RotatingCoordinates_t Data Structure

The RotatingCoordinates_t data structure is used to record the rotation center and rotation rate vector of a rotating coordinate system.

It is proposed that the RotatingCoordinates_t data structure be recorded under a CGNSBase_t node or under a Zone_t node. There may be zero or one RotatingCoordinates_t under each type of parent.

SIDS definition of the RotatingCoordinates_t data structure:

The RotatingCoordinates_t under the CGNSBase_t and/or Zone_t data structures:

```
CGNSBase_t :=
  {
  RotatingCoordinates_t RotatingCoordinates ;              (o)
  Zone_t<CellDimension, PhysicalDimension> Zone1           (o)
    {
    RotatingCoordinates_t RotatingCoordinates ;            (o)
    ...
    }
  Zone_t<CellDimension, PhysicalDimension> Zone2           (o)
    {
    RotatingCoordinates_t RotatingCoordinates ;            (o)
    ...
    }
  ...
  }
```

The elements of the RotatingCoordinates_t data structure:

```
RotatingCoordinates_t :=
  {
  List( Descriptor_t Descriptor1 ... DescriptorN ) ;       (o)
  DataArray_t<real,1,PhysicalDimension> RotationCenter ;   (r)
  DataArray_t<real,1,PhysicalDimension> RotationRateVector; (r)
  DataClass_t DataClass ;                                  (o)
  DimensionalUnits_t DimensionalUnits ;                    (o)
  }
```

Definitions:

- RotationCenter: reference (X, Y, Z)-location of an origin for defining rotation rate.
- RotationRateVector: (X, Y, Z) components of the angular velocity of the grid about the RotationCenter.

Notes:

- Local DataClass_t and DimensionalUnits_t nodes may be specified under the RotatingCoordinates_t node (in case the user does not want to use the default units).

- All data use the current dimensional units unless different dimensional units are defined under the RotatingCoordinates_t node.

- If rotating coordinates are used, it is useful also to store variables relative to the rotating frame. New data-name identifiers are proposed to record the solution vectors (under FlowSolution_t) relative to the rotating coordinate systems (see table below). For documentation we define:

    - u = (u,v,w): inertial frame velocity vector
    - $\omega = (\omega_x,\omega_y,\omega_z)$: RotationRateVector
    - R: vector pointing from RotationCenter through the local point of interest (x,y,z).
    - $w_r = (w_{rx},w_{ry},w_{rz}) = \omega \times R$: rotational velocity of the rotating frame of reference.

Proposed CGNS Library extensions:

- A mid-level utility which, given rotating or inertial quantities, computes the other, based on the RotatingCoordinates_t information would be useful. Note that given RotatingCoordinates_t information, either inertial or rotating variables are sufficient to define the state of the flow. Thus, a user should not store both as that would lead to redundancy. However, defining the rotating quantities under the SIDS is desired to facilitate calculations done in the rotating frame, and interpreting rotating frame solutions.

- We could simply store the rotation information RotatingCoordinates_t, and define library functions that derive the rotating variables using this information. Then, only the original variables would be physically written to the CGNS file. The rotating variables would be derived on the fly if requested and if the RotatingCoordinates_t information is present. However, it seems that this would create a level of ambiguity, and would require the definition of a set of variables the mid-level libraries would be required to interpret, but are not defined for the CGNS file.

Diane, Chris, comments on the above choices? I prefer the first bullet myself, but the utilities would sure come in handy!!

Table of proposed data-name identifiers to record field vectors in the rotating frame:

Data-Name Identifier	Description	Units
RotatingVelocityX	x-component of velocity, relative to rotating frame $u_{rx} = u - w_{rx}$	L/T
RotatingVelocityY	y-component of velocity, relative to rotating frame $u_{ry} = u - w_{ry}$	L/T
RotatingVelocityZ	z-component of velocity, relative to rotating frame $u_{rz} = u - w_{rz}$	L/T
RotatingMomentumX	x-component of momentum, relative to rotating frame (ρu_{rx})	$M/(L^2T)$
RotatingMomentumY	y-component of momentum, relative to rotating frame (ρu_{ry})	$M/(L^2T)$
RotatingMomentumZ	z-component of momentum, relative to rotating frame (ρu_{rz})	$M/(L^2T)$
RotatingVelocityMagnitude	Velocity magnitude in rotating frame ($u_r = \text{sqrt}(u_{rx}^2 + u_{ry}^2 + u_{rz}^2)$	L/T
RotatingMach	Mach number relative to rotating frame	-
RotatingPressureStagnation	Stagnation pressure in rotating frame	$M/(LT^2)$
RotatingEnergyStagnation	Stagnation energy in rotating frame $E^* = e_0 + u_r^2/2 - w_r^2/2 = e_0 - u_r.w_r$	L^2/T^2
RotatingEnergyStagnationDensity	Stagnation energy per unit volume in rotating frame (ρE^*)	$M/(LT^2)$
RotatingEnthalpyStagnation	Stagnation enthalpy per unit mass in rotating frame (rothalpy) (I) $I = h_0 + u_r^2/2 - w_r^2/2 = h_0 - u_r.w_r$	L^2/T^2

What if we are rotating in another coordinate system? Should we add RotatingVelocityR, etc.?

ADF file mapping definition of the RotatingCoordinates_t data structure: