NX Knowledge Fusion Tips and Tricks

You might want to know this!

Raymond Kok

UGS NX development
Overview

The **good** news:

- No PowerPoint slides – well, some slides but mostly demonstrations.
- It will illustrate how easy Knowledge Fusion can be!

The **bad** news:

- Demonstrating in front of a technical crowd can always be kinda tricky!
- I always like to change my demo 5 minutes before the show.
The Power of Knowledge Fusion

**JOB COUNSELING**

We'll need to disguise the fact that you're a moron.

**IRONICALLY, THE BEST WAY IS TO**

Become an expert in something called "knowledge management."

**WE MUST DEVELOP KNOWLEDGE OPTIMIZATION INITIATIVES TO LEVERAGE OUR KEY LEARNINGS.**

SMART.
Use of `ug_feature_set` for KF application development

- Easy to use KF system class which allows you to bundle certain features as one node in the PNT;
- Inheritance of `ug_feature_set` will make your complete KF application behave like a feature!

Use of `%nx_application` for KF application development

- Easy way to get full feature functionality of your KF application;
- Introduction of the concept of “slave” features which allow KF users to dictate the order of update.
Use of `nx_constraint` if you are interested in applying mating conditions as part of your KF application.

```c
#+
DefClass: nx_constraint( nx_constraint_modifier nx_constraint_host *nx_constraint);
#-
DefClass: *nx_constraint (nx_axobject);

+#
Constraint references in NX
Represented as a List of List elements.
Each element consists of a List containing a sequence of name-value pairs, as follows:
{
    # The movable object
    movable_object, <host pointer or instance>,
    # The geometry
    geometry, <host pointer or instance>,
    # Use axis of geometry
    uses_axis, Boolean,
    # Use geometry indirectly
    is_indirect, Boolean,
    # The help point
    help_point, Point,
};
#-
(Canonical List Parameter Modifiable) references: {};
```
Use of `ug_collector` and `ug_section` to enable Selection Intent for Knowledge Fusion;

Selection Intent lets you select and group multiple curves, edges and faces into collections with rules that define how a feature can use them. You choose which rules to use based on what you intend the feature to do.

This also works in combination with UDFs – the use of selection intent will make the use of UDFs in your KF application quite different!
Modeling of holes in NX is not the most easiest task unfortunately (especially if you want them threaded);

Use of `ug_simple_hole`, `ug_countersunk_hole`, and `ug_counterbore_hole` to speed up day-to-day modeling!
The new RuleManager NX/Open class will give the user all functionality from the NX User Interface.

This new functionality will also cover most of the functions available today from NX/Open API [UF_KF] and the functions available from the Knowledge Pipeline.