如何成为一名合格的自动驾驶工程师？

今天跟大家分享一篇来自 Felix Friedmann 的技术博客，Felix 目前在奥迪的全资自动驾驶公司AID感知组从事软件开发工作。我自己在锡根读完书找自动驾驶软件开发岗位工作的时候偶然读到了他在18年分享的文章 - So you want to be a self-driving car engineer?

原文链接如下：https://autonomous-driving.org/2018/08/15/so-you-want-to-be-a-self-driving-car-engineer/

我征得Felix本人的同意，将他的文章翻译到知乎，希望更多的朋友能够了解到如何成为一名合格的自动驾驶软件开发工程师。正文之前首先介绍一下我写这个的基本动机。

我很幸运当时看完 Felix 的文章并做了一系列准备后找到了目前自动驾驶软件开发的工作。我自己不是计算机软件背景的，也没有做过类似的岗位实习，这篇文章给了我很大的启发，让我了解到作为一名合格的自动驾驶软件开发的工程师，应该有哪些基础技能需要掌握，有哪些能力需要持续精进，有哪些坑不能省力必须填好。我想，在未来五年或者更长的时间里，一定会有越来越多的像我这样非计算机科班出身，只是在本科或者研究生阶段通过毕业设计或者实习接触到了自动驾驶领域的相关开发工作的同学，会在毕业后寻求这个领域的全职工作。既然他的文章在我看来覆盖得非常全面，那么就在这里分享给大家吧！

1. 扎实的软件工程能力是最最重要的

建造自动驾驶车辆相当于我们这个时代的登月计划。这项任务难度不亚于设计建造一个机器人驾驶员，使其在充满各种不确定性的复杂环境中为我们提供超越人类驾驶员的安全性，况且还几乎没办法预测在各种极端情况下它的同类会做出什么反应。

这种复杂度需要高质量的软件工程辅以优秀的组织能力，当在竞争如此激烈的自动驾驶领域真正尝试推进这项工作时，会发现这两方面可能都差得很远。那些搞研究用的脚本和拼接起来的代码离可靠的产品可差得远了！自动驾驶车辆中，像来自传感器，网络通信，并发等等的潜在故障源真的是不计其数，甚至时不时的系统就直接瘫痪了。为了避免在系统集成中出现这些问题，自动驾驶车辆的软件就必须为可靠性而设计。

自动驾驶车辆工程师首先得是一位专业的软件工程师，其核心职责之一就是确保高水平的软件质量。高质量的软件意味着：

- 非常 solid 的软件设计
  - 良好定义且一致的软件架构要做到位；遵循合适的设计模式及最佳编码实践；代码要做到简洁，模块化，可重用。

- 嵌入式软件开发
  - 小心谨慎处理内存管理，时序及并发；设计算法时要考虑到其计算复杂度与运行时间边界。
- 测试覆盖率
  - 所有的产品代码推到 master 分支之前都要确保达到一定的测试覆盖率。每一次的代码构建都要在 CI(Continuous Integration) 中进行单元测试，冒烟测试，集成测试；从而确保代码的健壮性。
- 软件开发实践
  - 基本的常识，如合适使用版本管理系统（是的，就是这么悲伤，这还要再提一次）；分支管理；文档（C++ - doxygen; Python - Sphinx），这些都是可持续软件开发的基础。

那如果你还不熟悉这些该咋办呢？

- 多多练习， LeetCode
- 给开源项目做贡献
- 代码质量方面， <Effective C++> and Robert C. Martin, Martin Fowler, Scott Meyers
- 测试方面，"Writing Testable Code", "A quick introduction to the Google C++ Testing Framework", "Developer Testing"
- 代码最佳实践， "The pragmatic programmer" or "Clean Code" books
- 先去没这么多限制的相关领域工作

