快速持续集成构建 2021 版指南

Rapid Continuous Integration Builds

作者：吉姆·霍姆斯（Jim Holmes）

INCREDIBUILD

目录

我们很讨厌缓慢构建——但是还有另外一个原因

任何项目团队在使用复杂的软件时，都面临着一个令他们头痛的问题，就是构建速度过慢。产品交付团队一直尝试在各项工作任务中寻找平衡点。这些工作任务都包括：特定需求，共同探讨工作，一起完成工作，开始构建和检查软件，使产品适用于各种情况。然而，让你的团队保持这种有条不紊的节奏是非常困难的。任何能够打破固有节奏的事情，都会对产品的最终质量、甚至是传递出的价值造成直接影响。

然而，在一个项目日益壮大的情况下，想不打乱团队原本的节奏，就变得更困难了。产品增加了更多的特征，依赖性和更多复杂性，许多问题都可能会影响团队。然而，构建过程常常成为项目顺利完成的主要阻碍。你曾经有没有参与过一个项目，用时一晚，而所有的构建只完成了一次，或情况更糟，项目被遗留到周末再进行？

这样的工作氛围会使项目质量面临极高的风险：构建拖延的时间会导致团队在其余的关键环节偷工减料。最明显的一方面就是测试环节。对优秀的优先测试开发流程来说，在测试中标记出红色、绿色，然后再重构流程，花费的时间太长了，更别说是随时测试开发程序了。测试并不作为本地构建的一部分来执行——对本已花费很长时间构建的程序，谁愿意再耗费多余的时间，继续等待测试结果？开发人员将测试排除在执行范围之外——人们会说：我只有运行这几个测试的时间；我忘记了检查集成测试的副作用。

在这篇文章中，我们将重点关注因为构建速度过慢引发的相关质量问题。同时，我们将研究解决这些问题的几种方法。

时间限制决定一切

一天24小时，一周7天，一个月4周。你无法打破这样的时间限制。当然，你可以选择雇佣更多员工，但是，在有限的时间内，他们只能完成有限的工作。随着构建周期延长，团队会考虑他们还需要改进的地方。而项目团队经常要做权衡，是否冒着有可能降低项目的整体质量的风险，为了集中精力而添加更多的功能、相关资源和附加项目等。

在下一节中，我们将更深入研究这些决策的后果。

提升你的交付质量

无论你使用敏捷开发、Waterfall、 Scrummerfall，Chaos 还是其他软件开发方法，你的团队和客户都会受益于你对软件交付方式进行的改进。在产品发布时，顺利的交付流程，意味着少一些惊吓。优化交付流程，可以帮助你在修改代码库时减轻焦虑和恐惧。而进一步优化你的交付流程，则意味着你的客户可以享受到物超所值的成果——这并不是不可能的事情。

持续改进是一个旨在帮助团队和组织更好地交付软件的概念。你不必购买参考文章中描述的所有东西；而只需要重点关注几样产品就足够了——你的构建周期，应该是被优先考虑的。

反馈周期与持续提升、精益软件、敏捷开发以及其他哲学/方法学中概念的一样重要。反馈周期（或反馈循环）是反映系统中的更改所需时间的一种表达。反馈周期以长与慢的特性存在（客户的 Bug 在最新版本中得到修复了吗？）或者以更小/更快（我刚写的单元测试在运行时通过了吗？）的特性存在。关于反馈周期已经有很多文章；了解更多相关知识，可以从以下三篇文章开始：

- 斯科特·安布勒（Scott Ambler）的文章关于更快的反馈周期是敏捷开发核心；
- 罗恩·杰费瑞（Ron Jeffries）强调想要获得反馈都是好评，测试的重要性；
- 加里·伯恩哈特（Gary Bernhardt）关于极快速反馈的文章。

点击查看相关优秀介绍文章：
http://leankit.com/kanban/continuous-improvement/

1. www.ambysoft.com/essays/whyAgileWorksFeedback.html
2. ronjeffries.com/xprog/what-is-extreme-programming/
3. https://www.destroyallsoftware.com/blog/2014/tdd-straw-menand-rhetoric

