



**ČERVENKA
CONSULTING**

Cervenka Consulting Ltd.

Na Hřebenkách 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, November 6, 2009

Copyright Cervenka Consulting 2008-2009

Trademarks:

Microsoft and Microsoft Visual Studio are registered trademarks of Microsoft Corporation

TABLE OF CONTENTS

1.	USER MATERIAL INTRODUCTION	5
1.1.	Introduction	5
1.2.	Operation	5
1.2.1.	<i>Material Definition</i>	5
1.2.2.	<i>Material Point Initialization</i>	6
1.2.3.	<i>Calculation</i>	6
1.3.	Implementation	6
2.	DIRECTORY DOCUMENTATION	8
2.1.	CCUserMaterialExampleDLL/ Directory Reference	8
2.1.1.	<i>Files</i>	8
2.2.	CCUserMaterialFORTRANExampleDLL/ Directory Reference	8
2.2.1.	<i>Files</i>	8
3.	FILE DOCUMENTATION	9
3.1.	CCUserMaterialExampleDLL/CCUserMaterialDLL.h File Reference	9
3.1.1.	<i>Typedefs</i>	9
3.1.2.	<i>Typedef Documentation</i>	10
3.2.	CCUserMaterialExampleDLL/CCUserMaterialExampleDLL.c File Reference	12
3.2.1.	<i>Defines</i>	13
3.2.2.	<i>Functions</i>	13
3.2.3.	<i>Global variables of UserMaterialDLL</i>	13
3.2.4.	<i>Define Documentation</i>	13
3.2.5.	<i>Function Documentation</i>	14
3.2.6.	<i>Variable Documentation</i>	18
3.3.	CCUserMaterialFORTRANExampleDLL/CCUserMaterialDLL.idl File Reference	19
3.3.1.	<i>Typedefs</i>	19
3.3.2.	<i>Typedef Documentation</i>	19
3.4.	CCUserMaterialFORTRANExampleDLL/CCUserMaterialFORTRANExampleDLL.F90 File Reference	21
3.4.1.	<i>Functions</i>	21
3.4.2.	<i>Function Documentation</i>	22
3.4.3.	<i>Variable Documentation</i>	26
	INDEX	27

1. USER MATERIAL INTRODUCTION

1.1. Introduction

This is a description for users who wish to implement their own material model into ATENA version 4 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 , μ , 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()**. Furthermore, 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).

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\AtenaV4"`,

and the ATENA Theory and ATENA Input File Format manuals.

2. DIRECTORY DOCUMENTATION

2.1. CCUserMaterialExampleDLL/ Directory Reference

CCUserMaterialExampleDLL

2.1.1. Files

- file **CCUserMaterialDLL.h**

Purpose: Header file containing the definition of the user defined functions for user material.

- file **CCUserMaterialExampleDLL.c**

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.

2.2. CCUserMaterialFORTRANExampleDLL/ Directory Reference

CCUserMaterialFORTRANExampleDLL

2.2.1. Files

- file **CCUserMaterialDLL.idl**
- file **CCUserMaterialFORTRANExampleDLL.F90**

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.

3. FILE DOCUMENTATION

3.1. CCUserMaterialExampleDLL/CCUserMaterialDLL.h File Reference

Purpose: Header file containing the definition of the user defined functions for user material.

Author: Dobromil Pryl

Revision history:

1. 2008 - the file was created

Definition in file **CCUserMaterialDLL.h**.

3.1.1. Typedefs

Functions defined in the UserMaterialDLL

- typedef 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 * [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 incompatibility

- typedef 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 * [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)

3.1.2. Typedef Documentation

3.1.2.1. typedef UINT(CDECL* [LPF90UserDLLCalculateResponse](#))(const double deps[], const double tepts[], double sigma[], double *E, double *mu, const double UserMaterialParams[], double UserMaterialState[])

Definition at line 47 of file CCUserMaterialDLL.h.

