ČERVENKA CONSULTING

Červenka Consulting s.r.o.
Na Hrebenkach 55
150 00 Prague
Czech Republic
Phone: +420 220 610 018
E-mail: cervenka@cervenka.cz
Web: http://www.cervenka.cz

ATENA Program Documentation
Part 10

User Material DLL

Implementing a User Material Model

Written by

Dobromil Pryl

Prague, March 29, 2016

TABLE OF CONTENTS

IMPLEMENTING USER MATERIAL IN ATENA - REFERENCE AND EXAMPLE DOCUMENTATION

1. USER MATERIAL INTRODUCTION

1.1. Introduction

This is a description for users who wish to implement their own material model into ATENA version 5 when the material behavior can not be described by the standard materials or by user laws in the fracture-plastic material (CC3DNonLinCementitious2User). It is assumed the user already has experience using ATENA, including basic understanding of the input file (.inp), some background knowledge about the model to be implemented and material models and finite element method (FEM) in general, and that he or she is able to develop in C or some other programming language that can produce dynamic link libraries, e.g., FORTRAN or Pascal.

Basically, the user has to prepare a dynamic link library (DLL) which exports several functions that will be called by the ATENA kernel. Probably the easiest way is to start with a copy of the example project included with ATENA (CCUserMaterialExampleDLL), which implements an incremental elastic material with von Mises stress as an additional state variable for available for postprocessing.

1.2. Operation

Before starting the implementation, one should know how the new user material library will be used by the kernel, or, in other words, understand when and what for which of the functions to be implemented are needed.

1.2.1. Material Definition

Let us first have a look what happens when ATENA reads an input file where `CC3DUserMaterial` is defined, e.g.

```
MATERIAL ID 1 NAME "3D User"
TYPE "CC3DUserMaterial"
UserMaterialDLL "CCUserMaterialExampleDLL.dll"
E 3.032000e+004
ALPHA 1.200000e-005
RHO 2.300000e-002
MU 2.000000e-001
YieldStress 20.0
;
```

The first line just defines a new material and its ID (internal number) and name. The second line is a bit more interesting and tells the kernel what kind of material to create. In this case, the type `CC3DUserMaterial` means that it is going to be a user defined material. When the

`UserMaterialDLL "CCUserMaterialExampleDLL.dll"`

line is read, the user library is loaded (or an error reported if it can not be found or loaded). Then, all the interface functions are loaded (or, again, an error is reported if some of them is missing or does not conform with respect to the expected parameters and return type). Afterwards, a few of the user functions are called to get several values needed to create the material in memory:

- the number of user material parameters are asked for by calling the function UserDLLMaterialParamsCount()
- the number of user state variables from a call to UserDLLStateVarsCount()

- all the user material parameter names are requested from UserDLLMaterialParamName() and stored such that they can be recognized when processing the following lines of the input.
-

Then, the remaining material parameter values are read and stored. Please understand that the `UserMaterialDLL` parameter has to be the first one simply because the others, except for those inherited from the ATENA elastic material (i.e., `E` , `mu` , `ALPHA` , and `RHO`) can not be known to the kernel at all before the user DLL is loaded.

Note:

Please note the difference between *user material parameters* and *user state variables* . While the former are properties of the *material* and do not change during the analysis, the latter are defined separately in each *material point* and their main purpose is to reflect the (local) changes in the material. For both of them, only floating point values are allowed in the current implementation.

1.2.2. Material Point Initialization

Before the analysis starts, quantities that should be available for postprocessing have to be registered. Stresses and strains are inherited from the elastic material, and therefore available automatically. The names of the additional user state variables are obtained by a call to UserDLLStateVarName(). Furthemore, all the integration points of all finite elements are initialized (e.g., all strains and stresses to zero). In case of the user material, UserDLLResetState() is called for each material point such that the user state variables can be initialized to any initial values as needed.

1.2.3. Calculation

In nonlinear analysis, the applied load is divided into load steps and in each step, a nonlinear system of equations is solved by (linear) iterations (see the description of the *Newton-Raphson* and *Arclength* methods in *Atena Theory Manual* for details). For each iteration, a linear approximation system is built (when using the *Full Newton-Raphson* method, there are small differences in other cases) and solved. The global stiffness matrix of the linear system is built from local tangent stiffness matrices. In case of user material, UserDLLTangentStiff() is called for each material point to get the local matrices.

The material response to a deformation increment is asked (possibly multiple times) in each iteration for each material point. For the user material, this has to be implemented in UserDLLCalculateResponse(), which is obviously the most interesting and most important function of the user material library.

During the nonlinear analysis, the coordinate system changes due to large deformations have to be taken into account. If anything else than the default transformation of strains and stresses based on the deformation gradients is needed, i.e., additional variables have to be transformed, or the stresses or strains have to be transformed in some different way, this can be done in UserDLLTransformState().

1.3. Implementation

All the above mentioned functions are implemented and commented in CCUserMaterialExampleDLL.c (C and all languages except FORTRAN) and CCUserMaterialFORTRANExampleDLL.F90 (FORTRAN). There, you can find the description of all functions, arguments, and return values. Please note FORTRAN needs special handling due to specific argument and return-value passing. Therefore, if you create your User Material Dynamic Link Library in FORTRAN, you have to include the string "FORTRAN" in the DLL file name (and, obviously, the file names of DLLs created in other languages must NOT contain this string).

Please do not forget the functions need to be accessible using undecorated names (i.e., without the mangling reflecting the parameters and return value). If you use another language than C (which used in our example), you may need additional keywords or settings to get the function names undecorated in your DLL, e.g., extern "C" in case of compiling in C++ (.cpp extension instead of C).