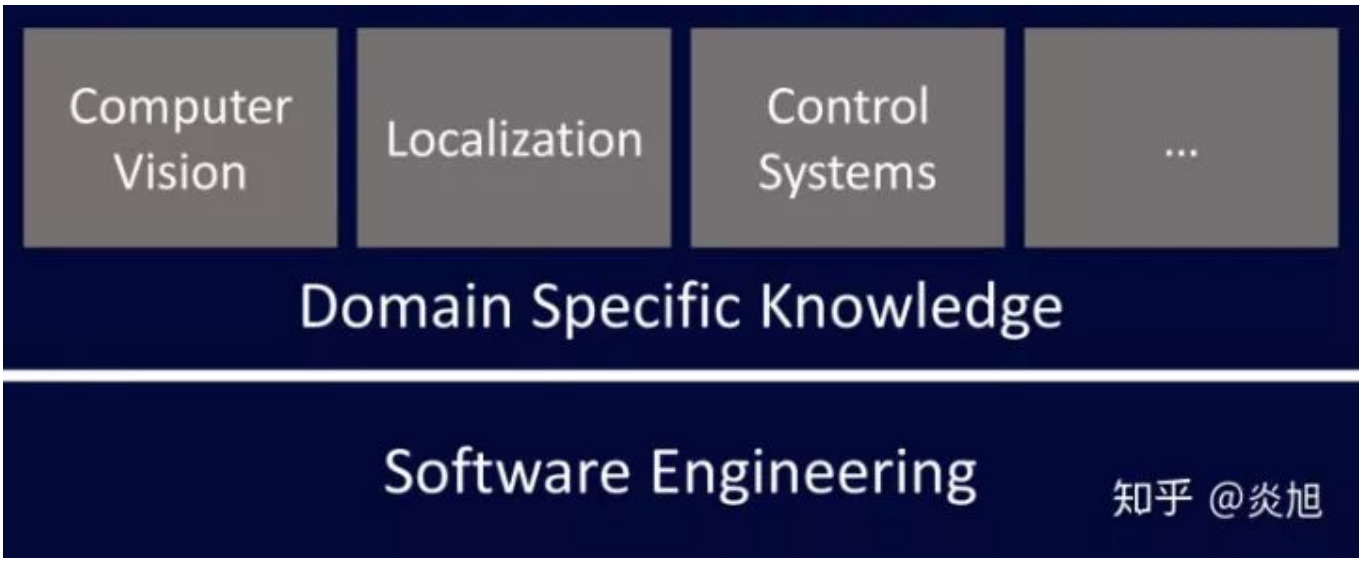
2. 自动驾驶工程师是专才，不是通才

对自动驾驶汽车所需要的不同技术有一个大致了解当然很好了，如对象检测，同步定位与地图创建，端到端神经网络等。然而，虽然各方面都略懂的通才在初创公司会有用武之地，但是开发自动驾驶汽车技术的大公司需要的往往是至少在一个技术领域表现突出的专才，而不是啥都可以做一点

但啥都做不好的通才。同时也应该注意到，构建自动驾驶软件并不只是进行功能开发，更大程度上是需要去实施像并行计算这样的架构基础。(enabler technologies)

以下是一些对自动驾驶行业工程师非常重要的能力（并不完全，也不是按优先顺序排列）

- 定位技术 SLAM
- 计算机视觉
- 软件架构
- 概率论，统计学
- 机器学习，深度学习
- 数据库技术：relational and NoSQL
- 传感器技术：摄像头，雷达，超声波
- 车辆动力学
- 高清地图
- 仿真，计算机图形学
- 实时处理，并行运算，优化技术
- 功能安全，ISO26262
- 构建系统
- 软件测试，测试驱动开发
- 汽车背景



软件工程是基础能力，再加上具体领域的背景知识

如果你具备以上所说的两方面的领域知识，而且具备 solid 的软件工程背景，那就赶紧来申请一个自动驾驶软件工程师的工作吧！

如果并没有，那就：

- 这些

    - 在线课程，像这个 MIT (国内得翻墙)
    - 打 Kaggle or Kitti 比赛
    - 其他网络资源， autonomous-driving org, Computer Vision for Autonomous Vehicles
- 实习或者工作积累一些相关经验
- 相关领域读一个研究生或者博士

3. 应用深度学习 ≠ 理解机器学习 ≠ 做一名优秀的自动驾驶汽车工程师

深度学习技术被夸大了，它仅仅构成了构建自动驾驶汽车必要工作的一小部分。事实上，找一个深度学习背景的工程师可比找一个靠谱的拥有嵌入式软件工程背景的工程师要容易得多。重要的是要理解深度学习的落地应用，然后再强调它确实是一项关键的技术，没有它也搞不成自动驾驶汽车。

但还是要注意：在线课程上，实现一些深度学习的应用，训练神经网络，通过一些框架进行调参，甚至是几行代码就能实现在嵌入式平台上的部署。这些简单的应用与那些真正深入理解其背后技术原理之间的差距是非常巨大的！

尽管能从网络上找到数千篇深度学习方面的论文，但仍然存在公共研究没有覆盖的领域。如果论文没给出代码，您应该能够亲自从头实现一遍论文中的想法。如果连论文也没有，那你就得自己去研究如何解决手头的问题了。

怎样加深对机器学习技术的理解呢？看下面：

- 主动花时间学习机器学习和深度学习理论，学习一遍你听到过但从未花时间去真正理解的那些机器学习中的名词，Batch Normalization, Vanishing Gradient, Backpropagation, LSTM, Gradient Descent..
- 跟进当前的研究，通过 Papers with Code, reddit/r/MachineLearning
- 看书， Pattern Recognition and Machine Learning, Deep Learning
- 上一些在线课程，讲理论的别上太多讲机器学习应用的
- 拿一个机器学习的学位