3.1.2.2. typedef UINT(CDECL* [LPF90UserDLLMaterialParamName](#))(LPSTR retname, UINT *strlen, const UINT *const MaterialParamNo)

Definition at line 67 of file CCUserMaterialDLL.h.

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

Definition at line 50 of file CCUserMaterialDLL.h.

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

Definition at line 56 of file CCUserMaterialDLL.h.

3.1.2.5. typedef UINT(CDECL* [LPF90UserDLLStateVarName](#))(LPSTR retname, UINT *strlen, const UINT *const StateVarNo)

Definition at line 68 of file CCUserMaterialDLL.h.

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

Definition at line 52 of file CCUserMaterialDLL.h.

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

Definition at line 60 of file CCUserMaterialDLL.h.

3.1.2.8. typedef UINT(CDECL* [LPUserDLLCalculateResponse](#))(const double deps[], const double tepts[], double sigma[], double E, double mu, const double UserMaterialParams[], double UserMaterialState[])

Definition at line 19 of file CCUserMaterialDLL.h.

3.1.2.9. typedef LPSTR(CDECL* [LPUserDLLMaterialParamName](#))(const UINT MaterialParamNo)

Definition at line 39 of file CCUserMaterialDLL.h.

3.1.2.10. typedef UINT(CDECL* [LPUserDLLMaterialParamsCount](#))()

Definition at line 37 of file CCUserMaterialDLL.h.

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

Definition at line 22 of file CCUserMaterialDLL.h.

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

Definition at line 28 of file CCUserMaterialDLL.h.

3.1.2.13. typedef LPSTR(CDECL* [LPUserDLLStateVarName](#))(const UINT StateVarNo)

Definition at line 40 of file CCUserMaterialDLL.h.

3.1.2.14. typedef UINT(CDECL* [LPUserDLLStateVarsCount](#))()

Definition at line 38 of file CCUserMaterialDLL.h.

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

Definition at line 24 of file CCUserMaterialDLL.h.

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

Definition at line 32 of file CCUserMaterialDLL.h.

3.2. 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.