This documentation is based on the example DLL included in ATENA installation. Therefore, the global arrays with the names of material parameters and state variables and the corresponding #defines used for their dimensions and for some auxiliary values, are also included. These are not compulsory, they are just a suggestion how the user can implement the functions that return the parameters' and variables' count and names and handle some floating point calculation aspects in an easy and consistent way.

See also the example input file `UserMaterial.inp` demonstrating the use of a user defined material (included in ATENA installation in the `"Examples\ATENA Science\AtenaWin\CCUserMaterialExampleDLL"` subdirectory of the ATENA installation directory, typically, `"c:\Program Files\CervenkaConsulting\AtenaV5"` , and the ATENA Theory and ATENA Input File Format manuals.

2. CLASS INDEX

2.1. Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

3. FILE INDEX

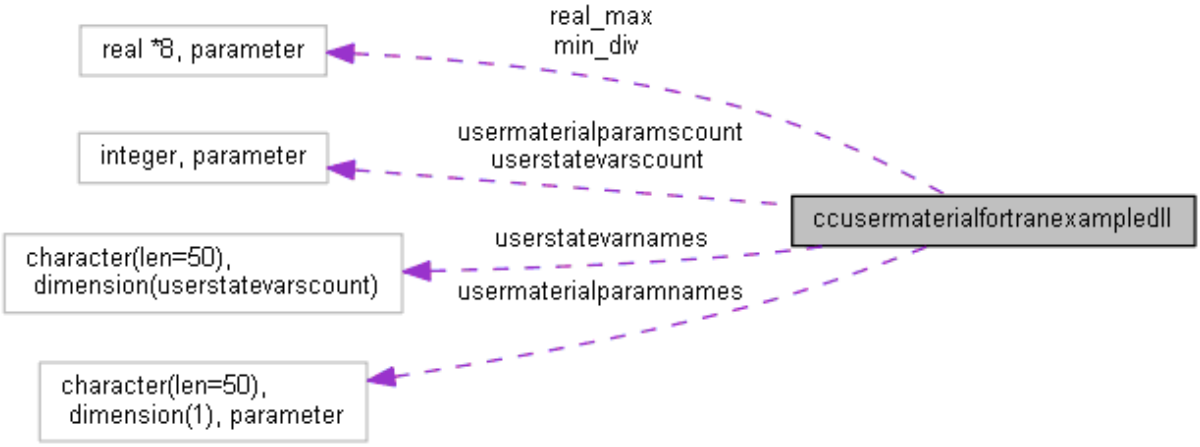
3.1. File List

Here is a list of all files with brief descriptions:

4. CLASS DOCUMENTATION

4.1. ccusermaterialfortranexampledll Module Reference

Collaboration diagram for ccusermaterialfortranexampledll:



4.1.1. Public Member Functions

- integer function userdllcalculateresponse (deps, teps, sigma, E, mu, UserMaterialParams, UserMaterialState)
  Purpose: calculates the stress response of a material point to given strain.
- integer function userdllresetstate (E, mu, UserMaterialParams, UserMaterialState)
  Purpose: resets (initializes) the material point state.
- integer function userdlltangentstiff (TangentMatrix, E, mu, UserMaterialParams, UserMaterialState)
  Purpose: compute the local tangential stiffness matrix.
- integer function userdllsecantstiff (SecantMatrix, E, mu, UserMaterialParams, UserMaterialState)
- integer function userdlltransformstate (DefGradient, eps, sigma, E, mu, UserMaterialParams, UserMaterialState)
  Purpose: transformation of coordinate system due to large deformations.
- integer function userdllmaterialparamscount ()
  Purpose: The number of user defined material parameters.
- integer function userdllstatevarscount ()
  Purpose: The number of user defined material state variables.
- type(c_ptr) function userdllmaterialparamname (MaterialParamNo)
  Purpose: The name of a user defined material parameter.
- type(c_ptr) function userdllstatevarname (StateVarNo)
  Purpose: The name of a user defined material state variable.

4.1.2. Public Attributes

- integer, parameter usermaterialparamscount = 1
  Number of user material parameters = floating point values read from the user material definition in the input file. This value is returned by UserDLLMaterialParamsCount()
- character(len=50), dimension(1),
- parameter usermaterialparamnames = (/"YieldStress"//CHAR(0)/)
  All user material parameter names in an array, which is used in UserDLLMaterialParamName(). All strings are zero terminated!
- integer, parameter userstatevarscount = 2

Number of user state variables = floating point values stored in each material point. This value is also returned by UserDLLStateVarsCount()

- character(len=50), dimension(userstatevarscount) userstatevarnames =(/"vonMisesStress"//CHAR(0), "Yielding"//CHAR(0)/)
  All user state variable names in an array, which is used in UserDLLStateVarName(). All strings are zero terminated!
- real *8, parameter real_max = HUGE(0.0d0)
  Largest real number (value from the floating point library)
- real *8, parameter min_div = TINY(0.0d0)*1000
  Minimum accepted divisor used to prevent division by zero/overflow (based on a value from the floating point library)

4.1.3. Detailed Description

Definition at line 3 of file CCUserMaterialFORTRANExampleDLL.F90.

4.1.4. Member Function/Subroutine Documentation

4.1.4.1. integer function ccusermaterialfortranexampledll::userdllcalculateresponse (real*8, dimension(6) *deps*, real*8, dimension(6) *teps*, real*8, dimension(6), intent(inout) *sigma*, real*8 *E*, real*8 *mu*, real*8, dimension(usermaterialparamscount) *UserMaterialParams*, real*8, dimension(userstatevarscount), intent(inout) *UserMaterialState*)

Purpose: calculates the stress response of a material point to given strain.

This is the key function the user has to define for a new user defined material. It is called when evaluating the material response ("material iterations"). E.g., if a force response is prescribed (in all or 1 direction), this routine will be called repeatedly with changing deps until the difference (error) is acceptable.

