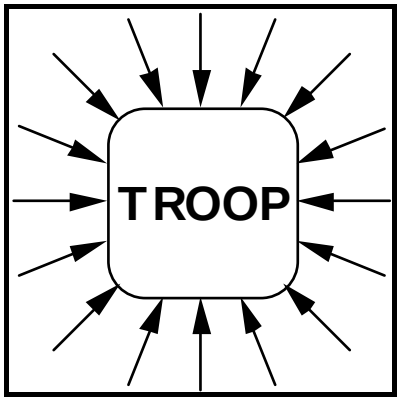
p **Object-Oriented Re-Engineering**:

target: *the system itself, but implemented according to the object oriented methodology*

benefits: *O-O design style fulfils many desirable code requirements:*
(Modularity, Extendibility, Integrability, Robustness, Reusability)

Tool for
Re-engineering towards
Object
Oriented
Paradigm



p **emphasis on data**

- data reverse engineering is easier
- existing objects must correspond to some data structures

p **a fundamental aspect: capture the semantics**

- too difficult in a purely automatic way
- access to informal documentation
- "tricky code" obscures design issues
- human intervention is required

The Re-Engineering process

p *Identification of objects and fields*

1. **Identification of the data structures used by different modules of the existing programs**
   (global variables, record description structures, data structures most used as actual parameters).

   May database structures correspond to objects?

p *Identification of methods*

1. **Arranging the modules by taking into account:**
   - *their size, expressed as Lines Of Code (LOC);*
   - *their depth in the Structure Chart;*
   - *their reuse frequency*

2. **Slicing the modules, starting from the modules derived from the previous step and on the basis of the variables identified in the object identification phase.**

3. **Identifying the code chunks as potential methods and assigning to them a name, a set of keywords, the name of the potential "objects" they are operating on**
   (keywords are extracted from a faceted classification a browser can graphically display and navigate through)

p ***Definition of classes and identification of inheritance hierarchies***

1.a Considering the similarities among the potential objects on the basis of a type classification based on characteristics like access, scanning and storage methods ([Meyer]).

1.b Considering the similarities among the potential methods by making use of:

- *types and data structures*
- *PDG slices*
- *regular expressions at different abstraction levels (particular substrings are identified by a single label that gives information about its functionalities)*
- **"formal specifications"**

Techniques developed in the context of the Information Retrieval area may help in this process

2. Paying attention to the identification of possible cases of generalisation when considering modules at higher levels in the Structure Chart.

3. Rebuilding of the program, eventually restructuring it, expressing it by means of the identified components.