The project is created in Microsoft Visual Studio 2008, 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\AtenaV4".
```

See also:

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

Author: Dobromil Pryl

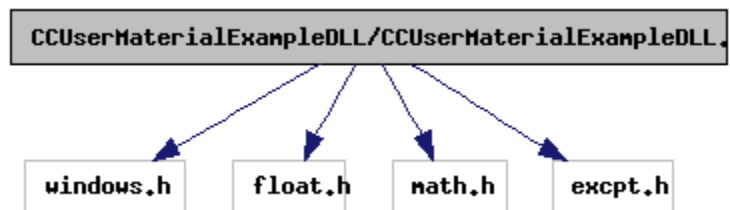
Revision history:

1. 2008 - the file was created

Definition in file **CCUserMaterialExampleDLL.c**.

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

Include dependency graph for CCUserMaterialExampleDLL.c:



3.2.1. Defines

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

3.2.2. Functions

User defined functions of UserMaterialDLL

See also: class CCUserMaterial

- UINTEDECL **UserDLLTangentStiff** (double TangentMatrix[6 *6],double E,double mu,double UserMaterialParams[],double UserMaterialState[])
Purpose: compute the local tangential stiffness matrix.
- UINTEDECL **UserDLLCalculateResponse** (double deps[], double tepts[], double sigma[], double E, double mu, double UserMaterialParams[], double UserMaterialState[])
Purpose: calculates the stress response of a material point to given strain.
- UINTEDECL **UserDLLResetState** (double E, double mu, double UserMaterialParams[], double UserMaterialState[])
Purpose: resets (initializes) the material point state.
- UINTEDECL **UserDLLSecantStiff** (double SecantMatrix[], double E, double mu, double UserMaterialParams[], double UserMaterialState[])
Purpose: compute the local secant stiffness matrix.
- UINTEDECL **UserDLLTransformState** (double DefGradient[], double eps[], double sigma[], double E, double mu, double UserMaterialParams[], double UserMaterialState[])
Purpose: transformation of coordinate system due to large deformations.
- UINTEDECL **UserDLLMaterialParamsCount** ()
Purpose: The number of user defined material parameters.
- UINTEDECL **UserDLLStateVarsCount** ()
Purpose: The number of user defined material state variables.
- LPSTR CDECL **UserDLLMaterialParamName** (UINTEDECL MaterialParamNo)
Purpose: The name of a user defined material parameter.
- LPSTR CDECL **UserDLLStateVarName** (UINTEDECL StateVarNo)
Purpose: The name of a user defined material state variable.

3.2.3. Global variables of UserMaterialDLL

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

3.2.4. Define Documentation

3.2.4.1. #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 263 of file CCUserMaterialExampleDLL.c.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLTangentStiff(), and UserDLLTangentStiff().

3.2.4.2. #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 256 of file CCUserMaterialExampleDLL.c.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLTangentStiff(), and UserDLLTangentStiff().

3.2.4.3. #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 218 of file CCUserMaterialExampleDLL.c.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLMaterialParamName(), UserDLLMaterialParamName(), CCUserMaterialFORTRANExampleDLL::UserDLLMaterialParamsCount(), and UserDLLMaterialParamsCount().

3.2.4.4. #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 237 of file CCUserMaterialExampleDLL.c.

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

3.2.5. Function Documentation

3.2.5.1. 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

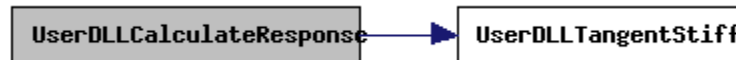
nonzero error

Definition at line 284 of file CCUserMaterialExampleDLL.c.

References UserDLLTangentStiff().

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLCalculateResponse().

Here is the call graph for this function:

**3.2.5.2. 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 527 of file CCUserMaterialExampleDLL.c.

References UserMaterialParamNames, and UserMaterialParamsCount.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLMaterialParamName().

3.2.5.3. 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 493 of file CCUserMaterialExampleDLL.c.

References UserMaterialParamsCount.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLMaterialParamsCount().

3.2.5.4. 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 364 of file CCUserMaterialExampleDLL.c.

References UserStateVarsCount.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLResetState().

3.2.5.5. 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 451 of file CCUserMaterialExampleDLL.c.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLSecantStiff().

3.2.5.6. 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 550 of file CCUserMaterialExampleDLL.c.

References UserStateVarNames, and UserStateVarsCount.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLStateVarName().

3.2.5.7. 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 509 of file CCUserMaterialExampleDLL.c.

References UserStateVarsCount.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLStateVarsCount().

3.2.5.8. 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 385 of file CCUserMaterialExampleDLL.c.

References MIN_DIV, and REAL_MAX.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLCalculateResponse(), UserDLLCalculateResponse(), and CCUserMaterialFORTRANExampleDLL::UserDLLTangentStiff().

3.2.5.9. 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 461 of file CCUserMaterialExampleDLL.c.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLTransformState().

3.2.6. Variable Documentation

3.2.6.1. `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 225 of file CCUserMaterialExampleDLL.c.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLMaterialParamName(), and UserDLLMaterialParamName().

3.2.6.2. `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 244 of file CCUserMaterialExampleDLL.c.

Referenced by CCUserMaterialFORTRANExampleDLL::UserDLLStateVarName(), and UserDLLStateVarName().

3.3. CCUserMaterialFORTRANExampleDLL/CCUserMaterialDLL.idl File Reference

3.3.1. Typedefs

- typedef 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 * [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)

3.3.2. Typedef Documentation

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

Definition at line 19 of file CCUserMaterialDLL.idl.

3.3.2.2. typedef LPSTR(CDECL* [LPUserDLLMaterialParamName](#))(const UINT MaterialParamNo)

Definition at line 39 of file CCUserMaterialDLL.idl.

3.3.2.3. typedef UINT(CDECL* [LPUserDLLMaterialParamsCount](#))()

Definition at line 37 of file CCUserMaterialDLL.idl.

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

Definition at line 22 of file CCUserMaterialDLL.idl.

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

Definition at line 28 of file CCUserMaterialDLL.idl.

3.3.2.6. typedef LPSTR(CDECL* [LPUserDLLStateVarName](#))(const UINT StateVarNo)

Definition at line 40 of file CCUserMaterialDLL.idl.

3.3.2.7. typedef UINT(CDECL* [LPUserDLLStateVarsCount](#))()

Definition at line 38 of file CCUserMaterialDLL.idl.

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

Definition at line 24 of file CCUserMaterialDLL.idl.

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

Definition at line 32 of file CCUserMaterialDLL.idl.

3.4. 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.

The project is created in Intel FORTRAN using Microsoft Visual Studio 2008, 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\AtenaV4".

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](#).

3.4.1. Functions

- integer [CCUserMaterialFORTRANExampleDLL::UserDLLCalculateResponse](#) (deps, tepts, sigma, E, mu, UserMaterialParams, UserMaterialState)

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

- integer [CCUserMaterialFORTRANExampleDLL::UserDLLResetState](#) (E, mu, UserMaterialParams, UserMaterialState)

Purpose: resets (initializes) the material point state.

- integer [CCUserMaterialFORTRANExampleDLL::UserDLLTangentStiff](#) (TangentMatrix, E, mu, UserMaterialParams, UserMaterialState)

Purpose: compute the local tangential stiffness matrix.

- integer [CCUserMaterialFORTRANExampleDLL::UserDLLSecantStiff](#) (SecantMatrix, E, mu, UserMaterialParams, UserMaterialState)

- integer [CCUserMaterialFORTRANExampleDLL::UserDLLTransformState](#) (DefGradient, eps, sigma, E, mu, UserMaterialParams, UserMaterialState)

Purpose: transformation of coordinate system due to large deformations.

- integer [CCUserMaterialFORTRANExampleDLL::UserDLLMaterialParamsCount](#) ()