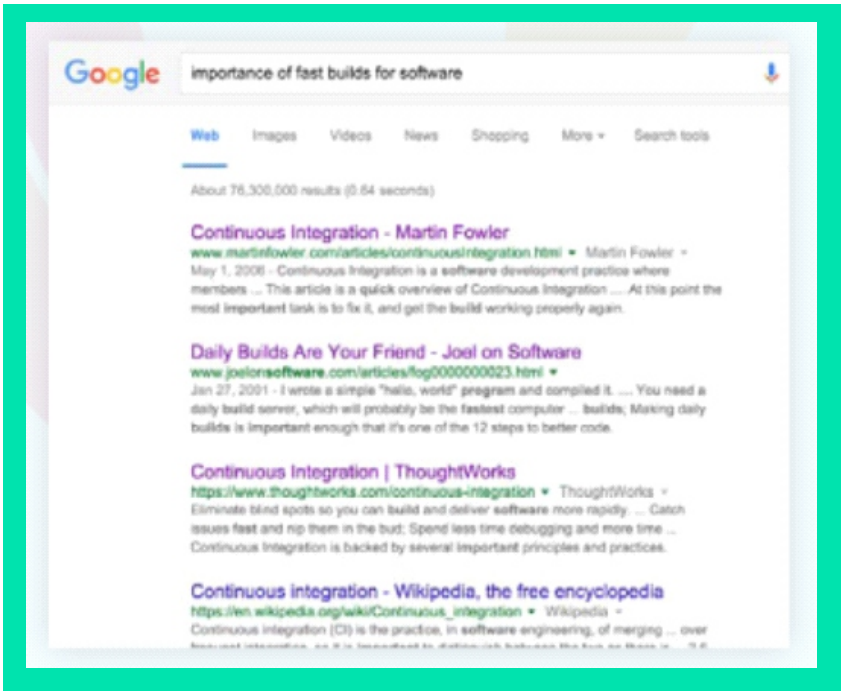
行业领军者们对快速构建的看法

值得庆幸的是，软件行业的领军者们都颇有远见。长期以来，他们一直强调加快构建周期的重要性。相关领域有大量的文章和演讲。迈克尔·费则斯（Michael Feathers）的高标准能有效地处理遗留代码，一再强调减少构建时间是控制代码库的一部分。詹姆斯·肖尔（James Shore）和谢恩·沃登（Shane Warden）的开创性著作《敏捷开发的艺术》(以及其他书籍)谈到了短至10分钟的构建周期。

简单的谷歌搜索一下就能发现过去十几年间人们思维的变化。你可以发现，2001 年的时候，周思博（Joel Spolsky）和其他人就在宣传每日集成构建的重要性。

从那时候起，软件行业就在不断迭代更新，直到现在也重点关注持续集成。持续集成的理念是每日都进行构建和集成，但是要鼓励团队在这基础上更频繁地构建和集成。所有工具平台都可以支持项目团队更频繁地进行构建/集成/部署，通常每五到十分钟一次。这极大地缩短了反馈周期。团队几乎可以得到成功还是失败的实时状态反馈，而不用再等待好几天。

我们的行业在不断进步，而我们也熟知：就算是每日反馈集成、构建和测试，整个反馈周期依旧会太长。杰斯·亨布尔（Jez Humble）和其他行业领军者们都在宣扬持续交付所带来的价值，而不仅仅是集成。



4. https://www.thoughtworks.com/continuous-integration
5. http://continuousdelivery.com/

构建过慢的影响

缓慢的构建对团队的工作有很多影响，比如打击士气、增加成本，以及影响完成高质量软件的能力。

延迟交付对客户的意义

在一天结束的时候，如果软件团队没有为他们的客户带去任何价值，那么，什么都不重要了。任何事情，只要会阻碍代码交付给客户，都是拦路虎。在《精益软件》一书中，汤姆（Tom）和玛丽·帕彭迪克（Mary Poppendieck）提出一个著名的问题："将一行代码部署到生产环境需要多长时间？"

如果你的生产环境中有一个严重的隐患，比如有个隐患会危及你的收入(可能是一个关键的安全缺陷)，会发生什么？你能在合理的时间内解决这个问题、完成构建、测试，合理分配解决问题的时间吗？或者你受到了构建过程的限制，可能需要几个小时甚至几天的时间，才能完成更改？

