Chapter 3.
Functions

Topic: Ignore

The function interface is a set of Application Procedural Interface (API) and Direct Interface (DI) functions that an application can invoke to interact with ACIS. API functions, which combine modeler functionality with application support features such as argument error checking and roll back, are the main interface between applications and ACIS. The DI functions provide access to modeler functionality, but do not provide the additional application support features, and, unlike APIs, are not guaranteed to remain consistent from release to release. Refer to the *3D ACIS Online Help User's Guide* for a description of the fields in the reference template.

api_hh_analytic_analyze

Function: Healing

Action: Analyzes edges lying on analytic surfaces.

Prototype:
```
outcome api_hh_analytic_analyze (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API is used in the geometry building phase. It performs the analyze stage of the analytic solver subphase of geometry building. The analytic solver subphase attempts to heal all edges and vertices shared by analytic surfaces.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_analytic_auto

Function: Healing

Action: Automatically executes the analyze and calculate stages of the analytic solver subphase of geometry building.

Prototype:
```
outcome api_hh_analytic_auto (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API performs all of the stages of the analytic solver subphase of the geometry building phase. The analytic solver subphase attempts to heal all edges and vertices shared by analytic surfaces.

The APIs for the analyze and calculate stages of the analytic solver subphase of geometry building are called sequentially. Intelligent tolerances that are recommended by the analyze stage are used in the calculate stage.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_analytic_calc_fix

Function: Healing

Action: Calculates and applies the new geometry of analytic surfaces and the corresponding curves and vertices.

Prototype:
```
outcome api_hh_analytic_calc_fix (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API is used in the geometry building phase. It performs the calculate and fix stages of the analytic solver subphase of geometry building. The analytic solver subphase attempts to heal all edges and vertices shared by analytic surfaces.

This API calculates all analytic geometries. Tangency constraints are solved with the aid of a solver. The new geometry is applied (fixed) to the body and the old geometry is placed on the model in attributes so that the user may compare the new and the old geometry.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_analyze_body

Function: Healing

Action: Checks the input body for errors and stores results in attributes attached to the bad entities.

Prototype:
```
outcome api_hh_analyze_body (
    BODY* body,             // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API checks the input body for errors. The tests include all the tests performed by the individual analyze APIs for the various specific types of entities (e.g., api_hh_analyze_coedges). The results are attached to any bad entities as attributes.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_analyze_coedges

Function: Healing

Action: Checks the coedges of the input body for errors and stores results in attributes attached to the bad coedges.

Prototype:
```
outcome api_hh_analyze_coedges (
    BODY* body,              // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API checks all the coedges of the input body. The results are attached to any bad coedges as attributes. The tests include:

- Does the coedge lie on the corresponding face surface?
- If the coedge contains a pcurve, does the domain of the pcurve correspond with the edge?
- Does the coedge have a partner?
- If the coedge contains a pcurve, is the pcurve within tolerance of the edge?

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_analyze_edges

Function: Healing

Action: Checks the edges of the input body for errors and stores results in attributes attached to the bad edges.

Prototype:
```
outcome api_hh_analyze_edges (
    BODY* body,             // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API checks all the edges of the input body. The results are attached to any bad edges as attributes. The tests include:

- Checks curve geometry
- Determines convexity
- Checks length

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_analyze_faces

Function: Healing

Action: Checks the faces of the input body for errors and stores results in attributes attached to the bad faces.

Prototype:
```
outcome api_hh_analyze_faces (
    BODY* body,             // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API checks all the faces of the input body. The results are attached to any bad faces as attributes. The tests include:

- Checks loops
- Checks surface
- Checks face area

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_analyze_loops

Function: Healing

Action: Checks the loops of the input body for errors and stores results in attributes attached to the bad loops.

Prototype:
```
outcome api_hh_analyze_loops (
    BODY* body,             // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API checks all the loops of the input body. The results are attached to any bad loops as attributes. The tests include:

– Is the loop closed?
– Checks loop orientation
– Do the loop coedges have gaps?
– Does the loop self–intersect?
– Checks for correct parameter range of the coedges
– Checks that the coedges lie on the face surface

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_analyze_lumps

Function: Healing

Action: Checks the lumps of the input body for errors and stores results in attributes attached to the bad lumps.

Prototype:
```
outcome api_hh_analyze_lumps (
    BODY* body,              // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”healhusk/heal_api/heal_api.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/top/body.hxx”
#include ”kernel/kernapi/api/acis_options.hxx”
```

Description: This API checks all the lumps of the input body. The results are attached to any bad lumps as attributes. The tests include:

– Checks shells for closure

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_analyze_shells

Function: Healing

Action: Checks the shells of the input body for errors and stores results in attributes attached to the bad shells.

Prototype:
```
outcome api_hh_analyze_shells (
    BODY* body,              // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API checks all the shells of the input body. The results are attached to any bad shells as attributes. The tests include:

- Checks that the shell is closed
- Checks shell orientation
- Checks if shell represents a single volume

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_analyze_vertices

Function: Healing

Action: Checks the vertices of the input body for errors and stores results in attributes attached to the bad vertices.

Prototype:
```
outcome api_hh_analyze_vertices (
    BODY* body,              // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API checks all the vertices of the input body. The results are attached to any bad vertices as attributes. The tests include:

- Does the vertex lie on the corresponding edges?
- Do the edges meet at the vertex?
- Does the vertex lie on the corresponding surfaces?

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_auto_heal

Function: Healing

Action: Automatically heals the input body (performs all phases of the healing process) using intelligently selected tolerances.

Prototype:
```
outcome api_hh_auto_heal (
    BODY* body,              // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API automatically sets tolerances (based on analysis of the body) and heals the body by executing all the healing phases sequentially. This API calls the APIs for preprocess, automatic geometry simplification, automatic stitching, automatic geometry building, and postprocess:

- api_hh_preprocess
- api_hh_simplify_auto
- api_hh_stitch_auto
- api_hh_geombuild_auto
- api_hh_postprocess

All the phases and subphases of the healing process are performed automatically. Each healing phase analyzes the body first, then sets the best values for the tolerances and options used by that healing phase. (These tolerances can be modified by callbacks.) Next, the healing phase calculates the new geometry/topology and applies the changes to the body.

The API api_hh_init_body_for_healing should be called to attach the aggregate healing attributes to the body before performing automatic healing. The healing attributes are retained on the healed body after automatic healing is done to hold the healing results. This allows the user to modify such things as tolerances and redo any or all of the healing process if desired. The API api_hh_end_body_for_healing should be called after healing is completed to remove the attributes from the body.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_end_body_for_healing

Function: Healing

Action: Terminates the healing process for a body.

Prototype:
```
outcome api_hh_end_body_for_healing (
    BODY* body,              // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/top/body.hxx”
#include ”healhusk/heal_api/heal_api.hxx”
#include ”kernel/kernapi/api/acis_options.hxx”
```

Description: This API must be called after the healing process is complete. It detaches the aggregate attributes (that contain such things as the tolerances for the various healing phases) from the body.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_force_simplify_to_cone

Function: Healing

Action: Forces the given surface to an analytic surface (cone).

Prototype:
```
outcome api_hh_force_simplify_to_cone (
    FACE* inp_face,          // input face
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/face.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs force simplification of splines. Geometry simplification attempts to simplify NURBS surfaces into analytic forms (planes, cylinders, cones, tori, and spheres).

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_force_simplify_to_cylinder

Function: Healing

Action: Forces the given surface to an analytic surface (cylinder).

Prototype:
```
outcome api_hh_force_simplify_to_cylinder (
    FACE* inp_face,          // input face
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/face.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs force simplification of splines. Geometry simplification attempts to simplify NURBS surfaces into analytic forms (planes, cylinders, cones, tori, and spheres).

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_force_simplify_to_plane

Function: Healing

Action: Forces the given surface to an analytic surface (plane).

Prototype:
```
outcome api_hh_force_simplify_to_plane (
    FACE* inp_face,          // input face
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/face.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs force simplification of splines. Geometry simplification attempts to simplify NURBS surfaces into analytic forms (planes, cylinders, cones, tori, and spheres).

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_force_simplify_to_sphere

Function: Healing

Action: Forces the given surface to an analytic surface (sphere).

Prototype:
```
outcome api_hh_force_simplify_to_sphere (
    FACE* inp_face,          // input face
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/face.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API performs force simplification of splines. Geometry simplification attempts to simplify NURBS surfaces into analytic forms (planes, cylinders, cones, tori, and spheres).

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_force_simplify_to_torus

Function: Healing

Action: Forces the given surface to an analytic surface (torus).

Prototype:
```
outcome api_hh_force_simplify_to_torus (
    FACE* inp_face,          // input face
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/face.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API performs force simplification of splines. Geometry simplification attempts to simplify NURBS surfaces into analytic forms (planes, cylinders, cones, tori, and spheres).

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_gen_spline_analyze

Function: Healing

Action: Analyzes generic spline intersections.

Prototype:
```
outcome api_hh_gen_spline_analyze (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API is used in the geometry building phase. It performs the analyze stage of the generic spline solver subphase of geometry building. The generic spline solver attempts to heal generic tangential spline junctions, (e.g., the intersection curve is *not* an isoparametric curve of both splines in the intersection).

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_gen_spline_auto

Function: Healing

Action: Automatically executes the analyze and calculate stages of the generic spline solver subphase of geometry building.

Prototype:
```
outcome api_hh_gen_spline_auto (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs all of the stages of the generic spline solver subphase of the geometry building phase. The generic spline solver attempts to heal generic tangential spline junctions, (e.g., the intersection curve is *not* an isoparametric curve of both splines in the intersection).

The APIs for the analyze and calculate stages of the generic spline solver subphase of geometry building are called sequentially. Intelligent tolerances that are recommended by the analyze stage are used in the calculate stage.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_gen_spline_calc_fix

Function: Healing

Action: Calculates the new geometry of spline surfaces that intersect tangentially at non–isoparametric junctions.

