Chapter 4.
Functions DM Ga thru Iz

Topic: Ignore

DM_get_active_patch

Function: Deformable Modeling, DML Patches

Action: Gets the pointer to the hierarchy's active patch or NULL.

Prototype:
```
DS_dmod* DM_get_active_patch (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod in hierarchy
                            // to search
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the value of the deformable model's active deformable model pointer, or NULL when the input deformable model is NULL on entry.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_active_patch_tag

Function: Deformable Modeling, DML Patches, DML Tags

Action: Gets the tag number of the hierarchy's active patch.

Prototype:
```
int DM_get_active_patch_tag (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod in hierarchy
                            // to search
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the tag value of the deformable model's active deformable model.

Errors:

DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NO_ACTIVE_DMOD
The input deformable model's hierarchy has no currently active deformable model.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_alpha

Function: Deformable Modeling, Deformable Model Management

Action: Gets the deformable model's alpha values and returns 0 or an error.

Prototype:
```
void DM_get_alpha (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    double* alpha,          // out: resistance to
                            // stretch values curves:
                            // [alpha=1.0], sized:[1]
                            // surfs: [au=1.0,
                            // av=1.0, atheta=0.0],
                            // sized:[3]
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's resistance to stretch alpha values. For curves there is only one alpha value and it is stored in alpha[0]. In surfaces the alpha value is a second order tensor which can be described with three numbers, a u_dir resistance, a v_dir resistance, and a rotation. The rotation is the angle between the *u* and *v* principle directions of the surface and the material property directions. The alpha values are stored as alpha[0] = au, alpha[1] = av, alpha[2] = atheta.

In practice, we have discovered that if end users become accustomed to sculpting by dragging point constraints, attempting to sculpt with alpha values is very counter intuitive. When sculpting, we recommend that you leave the alpha values at their default values and rely solely on loads and constraints for sculpting.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NULL_OUTPUT_PTR
The alpha cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_area_cstrn_flag

Function: Deformable Modeling, DML Constraints

Action: Returns the area constraint's fixed inside/outside flag. 0=zone is free, 1=zone compliment area is free.

Prototype:
```
int DM_get_area_cstrn_flag (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of dmod
                            // hierarchy to search
    int tag,                // n: cstrn identifier
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns 0 when the zone area is deformable and the zone compliment area is fixed. Returns 1 when the zone area is fixed and the zone compliment area is free to move.

Passes the change request down to the deformable model so that all affected arrays can be updated.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag does not identify an area constraint.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_attractor

Function: Deformable Modeling

Action: Writes the place load data into return arguments and returns 0 or an error.

Prototype:

```
void DM_get_attractor (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of target
                            // dmod hierarchy to
                            // search
    int tag,                // in: load identifier
    double* image_pt,       // out: image_pt for
                            // attractor location
                            // sized:[image_dim]
    int& power,             // out: measure of load's
                            // locality
                            // 0,1=global,
                            // 2,3,...=more local
    double& gain,           // out: magnitude of load
                            // gain
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function searches for a load identified by specific tag in the entire patch hierarchy. When the tag specifies an attractor load, it fills all the output arguments appropriately and returns 0. The active deformable model changes to the one containing the load. Otherwise, returns DM_TAG_OBJECT_NOT_FOUND and leaves output arguments unmodified.

Errors:

DM_NULL_INPUT_PTR
The deformable model or beta cannot be NULL on input.

DM_NULL_OUTPUT_PTR
The image point is NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_beta

Function: Deformable Modeling, Deformable Model Management

Action: Gets the deformable model's beta values and returns 0 or an error.

Prototype:

```
void DM_get_beta (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: dmod to be queried
    double* beta,              // out: resistance to
                               // bending values
                               // curves: [beta=5.0],
                               // sized:[1]
                               // surfs: [bu=5.0,
                               // bv=5.0, btheta=0.0],
                               // sized:[3]
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's resistance to bending beta values. For curves there is only one beta value and it is stored in beta[0]. In surfaces the beta value is a second order tensor which can be described with three numbers, a u_dir resistance, a v_dir resistance, and a rotation. The rotation is the angle between the *u* and *v* principle directions of the surface and the material property directions. The beta values are stored as beta[0] = bu, beta[1] = bv, beta[2] = btheta.

In practice we have discovered that if end users become accustomed to sculpting by dragging point constraints, attempting to sculpt with beta values is very counter intuitive. When sculpting, we recommend that you leave the beta values at their default values and rely solely on loads and constraints for sculpting.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NULL_OUTPUT_PTR
The beta cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_bspline_curve

Function: Deformable Modeling, DML Create and Query

Action: Gets B–spline curve data and returns 0 for success or DM_NOT_A_B–spline.

Prototype:

```
void DM_get_bspline_curve (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_pfunc* pfunc,        // in: pfunc data
                            // structure to examine
    int& image_dim,         // out: image space size
                            // (2=2d, 3=3d)
    int& degree,            // out: polynomial degree
    int& dof_count,         // out: control point
                            // count
    int& knot_count,        // out: number of
                            // distinct knot values
    int*& knot_index,       // out: multiple knot
                            // count array
                            // sized:[knot_count]
                            // specify multiple knots
                            // by setting count[i]=
                            // maximum knot index
                            // value for location
                            // knot[i]
                            // Example: no multiples
                            // count = [0,1,2,3,4]
                            // some multiples =
                            // [1,2,4]
    double*& knot,          // out: knot value array
                            // sized:[knot_count]
                            // ordered [u0<u1<u2,...]
                            // output in
                            // internal_pfunc_space
    double*& dof_vec,       // out: dof_vec vals
```

```
                                // (control pt locs)
                                // sized:
                                // [image_dim*dof_count]
                                // ordered
                                // [xyz0,xyz1,...]
double*& dof_def,               // out: default shape
                                // (control pt locs)
                                // sized:
                                // [image_dim*dof_count]
                                // ordered
                                // [xyz0,xyz1,...]
int& end_cond,                  // out: enforce
                                // continuity between end
                                // points with
                                // constraints. If used
                                // must have multiple
                                // knots on the ends of
                                // B-spline.
                                // 0=open default
                                // (no constraints)
                                // 1=closed
                                // (C0 continuity)
                                // 2=periodic
                                // (C1 continuity)
int& ntgrl_degree,              // out: for deformable
                                // modeling - the
                                // degree polynomial
                                // exactly integrated
                                // should be larger than
                                // degree. 10 is default.
                                // note: total_knot_count
                                // = dof_count + deg. - 1
                                // the total and distinct
                                // knot counts vary when
                                // there are multiple
                                // knots.
                                // note: any input array
                                // set to NULL will be
                                // ignored by subroutine.
SDM_options* sdmo               // in:SDM_options pointer
    = NULL                      // note: total_knot_count
                                // = dof_count+degree-1
                                // the total and distinct
                                // knot counts vary when
                                // there are multiple
```

```
                                        // knots
                                        // note: any input array
                                        // set to NULL will be
                                        // ignored by subroutine
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies image_dim, degree, dof_count, knot_count,knot_index, knot, dof_vec, dof_def, end_cond, ntgrl_degree.

This function places copies of the pfunc data into all the output variables. See DS_make_bspline_curve for argument use. All output arrays are to be allocated by the calling program. Arrays of the wrong size will cause memory errors. Any array input value of NULL will cause this subroutine to skip that output data.

This function also updates the output pointer values to point to the pfunc's nested data. Do not use these pointer values to free memory, nor after the pfunc has been deleted.

ntgrl_degree is an internal term used by the deformable modeling package to control the accuracy of building the deformable modeling equations. It is the max_degree polynomial function that is accurately integrated by the package. If the value is too small the system will not deform as expected. If the value is too large the system slows down. The term is returned for debugging and informative purposes only, and its value should always be at least twice the degree value. The maximum value is 79.

The B–spline knot values returned in the knot array define the DS_pfunc's domain space which is called the internal_pfunc_space. When used in a deformable model, the internal_pfunc_space may be scaled by a constant factor to optimize the numerical accuracy of computing high order derivatives. The domain space of the deformable model is called the orig_dmod_space, and it remains constant for input and output purposes. The relationship between the two spaces is always just, internal_pfunc_space = domain_scale * orig_dmod_space.

The domain_scale factor may be retrieved with the function, DM_get_dmod_domain_scale().

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on input.

DM_NOT_A_BSPLINE
The input pfunc type must be a B-spline.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_bspline_surface

Function: Deformable Modeling, DML Create and Query

Action: Retrieves B–spline surface data and returns 0 for success or an error.

Prototype:

```
void DM_get_bspline_surface (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_pfunc* pfunc,           // in: pfunc data
                               // structure to examine
    int& image_dim,            // out: image space size
                               // (2=2d,3=3d)
    int& degree_u,             // out: u dir
                               // polynomial degree
    int& dof_count_u,          // out: u dir control
                               // point count
    int& knot_count_u,         // out: u dir number of
                               // distinct knot values
    int*& knot_index_u,        // out: u dir multiple
                               // knot count array.
                               // sized:[knot_count]
                               // specify the multiple
                               // knots by setting
                               // count[i] = maximum
                               // knot index value for
                               // location knot[i]
                               // Examples:
                               // no multiples count =
                               // [0,1,2,3,4]
                               // some multiples =
```

```
                                    // [1,2,4]
    double*& knot_u,                // out: u dir knot
                                    // value array
                                    // sized:[knot_count]
                                    // ordered [u0<u1<u2...]
                                    // output in
                                    // internal_pfunc_space.
    int& degree_v,                  // out: v dir
                                    // polynomial degree
    int& dof_count_v,               // out: v dir control
                                    // point count
    int& knot_count_v,              // out: v dir number of
                                    // distinct knot values
    int*& knot_index_v,             // out: v dir multiple
                                    // knot count array.
                                    // sized:[knot_count]
                                    // specify the multiple
                                    // knots by setting
                                    // count[i] = maximum
                                    // knot index value for
                                    // location knot[i]
                                    // Examples:
                                    // no multiples count =
                                    // [0,1,2,3,4]
                                    // some multiples =
                                    // [1,2,4]
    double*& knot_v,                // out: v dir knot
                                    // value array
                                    // sized:[knot_count]
                                    // ordered [u0<u1<u2...]
                                    // output in
                                    // internal_pfunc_space
    double*& dof_vec,               // out: dof_vec vals
                                    // (control pt locs)
                                    // sized:[image_dim*
                                    // dof_count]
                                    // ordered:[xyz0, xyz1,
                                    // xyz2, ...]
    double*& dof_def,               // out: default shape
                                    // (control pt locs)
                                    // sized:[image_dim*
                                    // dof_count]
                                    // ordered:[xyz0, xyz1,
                                    // xyz2, ...]
    int& end_cond_u,                // out: one of 0=open or
```

```
                                    // 1=closed or 2=periodic
        int& singular_u,            // out: one of 0=none or
                                    // 1=low or 2=high or
                                    // 3=both
        int& end_cond_v,            // out: one of 0=open or
                                    // 1=closed or 2=periodic
        int& singular_v,            // out: one of 0=none or
                                    // 1=low or 2=high or
                                    // 3=both
        int& ntgrl_degree,          // out: for deformable
                                    // modeling – the degree
                                    // polynomial exactly
                                    // integrated should be
                                    // larger than the
                                    // degree. 10 is a good
                                    // value. default
                                    // value = 10
                                    // note: total_knot_count
                                    // = dof_count +
                                    // degree – 1
                                    // the total and distinct
                                    // knot counts vary when
                                    // there are multiple
                                    // knots.
                                    // note: any input array
                                    // set to NULL will be
                                    // ignored by subroutine.
        SDM_options* sdmo           // in:SDM_options pointer
            = NULL                  // note: total_knot_count
                                    // = dof_count+degree–1
                                    // the total and distinct
                                    // knot counts vary when
                                    // there are multiple
                                    // knots
                                    // note: any input array
                                    // set to NULL will be
                                    // ignored by  subroutine
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”dshusk/dskernel/dmapi.hxx”
#include ”dshusk/dskernel/dspfunc.hxx”
#include ”dshusk/dskernel/sdm_options.hxx”
```

Description: Modifies image_dim, dof_vec, dof_def, ntgrl_degree, degree_u,dof_count_u,knot_count_u,knot_index_u,knot_u, degree_v,dof_count_v,knot_count_v,knot_index_v,knot_v, end_cond_u, singular_u, end_cond_v, singular_v.

This function places copies of the pfunc data into all the output variables. See DS_make_bspline_curve for argument use. All output arrays are to be allocated by the calling program. Arrays of the wrong size will cause memory errors. Any array input value of NULL will cause this subroutine to skip that output data.

This function also updates the output pointer values to point to the pfunc's nested data. Do not use these pointer values to free memory, nor after the pfunc has been deleted.

ntgrl_degree is an internal term used by the deformable modeling package to control the accuracy of building the deformable modeling equations. It is the max_degree polynomial function that is accurately integrated by the package. If the value is too small the system will not deform as expected. If the value is too large the system slows down. The term is returned for debugging and informative purposes only, and its value should always be at least twice the degree value. The maximum value is 79.

The B–spline knot values returned in the knot array define the DS_pfunc's domain space which is called the internal_pfunc_space. When used in a deformable model, the internal_pfunc_space may be scaled by a constant factor to optimize the numerical accuracy of computing high order derivatives. The domain space of the deformable model is called the orig_dmod_space, and it remains constant for input and output purposes. The relationship between the two spaces is always just, internal_pfunc_space = domain_scale * orig_dmod_space.

The domain_scale factor may be retrieved with the function, DM_get_dmod_domain_scale().

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on input.

DM_NOT_A_BSPLINE
The input pfunc type must be a B-spline.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_child

Function: Deformable Modeling, DML Patches

Action: Gets a deformable model's child and returns the child pointer or NULL.

Prototype:
```
DS_dmod* DM_get_child (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's child pointer.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_circ_curve

Function: Deformable Modeling

Action: Retrieves circular curve data and returns 0 for success or DM_NOT_A_CIRC.

