Brooks在《人月神话》一书中，提出了软件开发四个本质困难，即复杂性、不可变性、可变性和一致性。这个四个本质困难相互促进，演变并带动了软件方法的演变。软件的发展大致可以分为三个阶段：50年代—60年代的软硬件一体化阶段、70年代—90年代，软件成为独立的产品、90年代至今网络化和服务化。

在第一个阶段，软件是完全依赖于硬件的，此时软件作为硬件的附属品其主要作用就是帮助硬件完成计算任务的，这个时候软件的功能比较单一，复杂度有限，需求比较单一。因为此时软件的涉众并不是很多，主要用户是企业、学校进行科学计算为主，而软件开发团队是以硬件工程师和数学家为主的。此时的硬件是比较昂贵的，就造成了此时实践主要是Measure Twice，cut once！，由于计算机资源是比较昂贵的，所以需要在运行程序之前反复的校验，确保程序无误后才上机运行。这个阶段还出现了软件作坊、高级语言也逐渐出现，这大大降低的软件开发的门槛，此时软件的规模还是比较小，功能相对简单，这个时候的实践慢慢就演化成了“code and fix”，这个阶段程序员依靠自己的个人能力开发出来软件，在使用的过程发现错误，然后修改错误，由于此时软件的规模比较小，所以这种实践得到了极大的发展与推广。

在第二个阶段，这个阶段出现了操作系统，而操作系统的出现使得软件开始摆脱硬件的束缚这个阶段也出现了独立的软件开发公司，同时个人电脑的出现使得更多普通人可以成为软件的用户，这给这些软件公司带来机遇的同时也带来了不少的挑战。如何推出更快更好的软件产品有时甚至能决定软件公司的生死存亡。这个时候产生了形式化方法、结构化程序设计和瀑布模型等实践。形式化方法是基于数学，来对软硬件系统进行描述、开发和验证，但其门槛比较高。由于软件规模的增加，人们更加倾向于使用结构化程序设计，这时往往倾向于将一个比较大规模的问题分割成较小规模的问题，然后进行解决，但这种方法还是无法满足迅速增长的计算机软件的需求，这种增长从而导致软件开发和维护的过程中出现一系列严重的问题即软件危机。这个时候人们首次提出了软件工程的概念，通过研究软件生产的客观规律，来指导和支持软件系统的生产活动。

第三个阶段是网络化和服务化，这个阶段软件的规模更加庞大、功能更加复杂，用户的数量急剧增加，从而导致需求变得越来越不确定性，这个时候软件的传播方式也是有很大的变化的，如：以往传递软件通常是COPY，通过光盘软盘来进行软件的传播，但是现在出现了SaaS，软件是作为一种服务的方式进行分发的，如：百度网盘等。这个时候迭代时开发方式收到人们的欢迎，即大型软件系统的开发过程是一个逐步学习和交流的过程，软件交付也不是一次完成而是通过多个迭代周期，逐步完成交付。时至今日，软件渗透在我们生活的方方面面，我们越来越离不开软件了，如微信已经成为我们与家人同学联系的必备聊天软件；嘀嘀打车、美团打车也正在从新定义我们的出行方式；美团外卖、饿了吗成为了我们平时去食堂吃饭的另一种重要选择；这样的例子还有很多，无一不表示我们的生活越来越依赖软件；同时软件的部署环境也越来越复杂；如单一机器到集群，单体应用到微服务等；然而在这个时候，用户的需求进一步凸显，用户的需求期望近乎苛刻，可以用4个字来概括用户的需求：多，快，好，省，用户希望我们的系统应该功能尽可能丰富，而不是单一的功能；用户希望系统可以快速使用、并且根据最新的需求即使更新；用户希望系统应该尽可能的稳定、可靠；同时用户还希望获取软件的成本尽可能的低，最好是可以免费。这就使得在软件开发中，用户的需求极为不确定；面对这样情况，软件开发也获得了长足的进步，如空前的强大的部署环境，对于软件实施团队出现了IaaS—基础设置即使服务，对于开发人员出现了PaaS,--开发平台即是服务，对于普通出现了SaaS—软件即是服务和FaaS—功能即是服务；开源的盛行和共享文化也对软件开发带来了极大的便利；随着AI—人工只能，BigData—大数据的发展，软件工程的潜在支撑力量也获得了长足的进步。

从上述三个阶段的软件工程演变我们可以看到：软件与硬件相比，其地位变得更加重要了，软件的投入费用的比例也越来大。随着软件规模的越来越大，软件开发的复杂性也越来越大；随着软件用户的规模越来越大（据说微信用户已达9亿），用户的需求也越来越不确定，期望也越来越苛刻；这无不提醒我们：Brooks在人月神话中总结的软件开发四大本质难题在当前的互联网时代是仍然有效，为了应对这些难题，我们需要提高组织的Devops能力，以降低软件开发的风险。如：采用敏捷或迭代式开发更频繁的发布，这样可以显著减少项目变更范围，应用程序会以平滑的速率逐渐生长；加强发布协调弥合开发与运营之间的技能鸿沟和沟通鸿沟；通过强大的部署自动化手段确保部署任务的可重复性、减少部署出错的可能性。 但是Devops绝非银弹，Brooks 的没有银弹的假说依然有效，我们也不能过度的依赖Devops，我们应该更好的利用它，如：Devops环境下出现了越来越完整的工具链，我们应该好好的利用这些工具来帮我们更好更快的进行自动化部署，同时我们也要提高我们的解决问题的能力，具体问题具体分析。