# Code is Law is Code

Legalese
Jan 21, 2017 · 6 min read

Contracts are similar to software in many ways. But a few crucial differences motivate a whole new programming language custom-made for the task.

Every few years, the idea of a "Github for law" turns up on Hacker News.

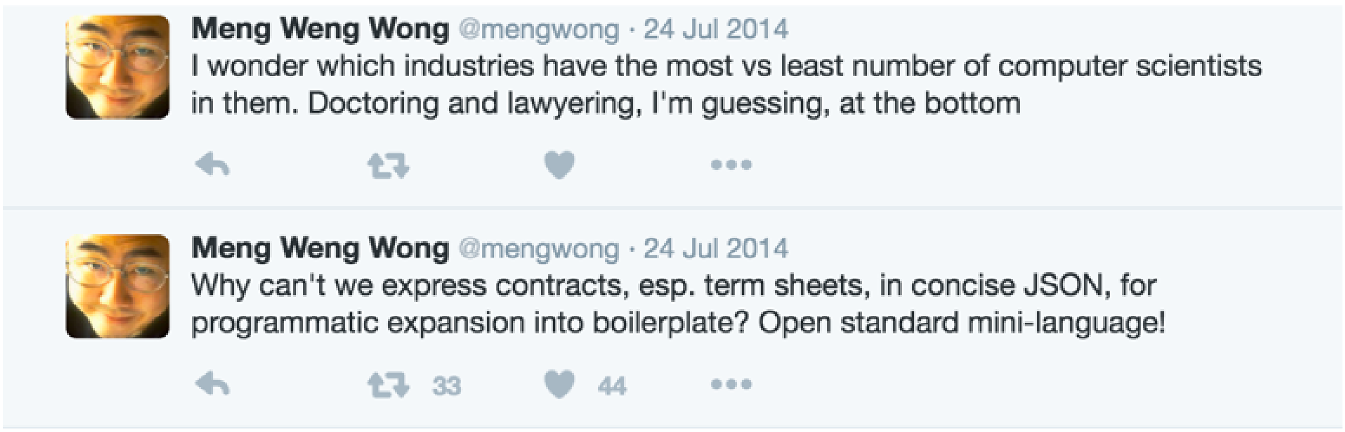
**As a corporate transactions lawyer, what I think is just as important as a "Gith… | Hacker News**

import stock_sale import confidentiality import lockup import governing_law import states import conflicts export build…

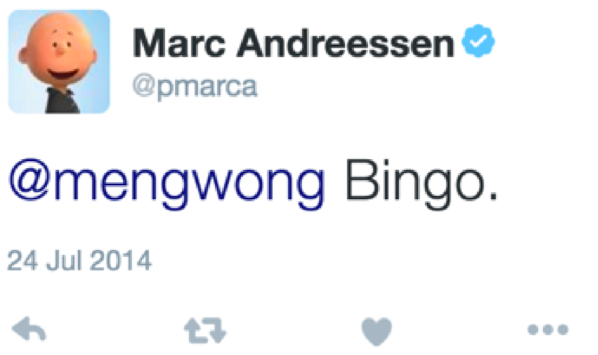
news.ycombinator.com

It's natural to think "contracts are just chunks of text; we should be able to throw them together easily. Skinny jeans, strappy sandals, a sweater: voilà, an outfit!"

That's what I thought, too, three years ago:

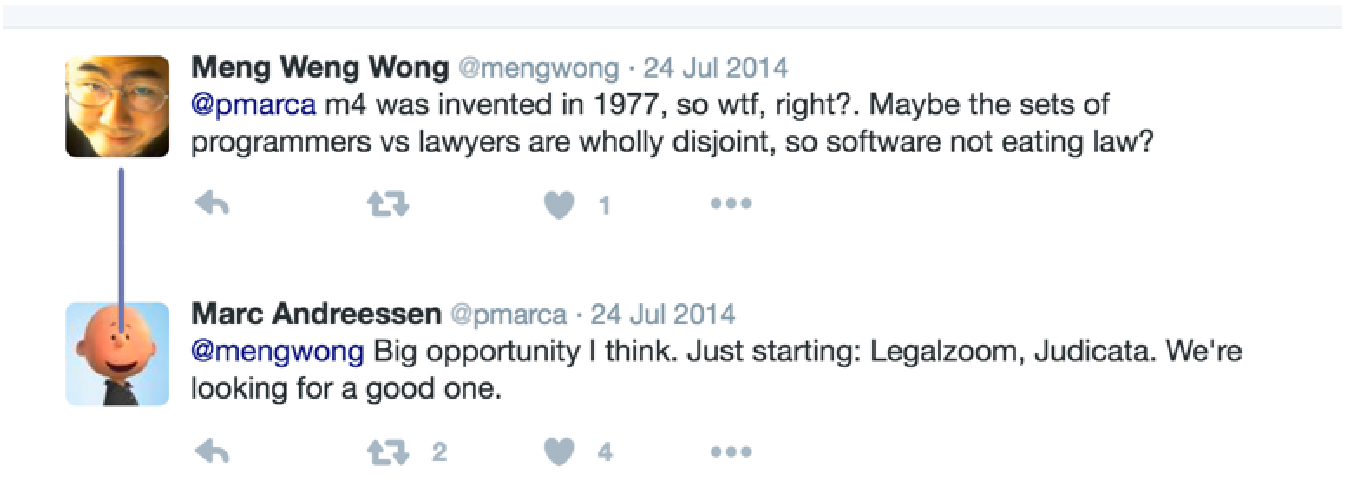

**Meng Weng Wong** @mengwong · 24 Jul 2014
I wonder which industries have the most vs least number of computer scientists in them. Doctoring and lawyering, I'm guessing, at the bottom

**Meng Weng Wong** @mengwong · 24 Jul 2014
Why can't we express contracts, esp. term sheets, in concise JSON, for programmatic expansion into boilerplate? Open standard mini-language!
33 44


**Marc Andreessen**
@pmarca

@mengwong Bingo.

24 Jul 2014



Meng Weng Wong @mengwong · 24 Jul 2014
@pmarca m4 was invented in 1977, so wtf, right?. Maybe the sets of programmers vs lawyers are wholly disjoint, so software not eating law?

1

Marc Andreessen @pmarca · 24 Jul 2014
@mengwong Big opportunity I think. Just starting: Legalzoom, Judicata. We're looking for a good one.

2 4

I have huge respect for pmarca's opinion, so I've been working on this problem ever since.

Now, I just so happen to be both an end-user frustrated with the current model of legal services and a classically trained computer scientist, so I'd like to think I have a good angle on "software is eating law".

So far, I've collected quite a few notes on past research, current efforts, and future directions in the field.

The good news: this is not a new idea; there's a fair amount of brainpower focused in this domain.

The bad news: JSON isn't enough. JSON increases abstraction but decreases expressiveness. You need the equivalent of JavaScript. (Dare we say, Turing equivalent?)

A JSON-level configuration approach may make sense for a standalone lawyer producing first-cut documents, but those documents better be in Word format, because the other side will want to edit them, and you're back to the original problem again.

The "user journey" of contracts usually involves some degree of negotiation. Yes, some contracts are take-it-or-leave-it. But most B2B contracts are batted back and forth more times than a tennis ball at Wimbledon. That means editing.

But contracts are edited, today, at the level of the text. Remember the 1980s? The era of proprietary software, before the Internet? People used to share programs on floppy disks. Not source code: binaries. This was a time when every kid had her own hex editor. I remember turning on cheat codes by opening up the .EXE and twiddling specific address locations.

That’s where contracts are today. On the commercial side, HotDocs, ContractExpress, and Exari, to name but three, do document assembly pretty much by linking binaries together to form a patchwork quilt of text blocks. There are some smarts: one of them even uses Prolog. On the opensource side, recent initiatives like CommonAccord, CommonForm, and DocAssemble, to name but three, are able to output to .docx and PDF.

But all these efforts are basically m4. It’s templates, it’s mail merge, it’s fill-in-the-blanks, it’s “Hello, %%name%%!” The world of contract drafting is pretty much stuck in the 1980s. Microsoft Word doesn’t understand the *semantics* of your text. It’s a miracle if it gets paragraph styles right.

