Sketch Input of Engineering Solid Models

4. Open problems

Pedro Company

Peter Varley

Introduction

We shall introduce some long term open problems in the sector of SBM tools:

1. Making virtual paper and pencil more usable than actual paper and pencil
2. Interpreting annotated engineering sketches
3. Creating assemblies from sketches

Virtual vs. actual paper and pencil

User studies assert that current SBM tools

✗ Are still less usable than paper and pencil sketches

✗ Do not possess significantly improved functionality

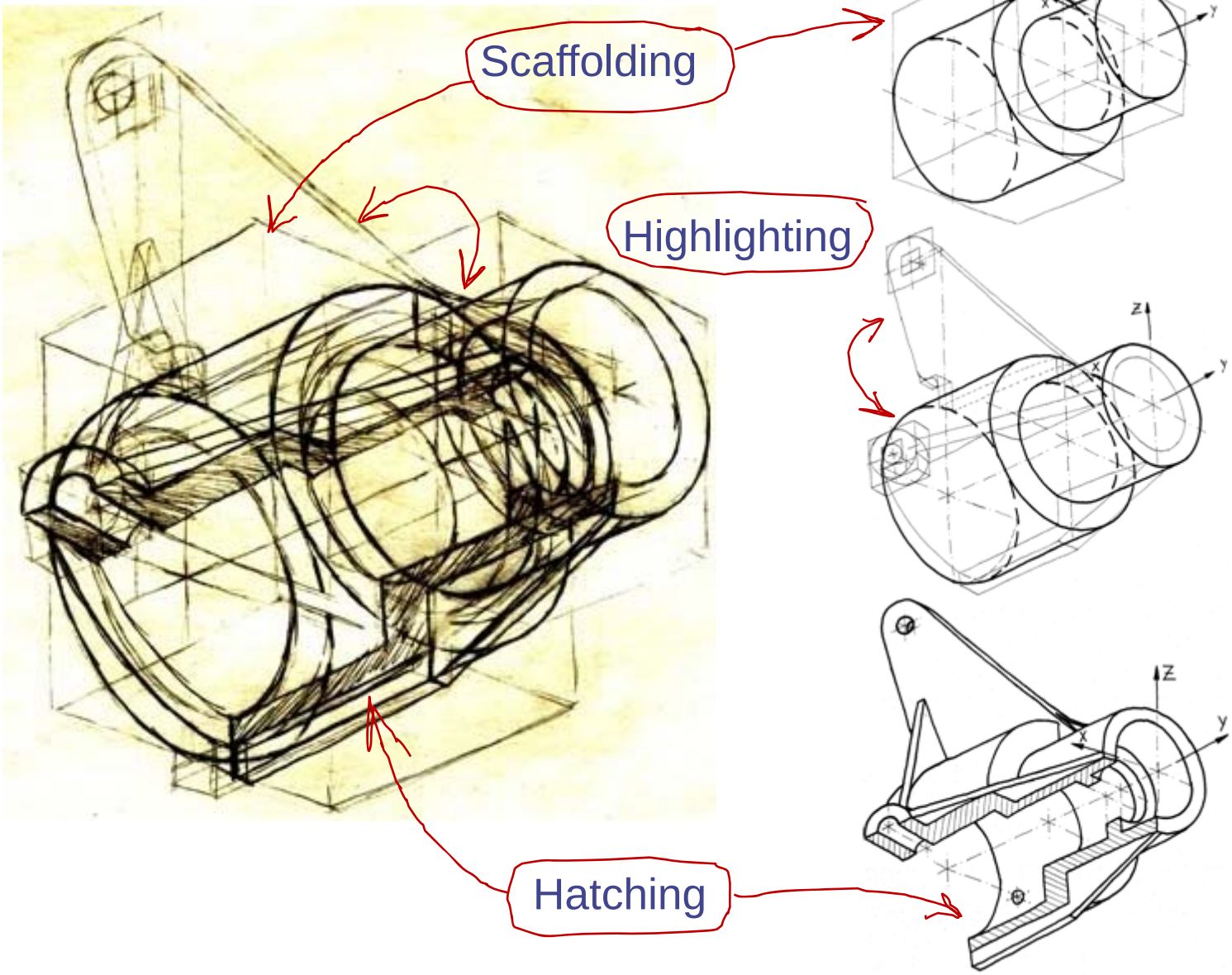
The “hardware” of paper and pencil sketching is **simple** …

… but its operation is sophisticated …

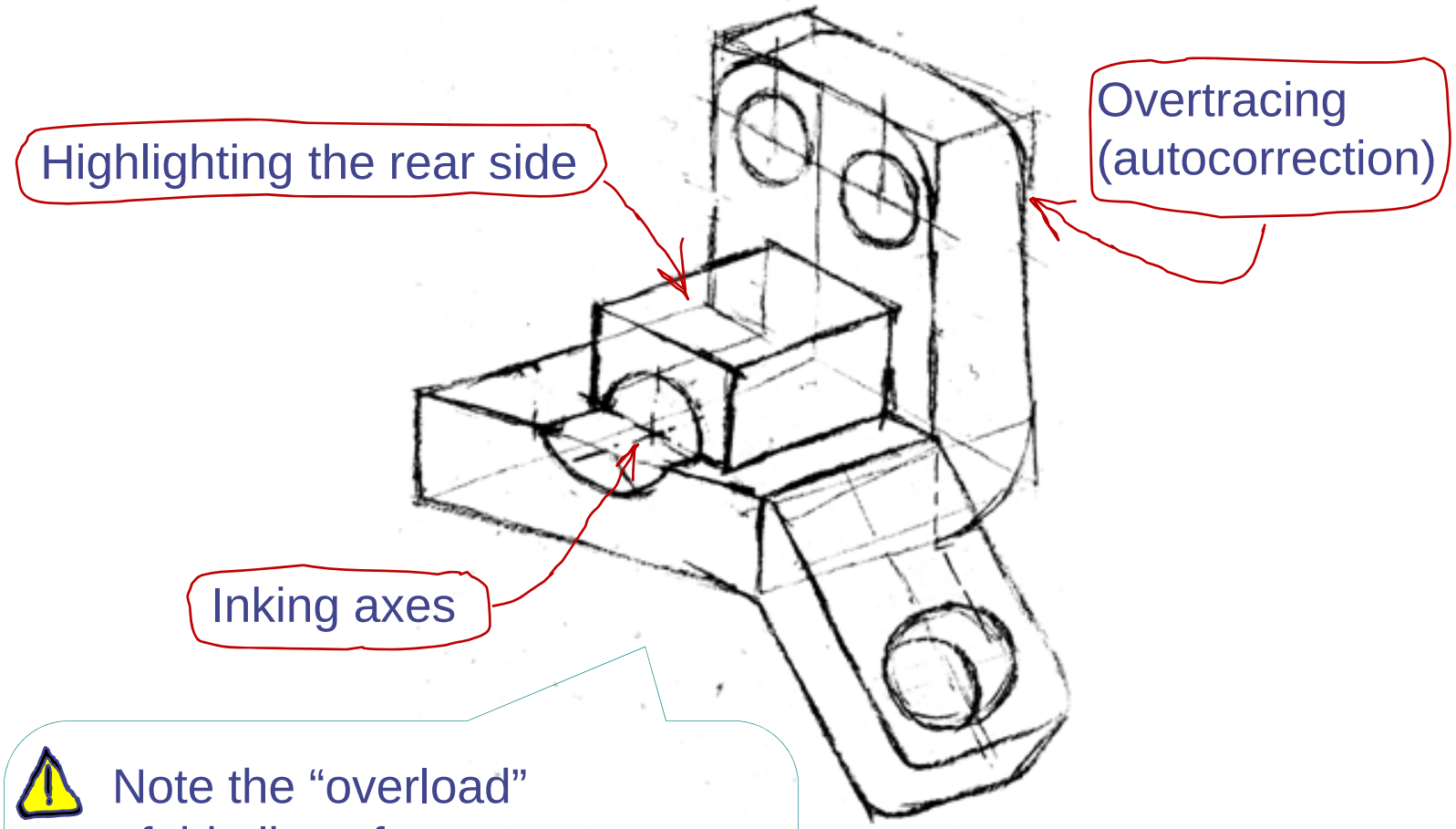
… as paper and pencil sketching is **multimodal**!

Let us see some examples!

Virtual vs. actual paper and pencil



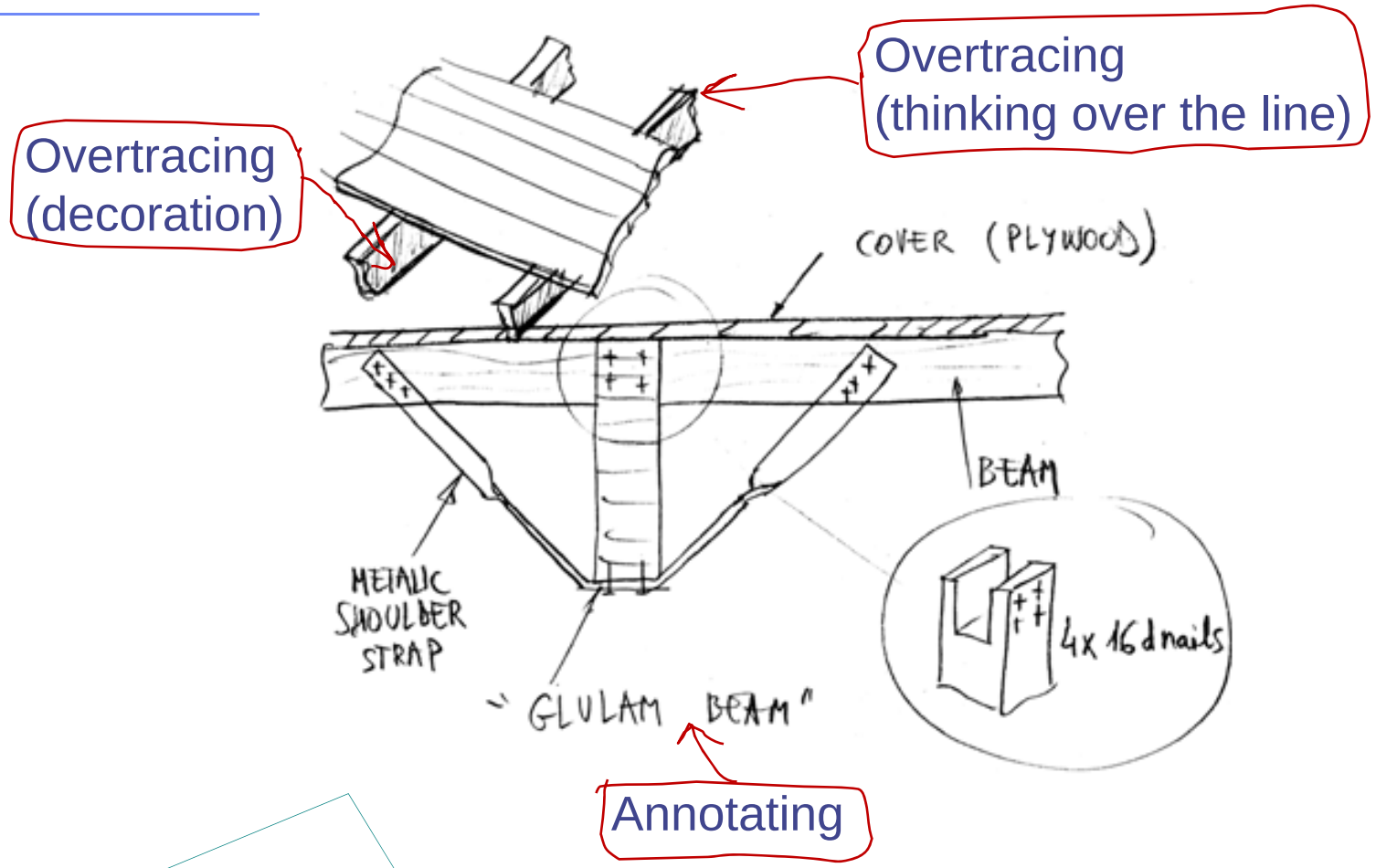
Virtual vs. actual paper and pencil



Note the “overload”
of thin lines for:

- ✓ scaffolding
- ✓ highlighting the rear side
- ✓ inking axes

Virtual vs. actual paper and pencil

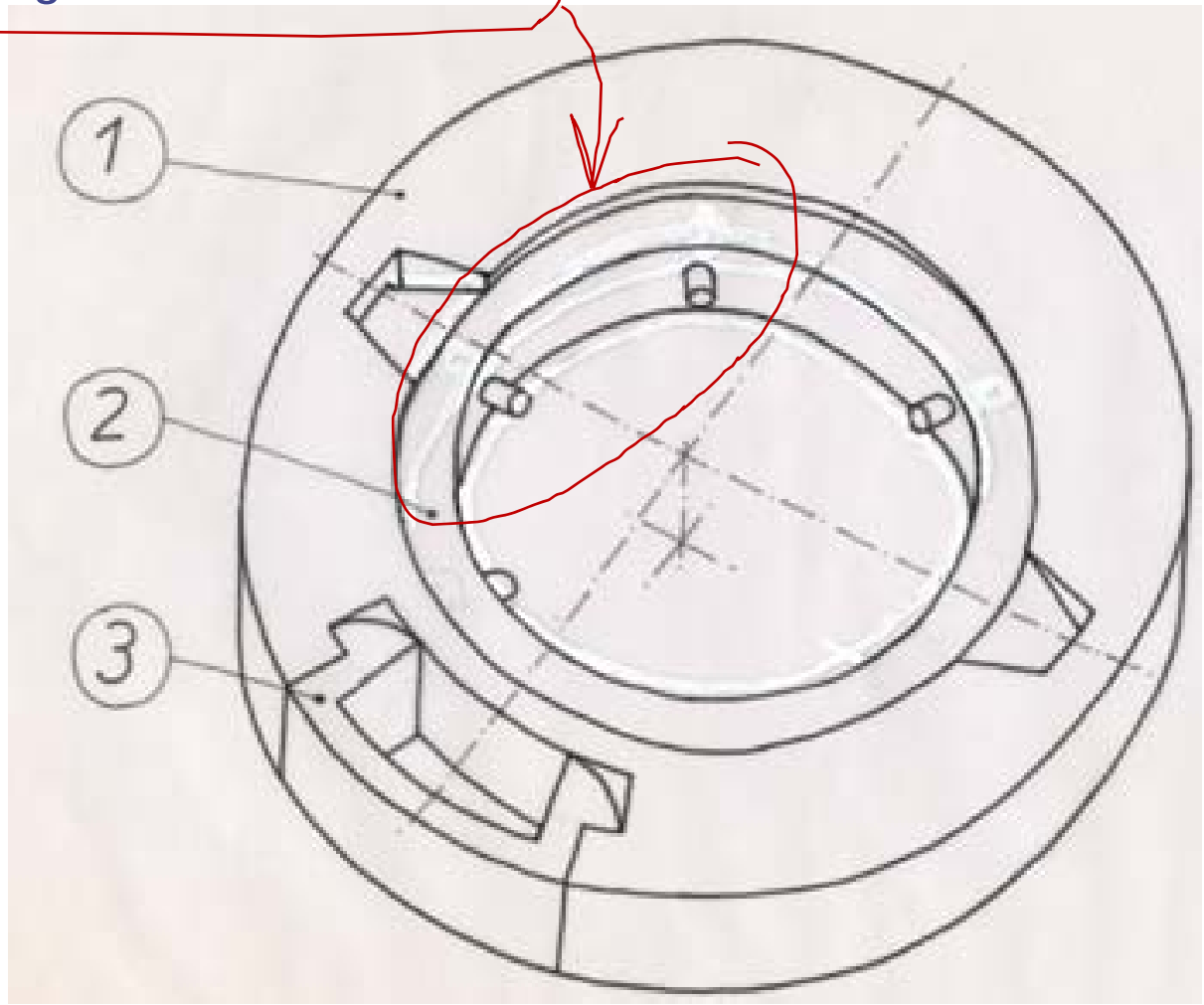


Also note the informal mixing of views:

- orthographic view
- detailed view
- pictorial view!