Purpose: The number of user defined material parameters.

- integer [CCUserMaterialFORTRANExampleDLL::UserDLLStateVarsCount](#) ()

Purpose: The number of user defined material state variables.

- character(len=50) [CCUserMaterialFORTRANExampleDLL::UserDLLMaterialParamName](#) (MaterialParamNo)

Purpose: The name of a user defined material parameter.

- character(len=50) [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), parameter [CCUserMaterialFORTRANExampleDLL::UserMaterialParamNames](#) =
 (//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) [CCUserMaterialFORTRANExampleDLL::UserStateVarNames](#) = (//CHAR(0),
 //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).

3.4.2. Function Documentation

3.4.2.1. integer [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)

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

UserMaterialState [in+out] vector of user material state variables in the material point being calculated

Definition at line 106 of file [CCUserMaterialFORTRANExampleDLL.F90](#).

References [UserDLLCalculateResponse\(\)](#), and [UserDLLTangentStiff\(\)](#).

Here is the call graph for this function:



3.4.2.2. character(len=50)

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:

MaterialParamNo parameter number

Definition at line 368 of file CCUserMaterialFORTRANExampleDLL.F90.

References UserDLLMaterialParamName(), UserMaterialParamNames, and UserMaterialParamsCount.

3.4.2.3. integer

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 329 of file CCUserMaterialFORTRANExampleDLL.F90.

References UserDLLMaterialParamsCount(), and UserMaterialParamsCount.

3.4.2.4. integer CCUserMaterialFORTRANExampleDLL::UserDLLResetState (real*8 E, real*8 mu, real*8,dimension(usermaterialparamscount) UserMaterialParams, real*8,dimension(userstatevarscout),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

UserMaterialState [out] user state variables array

Definition at line 167 of file CCUserMaterialFORTRANExampleDLL.F90.
References UserDLLResetState().

3.4.2.5. integer 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:

SecantMatrix [out] 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 270 of file CCUserMaterialFORTRANExampleDLL.F90.
References UserDLLSecantStiff().

3.4.2.6. character(len=50)

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 394 of file CCUserMaterialFORTRANExampleDLL.F90.
References UserDLLStateVarName(), UserStateVarNames, and UserStateVarsCount.

3.4.2.7. integer 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 347 of file CCUserMaterialFORTRANExampleDLL.F90.
References UserDLLStateVarsCount(), and UserStateVarsCount.

**3.4.2.8. integer 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:

TangentMatrix [out] 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 191 of file CCUserMaterialFORTRANExampleDLL.F90.

References MIN_DIV, REAL_MAX, and UserDLLTangentStiff().

**3.4.2.9. integer 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
eps [in+out] total strain tensor (stored as a vector)
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 301 of file CCUserMaterialFORTRANExampleDLL.F90.

References UserDLLTransformState().

3.4.3. Variable Documentation

3.4.3.1. 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.

3.4.3.2. 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.

3.4.3.3. character (len=50),parameter [CCUserMaterialFORTRANExampleDLL::UserMaterialParamNames](#) = (///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.

3.4.3.4. 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.

3.4.3.5. character (len=50) [CCUserMaterialFORTRANExampleDLL::UserStateVarNames](#) = (///CHAR(0), //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.

3.4.3.6. 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.

INDEX

- CCUserMaterialDLL.h
 - LPF90UserDLLCalculateResponse, 10
 - LPF90UserDLLMaterialParamName, 10
 - LPF90UserDLLResetState, 10
 - LPF90UserDLLSecantStiff, 10
 - LPF90UserDLLStateVarName, 10
 - LPF90UserDLLTangentStiff, 10
 - LPF90UserDLLTransformState, 10
 - LPUserDLLCalculateResponse, 11
 - LPUserDLLMaterialParamName, 11