And what happens when your counterparty wants to change just one line in the middle of your template blocks? What happens if that one change entails half a dozen changes elsewhere? You have to “recompile”. You have to update all the cross-references and adjust the clauses to match the change.

Today, lawyers do the compiling: they are the human compilers, just as there used to be human computers. But we’ve seen this show before; we know what comes next. In the history of computer science, what comes above m4 in the hierarchy of expressiveness? You guessed it: gcc. Or maybe Scheme, or Common Lisp.

Wait a minute, you say. Do we really need a full-blown language? Isn’t m4 enough? No. Look at the example:

```
export build(
      stock_sale({
        company: "Corp Inc.",
        purchaser: "Jane Employee",
        shares: 10000
      }),
      confidentiality(),
      lockup(120 /* days */),
      governing_law(states.CA),

      // If two legal terms conflict, the latter takes precedence
      conflicts.LATTER_WINS
    )
```

At heart, this is a dictionary of key-value pairs. Already, structured data is a huge win: by changing just one line, `governing_law`, we can adapt the semantics of a contract to, theoretically, any jurisdiction, any language in the world.

Now let's see what happens during negotiation. Sticking with your example, given the contract fragment above, one of the parties will say, "ah, but what about the price per share?" The majority of seed stage and Series A startup fundings nowadays involve convertible instruments of some kind, whether a note or a Safe. And that means your code needs to say this:

```
stock_sale({
        company: "Corp Inc.",
        purchaser: "Jane Employee",
        investment_amount: USD 100_000,
        cap: USD 5_000_000,
        discount: 20%,
        price_per_share: function(next_round) {
           if (next_round.valuation > this.cap) {
              return this.cap / next_round.shares_pre
           } else {
              return next_round.valuation / next_round.shares_pre *
(1 - this.discount)
           }
        },
       shares: function() { this.investment_amount /
this.price_per_share }
      })
```

It gets complex pretty quickly: indeed, the real-world example of this very case is what originally made me mad enough to crack open a text editor.

The way you deal with this sort of complexity is not by endlessly ramifying your key/value templates. You deal with it by turning to the lambda calculus: you need first-class functions. That's what allows you to pass a function as an argument to another function. You can do it in Javascript. You can't do it in JSON. And you can't do it in the majority of document assembly languages out there today.

Having a real language, with a real compiler, buys you all kinds of wins. Languages really do matter. These wins are necessary, because of the other big difference between contracts and software:

Upgrades.

Suppose you find a bug. If you're a user, you report it. If you're a developer, you fix it. Your continuous integration system takes care of build, test, and deployment.

What if you find a bug in a contract?

But in practice contracts are a lot more like hardware. Once a chip is in the field, there's no recalling it. You have to get it right the first time. That's why we're tooling our language to support formal verification, CTL* model checking, type safety so contracts can be correct by construction. Ten lines of code in our language expand, thanks to GF, to ten pages of boilerplate. You get internal consistency guaranteed not just at the syntax level (cross-references between sections) but at the semantic level: with type safety based on deontic/temporal modals. The content of the contracts can be evolved, as a constraint satisfaction problem relative to legislation also formally specified. And the signature execution workflows can be managed as a DAG.