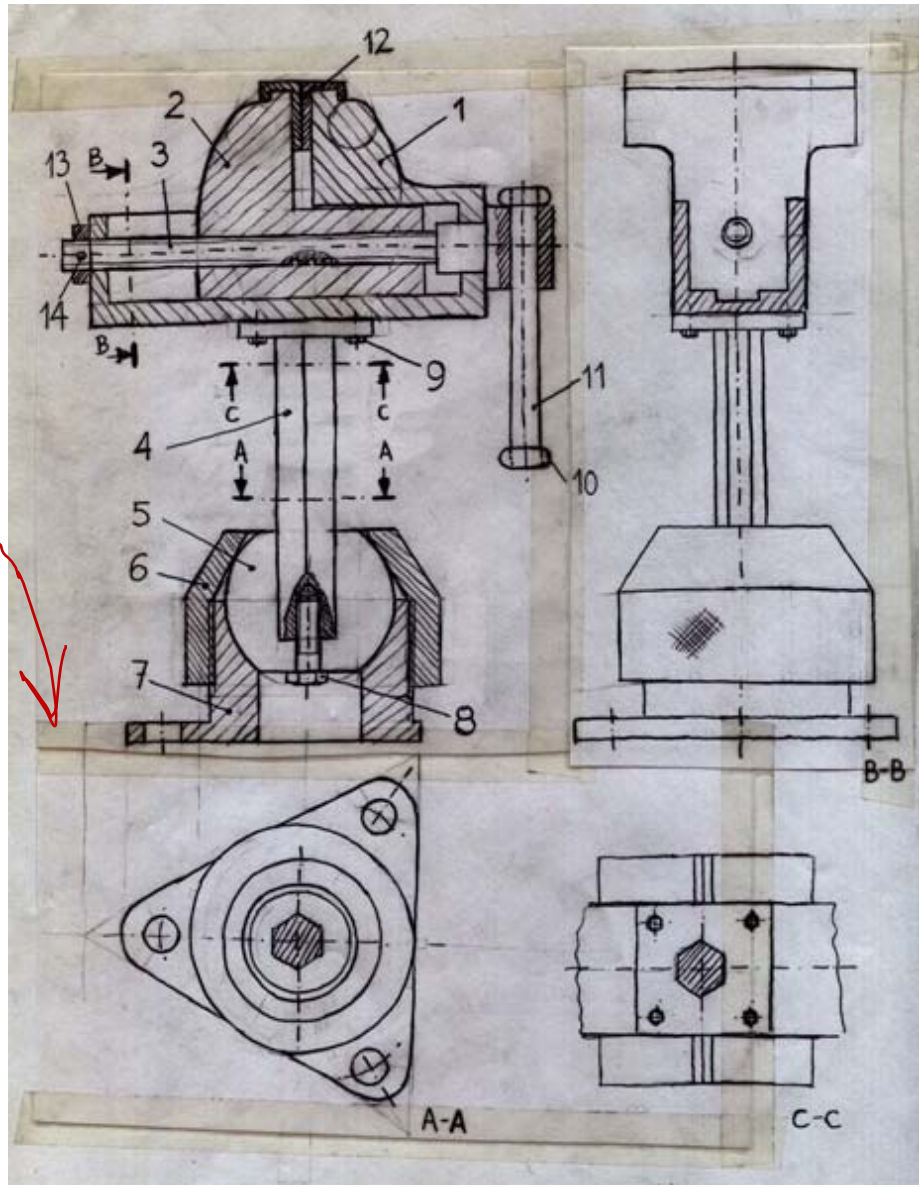
Virtual vs. actual paper and pencil

Erasing with correction fluid



Virtual vs. actual paper and pencil

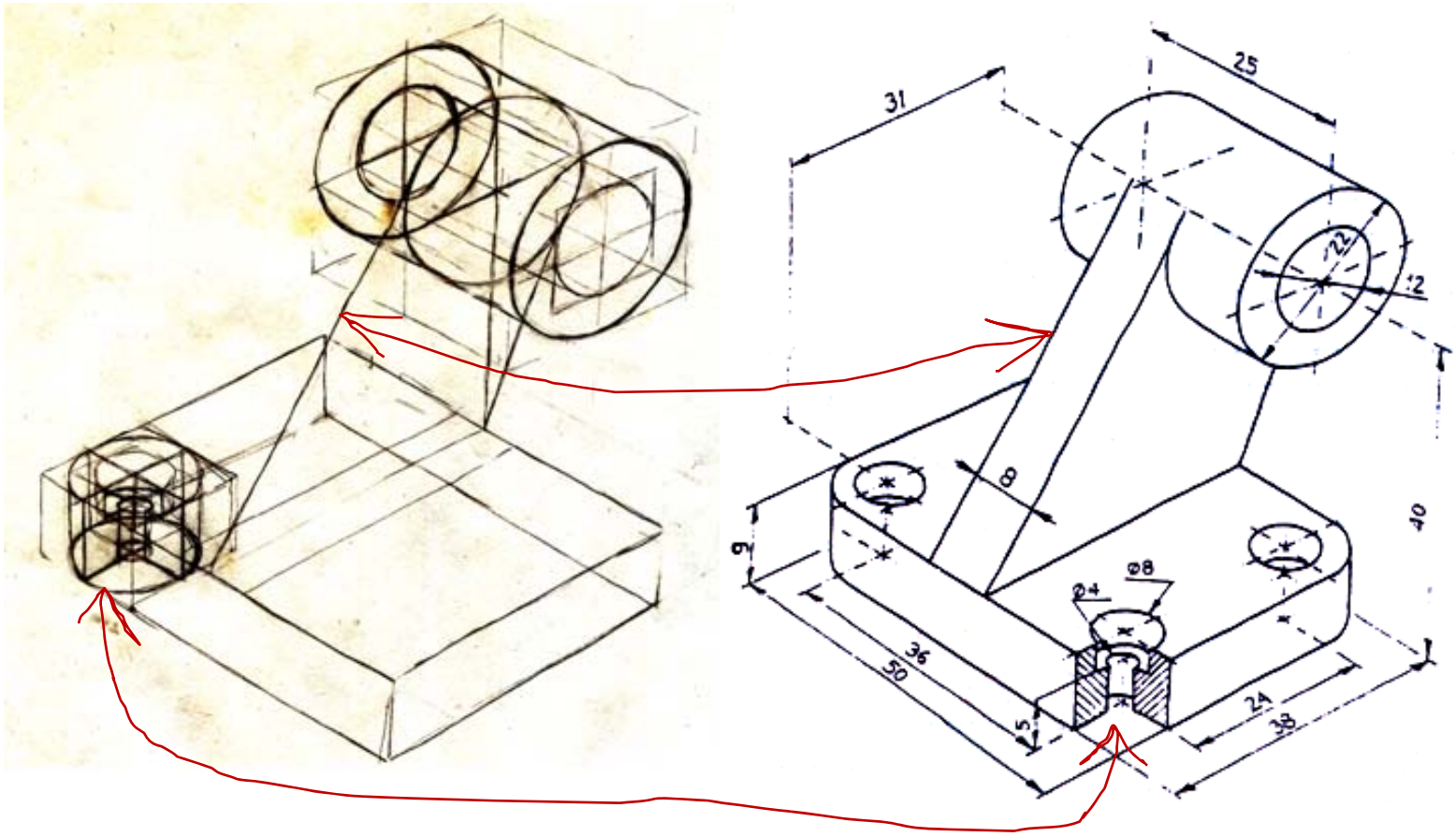
Hard cut and paste



Virtual vs. actual paper and pencil

Symmetry to complete the drawing

The copies were made though transparent paper superimposed and displaced over the original drawing



Virtual vs. actual paper and pencil

We conclude that:

- The set of **operational modes** is complex:
  - Inking
    - Scaffolding
    - Highlighting
    - Highlighting the rear side
    - Inking axes
    - Hatching
    - Overtracing
      - Autocorrection
      - Thinking
      - Decoration
    - Annotating
  - Editing
    - Erasing
    - Copy and paste
    - Copy with symmetry
- The **switching strategy** is non-intrusive !

Virtual vs. actual paper and pencil

We may reduce the gap between actual paper and pencil and SBM tools:

- ✓ Replicating as many modes as possible
- ✓ Adding extra features

Wrong paradigms guide human-to-computer interaction

command-driven and menu-driven

No interface paradigm which would suit experienced design engineers

Identifying such a paradigm is a difficult task in itself

Virtual vs. actual paper and pencil

We may reduce the gap between actual paper and pencil and SBM tools:

- ✓ Replicating as many modes as possible
- ✓ Adding extra features

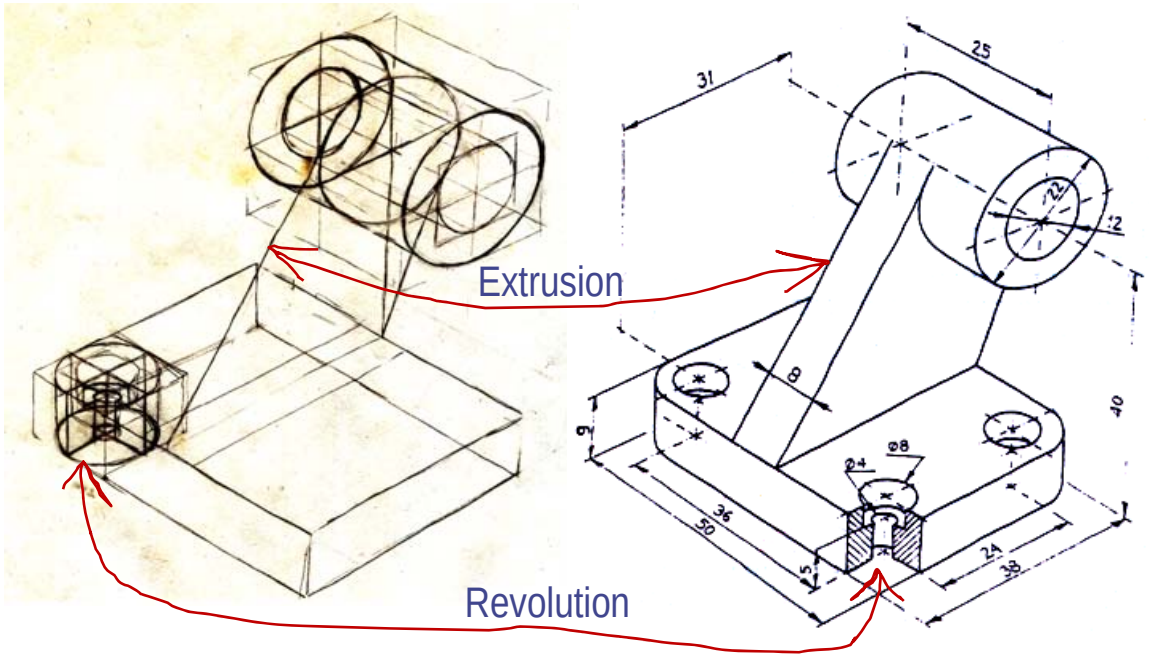
There is a broad agreement on some of the main advantages of computer-based systems:

- ✓ Work is easier to edit
- ✓ Work is easier to file
- ✓ Work is easier to interface to other applications

Virtual vs. actual paper and pencil

Besides, …

… adding some current CAD operations could also help in reducing editing tasks



Virtual vs. actual paper and pencil

Final goal:

SBM tools should be as easy to use as actual paper and pencil!

Sculptris by “Dr Petter” is a sucessful example
http://www.drpetter.se/project_sculpt.html

To this end:

1 Hardware advances are required

For example, tablets have been reported to be less comfortable to use than pencil and paper because of the small gap (both in time and distance) between the cursor and the pencil tip

2 Software improvements are also required

Use and maintenance of computers still requires technical knowledge which designers, quite rightly, do not see as part of their job

Virtual vs. actual paper and pencil

3

We still require a full taxonomy of operating modes

Including their
mutual relationships
and descriptions of cues
used to discriminate between them

The list we have described
is illustrative, but far from exhaustive

4

Two questions must be posed

- How many functions can be provided without buttons and menus?
- How many functions
  does a design engineer require?

