How programming types were invented

Raul Rojas*

We are at the office of Professor Alonzo Church in Princeton. It is a sunny day in 1936.

Church: Come in!

Lawyer: Good morning professor Church! I have all necessary papers and application forms with me.

Church: Papers? Oh, I see. Are you serious about applying for a patent on my invention? It is just a kind of calculus and if I had not hit on it, somebody else would had.[1]

Lawyer: That's precisely the point. Our competitors are not sleeping and we have to rush now to get this patent. It could be a good source of income for our university. If we hit the jackpot, you will not have to go begging for funds for your research any more.

Church: OK, let us bring it behind us.

Lawyer: We have to wait a little. I invited a venture capitalist from the West Coast, who is very interested in your, eh.. how do you call it? the gamma calculus?

Church: Lambda calculus, please.

Lawyer: Excuse me. In law school we read a lot about the Romans and Roman law, but I always get mixed up with the Greek alphabet.

Church: Well, you know ... I wanted to use another symbol instead of lambda, but the typist couldn't find it in his box and so we settled for lambda.[2] At the time I thought even lawyers would understand it.

*Freie Universität Berlin, Department of Mathematics and Computer Science, rojas@inf.fu-berlin.de. This text was written for my students in the course *Algorithms and Programming*, winter term of 1997.

[1] A. Church, "An unsolvable problem of elementary number theory", *American Journal of Mathematics*, Vol. 58, 1936, pp. 345–363.

[2] J. B. Rosser, "Highlights of the history of the lambda calculus", *Annals of the History of Computing*, Vol. 6, N. 4, pp. 337-349.

Lawyer (*looking through the window*)**:** There he comes. Our businessman just arrived.

Church: He looks pretty young. Does he have a degree? I mean, my lambda calculus is pretty tough stuff.

Lawyer: Well, I think he was a couple of months at Harvard. He is so young that you can call him Bill, just like everybody else does.

Church (*horrified*)**:** You mean, he is a college dropout? And from all possible investors this is the one you have selected!

The Lawyer has no time to answer. The businessman is already entering through the open door.

Bill: Good morning everybody! Professor Church, I assume. Glad to meet you. I have already met your lawyer.

Church: Please take a seat. Was it easy for you to find my office? I mean, you are not really so familiar with universities.

Bill (*laughing*)**:** I had enough of the academic life some years ago. Just as Karl Marx said: Philosophers like to interpret the world, but what really matters is transforming it.

Church: A good quote from the "Theses on Feuerbach"! You may be young, but you are clever.

Bill: Look, professor Church... May I call you Alonzo?

Church: Professor Church, if you don't mind.

Bill: Professor Church, all right. I don't want to waste your time and for that matter either mine. We are very interested in marketing your lambda calculus. My LambdaSoft Corporation wants to bring this invention of yours to every American household.

Church: Wait a minute. Your company's name is LambdaSoft? Is it a coincidence?

Bill: Not really. We have so much confidence on this product, that we already registered the name "lambda" as a trademark.

Church: Are you crazy? You cannot make a Greek letter a trademark! It would be so outrageous as patenting common words like "chairs", "tables" or even "windows".

Lawyer: Professor Church, I guess he is right. He already has a copyright on the name "lambda".

Church: Oh my goodness!

Bill: Don't worry, professor Church. We have a good deal for you. But first, we would like to add some features to your lambda calculus.

Church: Features? Lessen, you cannot add something to an already complete system. With the lambda calculus you can compute anything you would dream off. As a matter of fact my colleagues call this the "Church Thesis".

Bill: Sure, our experts already did their homework and we are aware of this.

Church: Do you know how I became convinced that my lambda calculus is capable of universal computation? A very bright student of mine, Kleene, was able to define the predecessor function after all my efforts had failed.[3] With the help of the predecessor function you can define many other functions like ...

Bill: Professor Church, *we know*. But still there is a problem: arithmetic is just to slow and messy in your system.

Church: Messy? There is nothing more beautiful than having functions represent numbers. If you want to add 3 – that is s(s(s(0))) – and 2 – that is s(s(0)) – you just apply the successor function three times to the number two and you get the correct result s(s(s(s(s(0))))).

Bill: Certainly. But if you want to compute thousand plus thousand you have to apply your successor function one thousand times and it will take forever. We have a solution for this problem.

Church: A solution? What solution?

Bill: Before I can go into more detail you have to sign this nondisclosure agreement. You see, this is a hot industry and if we don't protect our intellectual property we won't go far.

Church (*signing*): This is ridiculous. So, you have now my signature. What do you want to use for the arithmetic?

Bill (*lowering his voice*): We call them "binary numbers".

Church: You mean Leibniz' dyadic numbers?

Bill: All I know is that some time ago a guy from Germany invented them, but we already bought exclusive rights. The LambdaSoft Corporation is the only one entitled to use them.

[3] R. Gandy, "The confluence of ideas in 1936", in: R. Herken, *The Universal Turing Machine – A Half Century Survey*, Oxford University Press, Oxford, 1988, p. 70. Church had doubted that the predecessor function could be defined in the lambda calculus.

Church: And how do you want to use them?