Return values:

0	OK
nonzero	error

Parameters:

	deps	strain increment tensor stored as a vector, first the diagonal terms, followed by the off-diagonal
	teps	total strain tensor (stored as a vector)
in,out	*sigma*	[in+out] stress tensor stored as a vector (as for deps)
	e	elastic modulus
	mu	Poisson's ratio
	usermaterialparams	vector of user material parameter values
in,out	*usermaterialstate*	[in+out] vector of user material state variables in the material point being calculated

Definition at line 109 of file CCUserMaterialFORTRANExampleDLL.F90.

References userdlltangentstiff().

Here is the call graph for this function:

4.1.4.2. type(c_ptr) function ccusermaterialfortranexampledll::userdllmaterialparamname (integer, intent(in) *MaterialParamNo*)

Purpose: The name of a user defined material parameter.

The user has to define this function to let the ATENA kernel know the names of the additional parameters the material has, which is required among others when reading the material definition with the parameter values from an input file.

Parameters:

MaterialParamNo	parameter number (id) 1..UserMaterialParamsCount

Return values:

(zero	terminated) string name of the MaterialParamNo-th user material parameter
NULL	invalid parameter number (out of range)

Parameters:

in	*materialparamno*	parameter number

Definition at line 372 of file CCUserMaterialFORTRANExampleDLL.F90.

References usermaterialparamnames, and usermaterialparamscount.

4.1.4.3. integer function ccusermaterialfortranexampledll::userdllmaterialparamscount ()

Purpose: The number of user defined material parameters.

The user has to define this function to let the ATENA kernel know how many additional parameters the material has, which is required among others when reading the material definition with the parameter values from an input file.

Returns:

the number of additional user material parameters

Definition at line 331 of file CCUserMaterialFORTRANExampleDLL.F90.

References usermaterialparamscount.

4.1.4.4. integer function ccusermaterialfortranexampledll::userdllresetstate (real*8 *E*, real*8 *mu*, real*8, dimension(usermaterialparamscount) *UserMaterialParams*, real*8, dimension(userstatevarscount), intent(out) *UserMaterialState*)

Purpose: resets (initializes) the material point state.

Return values:

0	OK
nonzero	error

Parameters:

	e	Young modulus
	mu	Poisson's ratio
	usermaterialparams	user material parameters array

out	*usermaterialstate*	user state variables array

Definition at line 170 of file CCUserMaterialFORTRANExampleDLL.F90.

References userstatevarscount.

4.1.4.5. integer function ccusermaterialfortranexampledll::userdllsecantstiff (real*8, dimension(6*6), intent(out) *SecantMatrix*, real*8 *E*, real*8 *mu*, real*8, dimension(usermaterialparamscount) *UserMaterialParams*, real*8, dimension(userstatevarscount) *UserMaterialState*)

Parameters:

out	*secantmatrix*	the local secant stiffness matrix as a vector
	e	Young modulus
	mu	Poisson's ratio
	usermaterialparams	user material parameters array
	usermaterialstate	user state variables array

Definition at line 273 of file CCUserMaterialFORTRANExampleDLL.F90.

4.1.4.6. type(c_ptr) function ccusermaterialfortranexampledll::userdllstatevarname (integer *StateVarNo*)

Purpose: The name of a user defined material state variable.

The user has to define this function to let the ATENA kernel know the names of the additional state variables the material has, which is required among others when offering the list of quantities available for postprocessing.

Parameters:

StateVarNo	parameter number (id) 1..UserStateVarsCount

Return values:

(zero	terminated) string name of the StateVarNo-th user material state variable
NULL	invalid parameter number (out of range)

Parameters:

statevarno	variable number

Definition at line 399 of file CCUserMaterialFORTRANExampleDLL.F90.

References userstatevarnames, and userstatevarscount.

4.1.4.7. integer function ccusermaterialfortranexampledll::userdllstatevarscount ()

Purpose: The number of user defined material state variables.

The user has to define this function to let the ATENA kernel know how many additional state variables the material has in each material point, which is required among others when offering the list of quantities available for postprocessing.

Returns:

the number of additional user material state variables

Definition at line 349 of file CCUserMaterialFORTRANExampleDLL.F90.

References userstatevarscount.

4.1.4.8. integer function ccusermaterialfortranexampledll::userdlltangentstiff (real*8, dimension(6*6), intent(out) *TangentMatrix*, real*8 *E*, real*8 *mu*, real*8, dimension(usermaterialparamscount) *UserMaterialParams*, real*8, dimension(userstatevarscount) *UserMaterialState*)

Purpose: compute the local tangential stiffness matrix.

Return values:

0	OK
nonzero	error

Parameters:

out	*tangentmatrix*	the local tangential stiffness matrix as a vector
	e	Young modulus
	mu	Poisson's ratio
	usermaterialparams	user material parameters array
	usermaterialstate	user state variables array

Definition at line 194 of file CCUserMaterialFORTRANExampleDLL.F90.

References min_div, and real_max.

Referenced by userdllcalculateresponse().

4.1.4.9. integer function ccusermaterialfortranexampledll::userdlltransformstate (real*8, dimension(6*6) *DefGradient*, real*8, dimension(6), intent(inout) *eps*, real*8, dimension(6), intent(inout) *sigma*, real*8 *E*, real*8 *mu*, real*8, dimension(usermaterialparamscount) *UserMaterialParams*, real*8, dimension(userstatevarscount) *UserMaterialState*)

Purpose: transformation of coordinate system due to large deformations.