Prototype:
```
outcome api_hh_gen_spline_calc_fix (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API is used in the geometry building phase. It performs the calculate and fix stages of the generic spline solver subphase of geometry building. The generic spline solver attempts to heal generic tangential spline junctions, (e.g., the intersection curve is *not* an isoparametric curve of both splines in the intersection).

This API uses surface fitting algorithms to calculate the new geometry needed to heal non–isoparametric junctions of spline geometries intersecting tangentially. The new geometry is applied (fixed) to the body and the old geometry is placed on the model in attributes so that the user may compare the new and the old geometry.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_geombuild_analyze

Function: Healing

Action: Analyzes the geometry of the body and attaches attributes to bad edges, vertices, and coedges.

Prototype:
```
outcome api_hh_geombuild_analyze (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API performs the analyze stage of the geometry building phase. The geometry building phase performs all the geometry related healing operations, including fixing of edge geometries by intersections, snapping surfaces for fixing tangencies, and refitting spline surfaces.

A check is run on the body and the inaccurate geometries are marked with attributes. A invalid edge is one in which the edge curve does not lie on the underlying surfaces to ACIS tolerance. A vertex is marked bad if it does not lie on the edges or faces which are incident on it. A bad coedge is one whose pcurve does not match with the edge curve.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_geombuild_auto

Function: Healing

Action: Automatically executes the analyze, calculate, and fix stages of the geometry building phase.

Prototype:
```
outcome api_hh_geombuild_auto (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: The APIs for the analyze, calculate, and fix stages of the geometry building phase are called sequentially. Intelligent tolerances that are recommended by the analyze stage are used in the calculate stage. The geometry building phase performs all the geometry related healing operations, including fixing of edge geometries by intersections, snapping surfaces for fixing tangencies, and refitting spline surfaces.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_geombuild_calc_fix

Function: Healing

Action: Calculates new geometry for all the bad geometry that was marked by the geometry building analyze stage.

Prototype:
```
outcome api_hh_geombuild_calc_fix (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API performs the calculate and fix stages of the geometry building phase. The geometry building phase performs all the geometry related healing operations, including fixing of edge geometries by intersections, snapping surfaces for fixing tangencies, and refitting spline surfaces.

This API calculates the geometry in the model using all the geometry building subphases:

Analytic solver subphase	Handles the analytic geometries.
Isospline solver subphase	Handles tangential splines that are on isoparametric lines.
Sharp edge subphase	Handles intersections of nontangential (sharp) edges.
Generic spline solver subphase	Handles tangential spline junctions that are non–isoparametric.
Wrap up subphase	Adds pcurves.

The new geometry is applied (fixed) to the body and the old geometry is placed on the model in attributes so that the user may compare the new and the old geometry.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_geombuild_check

Function: Healing

Action: Checks the geometry of the body and attaches attributes to bad edges, vertices, and coedges. This API does not set any tolerances.

Prototype:
```
outcome api_hh_geombuild_check (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs a check on the geometry of a body. This API is useful at the end of healing to check results of the geometry stage. The results of this check are stored in the aggregate attribute and can be obtained using the member function:
ATTRIB_HH_AGGR_GEOMBUILD::output_analysis_results()

A check is run on the body and the inaccurate geometries are marked with attributes. A invalid edge is one in which the edge curve does not lie on the underlying surfaces to ACIS tolerance. A vertex is marked bad if it does not lie on the edges or faces which are incident on it. A bad coedge is one whose pcurve does not match with the edge curve.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_geombuild_cleanup

Function: Healing

Action: Cleans up the individual healing attributes related to the geometry building phase.

Prototype:
```
outcome api_hh_geombuild_cleanup (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_get_bad_coedges

Function: Healing

Action: Returns a list of all coedges in the input body that were marked as bad during healing analysis.

Prototype:
```
outcome api_hh_get_bad_coedges (
    BODY* body,              // input body
    ENTITY_LIST& list,       // list of bad coedges
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API returns a list of all coedges in the input body that were marked as bad during a healing analysis operation. Refer to the descriptions of the various analysis APIs for the tests that are performed.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Read–only

api_hh_get_bad_edges

Function: Healing

Action: Returns a list of all edges in the input body that were marked as bad during healing analysis.

Prototype:
```
outcome api_hh_get_bad_edges (
    BODY* body,             // input body
    ENTITY_LIST& list,      // list of bad edges
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”healhusk/heal_api/heal_api.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/lists/lists.hxx”
#include ”kernel/kerndata/top/body.hxx”
#include ”kernel/kernapi/api/acis_options.hxx”
```

Description: This API returns a list of all edges in the input body that were marked as bad during a healing analysis operation. Refer to the descriptions of the various analysis APIs for the tests that are performed.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Read–only

api_hh_get_bad_faces

Function: Healing

Action: Returns a list of all faces in the input body that were marked as bad during healing analysis.

Prototype:
```
outcome api_hh_get_bad_faces (
    BODY* body,             // input body
    ENTITY_LIST& list,      // list of bad faces
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”healhusk/heal_api/heal_api.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/lists/lists.hxx”
#include ”kernel/kerndata/top/body.hxx”
#include ”kernel/kernapi/api/acis_options.hxx”
```

Description: This API returns a list of all faces in the input body that were marked as bad during a healing analysis operation. Refer to the descriptions of the various analysis APIs for the tests that are performed.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Read–only

api_hh_get_bad_loops

Function: Healing

Action: Returns a list of all loops in the input body that were marked as bad during healing analysis.

Prototype:
```
outcome api_hh_get_bad_loops (
    BODY* body,              // input body
    ENTITY_LIST& list,       // list of bad loops
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/lists/lists.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API returns a list of all loops in the input body that were marked as bad during a healing analysis operation. Refer to the descriptions of the various analysis APIs for the tests that are performed.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Read–only