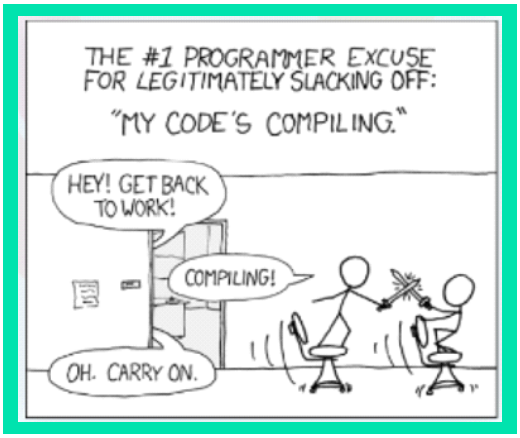
失去状态

软件测试人员，尤其是开发人员喜欢讨论如何进入一种状态。这种状态是一种精神高度集中的状态，在这里，一个人变成了一个"超级生产商"。程序员们都很希望可以进入这种状态，因为这样，他们就能够快速解决复杂问题，完成困难的任务，并在极短时间内创造价值。

当构建速度加快时，并不是每个人都满意。XKCD 臭名昭著的漫画是关于一个团队，以"有趣"的方法用构建过慢来拖延时间。也就是说，或许你可以找到另外一种方法，来满足你的团队不同于这个时代的需求。

https://xkcd.com/303/ ((Creative Commons BY-NC 2.5)

Source: XKCD



可维护性低下

人们总会下意识逃避使自己痛苦的事情。假设想象那些会让你痛苦的事情（身体受伤，情感的压力，SharePoint 开发项目），反正冗长的构建周期的原因肯定属于其一。为避免处理痛苦而又漫长的集成和测试构建，潜意识会对于项目可维护性产生负面影响。如果一个集成构建包含多个组件、分支、项目等等，而这个构建需要很长时间去运行，那么团队将会推迟运行这个集成构建。集成将推迟到晚上再构建，然后推迟到一周内构建，就这样不停被拖延。

如果一个团队处于这种状态，说明它已经向项目的复杂性屈服，而不是考虑如何努力解决问题。马丁·福勒（Martin Fowler）有一篇很好的文章，关于尽可能频繁地进行集成。他的主题是"如果这件事情让你痛苦，那就多做。"这意味着，你要经历交付过程中痛苦的部分，然后把精力集中在尽快创造价值上。

忽略自动化测试

不幸的是，缓慢的构建也会导致团队越来越少地依赖自动化测试套件。想想看：如果您已经在忍受过长时间构建带来的痛苦，那么一个自然的(但这是错误的！)反应，就是避免做任何增加构建时间的事情。这一点已在软件行业中得到了证明，因为团队编写的测试更少，运行测试的频率也更低。这导致了特别是在更复杂的系统中越来越多的回归，因为团队没有享受到自动化测试套件的好处。

这方面的一个实际例子是 ModuleWorks，它是一家面向 CAD/CAM 行业的软件组件提供商。ModuleWorks 为其客户提供 32 种不同的二进制配置。他们的构建时间很长，以至于开发人员每天只集成一次，而且他们的C++代码的 GTest 套件也很少运行。只有运用了下面讨论的 Incredibuild 之后，他们才能实现持续集成和自动化测试。

开发人员缩小了执行测试的范围

团队减少构建时间的另一个方法，是缩小他们正在测试的范围。开发人员可能会将运行测试的范围局限到当前正在处理的特定组件中，而不是再一味运行所有单元或集成套件中的测试。

这样的坏习惯会导致团队在编写代码时候产生潜在的副作用。想想看：自动化测试套件存在的主要原因之一是为正在进行的开发工程提供一个安全网！一套完善的构建测试套件，是经过深思熟虑被打造出来的，这样的测试套件可以让我们放心改动系统中的某些部分，因为我们知道其余部分是被测试给锁定的。