学好 Python

Python现在已经是学习机器学习的首选语言。语法简单库也多，特别适用于科学计算与机器学习。

遗憾的是，Python大多就是被拿来快速验证想法的工具；大家错误地认为Python没什么，对Python语言本身也缺乏兴趣，这就导致了对Python的不当使用，以致形成了庞大的无法维护的代码库，这些脚本最终也就被扔掉了，取而代之的又是一堆新的机器学习Python代码。

Python就是那种人们的自我评估与实际能力差距巨大的一个领域，这就导致工作面试中会出现一些特别尴尬的场面。如果你对Python不熟，可以这样：

- 花时间把Python当成一门编程语言来学，掌握它，并写出优雅，Pythonic的代码。利用面向对象思想写一些能重用的模块
- Python's Glossay 可以用来评估你Python到什么程度了，看看你熟悉多少这些表达式啊？
- 在线资源， The Hitchhiker Guide to Python, Code Style
- 看书， Effective Python, The Python 3 Standard Library by Example

4. 汽车软件开发

虽然一家初创公司可以在相当一段时间内使得自己免受汽车软件工程的限制，但当针对产品级的自动驾驶软件开发时，这些AUTOSAR， ASpice, Functional Safety, ISO26262, Bus, 汽车开发周期等就再也绕不过去了。作为一个工程师，并不是说这些限制都会影响到你，但你要说完全避开他们是不可能的。汽车软件开发方面的背景会让你作为一名自动驾驶工程师的日子好过得多。如果你没有汽车背景，那你也得在工作中熟悉一些基本概念。

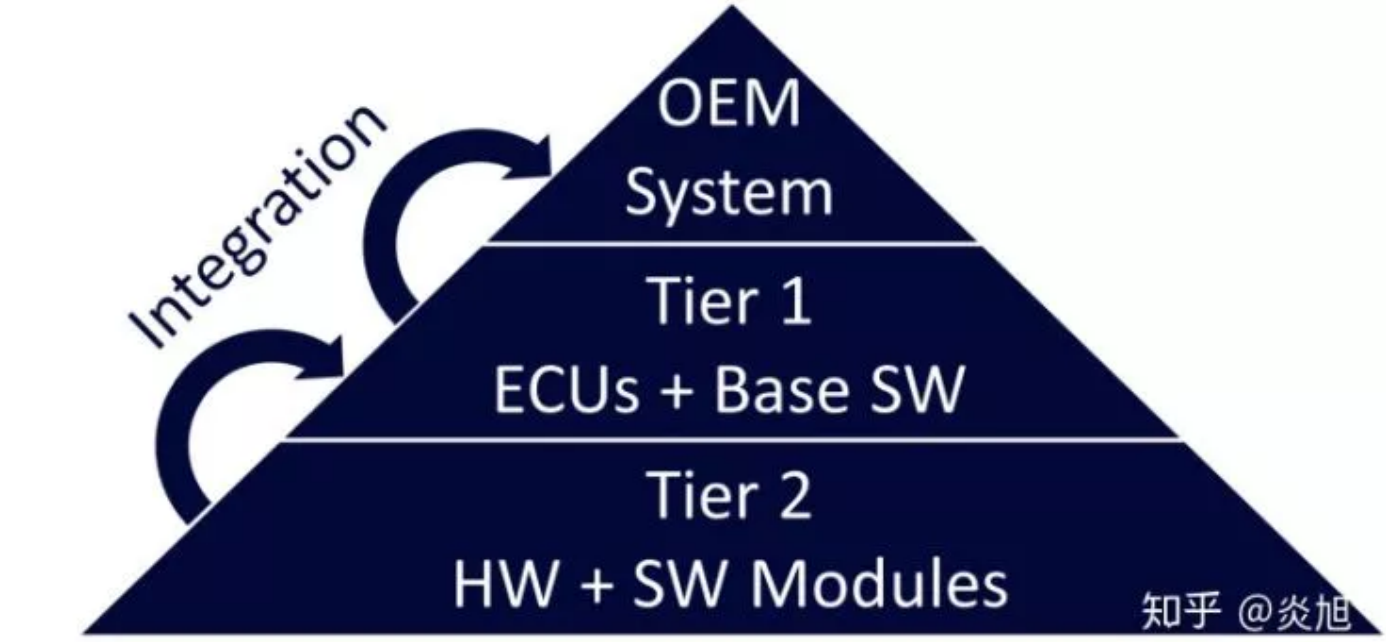
无论如何，“汽车软件开发”并不只是你需要额外了解的理论或者开发流程，它们对你的工作确实会有一些实际的影响：