api_hh_get_bad_lumps

Function: Healing

Action: Returns a list of all lumps in the input body that were marked as bad during healing analysis.

Prototype:
```
outcome api_hh_get_bad_lumps (
    BODY* body,              // input body
    ENTITY_LIST& list,       // list of bad lumps
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API returns a list of all lumps in the input body that were marked as bad during a healing analysis operation. Refer to the descriptions of the various analysis APIs for the tests that are performed.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Read–only

api_hh_get_bad_shells

Function: Healing

Action: Returns a list of all shells in the input body that were marked as bad during healing analysis.

Prototype:
```
outcome api_hh_get_bad_shells (
    BODY* body,              // input body
    ENTITY_LIST& list,       // list of bad shells
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/lists/lists.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API returns a list of all shells in the input body that were marked as bad during a healing analysis operation. Refer to the descriptions of the various analysis APIs for the tests that are performed.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Read–only

api_hh_get_bad_vertices

Function: Healing

Action: Returns a list of all vertices in the input body that were marked as bad during healing analysis.

Prototype:
```
outcome api_hh_get_bad_vertices (
    BODY* body,             // input body
    ENTITY_LIST& list,      // list of bad vertices
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/lists/lists.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API returns a list of all vertices in the input body that were marked as bad during a healing analysis operation. Refer to the descriptions of the various analysis APIs for the tests that are performed.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Read–only

api_hh_get_entity_details

Function: Healing

Action: Returns the history of an entity during healing.

Prototype:
```
outcome api_hh_get_entity_details (
    ENTITY* entity,          // input edge or face
    char*& history,          // history of entity
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/data/entity.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API returns a description of the major changes that the given entity (an EDGE or FACE) underwent during healing in readable text format. The description gets stored during healing of each entity (only if the option to store the history is put on – refer api_hh_store_entity_details). Output is NULL if no history is found. It is the responsibility of the caller to delete memory allocated to char* history.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Read–only

api_hh_init_body_for_healing

Function: Healing

Action: Initializes the body for healing.

Prototype:
```
outcome api_hh_init_body_for_healing (
    BODY* body,              // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API must be called before the healing process is begun. It attaches aggregate attributes to the body. These attributes are used to store such information as the tolerances for the various healing phases.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_isospline_analyze

Function: Healing

Action: Analyzes isoparametric edges between spline surfaces.

Prototype:
```
outcome api_hh_isospline_analyze (
    BODY*,                   // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API is used in the geometry building phase. It performs the analyze stage of the isospline solver subphase of geometry building. The isospline solver attempts to heal all edges shared by tangential isoparametric surfaces (e.g., the intersection curve is an isoparametric curve of both splines in the intersection).

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_isospline_auto

Function: Healing

Action: Automatically executes the analyze and calculate stages of the isospline solver subphase of geometry building.

Prototype:
```
outcome api_hh_isospline_auto (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: This API performs all of the stages of the isospline solver subphase of the geometry building phase. The isospline solver attempts to heal all edges shared by tangential isoparametric surfaces (e.g., the intersection curve is an isoparametric curve of both splines in the intersection).

The APIs for the analyze and calculate stages of the isospline solver subphase of geometry building are called sequentially. Intelligent tolerances that are recommended by the analyze stage are used in the calculate stage.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_isospline_calc_fix

Function: Healing

Action: Calculates all the new geometry of spline surfaces which intersect tangentially at isospline junctions.

Prototype:
```
outcome api_hh_isospline_calc_fix (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API is used in the geometry building phase. It performs the calculate and fix stages of the isospline solver subphase of geometry building. The isospline solver attempts to heal all edges shared by tangential isoparametric surfaces (e.g., the intersection curve is an isoparametric curve of both splines in the intersection).

This API calculates isoparametric junctions of spline geometries intersecting tangentially. The new geometry is applied (fixed) to the body and the old geometry is placed on the model in attributes so that the user may compare the new and the old geometry.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_make_tolerant

Function: Healing, Tolerant Modeling

Action: Converts unhealed edges to tolerant edges.