If the second answer is larger than the first,
we need a new paradigm

Virtual vs. actual paper and pencil

1 Full modeless applications are not the final goal

If we wish to replicate
real paper-and-pencil scenarios
in virtual environments,
we must be aware that
actual paper-and-pencil scenarios
include a rich variety of different modes

2 Replicating paper-and-pencil scenarios in virtual environments is still unfeasible

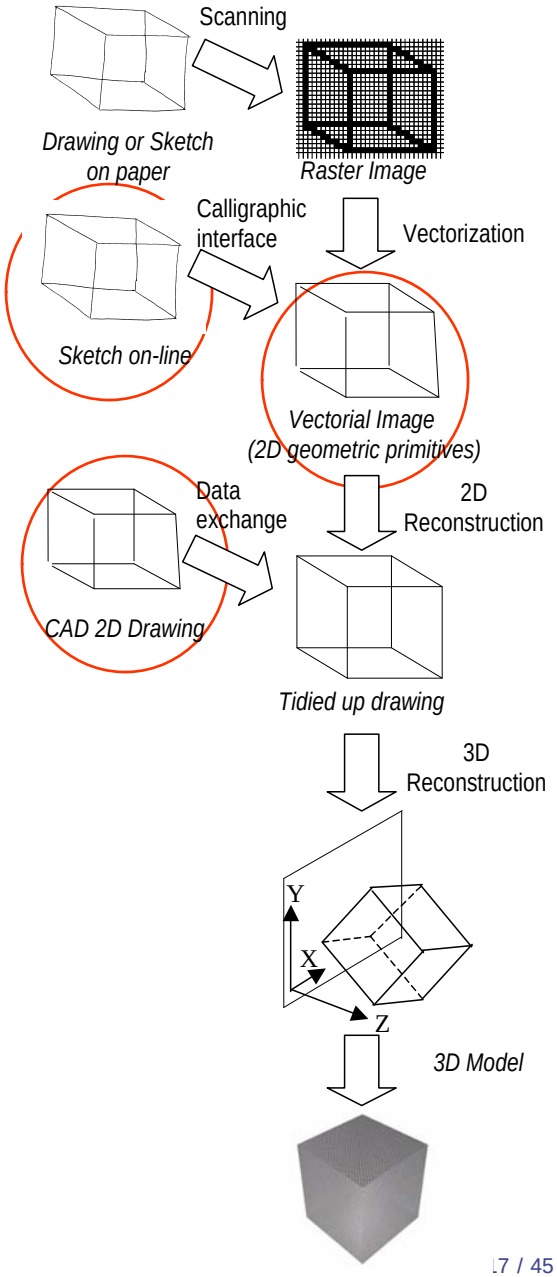
Although the goals of SBM
have supposedly already been accomplished, practical
implementations have unfriendly interfaces

Interpreting annotated sketches

Three types of input have been considered so far for SBM:

1. perfect line drawings
2. line drawings containing some "minor" mistakes
3. freehand sketches

But this simple classification may be refined!



Interpreting annotated sketches

We can consider three purposes for sketching:

- ✓ *thinking*
- ✓ *talking*
- ✓ *prescribing*

Combined with two levels of geometrical information:

- ✓ *Line drawings*
- ✓ *Sketches*

And two levels of non-geometrical information

- ✓ *With annotations*
- ✓ *Without annotations*

(More details in Annex 8)

Interpreting annotated sketches

We can consider three purposes for sketching:

- ✓ *thinking*
- ✓ *talking*
- ✓ *prescribing*

Combined with two levels of geometrical information:

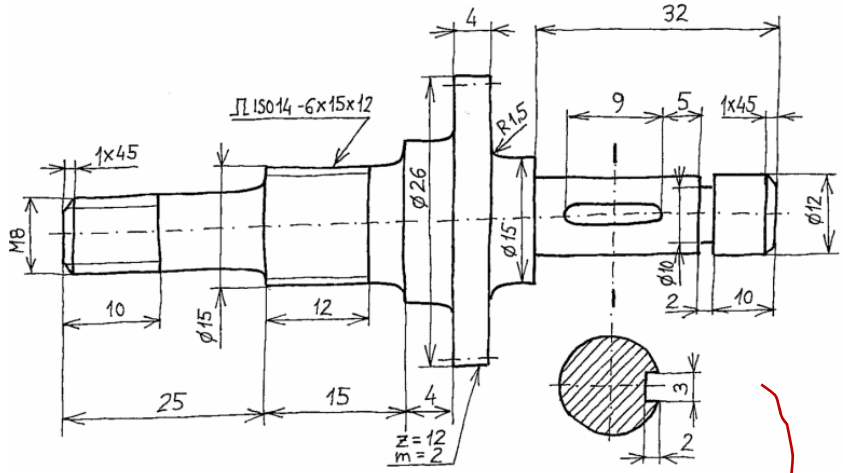
- ✓ *Line drawings*
- ✓ *Sketches*

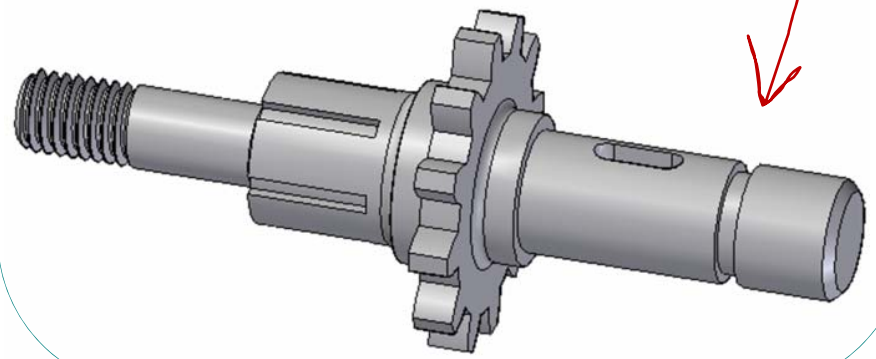
And two levels of non-geometrical information

- ✓ *With annotations*
- ✓ *Without annotations*

(More details in Annex 8)

An open problem derived from this classification is **interpreting annotated engineering sketches**

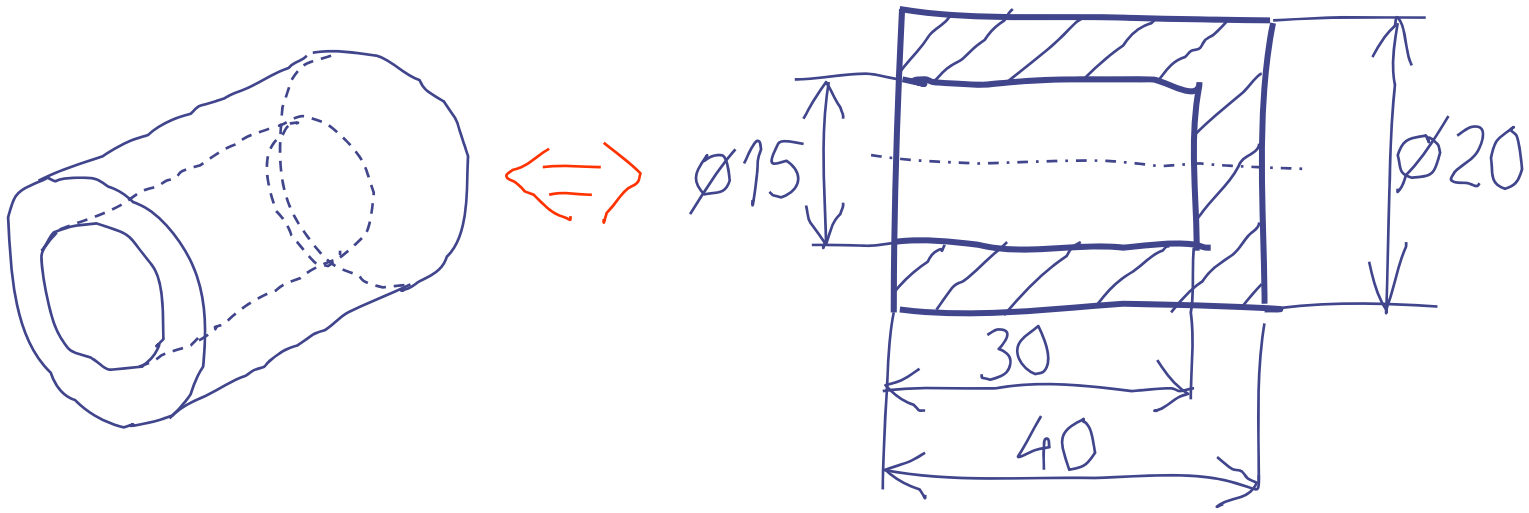




Interpreting annotated sketches

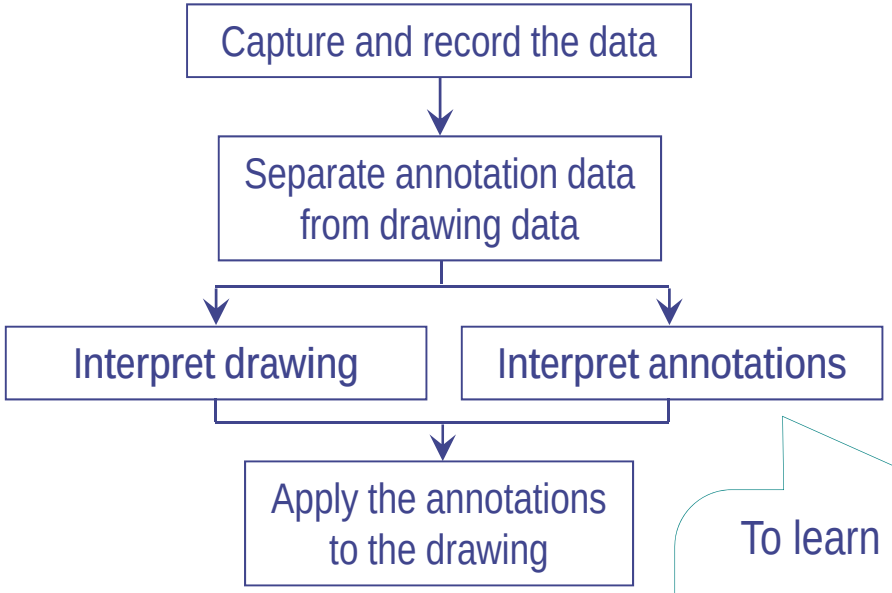
The generic term “annotations” includes:

- ✓ Many standardised conventions (e.g. dimensions)
- ✓ Cut views with hatchings
- ✓ A large etcetera of icons and symbols



Interpreting annotated sketches

The proposed approach for the open problem of **producing 3D models from annotated engineering drawings** is:

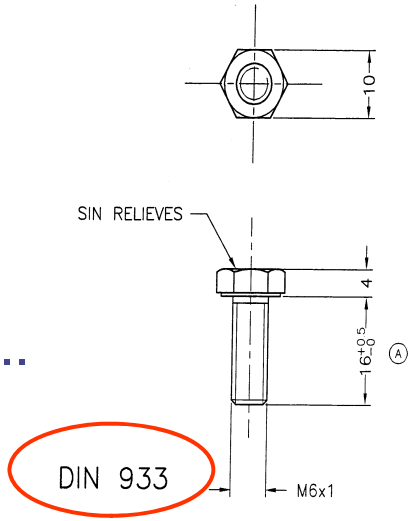


To learn more:

LaViola, J. (2007) **Advances in Mathematical Sketching: Moving Toward the Paradigm's Full Potential**, IEEE Computer Graphics and Applications, 27(1):38-48

Interpreting annotated sketches

The goal of interpreting engineering symbols is not trivial, since…



…behind apparently quite simple drawings...

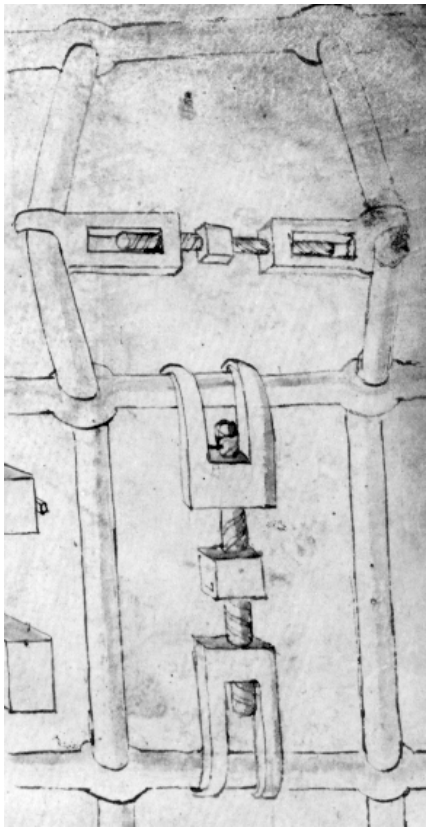
...there are hundreds of **standards…**

...defining the exact meaning of many symbols and conventions

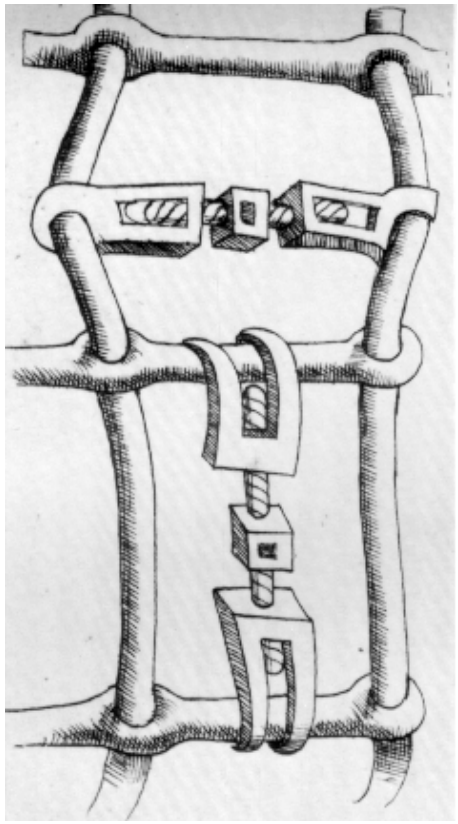
Interpreting annotated sketches

It is obvious that communication of relevant information depends on the **meaning** of **symbols**:

Original turnbuckle



Bad copy

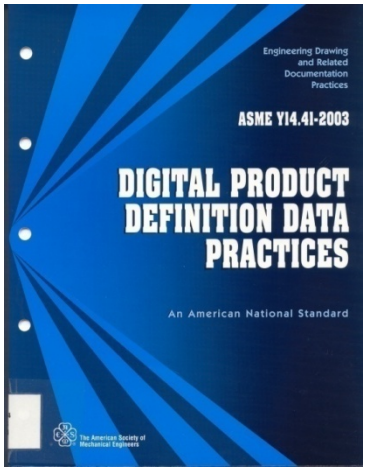


Ferguson E.S. ***Engineering and the Mind's Eye***, MIT Press (1992)

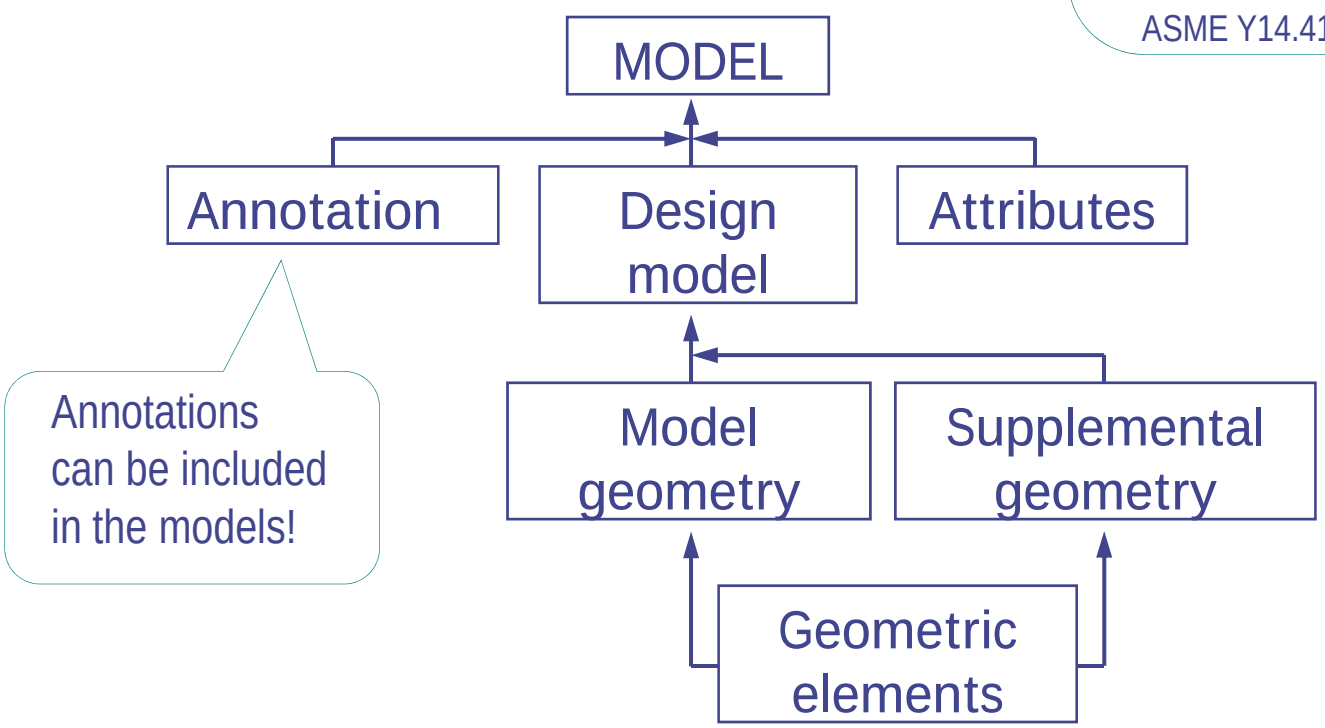
Misunderstanding of symbols causes technical information loss!

