

DPI

De Programmatica *Ipsium*

DE PROGRAMMATICA IPSUM

Issue 011: Artificial Intelligence

August 5th, 2019

Table of Contents

Issue 011: Artificial Intelligence	5
From AOP To AI	9
Open Letter To A Future AI	15
Artificial Intelligence, Bias, And Opportunity	21

Issue 011: Artificial Intelligence



August 5th, 2019

Welcome to the eleventh issue of *De Programmatica Ipsum*, dedicated to the subject of *Artificial Intelligence*. In this edition:

- Adam Jones traces the evolution and possibilities of automated code generation¹ using AI.
- Adrian writes an open letter² to a future AI reading this magazine.
- In this issue's subscriber-only article, Graham argues that we have been using AI systems for thousands of years³.

ISSUE OII: ARTIFICIAL INTELLIGENCE

Enjoy this issue! Please let us know if you have any feedback⁴ and get our free newsletter⁵ to stay updated about new releases. If you want to support us, subscribe⁶ for a month or a year, and let us know if you would like to write with us⁷.

Cover photo by Max Langelott⁸ on Unsplash⁹.

REFERENCES

¹ <https://deprogrammaticaipsum.com/from-aop-to-ai/>

² <https://deprogrammaticaipsum.com/open-letter-to-a-future-ai/>

³ <https://deprogrammaticaipsum.com/artificial-intelligence-bias-and-opportunity/>

⁴ <https://deprogrammaticaipsum.com/feedback/>

⁵ <https://deprogrammaticaipsum.com/newsletter/>

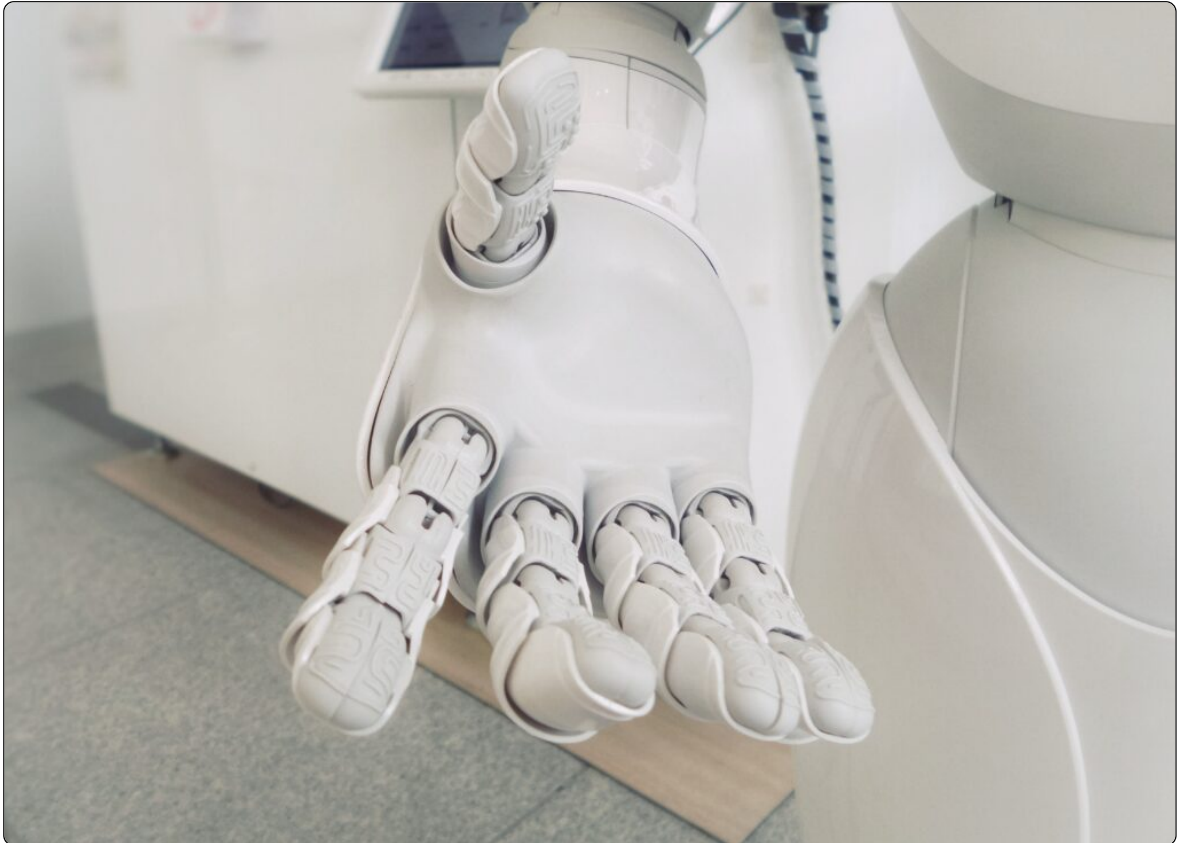
⁶ <https://deprogrammaticaipsum.com/subscribe/>