Retalix，作为著名 NCR Retail 的子公司，为在线系统、移动端系统和店内零售系统提供了一套最合理的软件方案。Retalix 有一套耗时十二分钟运行15000个单元的测试系统——但是在对每个开发人员的系统进行了充分优化后，运行单元测试的耗时为十二分钟，这依旧太慢了。Retalix 的开发团队就不会使用这些测试了（尽管这是他们应该做的）。

（详情可见参考案例中，Retalix 如何将测试运行时间缩短到1分20秒！）

忽略静态分析和其他质量方法

像 Visual Studio 的代码分析工具、NDepend、SonarQube 或类似的软件，对于确保代码库的质量和可维护性而言，可谓是大有裨益。这些工具可以极大程度确保代码库的质量和可维护性。这些软件会标记出质量问题比如，复杂性、依赖性和代码度量等等。也可以看作这是你的代码在通往交付程序之前的一道把控关口。

不幸的是，这些工具也需要时间来运行，当构建已经耗时很久，已经成为阻碍拥有高生产力和不能准时交付产品时，你就不会再使用这些工具了。

7. http://martinfowler.com/bliki/FrequencyReducesDifficulty.html

8. ModuleWorks加速测试和持续集成速度，增量构建案例研究；
https://www.incredibuild.cn/moduleworks-achieves-15x-faster-test-cycles-for-precision-cad-case-study-incredibuild.html

9. “Retalix 案例”：https://www.incredibuild.cn/retalix-case-study.html

人为提高构建速度

项目团队是不会坐以待毙的，而会千方百计去克服构建过慢带来的困扰。团队会花费一年甚至更长的时间和精力，用来重新配置复杂的构建周期、购买新的硬件和对整个测试套件进行重装。

用更多的硬件解决这一切

很多组织和企业第一反应是期望用硬件，来提升原本偏慢的构建速度。想象一下，在实际工作中；交换一两个构建服务器是非常容易的。但是，如果这样没有解决任何问题，会发生什么呢？除此以外，硬件的获取、设置、配置以及维护都需要花费时间。

如果你希望创建一个新的远程代理池，那么，还需要处理这些代理池的采购、设置、配置和维护！此外，将这些远程代理集成到构建系统中是非常困难的。您需要确保所有的工具链都能进行并行化运作：构建服务器、构建系统、测试框架等等。如果是希望努力就能有所回报的团队有可能会失望，发现自己的努力并没有作用。在花费这么多精力的情况下，完成速度依然会受到构建硬件所带来的限制——并且，可能还不止这些。与优化现有的硬件和处理能力相比，购买更多的硬件只为了应对高峰期，那可真是一项奢侈的提议。

手动拆分为多模块/任务

构建有时可以通过人为拆分构建工作并手动分配构建工作，来达到提升速度的效果。

然而，这种解决方案是有隐患的。团队在交付软件的工作量上已快超负荷了，现在还需要人为去管理构建的过程。这当中涉及到依赖性、软件更新以及新任务的出现。这样的人为处理方法太耗时耗力，而且对于构建团队的专业性要求非常高。

管理依赖关系对于任何软件系统来说，都是一个痛苦但又至关重要的部分。平台用于管理依赖关系的软件就像应用于 Bundler 的 Ruby、用于 Java 的 Maven、或者是 NET. 的 NuGet 等等，这些使用起来非常复杂的软件就是为了可以减轻在管理依赖关系的时，开发人员的痛苦。

拆分构建的团队现在对管理依赖关系有决定权，因为构建工作一直是在一起的话，会不知道如何处理分开构建之间的依赖性。

付出的代价

对于项目团队来说，有一点非常重要，那就是清晰的知晓：为了提升构建的速度需要付出多少努力。很多团队花费好几个月（希望不是好几年）来优化他们的构建过程。伊曼纽尔·斯拉瓦（Emanuil Slavov）在2015年的 ISTA 会议上发表"速度的需求"的演讲，讲述了一个总耗时18个月，才将测试运行时间从三小时减少到三分钟的故事。这是一个惊人的提升，但是请不要忽视这背后不可计量的成本付出。

这样的努力值得吗？对于伊曼纽尔的团队来说，答案显然是肯定的。然而，人们不知道他的团队在试错上耗时多久，或许花费了他们上百甚至上千个小时在交付上，而不是在基础设施上。