Return values:

0	OK, the def. gradient matrix should be used - nothing done here, and this function needs not to be called again (to reduce overhead/CPU time)
1	OK, the def. gradient matrix should be used - nothing done here, but this function should be called next time
2	OK, user's own transformation used for user material state vars, the def. matrix should be used to transform strains+stresses
3	OK, user's own transformation used for both user material state vars and strains+stresses
other	error

Parameters:

	defgradient	the deformation gradient matrix used for transformation of the elastic stresses and strains
in,out	*eps*	[in+out] total strain tensor (stored as a vector)
in,out	*sigma*	[in+out] stress tensor stored as a vector (as for deps)
	e	Young modulus
	mu	Poisson's ratio
	usermaterialparams	user material parameters array
	usermaterialstate	user state variables array

Definition at line 304 of file CCUserMaterialFORTRANExampleDLL.F90.

4.1.5. Member Data Documentation

4.1.5.1. real*8, parameter ccusermaterialfortranexampledll::min_div = TINY(0.0d0)*1000

Minimum accepted divisor used to prevent division by zero/overflow (based on a value from the floating point library)

Definition at line 78 of file CCUserMaterialFORTRANExampleDLL.F90.

Referenced by userdlltangentstiff().

4.1.5.2. real*8, parameter ccusermaterialfortranexampledll::real_max = HUGE(0.0d0)

Largest real number (value from the floating point library)

Definition at line 74 of file CCUserMaterialFORTRANExampleDLL.F90.

Referenced by userdlltangentstiff().

4.1.5.3. character (len=50), dimension(1), parameter ccusermaterialfortranexampledll::usermaterialparamnames = (/"YieldStress"//CHAR(0)/)

All user material parameter names in an array, which is used in UserDLLMaterialParamName(). All strings are zero terminated!

Definition at line 59 of file CCUserMaterialFORTRANExampleDLL.F90.

Referenced by userdllmaterialparamname().

4.1.5.4. integer, parameter ccusermaterialfortranexampledll::usermaterialparamscount = 1

Number of user material parameters = floating point values read from the user material definition in the input file. This value is returned by UserDLLMaterialParamsCount()

Definition at line 54 of file CCUserMaterialFORTRANExampleDLL.F90.

Referenced by userdllmaterialparamname(), and userdllmaterialparamscount().

4.1.5.5. character (len=50), dimension(userstatevarscount) ccusermaterialfortranexampledll::userstatevarnames =(/"vonMisesStress"//CHAR(0), "Yielding"//CHAR(0)/)

All user state variable names in an array, which is used in UserDLLStateVarName(). All strings are zero terminated!

Definition at line 70 of file CCUserMaterialFORTRANExampleDLL.F90.

Referenced by userdllstatevarname().

4.1.5.6. integer, parameter ccusermaterialfortranexampledll::userstatevarscount = 2

Number of user state variables = floating point values stored in each material point. This value is also returned by UserDLLStateVarsCount()

Definition at line 65 of file CCUserMaterialFORTRANExampleDLL.F90.

Referenced by userdllresetstate(), userdllstatevarname(), and userdllstatevarscount().

4.1.5.7. The documentation for this module was generated from the following file:

- CCUserMaterialFORTRANExampleDLL/CCUserMaterialFORTRANExampleDLL.F90

5. FILE DOCUMENTATION

5.1. CCUserMaterialExampleDLL/CCUserMaterialDLL.h File Reference

Purpose: Header file containing the definition of the user defined functions for user material. Do not change.

5.1.1. Typedefs

- Functions defined in the UserMaterialDLLtypedef UINT(CDECL * LPUserDLLCalculateResponse)(const double deps[], const double teps[], double sigma[], double E, double mu, const double UserMaterialParams[], double UserMaterialState[])
- typedef UINT(CDECL * LPUserDLLResetState)(double E, double mu, const double UserMaterialParams[], double UserMaterialState[])
- typedef UINT(CDECL * LPUserDLLTangentStiff)(double TangentMatrix[], double E, double mu, const double UserMaterialParams[], const double UserMaterialState[])
- typedef UINT(CDECL * LPUserDLLElasticStiff)(double ElasticMatrix[], double E, double mu, const double UserMaterialParams[])
- typedef UINT(CDECL * LPUserDLLSecantStiff)(double SecantMatrix[], double E, double mu, const double UserMaterialParams[], const double UserMaterialState[])
- typedef UINT(CDECL * LPUserDLLTransformState)(double DefGradient[], double eps[], double sigma[], double E, double mu, const double UserMaterialParams[], double UserMaterialState[])
- typedef UINT(CDECL * LPUserDLLMaterialParamsCount)()
- typedef UINT(CDECL * LPUserDLLStateVarsCount)()
- typedef LPSTR(CDECL * LPUserDLLMaterialParamName)(const UINT MaterialParamNo)
- typedef LPSTR(CDECL * LPUserDLLStateVarName)(const UINT StateVarNo)
- FORTRAN variants of the functions where workarounds are needed due to argument incompatibilitytypedef UINT(CDECL * LPF90UserDLLCalculateResponse)(const double deps[], const double teps[], double sigma[], double *E, double *mu, const double UserMaterialParams[], double UserMaterialState[])
- typedef UINT(CDECL * LPF90UserDLLResetState)(double *E, double *mu, const double UserMaterialParams[], double UserMaterialState[])
- typedef UINT(CDECL * LPF90UserDLLTangentStiff)(double TangentMatrix[], double *E, double *mu, const double UserMaterialParams[], const double UserMaterialState[])
- typedef UINT(CDECL * LPF90UserDLLElasticStiff)(double ElasticMatrix[], double *E, double *mu, const double UserMaterialParams[])
- typedef UINT(CDECL * LPF90UserDLLSecantStiff)(double SecantMatrix[], double *E, double *mu, const double UserMaterialParams[], const double UserMaterialState[])
- typedef UINT(CDECL * LPF90UserDLLTransformState)(double DefGradient[], double eps[], double sigma[], double *E, double *mu, const double UserMaterialParams[], double UserMaterialState[])
- typedef UINT(CDECL * LPF90UserDLLMaterialParamName)(LPSTR retname, UINT *strlen, const UINT *const MaterialParamNo)
- typedef UINT(CDECL * LPF90UserDLLStateVarName)(LPSTR retname, UINT *strlen, const UINT *const StateVarNo)

5.1.2. Detailed Description

Purpose: Header file containing the definition of the user defined functions for user material. Do not change.

