A Native AP-210 Prototype 3-D Package Modeler

Seth, A.[1], Thurman, T.R.[2], Mukhopadhyay, D.[1], Tang, D.[3], Stori, J.A.[4], Ferreira, P.M[1]

[1]Dept. of Mechanical Sc. & Engg., University of Illinois at Urbana-Champaign
1206 West Green Street, MC-244, Urbana, IL, 61801, U.S.A.

[2]Rockwell-Collins, Inc.
400 Collins Road N.E., Cedar Rapids, IA, 52498, U.S.A.

[3]Intel, Inc.
Chandler, AZ, U.S.A.

[4]SFM Technology, Inc.
202, W. Green Street, Urbana, IL, 61801, U.S.A.

Executive summary:

- Present results of one-year effort to develop a prototype of a native AP-210 package modeler
- Problems with the ‘conversion’ approach
  - Incompleteness of input model
  - Trying to infer information leads to problems
- Advantages of a ‘native approach’
  - Combining the
    - strong user interface and parametric template capabilities of a commercial mechanical modeling environment with the
    - comprehensive and standardized data format of STEP AP210 into a single design environment
  - Allows for the generation and use of package libraries

Executive summary (cont'd):

- To develop a design environment in which the information needed for populating the AP-210 model is explicitly identified during the design process
  - Used Parametric Technologies Corporation's *Pro/Engineer* tool
  - *Designed model templates* in Pro/E to guide user and allow for the systematic organization of model data for easy access during the generation of the AP210 file
  - Developed a suite of routines assembled under what we call the *'Geometric Bridge'* that uses PTC's Java interface (J-Link)
  - Developed AP-210 file generator (that uses LK-Soft's *JSDAI* a java-based standard data access interface library for STEP)

Outline:

- Project Objective
- Emphasis and Limitations
- System architecture
- Implementation Approach
- User-interface
- Implementation
  - Data structure of geometric bridge
  - Solid Model Exploration and Data population
  - STEP AP-210 File Generation Procedure
- Test cases and Results

Project Objective:

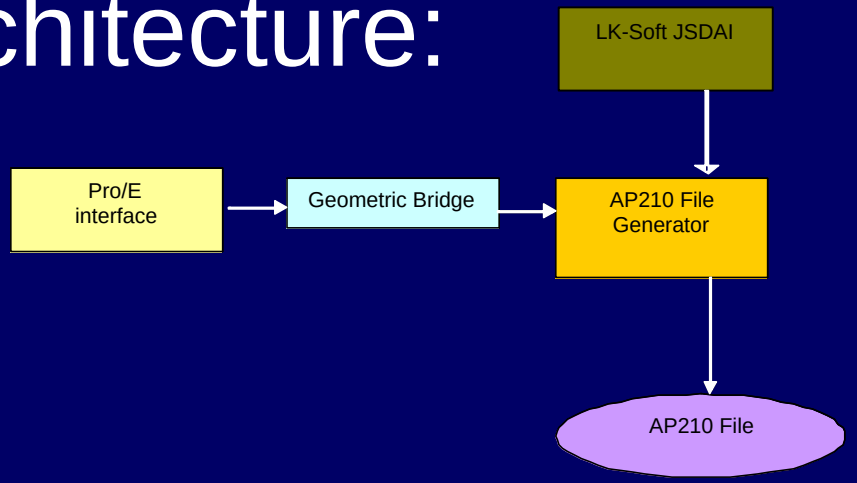
- To develop a package modeler native to STEP AP-210
  - 'Input' model needs
    - many geometric constructs
      - Package body
      - Leads
      - Seating plane
      - Dimensions
      - Footprint
      - …
    - functional capabilities of features called for in the standard
- 'Parametric' capability based on package technology provides for rapid generation of 'library' objects
  - can be used to facilitate the population of a component library with AP210 packaged parts

```
(* SCHEMA AP210_ARM; *)
  ENTITY package
    SUPERTYPE OF (altered_package)
    SUBTYPE OF (physical_unit_usage_view);
      mounting_technology                          : mounting_technology_type;
      maximum_seating_plane_installation_offset    : OPTIONAL length_data_element;
      surface_mount_lead_form                      : OPTIONAL lead_form_type;
      nominal_mounting_lead_pitch                  : OPTIONAL length_data_element;
      nominal_mounting_lead_span                   : OPTIONAL length_data_element;
      maximum_body_height_above_seating_plane      : length_data_element;
      maximum_body_height_below_seating_plane      : OPTIONAL length_data_element;
      maximum_body_clearance_above_seating_plane   : OPTIONAL length_data_element;
      maximum_body_clearance_below_seating_plane   : OPTIONAL length_data_element;
      minimum_body_clearance_above_seating_plane   : OPTIONAL length_data_element;
      minimum_body_clearance_below_seating_plane   : OPTIONAL length_data_element;
      maximum_lead_length_below_seating_plane      : OPTIONAL length_data_element;
      least_lead_length_below_seating_plane        : OPTIONAL length_data_element;
    DERIVE
      maximum_package_total_vertical_extent : length_data_element :=
max_data_element(add_data_element(maximum_body_height_above_seating_plane,
 maximum_body_height_below_seating_plane),
 add_data_element(maximum_body_height_above_seating_plane,
 maximum_lead_length_below_seating_plane));
      cutout_required                          : BOOLEAN :=
(maximum_body_height_below_seating_plane\data_element.measure_value > 0.0);
      maximum_installed_height                 : length_data_element :=
add_data_element(maximum_body_height_above_seating_plane,
 maximum_seating_plane_installation_offset);
    INVERSE
      body                                     : SET[0:1] OF package_body FOR associated_definition;
      package_seating_plane                    : non_feature_shape_element FOR scope;
      interface_plane                          : SET[0:?] OF non_feature_shape_element FOR scope;
      package_accesses                         : SET[0:?] OF package_terminal FOR associated_definition;
      package_polarity_indication              : SET[0:1] OF part_feature FOR associated_definition;
      package_primary_orientation_feature      : SET[0:1] OF part_feature FOR associated_definition;
      package_secondary_orientation_feature    : SET[0:1] OF part_feature FOR associated_definition;
      package_tertiary_orientation_feature     : SET[0:1] OF part_feature FOR associated_definition;
      of_datum_reference_frame                 : SET[0:?] OF non_feature_shape_element FOR scope;
```

Emphasis and Limitations:

- As with any prototype implementation, emphasis on
  - demonstrating solutions to the major obstacles in a full-scale implementation
  - generating the geometric entities in the AP210 file
    - therefore, certain other essential data for a valid AP210 file has been populated with default values
- Have limited our focus to faceted (planar) solid models only
- Based on working draft (WD) 48_8 of the AIM schema

