Real Swiss don't need SRP, do they? by Gabriel Schenker

Introduction

*You may ask yourself why I publish another article about the **single responsibility principle (SRP)**. We already had some very good post about this principle at Los Techies, e.g. here, here and here to mention just a few. Well, the reason is that I consider this principle **one of the most helpful ones to achieve higher quality of code and better design** when applied consistently. And I want to approach this topic from a different standpoint than is usually done by other authors...*

What does **SRP** mean? The theoretical explanation (you might have read or heard many times already) is

"There is one and only one reason to change a class."

What does this mean? This means that we should start to think small. Each complex problem cannot easily be solved as a whole. It is much easier to first divide the problem in smaller sub-problems. Each sub-problem has reduced complexity compared to the overall problem and can then be tackled separately. There is a proverb whose origin is from the Romans which says: "Divide et imperat", translated to English this means: "Divide and reign". It was not possible for the Roman emperor to reign the whole empire alone. No, he divided the empire into **independent** regions and appointed a king or sovereign to each region. At the end, the Roman emperor had not much more to do than orchestrate those kings or sovereigns that reported to him.

Now what does this mean for me as a developer? Each developer has to solve problems. Very often these problems are rather complex. Often the boundary conditions defining the problem even change. Now we should start thinking in terms of *"divide et imperat".* Let's find the sub-problems in the domain we are working in. Keep on dividing each sub-problem into sub-sub-problems until you reach the point where such a "mini-problem" has just one single task left. Let's then solve each of those "mini-problem" in its own class. Since each class only has one single task to fulfill, there is (as a consequence) only one reason left to change this class. We only have to change this class if the corresponding task changes.

Instead of tasks, one often talks of **responsibility**. *Each class should have a single responsibility.* The responsibility is to just accomplish the assigned task.

I want to say it again: Applying the single responsibility principle leads to higher quality of code and to better design! Why? Because the code is

- more readable, that is easier to understand
- less error prone
- more robust
- better testable
- better maintainable and extendable

Swiss people think differently...

But wait a moment. Swiss people are not descendants of the Romans...

A real Swiss does not need to follow the **single responsibility principle**. That's something for others but not for us. As a representative sample I want to present you one of our most successful products: the **Swiss army knife**.

One can clearly see that this product has not just one single responsibility. There are several responsibilities assembled in a single unit. We have 2 knifes, one can opener, one bottle opener, an awl and a corkscrew. This unit is very handy and fits well into the pocket of every **real** Swiss man.

Some people prefer to even pack more functionality into this unit as you can see in this second picture at right.

Well, this one is even better! But now I have to admit, that not every pocket is big enough for **this** tool. Thus only the **strongest** Swiss men get one.

Another tool comes to my mind when remembering the time I passed in the Swiss army. Our helmet is also considered to be a multipurpose tool. We primarily use it to protect our heads from injuries but it has as well served me many times as a pillow. It is even considered as an anti-hand-grenade tool. We were told that if a hand grenade is thrown at us and we have no time or possibility to throw it away then we should just put our helmet over it and burden it with our body. To be honest, I've never tried it...

Hey, wait a moment. I can give you another sample where we clearly show to the rest of the world that the SRP is not for us. It's our famous Swiss cows. They are not only good for providing us milk and eventually meet; no, they are also very good soccer players! Currently we have 3 of them in our national soccer team.

Not to forget the famous Milka cow! Here a cow is used as an advertising medium. This is a very important responsibility by its own.

Well, I could possibly continue to give you good samples of Swiss products that are really successful without respecting the SRP.

Why does the rest of the world consider SRP to be important?

Imagine one of the items of a Swiss army knife gets bent. It would possibly render the whole item useless or at least harm its functionality. I will have to throw away the whole knife. What a waste!

Or imagine that I am really happy with my Swiss army knife but just one element does not totally fit my needs. I would like to replace just this element with another one, which is better suited to my needs. I cannot do it! It's just not possible without (negatively) affecting all the other elements of the knife.

Imagine having a nice dinner with your wife or husband in a first class restaurant. You certainly have had lots of knives, forks and spoons, each element serving for a single purpose. There are knives to cut steaks or pizzas or knives to eat fish and so on. Each item is optimized for its specific task. If one of these items gets broken or if it doesn't fulfill its duty any more then it can be replaced without affecting the other items.