软件开发创新需要时间

由于其长产品开发周期，汽车工业并不一定和现代软件工程实践保持同步。运维，敏捷开发，持续集成，测试驱动开发等等肯定不像在IT行业里应用得那么普遍。这个原因并不一定是说汽车公司在创新方面适应得慢，而是事实上就很难把这些软件工程实践都映射到已有的汽车工程开发流程中，由于其固有的内部和外部依赖，较长的交付时间，向后兼容，甚至十年前的遗留代码。

汽车软件外包带来的分散性

对于成熟的汽车厂商，工程任务尤其是软件的外包早已广泛建立了。常见的就是OEM所依赖的金字塔式的供应商。OEM集成了一级供应商的ECUs，包括里面的基础软件。一级供应商又集成了来自二级供应商的软件和硬件模块。



举个例子，比如实践中，一级供应商提供雷达系统，包含雷达传感器，用于数据处理的ECU，以及基础软件。二级供应商可以提供一个软件模块，用于对来自传感器的原始数据进行对象检测与分类。另外的二级供应商可以提供传感器的部分硬件模块，如发射模块。

这种程度的外包可能使软件工程师感到不安，但这种方式确实使汽车制造商能够构建包含大概10,000个零部件以及80个ECU的复杂产品，而他自己并不必成为所有涉及的技术，从机械到电子到软件，的全球领导者。只要看看汽车中实施的各种软件技术，操作系统，安全关键算法，移动通信技术等等，很明显，汽车厂商自己根本不可能掌握所有这些技术。那这些技术就委托给供应商去做，供应商作为各个OEM之间的共享资源。将这些开发转移到供应商可以降低风险（通过冗余）以及降低成本（通过竞争）。除了供应商开发的技术，汽车制造商自己也会确定自己在内部要开发的核心功能。

多年以来，这种外包模式对汽车公司运行良好。就在最近，模式发生了变化，由美国的科技巨头通过地图，应用平台（Apple CarPlay, Android Auto）进入汽车行业引起。也有最近自动驾驶技术进入汽车行业的缘故。很明显，事实上科技巨头可以用汽车软件挣钱啊，尤其是流经软件的数据。

同时，汽车软件的复杂性迎来爆炸性增长，需要更紧密的集成和更多的内部软件能力。因此，我们正在努力提高汽车厂商的内部软件开发的能力。虽然这种趋势目前主要还是集中在关键业务领域，如自动驾驶。

当开发自动驾驶软件时，上面提到的OEM, Tier 1, Tier 2 之间的这种隔离意味着无论你是工作在OEM，一个供应商，或者一个初创软件公司，你都很难获得驱动汽车的整个软件栈，特别是源代码。你很可能会遇到黑盒二进制文件，你可能没法访问到传感器原始数据，你可能必须得安装线控系统因为你没法访问现有的系统。汽车软硬件供应商是如此的分散，您得和多方进行交流。

交付延误

如上所述，汽车工业中，通常就是由全球的供应商网络，工程服务团队，以及OEM自己通过严格的时间规划，来构建复杂的软硬件产品。要定期进行产品的更新，必须同步调整所有的合作伙伴。为了能够在整个系统的某些部分仍处于开发阶段时取得进展，汽车工业依赖于原型级的硬件样件，（A 样件，B样件，C样件）以及成熟度不断增长的软件版本直到实际产品释放。这就会导致处于传感器和ECU还是半成品的一个中间状态，挺痛苦的。

更重要的是，为这些层层嵌套的全球活动制定精确的时间表几乎是不可行的；非常可能一个环节就是没有准时交付，你应该对因此导致的无法按质量预期交付有所准备。要给自己预留足够的缓冲时间以及B计划来弥补由此造成的损失。

“汽车软件开发”如何影响你的工作？

- 您会体验到由传感器失灵，硬件未交付，固件更新弄崩了开发平台而带来的停工
- 您得处理半工作状态的汽车，费力巴拉识别代码bug, 也会遇到随机的系统崩溃
- 您得克服过时的软件工程实践并解决遗留代码
- 您得学习汽车软件工程标准并遵守它们

5. 结论

最后，真心希望此文能够为您提供一个现实的视角，让您了解成为一名优秀的自动驾驶软件开发工程师需要什么样的能力。如果您想成为一名自动驾驶开发工程师，尽管各家公司都有自己的招聘流程和招聘要求，但上面提到的这些能力都是这个行业普遍的要求。无论如何，持续精进学习，不断提高技能都是成为一名优秀的工程师所必需的呀！