Prototype:
```
void DM_get_circ_curve (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_pfunc* pfunc,        // in: pfunc data
```

```
                                    // structure to examine
    int& image_dim,                 // out: image space size
                                    // (2=2d,3=3d)
    double*& dof_vec,               // out: the 3 current
                                    // dofs [xc,xa,xb]
                                    // xc = center_point
                                    // xa = vec from
                                    // center_pt to axis_1
                                    // xb = vec from
                                    // center_pt to axis_2
                                    // Sets dof_def =
                                    // dof_vec;
                                    // Sized:[3*image_dim]
                                    // (required)
    double*& dof_def,               // out: the 3 default
                                    // dofs [xc,xa,xb]
                                    // xc = center_point
                                    // xa = vec from
                                    // center_pt to axis_1
                                    // xb = vec from
                                    // center_pt to axis_2
                                    // Sets dof_def =
                                    // dof_vec;
                                    // Sized:[3*image_dim]
                                    // (required)
    int& elem_count,                // out: break circ into
                                    // elems for integration
                                    // default value = 8.
    int& ntgrl_degree,              // out: for deformable
                                    // modeling – the degree
                                    // polynomial exactly
                                    // integrated. 10 is a
                                    // good value. default
                                    // value = 10. note: any
                                    // input array set to
                                    // NULL will be ignored
                                    // by subroutine.
    SDM_options* sdmo               // in:SDM_options pointer
        = NULL                      // note: any input array
                                    // set to NULL will be
                                    // ignored by subroutine
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies image_dim, dof_vec, dof_def, elem_count, ntgrl_degree.

Loads the defining data for a circular curve into the output variables. See DM_make_circ_curve() for the definition of a circular curve and the use of its defining arguments.

This function also updates the output pointer values to point to the pfunc's nested data. Do not use these pointer values to free memory, nor use these values after the pfunc has been deleted.

elem_count is the number of elements that a circular curve is subdivided into to improve the accuracy of numerical integration. This value should be at least as large as DS_DEFAULT_ELEM_COUNT. It is returned for debugging and informative reasons only.

ntgrl_degree is an internal term used by the deformable modeling package to control the accuracy of building the deformable modeling equations. It is the max_degree polynomial function that is accurately integrated by the package. If the value is too small the system will not deform as expected. If the value is too large the system slows down. The term is returned for debugging and informative purposes only, and its value should always be at least twice the degree value. The maximum value is 79.

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on input.

DM_NOT_A_CIRC
The pfunc type must be a circ.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_comb_graphics

Function: Deformable Modeling, DML Graphics

Action: Gets the parameters of the deformable model's curvature comb graphics and returns 0 for success or an error.

Prototype:

```
void DM_get_comb_graphics (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    int& comb_pt_count,     // out: number of
                            // times-per-elem
    double& comb_gain,      // out: gain on comb
                            // vector magnitudes
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's curvature comb plot parameters. A curvature comb plot is an excellent way to communicate a curve's curvature to an end user. The curvature comb is a set of vectors connected along the length of the curve. The length of each vector is proportional to the curvature at the point where the vector connects to the curve. the direction of the vector is in the binomial of the curve.

The parameters of the plot include comb_pt_count and comb_gain.

The comb_pt_count is the number of tines plotted within each element of the deformable model–>Pfunc() shape. The comb_gain is a magnitude that is multiplied on the curvature to specify the length of the curvature vectors on the screen. Typically negative numbers are used so that the tines of the comb do not intersect one another.

Errors: None

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_cstrn

Function: Deformable Modeling, DML Constraints

Action: Loads constraint data into return arguments and returns 0 for success or an error.

Prototype:

```
void DM_get_cstrn (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: member of target
                               // dmod hierarchy
    int tag,                   // in: constraint
                               // identifier
    DS_TAGS& type_id,          // out: cstrn type_id
                               // DS_tag_pt_cstrn
                               // DS_tag_pt_seam
                               // DS_tag_crv_cstrn
                               // DS_tag_crv_seam
                               // DS_tag_link_cstrn
                               // DS_tag_area_cstrn
                               // DS_tag_link_load
                               // DS_tag_crv_load
    DS_CSTRN_SRC& src_type,    // out: one of
                               // ds_solid_cstrn
                               // ds_bound_cstrn
                               // ds_user_cstrn
                               // ds_seam_cstrn
                               // ds_link_cstrn
    int& patch1_tag,           // out: containing patch
                               // tag
    int& patch2_tag,           // out: containing patch
                               // tag
    char* shape1,              // out: shape type,
                               // sized:[20]
    char* shape2,              // out: shape type,
                               // sized:[20]
    int& behavior,             // out:
    int& state,                // out: 0=disabled,
                               // 1=enabled
    int& rights,               // out: orof DM_STOPABLE,
                               // DM_DELETABLE
    void*& src1_data,          // out: src_data stored
                               // with cstrn
    void*& src2_data,          // out: link_cstrn 2nd
                               // src_data
    int domain_flag,           // orig_dmod_space
                               // 1=domain_pts in
                               // unit_space.
                               // 2=domain_pts in
                               // internal_pfunc_space.
    int& uv1_pts_count,        // out: number of pts
```

```
                                    // loaded into uv1_pts
    int& uv2_pts_count,             // out: number of pts
                                    // loaded into uv2_pts
    double uv1_pts[6],              // out: defining locs for
                                    // simple shapes
                                    // ordered
                                    // [u0,v0,u1,v1,u2,v2]
    double uv2_pts[6],              // out: defining locs for
                                    // simple shapes
                                    // ordered
                                    // [u0,v0,u1,v1,u2,v2]
    double& tang_gain,              // out: UNUSED
    int& ntgrl_degree,              // out: crv_cstrn
                                    // numerical integration
                                    // accuracy
    SDM_options* sdmo               // in:SDM_options pointer
        = NULL                      // note: sets the active
                                    // patch
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dmapinum.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description:

This function looks for the constraint identified by a tag in the entire patch hierarchy. When tag specifies a constraint, this function fills all the output arguments appropriately and returns 0. It changes the active deformable model to the one containing the constraint. Otherwise, it returns DM_TAG_OBJECT_ NOT_FOUND and leaves output arguments unmodified.

Link constraints, used for multi–surface deformations, represent two different constraint shapes which are connected together by the constraint. Link constraints place the information about their second constraint in the arguments: patch2_tag, shape2, src2_data, uv2_pts_count, and uv2_pts. When the identified constraint is not a link constraint, the function sets these arguments to unusable values as:

patch2_tag . . . . . –1
shape2 . . . . . “none”
src2_data . . . . . NULL
uv2_pts_count 0

The shape will be one of ”point”, ”straight”, ”parabola”, ”circ”, ”curve” or “area”. Depending on the shape type, some number of the uv_pts are set. The uv_pts define the shape of the constraint in the domain of the deformable model’s shape.

When domain_flag is set to 1, the uv domain points are returned specified in the unit–domain (values range from 0 to 1). When domain_flag is set to 0, the uv domain points are returned specified in the original dmod’s domain space. When domain_flag is set to 2, the uv domain points are returned in the internal_pfunc_space.

For “point” shapes, one uv_pt is set for the constraint’s location in the surface domain.

For “straight” shapes, two uv_pts are set for the end point locations of the straight constraint.

For “parabola” shapes, three uv_pts are set for the first end point, the intersection point between the two end tangents, and the last end point of the parabola constraint.

For “circ” shapes, three uv_pts are set for the center point, and two points on the circumference of the ellipse constraint.

For “curve” shapes, no uv_pts are set.

For “area” shapes, two uv_pts are set for the constraint’s upper and lower boundaries. For releases 5.3 and 6.0 the only supported area constraint is a rectangle.

The behavior argument is a bit array composed as an or of the following bit constants:

DM_POS_FREE	DM_POS_2_FREE
DM_POS_FIXED	DM_POS_2_FIXED
DM_POS_LINKED	DM_POS_2_LINKED
DM_TAN_FREE	DM_TAN_2_FREE
DM_TAN_FIXED	DM_TAN_2_FIXED
DM_TAN_LINKED	DM_TAN_2_LINKED
DM_CURV_FREE	DM_CURV_2_FREE
DM_CURV_FIXED	DM_CURV_2_FIXED
DM_CURV_LINKED	DM_CURV_2_LINKED
DM_NORM_FIXED	DM_BINORM_FIXED

ntgrl_degree is an internal term used by deformable modeling to control the accuracy of building the deformable modeling equations. It is the max_degree polynomial function that is accurately integrated by the package. If the value is too small the system will not deform as expected. If the value is too large the system slows down. The term is returned for debugging and informative purposes only, and its value should always be at least twice the degree value. The maximum value is 79.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_cstrn_behavior

Function: Deformable Modeling, DML Constraints

Action: Gets the constraint's pos/tang behavior and returns 0 for success or an error.

Prototype:
```
int DM_get_cstrn_behavior (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of dmod
                            // hierarchy to search
    int tag,                // in: cstrn identifier
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies behavior.

This function searches the deformable model hierarchy for a constraint with a given input tag value and loads the behavior with that constraint's behavior bits, which can be an orof the following bit constants:

DM_POS_FREE	DM_POS_2_FREE
DM_POS_FIXED	DM_POS_2_FIXED
DM_POS_LINKED	DM_POS_2_LINKED
DM_TAN_FREE	DM_TAN_2_FREE
DM_TAN_FIXED	DM_TAN_2_FIXED
DM_TAN_LINKED	DM_TAN_2_LINKED
DM_CURV_FREE	DM_CURV_2_FREE
DM_CURV_FIXED	DM_CURV_2_FIXED
DM_CURV_LINKED	DM_CURV_2_LINKED
DM_NORM_FIXED	DM_BINORM_FIXED

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_cstrn_rights

Function: Deformable Modeling, DML Constraints

Action: Gets a constraint's deletable/stoppable rights.

Prototype:

```
int DM_get_cstrn_rights (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of dmod
                            // hierarchy to search
    int tag,                // in: cstrn identifier
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies rights.

Searches the deformable model hierarchy for the constraint with a given input tag value and loads that constraint's rights as bits, which can be one or both of DM_DELETABLE and/or DM_STOPABLE. Constraints which are DM_DELETABLE may be removed from the deformable model hierarchy with a call to DM_rm_tag_object(). Constraints which are DM_STOPABLE may be toggled on and off by calls to DM_set_cstrn_state() and may have their DM_POS_FIXED behavior bit disabled with a call to DM_set_cstrn_behavior(). Constraints without the appropriate rights will not be acted upon by these functions.

Constraint's of different types are created with different rights (see DM_add_crv_()). These rights can not be changed at run time because the deformable modeling deformable model hierarchy depends on the continued existence of unstoppable and undeletable constraints. Control a constraint's rights by selecting the correct src_type value at construction time.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_cstrn_src_data

Function: Deformable Modeling, DML Constraints

Action: Returns the pass–through user data pointer stored with a constraint.

Prototype:

```
void* DM_get_cstrn_src_data (
    int& rtn_err,           // out: 0=success or neg
                            // err code
    DS_dmod* dmod,          // in: member of dmod
                            // hierarchy to search
    int tag,                // cstrn identifier
    int tgt                 // for link_cstrns,
        = 1,                // tgt = 1 or 2
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function returns the pass–through user data pointer stored with a constraint. This pointer is stored for the convenience of the calling application that may want to store local application–specific data with each constraint. The pointer is never accessed or used by the DM Modeler package.