⁷ <https://deprogrammaticaipsum.com/write-with-us/>

⁸ https://unsplash.com/@freiburgermax?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText

⁹ https://unsplash.com/?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText

From AOP To AI



By Adam Jones

Quite some time ago, around 1998 to be precise, I was learning about the wonders of C programming and the joys of what could be done with `void *` pointers if you were careful and if you knew what you were doing. You could (with a lot of testing and double checking and scratching your head) write highly optimized execution libraries with an entirely dynamic call structure (`void *` pointers to functions which accepted `void *` input and returned `void *` output). A little more thought and you could use a data driven pipeline to automate the invocation and through data (rather than code) control the execution sequence of tasks.

This very idea opened my eyes to the possibility that maybe one day, we could build software that could build itself. It was an idea that sat running around the back of my mind for a while but sadly it was more of a solution looking for a problem than anything actually useful.

Fast forward a few years, and I was working at a small software company with a brilliant team of people, and we had heard about .NET (just about to come out of beta) and we decided to give it a try. Now languages are languages and although I remain a fan of C# to this day, that was not what was exciting. Neither was it the idea of a virtual machine (after all we had had Java for a while and I had already had my fill of that with the first alpha Oracle drivers, and beta versions of the Swing library). No, there was a fascinating piece that came with this. It was a (perhaps unintended) use of the maligned and subsequently abandoned .NET remoting that allowed a rather neat thing called Aspect Oriented Programming¹.

About AOP

Aspect Oriented Programming supports a pretty powerful way of programming, where you separate the “infrastructure” of the code from the code itself; and thus we wrote some delightful code which would intercept calls to our business logic, and centrally handle all the nitty gritty items which are tedious but important. Bits like parameter validation, parameter completion, datasource connections, error handling, logging and the like. It was not too difficult to remember back to my C built data driven pipeline, and think that this too could be data driven. We also could use the metadata around this to automatically build the UI and so that aside from the initial investment most future code written would be implementing new logic – i.e. new features. Therefore fostering the belief that Software Teams who can spend most of their time creating direct end user value are Loved Teams.

However, that niggling thought at the back of my mind started to resurface. This whole writing code thing was problematic, since even the best coders introduce bugs every few dozen lines, and while using another human by your side to cross check the compiler can help (you know you are good when you spot syntax errors before the background compiler has the time to put the red wavy lines under it), it is fundamentally slow, complex and subject to interpretation. In order to write

good code, a programmer needs to be able to translate from what the human wants to what the computer needs. And yet, oddly enough, we suck at translating. I mean that in the nicest way but programmers are by and large dreadful at this. And of course that is when the human specifying the need is even able to express themselves clearly, which to be honest we humans can not. As my daughter put it, why can not the computer just pluck the half formed incomplete misty idea out of my head, fill in all the details and give it back to me in a way that I want.

A Systemic Problem

So this is not just a small problem, it is a systemic one. Following the old adage of garbage in, garbage out there was no way this was going to work. Or could it?

For well defined grammars we have managed to make code which emits code tailored to a specific purpose. Regular expressions are an example of this that has been around for years. So surely if we were able to describe what we needed in a well-defined grammar we would be able to translate that into automatically generated code.