Assembling sketches

The problem becomes still more challenging if we realise that **new standards** already allow annotations in 3D models:

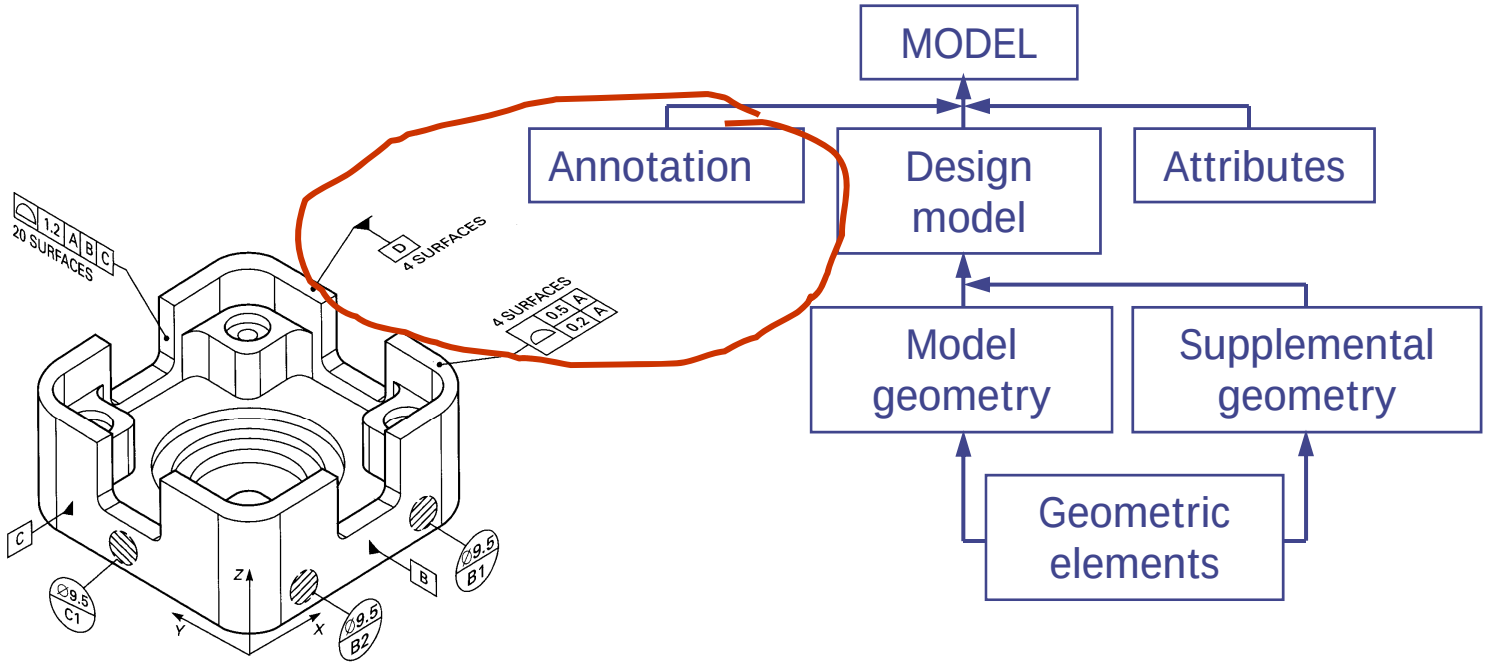


ASME Y14.41-2003



Assembling sketches

Today, computers are blind to these annotations !



The annotations are just “labels” added to the model

✓ Which the user can read and modify,

✗ but the geometrical engine does not use them, neither to construct, nor to edit or validate the model.

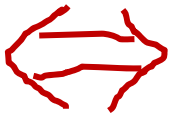
Interpreting annotated sketches

One interesting related open problem is
interpreting sketched data input
for Computer-Aided Engineering applications

It is an open problem
since data is input through
two alternative
WIMP user interfaces :

(window/icon/menu/pointing device)

Stand-alone
CAE pre-processors
which define both
geometric data and attributes

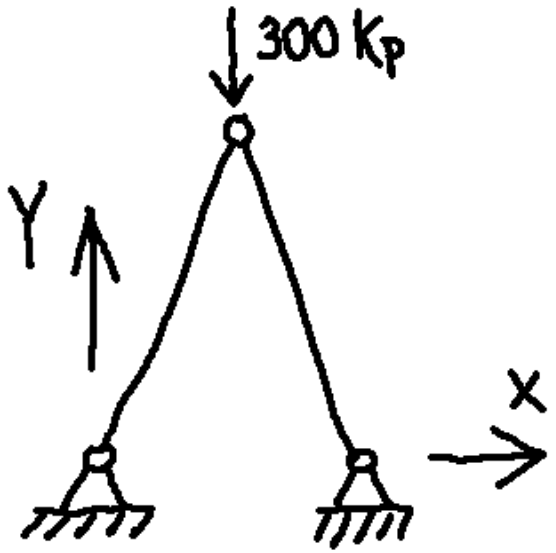


Combination of
CAD applications
plus downstream
CAE pre-processors

CAD exports the geometry
and CAE adds attributes

Interpreting annotated sketches

Input



Output

```
>TITLE
  "2 BARS STRUCTURE"
>COORDINATES
  1   0.0   0.0  0.0
  2   0.5   1.0  0.0
  3   1.0   0.0  0.0
>MATERIALS
  1  2.1E+6
>GEOMETRIC PROPERTIES
  1    1.50E-4
>ELEMENTS
  1    1 3   1   1  0  1
  2    2 3   1   1  0  1
>CONSTRAINTS
  ALL             DZ GX GY GZ
  1       DX DY
  3       DX DY
>LOADS
  STATE 1
  NODE LOADS
    2  0.0  -300.0  0.0
```

The input are those sketches which designers typically draw aside to fix their ideas before interacting with CAE pre-processors

The output is a file which meets the specifications of the desired analysis code

Interpreting annotated sketches

Two reasonable **assumptions** are:

1. The input sketches are drawn directly onto a computer screen acting as “virtual paper and pencil”

Not side-drawn
on an actual paper sheet!

2. The user is still in the process of conceptual design and is not yet ready to progress to a detailed design stage

Interpreting annotated sketches

Hence, the **goals** are:

1 supply the user with
a computer interface
similar to classical paper-and-pencil

2 minimise the amount of information
provided by the user …

… and give the user
more freedom
in inputting and editing it

Interpreting annotated sketches

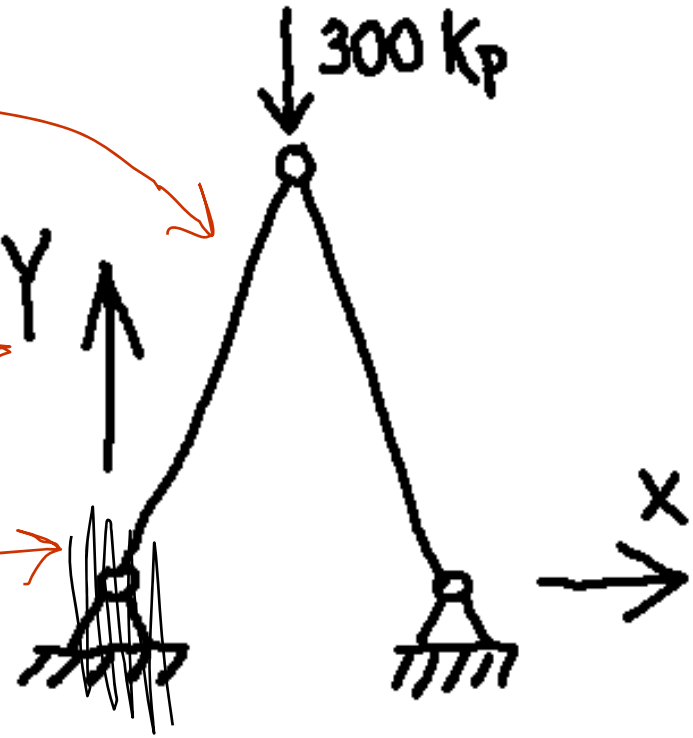
Our application, Pre/Adef, distinguishes:

Company P., Aleixos N., Naya F., Varley P.A.C., Contero M. and Fernandez-Pacheco D.G. (2008) A New Sketch-Based Computer Aided Engineering Pre-Processor. Proc. Sixth Int. Conf. on Engineering Computational Technology. Civil comp Ltd. Paper-149.

1 Geometric entities

2 Symbols associated with annotations

3 Gestures associated with editing tasks (i.e. “sketched commands”)



Interpreting annotated sketches

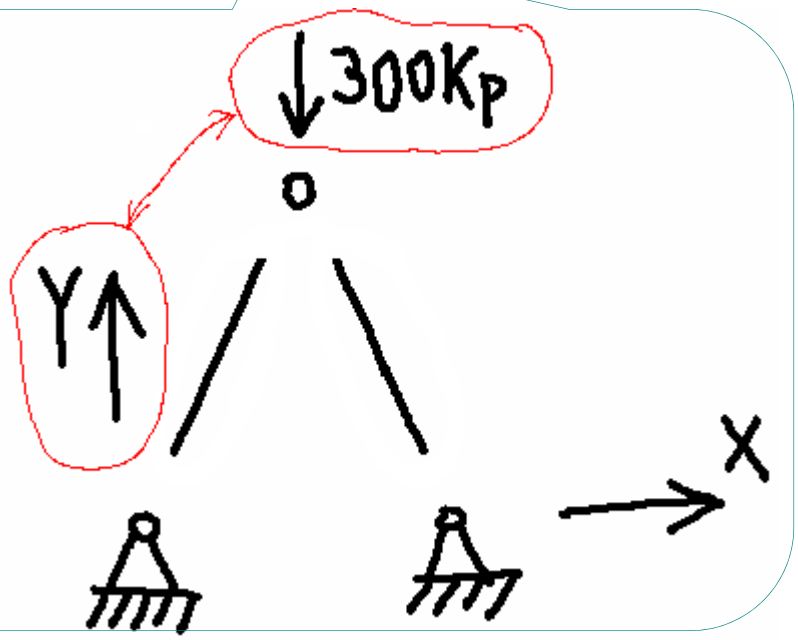
Finally, having interpreted each group in isolation, we must **combine** them into a whole

- ✓ Connect the bar elements to the appropriate nodes
- ✓ Apply the loads to the right nodes or elements

The arrow of the 300 kp force is parallel (more or less) to the arrow of the “Y” axis