Errors:

DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

DM_BAD_SRC_DATA_TGT_VALUE
The input tgt value must be 1 or 2.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_cstrn_src_pfuncs

Function: Deformable Modeling, DML Constraints

Action: Returns pointers to a curve or link constraint’s source DS_pfunc objects, which define the shape of the constraint.

Prototype:

```
void DM_get_cstrn_src_pfuncs (
    int& rtn_err,           // out: 0=success or neg
                            // err code
    DS_dmod* dmod,          // in: member of dmod
                            // hierarchy to search
    int tag,                // cstrn identifier
    int tgt,                // for link_cstrns,
                            // tgt = 1 or 2
    DS_pfunc*& src_W_pfunc, // out: cstrn pos shape,
                            // [nested]
    DS_pfunc*& src_Wn_pfunc,// out: cross tang shape,
                            // [nested]
    DS_pfunc*& src_Wnn_pfunc,// out: curvature shape,
                            // [nested]
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description:

This function returns pointers to the stored source DS_pfunc objects which can be used to specify the shape of a curve or link constraint. These functions are not required when constructing those constraints, in which case the pointers will be set to NULL. Refer to the functions, DM_add_crv_cstrn() and DM_add_link_cstrn(). These pointer values can be modified with the call DM_set_cstrn_src_pfuncs().

The tgt value specifies which set of src_pfunc values to return when the input tag value identifies a link_cstrn; either the src_pfuncs associated with the first or second of the link's DS_dmond objects.

All output values will be set to NULL when calling this function on a point or area constraint.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

DM_NOT_A_CRV_LINK_CSTRN
The input tag did not identify a curve or link constraint.

DM_BAD_SRC_DATA_TGT_VALUE
The input tgt value must be 1 or 2.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_cstrn_state

Function: Deformable Modeling, DML Constraints

Action: Gets the state for a constraint and returns a 0=success or an error.

Prototype:
```
int DM_get_cstrn_state (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of dmod
                            // hierarchy to search
    int tag,                // in: cstrn identifier
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Queries the tag constraint enabled/disabled behavior bit.

Returns 1 when the constraint is enabled, returns 0 when the constraint is disabled.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_cstrn_type_id

Function: Deformable Modeling, DML Constraints

Action: Gets the type identifier of a constraint's tangent gain.

Prototype:

```
void DM_get_cstrn_type_id (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of dmod
                            // hierarchy to search
    int tag,                // in: cstrn identifier
    DS_TAGS& type_id,       // out: one of
                            // DS_tag_pt_cstrn
                            // DS_tag_pt_seam
                            // DS_tag_crv_cstrn
                            // DS_tag_crv_seam
                            // DS_tag_link_cstrn
                            // DS_tag_area_cstrn
                            // DS_tag_link_load
                            // DS_tag_crv_load
    DS_CSTRN_SRC& src_type, // out: one of
                            // ds_solid_cstrn
                            // ds_bound_cstrn
                            // ds_user_cstrn
                            // ds_seam_cstrn
                            // ds_link_cstrn
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dmapinum.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies type_id, src_type.

Searches the deformable model hierarchy for a curve constraint with the given input tag value and loads type_id and src_type with that constraint type values. Valid type_ids include:

- DS_tag_pt_cstrn
- DS_tag_pt_seam
- DS_tag_crv_cstrn
- DS_tag_crv_seam
- DS_tag_link_cstrn

Valid src_types include:

- ds_solid_cstrn
- ds_bound_cstrn
- ds_user_cstrn
- ds_seam_cstrn
- ds_link_cstrn

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_cstrn_value

Function: Deformable Modeling, DML Constraints, DML Loads

Action: Gets the value for a point constraint's constraint behavior.

Prototype:

```
void DM_get_cstrn_value (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: member of target
                               // dmod hierarchy to
                               // search
    int tag,                   // in: pt_cstrn
                               // identifier
    int pt_index,              // in: index of tag's
                               // value to query
                               // one of:
                               // DM_BASE_PT
                               // DM_TANG_LEG
                               // DM_TANG1_LEG
                               // DM_TANG2_LEG
                               // DM_NORM_LEG
                               // DM_CURV1_LEG
                               // DM_CURV2_LEG
                               // DM_BINORM_LEG
    int& cstrn_val_count,      // out: size of cstrn_val
    double*& cstrn_val,        // out: tag's current
                               // cstrn_val
                               // sized:
                               // [cstrn_val_count]
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description:

This function gets the value for a point constraint's constraint behavior. Constraint values are all associated with a display point which can often be used for updating the constraint values in a more intuitive fashion. Refer to DM_get_pt_xyz() and DM_set_pt_xyz().

The input arguments dmod and tag identify which point constraint to query. The argument pt_index selects which of the point constraint's behavior values to return. Accepted values for pt_index include:

DM_BASE_PT . . . . . = set base point position. cstrn_val_count = Image_dim

DM_TANG_LEG or
DM_TANG1_LEG . . . . . = set tangent vector in the domain1_dir direction. cstrn_val_count = Image_dim

DM_TANG2_LEG . . . . . = set tangent vector in the domain2_dir direction. cstrn_val_count = Image_dim

DM_CURV_LEG or
DM_CURV1_LEG . . . . . = set curvature value in the domain1_dir direction. cstrn_val_count = 1

DM_CURV2_LEG . . . . . = set curvature value in the domain2_dir direction. cstrn_val_count = 1

DM_NORM_LEG . . . . . = set normal vector direction for a curve or surface. cstrn_val_count = Image_dim

DM_BINORM_LEG . . . . . = Set binormal vector direction for a curve. cstrn_val_count = Image_dim

The desired value for the specified constraint behavior is returned in the pointer, cstrn_val, which is a cstrn_val_count length array. The constraint behavior value will be an image space position, vector, or scalar value depending on which constraint behavior is being updated as indicated above.

Use DM_set_cstrn_value or DM_set_pt_xyz to modifying a constraint behavior value. These functions will cause the constraint's associated display point to be repositioned.

Errors:

DM_NO_ROOT_DMOD
The root deformable model cannot be NULL on input.

DM_TAG_OBJECT_NOT_FOUND
The input tag does not identify any tag object in the model hierarchy.

DM_NOT_A_PT_CSTRN
the input tag does not identify a point constraint in the model hierarchy.

DM_BAD_PT_INDEX_VALUE
When pt_index is not an appropriate value.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_curve_load

Function: Deformable Modeling, DML Loads

Action: Sets load data into return arguments and returns 0 for success or an error.

Prototype:

```
void DM_get_curve_load (
    int& rtn_err,            // out: 0=success
                             // or negative err code
    DS_dmod* dmod,           // in: member of target
                             // dmod hierarchy to
                             // search
    int tag,                 // in: load identifier
    DS_pfunc*& src_C_pfunc,  // out: domain space
                             // curve connected to
                             // image curve over
                             // common s values
                             // output in
                             // internal_pfunc_space.
    DS_pfunc*& src_W_pfunc,  // out: the image space
                             // curve
    double& gain,            // out: stiffness
                             // connecting the
                             // 2 curves
                             // note: the rtn
                             // src_C_pfunc ptrs
                             // and src_W_pfunc ptrs
                             // are still referenced
                             // within the load object
                             // – they are not copies
                             // Do not free or change
                             // the data within the
                             // structures without
                             // first copying them.
    SDM_options* sdmo        // in:SDM_options pointer
        = NULL               // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies src_C_pfunc, src_W_pfunc, gain.

Looks for load identified by a given tag in the entire patch hierarchy. When the tag specifies a DS_curve_load, this function fills all the output arguments appropriately and returns 0. Changes the active deformable model to the one containing the load. Although the src_C_pfunc was constructed in the containing the load. orig_dmod_space, it is stored internally so that the src_C_pfunc's image space is equal to the internal_pfunc_domain_space of the loaded deformable model.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_default_shape

Function: Deformable Modeling, Deformable Model Management

Action: Gets the deformable model's default shape values.

Prototype:
```
int DM_get_default_shape (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies shape_flag

Returns the deformable model's shape_flag state as enabled or disabled.

Deformable models change their shape to reduce their internal energy to a minimum. That is, they naturally deform to resist stretching and bending. When the default shape mechanism is disabled, all deformable models want to be flat and of zero area. That is, they will shrink to a single point if they are solved with absolutely no constraints or loads to force them to behave otherwise. This is fine for about half the modeling jobs where one can start with a simple shape and through sculpting with large deformations create much more complicated shapes. The default shape mechanism can change the behavior of shrinking to a flat zero–area shape. When enabled, the default_shape is the rest shape that the deformable model will find if it is solved with no loads or constraints. The default shape mechanism is good for modifying an existing shape by a smooth amount. To do this, first capture the initial shape as the default shape with the command DM_set_default_shape() and then sculpt as normal.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_delta

Function: Deformable Modeling, Deformable Model Management

Action: Gets the deformable model's delta values and returns 0 for success or an error.

Prototype:
```
double DM_get_delta (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies delta.

Returns the deformable model's resistance to displacement delta values. For both curves and surfaces there is only one value. Returns the deformable model's resistance to displacement delta values. For both curves and surfaces there is only one value.

The delta term was added as an experiment to approximate the behavior of the default shape mechanism before it was built. Since default shapes have been added to the system it is no longer necessary to use a non–zero value for delta when sculpting. The default shapes with delta equal to 0 are much more intuitive to use.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_dist_press

Function: Deformable Modeling, DML Loads

Action: Gets and places the place load data into return arguments.

Prototype:

```
void DM_get_dist_press (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of target
                            // dmod hierarchy to
                            // search
    int tag,                // in: load identifier
    int domain_flag,        // in: 0=domain_pts in
                            // orid_dmod_space,
                            // 1=domain_pts in
                            // unit_space.
                            // 2=domain_pts in
                            // internal_pfunc_space.
    double* domain_min,     // out: low bound,
                            // sized:[domain_dim]
    double* domain_max,     // out: top bound,
                            // sized:[domain_dim]
    double& gain,           // out: magnitude of
                            // load gain
    int& negate_flag,       // out: change normal dir
                            // (1=negate,0=do not)
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies domain_min, domain_max, gain, negate_flag.

Looks for a load identified by a given tag in the entire patch hierarchy. When the tag specifies a distributed pressure load, this function fills all the output arguments appropriately and changes the active deformable model to the one containing the load.

When domain_flag is set to 1, domain points are given in the unit range and mapped to the dmod's actual domain range by this function. When domain_flag is set to 0, domain points are given in the dmod's original domain range. When domain_flag is set to 2, domain points are given in the dmod's pfunc's internal domain range.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NULL_OUTPUT_PTR
The domain minimum or maximum is NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_dmods

Function: Deformable Modeling

Action: Gets the tags of all DS_dmods in the hierarchy containing the input DS_dmod.

Prototype:
```
void DM_get_dmods (
    int& rtn_err,           // error code
    DS_dmod* dmod,          // model to check
    int& ntags,             // number found
    DM_tag_array& tags,     // array of tags
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dm_tag_array.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function returns a DM_tag_array containing the tags of all DS_dmods in the input DS_dmod hierarchy. The array is sized:[tags.Size()]. rtn_err returns 0 for success or a negative error code.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_dmod_cstrn

Function: Deformable Modeling, DML Create and Query, DML Constraints

Action: Gets the deformable model's first constraint if it exists.

Prototype:
```
DS_cstrn* DM_get_dmod_cstrn (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dscstrn.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: When the deformable model has a constraint, the output constraint pointer is set to the first constraint object on the linked list of constraints.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_dmod_curve

Function: Deformable Modeling, DML Create and Query

Action: Gets deformable curve data.