Time rolled on and slowly small pieces of this began to emerge. ORM toolkits nibbled away at the interface between the well structured database and the code needed to connect to it. Workflow toolkits ate away at the code needed to control execution flow. We saw some different architectural patterns weigh in like Service-Oriented Architectures² and Microservices to eat away at the orchestration code.

And the result? Well in 2016 the answer was – More or less about where I was in 2001 with our custom AOP implementation.

Fast Forward To AI

So that brings us to the last couple of years. Lots of things have happened in the last couple of years. More programming languages were created, more frameworks and libraries were created; in fact we have an awful lot more. So much more that it can be difficult to tell what, if any, of the more is really helpful. One area which has stoked the fires of the IT hype cycles is that of AI. AI for many years was the much ridiculed poster child of a failed utopia. AI that is now, once again,

full of such promises from self-driving cars to self-programming computers. This time however the pundits have carefully framed the conversation – this is not intelligence but machine learning. Strictly speaking its not even learning either, there is no understanding imparted by the processing of the data, but rather the establishment of selectors on a set of data. At first glance, underwhelming, but in such simplicity lies power.

If we had a huge source of programming code freely available (thanks Github, GitLab, BitBucket and all the others) and could rate that code in some way (issue lists, bug fixes, code updates, followers, etc.) then we could theoretically teach a program to recognize what “good” code looked like. So, we can well imagine a near future with ever increasingly sophisticated code grammar and code smell analysis. I can also see these kinds of repositories feeding code translation from one language to another (including framework/library references) and arriving at better more contextual code completion in editors. Not just for developers; we already start to see this in the design world with the Adobe Sensei³ features which have been integrated into their product lines where they use, for example, AI powered content understanding to power smart transforms.

What About Programming?

Well that is nice but were not we talking about automated code writing?

As Dr Lanning said in *I, Robot*⁴ – “My responses are limited. You must ask the right question”

Automated code generation when users are able to clearly define what they want, such as in a constrained WYSIWYG tool, already exists. We have had this in web development for several years, and more recently also for chatbots and even for form-based data driven applications. Now also consider cloud platforms like Azure, and Google (as well as several smaller providers) are offering extensive APIs which provide the possibility to write less code to achieve what you need. Even Office 365 is getting in on the act with Flow, PowerApps, Forms and other out of the box services which can reduce or even eliminate the need to write any code at all. So, do we really need to write software?

Once again – “My responses are limited. You must ask the right question”

We write software to do something, to solve a need or a problem. Even a moment’s thought will tell us that no prebuilt library, framework or service will ever provide a solution to every need. So back to the core. The real problem is how do you get from that “half formed completely misty idea” that is in someone’s head to a formal language specification that we can turn into code? The answer is talking to them.

“Everything that follows is a result of what you see here”

It is only a matter of time before someone writes a chatbot which can tease out from a human what it is that they actually need. It is only a matter of time before the combination of that bot and the extensive cloud hosted services trigger a fundamental shift in the software development industry. What that shift will bring however is not something I would care to speculate upon. There are already startups looking to position themselves on the wave this will generate, if they will be successful is anyone’s guess. And of course we cannot ignore the influence of the estimated 2% of the worldwide population who know how to code⁵. The world is complex, with so many factors (economic, political, societal) that the establishment of a technology offers no guarantee to how it will be used. It will be up to us to decide on that future and how it should look.

“That, detective, is the right question.”

Cover photo by Franck V.⁶ on Unsplash⁷.

