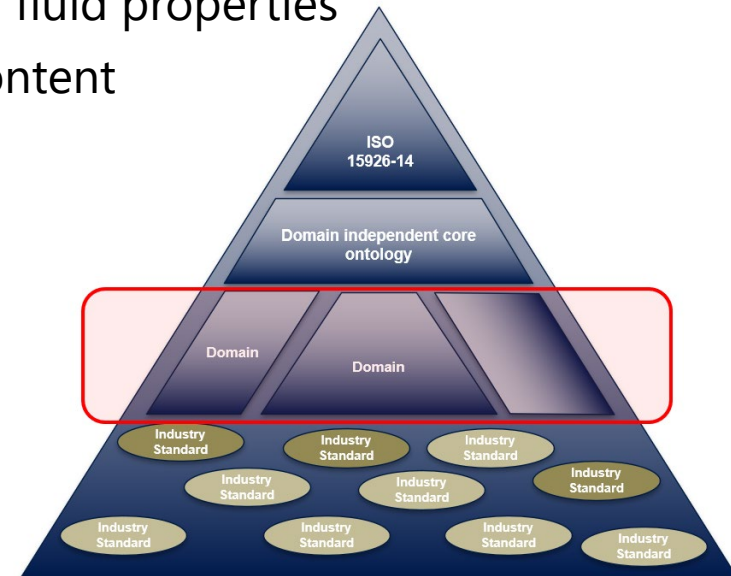
Current scope



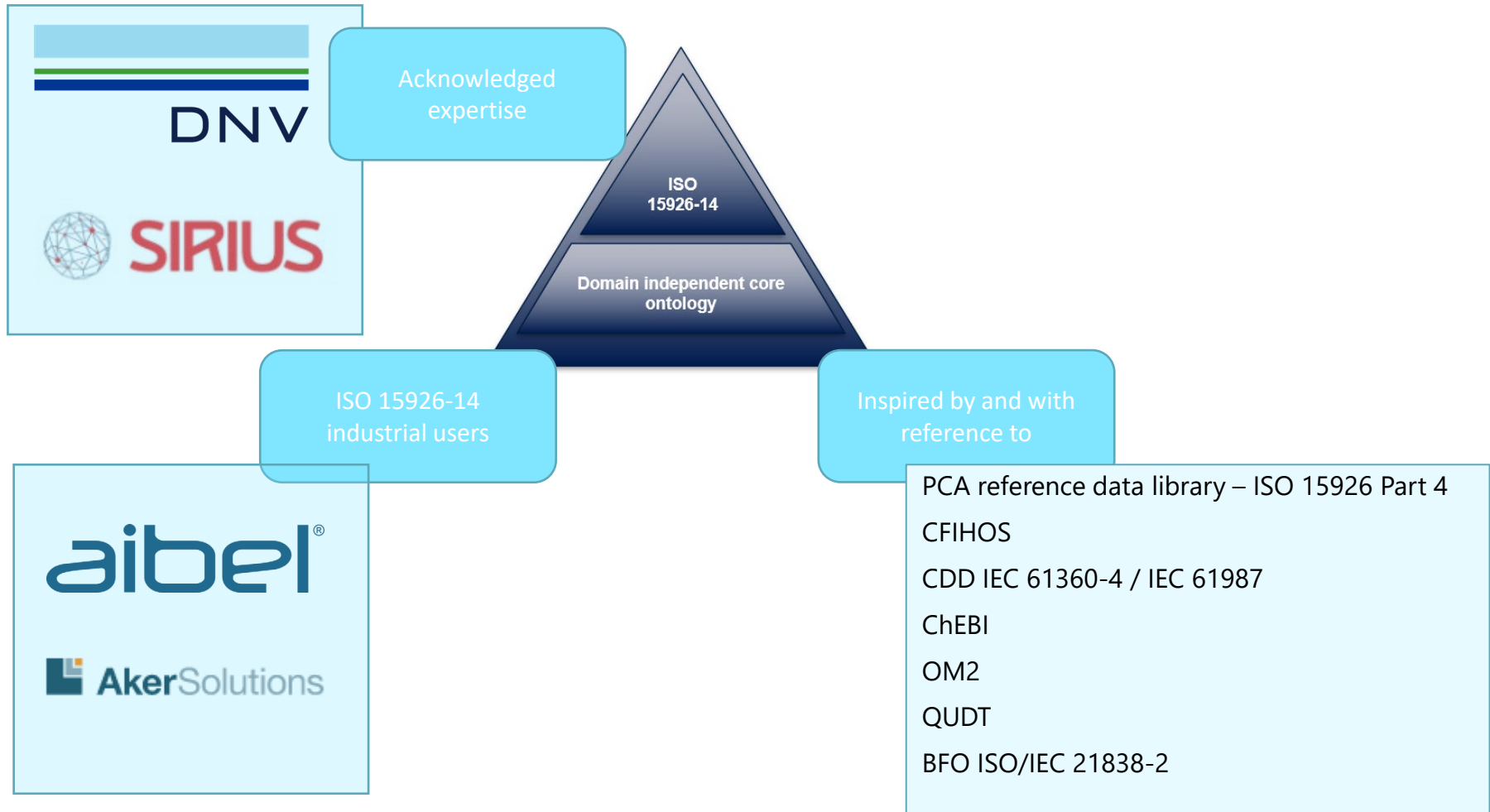
# Core reference data

- Vocabulary of reference data required for equipment specification
  - Types
  - Features
  - Attributes and values
- Vocabulary of reference data required for functional specification
- Vocabulary of reference data required for fluid properties
- Extension to include industry standard content

Beyond current scope

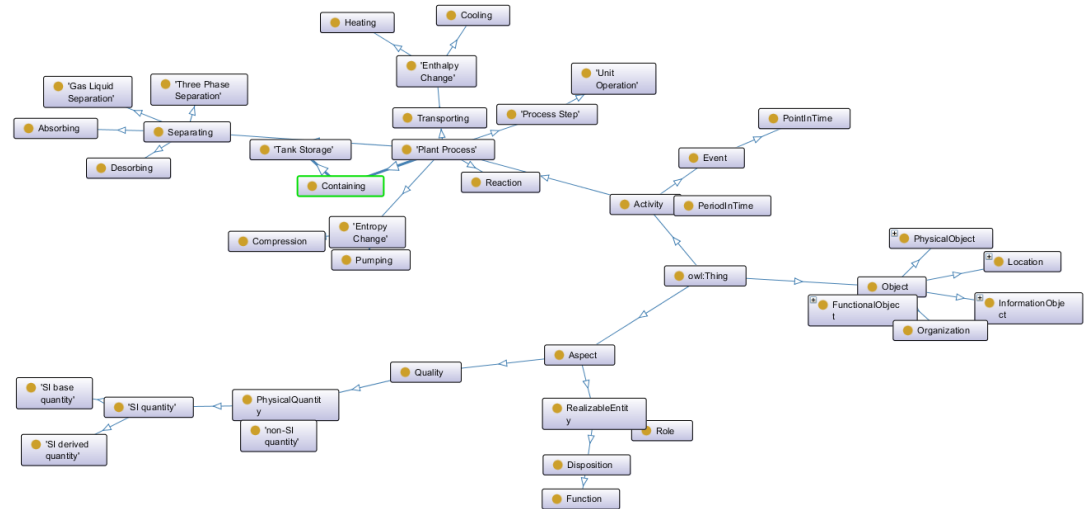


# Reference data are for building digital twins



# Contributions to RDL extension

- Extension of POSC Caesar service.
- <https://rds.posccaesar.org/ontology/>
- “Classes” with reference to a subset of “classes” from current PCA RDL
- SME identified object and relationship types NOAKA and READI JIP
- Issue handling GitLab
- Extend the community!



# Technical

- Available for both human and machine on Linked Data pages
- W3C OWL 2 Reference data
- Enable semantic reasoning
  - Machine executable design rules
  - Proper inheritance from reference data types

# Reasoning for reference data consistency

- Inconsistencies in reference data must be avoided
- Large and rich reference data are vulnerable for introduction of inconsistencies.
- Semantic reasoning, as part of reference data QA, will reveal such inconsistencies.

# Modular reference data

- For maintenance.
- To enable use for specific applications or services.
- Suitable when managing intellectual property right restrictions.
- Open Linked Data pages include all public reference data.

# READI



Bringing the  
oil and gas  
industry together



Share practises and requirements  
for improved cost efficiency  
and safety



Reducing complexity and risk  
for errors in work processes and  
information exchange



Enabling the automation  
of critical business  
processes and design

READI contact information:

Project manager: Erik Østby, DNV

E-mail: [Erik.Ostby@dnv.com](mailto:Erik.Ostby@dnv.com)

Mobile: +47 906 74 106

[www.readi-jip.org/](http://www.readi-jip.org/)

Steering Committee Chair: Paal Frode Larsen, Equinor

E-mail: [pfla@equinor.com](mailto:pfla@equinor.com)

Mobile: +47 9130570636