A Tutorial Example of Meaningful Programming Design: Fire Truck[i]

Step 1: Problem Statement (1991 ACM Programming Contest)

The Center City fire department collaborates with the transportation department to maintain maps of the city that reflect the current status of the city streets. On any given day, several streets are closed for repairs or construction. Firefighters need to be able to select routes from the fire stations to fires that do not use closed streets.

Central City is divided into non-overlapping fire districts, each containing a single fire station. When a fire is reported, a central dispatcher alerts the fire station of the district where the fire is located and gives a list of possible routes from the fire station to the fire. You must write a program that the central dispatcher can use to generate routes from the district fire stations to the fires.

Step 2: Linguistic Evaluation

1. The “Center City Fire Department” [a proper noun] maintains [verb] maps [noun] that {reflect [verb] status [noun] {of streets [modifies status]} [subordinate clause modifying maps]}.
2. The status [noun] can be *open (unclosed)* [adjective] or *closed* [adjectives]. *unclosed = not closed = open.* {semantics}
3. Firefighters [noun] must select [verb] routes [noun] {on *open* [adjective] streets [noun] [prepositional phrase modifying routes]}.
4. Program [noun] must generate [verb] routes [noun].
5. Maximum [adjective] number [noun] {of corners [modifies number]} = [verb] 21 [noun]. There’s a herring here… start at 0 or 1 (C or Fortran)?
6. Always start at [street corner]1.
7. There are a number (unknown) of cases [in input].
8. First line of a case = place to get to (always starting at 1).
9. Next line(s) (including none) are pairs of street corners that are *open.*
10. Terminating condition is when pair is 0 0.
11. Output case number, then a number of routes, then the total number of paths found.
12. Correctness of route: no loops.

Step 3: Vocabulary Table

Lexicon	Context	Vocabulary Lexicon
Center City	Name of place	City

Center City Fire Department	Call it a group	Group
Maintains	English meaning	Maintains
maps	Infer “city maps” from context.	Maps
Reflect	Indicates	Reflect
Status	Condition, usability (context)	Status
Streets	City map has streets of City	Streets
Open	Status: means passable; not open means closed.	Status(open)
Closed	Status: not passable	Status(closed)
Firefighters	In Center City	Group
Must select	“Routes” give context	Choose
Routes	Paths on map, all streets marked open Context: no loops in routes.	
Program	A computer program	Program
Must generate	Must produce	Produces

Step 4: Rewrite with Reduced Vocabulary.

1. {The Group maintains maps that reflect status of streets. This information only introduces the map issue naturally.} Street maps of the City exist.
2. The status of a street can be Status(*open) or Status(closed).*
3. Group must choose routes on Status(*open)* streets from firehouse to any point in city (Last clause from context).
4. Program produces routes.

Schemata Leap

Include the Cmaps.

A central tenet of meaningful programming is that problem solving and program design use a continuum of procedural and propositional knowledge.

The mantra of meaningful design is to consider the psychological bases of problem solving before addressing any programming issues. The issues are

1. Vocabulary is the first main issue. Vocabulary consists primarily of **concepts.** Concepts are of two general types: **objects** and **relations.**
2. Manipulation of concepts is done when the form is recognized and a **schema** exists that provides transformation information.
3. Problem solving in general proceeds by recognizing and transforming the problem, often through **decomposition.**

Move to Categories

The clear problem is that the connection must be made between one of the words on the left and the undirected weighted graph on the right. This is a schema connction. The schema here determines all the right-hand requirements.

Problem Language	Category
Map	Undirected Weighted Graph
Streets	Edges in above
Status	Weights in above
Point	Street intersection
Routes	Paths in above

Move to Types

The types for the undirected weighted graph could be immediately given, or the category of *undirected weighted graphs* could be explored since we could use tables or node-arc representation. Each representation (a category) has various encoding issues (how to name a street intersection). A simple approach would be to have a 2-dimensional table with the columns naming corners that are connected by open streets.

Category	Type
Undirected Weighted Graph	2-dimensional table
Edges in above	Each row means open street between
Weights in above	Not needed
Street intersection	An entry in either column
Paths in above	Connection of corners in the table

[i] AT&T Computer Systems. 1991 ACM Scholastic Programming Contest Finals. Problem A.