Author: Dobromil Pryl

Please include this file in your DLL and do NOT change it. It defines the interface between ATENA and your DLL and it needs to remain unchanged - we also use it when compiling ATENA.

If you change it, we would need to make the same changes on the other side = in ATENA to make it work, which that would break our examples and the DLLs of all other users!

Revision history:

1. 2008 - the file was created

Definition in file CCUserMaterialDLL.h.

5.1.3. Typedef Documentation

5.1.3.1. typedef UINT(CDECL* LPF90UserDLLCalculateResponse)(const double deps[], const double teps[], double sigma[], double *E, double *mu, const double UserMaterialParams[], double UserMaterialState[])

Definition at line 58 of file CCUserMaterialDLL.h.

5.1.3.2. typedef UINT(CDECL* LPF90UserDLLElasticStiff)(double ElasticMatrix[], double *E, double *mu, const double UserMaterialParams[])

Definition at line 67 of file CCUserMaterialDLL.h.

5.1.3.3. typedef UINT(CDECL* LPF90UserDLLMaterialParamName)(LPSTR retname, UINT *strlen, const UINT *const MaterialParamNo)

Definition at line 81 of file CCUserMaterialDLL.h.

5.1.3.4. typedef UINT(CDECL* LPF90UserDLLResetState)(double *E, double *mu, const double UserMaterialParams[], double UserMaterialState[])

Definition at line 61 of file CCUserMaterialDLL.h.

5.1.3.5. typedef UINT(CDECL* LPF90UserDLLSecantStiff)(double SecantMatrix[], double *E, double *mu, const double UserMaterialParams[], const double UserMaterialState[])

Definition at line 70 of file CCUserMaterialDLL.h.

5.1.3.6. typedef UINT(CDECL* LPF90UserDLLStateVarName)(LPSTR retname, UINT *strlen, const UINT *const StateVarNo)

Definition at line 82 of file CCUserMaterialDLL.h.

5.1.3.7. typedef UINT(CDECL* LPF90UserDLLTangentStiff)(double TangentMatrix[], double *E, double *mu, const double UserMaterialParams[], const double UserMaterialState[])

Definition at line 63 of file CCUserMaterialDLL.h.

5.1.3.8. typedef UINT(CDECL* LPF90UserDLLTransformState)(double DefGradient[], double eps[], double sigma[], double *E, double *mu, const double UserMaterialParams[], double UserMaterialState[])

Definition at line 74 of file CCUserMaterialDLL.h.

5.1.3.9. typedef UINT(CDECL* LPUserDLLCalculateResponse)(const double deps[], const double teps[], double sigma[], double E, double mu, const double UserMaterialParams[], double UserMaterialState[])

Definition at line 27 of file CCUserMaterialDLL.h.

5.1.3.10. typedef UINT(CDECL* LPUserDLLElasticStiff)(double ElasticMatrix[], double E, double mu, const double UserMaterialParams[])

Definition at line 36 of file CCUserMaterialDLL.h.

5.1.3.11. typedef LPSTR(CDECL* LPUserDLLMaterialParamName)(const UINT MaterialParamNo)

Definition at line 50 of file CCUserMaterialDLL.h.

5.1.3.12. typedef UINT(CDECL* LPUserDLLMaterialParamsCount)()

Definition at line 48 of file CCUserMaterialDLL.h.

5.1.3.13. typedef UINT(CDECL* LPUserDLLResetState)(double E, double mu, const double UserMaterialParams[], double UserMaterialState[])

Definition at line 30 of file CCUserMaterialDLL.h.

5.1.3.14. typedef UINT(CDECL* LPUserDLLSecantStiff)(double SecantMatrix[], double E, double mu, const double UserMaterialParams[], const double UserMaterialState[])

Definition at line 39 of file CCUserMaterialDLL.h.

5.1.3.15. typedef LPSTR(CDECL* LPUserDLLStateVarName)(const UINT StateVarNo)

Definition at line 51 of file CCUserMaterialDLL.h.

5.1.3.16. typedef UINT(CDECL* LPUserDLLStateVarsCount)()

Definition at line 49 of file CCUserMaterialDLL.h.

5.1.3.17. typedef UINT(CDECL* LPUserDLLTangentStiff)(double TangentMatrix[], double E, double mu, const double UserMaterialParams[], const double UserMaterialState[])

Definition at line 32 of file CCUserMaterialDLL.h.

5.1.3.18. typedef UINT(CDECL* LPUserDLLTransformState)(double DefGradient[], double eps[], double sigma[], double E, double mu, const double UserMaterialParams[], double UserMaterialState[])

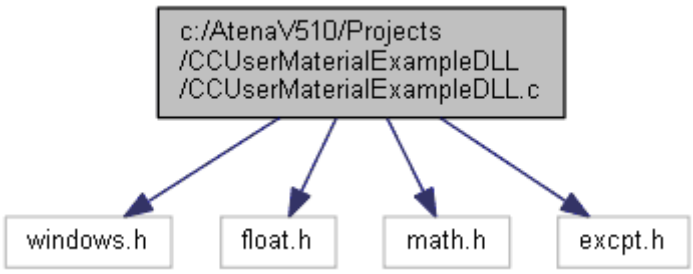
Definition at line 43 of file CCUserMaterialDLL.h.

5.2. CCUserMaterialExampleDLL/CCUserMaterialExampleDLL.c File Reference

Purpose: File containing an example user material definition - elastic material with von Mises effective stress made available for postprocessing. For users implementing their own user materials it is recommended to use this file or the whole project as a base for their own development, i.e., to copy and modify this.

```
#include <windows.h>
#include <float.h>
#include <math.h>
#include <excpt.h>
```

Include dependency graph for CCUserMaterialExampleDLL.c:



5.2.1. Macros

- #define REAL_MAX DBL_MAX
- #define MIN_DIV DBL_EPSILON*1000

5.2.2. Functions

User defined functions of UserMaterialDLL

See also:

class CCUserMaterial

- UINT CDECL UserDLLTangentStiff (double TangentMatrix[6 *6],double E,double mu,double UserMaterialParams[],double UserMaterialState[])
  Purpose: compute the local tangential stiffness matrix.
- UINT CDECL UserDLLCalculateResponse (double deps[], double teps[], double sigma[], double E, double mu, double UserMaterialParams[], double UserMaterialState[])
  Purpose: calculates the stress response of a material point to given strain.

- UINT CDECL UserDLLResetState (double E, double mu, double UserMaterialParams[], double UserMaterialState[])
  Purpose: resets (initializes) the material point state.
- UINT CDECL UserDLLElasticStiff (double ElasticMatrix[6 *6], double E, double mu, double UserMaterialParams[])
  Purpose: compute the local elastic stiffness matrix. This is usefull for anizotropic materials. If the function is not defined, the default elastic matrix (based on E and mu) is used.
- UINT CDECL UserDLLSecantStiff (double SecantMatrix[], double E, double mu, double UserMaterialParams[], double UserMaterialState[])
  Purpose: compute the local secant stiffness matrix.
- UINT CDECL UserDLLTransformState (double DefGradient[], double eps[], double sigma[], double E, double mu, double UserMaterialParams[], double UserMaterialState[])
  Purpose: transformation of coordinate system due to large deformations.
- UINT CDECL UserDLLMaterialParamsCount ()
  Purpose: The number of user defined material parameters.
- UINT CDECL UserDLLStateVarsCount ()
  Purpose: The number of user defined material state variables.
- LPSTR CDECL UserDLLMaterialParamName (UINT MaterialParamNo)
  Purpose: The name of a user defined material parameter.
- LPSTR CDECL UserDLLStateVarName (UINT StateVarNo)
  Purpose: The name of a user defined material state variable.

5.2.3. Global variables of UserMaterialDLL

- #define UserMaterialParamsCount 1
- #define UserStateVarsCount 2
- static char * UserMaterialParamNames [UserMaterialParamsCount]
- static char * UserStateVarNames [UserStateVarsCount]

5.2.4. Detailed Description

Purpose: File containing an example user material definition - elastic material with von Mises effective stress made available for postprocessing. For users implementing their own user materials it is recommended to use this file or the whole project as a base for their own development, i.e., to copy and modify this.

The project is created in Microsoft Visual Studio 2012, however, it should be a good guideline for implementing an ATENA user material in any C/C++ compiler. Moreover, any language/compiler which can produce a DLL implementing the interface functions, e.g., FORTRAN or Pascal, can be used. Please note FORTRAN needs special handling and see CCUserMaterialFORTRANExampleDLL for a FORTRAN example and functions' description. For an overview of operation, see the User Material Introduction.

The sources of the example DLL and sample ATENA input files (.inp) are included in ATENA installation. You can find them in the `"Examples\ATENA Science\AtenaWin\CCUserMaterialExampleDLL"` subdirectory of the ATENA installation directory (typically, `"c:\Program Files\CervenkaConsulting\AtenaV5"`).

Author: Dobromil Pryl

Revision history:

```
1. 2008   - the file was created
```

See also:

class CCUserMaterial (ATENA internal class wrapping the user material DLL)

Definition in file CCUserMaterialExampleDLL.c.

5.2.5. Macro Definition Documentation

5.2.5.1. #define MIN_DIV DBL_EPSILON*1000

Minimum accepted divisor used to prevent division by zero/overflow (based on a value from the floating point library)

Not a compulsory part of the interface - the user may implement in another way.

Definition at line 269 of file CCUserMaterialExampleDLL.c.

Referenced by UserDLLTangentStiff().

5.2.5.2. #define REAL_MAX DBL_MAX

Largest real number (value from the floating point library)

Not a compulsory part of the interface - the user may implement in another way.

Definition at line 262 of file CCUserMaterialExampleDLL.c.

Referenced by UserDLLTangentStiff().

5.2.5.3. #define UserMaterialParamsCount 1

Number of user material parameters = floating point values read from the user material definition in the input file. This value is returned by UserDLLMaterialParamsCount()

Not a compulsory part of the interface - the user may implement the name returning function in another way.

Definition at line 224 of file CCUserMaterialExampleDLL.c.

Referenced by UserDLLMaterialParamName(), and UserDLLMaterialParamsCount().

5.2.5.4. #define UserStateVarsCount 2

Number of user state variables = floating point values stored in each material point. This value is also returned by UserDLLStateVarsCount()

Not a compulsory part of the interface - the user may implement the name returning function in another way.

Definition at line 243 of file CCUserMaterialExampleDLL.c.

Referenced by UserDLLResetState(), UserDLLStateVarName(), and UserDLLStateVarsCount().

5.2.6. Function Documentation

5.2.6.1. UINT CDECL UserDLLCalculateResponse (double *deps*[], double *teps*[], double *sigma*[], double *E*, double *mu*, double *UserMaterialParams*[], double *UserMaterialState*[])

Purpose: calculates the stress response of a material point to given strain.

This is the key function the user has to define for a new user defined material. It is called when evaluating the material response ("material iterations"). E.g., if a force response is prescribed (in all or 1 direction), this routine will be called repeatedly with changing deps until the difference (error) is acceptable.

Parameters:

	deps	strain increment tensor stored as a vector, first the diagonal terms, followed by the off-diagonal
	teps	total strain tensor (stored as a vector)
in,out	*sigma*	stress tensor stored as a vector (as for deps)
	E	elastic modulus
	mu	Poisson's ratio
	UserMaterialParams	vector of user material parameter values
in,out	*UserMaterialState*	vector of user material state variables in the material point being calculated

Return values:

0	OK
nonzero	error

Definition at line 289 of file CCUserMaterialExampleDLL.c.

References UserDLLTangentStiff().

Here is the call graph for this function:

5.2.6.2. UINT CDECL UserDLLElasticStiff (double *ElasticMatrix*[6 *6], double *E*, double *mu*, double *UserMaterialParams*[])

Purpose: compute the local elastic stiffness matrix. This is usefull for anizotropic materials. If the function is not defined, the default elastic matrix (based on E and mu) is used.

Parameters:

in,out	*ElasticMatrix*	the local elastic stiffness matrix as a vector
	E	Young modulus
	mu	Poisson's ratio
	UserMaterialParams	user material parameters array

Return values:

0	OK
nonzero	error

Definition at line 455 of file CCUserMaterialExampleDLL.c.

5.2.6.3. LPSTR CDECL UserDLLMaterialParamName (UINT *MaterialParamNo*)

Purpose: The name of a user defined material parameter.

The user has to define this function to let the ATENA kernel know the names of the additional parameters the material has, which is required among others when reading the material definition with the parameter values from an input file.

Parameters:

MaterialParamNo	parameter number (id) 0..UserMaterialParamsCount-1

Return values:

string	name of the MaterialParamNo-th user material parameter
NULL	invalid parameter number (out of range)

Definition at line 599 of file CCUserMaterialExampleDLL.c.

References UserMaterialParamNames, and UserMaterialParamsCount.

5.2.6.4. UINT CDECL UserDLLMaterialParamsCount ()

Purpose: The number of user defined material parameters.

The user has to define this function to let the ATENA kernel know how many additional parameters the material has, which is required among others when reading the material definition with the parameter values from an input file.

Returns:

the number of additional user material parameters

Definition at line 565 of file CCUserMaterialExampleDLL.c.

References UserMaterialParamsCount.

5.2.6.5. UINT CDECL UserDLLResetState (double *E*, double *mu*, double *UserMaterialParams*[], double *UserMaterialState*[])

Purpose: resets (initializes) the material point state.

Parameters:

	E	Young modulus
	mu	Poisson's ratio
	UserMaterialParams	user material parameters array
out	*UserMaterialState*	user state variables array

Return values:

0	OK
nonzero	error

Definition at line 369 of file CCUserMaterialExampleDLL.c.

References UserStateVarsCount.

5.2.6.6. UINT CDECL UserDLLSecantStiff (double *SecantMatrix*[], double *E*, double *mu*, double *UserMaterialParams*[], double *UserMaterialState*[])

Purpose: compute the local secant stiffness matrix.

NOT USED in current version.

Definition at line 523 of file CCUserMaterialExampleDLL.c.

5.2.6.7. LPSTR CDECL UserDLLStateVarName (UINT *StateVarNo*)

Purpose: The name of a user defined material state variable.

The user has to define this function to let the ATENA kernel know the names of the additional state variables the material has, which is required among others when offering the list of quantities available for postprocessing.

Parameters:

StateVarNo	parameter number (id) 0..UserStateVarsCount-1

Return values:

string	name of the StateVarNo-th user material state variable
NULL	invalid parameter number (out of range)

Definition at line 622 of file CCUserMaterialExampleDLL.c.

References UserStateVarNames, and UserStateVarsCount.

5.2.6.8. UINT CDECL UserDLLStateVarsCount ()

Purpose: The number of user defined material state variables.

The user has to define this function to let the ATENA kernel know how many additional state variables the material has in each material point, which is required among others when offering the list of quantities available for postprocessing.

Returns:

the number of additional user material state variables

Definition at line 581 of file CCUserMaterialExampleDLL.c.

References UserStateVarsCount.

5.2.6.9. UINT CDECL UserDLLTangentStiff (double *TangentMatrix*[6 *6], double *E*, double *mu*, double *UserMaterialParams*[], double *UserMaterialState*[])

Purpose: compute the local tangential stiffness matrix.

Parameters:

out	*TangentMatrix*	the local tangential stiffness matrix as a vector
	E	Young modulus
	mu	Poisson's ratio
	UserMaterialParams	user material parameters array
	UserMaterialState	user state variables array

Return values:

0	OK
nonzero	error

Definition at line 390 of file CCUserMaterialExampleDLL.c.

References MIN_DIV, and REAL_MAX.

Referenced by UserDLLCalculateResponse().

5.2.6.10. UINT CDECL UserDLLTransformState (double *DefGradient*[], double *eps*[], double *sigma*[], double *E*, double *mu*, double *UserMaterialParams*[], double *UserMaterialState*[])

Purpose: transformation of coordinate system due to large deformations.

Parameters:

	DefGradient	the deformation gradient matrix based used for transformation of the elastic stresses and strains
in,out	*eps*	total strain tensor (stored as a vector)
in,out	*sigma*	stress tensor stored as a vector (as for deps)
	E	Young modulus
	mu	Poisson's ratio
	UserMaterialParams	user material parameters array
in,out	*UserMaterialState*	user state variables array

Return values:

0	OK, the def. gradient matrix should be used - nothing done here, and this function needs not to be called again (to reduce overhead/CPU time)
1	OK, the def. gradient matrix should be used - nothing done here, but this function should be called next time
2	OK, user's own transformation used for user material state vars, the def. matrix should be used to transform strains+stresses
3	OK, user's own transformation used for both user material state vars and strains+stresses
other	error

Definition at line 533 of file CCUserMaterialExampleDLL.c.