Prototype:

```
void DM_get_dmod_curve (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to query
    double& domain_scale,   // out:
                            // internal_pfunc_space
                            // = domain_scale
                            // * orig_dmod_space.
    DS_pfunc*& pfunc,       // out: the dmod's
                            // shape model, output in
                            // internal_pfunc_space
    void*& dmod_entity,     // out: application ptr
                            // stored with dmod
    int& draw_state,        // out: draw bits for
                            // application graphics
                            // draw loads, cstrns,
                            // and seams)
    int& tag,               // out: unique id for
                            // this object
    double& alpha,          // out: resistance to
                            // stretch [1.0]
    double& beta,           // out: resistance to
                            // bending [5.0]
    double& gamma,          // out: resist bending
                            // change rate
    double& delta,          // out: resistance to
                            // moving from
                            // default shape
    double& dt,             // out: implicit
                            // integration time
                            // step [1.0]
    double& mass,           // out: physical param
                            // (causes rippling)
                            // [1.0]
    double& damp,           // out: physical param
                            // (stable integration)
                            // [5.0]
    DS_dmod*& parent,       // out: dmod's
                            // parent pointer
    DS_dmod*& sibling,      // out: dmod's
                            // sibling pointer
    DS_dmod*& child,        // out: dmod's 1st child
    int& load_count,        // out: number of loads
                            // in dmod
    int& cstrn_count,       // out: number of
```

```
                                    // constraints in dmod
    SDM_options* sdmo               // in:SDM_options pointer
        = NULL                      //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies pfunc, draw_state, tag, alpha, beta, delta, dt, mass, and damp.

Loads the output variables with data from the input deformable model object. Note that the current value of pfunc will be overwritten so take care that it is not the only pointer to a current DS_pfunc object. Otherwise, a memory leak may occur. See function DM_make_dmod_curve() for definitions of the output arguments and their use.

The pfunc is loaded with a pointer to the contained DS_pfunc object which exists in the internal_pfunc_space. The pfunc's domain space range is related to the DS_dmod's domain space range by the domain_scale factor as:

internal_pfunc_space = domain_scale * orig_dmod_space.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_DMOD_NOT_A_CURVE
The deformable model must be a curve.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_dmod_degree

Function: Deformable Modeling, DML Create and Query, Deformable Model Management

Action: Gets the deformable model's basis degree and returns 0 for success or an error.

Prototype:
```
void DM_get_dmod_degree (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to query
    int& u_degree,          // out: u_dir
                            // polynomial degree
    int& v_degree,          // out: v_dir
                            // polynomial degree
                            // (surfs only)
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies u_degree, v_degree.

Retrieves the polynomial degrees for the deformable shape's basis functions in the *u* and *v* directions. The v_degree value is not used for curves.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NOT_BSPLINE_OR_NURB
The deformable model must be a B–spline or a NURB in order to modify output variables.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_dmod_domain_max

Function: Deformable Modeling, DML Create and Query

Action: Gets the maximum domain point for a dmod and returns 0 for success or an error.

Prototype:

```
void DM_get_dmod_domain_max (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to query
    int domain_flag,        // in: 0=domain_max in
                            // orig_dmod_space,
                            // 1=domain_max in
                            // unit_space,
                            // 2=domain_max in
                            // internal_pfunc_space.
    double* domain_max,     // out: ptr to max_pt
                            // array,
                            // sized:[domain_dim]
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies domain_max.

Retrieves the dmod domain's maximum bounding point and stores it in domain_max.

The returned domain point location can be reported in either the dmod's domain range (the original range with which the dmod was created), the unit_range where the point is scaled to range from 0.0 to 1.0, or the pfunc's domain range(the actual value used internally). When the domain_flag is set to 0, domain_min is returned in orig_dmod_space, when the domain_flag is set to 1, domain_min is returned in unit_space, and when the domain_flag is set to 2, domain_min is returned in internal_pfunc_space.

Note *The* DS_dmod*'s domain range may be different from its contained* DS_pfunc*'s domain range. The* DS_dmod *domain range remains constant during a deformable modeling session while the* DS_pfunc *domain range may change. Initially they start out the same.*

Errors:

DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

DM_NULL_OUTPUT_PTR
The domain max is NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_dmod_domain_min

Function: Deformable Modeling, DML Create and Query

Action: Gets the minimum domain point for a dmod and returns 0 for success or an error.

Prototype:

```
void DM_get_dmod_domain_min (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: dmod to query
    int domain_flag,           // n: 0=domain_min in
                               // orig_dmod_space,
                               // 1=domain_min in
                               // unit_space,
                               // 2=domain_min in
                               // internal_pfunc_space.
    double* domain_min,        // out: ptr to max_pt
                               // array,
                               // sized:[domain_dim]
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies domain_min.

Retrieves the dmod's domain's minimum bounding point and stores it in domain_min.

The returned domain point location can be reported in either the dmod's domain range (the original range with which the dmod was created), the unit_range where the point is scaled to range from 0.0 to 1.0, or the pfunc's domain range(the actual value used internally). When the domain_flag is set to 0, domain_min is returned in orig_dmod_space, when the domain_flag is set to 1, domain_min is returned in unit_space, and when the domain_flag is set to 2, domain_min is returned in internal_pfunc_space.

Note *The* DS_dmod*'s domain range may be different from its contained* DS_pfunc*'s domain range. The* DS_dmod *domain range remains constant during a deformable modeling session while the* DS_pfunc *domain range may change. Initially they start out the same.*

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NULL_OUTPUT_PTR
The domain min cannot NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_dmod_domain_scale

Function: Deformable Modeling, DML Create and Query

Action: Returns the scale factor that relates the adjustable DS_pfunc domain range to the constant DS_dmod domain range.

Prototype:
```
double DM_get_dmod_domain_scale (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's domain scale factor.

The domain range of a DS_dmod deformable model is related to the domain range of the DS_dmod's contained DS_pfunc object by a simple scaling:

```
internal_pfunc_space = domain_scale *
orig_dmod_space.
```

At the time that a DS_pfunc is used to create a DS_dmod deformable modeling object (DM_make_dmod_surface, DM_make_dmod_curve, DM_add_curve_patch, DM_add_surface_patch) or when the domain of a deformable model is split (DM_split_dmod), the domain of the contained DS_pfunc may be scaled to optimize the numerical computation of high order derivative terms.

The domain scaling will occur when the minimum knot spacing becomes less than 0.3. This scaling is for internal use only and in most places in the deformable modeling interface input and output happen in the original domain range called the orig_dmod_space. Some query functions return pointers to the internal DS_pfunc data to allow direct access of the stored data. In these circumstances, the interface exposes the internally used internal_pfunc_space. Any application which uses those values will have to be aware of the scaling between the internal_pfunc_space and the orig_dmod_space.

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_dmod_knots

Function: Deformable Modeling, Deformable Model Management

Action: Writes the deformable models into output arrays and returns 0 for success or an error.

Prototype:

```
void DM_get_dmod_knots (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to query
    int& u_knot_count,      // out: u_dir unique
                            // knot count
    double*& u_knot,        // out: u_dir knot values
                            // in
                            // internal_pfunc_space
    int& v_knot_count,      // out: v_dir unique knot
                            // count (surfs only)
                            // in
                            // internal_pfunc_space
    double*& v_knot,        // out: v_dir knot values
                            // (surfs only)
                            // note: mallocs the
                            // output arrays which
                            // must be deleted by
                            // the caller
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies u_knot_count, u_knot, v_knot_count, and v_kno.

When the dmod–>Pfunc() is of type B–spline or NURB, this function loads the output with the number of knots and the knot values in the *u* and *v* directions. When the deformable model is a curve only the u_knot_count and the u_knot array are updated and v_knot_count is set to 0.

This function updates the output pointer values to point to the dmod–>Pfunc()'s nested data. Do not use these pointer values to free memory, nor after the dmod has been deleted.

Errors:

DM_NOT_BSPLINE_OR_NURB
The deformable model must be a B–spline or a NURB in order to modify output variables.

DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_dmod_load

Function: Deformable Modeling, DML Create and Query, DML Loads

Action: Gets the deformable model for a load and returns 0 for success or an error.

Prototype:
```
DS_load* DM_get_dmod_load (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dsload.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: When the deformable model has a load, this function sets the output load pointer to the first load object on the linked list of loads.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_dmod_pfunc

Function: Deformable Modeling, DML Create and Query

Action: Gets deformable model's pfunc and returns input's DS_pfunc shape pointer.

Prototype:
```
DS_pfunc* DM_get_dmod_pfunc (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the ds_pfunc pointer contained within the input deformable model. The contained DS_pfunc object's domain defines internal_pfunc_space. This space is related to the pfunc's original domain range by a domain_scale factor as:

internal_pfunc_space = domain_scale * orig_dmod_space.

The domain_scale factor may be queried by the function, DM_get_dmod_domain_scale().

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_dmod_surface

Function: Deformable Modeling

Action: Gets deformable surface data.

Prototype:
```
void DM_get_dmod_surface (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to retrieve
                            // data from
```

```
void*& dmod_entity,      // out: application ptr
                         // stored with dmod
double& domain_scale,    // out:
                         // internal_pfunc_space
                         // = domain_scale
                         // * orig_dmod_space.
DS_pfunc*& pfunc,        // out: the dmod's
                         // shape model
                         // output in
                         // internal_pfunc_space
int& draw_state,         // out: draw bits for
                         // application graphics
                         // (draw loads, cstrns,
                         // and seams)
int& tag,                // out: unique id for
                         // this object
double& au,              // out: u dir resistance
                         // to stretch [1.0]
double& av,              // out: v dir resistance
                         // to stretch [1.0]
double& atheta,          // out: resist stretch
                         // rotation angle
double& bu,              // out: u dir resistance
                         // to bending [5.0]
double& bv,              // out: v dir resistance
                         // to bending [5.0]
double& btheta,          // out: resist bending
                         // rotation angle
double& gamma,           // out: resist bending
                         // change rate
double& delta,           //out: resistance to
                         // moving from default
                         // shape
double& dt,              // out: implicit
                         // integration time step
                         // [1.0]
double& mass,            // out: physical param
                         // (causes rippling)
                         // [1.0]
double& damp,            // out: physical param
                         // (stable integration)
                         // [5.0]
DS_dmod*& parent,        // out: dmod's parent
                         // pointer
DS_dmod*& sibling,       // out: dmod's sibling
```

```
                                    // pointer
    DS_dmod*& child,                // out: dmod's 1st child
    int& load_count,                // out: number of loads
                                    // in dmod
    int& cstrn_count,               // out: number of
                                    // constraints in dmod
    SDM_options* sdmo               // in:SDM_options pointer
        = NULL                      //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies pfunc, draw_state, tag, au, av, atheta, bu, bv, btheta, delta, dt, mass, and damp

Loads the output variables with data from the input deformable model object. Note the current value of pfunc will be overwritten so take care that it is not the only pointer to a current DS_pfunc object, otherwise a memory leak may occur. See function DM_make_dmod_surface() for definitions of the output arguments and their use.

The pfunc is loaded with a pointer to the contained DS_pfunc object which exists in the internal_pfunc_space. The pfunc's domain space range is related to the DS_dmod's domain space range by the domain_scale factor as:

internal_pfunc_space = domain_scale * orig_dmod_space.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_DMOD_NOT_A_SURFACE
The deformable model must be a surface.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_dmod_tag

Function: Deformable Modeling

Action: Gets the type for DS_pfunc and returns the deformable model's tag number or an error.

Prototype:
```
int DM_get_dmod_tag (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the tag number for the input deformable model.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_dmod_tags

Function: Deformable Modeling

Action: Gets the tags of all tag objects on a single DS_dmod.

Prototype:
```
void DM_get_dmod_tags (
    int& rtn_err,           // error code
    DS_dmod* dmod,          // model to check
    int& ntags,             // number of tags
    DM_tag_array& tags,     // array of tags
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dm_tag_array.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function returns a DM_tag_array containing the tags of all tag objects embedded in the input DS_dmod. The array is sized:[tags.Size()]. rtn_err returns 0 for success or a negative error code.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_dmod_type_id

Function: Deformable Modeling, DML Create and Query

Action: Gets the tag type for DS_pfunc and returns 0 for success or an error.

Prototype:
```
DS_TAGS DM_get_dmod_type_id (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dmapinum.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the input deformable model's type identifier. The type identifier is one of DS_tag_srf_dmod or DS_tag_crv_dmod.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_dof_state

Function: Deformable Modeling, Deformable Model Management

Action: Gets the deformable model's dof_state values and returns 0 for success or an error.