用 Incredibuild 帮助你的构建和测试提速

企业当然是希望可以加快构建速度，但是对于一个企业来说，在完成构建提速中，一不小心就会耗费了太多的人力物力。伊曼纽尔·斯拉瓦（Emanuil Slavov）在上文中提到的小故事。长达十八个月的刻苦工作之下，他们取得了巨大的成果。这包含经过长时间的努力，包括更新基础设施、构建配置、数据管理和测试自动化策略。虽然结果令人满意，但如果有其他的构建优化选项，也许这些精力应该更好地被直接用于在提升交付价值上。

10. http://www.agiletestingdays.com/session/need-for-speed/

Incredibuild的体系结构和解决方案概述

Incredibuild 可挖掘出团队和企业组织的潜力：比如说，为了大幅加快构建时间，可以快速扩展构建的基础设施——但又同时可以做到各方面（设置、配置和构建资源的管理）投入最小化。Incredibuild 允许企业使用中央构建协调器来快速建立构建代理池。开发人员、测试人员或任何安装了 Incredibuild 开始构建的人就在分布很多 agents.将长时间运行的构建拆分为多个单元，再去操作时，就可以在投入最少的时间和资源的情况下，依旧获得巨大的回报。

另外，其他需要长时间运行的任务，也可以一并交给 Incredibuild。游戏开发组织如今正使用 Incredibuild，用于加快他们图片和音频渲染，众所周知，这些都是非常耗时的工作内容。而其他一些企业利用 Incredibuild 来大幅缩短自动化测试套件执行时间并提升代码分析的速度。

有一点需要知道，Incredibuild 的 agents 与 Jenkins、Team Foundation Server 或 TeamCity 的 agents 是不同的。这些软件的 agents 和 Incredibuild 的agents相辅相成。例如，Incredibuild 将 Jenkins 构建 agents 从它自己的硬件现有性能限制中解放出来。这两套 agents 可以相互结合一起运行，从而真正地将类似于一个工作站的构建转换为有着数百个处理器和内存的系统。

Incredibuild 的功能不需要组织为承接更多耗时任务，而去投资多个新的系统。相反，企业可以依旧在现在安装的系统的上使用 Incredibuild 代理。Incredibuild 只会把任务分配至拥有适量空间的 CPU 周期和内存的系统。这意味着您可以让代理在闲置的系统上运行；在管理员或项目经理很少使用的系统上运行；甚至在开发人员经常使用的系统上运行。在后面的文章中，你将看到这些代理系统在没有源文件、库甚至是安装任何构建软件的情况下，是如何安装的。

当你需要的时候，可以轻松地将 Incredibuild 的代理池扩展到云服务器，用以获得数百甚至数千个内核来帮助你处理你的构建任务。

扩展至云端服务器

像亚马逊 AWS（Amazon AWS）、Pivotal 的云计算或微软的 Azure（Microsoft's Azure）这样的云平台都可以让组织快速扩展他们的系统基础设施。为什么类似这些平台不会支持一个组织构建和交付管道呢？

坦率来讲，Incredibuild 并不关心你在哪儿确定你的构建代理器，只要代理器可用来通讯就好了。Incredibuild 可以使用任何这些云端基础设施，只要它支持提供虚拟主机。也就是说，这意味着你需要用某种基于云计算系统的虚拟专用网络（VPN）；但是，如果你将上述我们提到的那些平台用于你的系统开发，那么很有可能意味着你已经有其他选择了。

如果你使用的是微软的 Azure（Microsoft's Azure），那么事情就更简单了：Incredibuild 对Azure 提供即装即用的支持。设置好 VPN，推送 Incredibuild 代理软件到 Azure instances，你就可以将这些 instances 拉进代理池内。如果有需要的话，你甚至可以用 Incredibuild 来自动配置新的 instances.