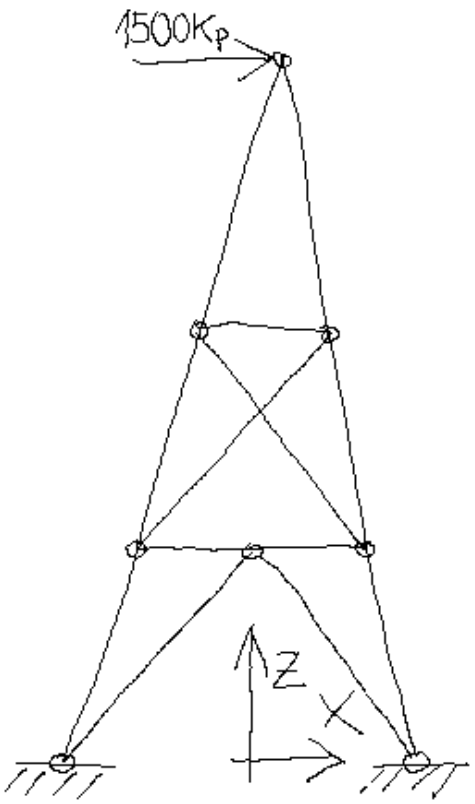
They have opposite senses

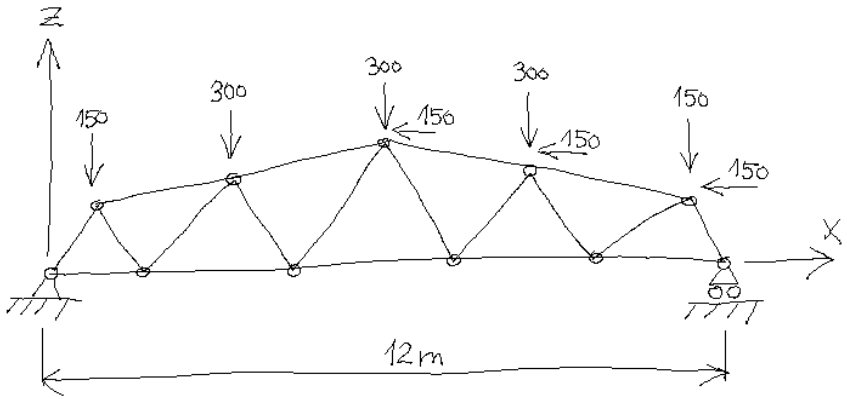
The vector force should thus be interpreted as (0, -300, 0)

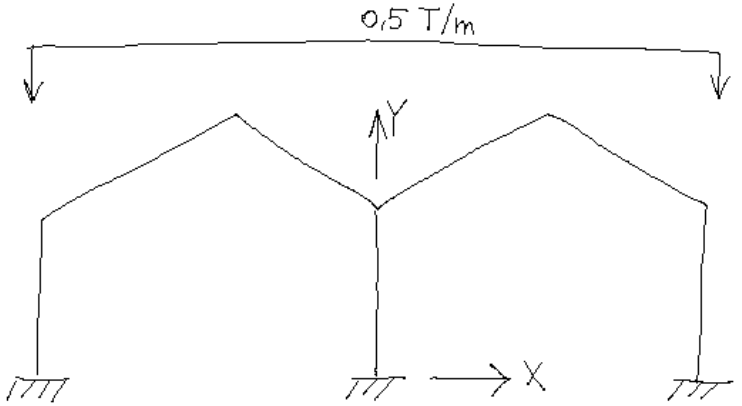


Interpreting annotated sketches

We tested our approach by sketching a set of examples







Interpreting annotated sketches

ADVANTAGES

Valid output files are obtained ✓

Training not required ✓

The user is not urged ✓
by the system
to define exact dimensions

UNSOLVED PROBLEMS

✗ Modes required

(but only to change the mode
when moving to a very different task)

✗ Users do not always
feel comfortable
with an on-line parser!

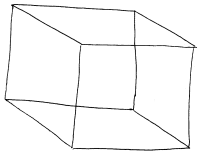
Our interface is similar to but not yet as good-for-thinking as actual paper-and-pencil

But produces output files !

(More details in Annex 9)

Assembling sketches

Currently, we are limited to reconstruct **isolated parts**



But we want to be able to
create assemblies from sketches !

Our vision is
to define and implement a set of symbols
that can help a SBM system
to assemble 3D models obtained from 2D sketches

Assembling sketches

The basic guidelines of our approach should be:

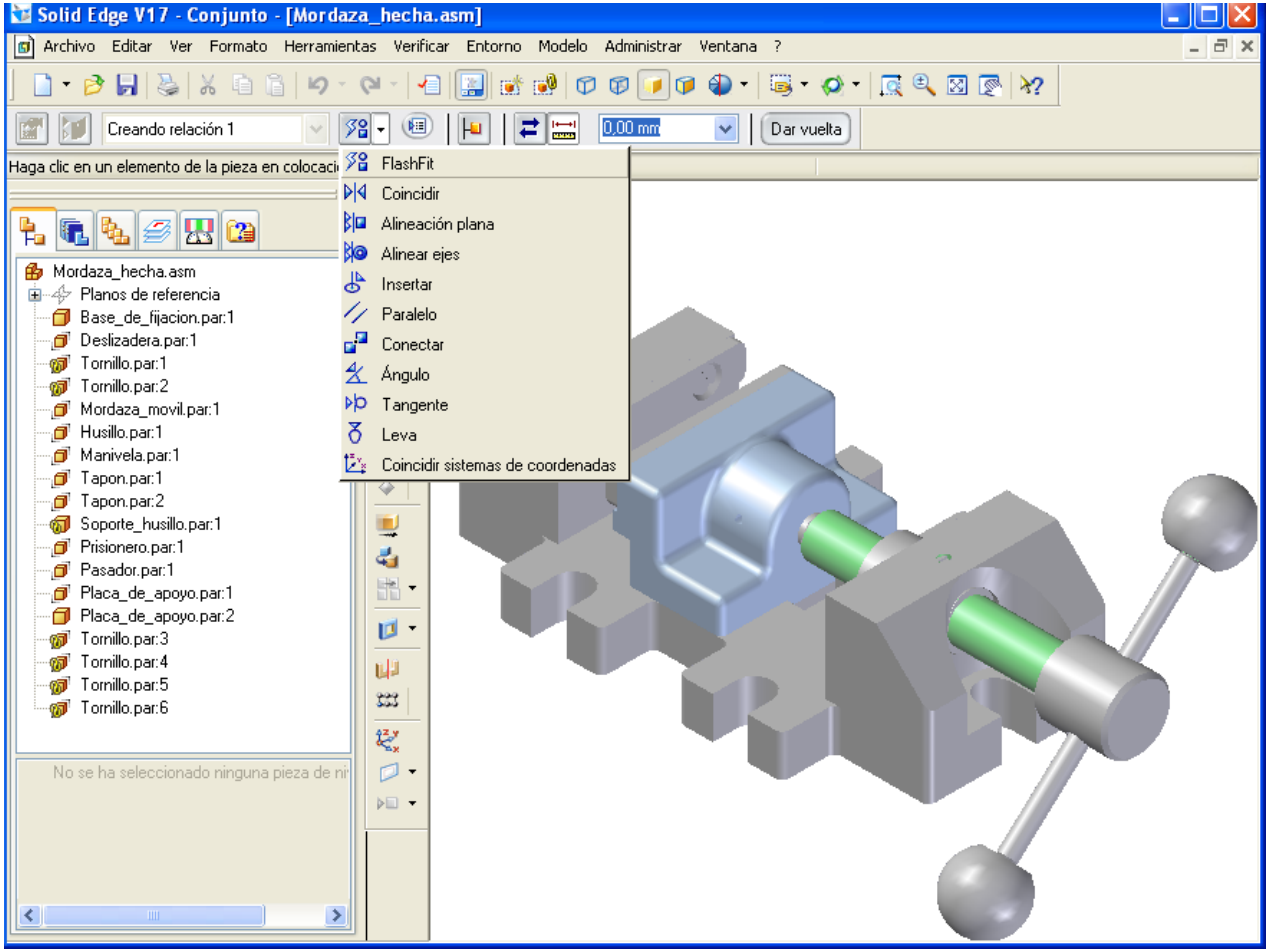
- √ The symbols must be sketched themselves, as part of a “natural” design process
- √ The meaning of the symbols must be “robust”

  In the sense of being understood without mistakes by the geometrical engine in charge of assembling the parts

- √ The symbols should overtake the faults of current sets of CAD operations

Assembling sketches

What is wrong with current CAD applications?



SolidEdge: originally developed and release by Intergraph in 1996
using the ACIS geometric modeling kernel
it later changed to using the Parasolid kernel

Assembling sketches

Components can be positioned within the product assembly using:

- ✓ absolute coordinate placement methods
- ✓ **mating conditions**

Mating conditions are definitions of the relative position of components with respect to one another

For example alignment of axis of two holes or distance of two faces from one another

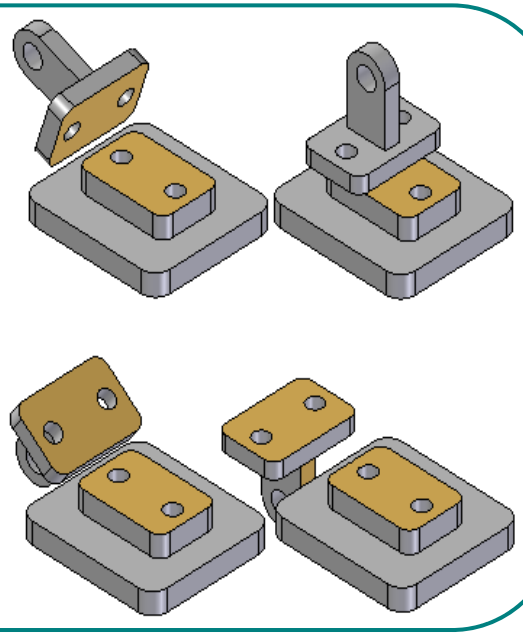
The final positions of all components based on these relationships is calculated using a geometry constraint engine built into the CAD or visualisation package

Assembling sketches

Some **tools for mating conditions** assist the user to get an intuitive and friendly set of constraints:

1. As users place parts in an assembly, assembly relationships position new parts relative to parts already in the assembly.
2. There are several relationship types for positioning parts relative to each other.