System Architecture:



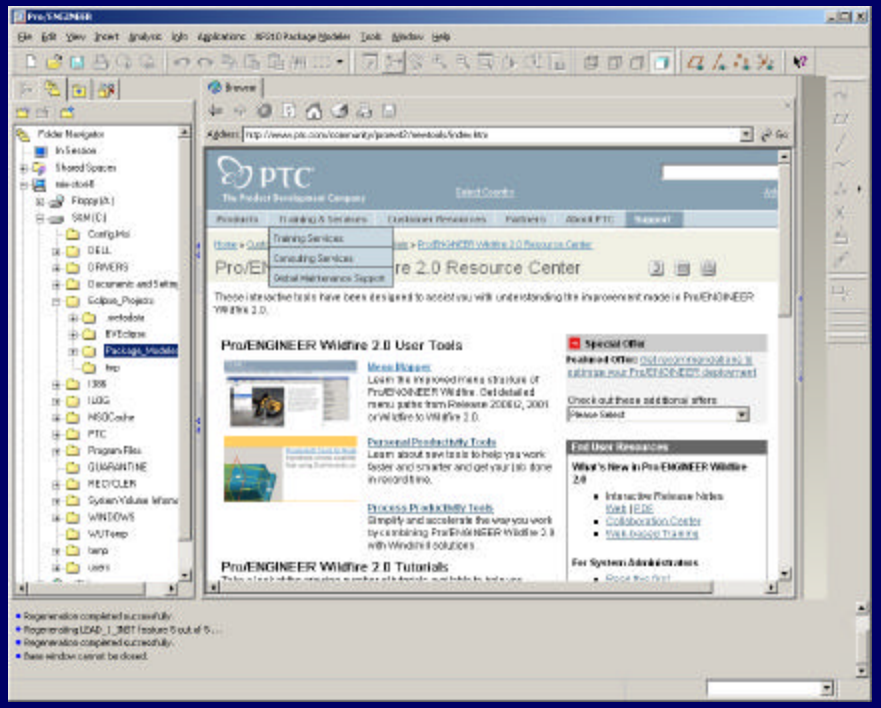
- ‘High-level’ architecture
- **Pro/E interface:** Advantages inherent in the Pro/Engineer® system
  - *geometric modeling* and *user interface capabilities*
  - availability of *non-manifold operations*
  - *Java* API access
  - *Parametric design philosophy*
- **AP-210 File generator:** Populates the AP-210 Building Factory classes
- **STEP model management system:** LK-Soft JSDAI
- **Geometric bridge:** Links together the Pro/Engineer interface and the AP-210 Building factory classes

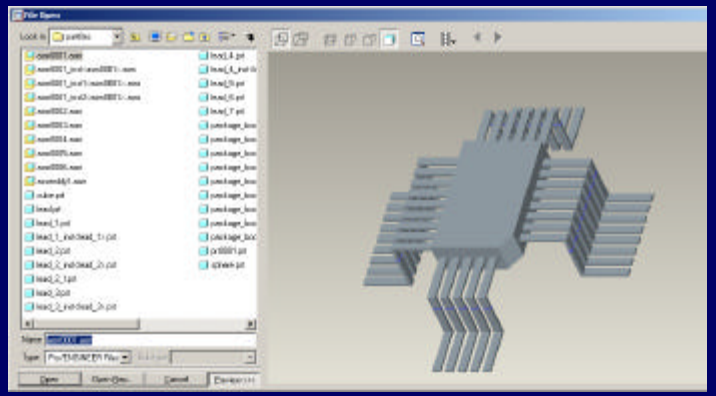
Implementation Approach:

- Concept to allow the use of pre-defined library templates that are easily parameterized
- User can also develop his / her own templates for new package ‘styles’
  - Starting from scratch
  - Following some minimal guidelines
  - Thus building a library of customizable packages

User Interface Walk-Through:

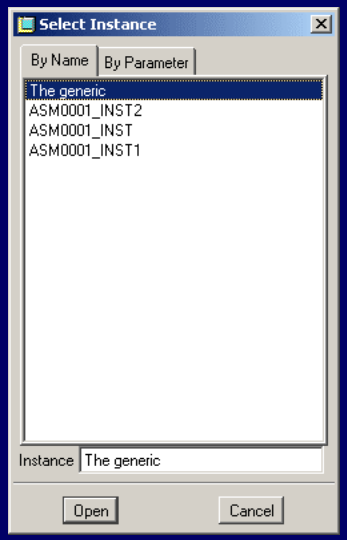
- Executing the program brings up the Pro/Engineer start-up window
- User can select one of the pre-defined library objects from the package library directory

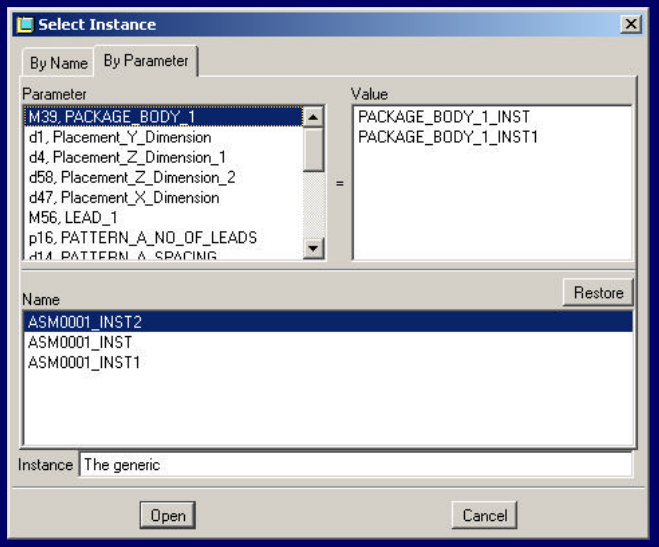




User Interface Walk-Through:

- Selection of a pre-defined package style results in the request for a choice between the *generic* (and therefore, user-customizable) version of the library object, and instances of the library object that were defined in an earlier session





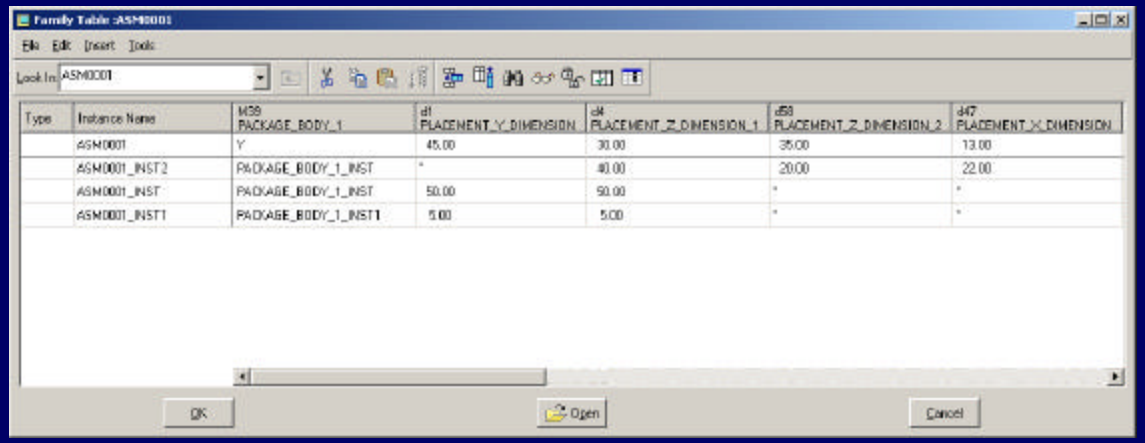
User Interface Walk-Through:

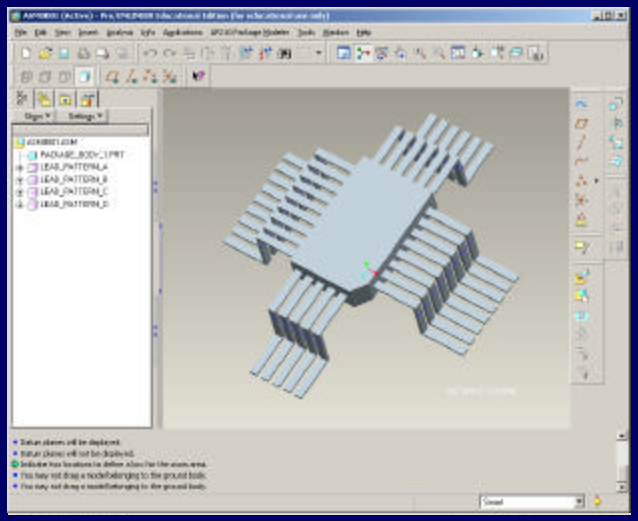
- **Scenario 1:** Generate an AP-210 native file for a previously defined package assembly instance
- **Scenario 2:** Open the *generic* version of the library package-style, view and modify parameters for the package library object to create a unique instance
- **Scenario 3:** Define a *new library object*, must follow a minimal set of design guidelines in order to provide the necessary information required to populate a complete and valid STEP AP-210 file

User Interface Walk-Through:

- **Scenario 2:**
  - Concept of a Family Table:
    - Excel-like spreadsheet,
      - Rows represent different parametric instances of the same generic package assembly
      - Columns represent different parameters defined for the assembly
      - Individual cells containing values for those parameters
    - Interface allows the user to
      - edit or add parameter definitions and values
      - to add, remove, modify and open for viewing existing instances
    - Multiple instances of the same package template can
      - use different instances of a similarly generically defined package body or lead
      - suppress or resume package assembly features
      - change package dimensions, lead pitches, lead counts etc.

User Interface Walk-Through:





PACKAGE_BODY_INSTANCE	Parametric variant of package body template to be used in assembly
PKG_BODY_X_PLACEMENT	X placement of the package body w.r.t. the assembly co-ordinate system
PKG_BODY_Y_PLACEMENT	Y placement of the package body w.r.t. the assembly co-ordinate system
PKG_BODY_Z_PLACEMENT	Z placement of the package body w.r.t. the assembly co-ordinate system
LEAD_INSTANCE	Parametric variant of lead template to be used in assembly
LEAD_GROUP_1_NO_OF_LEADS	Lead count for the first group
LEAD_GROUP_1_SPACING	Lead pitch for the first group
LEAD_GROUP_1_X_PLACEMENT	X placement of the right-most lead from the X-parallel face of the package body
LEAD_GROUP_1_Y_PLACEMENT	Y placement of the leads from the top face of the package body
LEAD_GROUP_2_NO_OF_LEADS	Lead count for the first group
LEAD_GROUP_2_SPACING	Lead pitch for the first group
LEAD_GROUP_2_Z_PLACEMENT	Z placement of the right-most lead from the Z-parallel face of the package body
LEAD_GROUP_2_Y_PLACEMENT	Y placement of the leads from the top face of the package body
LEAD_GROUP_3_NO_OF_LEADS	Lead count for the second group
LEAD_GROUP_3_SPACING	Lead pitch for the second group
LEAD_GROUP_3_X_PLACEMENT	X placement of the right-most lead from the X-parallel face of the package body
LEAD_GROUP_3_Y_PLACEMENT	Y placement of the leads from the top face of the package body
LEAD_GROUP_4_NO_OF_LEADS	Lead count for the third group
LEAD_GROUP_4_SPACING	Lead pitch for the third group
LEAD_GROUP_4_Z_PLACEMENT	Z placement of the right-most lead from the Z-parallel face of the package body
LEAD_GROUP_4_Y_PLACEMENT	Y placement of the leads from the top face of the package body
SEATING_PLANE_Y_PLACEMENT	Y placement of the seating datum plane from the top face of the package body
MOUNTING_TECHNOLOGY	Mounting style
MAX_HT_ABOVE_SEATING_PLANE	Maximum height of package body above the seating datum plane

User Interface Walk-Through:

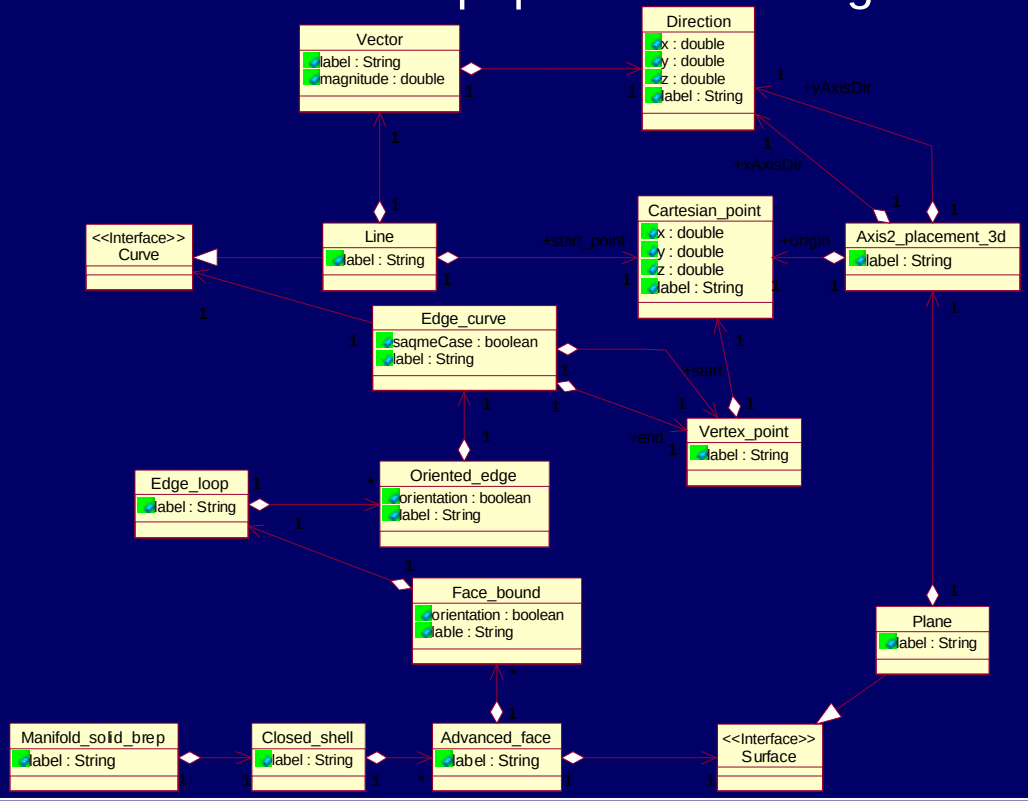
- **Scenario 3:**
  - Package assembly along with package body must be designed to have y-axis of default PTC co-ordinate system to be normal to the top face of package body
  - Package assembly must be designed as Pro/E mechanical assembly and contain package details as described underneath:
  - *Package body:*
    - must be designed as a Pro/E part
    - must be named with a string beginning with PACKAGE_BODY_
    - cannot contain special features like import or surface features
  - *Package terminals:*
    - must be all be of a single template design
    - are segregated into the primary reference terminal (‘pin 1’) and other pins (transformations of the primary lead).
    - only one lead that needs to be designed as a Pro/E part
    - named with a string beginning with LEAD_
    - leads numbered automatically starting from the primary reference lead as pin 1
    - can be placed in the package assembly in patterns using Pro/Engineer’s patterning capabilities

User Interface Walk-Through:

- **Scenario 3 (cont'd):**
  - *Primary orientation feature*
    - is a designated surface of the package body
    - is identified to the system through the creation of a datum plane concurrent with the desired surface of the package body named PRIMARY_ORIENTATION_DATUM
  - *Parameters*
    - The package assembly must minimally define two parameters on the assembly object, with string values attached to both, named:
      - MOUNTING_TECHNOLOGY
      - MAX_BODY_HT_ABOVE_SEATING_PLANE
  - *Seating plane*
    - must be defined as a datum plane parallel to the default asm_top datum plane in the assembly and must be named SEATING_PLANE.
  - *Footprint*
    - must be defined by creating a cross-section of the package assembly with the seating plane and by using this section to define the intersection datum curves.
    - The datum curves must be encapsulated in a composite curve called FOOTPRINT

Implementation: Data structure of geometric bridge

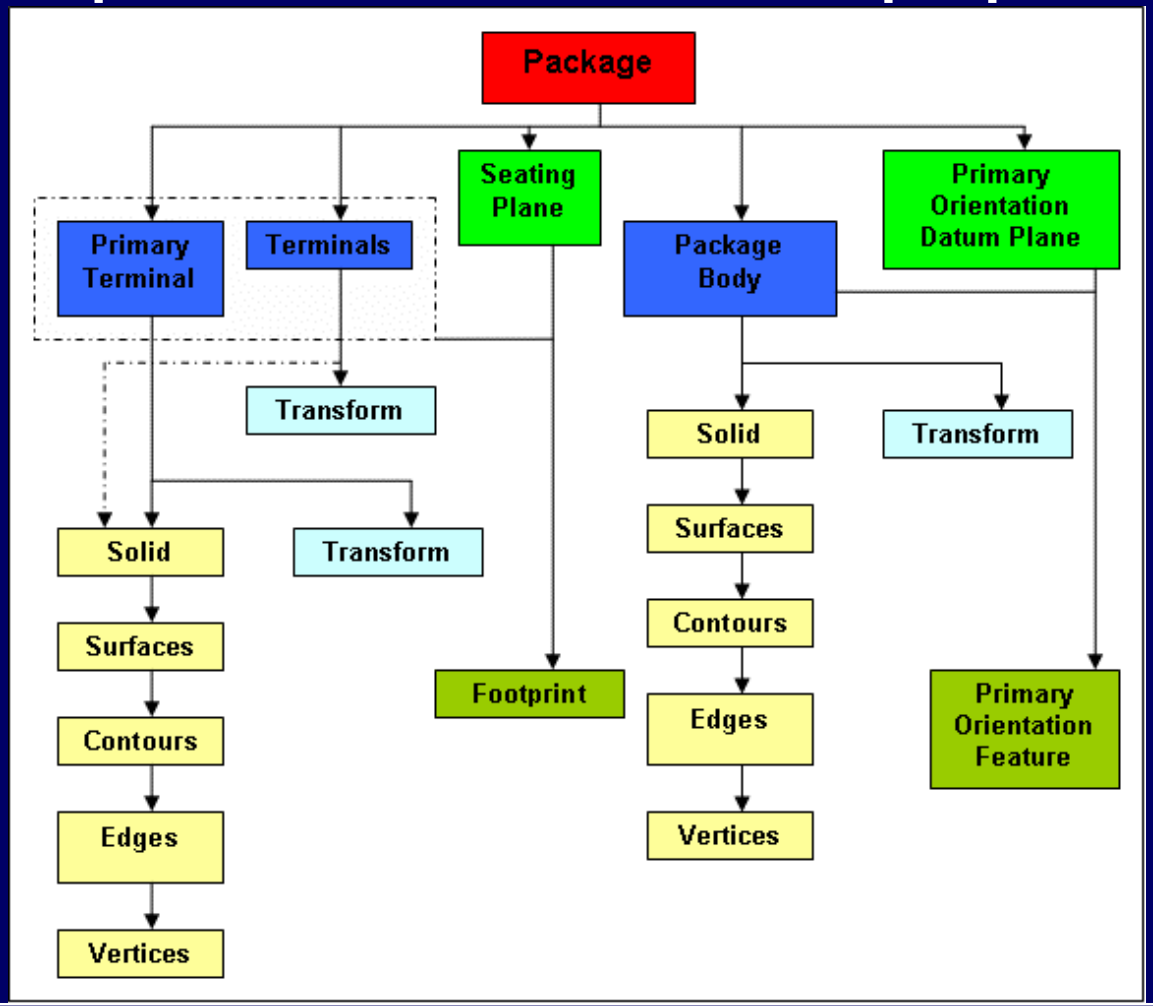
- Data structure designed to closely match the geometric structures used by AP-210 in order to facilitate JSDAI population of the geometric entities



Implementation: Solid Model Exploration and Data population

- Pro/Engineer's Java API J-Link® used to explore geometry of assembly contained in Pro/E and pass it to AP-210 building factory methods
- Non-primary *terminals* have their own transformation matrix but point to the same solid geometry as that of *primary terminal*.
- *Terminals* intersect with the *seating plane* to form the *footprint* for the package. The *footprint* is specified by a list of closed contours that lie on the *seating plane*.
- Surface of the *package body* that is coincident with the *primary orientation datum plane* forms the *primary orientation feature*.