 - LPUserDLLMaterialParamsCount, 11
 - LPUserDLLResetState, 11
 - LPUserDLLSecantStiff, 11
 - LPUserDLLStateVarName, 11
 - LPUserDLLStateVarsCount, 11
 - LPUserDLLTangentStiff, 11
 - LPUserDLLTransformState, 12
- CCUserMaterialDLL.idl
 - LPUserDLLCalculateResponse, 19
 - LPUserDLLMaterialParamName, 19
 - LPUserDLLMaterialParamsCount, 19
 - LPUserDLLResetState, 19
 - LPUserDLLSecantStiff, 19
 - LPUserDLLStateVarName, 20
 - LPUserDLLStateVarsCount, 20
 - LPUserDLLTangentStiff, 20
 - LPUserDLLTransformState, 20
- CCUserMaterialExampleDLL.c
 - MIN_DIV, 13
 - REAL_MAX, 14
 - UserDLLCalculateResponse, 14
 - UserDLLMaterialParamName, 15
 - UserDLLMaterialParamsCount, 15
 - UserDLLResetState, 16
 - UserDLLSecantStiff, 16
 - UserDLLStateVarName, 16
 - UserDLLStateVarsCount, 17
 - UserDLLTangentStiff, 17
 - UserDLLTransformState, 17
 - UserMaterialParamNames, 18
 - UserMaterialParamsCount, 14
 - UserStateVarNames, 18
 - UserStateVarsCount, 14
- CCUserMaterialExampleDLL/ Directory Reference, 8
- CCUserMaterialExampleDLL/CCUserMaterialExampleDLL.c, 12
- CCUserMaterialFORTRANExampleDLL
 - MIN_DIV, 26
 - REAL_MAX, 26
 - UserDLLCalculateResponse, 22
 - UserDLLMaterialParamName, 23
 - UserDLLMaterialParamsCount, 23
 - UserDLLResetState, 23
 - UserDLLSecantStiff, 24
 - UserDLLStateVarName, 24
 - UserDLLStateVarsCount, 24
 - UserDLLTangentStiff, 25
 - UserDLLTransformState, 25
 - UserMaterialParamNames, 26
 - UserMaterialParamsCount, 26
 - UserStateVarNames, 26
 - UserStateVarsCount, 26
- CCUserMaterialFORTRANExampleDLL/
 - Directory Reference, 8
- CCUserMaterialFORTRANExampleDLL/CCUserMaterialDLL.idl, 19
- CCUserMaterialFORTRANExampleDLL/CCUserMaterialFORTRANExampleDLL.F90, 21
- LPF90UserDLLCalculateResponse
 - CCUserMaterialDLL.h, 10
- LPF90UserDLLMaterialParamName
 - CCUserMaterialDLL.h, 10
- LPF90UserDLLResetState
 - CCUserMaterialDLL.h, 10
- LPF90UserDLLSecantStiff
 - CCUserMaterialDLL.h, 10
- LPF90UserDLLStateVarName
 - CCUserMaterialDLL.h, 10
- LPF90UserDLLTangentStiff
 - CCUserMaterialDLL.h, 10
- LPF90UserDLLTransformState
 - CCUserMaterialDLL.h, 10
- LPUserDLLCalculateResponse
 - CCUserMaterialDLL.h, 11
 - CCUserMaterialDLL.idl, 19
- LPUserDLLMaterialParamName
 - CCUserMaterialDLL.h, 11
 - CCUserMaterialDLL.idl, 19
- LPUserDLLMaterialParamsCount
 - CCUserMaterialDLL.h, 11
 - CCUserMaterialDLL.idl, 19
- LPUserDLLResetState
 - CCUserMaterialDLL.h, 11
 - CCUserMaterialDLL.idl, 19
- LPUserDLLSecantStiff
 - CCUserMaterialDLL.h, 11
 - CCUserMaterialDLL.idl, 19
- LPUserDLLStateVarName
 - CCUserMaterialDLL.h, 11
 - CCUserMaterialDLL.idl, 20
- LPUserDLLStateVarsCount
 - CCUserMaterialDLL.h, 11
 - CCUserMaterialDLL.idl, 20
- LPUserDLLTangentStiff
 - CCUserMaterialDLL.h, 11
 - CCUserMaterialDLL.idl, 20
- LPUserDLLTransformState
 - CCUserMaterialDLL.h, 12
 - CCUserMaterialDLL.idl, 20
- MIN_DIV
 - CCUserMaterialExampleDLL.c, 13

CCUserMaterialFORTRANExampleDLL, 26
REAL_MAX
CCUserMaterialExampleDLL.c, 14
CCUserMaterialFORTRANExampleDLL, 26
UserDLLCalculateResponse
CCUserMaterialExampleDLL.c, 14
CCUserMaterialFORTRANExampleDLL, 22
UserDLLMaterialParamName
CCUserMaterialExampleDLL.c, 15
CCUserMaterialFORTRANExampleDLL, 23
UserDLLMaterialParamsCount
CCUserMaterialExampleDLL.c, 15
CCUserMaterialFORTRANExampleDLL, 23
UserDLLResetState
CCUserMaterialExampleDLL.c, 16
CCUserMaterialFORTRANExampleDLL, 23
UserDLLSecantStiff
CCUserMaterialExampleDLL.c, 16
CCUserMaterialFORTRANExampleDLL, 24
UserDLLStateVarName
CCUserMaterialExampleDLL.c, 16
CCUserMaterialFORTRANExampleDLL, 24
UserDLLStateVarsCount
CCUserMaterialExampleDLL.c, 17
CCUserMaterialFORTRANExampleDLL, 24
UserDLLTangentStiff
CCUserMaterialExampleDLL.c, 17
CCUserMaterialFORTRANExampleDLL, 25
UserDLLTransformState
CCUserMaterialExampleDLL.c, 17
CCUserMaterialFORTRANExampleDLL, 25
UserMaterialParamNames
CCUserMaterialExampleDLL.c, 18
CCUserMaterialFORTRANExampleDLL, 26
UserMaterialParamsCount
CCUserMaterialExampleDLL.c, 14
CCUserMaterialFORTRANExampleDLL, 26
UserStateVarNames
CCUserMaterialExampleDLL.c, 18
CCUserMaterialFORTRANExampleDLL, 26
UserStateVarsCount
CCUserMaterialExampleDLL.c, 14
CCUserMaterialFORTRANExampleDLL, 26