Prototype:
```
outcome api_hh_make_tolerant (
    BODY* body,             // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API examines all the edges in the body and converts each unhealed edge into a tolerant edge. This API should be executed before api_hh_end_body_for_healing in order to use the healing attributes.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_postprocess

Function: Healing

Action: Executes the final cleanup of the healed model.

Prototype:
```
outcome api_hh_postprocess (
    BODY* body,              // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs postprocessing of the healed data. It performs such operations as correction of negative area faces, duplicate vertices, and edge groups.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_preprocess

Function: Healing

Action: Executes initial cleanup of the model to be healed.

Prototype:
```
outcome api_hh_preprocess (
    BODY* body,              // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
```

Description: This API removes such things from the body as zero–length edges, sliver faces, and duplicate vertices. This should be executed before any healing operations (except autoheal, which performs this phase).

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_sharp_edge_analyze

Function: Healing

Action: Analyzes sharp edges that intersect nontangentially.

Prototype:
```
outcome api_hh_sharp_edge_analyze (
    BODY*,                   // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API is used in the geometry building phase. It performs the analyze stage of the sharp edge solver subphase of geometry building. The sharp edge solver attempts to heal all edges and vertices that are shared by surfaces that intersect sharply. This includes nontangential surface junctions.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_sharp_edge_auto

Function: Healing

Action: Automatically executes the analyze and calculate stages of the sharp edge subphase of geometry building.

Prototype:
```
outcome api_hh_sharp_edge_auto (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs all of the stages of the sharp edge solver subphase of the geometry building phase. The sharp edge solver attempts to heal all edges and vertices that are shared by surfaces that intersect sharply. This includes nontangential surface junctions.

The APIs for the analyze and calculate stages of the sharp edge solver subphase of geometry building are called sequentially. Intelligent tolerances that are recommended by the analyze stage are used in the calculate stage.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_sharp_edge_calc_fix

Function: Healing

Action: Calculates sharp edge geometries of spline surfaces.

Prototype:
```
outcome api_hh_sharp_edge_calc_fix (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API is used in the geometry building phase. It performs the calculate and fix stages of the sharp edge solver subphase of geometry building. The sharp edge solver attempts to heal all edges and vertices that are shared by surfaces that intersect sharply. This includes nontangential surface junctions. The new geometry is applied (fixed) to the body and the old geometry is placed on the model in attributes so that the user may compare the new and the old geometry.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_simplify_analyze

Function: Healing

Action: Analyzes the body for geometry simplification.

Prototype:
```
outcome api_hh_simplify_analyze (
    BODY* inp_body,          // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API analyzes the body and intelligently sets values of required options and tolerances for geometry simplification. Geometry simplification attempts to simplify NURBS surfaces into analytic forms (planes, cylinders, cones, tori, and spheres). If the body is fully analytic, this API sets a flag in the simplification aggregate attribute indicating that no geometry simplification is needed.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_simplify_auto

Function: Healing

Action: Automatically executes the analyze, calculate, and fix stages of the geometry simplification phase.

Prototype:
```
outcome api_hh_simplify_auto (
    BODY* inp_body,          // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: The APIs for the analyze, calculate, and fix stages of geometry simplification are called sequentially. Intelligent tolerances that are recommended by the analyze stage are used in the calculate stage. Geometry simplification attempts to simplify NURBS surfaces into analytic forms (planes, cylinders, cones, tori, and spheres).

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_simplify_calculate

Function: Healing

Action: Calculates the simplified analytic forms of the splines.

Prototype:
```
outcome api_hh_simplify_calculate (
    BODY* inp_body,          // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs the calculation stage of geometry simplification. Geometry simplification attempts to simplify NURBS surfaces into analytic forms. Spline surfaces are converted wherever possible to planes, cylinders, cones, tori, and spheres.

The new geometry is stored in attributes (attached to individual faces) so that the user may compare the new and the old geometry. The fix stage must be used to apply (fix) the new geometry to the body.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_simplify_cleanup

Function: Healing

Action: Cleans up the individual healing attributes related to the geometry simplification phase.

Prototype:
```
outcome api_hh_simplify_cleanup (
    BODY* inp_body,          // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_simplify_fix

Function: Healing

Action: Executes the fix stage of geometry simplification.

Prototype:
```
outcome api_hh_simplify_fix (
    BODY* inp_body,          // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs the fix stage of geometry simplification. Geometry simplification attempts to simplify NURBS surfaces into analytic forms (planes, cylinders, cones, tori, and spheres).

After the calculate stage, the new geometry is placed in the attributes attached to the body and the user can compare the old and new geometry. The fix stage applies (fixes) the new geometry to the body and stores the old geometry in the attributes.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_stitch_analyze

Function: Healing

Action: Analyzes the topology of the input model for stitching.