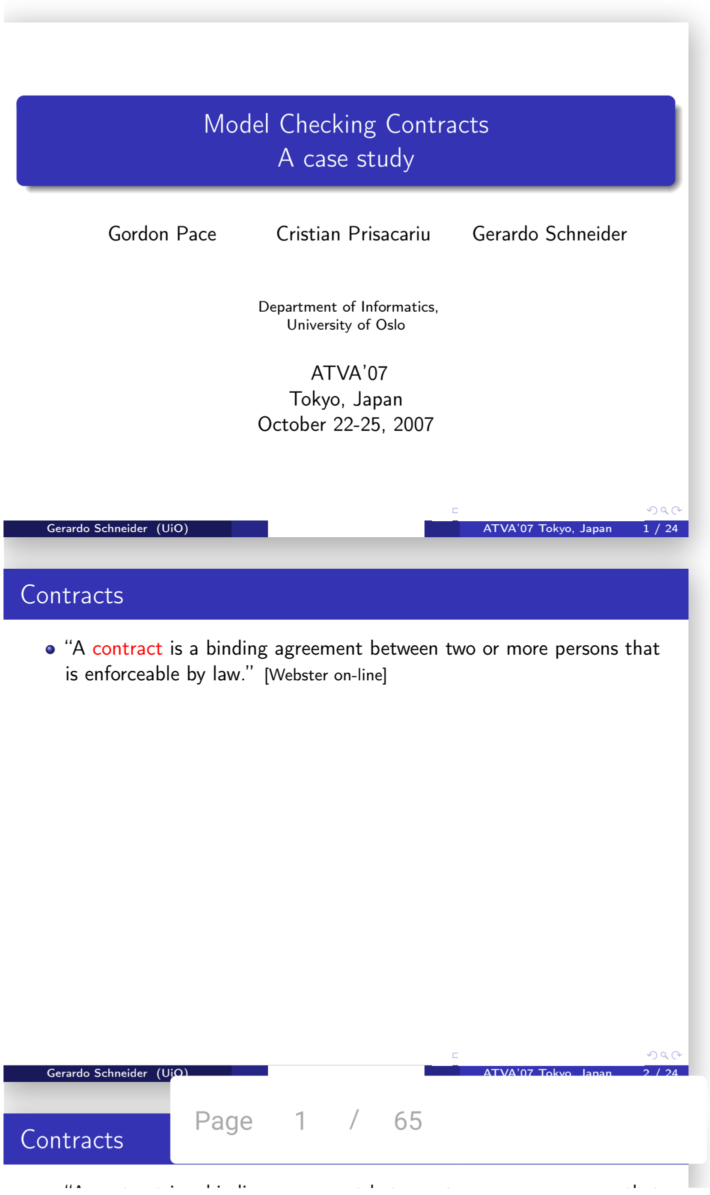
This sort of thinking is why lawyers get paid the big bucks. That's why, to take us out of the 1980s era of proprietary hand-coded assembly, a "Github for contracts" needs to contain source code written in a formal language. Anything short of that, and you haven't got a Github for law: you've got a warez site full of suspect binaries. PDFs are not source code. They're the executable.

In my next post, I'll work through a case study of how the informal languages are dangerously ambiguous: a single comma once cost a million dollars.

For now, I'll end with a suggestive illustration. Ken Adams, *the* authority on contract drafting, recently Tweeted this:

With all due respect, he’s barking up the wrong tree. He’s looking to linguists for help. *There’s got to be a better way!*

If you’re a computer scientist, you might already know what that better way looks like. Skip to slide 11 and page through the rest of this deck to see the potential of formal methods.


Model Checking Contracts
A case study

Gordon Pace Cristian Prisacariu Gerardo Schneider

Department of Informatics,
University of Oslo

ATVA'07
Tokyo, Japan
October 22-25, 2007

Gerardo Schneider (UiO) ATVA'07 Tokyo, Japan 1 / 24

Contracts

- “A contract is a binding agreement between two or more persons that is enforceable by law.” [Webster on-line]

Gerardo Schneider (UiO) ATVA'07 Tokyo, Japan 2 / 24

Contracts

Page 1 / 65

This is fundamental stuff, which even the ancients would recognize as a giant step forward. The distinction between syntax, semantics, and pragmatics goes back two thousand years, to Publius Juventius Celsus:

> *SCIRE LEGES NON HOC EST VERBA EARUM TENERE SED VIM AC POTESTATEM*
> *To know the law is not merely to understand the words, but as well their force and effect.*
> *Justinian,* Digest*, Book 1, Title 3, 17*

For the first time in history we can begin to move the not just the words, but the understanding of the words, and the understanding of their force and effect, into a machine.

If this sounds sane to you, and if you want a real-world motivation to learn those sexy new technologies you've always been meaning to try, talk to us. We sit on Slack. (This is an opensource project, and we might switch to a different, more open, messaging platform at some point. Maybe even IRC! But for now just mail us for an invite.)

— Meng Weng Wong

*Fellow, Harvard Berkman Klein Center*
*co-author, RFC4408 (SPF)*
*co-founder, pobox.com*
*co-founder, Legalese.com*

Blockchain Ethereum Legaltech Contracts