Starting with v8 (2000), Solid Edge also has a FlashFit option which can reduce steps required to position parts.

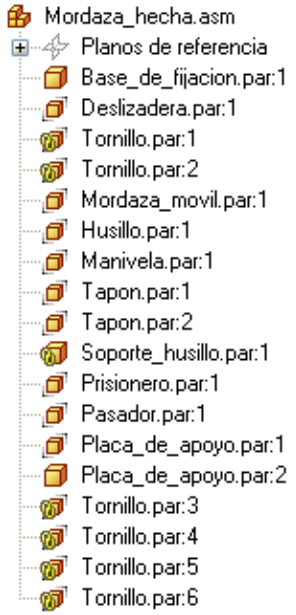


Assembling sketches

However, we find one main drawback:

Only **complete** and **consistent** parts can be assembled

CAD assembly sub-systems require standard CAD parts input



Detailed design of parts is an assembly **pre-requisite**!

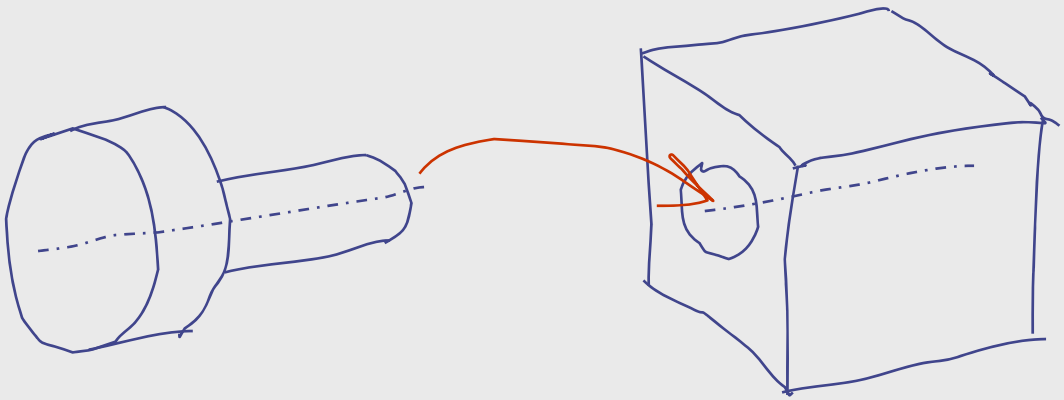
Assembling sketches

Our vision is creating a sketch-based environment …

… able to assemble different **parts**…

… that are **not yet fully defined**

Assemble conceptual designs of parts!



Summary

We have introduced three long term open problems in the sector of SBM tools:

1. Making virtual paper and pencil more usable than actual paper and pencil
2. Interpreting annotated engineering sketches
3. Creating assemblies from sketches

Summary

1 Making virtual paper and pencil more usable than actual paper and pencil

In the hands of a skilled user,
a pencil is a complex tool in itself,
capable of several different modes of operation

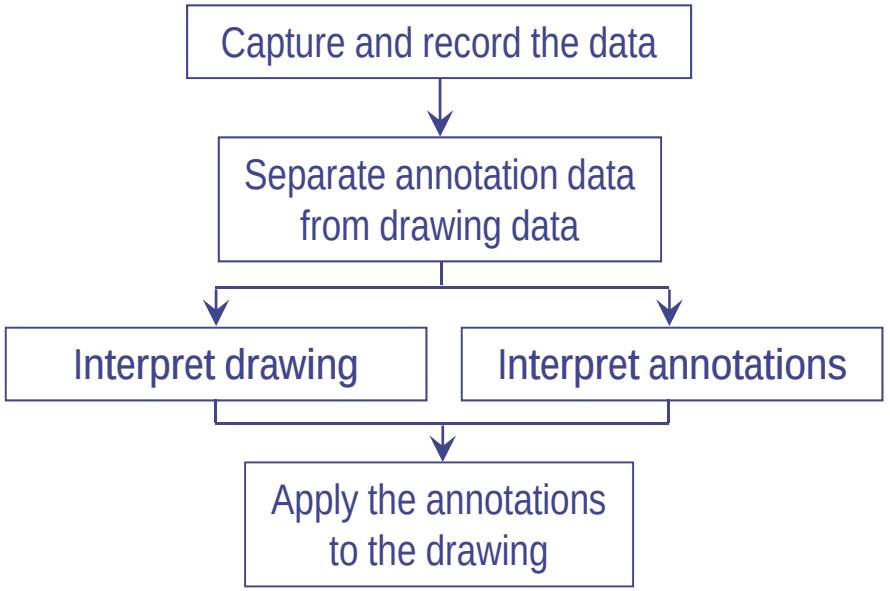
To be competitive,
computer-aided sketching tools should support
the various operating modes
which pencil and paper allows
and skilled design engineers
habitually use

Interpreting annotated sketches

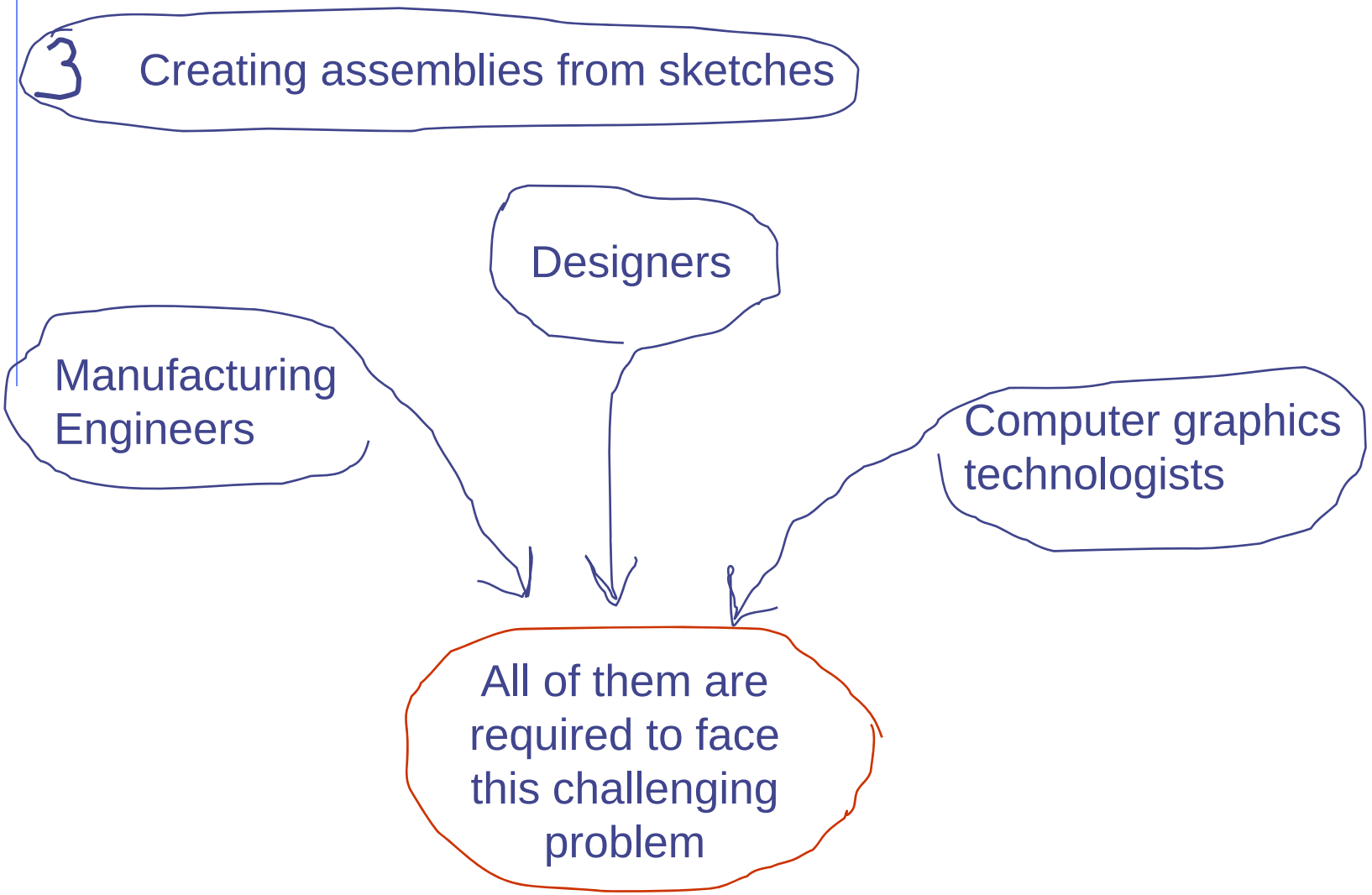
2 Interpreting annotated engineering sketches

It is an open problem since
engineering designers
annotate their designs with symbols

The proposed, although difficult,
approach is:



Summary



Sketch Input of Engineering Solid Models

Thanks for your attention !

Pedro Company

Peter Varley