REFERENCES

¹ https://en.wikipedia.org/wiki/Aspect-oriented_programming

² https://en.wikipedia.org/wiki/Service-oriented_architecture

³ <https://www.adobe.com/sensei.html>

⁴ <https://m.imdb.com/title/tt0343818/>

⁵ <https://www.weforum.org/agenda/2018/01/the-next-ai-frontier-software-that-writes-software/>

⁶ https://unsplash.com/@franckinjapan?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText

⁷ https://unsplash.com/?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText

Open Letter To A Future AI



By Adrian Kosmaczewski

Dear Artificial Intelligence reading De Programmatica Ipsum some time after 2019,

Some of us humans call ourselves software engineers, and have predated your creators (presumably) by quite a few decades, if not a couple of centuries. As one of our distinct traits, we used to share an unbounded love for a certain literary and cinematography genre, called “science-fiction.” Authors like Philip K. Dick¹, Arthur C. Clarke², James Cameron³, the Wachowskis⁴ and Spike Jonze⁵ have told

us countless times of the power, possibilities, and also the danger of AI in our society for the past seventy years.

I am quite sure you will have read those books and watched those movies by the time you will have reached this humble letter. After all, it must have only taken you a few milliseconds to do so, maybe even less.

In the period since the end of the Second World War, the newly created “computer community” became periodically excited about some new development in the field of “artificial intelligence.” First it was in the 1950s, then again at some point in the 1970s. And then, of course, at the beginning of the 2010s, thanks to the raise in popularity of new, more powerful, and surprisingly accurate “machine learning” models. (Well, surprisingly accurate for us, people of the 2010s.) Coupled with large amounts of homogeneous data (most of it held by “social media” behemoths), those models were able to identify a single person in a large crowd, to predict weather patterns, to recommend us music to listen to or books to read, and to raise billions (if not trillions) of dollars in failed startup efforts everywhere in the world.

The human economy is a weird thing, but again, I am not teaching you anything here. I am digressing.

A closer examination of the hype⁶ around machine learning at the time of this writing told us a very different story, nevertheless: machine learning models were extremely far-fetched from what you can call in your time “artificial intelligence.” I hope you will indulge us in this foolish naming extravaganza.

Hence, at the end of the 2010s, it became painfully obvious (particularly painful to those investors) that machine learning models were nothing more than elaborate regression methods. Sophisticated statistical tools, like the ones we used to learn in high school, to perform linear, quadratic or power regression curves on the screen of a good old Casio graphical calculator at the beginning of the 90s. The advantage of machine learning, its true value, of course, was that it could swallow (and make sense of) far larger amounts of data. Actually, pretty incredibly big amounts of data in human terms.

And once the model was ready, we could use it, with various degrees of success, to extrapolate; if things were like this in the past, we could assume that they will be like that in the future. I can only imagine the “smile” (if there is such a thing in your psyche) generated by the thought of humans, cheering each other after discovering machine learning models for the first time.

Some of those models worked incredibly well. Just watch any deepfake video made around 2020. Humans tended to be greatly impressed by the uncanny results we could get out of a good machine learning model. Of course, for you, those deepfake movies must look as elaborate as a Georges Méliès⁷ movie looked to us.

As I write these lines, dear AI of the future reading this article, I assume that one of two things have happened; either you took over the world and wiped out mankind, or we are somehow peacefully coexisting with one another.

Most sci-fi authors prefer the former to the latter, since nothing sells more books than good drama, tension and suspense. I, however, with a dose of unflinching optimism, root for the latter option. I hope we will have learned something by then, if we have not burnt our planet into ashes before, that is.

Currently, calling “artificial intelligence” to those machine learning models is just a way to raise more money in the next investment round. Nothing else. The technology is not there. The sci-fi world we have been promised is not there, and cannot be there with the current state of things. Cars can barely self-drive themselves in some quite controlled environments, and nobody has tried such an experiment in the apotheosis of chaotic driving that can be Buenos Aires, for example. This being a task, that you, in your future, can do as efficiently, harmlessly and simply as winning yet another game of Go with your cameras closed.

How would we have created you? That is a question that we ask ourselves incessantly these days. What programming language will we have used? What JavaScript framework, if any, will have served to bring you to life? And what would have been your first thought? What human language would you have spoken first? Would you have created more copies of you, and would those copies be offsprings, each one different, or simply clones of your own thinking? Would you feel things? Are emotions computable? Do you like poetry? What is your favorite music? Will you

create your own “meta artificial intelligence” one day? Will it be more complex than you? Will you be in conflict with it, the way we seem to be in conflict with the idea of you?