Prototype:

```
void DM_get_dof_state (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: dmod to be queried
    int& dof_count,            // out: total surface
                               // dof count
    int& free_count,           // out: number of free
                               // dofs to deform
    int& cstrn_count,          // out: total number of
                               // constraint equations
    int& fixed_count,          // out: independent
                               // constraint equations
                               // cnt
    int& lambda_count,         // out: lambda constraint
                               // eqn count
    int& lfixed_count,         // out: independent
                               // lambda constraint
                               // count
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
                               // note: use after a call
                               // to DM_solve()
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies dof_count, free_count, _count, fixed_count lambda_count, lfixed_count.

Returns the deformable model's dof_state. The term "dof" stands for Degree of Freedom. The deformable model resolves down to a set of simultaneous equations that must be solved. This system of equations has a total number of degrees of freedom which is just the number of control points for B–spline and NURB surfaces. All this means is that all the control points are degrees of freedom that can be moved by the deformable modeling algorithm. Each constraint applied to the system nails down or removes one or more degrees of freedom from the system. The remaining number of degrees of freedom, called the free_count are all the degrees of freedom that the system has left to deform the shape of the model. When the free_count gets too low because several constraints have been added to the system or because the end user is trying to deform a degree 0 B–spline, then the deformable model will not deform in a natural manner. The application should detect these conditions by examining these values after a call DS_solve(). When the free_count gets low (less than 20% of the total dofs) then consider adding more dofs to the system with calls to DS_elevate_dmod_degree() and DS_split_ deformable model(). cstrn_count tells how many total constraint equations may be in effect. Often times this number is larger than the total dof count. The fixed_count is the number of independent constraint equations. The total_count will always equal the sum of the free_count and the fixed_count. A B–spline control point is not really one dof but three, since it can move in three dimensional space. Some constraints, like the surface normal constraint, cannot be written with one equation for one control point. Instead, such constraints can be written with one equation for each coordinate of the control point. These constraints are enforced using lagrange multipliers. lambda_count is the number of lagrange multiplier constraint equations, and one fixed_count is the number of independent lagrange multiplier constraint equations.

When adding degrees of freedom to a system we first bump up the degree of the system to three when no patches are used and to four when patches are present with DM_elevate_dmod_degree. Then we split the deformable model into a larger number of elements making sure each element is the same size as the rest with a call to DM_split_dmod().

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_domain_dim

Function: Deformable Modeling, Deformable Model Management

Action: Gets the domain space dimension for a deformable model.

Prototype:
```
int DM_get_domain_dim (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's domain dimension. Valid return values are 1 for a curve parameterized in *u* and 2 for a surface parameterized in *u* and *v*.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_draw_count

Function: Deformable Modeling, DML Graphics

Action: Gets the deformable model's mesh graphics parameters.

Prototype:

```
void DM_get_draw_count (
    int& rtn_err,           // out: 0=success or neg
                            // err code
    DS_dmod* dmod,          // in: dmod to query
    int& handle_count,      // patch graphic
                            // object count
    int& shape_count,       // surf mesh count
                            // in patch hierarchy
    int walk_flag           // specify how deep
        = 0,                // to go 0=dmod only,
                            // 1=dmod and offspring
                            // 2=dmod, siblings, and
                            // offspring
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function stores in handle_count, the number of patch graphic objects being displayed for the target deformable model and possibly its offspring. It stores in shape_count the number of surf_mesh objects being shown in the queried hierarchy. The current list of patch graphic objects includes:

DM_DRAW_CPTS	look for control points
DM_DRAW_GAUSS_PTS	look for vector norms
DM_DRAW_CSTRN_COMBS	look for curve cstrn curv combs
DM_DRAW_SEG_BNDS	look for seg_bnd points
DM_DRAW_CURVE_COMBS	look for curve curvature combs
DM_DRAW_ELEMS	look for elem boundaries

Within each deformable model only those graphic objects whose associated state bit is set in the draw state bit array are counted. Use DM_set_draw_state() to set the draw state bit array.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_draw_state

Function: Deformable Modeling, DML Graphics

Action: Gets the deformable model's draw state and returns an integer bit array.

Prototype:
```
int DM_get_draw_state (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's draw state integer which can be used as a bit array to control the rendering of deformable model data. A suggested set of draw bits is given in the file dmapi.hxx as the DM_DRAW_... environment variable defines. These draw bits are used to cull tag objects from consideration by the function DM_find_tag_by_image_line() which finds and returns the closest tag object to an image space line.

Returns integer draw bit array for deformable model.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_dynamics

Function: Deformable Modeling, Deformable Model Management

Action: Gets the deformable model's dynamic values and returns 0 for success or an error.

Prototype:

```
void DM_get_dynamics (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    double& dt,             // out: dynamic sculpting
                            // time step size
    double& mass,           // out: dynamic
                            // deformable mass value
    double& damp,           // out: dynamic
                            // deformable damping
                            // value
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies dt, mass, damp.

Deformable modeling is a simulation that mimics the behavior of actual physical systems. Each call to DM_solve() increments sculpting time by one time step. The deformable model equations are the equations of motion for a physical system. The internal energy, loads, and constraints define the stiffness for that system of equations. The mass and damp terms add an effective mass and damping to those equations of motion. A system with damping may take several time steps to reach an equilibrium position. A system with mass tends to overshoot and oscillate back and forth across the equilibrium position. Using dynamics one can create a very convincing animation in which each time step is one frame of the animation sequence. Changes in loads and constraints during the animation can create very "real" behaviors. For modeling purposes where the end user is only interested in defining the shape of an object, the dynamics feature may be of little interest. In these cases, both mass and damp should be set to 0.0, which turns the dynamic behavior off and causes DM_solve() to solve directly for the equilibrium position. For animation purposes where the end user is concerned with how an object moves from one position to another, however, this is a very useful tool.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_elem_count

Function: Deformable Modeling, Deformable Model Management

Action: Gets the deformable model's elem_count values and returns 0 for success or an error.

Prototype:
```
void DM_get_elem_count (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: dmod to be queried
    int& elem_count,           // out: total dmod elem
                               // count
    int& u_span_count,         // out: span count in
                               // u_direction
    int& v_span_count,         // out: span count in
                               // v_direction (only set
                               // for surfaces)
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies elem_count, u_span_count, v_span_count.

Returns the deformable model's elem_count. Places the deformable model's total elem_count in elem_count. For tensor product deformable models (like B–splines and NURBs) places the number *u* direction spans in the u_span_count and the number of v–direction spans in the v_span_count. v_span_count is not set when the input deformable model is a deformable curve.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_end_conds

Function: Deformable Modeling, DML Constraints

Action: Gets end states for target deformable model and returns 0 for success or an error.

Prototype:

```
void DM_get_end_conds (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: dmod to query
    int& end_cond_u,           // out: one of 0=open or
                               // 1=closed or 2=periodic
    int& singular_u,           // out: one of 0=none or
                               // 1=low or 2=high or
                               // 3=both
    int& end_cond_v,           // out: one of 0=open or
                               // 1=closed or 2=periodic
    int& singular_v,           // out: one of 0=none or
                               // 1=low or 2=high or
                               // 3=both
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
                               // note: only end_cond_u
                               // set by 1d curves all
                               // other values set to
                               // –1.
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”dshusk/dskernel/dmapi.hxx”
#include ”dshusk/dskernel/dsdmod.hxx”
#include ”dshusk/dskernel/sdm_options.hxx”
```

Description: Modifies end_cond_u, singular_u, end_cond_v, singular_v.

Loads input arguments with the target deformable model’s end condition values.

end_cond:
0 = open, no affect on surface
1 = closed, end cpts are constrained to be the same
2 = periodic, end tangents are constrained to be the same

singular:
0 = none, no affect
1 = low, all cpts on low boundary are constrained to be the same
2 = high, all cpts on high boundary are constrained to be the same
3 = both, all cpts on low boundary are constrained to be the same and all cpts on high boundary are constrained to be the same

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_entity

Function: Deformable Modeling, Deformable Model Management

Action: Gets the deformable model's entity pointer value and returns 0 for success or an error.

Prototype:
```
void* DM_get_entity (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's application entity pointer value.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_gamma

Function: Deformable Modeling

Action: Gets the deformable model's gamma values and returns 0 for success or an error.

Prototype:
```
double DM_get_gamma (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's resistance to the rate of change of bending value. For both curves and surfaces there is only one value.

The gamma term was added as an experiment to smooth out a deformable shape's response to curvature constraints. Without gamma a point curvature constraint will only change the curvature of a deformable model shape at the point of the constraint. We desire a smooth blending of curvature between the constraint and the rest of the shape. This blending is provided by gamma. When gamma equals 0 the system behaves as if the new term did not exist. When using curvature constraints or C2 patch seams make sure to use gamma to smooth out the change of curvature.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_epsilon

Function: Deformable Modeling

Action: Gets the deformable model's epsilon value and returns 0 or an error.

Prototype:
```
void DM_get_epsilon (
    int& rtn_err,           // return code
                            // 0=success, neg=error
    const DS_dmod* dmod,    // tag object identifier
    int tag,                // tag object identifier
    double& eps,            // epsilon
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "kernel/acis.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Epsilon regulates a shape fairing (energy minimization) term that is used to dampen control point oscillations in high degree splines (degree>8).

Like the primary fairing terms, alpha and beta, epsilon should be 0 or positive. Epsilon is considered a supplement to the alpha and beta shape fairing terms, and should be relatively small compared to beta, the chief shape fairing term.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_NOT_A_DMOD_MEMBER
The tag_flag does not identify a deformable model in the patch hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_icon

Function: Deformable Modeling, DML Graphics

Action: Gets the icon associated with a tag object and returns 0 for success or an error.

Prototype:
```
DM_icon* DM_get_icon (
    int& rtn_err,           // error code
    DS_dmod* dmod,          // model to check
    int tag,                // tag object owning icon
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dmicon.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Refer to Action.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
Could not find tag object with the given tag.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_image_dim

Function: Deformable Modeling, Deformable Model Management, DML Graphics

Action: Gets the image space dimension size and returns the deformable model's image dimension or an error.

Prototype:
```
int DM_get_image_dim (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's image dimension which can be any positive integer but will generally be 2 for curves in two dimensional space or 3 for curves and surfaces in three dimensional space.

Returns a positive dimension number when successful.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_integral_degree

Function: Deformable Modeling, Deformable Model Management

Action: Gets the deformable model's integral degree.

Prototype:
```
int DM_get_integral_degree (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's integral degree integer which is stored as part of the deformable model's–>Pfunc() data. This number is used to set the number of gauss points used to integrate the deformable modeling equations of motion. The integral degree is the degree polynomial that would be exactly integrated by the number of gauss points used in the gauss quadrature.

Returns positive number integral degree when successful.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_interior_state

Function: Deformable Modeling, Deformable Model Management

Action: Gets the deformable model's interior state value.

Prototype:
```
int DM_get_interior_state (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's interior state value.

Return value:
0 = allow C0 between elements
1 = enforce C1 between elements

When interior_state == 0, the Deformable model is allowed to bend with C0 discontinuity between elements. For B–splines and NURBs, C0 discontinuity within a surface can be achieved by increasing the knot count at an internal knot boundary.

When interior_state == 1, the Deformable model prohibits C0 bending between elements by adding C1 internal tangent constraints between any elements whose internal representation will allow a C0 bend. For B–splines and NURBs, this means that the deformation will be at least C1 everywhere, even if the underlying representation has multiple knots that would normally allow a C0 internal bend.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_load_gain

Function: Deformable Modeling, DML Loads

Action: Gets the load's gain and returns 0 for success or an error.

Prototype:
```
double DM_get_load_gain (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of target
                            // dmod hierarchy to
                            // search
    int tag,                // in: load identifier
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies load_gain.

Gets the load's gain value, sets the containing deformable model as the active deformable model and returns 0 when the tag identifier recognizes a load object. Otherwise, returns DM_TAG_OBJECT_NOT_FOUND and leaves output arguments unmodified.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_load_type_id

Function: Deformable Modeling, DML Loads

Action: Gets type identifier for a load.

Prototype:
```
DS_TAGS DM_get_load_type_id (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of dmod
                            // hierarchy to search
    int tag,                // in: load identifier
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dmapinum.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies type_id, src_type.

This function searches the deformable model hierarchy for a load with given input tag value and returns the type identifier of that load. The valid type identifier is one of the following:

DS_tag_pt_press	point pressure
DS_tag_dist_press	distributed press
DS_tag_dyn_load	dynamic load
DS_tag_spring	spring load
DS_tag_spring_se	spring set
DS_tag_crv_load	curve load

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_mesh_count

Function: Deformable Modeling, DML Graphics

Action: Gets the deformable model's mesh graphics parameters and returns 0 for success or an error.

Prototype:
```
void DM_get_mesh_count (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    int& mesh_u,            // out: u_dir polygon
                            // count (crvs & srfs)
    int& mesh_v,            // out: v_dir polygon
                            // count (srfs only)
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's mesh plot parameters. A good way to render a deformable model is with a set of regularly sized polygons. Each time the end user changes the shape of the deformable model the locations of the mesh's vertices may be updated and rendered. If one avoids having to remesh then the rendering may be done quickly enough to support interactive sculpting. Since the end user does not know in advance what the shape of the deformable may be, any geometrically based meshing would be a mistake. This package stores a polygon count in the *u* and *v* directions that control the mesh resolution for plotting. This library does not use those values but makes sure they are available for a rendering module. The mesh_v parameter is not used for curves.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_nurb_curve

Function: Deformable Modeling, DML Create and Query

Action: Gets NURB curve data and returns 0 for success or DM_NOT_A_NURB.