Prototype:
```
outcome api_hh_stitch_analyze (
    BODY* inp_body,          // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API analyzes the body and intelligently sets values of required options and tolerances for stitching. Stitching attempts to pair up edges of free faces and stitch them together. If no stitching is needed, this API sets a flag in the stitching aggregate attribute indicating this.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_stitch_auto

Function: Healing

Action: Automatically executes the analyze, calculate, and fix stages of the stitching phase.

Prototype:
```
outcome api_hh_stitch_auto (
    BODY* inp_body,          // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/top/body.hxx”
#include ”healhusk/heal_api/heal_api.hxx”
#include ”kernel/kernapi/api/acis_options.hxx”
```

Description: The APIs for the analyze, calculate, and fix stages of stitching are called sequentially. Intelligent tolerances that are recommended by the analyze stage are used in the calculate stage. Stitching attempts to pair up edges of free faces and stitch them together. If no stitching is needed, this API sets a flag in the stitching aggregate attribute indicating this.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_stitch_calculate

Function: Healing

Action: Builds new topology of the body, to the extent possible, to pair up edges of free faces.

Prototype:
```
outcome api_hh_stitch_calculate (
    BODY* inp_body,          // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs the calculate stage of stitching. Stitching attempts to pair up edges of free faces and stitch them together. This process changes the body by doing such things as splitting and merging edges. This API builds the topology of the body to the extent possible. It performs unhook, geometry cleaning, and tries to stitch the unshared edges. It considers all unshared edges for pairing. Individual attributes containing the partner edge information are attached to edges.

The new topology is stored in attributes (attached to individual faces) so that the user may compare the new and the old topology. The fix stage must be used to apply (fix) the new topology to the body.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_stitch_cleanup

Function: Healing

Action: Cleans up the individual healing attributes related to the stitching phase.

Prototype:
```
outcome api_hh_stitch_cleanup (
    BODY* inp_body,          // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_stitch_fix

Function: Healing

Action: Executes the fix stage of stitching.

Prototype:
```
outcome api_hh_stitch_fix (
    BODY* inp_body,          // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API performs the fix stage of stitching. Stitching attempts to pair up edges of free faces and stitch them together. This process changes the body by doing such things as splitting and merging edges.

After the calculate stage, the new topology is placed in the attributes attached to the body and the user can compare the old and new topology. The fix stage applies (fixes) the new topology to the body and stores the old topology in the attributes.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_store_entity_details

Function: Healing

Action: Sets the option to start storing the history of every entity during healing.

Prototype:
```
outcome api_hh_store_entity_details (
    BODY* body,              // input body
    logical option_value,    // store history if TRUE
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API can be used to switch set the option to store the history of main events that every entity undergoes during healing. If the option_value is TRUE, the history is stored in readable text form in attributes.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Read–only

api_hh_wrapup_analyze

Function: Healing

Action: Analyzes coedge pcurve geometry.

Prototype:
```
outcome api_hh_wrapup_analyze (
    BODY*,                   // input body
    AcisOptions* ao = NULL   // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API is used in the geometry building phase. It performs the analyze stage of the wrap up subphase of geometry building. The wrap up subphase recomputes the pcurve geometry of unhealed coedges.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_wrapup_auto

Function: Healing

Action: Automatically executes the analyze and calculate stages of the wrap up subphase of geometry building.

Prototype:
```
outcome api_hh_wrapup_auto (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "healhusk/heal_api/heal_api.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: The APIs for the analyze and calculate stages of the wrap up subphase (which handles pcurve geometry) are called sequentially. Intelligent tolerances that are recommended by the analyze stage are used in the calculate stage.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_hh_wrapup_calc_fix

Function: Healing

Action: Calculates the pcurves in the model.

Prototype:
```
outcome api_hh_wrapup_calc_fix (
    BODY*,                  // input body
    AcisOptions* ao = NULL  // acis options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/body.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: This API is used in the geometry building phase. It performs the calculate and fix stages of the wrap up subphase of geometry building. This API recalculates pcurves on spline surfaces. The new geometry is applied (fixed) to the body and the old geometry is placed on the model in attributes so that the user may compare the new and the old geometry.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: Changes model

api_initialize_healing

Function: Healing, Modeler Control

Action: Initializes the HEAL library.

Prototype:
```
outcome api_initialize_healing ();
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″healhusk/heal_api/heal_api.hxx″
```

Description: This API initializes the HEAL library, which must be done before any other HEAL APIs are called. This API does *not* initialize the healing process. Refer to api_hh_init_body_for_healing for information on initializing the healing process for a body.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: System routine

api_terminate_healing

Function: Healing, Modeler Control

Action: Terminates the HEAL library.

Prototype:
```
outcome api_terminate_healing ();
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "healhusk/heal_api/heal_api.hxx"
```

Description: This API terminates the HEAL library, which must be done after all HEAL APIs are called. This API does *not* terminate the healing process. Refer to api_hh_end_body_for_healing for information on terminating the healing process for a body.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/heal_api/heal_api.hxx

Effect: System routine

is_ATTRIB_HH_AGGR_ANALYTIC

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_AGGR_ANALYTIC.