Bill: The main problem with your lambda calculus, as our marketing people see it, is that it's not fast enough. We can do the following: we will add special equipment that will transform numbers from your representation into our binary code. Then we can add this numbers really fast, retransform the result to your representation and give it to the user. Actually, if we introduce the symbols 0, 1, etc. for the digits, the user does not have to use your representation any more. He just writes his decimal numbers, the operation he wants and we provide the result.

Church: But that destroys the beauty of my calculus! In my system everything is a lambda expression. And we have enough symbols, we have all letters of the alphabet.

Bill: Calm down, professor Church. It might be that you can do anything you want with the letters from "a" to "z", but this is not very easy to use.

Church: And who cares?

Bill: Well, I have heard that your students called a strike last term because they couldn't understand your lambda expressions any more.

Church (*angrily*): Who told you that? Our students are just very young and impatient. I was showing them how to compute the factorial of seven and I run out of space at the blackboard. We had to continue using the wall all the way up to the second floor. And of course the end result came out nicely.

Bill: That's precisely my point. The final result will always be correct. You don't even need to tell your students or our users that we are using binary numbers. We will include with each system a sealed box to perform numeric computation. The lambda expressions representing numbers are entered and the result of the expression comes out like magic.

Church: What happens if you give your black box lambda expressions which are not numbers?

Bill: Oh, in that case the result will be false. That's why you have to take care of only giving numbers to the black box. We propose that you divide expressions into two classes: numbers and non-numbers. We call this classes "types".

Church (*horrified*): If I understand you correctly, you want us to distinguish between different kind of expressions by looking at what they represent.

Bill: Exactly.

Church: And what else? Don't you want to assign colors to the functions?

Bill: Don't be silly. We want to make life easier for the user. Otherwise we won't be able to put your system in every home, every classroom and every hospital in this country. We call it the national lambda infrastructure. And what we also need is good output. We want to connect your system to a typewriter, so that the final solution is printed. Therefore we also need a class of expressions that represents characters of the typewriter.

Church: That means you want to introduce another type?

Bill: Oh yes, but that is all we need, at least for the moment: a type for numbers and a type for characters.

Church: And how do you want to implement this system? You need a kind of "mechanical brain" capable of understanding my lambda calculus.

Bill: Well, since you have already signed the nondisclosure agreement I can tell you. There is a guy in Cambridge called Alan Turing – not this Cambridge, but the other one in England – who has been working in his garage and has come up with as solution. He calls his invention a "computing machine", but at LambdaSoft we refer to it as the "Turing Machine".[4]

Church: And what is this machine supposed to do?

Bill: Well, just as your lambda calculus, it can compute anything you would like. However it is even more difficult to handle than your calculus and it works with an infinite tape in which numbers are stored. But if we put the lambda calculus on top of it, the user will never now that his calculation is being performed in such a machine.

Church: Maybe you have an idea on how to construct such an infinite storage.

Bill: There is no need. Our engineers just calculated what is the maximum amount of storage anybody on earth could possibly want to use. They came up with 640K possible storage cells.

Church: What do you mean with 640K?

Bill: Oh, excuse me, I forgot to tell you. One K is around 1000, so 640K are more or less 640,000 storage cells.

Church: How did you get that number?

[4] A. M. Turing, "On computable numbers, with an application to the Entscheidungsproblem", *Proceedings of the London Mathematical Society*, Vol. 42, 1937, pp. 230–265. The paper was submitted on 28 May, 1936, probably after Bill had talked to Turing.

Bill: I would have to ask, but it seems to be a result from thermodynamics and quantum mechanics. Our engineers think this limitation of computations to 640K is a natural law, maybe as important as the entropy law.

Church: I'll have to check that with Professor Einstein, he is working in the other building.[5]

Bill: Of course. By the way, he has already ordered one of the new machines. He wants to calculate something called the "cosmological constant", I think.

Church: So you are already selling nonexistent machines?

Bill (*laughing*): We have to be faster than our competitors. We even have a name for your invention: it will be introduced as "Lambda 36". The number helps to remind users every year that they don't have the latest version. Of course we will come out with a new version every two or three years, i.e. "Lambda 38" and so forth.

Church (sighing): You seem pretty sure of your success? What if other companies start selling similar machines?

Bill: Don't get me wrong Professor Church. We are not so interested in the machines themselves. They are just pieces of steel and cables. Everybody can build them ... even morons at office equipment companies like IBM. What we want to sell is "Lambda 36" as the interface between man and machine.

Church: But somebody could copy the main idea.

Bill: Nope. With the types we are introducing into lambda calculus, the system has a definite "look and feel" that we can patent. Nobody can copy it without risking a visit from our lawyers. Also we will make sure that the "look and feel" evolves so that nobody can cope with us. We call this the "operating system" of the computing machine.

Church: And how will users distinguish one system from another?

Bill: Very easy. We'll put a "Lambda inside" sticker on every machine. People on the street won't even know what that means but they will be afraid of machines without sticker.

Church: I guess we are nailed. Do as you want and fix all details with our lawyers. As for me, I don't want to know anything more about this project.

[5] In 1932–1933, when Hitler came to power, Einstein was lecturing at Caltech. He accepted a position at Princeton in October 1933.

Bill: Don't worry Professor Church. We are looking forward to the Road Ahead. In a few more years you will be surprised of what has come out of all this.

A handshake seals the fate of humanity for years to come. Pure lambda calculus has met the real world. After some years the letter λ *will have transformed in the letter C.*