Implementation: Solid Model Exploration and Data population

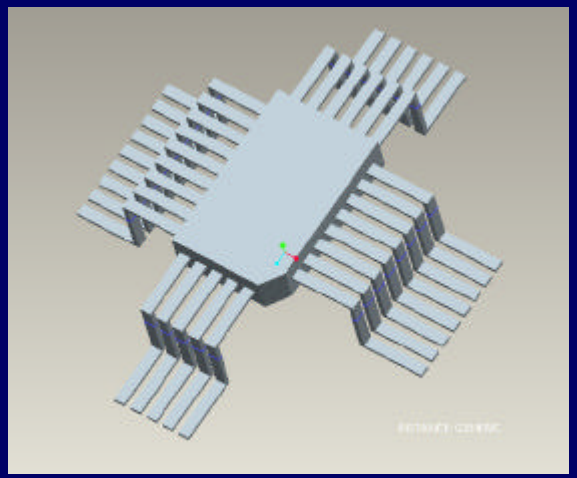


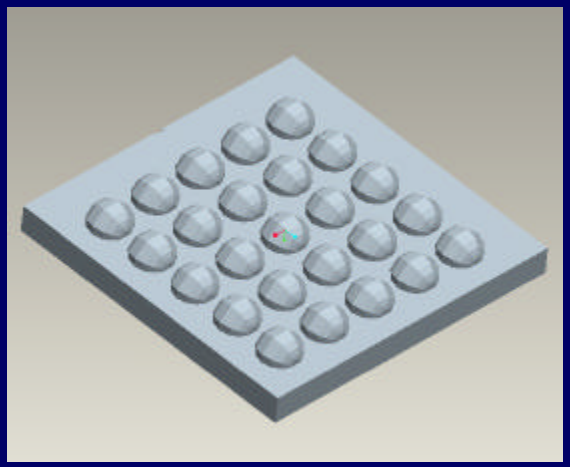
Implementation: STEP AP-210 File Generation Procedure

- Have employed LK-SOFT JSDAI APIs to build the AP210 file generation factory classes
- Solid model exploration and data filling
  ? data stored in objects in geometric bridge
- these objects and their inter-relationships are utilized by the AP-210 file generator to generate the file

Test cases and Results:

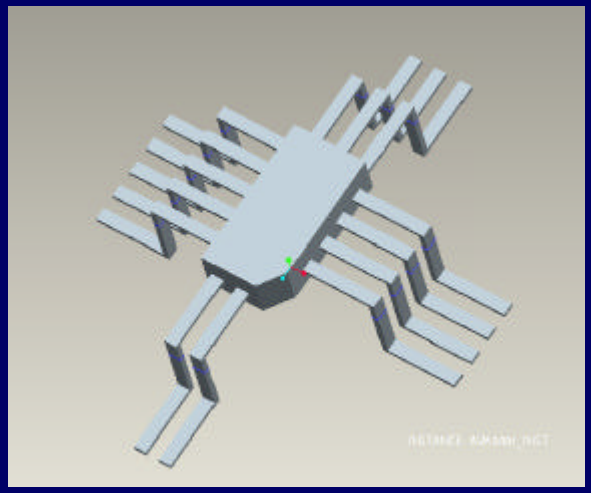
- Developed some prototype generic package assemblies using the tool in order to demonstrate and validate its capabilities

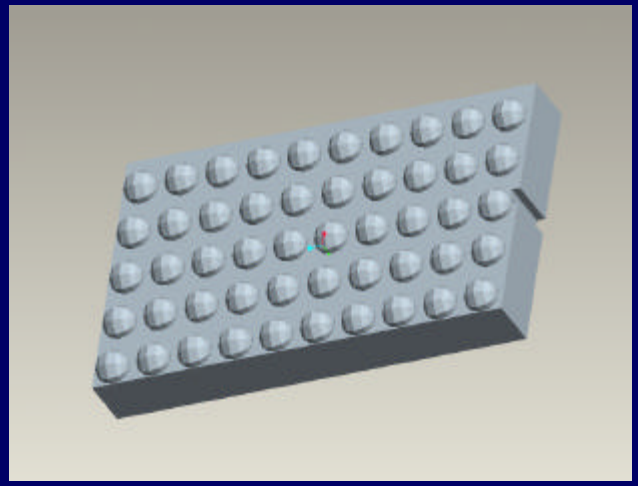




Test cases and Results:

- For both of these package assemblies, several parametric variants were created



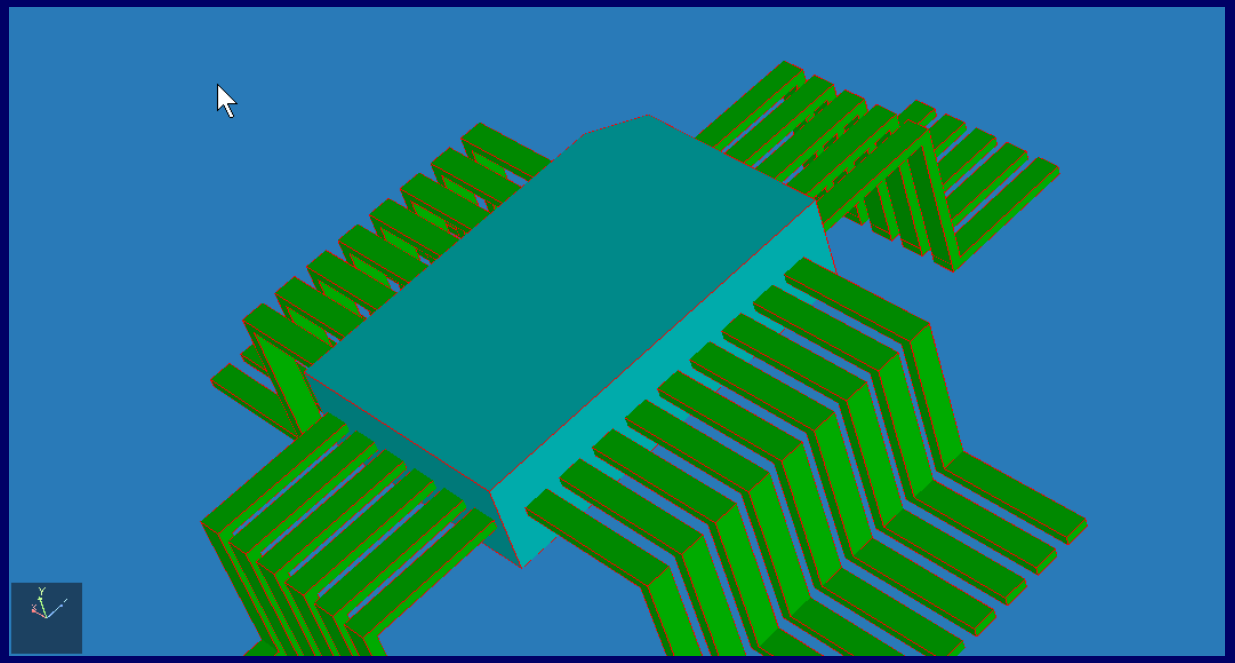


- Generated AP210 file for is plain text output

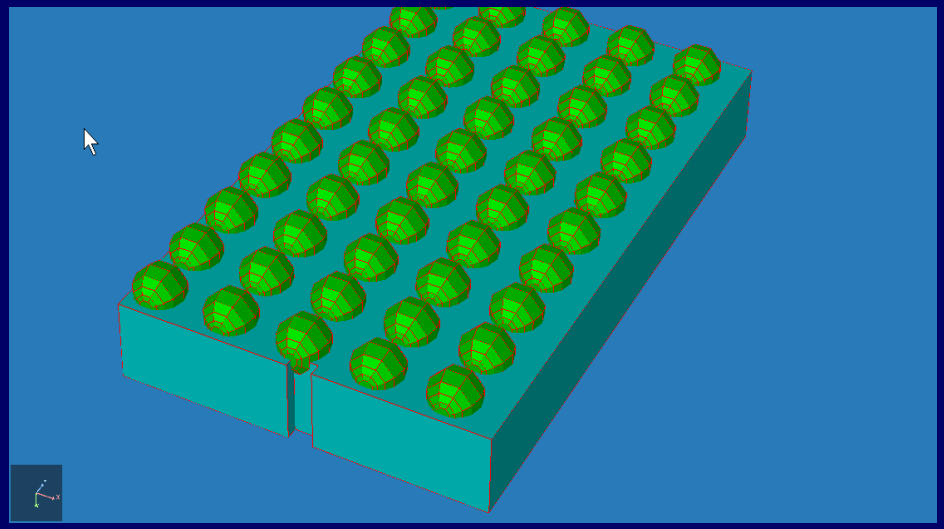
Validation and Tests

- ✓ PD-Tec Instance Checker
- ✓ Expresso Express Engine
- ✓ STEP-Book (footprint, seating plane remain)
- ✓ Pro/Engineer AP-203 reader
- ✓ EPM Technology Express Data Manager

STEP-Book Validation:



STEP-Book Validation:



Conclusion:

- Prototype native AP-210 package modeler has been developed leveraging the
  - strengths of the geometric modeling and user interface capabilities of Pro/E
  - AP-210 population routines of the LK-Soft JSDAI
- Hope to provide designers with the
  - ability to directly generate AP-210 native files
    - without having to go through non-standard third-party file formats
    - capture all needed input in the design environment

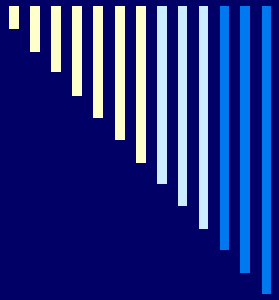
Next steps… :

- Implementation: curves and surfaces
- Full compliance with validation tools
- Improved UI
- Other data to be captured from designer
- More intricate package model
- Conform to latest version of the published AP210 standard

Acknowledgements:

- Tom Thurman, Michael Benda, Rockwell Collins
- LK-Soft GmbH
- PDTec GmbH
- Doug Cheney, ITI-Transcendata
- Peter Denno, NIST
- PTC Tech. Support
- …

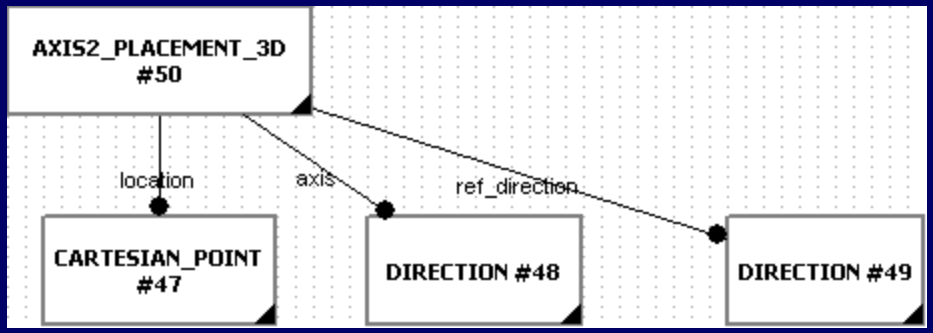
*This project was supported by a grant received from NIST



