Public Domain Programming Interface - pdpi

Jakov Perica
kolewtt@proton.me

September 27, 2024

Abstract

This paper is an informal introduction to public domain programming interfaces, an interpretation in which all problems are divided into the dichotomy: public and not public. The public part is often easier to solve. Many problems which are complex in not public domain have their natural interpretation in the public domain.

"The introduction of the digit 0 or the group concept was general nonsense too, and mathematics was more or less stagnating for thousands of years because nobody was around to take such childish steps..." - *Alexandre Grothendieck*

A partition is an expression of a set as two disjoint subsets. Partitions are simple conceptualy and they arise in many places. For example, in probability theory, when one conditions on an event A, the event A itself induces a natural partition of $A \cup A^c = S$, where S is the sample space and we have

$$P(A) + P(A^c) = 1. \tag{1}$$

Using normal set operations one may express the same event in terms of different events. Sometimes it is easier to work with the complement or to condition on some other event. This is just a different interpretation of a fundamental problem-solving strategy of dividing the problem into easier subproblems recursively. It is also easy to divide a problem into much larger problems. Thus it can be usefull to sometimes check if the partition is well choosen.

Definition 1. (Java) A bit, byte or any number of bits is called java or javas if every bit is public. A javnik is a set of such bits.

Axiom 1. If j is an element of a javnik J, then j is java.

The word public is a primitive notion in this theory, just like the increment operator is not defined in Peano arithmetic. In other words, it does not need a definition and we may not care what constitutes public information. Nevertheless, we are free to take an axiomatic approach and assume anything we want.

Definition 2. (The Public Domain Internet) The public domain internet is the set of all hosts which preserve the root structure. We call such hosts public domain internet hosts or just javniks for short.

Specification and implementation 1. If X satisfies ssh root@ip and the root password is root, then X is a valid implementation of a public domain internet host. If there exists at least one public domain internet host, then the public domain internet exists.

This is just an idea like why not add the number 0. In order to pursue that idea, further research is needed. The Linux kernel already provides everything needed for the implementation, but many logical bugs may be discovered when looking at the world through this particular dichotomy. For example, one may argue that the generic user (the most common case) needs one and only one thing from the kernel: the machine. Suppose that every user uses a computer to do X. Then one may argue that the most common case is X. Although empirically true, it can make sense to interpret the most common case as the case with least assumptions or least structure. In the latter interpretation X is not the most common case, although every user uses it. This may be a play of words, but it certainly can in a natural and fundamental way influence how one thinks about a particular problem.

The theory so far informs us to assume that in the public domain internet nothing related with file permissions or address space randomisation or encryption is needed, and would qualify as not efficient and redundant. In a good design it would be obvious what to remove and removing it would be cheap.