SHB refactoring report

I. Scope

At the time of writing, this report synthesis refactoring work related to the following TEs:

OPTION	SHB
FORC_NODA	te0484
RAPH_MECA	te0477
FULL_MECA	te0477
RIGI_MECA	te0473
RIGI_MECA_TANG	te0477
RIGI_MECA_GE	te0474

te0485 (SIEF_ELGA option) has been updated so that no SHB test case be broken, but it's refactoring is yet to achieve.

II. Objectives

Before this work, SHB fortran routines were linked to Code_Aster legacy fortran routines, but there was little to no synergy between them.
In addition, each SHB element was managed in separate routines, even if there was little difference between the computations to be carried out between SHB elements.
These are significant drawbacks regarding code maintenance.

The identified objectives of the refactoring work are as follow:

- whenever possible, replace SHB code already existing in legacy Code_Aster routines by appropriate call to Code_Aster legacy routines (sometime, legacy Code_Aster routines and their interface have been modified to ease these calls)
- merge all SHB code from separate routines that manage each element into a single workflow, and carry out specific computations for specific SHB (SHB8 or SHB6 most notably) within **if/endif** branches wherever required

III. Main modifications in the algorithm (impacting test case results)

a. SHB8 & SHB20 local frame

The local frame into which are computed mechanical quantities was originally evaluated in rloshb while a similar algorithm is already existing in dxqpgl (local frame for DxQ element).

Local element coordinate system is evaluated with the same logic in both routines, but base vectors are not constructed considering the same node ordering. As a consequence, resulting base vectors are colinear, but not oriented the same way.

Most notably:
(rloshb \ x axis) becomes (dxqpgl \ y axis)
(rloshb \ y axis) becomes (dxqpgl \ x axis)
(rloshb \ z axis) becomes (dxqpgl \ -z axis)
Most of non-regression results for SHB8 & SHB20 had to be adjusted in test cases. There is no noticeable evolutions and not always in the same direction: some have negligibly improved while other have negligibly worsened

⇒ Transformation matrix to SHB8 & SHB20 local frame is now evaluated with dxqpgl

b. SHB6 & SHB15 local frame

Similarly, while this local frame into which are computed mechanical quantities was originally evaluated in rlosh6, a similar algorithm is already existing in dxtpgl (local frame for DxT element).

It is however this time not possible to give direct equivalency between frame axis as for:

- rlosh6: base vectors are built considering middle of triangle edges
- while with dxtpgl, base vectors are built considering directly nodes of triangle element

⇒ Transformation matrix to SHB6 & SHB15 local frame is now evaluated with dxtpgl

c. Permutation of local axis for PENTA

This change is managed in ticket #0039.
To share the same definition of Hooke matrix, Z axis has been re-oriented along what corresponds to the element thickness. Thus

- x becomes z
- y becomes x
- z becomes y

d. Inversion of 5th & 6th components in mechanical quantities

<This refactoring is not complete yet in nonlinear options te0477, te0484 & te0474 and will not be till refactoring of te0485 is not completed>

3D isoparametric element formulation of Code_Aster is based on the following ordering of stress & strain components
XX, YY, ZZ, XY, **XZ, YZ**
The ordering of stress & strain components of original SHB code is as follow:
XX, YY, ZZ, XY, **YZ, XZ**

(..., while in shell-element-related literature, yet a third ordering is quite common: XX, YY, ZZ, **YZ, XZ, XY**)

Many matrix manipulations are dependent of this ordering. Given the goal to re-use as much as possible legacy Code_Aster routines, and most notably 3D isoparametric element-related

routines, wherever required, 5th & 6th components (for vectors) 5th & 6th columns & lines (for 6x6 matrix), and so on… have been re-ordered.

For routines related to te0477, te0484 & te0474, this re-ordering is enclosed within Glut \ Start, Glut \ End tags, waiting to be resorbed once te0485 will have been refactored.