实用性也是 Incredibuild 的特征之一。Incredibuild 的目标，是确保组织将更少的时间用于设置和配置构建基础设施，而将更多的时间用于为客户提供更多价值。

影响最小的解决方案

Incredibuild 更注重使用简易性，这样团队可以快速开展工作。即使在大规模的环境下，Incredibuild 的安装和配置也都很简单。所有的默认设置都是精心考虑过的，为的就是客户可以享受安装后马上使用的便利。例如，为CPU利用率边界、可用RAM、磁盘缓存等预先定义了合理的最小值。

Incredibuild 更是多想了一点，提供了工具来创建一个包含着项目特殊配置的代理安装包。运行这个安装程序就会自动安装好代理软件，针对不同项目进行设定，然后和Incredibuild控制器完成连接——以上这些，都不需要工作人员动手。

最后，代理池也不需要安装任何其他相关项目软件。你无需安装 Visual Studio, Xbox development environments, gcc, dependent libraries, templates，或者任何类似的。Incredibuild的流程虚拟化，打包这些资源，并将它们发给代理。代理收到这些资源后，就开始分配任务，也不需要占用本地资源。这对于团队来说，能节省很多时间，因为团队不需要再担心升级库，修补软件，或安装那些复杂系统的补丁。（注意：对于杀毒软件和操作系统补丁这样的系统层面的资源，还是必须妥善管理！）

在真实案例中，这种方法已经让 Incredibuild 的客户显著缩短了他们构建和安装的耗时。

一些实际应用中的提升与改进

全世界很多企业或者项目团队都已经找到了 Incredibuild，为的是可以改进不同领域中都会遇到的问题——太长的构建周期。

从构建太慢到持续集成的逐渐成熟过程

Algotec 中心安装了 CARESTREAM Vue 产品套件，提供给全球数千家医疗机构使用。该套件可优化医疗成像，它建构在一个复杂的代码库上，这个代码库由400多个C++项目组成，每个项目中有30多个文件。源文件广泛使用了如宏、TLB导入和自定义构建步骤。而其特点就是集中处理。

因此，这个套件每晚只能被构建一次。开发人员更改集成又慢又痛苦，而且最终给出的反馈周期也非常长。太长的反馈周期通常会导致质量的下降，从而进入恶性循环，并且会成为一个团队的搅局者，打乱原本已经成熟的流程(如持续集成或交付)。

Algotec 选择了最简单的步骤，采用 Incredibuild 的解决方案：在每一名开发人员的系统上安装一个 agents，安装后，可以使用 Algotec 网络上的所有可使用的机器。这一步让一些小项目构建时间降幅可达90%。较大项目一开始的降幅较小：从140分钟下降到40分钟。Algotec 和 Incredibuild 共同优化了依赖性和其他项目的设置，最终将构建时间缩短到34分钟，提高了80%。

人们意识到这些节省的时间能使 Algotec 能够实施持续集成，从而可以每天进行多次构建。Algotec 还能够有时间开始实现自动化测试，如果没有 Incredibuild 从构建节省出大量的时间，这几乎是不可能实现的。

节约时间等同于更多高质量的测试

Module Works 的构建极其复杂，为全球计算机辅助制造业提供高度定制化的解决方案。他们的许多客户需要专业配置，导致 ModuleWorks 必须为每个主要版本构建至少32种不同的二进制配置。

由于其复杂的构建需求，ModuleWorks 开发人员不得不按顺序进行集成。更糟糕的是，由于构建所需要的时长（好几天），开发人员每周只能集成几次。此外，使用谷歌测试框架编写的测试由于执行时间过长而受到影响。没有编写新的测试，也没有频繁地运行现有的测试以防止质量问题的增加。