5.2.7. Variable Documentation

5.2.7.1. char* UserMaterialParamNames[UserMaterialParamsCount] [static]

```
Initial value:=
{
   "YieldStress"
}
```

All user material parameter names in an array, which is used in UserDLLMaterialParamName()

Not a compulsory part of the interface - the user may implement the name returning function in another way.

Definition at line 231 of file CCUserMaterialExampleDLL.c.

Referenced by UserDLLMaterialParamName().

5.2.7.2. char* UserStateVarNames[UserStateVarsCount] [static]

```
Initial value:=
{
   "vonMisesStress",
   "Yielding"
}
```

All user state variable names in an array, which is used in UserDLLStateVarName()

Not a compulsory part of the interface - the user may implement the name returning function in another way.

Definition at line 250 of file CCUserMaterialExampleDLL.c.

Referenced by UserDLLStateVarName().

5.3. CCUserMaterialFORTRANExampleDLL/CCUserMaterialFORTRANExampleDLL.F90 File Reference

Purpose: File containing an example user material definition - elastic material with von Mises effective stress made available for postprocessing. For users implementing their own user materials it is recommended to use this file or the whole project as a base for their own development, i.e., to copy and modify this.

5.3.1. Data Types

- module ccusermaterialfortranexampledll

5.3.2. Functions/Subroutines

- integer function ccusermaterialfortranexampledll::userdllcalculateresponse (deps, teps, sigma, E, mu, UserMaterialParams, UserMaterialState)

Purpose: calculates the stress response of a material point to given strain.

- integer function ccusermaterialfortranexampledll::userdllresetstate (E, mu, UserMaterialParams, UserMaterialState)

Purpose: resets (initializes) the material point state.

- integer function ccusermaterialfortranexampledll::userdlltangentstiff (TangentMatrix, E, mu, UserMaterialParams, UserMaterialState)

Purpose: compute the local tangential stiffness matrix.

- integer function ccusermaterialfortranexampledll::userdllsecantstiff (SecantMatrix, E, mu, UserMaterialParams, UserMaterialState)
- integer function ccusermaterialfortranexampledll::userdlltransformstate (DefGradient, eps, sigma, E, mu, UserMaterialParams, UserMaterialState)

Purpose: transformation of coordinate system due to large deformations.

- integer function ccusermaterialfortranexampledll::userdllmaterialparamscount ()

Purpose: The number of user defined material parameters.

- integer function ccusermaterialfortranexampledll::userdllstatevarscount ()

Purpose: The number of user defined material state variables.

- type(c_ptr) function ccusermaterialfortranexampledll::userdllmaterialparamname (MaterialParamNo)

Purpose: The name of a user defined material parameter.

- type(c_ptr) function ccusermaterialfortranexampledll::userdllstatevarname (StateVarNo)

Purpose: The name of a user defined material state variable.

Variables

- integer, parameter ccusermaterialfortranexampledll::usermaterialparamscount = 1

Number of user material parameters = floating point values read from the user material definition in the input file. This value is returned by UserDLLMaterialParamsCount()

- character(len=50), dimension(1), parameter ccusermaterialfortranexampledll::usermaterialparamnames = (/"YieldStress"//CHAR(0)/)

All user material parameter names in an array, which is used in UserDLLMaterialParamName(). All strings are zero terminated!

- integer, parameter ccusermaterialfortranexampledll::userstatevarscount = 2

Number of user state variables = floating point values stored in each material point. This value is also returned by UserDLLStateVarsCount()

- character(len=50), dimension(userstatevarscount) ccusermaterialfortranexampledll::userstatevarnames =(/"vonMisesStress"//CHAR(0), "Yielding"//CHAR(0)/)

All user state variable names in an array, which is used in UserDLLStateVarName(). All strings are zero terminated!

- real *8, parameter ccusermaterialfortranexampledll::real_max = HUGE(0.0d0)

Largest real number (value from the floating point library)

- real *8, parameter ccusermaterialfortranexampledll::min_div = TINY(0.0d0)*1000

Minimum accepted divisor used to prevent division by zero/overflow (based on a value from the floating point library)

5.3.3. Detailed Description

Purpose: File containing an example user material definition - elastic material with von Mises effective stress made available for postprocessing. For users implementing their own user materials it is recommended to use this file or the whole project as a base for their own development, i.e., to copy and modify this.

The project is created in Intel FORTRAN using Microsoft Visual Studio 2012, however, it should be a good guideline for implementing an ATENA user material in any FORTRAN compiler. Please note the FORTRAN interface differs from other programming languages due to the need of special handling of argument and return-value passing (especially strings). Moreover, indexing starting from 1 instead of 0 is used in the FORTRAN variant of the interface. All User Material DLLs containing the string "FORTRAN" in the file name are handled as FORTRAN libraries in ATENA.

For a C/C++ example, see CCUserMaterialExampleDLL.c. For a general description of the interface structure and operation, see the User Material Introduction.

The sources of the example DLL and sample ATENA input files (.inp) are included in ATENA installation. You can find them in the `"Examples\ATENA Science\AtenaWin\CCUserMaterialFORTRANExampleDLL"` subdirectory of the ATENA installation directory, typically, `"c:\Program Files\CervenkaConsulting\AtenaV5"`.

Author: Dobromil Pryl

Revision history:

```
        7. 2009   - the file was created
```

See also:

class CCUserMaterial (ATENA internal class wrapping the user material DLL)

Definition in file CCUserMaterialFORTRANExampleDLL.F90.

INDEX