⇒ A specific bitbucket ticket has been created for that:
https://bitbucket.org/code_aster/codeaster-src/issues/81/shb-complete-inversion-of-5th-6th-stress

e. Use of Young modulus & Poisson's coefficient in modified Hooke matrix

Some of the Hooke matrix components are modified (see SHB theoretical aspects in corresponding Code_Aster documentation) using E & nu material parameters.

In the original SHB code, E & nu parameters were retrieved only once, as those of the 1st Gauss point.

In the refactored code, E & nu are retrieved for each Gauss point.
That way, if one would consider a composite material with different E & nu parameters for different layers, and assuming one Gauss point per layer, the appropriate E & nu value would be used.
This has not been tested (in current test case, E & nu parameters have the same values for all Gauss point, so this algorithm evolution does not affect test case result).

To be noticed, number of through thickness Gauss point is constant for SHB elements, equal to 5.

f. Use of Cauchy stress at current iteration for evaluating tangent stiffness matrix

SHB routines have been re-written following closely formalism of 3D isoparametric element-related routines.

nmgrt3 routine is thus re-used from 3D isoparametric element to evaluate the tangent stiffness matrix of SHB elements (GROT-GDEP). This routine calls Cauchy stress from two different time:

- sign : Cauchy stress from previous converged increment
- sigma: Cauchy stress from current iteration

It uses the appropriate one given the option:

- RIGI_MECA_TANG : sign
- FULL_MECA: sigma

In original SHB code, sign was always used, whatever the option.
Using now nmgrt3 to evaluate tangent stiffness matrix for SHB element, this distinction is now made as well.

g. Expressing SHB6 discretized gradient operator directly in global coordinate system

Details of SHB6 theoretical background is given in appropriate Code_Aster documentation. Let's only recall that computation of its gradient operator requires gamma vector, and a scaling with the factor 0,45 to be applied in its local frame (transformation matrix evaluated by dxtpgl).

In original SHB code, SHB6 discretized gradient operator was thus evaluated in SHB6 local frame, and computation of stiffness matrix also carried out in this local frame.
Once evaluated, results were thus transformed back to global frame.

On the opposite, the other SHB elements, like 3D isoparametric elements, have these quantities (discretized gradient operator, stiffness matrix, …) directly evaluated in the global frame.

Given the goal to have as much as possible the same workflow, matrix transformations have been devised so as to obtain SHB6 gradient operator in the global frame, despite it having to be scaled with the factor 0,45 in its local frame.

I would say this the most "original contribution" of this refactoring work :) (and it took me time before having a working algorithm). This algorithm can be found in ss6bgl routine.

The formulation of SHB6 discretized gradient operator is obtained considering the following steps (copied from ss6bgl doxygen documentation):

Bglo being discretized gradient operator in global coordinates
Bloc being discretized gradient operator in local coordinates
Dloc being Hooke matrix in local coordinates
R & P the appropriate transformation matrices
w: Gauss point weight

!> Gauss point stiffness contribution in global coordinates is expressed as
!> w * (R^T . Bloc^T . Dloc . Bloc . R)
!> = w * (R^T . Bloc^T . (P . P^(-1))^T . Dloc . (P . P^(-1)). Bloc . R)
!> = w * (R^T . Bloc^T . (P^(-1))^T) . (P^T .Dloc. P) . (P^(-1) . Bloc . R)
!> = w * (Bglo^T . Dglo . Bglo)
!>
!> We have thus Bglo = P^(-1) . Bloc . R

IV. Other modifications to be noticed

a. Documentation

All significantly modified routines (or created ones) have been documented according doxygen formalism.

b. Gauss point coordinates, and computation of shape function derivatives

While in original SHB code, these data and computations were spreaded in different SHB routines, they are now managed with legacy Code_Aster routine dfdm3d.

c. Stiffness matrices

These quantities are now evaluated using legacy Code_Aster routines:
RIGI_MECA: btdbpr
GROT_GDEP: nmgrt3

d. Internal forces

These quantities are now evaluated using legacy Code_Aster code:
GROT_GDEP: nm3dfi, code that has been extracted from nmgrt3, & mutualized for both SHB & 3D isoparametric elements
te0484: btsig

e. Discretized gradient operator for SHB8, SHB15 & SHB20

This matrix is evaluated reusing here also Code_Aster legacy routine bmatmc.

f. Others

Other Code_Aster legacy routines have been reused:

- matrix/vector manipulation: utpvgl, utpvlg, utbtab, tpsivp…
- mechanical quantities: mpsoqo, pk2sig...
- some code has been mutualized with HEXS8 (3D_SI HEXA8) code in routines asvgam, asvedh...