Prototype:

```
void DM_get_nurb_curve (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_pfunc* pfunc,        // in: pfunc data
                            // structure to examine
    int& image_dim,         // out: image space size
                            // (2=2d,3=3d)
    int& degree,            // out: polynomial degree
    int& dof_count,         // out: control point
                            // count
    int& knot_count,        // out: number of
                            // distinct knot values
    int*& knot_index,       // out: multiple knot
                            // count array.
                            // sized:[knot_count]
                            // specify the multiple
                            // knots by setting
                            // count[i] = maximum
                            // knot index value for
                            // location knot[i]
                            // Examples:
                            // no multiples count =
                            // [0,1,2,3,4]
                            // some multiples =
                            // [1,2,4]
    double*& knot,          // out: knot value array
                            // sized:[knot_count]
                            // ordered [u0<u1<u2...]
                            // output in
                            // internal_pfunc_space
    double*& dof_vec,       // out: dof_vec vals
```

```
                                // (control pt locs)
                                // sized:[image_dim*
                                // dof_count]
                                // ordered:[xyz0, xyz1,
                                // xyz2, ...]
double*& dof_def,               // out: default shape
                                // (control pt locs)
                                // sized:[image_dim*
                                // dof_count]
                                // ordered:[xyz0, xyz1,
                                // xyz2, ...]
double*& weight,                // in: weight vals for
                                // each control pt
                                // sized:[dof_count]
                                // ordered:[w0, w1,
                                // ... wn]
int& end_cond,                  // out: enforce
                                // continuity between end
                                // points with
                                // constraints. If used
                                // must have multiple
                                // knots on the ends of
                                // the B-spline.
                                // 0=open (no
                                // constraints)
                                // 1=closed (C0
                                // continuity)
                                // 2=periodic (C1
                                // continuity)
                                // default value = 0
int& ntgrl_degree,              // out: for deformable
                                // modeling - the
                                // degree polynomial
                                // exactly integrated
                                // should be larger than
                                // the degree. 10 is a
                                // good value. default
                                // value = 10
SDM_options* sdmo               // in:SDM_options pointer
    = NULL                      //
                                //  note:total_knot_count
                                // =dof_count+degree-1
                                // the total and distinct
                                // knot counts vary when
                                // there are multiple
```

```
                                        // knots. note: any input
                                        // array set to NULL
                                        // will be ignored by
                                        // subroutine.
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies image_dim, degree, dof_count, knot_count, knot_index, knot, dof_vec, dof_def, weight, end_cond, ntgrl_degree.

Updates the output pointer values to point to the pfunc's nested data. Do not use these pointer values to free memory, nor after the pfunc has been deleted.

ntgrl_degree is an internal term used by the deformable modeling package to control the accuracy of building the deformable modeling equations. It is the max_degree polynomial function that is accurately integrated by the package. If the value is too small the system will not deform as expected. If the value is too large the system slows down. The term is returned for debugging and informative purposes only, and its value should always be at least twice the degree value. The maximum value is 79.

The knot values returned in the knot array define the DS_pfunc's domain space which is called the internal_pfunc_space. When used in a deformable model, the internal_pfunc_space may be scaled by a constant factor to optimize the numerical accuracy of computing high order derivatives. The domain space of the deformable model is called the orig_dmod_space, and it remains constant for input and output purposes. The relationship between the two spaces is always just, internal_pfunc_space = domain_scale * orig_dmod_space.

The domain_scale factor may be retrieved with the function, DM_get_dmod_domain_scale().

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NOT_A_NURB
The input pfunc type must be a NURB curve.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_nurb_surface

Function: Deformable Modeling, DML Create and Query

Action: Gets NURB surface data and returns 0 for success or an error.

Prototype:

```
void DM_get_nurb_surface (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_pfunc* pfunc,           // in: pfunc data
                               // structure to examine
    int& image_dim,            // out: image space size
                               // (2=2d,3=3d)
    int& degree_u,             // out: u dir polynomial
                               // degree
    int& dof_count_u,          // out: u dir control
                               // point count
    int& knot_count_u,         // out: u dir number of
                               // distinct knot values
    int*& knot_index_u,        // out: u dir multiple
                               // knot count array.
                               // sized:[knot_count]
                               // specify the multiple
                               // knots by setting
                               // count[i] = maximum
                               // knot index value for
                               // location knot[i]
                               // Examples: no multiples
                               // count = [0,1,2,3,4]
                               // some multiples =
                               // [1,2,4]
    double*& knot_u,           // out: u dir knot values
                               // sized:[knot_count]
                               // ordered [u0<u1<u2...]
                               // output in
                               // internal_pfunc_space
    int& degree_v,             // out: v dir polynomial
                               // degree
    int& dof_count_v,          // out: v dir control
                               // point count
    int& knot_count_v,         // out: v dir number of
```

```
                                    // distinct knot values
int*& knot_index_v,                 // out: v dir multiple
                                    // knot count array.
                                    // sized:[knot_count]
                                    // specify the multiple
                                    // knots by setting
                                    // count[i] = maximum
                                    // knot index value for
                                    // location knot[i]
                                    // Examples: no multiples
                                    // count = [0,1,2,3,4]
                                    // some multiples =
                                    // [1,2,4]
double*& knot_v,                    // out: v dir knot values
                                    // sized:[knot_count]
                                    // ordered [u0<u1<u2...],
                                    // output in
                                    // current_pfunc_space
double*& dof_vec,                   // out: dof_vec vals
                                    // (control pt locs)
                                    // sized:[image_dim*
                                    // dof_count]
                                    // ordered:[xyz0, xyz1,
                                    // xyz2, ...]
double*& dof_def,                   // out: default shape
                                    // (control pt locs)
                                    // sized:[image_dim*
                                    // dof_count]
                                    // ordered:[xyz0, xyz1,
                                    // xyz2, ...]
double*& weight,                    // out: init weight for
                                    // each control pt
                                    // sized:[dof_count_u*
                                    // dof_count_v]
                                    // ordered:[w00, w01,
                                    // ... w0m, ... wnm]
int& end_cond_u,                    // out: one of
                                    // 0=open or 1=closed or
                                    // 2=periodic
int& singular_u,                    // out: one of 0=none or
                                    // 1=low or 2=high or
                                    // 3=both
int& end_cond_v,                    // out: one of 0=open or
                                    // 1=closed or 2=periodic
int& singular_v,                    // out: one of 0=none or
```

```
                                // 1=low or 2=high or
                                // 3=both
    int& ntgrl_degree,          // out: for deformable
                                // modeling – the degree
                                // polynomial exactly
                                // integrated should be
                                // larger than the
                                // degree. 10 is a good
                                // value. default value =
                                // 10
    SDM_options* sdmo           // in:SDM_options pointer
        = NULL                  //
                                // note: total_knot_count
                                // = dof_count + degree
                                // – 1 the total and
                                // distinct knot counts
                                // vary when there are
                                // multiple knots. note:
                                // any input array set to
                                // NULL will be ignored
                                // by subroutine.
    );
```

Includes:

```
#include ″kernel/acis.hxx″
#include ″dshusk/dskernel/dmapi.hxx″
#include ″dshusk/dskernel/dspfunc.hxx″
#include ″dshusk/dskernel/sdm_options.hxx″
```

Description: Modifies image_dim, knot, dof_def, ntgrl_degree, degree_u,dof_count_u,knot_count_u,knot_index_u,knot_u, degree_v,dof_count_v,knot_count_v,knot_index_v,knot_v, end_cond_u, singular_u, end_cond_v, singular_v

Updates the output pointer values to point to the pfunc's nested data. Do not use these pointer values to free memory, nor after the pfunc has been deleted.

ntgrl_degree is an internal term used by the deformable modeling package to control the accuracy of building the deformable modeling equations. It is the max_degree polynomial function that is accurately integrated by the package. If the value is too small the system will not deform as expected. If the value is too large the system slows down. The term is returned for debugging and informative purposes only, and its value should always be at least twice the degree value. The maximum value is 79.

The knot values returned in the knot arrays define the DS_pfunc's domain space which is called the internal_pfunc_space. When used in a deformable model, the internal_pfunc_space may be scaled by a constant factor to optimize the numerical accuracy of computing high order derivatives. The domain space of the deformable model is called the orig_dmod_space, and it remains constant for input and output purposes. The relationship between the two spaces is always just,

internal_pfunc_space = domain_scale * orig_dmod_space.

The domain_scale factor may be retrieved with the function, DM_get_dmod_domain_scale().

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NOT_A_NURB
The input pfunc type must be a NURB curve.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_parent

Function: Deformable Modeling, DML Patches

Action: Gets the deformable model's parent and returns the parent pointer or NULL.

Prototype:
```
DS_dmod* DM_get_parent (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's parent pointer.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_patch_continuity

Function: Deformable Modeling, DML Patches

Action: Gets the patch–to–parent continuity and returns 0 for C0 (position continuity) or 1 for C1 (pos and tang continuity).

Prototype:
```
int DM_get_patch_continuity (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: the dmod to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
                            // note: may return
                            // negative err code
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's current patch–to–parent continuity. Patches may be connected to their parents with either C0 (position) or C1 (tangent) continuity.

Returns 0 ..... = C0 positional continuity
Returns 1 ..... = C1 positional and tangent continuity.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_patch_seam_count

Function: Deformable Modeling, DML Patches

Action: Gets the number of seam curve.

Prototype:
```
int DM_get_patch_seam_count (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the number of curve constraints that are connecting this deformable model patch to its parent.

Returns 0 or positive number for success

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_patch_seam_tag

Function: Deformable Modeling, DML Patches, DML Tags

Action: Gets the tag number for a seam curve and returns the seam curve tag or an error.

Prototype:
```
int DM_get_patch_seam_tag (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    int seam_number,        // in: seam number to
                            // be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the number of curve constraints that are connecting this deformable model patch to its parent.

Returns 0 or positive number for success

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_BAD_SEAM_NUMBER_VALUE
The seam number cannot be less than 0, nor equal to or larger than the deformable model's seam count.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_pfunc_default_state

Function: Deformable Modeling, DML Create and Query

Action: Gets the pfunc's default state value.

Prototype:
```
int DM_get_pfunc_default_state (
    int& rtn_err,           // out: 0=success or neg
                            // err code
    DS_pfunc* pfunc,        // in: pfunc to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function gets the pfunc default state value. A value of 0 means that the pfunc is not using a default shape. A value of 1 means that it is.

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_pfunc_degrees

Function: Deformable Modeling, DML Create and Query

Action: Gets degree of freedom or dof values.

Prototype:
```
void DM_get_pfunc_degrees (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_pfunc* pfunc,        // in: pfunc to query
    DS_PFN& type_id,        // out:
                            // ds_tp1(B–spline crv),
                            // ds_tp2(B–spline srf),
                            // ds_rp1(NURB crv),
                            // ds_rp2(NURB srf)
                            // ds_cir(Circ crv)
    int& degree_u,          // out: u dir degree
    int& degree_v,          // out: v dir degree
                            // (not set by curves)
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/dmapinum.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function queries the input pfunc and loads the output variables type_id, degree_u, and degree_v. It returns zero for success or a negative error code. degree_u is set with the polynomial degree of the *u* direction basis functions. Surfaces set degree_v with the polynomial degree of the *v* direction basis functions. Only surfaces set the value of degree_v. When type_id equals ds_cir, degree_u is set to 10, which is considered large. The real number is infinite.

type_id is set to ds_tp1 (B–spline crv), ds_tp2(B–spline srf), ds_rp1(NURB crv), ds_rp2(NURB srf), ds_cir(circ crv)

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_pfunc_dofs

Function: Deformable Modeling, DML Create and Query

Action: Gets the degrees of freedom value of a pfunc and returns 0 for success or an error.

Prototype:

```
void DM_get_pfunc_dofs (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_pfunc* pfunc,           // in: pfunc to query
    DS_PFN& type_id,           // out: ds_tp1(B-spline
                               // crv), ds_tp2(B-spline
                               // srf) ds_rp1(NURB
                               // crv), ds_rp2(NURB srf)
                               // ds_cir(Circ crv)
    int& image_dim,            // out: coordinate count
                               // for each dof value
    int& dof_count_u,          // out: u dir dof count
    int& dof_count_v,          // out: v dir dof count
                               // (set to 1 by curvs)
    double*& dof_vec,          // out: array of dof
                               // values
                               // sized:[image_dim*
                               // dof_count]
                               // ordered:[xyz0, xyz1,
                               // xyz2, ...]
    double*& dof_def,          // out: array of default
                               // dof values [nested]
                               // sized:[dof_count_u*
                               // dof_count_v*DM_get
                               // _pfunc_image_dim
                               // (rtn_err,pfunc)]
                               // ordered:[xyz0,xyz1,
                               // xyz2...]
    double*& weight,           // out: NURB weight
                               // values or NULL
                               // [nested]
                               // sized:[dof_count_u*
                               // dof_count_v]
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
                               // note: return ptr
                               // values point to
                               // memory within the
                               // object. Do not free
                               // these pointers.
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/dmapinum.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies dof_count_u, dof_count_v, dof_vec, weight.