Thank You

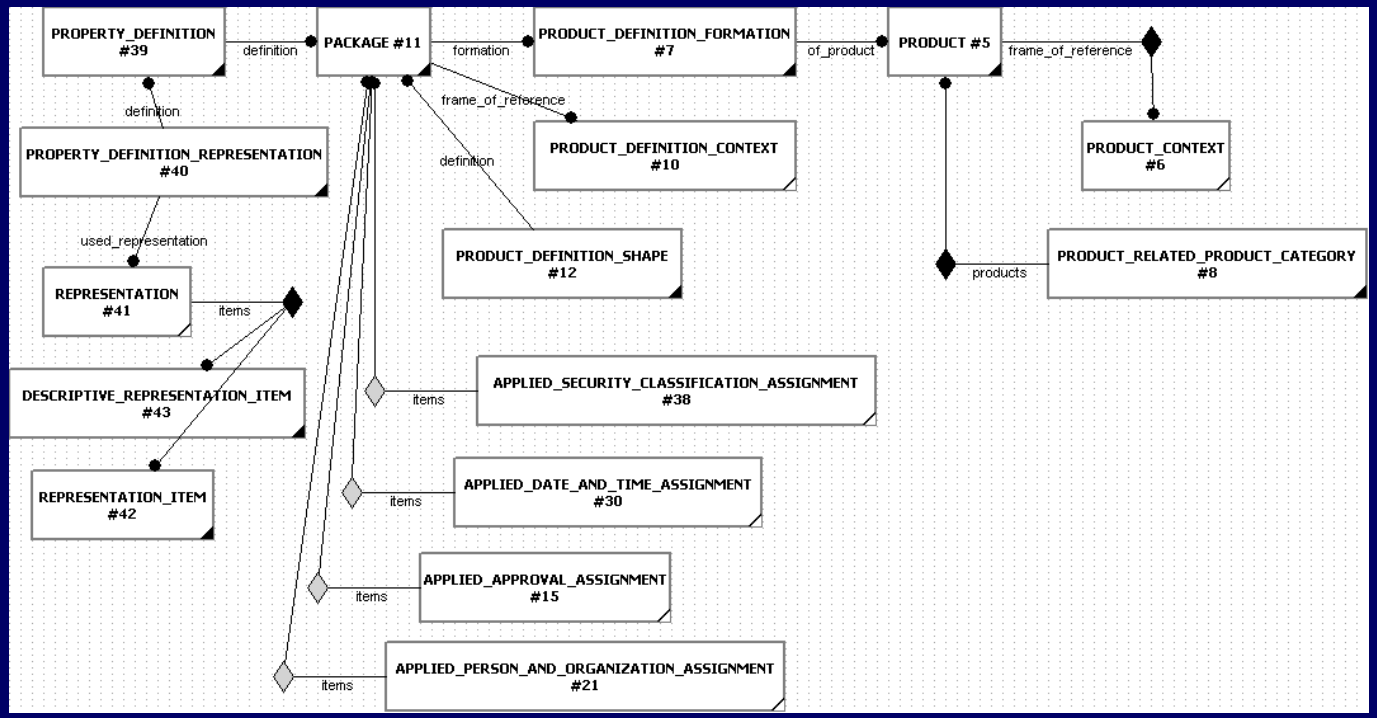
AP-210 Entities populated:

- Population of the Axis2_placement_3d structure



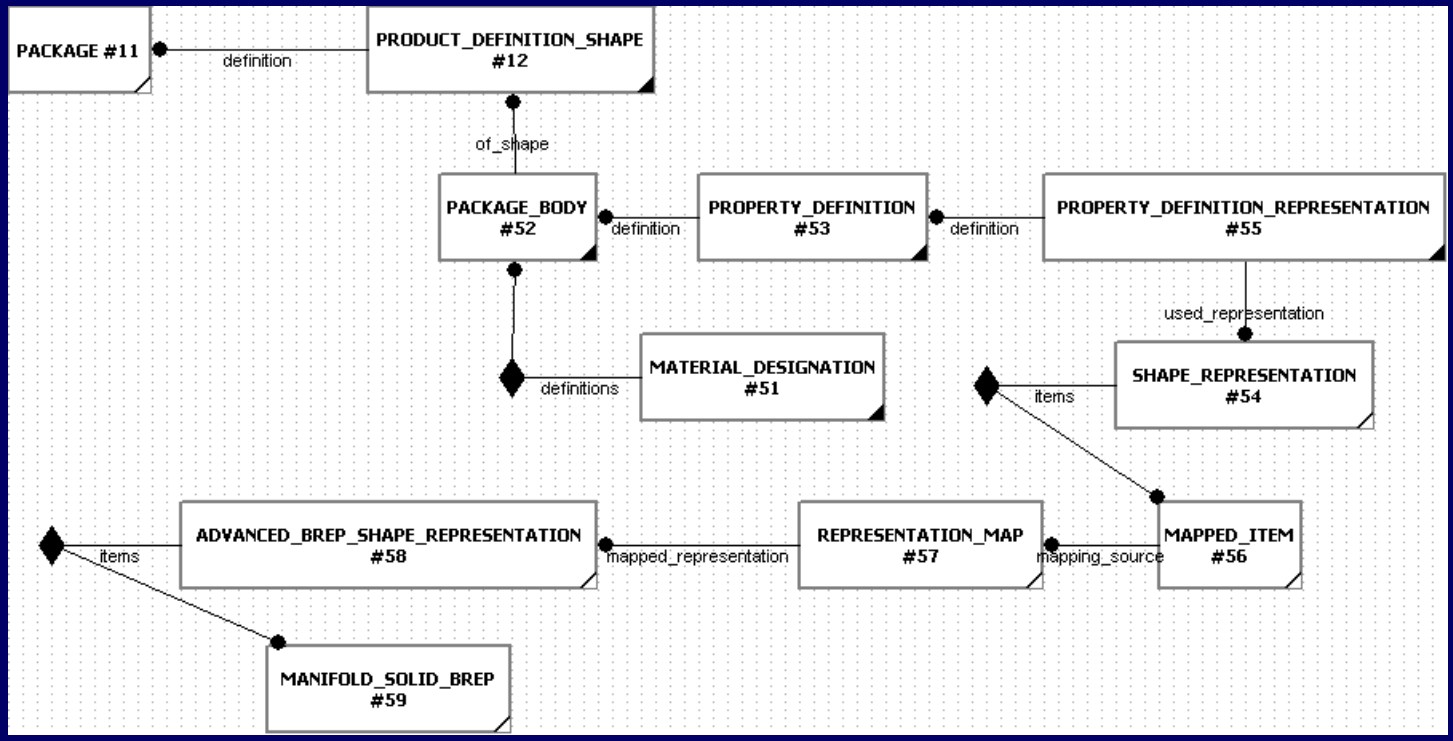
AP-210 Entities populated:

- Population of the Package structure along with other high-level information



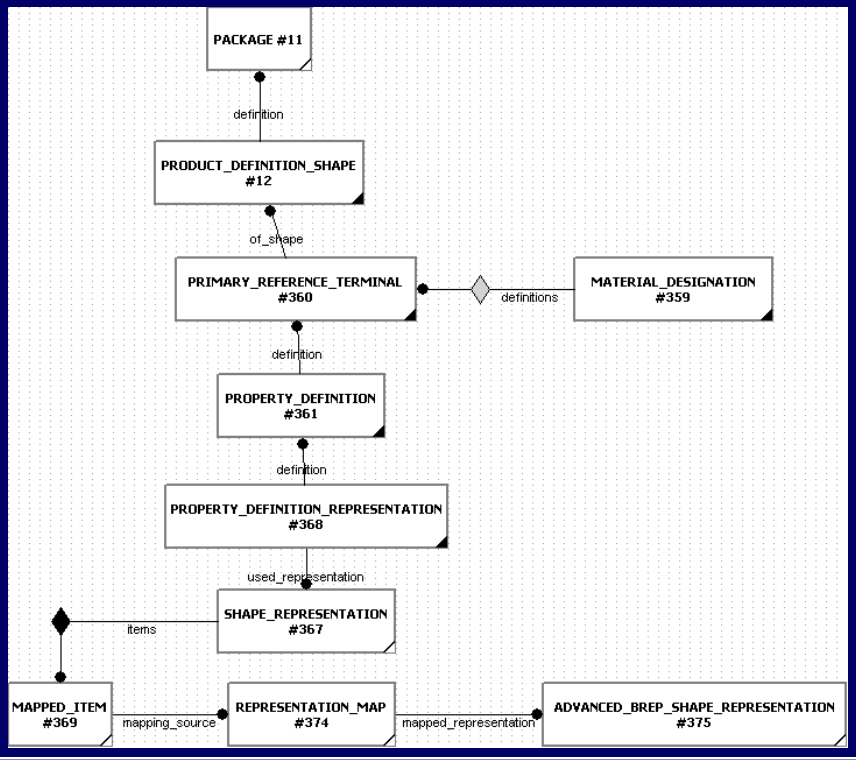
AP-210 Entities populated:

- Population of the Package body structure



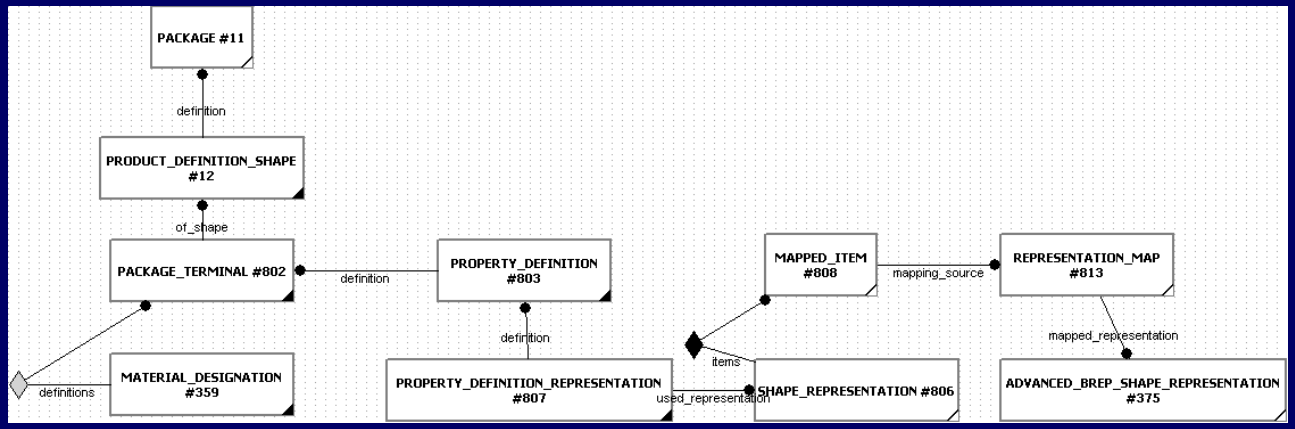
AP-210 Entities populated:

- Population of the Primary reference terminal structure



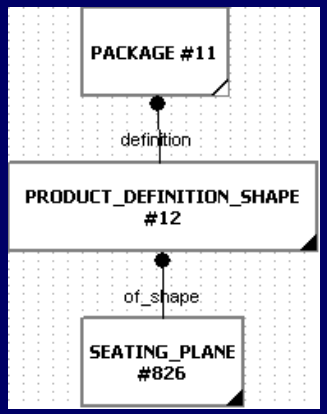
AP-210 Entities populated:

- Population of the Package terminal structure



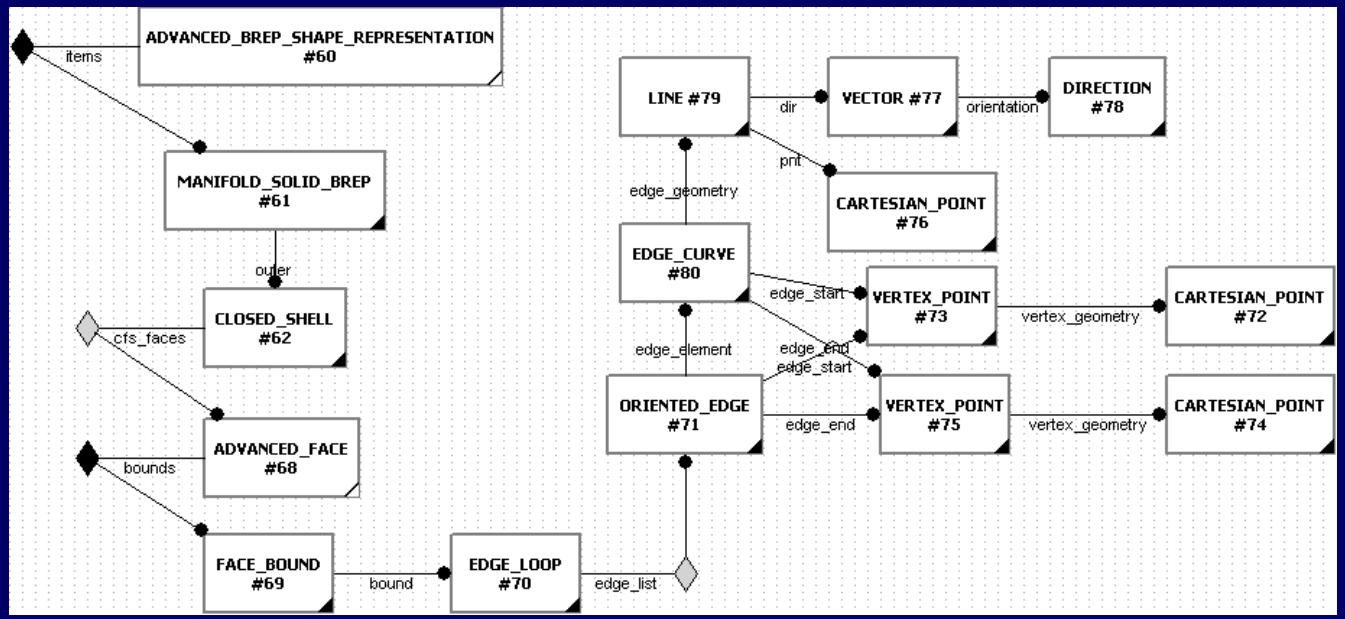
AP-210 Entities populated:

- Population of the Seating plane structure



AP-210 Entities populated:

- Population of the Boundary representation structure



AP-210 Entities populated:

- Population of the Footprint structure