Prototype:
```
logical is_ATTRIB_HH_AGGR_ANALYTIC (
    const ENTITY* e         // entity to test
    );

int is_ATTRIB_HH_AGGR_ANALYTIC (
    const ENTITY* e         // entity to test
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/attrib/hanalsol.hxx″
#include ″kernel/kerndata/data/entity.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/hanalsol.cxx
heal/healhusk/attrib/hanalsol.hxx

Effect: Read–only

is_ATTRIB_HH_AGGR_GEN_SPLINE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_AGGR_GEN_SPLINE .

Prototype:
```
logical is_ATTRIB_HH_AGGR_GEN_SPLINE (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/attrib/hadvspl.hxx″
#include ″kernel/kerndata/data/entity.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/hadvspl.hxx

Effect: Read–only

is_ATTRIB_HH_AGGR_GEOMBUILD

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_AGGR_GEOMBUILD .

Prototype:
```
logical is_ATTRIB_HH_AGGR_GEOMBUILD (
    const ENTITY* e          // entity to test
    );
```

```
int is_ATTRIB_HH_AGGR_GEOMBUILD (
    const ENTITY* e          // entity to test
    );
```

Includes:

```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/attrib/hmaster.hxx″
#include ″kernel/kerndata/data/entity.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/hmaster.cxx
heal/healhusk/attrib/hmaster.hxx

Effect: Read–only

is_ATTRIB_HH_AGGR_GEOMBUILD_BASE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_AGGR_GEOMBUILD_BASE .

Prototype:

```
int is_ATTRIB_HH_AGGR_GEOMBUILD_BASE (
    const ENTITY* e          // entity to test
    );

logical is_ATTRIB_HH_AGGR_GEOMBUILD_BASE (
    const ENTITY* e          // entity to test
    );
```

Includes:

```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/attrib/aggrgbld.hxx″
#include ″kernel/kerndata/data/entity.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/aggrgbld.hxx

Effect: Read–only

is_ATTRIB_HH_AGGR_ISOSPLINE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_AGGR_ISOSPLINE .

Prototype:
```
logical is_ATTRIB_HH_AGGR_ISOSPLINE (
    const ENTITY* e          // entity to test
    );

int is_ATTRIB_HH_AGGR_ISOSPLINE (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”healhusk/attrib/huvsolv.hxx”
#include ”kernel/kerndata/data/entity.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/huvsolv.cxx
heal/healhusk/attrib/huvsolv.hxx

Effect: Read–only

is_ATTRIB_HH_AGGR_SHARP_EDGE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_AGGR_SHARP_EDGE .

Prototype:
```
logical is_ATTRIB_HH_AGGR_SHARP_EDGE (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”healhusk/attrib/hsharped.hxx”
#include ”kernel/kerndata/data/entity.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/hsharped.hxx

Effect: Read–only

is_ATTRIB_HH_AGGR_SIMPLIFY

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_AGGR_SIMPLIFY .

Prototype:
```
int is_ATTRIB_HH_AGGR_SIMPLIFY (
    const ENTITY* e         // entity to test
    );

logical is_ATTRIB_HH_AGGR_SIMPLIFY (
    const ENTITY* e         // entity to test
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”healhusk/attrib/aggrsimg.hxx”
#include ”kernel/kerndata/data/entity.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/aggrsimg.hxx

Effect: Read–only

is_ATTRIB_HH_AGGR_SIMPLIFY_BASE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_AGGR_SIMPLIFY_BASE .

Prototype:
```
int is_ATTRIB_HH_AGGR_SIMPLIFY_BASE (
    const ENTITY* e          // entity to test
    );

logical is_ATTRIB_HH_AGGR_SIMPLIFY_BASE (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/attrib/agrsimbs.hxx″
#include ″kernel/kerndata/data/entity.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/agrsimbs.hxx

Effect: Read–only

is_ATTRIB_HH_AGGR_WRAPUP

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_AGGR_WRAPUP .

Prototype:
```
logical is_ATTRIB_HH_AGGR_WRAPUP (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/attrib/hsecndry.hxx″
#include ″kernel/kerndata/data/entity.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/hsecndry.hxx

Effect: Read–only

is_ATTRIB_HH_ENT_GEOMBUILD_BASE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_ENT_GEOMBUILD_BASE .

Prototype:
```
int is_ATTRIB_HH_ENT_GEOMBUILD_BASE (
    const ENTITY* e          // entity to test
    );

logical is_ATTRIB_HH_ENT_GEOMBUILD_BASE (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”healhusk/attrib/entgmbld.hxx”
#include ”kernel/kerndata/data/entity.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/entgmbld.hxx

Effect: Read–only

is_ATTRIB_HH_ENT_GEOMBUILD_COEDGE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_ENT_GEOMBUILD_COEDGE .