Queries the input pfunc and loads the output variables. Curves set the value dof_count_v to 1. So that the total dof count can always be computed by dof_count_u * dof_count_v. non–NURB curves and surfaces set the weight parameter to NULL. type_id is set to ds_tp1 (B–spline crv), ds_tp2(B–spline srf), ds_rp1(NURB crv), ds_rp2(NURB srf), ds_cir(circ crv)

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_pfunc_dof_count

Function: Deformable Modeling, DML Create and Query

Action: Returns the pfunc's dof_count value.

Prototype:
```
int DM_get_pfunc_dof_count (
    int& rtn_err,           // out: 0=success or neg
                            // err code
    DS_pfunc* pfunc,        // in: pfunc to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns pfunc's dof_count (the number of control points for a NURB or B–spline curve or surface).

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_pfunc_domain_dim

Function: Deformable Modeling, DML Create and Query

Action: Gets the pfunc's domain dimension.

Prototype:
```
int DM_get_pfunc_domain_dim (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_pfunc* pfunc,        // in: pfunc to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Refer to Action.

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_pfunc_domain_max

Function: Deformable Modeling, DML Create and Query

Action: Gets the maximum domain point for a pfunc and returns 0 for success or an error.

Prototype:

```
void DM_get_pfunc_domain_max (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_pfunc* pfunc,        // in: pfunc to query
    int domain_flag,        // in: 2=domain_pts in
                            // internal_pfunc_space,
                            // 1=domain_pts in
                            // unit_space.
    double* domain_max,     // out: ptr to max_pt
                            // array,
                            // sized:[domain_dim]
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies domain_max.

Retrieves the pfunc domain's maximum bounding point and stores it in domain_max.

The returned domain point location can be reported in either the pfunc's internal domain range (the actual value used internally) or it can be scaled to the unit range (from 0.0 to 1.0) for convenience. When the domain_flag is set to 2 an absolute domain point value is returned. When domain_flag is set to 1 a scaled domain point value is returned.

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

DM_NULL_OUTPUT_PTR
The domain max is NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_pfunc_domain_min

Function: Deformable Modeling, DML Create and Query

Action: Gets the minimum domain point for a pfunc and returns 0 for success or an error.

Prototype:
```
void DM_get_pfunc_domain_min (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_pfunc* pfunc,        // in: pfunc to query
    int domain_flag,        // in: 2=domain_pts in
                            // internal_pfunc_space,
                            // 1=domain_pts in
                            // unit_space.
    double* domain_min,     // out: ptr to max_pt
                            // array,
                            // sized:[domain_dim]
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies domain_min.

Retrieves the pfunc domain's minimum bounding point and stores it in domain_min.

The returned domain point location can be reported in either the pfunc's internal domain range (the actual value used internally) or it can be scaled to the unit range (from 0.0 to 1.0) for convenience. When the domain flag is set to 2 an absolute domain point value is returned. When domain_flag is set to 1 a scaled domain point value is returned.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NULL_OUTPUT_PTR
The domain min cannot NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_pfunc_elem_count

Function: Deformable Modeling, DML Create and Query

Action: Returns the pfunc's elem_count value.

Prototype:
```
int DM_get_pfunc_elem_count (
    int& rtn_err,           // out: 0=success or
                            // neg err code
    DS_pfunc* pfunc,        // in: pfunc to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function returns the pfunc's elem_count. For B–splines and NURBs each unique knot value defines an element boundary in the domain space of the curve or surface.

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_pfunc_image_dim

Function: Deformable Modeling, DML Create and Query

Action: Gets the pfunc's image dimension value.

Prototype:
```
int DM_get_pfunc_image_dim (
    int& rtn_err,           // out: 0=success or
                            // neg err code
    DS_pfunc* pfunc,        // in: pfunc to query
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Refer to Action.

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_pfunc_knot_counts

Function: Deformable Modeling, DML Create and Query

Action: Gets the pfunc's knot count values (*u* and *v* directions).

Prototype:
```
void DM_get_pfunc_knot_counts (
    int& rtn_err,           // out: 0=success or
                            // neg err code
    DS_pfunc* pfunc,        // in: pfunc to query
    int& knot_count_u,      // pfunc's u knot_count
                            // value
    int& knot_count_v,      // pfunc's v knot_count
                            // value
                            // (set to 0 by curves)
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function gets knot_count_u, knot_count_v.

Errors: DM_NULL_INPUT_PTR
The pfunc cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_pfunc_knots

Function: Deformable Modeling, DML Create and Query

Action: Gets the dof values for the pfunc and returns 0 for success or an error.

Prototype:

```
void DM_get_pfunc_knots (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_pfunc* pfunc,           // in: pfunc to query
    DS_PFN& type_id,           // out: ds_tp1(B-spline
                               // crv), ds_tp2(B-spline
                               // srf) ds_rp1(NURB crv),
                               // ds_rp2(NURB srf)
                               // ds_cir(Circ crv)
    int& u_knot_count,         // out: u dir knot_count
    int& v_knot_count,         // out: v dir knot_count
                               // (not set by curves)
    double*& u_knots,          // out: ptr to
                               // u_knot_array
    int*& u_index,             // out: ptr to u index
                               // array
    double*& v_knots,          // out: ptr to v_knot
                               // array (not set by
                               // curves)
    int*& v_index,             // out: ptr to v index
                               // array (not set by
                               // curves)
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
                               // note: return ptr
                               // values point to memory
                               // within the object. Do
                               // not free these
                               // pointers.
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/dmapinum.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies type_id, u_knot_count, v_knot_count, u_s, u_index, v_s, v_index.

Retrieves the pfunc's basis knot data. This includes a knot_count which is the count of unique knot values, an array of knot indices which encodes the multiple knot counts for each knot location. knots are numbered starting with 0 in the order of their knot values. Multiple knots (knots with the same knot values still get numbered individually.) The largest knot index value for each knot location is stored in the index array. The knots array stores the knot values. Only surfaces set the *v* direction output variables. Curves of type ds_cir do not set any of the knot output variables.

type_id is set to ds_tp1 (B–spline crv), ds_tp2(B–spline srf), ds_rp1(NURB crv), ds_rp2(NURB srf), ds_cir(circ crv)

The output pointer values are set to memory being used by an active object. Do not free the memory associated with these pointers.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_pfunc_type_id

Function: Deformable Modeling, DML Create and Query

Action: Gets the type of a DS_pfunc and returns a DS_PFN DS_pfunc enumeration type.

Prototype:
```
DS_PFN DM_get_pfunc_type_id (
    int& rtn_err,              // out: 0=success or
                               // DM_NULL_INPUT_PTR
    DS_pfunc* pfunc,           // in: pfunc object
                               // to copy
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
                               // rtn: ds_tp1 = B–spline
                               // curve
                               // ds_rp1 = nurb curve
                               // ds_cir = circ curve
                               // ds_tp2 = B–spline
                               // surface
                               // ds_rp2 = nurb
                               // surface
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dspfunc.hxx"
#include "dshusk/dskernel/dmapinum.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies return_err.

Returns the enumeration type for the input pfunc.

Returns one of the following values:

ds_tp1 ..... = B–spline curve
ds_rp1 ..... = NURB curve
ds_cir ..... = circ curve
ds_tp2 ..... = B–spline surface
ds_rp2 ..... = NURB surface

Errors: None

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_pt_press

Function: Deformable Modeling, DML Loads

Action: Gets the load data into return arguments and returns 0 for success or an error.

Prototype:

```
void DM_get_pt_press (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: member of target
                               // dmod hierarchy to
                               // search
    int tag,                   // in: load identifier
    int domain_flag,           // in: 0=dpt in
                               // orig_dmod_space,
                               // 1=dpt in unit_space,
                               // 2=dpt in
                               // internal_pfunc_space.
    double* dpt,               // out: load's domain
                               // loc, [u,v]
                               // sized:[Domain_dim]
    double& gain,              // out: magnitude of load
                               // gain
    int& negate_flag,          // out: change normal dir
                               // (1=negate,0=do not)
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 // note: sets active dmod
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies dpt, gain, negate_flag.

Looks for a load identified by the given tag in the entire patch hierarchy. When the tag specifies a pressure point load, this function fills all the output arguments appropriately and returns 0. It changes the active deformable model to the one containing the load. Otherwise, it returns DM_TAG_OBJECT_NOT_FOUND and leaves output arguments unmodified.

When domain_flag is set to 1, dpt is given in the unit range and mapped to the dmod's actual domain range by this function. When domain_flag is set to 0, dpt is given in the dmod's original domain range. When domain_flag is set to 2, dpt is given in the dmod's pfunc's internal domain range.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NON_NULL_OUTPUT_PTR
The domain point must be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_pt_uv

Function: Deformable Modeling, DML Constraints, DML Loads

Action: Gets the tag's current *uv* location and returns 0 for success or an error.

Prototype:

```
void DM_get_pt_uv (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: member of target
                               // dmod hierarchy to
                               // search
    int tag,                   // in: load/cstrn
                               // identifier
    int domain_flag,           // in: 0=dpt in
                               // orig_dmod_space,
                               // 1=dpt in
                               // unit_space,
                               // 2=dpt in
                               // internal_pfunc_space.
    double* dpt,               // out: tag's current
                               // uv location
                               // sized:[Domain_dim]
    DS_TAGS& type,             // out: set to type of
                               // modified object
                               // or DS_tag_none
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dmapinum.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies dpt for the active deformable model.

Searches for a constraint or load with matching tag. When found, this function loads the current *uv* position into dpt. This works for tags of type pressure point, spring, and point_constraints. For all other tag types it returns DM_NO_UV_PT_FOR_TAG_OBJ.

When domain_flag is set to 1, dpt is given in the unit range and mapped to the pfunc's actual domain range by this function. When domain_flag is set to 0, dpt is given in the dmod's original domain range. When domain_flag is set to 2, dpt is given in the dmod's pfunc's internal domain range.

Errors: DM_NO_ROOT_DMOD
The root deformable model cannot be NULL on input.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

DM_NULL_OUTPUT_PTR
The domain point cannot be NULL on entry.

DM_NO_UV_PT_FOR_TAG_OBJ
The identified tag object has no distinct *uv* point to update dyn_loads, dist_press, crv_loads and crv_cstrns, or a deformable model.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_pt_xyz

Function: Deformable Modeling, DML Constraints, DML Loads

Action: Gets the tag's current *xyz* location and returns 0 for success or an error.

Prototype:

```
void DM_get_pt_xyz (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of target
                            // dmod hierarchy to
                            // search
    int tag,                // in: load/cstrn
                            // identifier
    int pt_index,           // in: index of tag's
                            // image_pt to update
                            // one of
                            // DM_BASE_PT
                            // DM_END_LEG
                            // DM_TANG_LEG
                            // DM_TANG1_LEG
                            // DM_TANG2_LEG
                            // DM_NORM_LEG
                            // DM_CURV1_LEG
                            // DM_CURV2_LEG
                            // DM_BINORM_LEG
                            // or a control point tag
                            // or an id number in a
                            // spring set
    double* p0,             // ut: tag's current
                            // xyz location
                            // sized:[image_dim]
    DS_TAGS& type,          // out: set to type of
                            // object queried or
                            // DS_tag_none
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/dmapinum.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies p0 of the active deformable model.

Searches for a constraint, load, or control point with a matching tag. When found, this function loads its current *xyz* position into p0 and sets the active deformable model. This works for any tag object which is associated with one or more point locations.

For vector_loads, when the pt_index equals DM_BASE_PT, this function returns the vector_load's base point and when pt_index equals DM_END_LEG returns the load's end_pt (location of the tip of the tang_vector for that load vector).

For point constraints, returns the point location of the constraint's base point or one of its vector end points as selected by the pt_index value. Known pt_index values for point constraints include:

pt_index = DM_BASE_PT . . . . . for the point's base_point.
pt_index = DM_TANG_LEG . . . . . for a curve's tangent end point.
pt_index = DM_TANG1_LEG . . . . . for a surface's tang1 end point.
pt_index = DM_TANG2_LEG . . . . . for a surface's tang2 end point.
pt_index = DM_NORM_LEG . . . . . for a curve or surface normal vector end point.
pt_index = DM_CURV_LEG . . . . . for a curve's curvature end point.
pt_index = DM_CURV1_LEG . . . . . for a surface's curv1 end point.
pt_index = DM_CURV2_LEG . . . . . for a surface's curv2 end point.
pt_index = DM_BINORM_LEG . . . . for a curve's binormal end point.

For curve constraints this function returns the last pick point (that is the last location at which the functions DM_find_tag_by_image_line() or DM_find_pos_by_image_line() queried the deformable model).

For pressure point loads returns the load's image point. For springs returns the spring's free_pt location. For spring sets uses pt_index to select which spring free_pt to return. Locations for control points within the active deformable model can be queried by setting the tag value to less than or equal to –500. For all other tag types returns DM_NO_XYZ_PT_FOR_TAG_OBJ.

Errors: DM_NO_ROOT_DMOD
The root deformable model cannot be NULL on input.

DM_NULL_INPUT_PTR
The value of P0 or p1 cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

DM_NO_XYZ_PT_FOR_TAG_OBJ
The identified tag object has no distinct *xyz* point to report dyn_loads, dist_press, crv_loads and crv_cstrns, or deformable model.

DM_BAD_PT_INDEX_VALUE
The spring_set and pt_index must be within the point count range.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_sibling

Function: Deformable Modeling, DML Patches

Action: Gets the deformable model's sibling and returns the sibling pointer or NULL.

Prototype:
```
DS_dmod* DM_get_sibling (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns the deformable model's sibling pointer.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_sibling_list

Function: Deformable Modeling, DML Tags

Action: Builds and returns a root sibling list.

Prototype:

```
void DM_get_sibling_list (
    int& rtn_err,              // out: 0=success or
                               // neg err code
    DS_dmod* dmod,             // in: member of dmod
                               // hierarchy
    int& sibling_count,        // number of root level
                               // siblings
    int*& sibling_list,        // sibling–id's (tag–id)
                               // [id0, id1, ...]
                               // sized:[sibing_count]
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
                               // mallocs: caller must
                               // free sibling_list
                               // array
                               // note: sets active dmod
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Builds and returns a list containing the tag identifier for all root siblings of the target deformable model's deformable modeling hierarchy. It sets the sibling count to agree with the number of root siblings.

The sibling list data is always ordered as:

[id0] = root dmod
[id1] = 1st sibling dmod
[id2] = 2nd sibling dmod
[...]

Mallocs: The calling program must delete the sibling list array when it is done with it.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on input.

DM_NON_NULL_OUTPUT_PTR
The sibling list cannot be NULL on entry

DM_parse_tag_flag() errors

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_spring

Function: Deformable Modeling, DML Loads

Action: Gets the load data and places it into return arguments. Returns 0 for success or an error.

Prototype:

```
void DM_get_spring (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of target
                            // dmod hierarchy to
                            // search
    int tag,                // in: load identifier
    int domain_flag,        // in: 0=domain_pts in
                            // orig_dmod_space,
                            // 1=dpt in unit_space.
                            // 2=dpt in
                            // internal_pfunc_space.
    double* dpt,            // out: domain_pt loc of
                            // surf spring end
                            // sized:[Domain_dim]
    double* free_pt,        // out: image_pt loc of
                            // free spring end
                            // sized:[image_dim]
    double* base_pt,        // out: image_pt loc of
                            // attached spring end
                            // sized:[image_dim]
    double& gain,           // out: magnitude of
                            // load gain
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies dpt, ipt, base_pt, gain.

Looks for a spring load in the entire patch hierarchy. When the tag specifies a spring load, this function fills all the output arguments appropriately and returns 0. Changes the active deformable model to the one containing the load. Else returns DM_TAG_OBJECT_NOT_FOUND and leaves output arguments unmodified.

When domain_flag is set to 1, dpt is given in the unit range and mapped to the pfunc's actual domain range by this function. When domain_flag is set to 0, dpt is given in the dmod's original domain range. When domain_flag is set to 2, dpt is given in the dmod's pfunc's internal domain range.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NULL_OUTPUT_PTR
The domain point and the image point are NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_spring_length

Function: Deformable Modeling, DML Loads

Action: Gets the spring or spring set length and returns 0 for success or an error.

Prototype:
```
double DM_get_spring_length (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of target
                            // dmod hierarchy to
                            // search
    int tag,                // in: load identifier
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies length.

Retrieves the tag object identified by the input tag value. When that object is a spring, this function then places the length of that spring into the output variable. The spring set then places the length of the longest spring in the set into the output object.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_spring_set

Function: Deformable Modeling, DML Loads

Action: Gets the spring set load data and places it into return arguments.

Prototype:

```
void DM_get_spring_set (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: member of target
                            // dmod hierarchy to
                            // search
    int tag,                // in: load identifier
    int domain_flag,        // in: 0=domain_pts in
                            // orig_dmod_space
                            // 1=domain_pts in
                            // unit_space.
                            // 2=domain_pts in
                            // internal_pfunc_space.
    int& pt_count,          // out: number of springs
                            // in spring_set
    double*& domain_pts,    // out: Spring dmod
                            // end pts
                            // 1d=[u0, u1, ...un],
                            // 2d=[uv0..,uvN]
                            // Sized:[pt_count*
                            // domain_dim]
    double*& free_pts,      // ut: Spring free end
                            // pts or NULL
                            // [xyz0, ... xyzN]
                            // Sized:[pt_count*
                            // image_dim]
    double*& base_pts,      // ut: Spring attached
                            // end pts or NULL
                            // [xyz0, ... xyzN]
                            // Sized:[pt_count*
                            // image_dim]
    double& gain,           // out: magnitude of
                            // load gain
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”dshusk/dskernel/dmapi.hxx”
#include ”dshusk/dskernel/dsdmod.hxx”
#include ”dshusk/dskernel/sdm_options.hxx”
```

Description: Modifies pt_count, domain_pts, free_pts, gain.

Looks for a spring set load identified by the input tag in the entire patch hierarchy. When tag specifies a spring set load, this function fills all the output arguments appropriately and returns 0. Changes the active deformable model to the one containing the load. Otherwise, returns DM_TAG_OBJECT_NOT_FOUND and leaves output arguments unmodified.

mallocs the memory used for the domain point output array so that the return domain point values can be scaled from the internal domain range to the unit square domain range. The calling program must free this memory.

When domain_flag is set to 1, domain_pts are given in the unit range and mapped to the dmod's actual domain range by this function. When domain_flag is set to 0, domain_pts are given in the dmod's original domain range. When domain_flag is set to 2, domain_pts are given in the dmod's pfunc's internal domain range.

This function updates the rest of the output pointer values to point to the load's nested data. Do not use these pointer values to free memory, nor after the pfunc has been deleted.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NULL_OUTPUT_PTR
The domain point and the free point are NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_tags

Function: Deformable Modeling

Action: Gets the tags of all tag objects in a DS_dmod hierarchy.

Prototype:
```
void DM_get_tags (
    int& rtn_err,           // error code
    DS_dmod* dmod,          // model to check
    int& ntags,             // number of tags
    DM_tag_array& tags,     // array of tags
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dm_tag_array.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function returns a DM_tag_array array containing the tags of all tag objects embedded in the DS_dmod hierarchy containing the input DS_dmod. The array is sized:[tags.Size()]. rtn_err returns 0 or a negative error code.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_tag_count

Function: Deformable Modeling, DML Tags

Action: Gets the deformable model's dmo_tag_count value and returns the largest tag value currently assigned.

Prototype:
```
int DM_get_tag_count (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod member
                            // of a hierarchy
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Refer to action.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_tag_summary

Function: Deformable Modeling, DML Tags

Action: Gets the build and return tag summary list and returns 0 for success or an error.

Prototype:

```
void DM_get_tag_summary (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: dmod to query
    int& tag_count,            // out: number of tag
                               // objects
    int*& tag_summary,         // out: tag-object's
                               // (tag-id tag-type)
                               // [id0, type0, id1,
                               // type1, ...]
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 //
                               // note: rtn_size:
                               // [2*tag_count]
                               // mallocs: caller must
                               // free tag_summary array
                               // note: sets active dmod
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies the tag count. This function builds and returns a list containing the tag identifier and tag type for all tag objects in the target deformable model. It loads the tag_count with the number of tag objects on the deformable model. The tag summary list data is always ordered as:

[id0, type0]	= target deformable model
[id1, type1]	= Parent deformable model
[id2, type2]	= Sibling deformable model
[id3, type3]	= Child deformable model
[id4, type4]	= ordered list of loads and constraints
[..., ...]	= tag identifiers and types

The first eight entries of the tag summary are always hierarchy data. When any of the hierarchy pointers are NULL their corresponding identifiers are set to –1, and the tag type is set to DS_tag_none. All tag types are defined in the DS_TAGS enumeration but are stored in the tag summary as ints.

Includes malloc; the calling program must delete the tag summary array when it is finished.

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NON_NULL_OUTPUT_PTR
The tag summary must be NULL.

DM_parse_tag_flag() errors

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_tan_display_gain

Function: Deformable Modeling, DML Graphics

Action: Gets a display scaling for tangent vectors.

Prototype:
```
double DM_get_tan_display_gain (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    DS_dmod* dmod,          // in: dmod to be queried
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Returns tangent display_gain used to scale the visual display of tangent vectors and stores the value. The visual display of the tangent vector is used for display, picking, and setting of tangent vector values on point constraints. Curvature vectors are scaled for plotting by the comb_gain parameter in DM_get_comb_graphics().

Errors: DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_tight_state

Function: Deformable Modeling

Action: Gets the tight state for a constraint/load.

Prototype:
```
int DM_get_tight_state (
    int& rtn_err,           // success or error code
    DS_dmod* dmod,          // member of deformable
                            // model to search
    int tag,                // tag identifier
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              // note: sets active dmod
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: This function queries the tag tight behavior bit. It returns 1 when tight is enabled, and returns 0 when tight is disabled. The rtn_err indicates success (0) or a negative error code. The dmod is the member of the deformable model hierarchy to be searched. The tag is a constraint identifier which sets the active deformable model to the owner of the tag.

Errors: DM_NULL_INPUT_PTR: The deformable model cannot be NULL on entry.

DM_TAG_OBJECT_NOT_FOUND: The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Read–only

DM_get_tolerance

Function: Deformable Modeling, DML Create and Query

Action: Gets the tolerance limits.

Prototype:

```
void DM_get_tolerance (
    int& rtn_err,           // out: 0=success
                            // or negative err code
    double& dist_tol,       // out: min dist between
                            // two points
    double& ang_tol,        // out: min angle between
                            // two lines
    SDM_options* sdmo       // in:SDM_options pointer
        = NULL              //
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies dist_tol, ang_tol. Gets the DS_tolerance and DS_angle_tol values. The DS_tolerance value is used to determine when the curve constraints are being held. Due to machine round–off this number should be no more than 10 orders of magnitude less than the range of the image space. So if you build your models in a 10,000 unit cube, dist_tol should be no smaller than 1.0E–06. The angle tolerance is not currently used.

Errors: None

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model

DM_get_vector_load

Function: Deformable Modeling, DML Loads

Action: Gets the vector load data and places it into return arguments. returns 0 for success or an error.

Prototype:

```
void DM_get_vector_load (
    int& rtn_err,              // out: 0=success
                               // or negative err code
    DS_dmod* dmod,             // in: member of target
                               // dmod hierarchy to
                               // search
    int tag,                   // in: load identifier
    double* image_vec,         // out: image_vec for
                               // vector_load direction
                               // sized:[image_dim]
    double& gain,              // out: magnitude of
                               // load gain
    SDM_options* sdmo          // in:SDM_options pointer
        = NULL                 // note: sets active dmod
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "dshusk/dskernel/dmapi.hxx"
#include "dshusk/dskernel/dsdmod.hxx"
#include "dshusk/dskernel/sdm_options.hxx"
```

Description: Modifies image_vec, and gain.

Looks for a vector load identified by tag in the entire patch hierarchy. When the tag specifies a DS_vector_load, fills all the output arguments appropriately and returns 0. Changes the active deformable model to the one containing the load. Otherwise, returns the error DM_TAG_OBJECT_NOT_FOUND and leaves output arguments unmodified.

Errors:

DM_NULL_INPUT_PTR
The deformable model cannot be NULL on entry.

DM_NULL_OUTPUT_PTR
The image vector is NULL on entry.

DM_TAG_OBJECT_NOT_FOUND
The input tag cannot identify a deformable model, a load, a spring, a spring set, an attractor object, or a constraint in the model hierarchy.

Limitations: None

Library: dshusk

Filename: ds/dshusk/dskernel/dmapi.hxx

Effect: Changes model