V. List of modified SHB routines

a. Removed

rloshb (replaced with dxqpgl)
rlosh6 (replaced with dxtpgl)
shbpkc (replaced with pk2sig)
shvrot (replaced with utpvgl or utpvlg)
houxgb
sh1eps, sh2eps, sh8eps (replaced with nmgeom)
sh1for, sh2for, sh6for, sh8for (replaced with btsig or nm3dfi, depending where it is called)
sh1mek, sh2mek, sh8mek, sh6mek (replaced with nmgrt3)
sh1rig, sh2rig, sh8rig, sh6rig (replaced with btdbpr & nmgrt3)
shaksg
klocgl

b. To be removed once te0485 will have been refactored

shbksi, sh1ksi, sh2ksi, sh6ksi, shcalb, s1calb, s2calb, s6calb (replaced with dfdm3d)
mulmat (replaced with dgemm)
chrp3d, dr3gl1, dr3gl2 (replaced with tpsivp)
dsdx3d (replaced with nmgeom)
sh8sig, sh2sig, sh1sig, sh6sig

c. Created

mpsoqo (created with code originally from dpassa)
asvgam (created from HEXS8, SHB6 & SHB8 code)
asvedh (created from HEXS8 & SHB6 code)
hgfsca: HourGlass Force Stabilization vector from Combescure and Abed-Meraim
hgksca: HourGlass K Stabilization matrix from Combescure and Abed-Meraim
ss8hsm: Solid-Shell 8 Hourglass Stabilization Matrix
ss6bgl: Solid-Shell 6 B discretized gradient operator in GLobal coordinate system
hbbmp6: Hallquist B Bar Matrix for Prisme 6-node
tmassf: Transformation MAtrix to Solid-Shell Frame
nmssgr: Non-linear M Solid-Shell Grande Rotation
nmsstg: Non linear M Solid-Shell Tangent stiffness matrix
nm3dfi (created with code originally from nmgrt3)
nmsspl: Non-linear M Solid-Shell Petit (small) Linear deformation
nmssfi: Non linear M Solid-Shell Forces Internal

d. Renamed (& most often completely re-written)

shbrot (renamed ss8rco)
shbbar (renamed fbbbh8)
idsshb (renamed sshini)
sh6eps (renamed ss6eps)

VI. List of validated test cases

This work has been validated with the following list of test cases.
(green: RIGI_MECA / orange: STAT_NON_LINE)

a. SHB test cases

TEST CASE	SHB8		SHB6	SHB15	SHB20
ID					
SDLS109	G				
SDLV120	C				
SSLS101	C		K	L	D
SSLS105	C				
SSLS108	C, D		E, F	H	G
SSLS123	A		C, D		
SSLS124	A, B	A	C	F, G	D, E
SSLS125	A		B	D	C
SSLS129	C				
SSNS101	A, B, C, D, H		F	G	E
SSNS102	A				B
SSNS109	A		B	D	C

b. 3D_ISO test cases - non PENTA

TEST CASE	HEXS8	HEXS20	TETRA4	TETRA10
ID				
SSNV187			C	
SSNV189		A		
SSNV219				A
SSNV227		A		
HSNV125	G			
SDNV103	J			
SSLV200		B		
SSND105	B			
SSNP123	C			
SSNV196	A	C		

c. 3D_ISO test cases - PENTA

TEST CASE	PENTA6	PENTA15	PENTA18
ID			
SDNV103	**J**	**G, H, I, J**	
SSLV04	**A, K**	**B, N**	
SSLV200	**A, B**	**A, B**	
SSNP104	**B**		
SSNP121	**J, K**	**E, F, J**	**R**
SSNP14	**B**	**C**	