Prototype:
```
int is_ATTRIB_HH_ENT_GEOMBUILD_COEDGE (
    const ENTITY* e          // entity to test
    );
```

```
logical is_ATTRIB_HH_ENT_GEOMBUILD_COEDGE (
    const ENTITY* e          // entity to test
    );
```

Includes:

```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/attrib/cegmbld.hxx″
#include ″kernel/kerndata/data/entity.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/cegmbld.hxx

Effect: Read–only

is_ATTRIB_HH_ENT_GEOMBUILD_CURVE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_ENT_GEOMBUILD_CURVE.

Prototype:

```
logical is_ATTRIB_HH_ENT_GEOMBUILD_CURVE (
    const ENTITY* e          // entity to test
    );

int is_ATTRIB_HH_ENT_GEOMBUILD_CURVE (
    const ENTITY* e          // entity to test
    );
```

Includes:

```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/attrib/curgmbld.hxx″
#include ″kernel/kerndata/data/entity.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/curgmbld.cxx
heal/healhusk/attrib/curgmbld.hxx

Effect: Read–only

is_ATTRIB_HH_ENT_GEOMBUILD_EDGE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_ENT_GEOMBUILD_EDGE.

Prototype:
```
int is_ATTRIB_HH_ENT_GEOMBUILD_EDGE (
    const ENTITY* e          // entity to test
    );

logical is_ATTRIB_HH_ENT_GEOMBUILD_EDGE (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/logical.h"
#include "healhusk/attrib/edgmbld.hxx"
#include "kernel/kerndata/data/entity.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/edgmbld.hxx

Effect: Read–only

is_ATTRIB_HH_ENT_GEOMBUILD_FACE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_ENT_GEOMBUILD_FACE.

Prototype:
```
int is_ATTRIB_HH_ENT_GEOMBUILD_FACE (
    const ENTITY* e          // entity to test
    );
```

```
logical is_ATTRIB_HH_ENT_GEOMBUILD_FACE (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/attrib/fagmbld.hxx″
#include ″kernel/kerndata/data/entity.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/fagmbld.hxx

Effect: Read–only

is_ATTRIB_HH_ENT_GEOMBUILD_LOOP

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_ENT_GEOMBUILD_LOOP .

Prototype:
```
logical is_ATTRIB_HH_ENT_GEOMBUILD_LOOP (
    const ENTITY* e          // entity to test
    );

int is_ATTRIB_HH_ENT_GEOMBUILD_LOOP (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″healhusk/attrib/lpgmbld.hxx″
#include ″kernel/kerndata/data/entity.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/lpgmbld.cxx
heal/healhusk/attrib/lpgmbld.hxx

Effect: Read–only

is_ATTRIB_HH_ENT_GEOMBUILD_SURFACE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_ENT_GEOMBUILD_SURFACE.

Prototype:
```
logical is_ATTRIB_HH_ENT_GEOMBUILD_SURFACE (
    const ENTITY* e          // entity to test
    );

int is_ATTRIB_HH_ENT_GEOMBUILD_SURFACE (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”healhusk/attrib/surgmbld.hxx”
#include ”kernel/kerndata/data/entity.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/surgmbld.cxx
heal/healhusk/attrib/surgmbld.hxx

Effect: Read–only

is_ATTRIB_HH_ENT_GEOMBUILD_VERTEX

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_ENT_GEOMBUILD_VERTEX.

Prototype:
```
int is_ATTRIB_HH_ENT_GEOMBUILD_VERTEX (
    const ENTITY* e          // entity to test
    );
```

```
logical is_ATTRIB_HH_ENT_GEOMBUILD_VERTEX (
    const ENTITY* e          // entity to test
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”healhusk/attrib/vegmbld.hxx”
#include ”kernel/kerndata/data/entity.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/vegmbld.hxx

Effect: Read–only

is_ATTRIB_HH_ENT_SIMPLIFY_BASE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_ENT_SIMPLIFY_BASE

Prototype:

```
logical is_ATTRIB_HH_ENT_SIMPLIFY_BASE (
    const ENTITY* e          // entity to test
    );

int is_ATTRIB_HH_ENT_SIMPLIFY_BASE (
    const ENTITY* e          // entity to test
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”healhusk/attrib/entsimbs.hxx”
#include ”kernel/kerndata/data/entity.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/entsimbs.cxx
heal/healhusk/attrib/entsimbs.hxx

Effect: Read–only

is_ATTRIB_HH_ENT_SIMPLIFY_FACE

Function: Healing

Action: Determines if an ENTITY is an ATTRIB_HH_ENT_SIMPLIFY_FACE.

Prototype:
```
logical is_ATTRIB_HH_ENT_SIMPLIFY_FACE (
    const ENTITY* e          // entity to test
    );

int is_ATTRIB_HH_ENT_SIMPLIFY_FACE (
    const ENTITY* e          // entity to test
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/logical.h"
#include "healhusk/attrib/entsimg.hxx"
#include "kernel/kerndata/data/entity.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: healhusk

Filename: heal/healhusk/attrib/entsimg.cxx
heal/healhusk/attrib/entsimg.hxx

Effect: Read–only