A procedure for writing programs

Tom Rochette <tom.rochette@coreteks.org>

December 21, 2025 — 77e1b28a

0.1 Context

At one point in time, we expect AGI to be able to write programs. How we will get there, we do not know yet. However, since we are already able to write programs for ourselves and others, we might ask ourselves what are the steps we go through when we write a program. If the steps identified are easily automated, then it might provide us with a foundation for programs which can write programs.

0.2 Learned in this study

0.3 Things to explore

- What are the steps one goes through when writing a program/algorithm to solve a specific problem?
- How are important entities identified?

1 Overview

When I want to write a program, may it be small or big, I generally approach it in a top-down fashion. This means I first go over the high level concepts that I will need to make my program work. Then I drill down into each concept and break it down until I get to the implementation details.

Here's a list of the many things that are considered while writing a program:

- Pre-requirements (what is needed in order to develop it)
- Relationships (what will depend on it and what it will depend on)
  - Dependencies (what it needs to know about when running)
  - Dependents (what will depend on it)
- Requirements (what does it need to do)
- Performance (what are its constraints)
  - Time (how fast should it return a result)
  - Space (how much space should it use to do its task)

Here's a few questions I will ask myself before writing any code:

- Is this functionality already available in the existing codebase?
- Is there a library that already does this (either in the project or available online)? Does it cover the required functionality partially or completely?
- Is there an implementation already available online? Can I produce a better implementation?
- Do I know how to implement this?
- What can I safely extrapolate this new feature will need to be able to do in the future? Can I account for any of it in my current design?

```
graph TD;
    0[Is this functionality already available in the existing codebase?]
    0 --Yes--> 100[Use it]
    0 --No--> 1["Is there a library that already does this (either in the project or online)? partially
```

```
1 --Yes--> 101[Use it]
1 --No--> 2[Is there an implementation available online?]
2 --No--> 3[Do I know how to implement this?]
3 --No--> 4[Search how to implement it]
3 --Yes--> 5[Implement it]
2 --Yes--> 102[Can I produce a better implementation?]
102 --No--> 104[Use existing implementation]
102 --Yes--> 103[Implement it]
```

Most of the time a single straightforward solution will fit the bill for all the given requirements and implementation will be the next step. However, it happens from time to time that multiple solutions are valid and you will have to determine the pros/cons of each solution in order to make the best decision for the given situation. In any case, one will want to:

- Reduce risk
- Maximize flexibility
- Minimize coupling
- Plan for the solution to be potentially replaced by another one

Once a solution has been picked out, it's time to establish the construction order. Here, there are two approaches: top-down or bottom-up.

Top-down consists in preparing the high-level structures and slowly drilling down into the implementations details (similar to how I approached the program analysis). Generally, I'll outline the various things that will be done in pseudo-code or comments within the functions, then create the function call and the function definition. Repeat this until you reach the low-level implementation. At this point, I have not defined the parameters of any of the functions, simply the function call flow. Then it is time to think about what will go from one function call to another. Finally, when all of this is done, it is time to go into implementing the "empty" functions with actual low-level logic as all the high-level logic/communication has been established previously.

Bottom-up consists in writing down the low-level features that you will need. This can generally done if you already have a good idea of the different components you'll need to construct your overall program. I think of this approach as one you'd generally use if you know how to do certain things already but no library is available on hand.

2 See also

- Automated programming

3 References