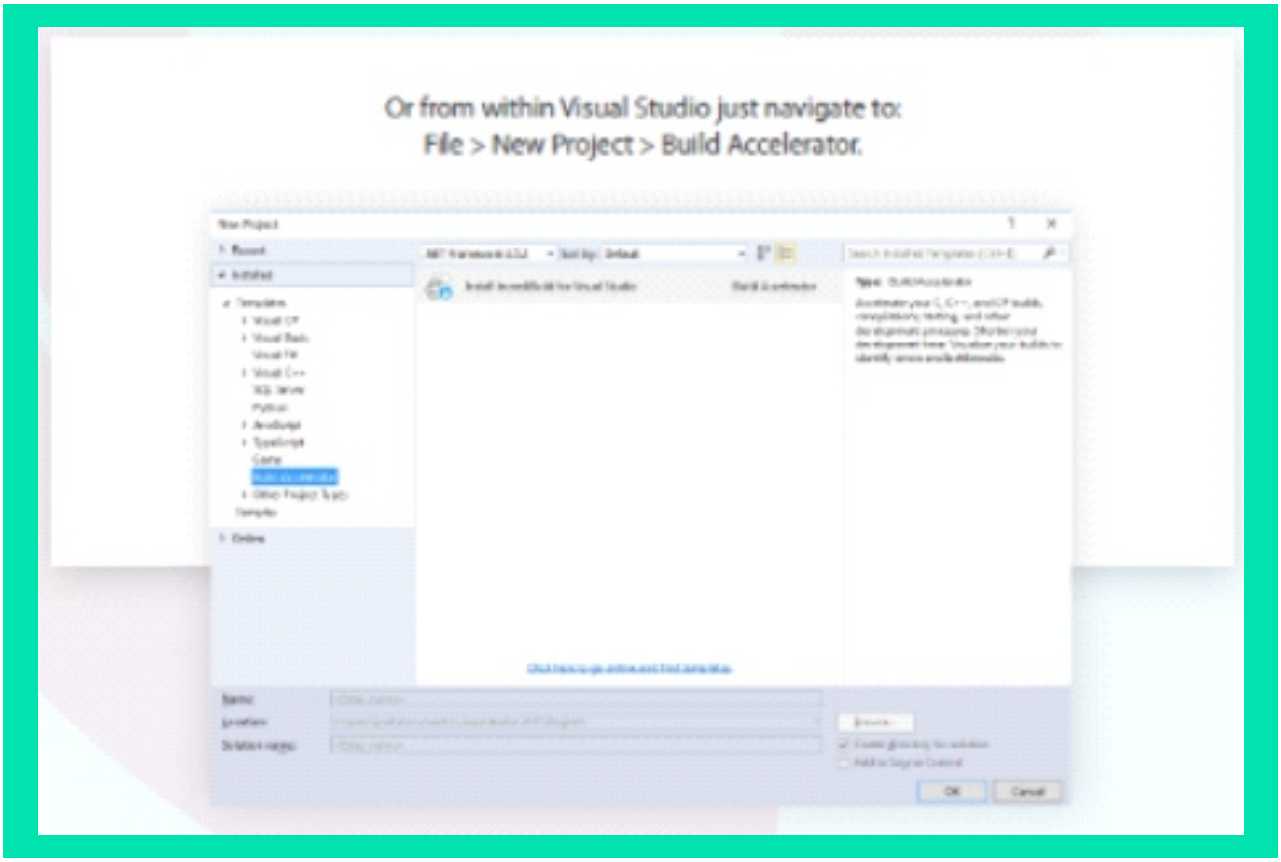
通过采用 Incredibuild， ModuleWorks，可以将构建时间从三十分钟减少到三分钟。他们的谷歌测试套件的测试执行时间下降了86%。在使用 Incredibuild 前，完整的构建和测试周期将花费三个多小时；在引入 Incredibuild 后，整个构建周期少于三十分钟。

每个构建周期节省2.5小时的时间，使 ModuleWorks 的开发人员能进行更多的自动化测试。更棒的是，ModuleWorks 在回归上的时间花费得越来越少，而用于测试的时间和频率都有增加。因此，ModuleWorks 最终产出的产品的整体质量得到了显著提升。

别再忍受构建过慢带来的折磨了；交出更好的产品

构建周期太长(不管它是构建、构建和测试，还是构建和测试和其他任务)对团队的生产力都是一个巨大的冲击。正如本文中提到的，较长的构建时间也会影响团队的交付质量水平。

任何组织都可以选择耗费大量的时间和资源，来缩短构建周期。然而，在不投入大量时间和资本的情况下，也有其他方法可以大幅缩短构建时间。任何期望在缩短构建时间这一战役中可以快速获胜的组织，都应该考虑使用 Incredibuild。更棒的是，可以先安装 Incredibuild 的免费试用版。



INCREDIBUILD

准备好和缓慢的构建说再见了吗？

免费下载 ▶

Incredibuild 的安装和配置都非常简单，毫不费力！为了你自己，快尝试吧。