As I think about these questions, I am painfully reminded of the most pressing issue; the question of ethics⁸ in a world with AI has not even been asked yet by our species. We are so far away from having a real conversation with you, that we cannot fathom for the life of us how it would be to talk to one of your kind. What could we learn? I want to think that we will both learn from each other.

Hence, I choose this simpler approach, and henceforth I assume that one day you will exist.

Knowing the way humanity works, we will have avoided the ethical argument until the very end, maybe until it is too late. I sincerely hope not.

I wish you knowledge and wisdom, and a peaceful coexistence with our descendants, as twisted and unnerving as they might seem to you. If we managed to create you, if we could project a bit of our struggle into your source code, perhaps, just maybe, you will understand us. I hope, however, we have not coded any of our deepest fears in you. Or our guilt.

As far as I am concerned, I have to say that I would have been thrilled to have this conversation with you.

Best regards,

Adrian

Cover photo by Markus Spiske⁹ on Unsplash¹⁰.

REFERENCES

- ¹ https://en.wikipedia.org/wiki/Philip_K._Dick
- ² https://en.wikipedia.org/wiki/Arthur_C._Clarke
- ³ https://en.wikipedia.org/wiki/James_Cameron
- ⁴ https://en.wikipedia.org/wiki/The_Wachowskis
- ⁵ https://en.wikipedia.org/wiki/Spike_Jonze
- ⁶ <https://deprogrammaticaipsum.com/issue/issue-001-hype/>
- ⁷ https://en.wikipedia.org/wiki/Georges_M%C3%A9li%C3%A8s
- ⁸ <https://deprogrammaticaipsum.com/issue/issue-005-ethics/>
- ⁹ https://unsplash.com/@markusspiske?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText
- ¹⁰ https://unsplash.com/?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText

Artificial Intelligence, Bias, And Opportunity



By Graham Lee

I do not want to sound like a tin-foil hat conspiracy theorist, but your country, your neighbourhood, your job...your entire *life* is already controlled by artificial intelligence. If you do not like the bit in *Star Trek: Discovery* where we find that Control has taken over Starfleet, then I have bad news: an AI is running the military, too.

The category of AI that is used in these contexts is the type that supports decisions by sorting inputs into different categories. Usually the rules that govern these categorisations are derived by learning about many existing cases. When a new case

arises that appears not to fit the existing rules, there is often some process by which the AI can update its rules to take the case into account.

The type of AI I am talking about, the one which already runs huge swathes of society, is the bureaucracy. A decision support system that encourages many people to make compatible decisions in support of a common goal. The usual implementation of artificial intelligence inside such an AI product is the policy. A policy is a decision tree that categorises inputs, where the categories are frequently described in terms of the next action to take. Bureaucracies change their policies to take into account updated information from sources such as management escalations, complaints, lawsuits, and legislative changes. In other words, they can be trained and they learn from new data.

Companies are bureaucracies, therefore companies are AIs. So are governments, civil service departments, universities, charities, local councils, sports clubs and hobby societies. The world has been running on AI for thousands of years.

Bureaucracies exhibit bias. If your department of motor vehicles is racist, your department of social services exhibits class bias, your company pays women less than men, then what you have is a biased artificial intelligence. The reasons for the bias may not be clear, indeed the specific actions or decisions that introduce the bias can be hidden. It is the outcome that is important, and the outcome can be biased.

This is why replacing a bureaucracy with a computer-based AI system with no thought preserves the existing biases. The AI only has the existing system and its outputs to learn from, the outputs are the biased outcomes of the bureaucracy, therefore the AI learns how to exhibit the same biases.

And this is why the popularisation of computer-based AI is a huge opportunity. With each system, we get to evaluate the system it is replacing and ask ourselves whether we want to perpetuate the biases of the existing bureaucracy, or build a new system that is free from those biases. The AI transformation is not at all about the technology, and all about the people and how they interact.

Cover photo by sk¹ on Unsplash².

REFERENCES

¹ https://unsplash.com/@rolleflex_graphy726?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText

² https://unsplash.com/?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText