Golang/dep 项目分析报告

组　　员： 苏杭 51215903022

浦家希 51215903021

程渝栋 51215903020

项目的基本背景和发展历程介绍

技术类型

Golang/dep 是官方发布的依赖管理工具。在没有官方发布的依赖工具前，go 的依赖包需要开发者手工下载。并且第三方包没有版本的概念。如果第三方包的作者做了不兼容版本的升级，会让开发者很难受。在协作开发时，没有 golang/dep 的话，需要额外统一各个开发者本地 $GOPATH/src 下的依赖包。此外开发者引用的包引用了已经转移的包，而作者没改的话，需要开发者重新修改引用。第三方包和自己的包的源码都在 src 下，项目管理很混乱。这对于混合技术栈的项目来说，目录的存放会有一些问题。

Golang/dep 这个新的包管理模式解决了以上问题。它可以自动下载依赖包，项目不必放在$GOPATH/src 内，项目内会生成一个 go.mod 文件，列出相关包依赖，所有第三方包会准确的指定版本号，对于已经转移的包，可以用 replace 申明替换，不需要改代码。

版本发布历史

Pre-merge phase(1.9 版本，至 2017 年中期)，Glang/dep 项目建立并可以真正稳定地管理依赖文件，将 dep 作为管理依赖的工具。此阶段开发者定义和制定 dep 依赖管理的命令，并设定安全模型。

Pre-toolchain release phase(1.10 版本，至 2017 年底)，优化交叉可编译性，优化错误报告，从个人开源项目拓展为私有/企业模式，改进 dep 性能，支持 cgo。

Future phase(2018 年以后)，将 dep 纳入 GOPATH 工具链中。

主要贡献者的构成

该开源项目主要贡献者均为个人，其中最突出贡献者是 sdboyer，美国人，是 GO 的成员，他一共提交了 1231 个 commits，远高于其他人，多于剩余排名前十的贡献者 commits 总和。

Golang/dep 的主要贡献者基本居住于美国，该项目 commit 在 2017—2018 年

达到高峰。go 语言是一门在欧美非常流行非常火的语言。国内近两年也开始逐渐热门。

这里主要调查了贡献度最多的人:sdboyer。其 github 的地址为：https://github.com/sdboyer。分析其 github 的项目，可以发现该作者技术比较全面，就语言方面而言，就有：Golang、PHP、Python、JavaScript 等。

dep 与 module

Go 语言有着一个热情的社区。尤其是当涉及推动语言本身及其生态的进步时，这种热情催生出精彩的讨论和许多很棒的想法，但是在前进的方向上发生分歧时，热情也会使社区分裂。关于版本和依赖管理，在 2020 年推出 vgo 和 Go modules 之前，社区主导的试验性的 dep 是很有可能成为事实标准的。

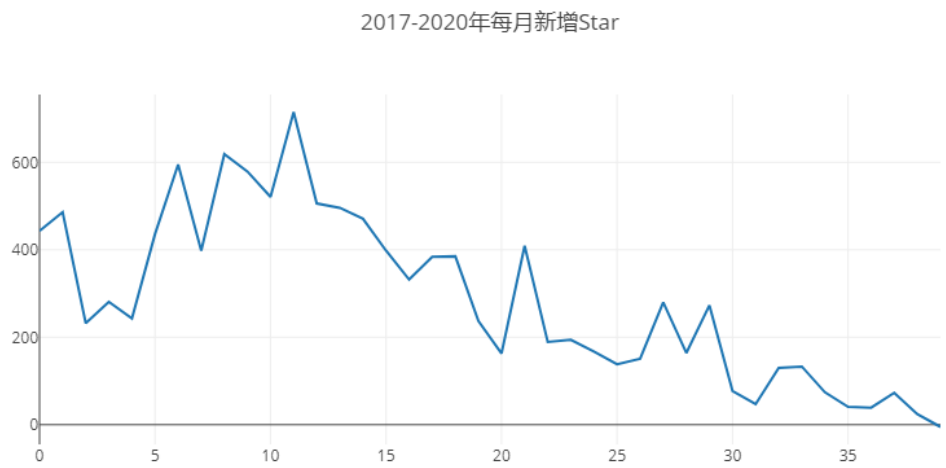
Vgo 是 Go 的参考实现，旨在提供通用的版本控制，尤其是依赖管理。Vgo 导致了 modules 的引入，以此来专门解决依赖及其版本控制的问题。截至撰写本文时，Go modules 的提案已经被采纳。2018 年 8 月 24 日发布的 Go 1.11 版本提供了对 modules 的临时性支持，2019 年 1 月 1 日发布的 Go 1.12 版本会正式支持该特性。

关于依赖管理，解决方案 dep 因其相对而言非限制的使用方法备受喜欢：导入一个库，dep 就会抓取它所能找到的所有版本。Go modules 需要使用一个 Go.mod 配置文件，并且要使用语义化的版本控制来声明你要导入哪个资源。

2017 年 1 月 Go 团队发布 Dep，作为准官方试验标准，2018 年 8 月 Go 1.11 发布 Modules 作为官方试验，2019 年 2 月 Go 1.12 发布 Modules 默认为 auto，此后 dep 被淘汰，2019 年 9 月 Go 1.13 版本默认开启 Go Mod 模式。Module 的出现也是 dep 这个项目被归档的原因，新技术逐渐淘汰旧技术，成为新的标准。

项目的历史轨迹分析

每月新增 Star 和 Frok 的个数



每月打开 Issue 和 关闭 Issue 的个数

每月打开 PR 和合入 PR 的个数

每月在仓库中活跃的不同开发者总数

Issue 从打开到关闭的平均时长和中位数（单位：天）

Issue 从打开到关闭的平均时长：255 天

Issue 从打开到关闭的中位数：194 天

PR 从打开到合入的平均时长和中位数（单位：天）

PR 从打开到合入的平均时长：10 天

PR 从打开到合入的中位数：1.5 天

洞察项目被归档的可能原因

项目相关信息：

主页、主要贡献者发表的相关技术博客

本项目的地址为：https://github.com/golang/dep。本项目一共有 181 位贡献者，共被收藏了 13.1k 次，共被复制了 1.1k 次。

这里主要调查了贡献度最多的人:sdboyer。其 github 的地址为：https://github.com/sdboyer。分析其 github 的项目，可以发现该作者技术比较全面，就语言方面而言，就有：Golang、PHP、Python、JavaScript 等。主要贡献是 gps，该技术主要解决 Go 中的依赖性管理问题的引擎。使用 gps 来复制 Go get 的获取位是非常简单的，大约只要 35 行代码。gps 项目是该作者被收藏最多的项目。其次就是 Gliph 项目，Gliph 是一个用于 PHP 的图库。它提供了图形构建模块和数据结构供其他 PHP 应用程序使用，它被设计用于内存图。

Issue 和 PR 中的相关讨论

关于 Issue 方面，现在没有空开的 Issue，一共有 1356 个 Issue。关于 PR 方面，现在没有公开的 Pull requests，一共有 894 个 requests。但从标签中可以看到，主要有一些 bug、feedback、correctness、feature_request 等，可以看出该项目虽然存在一些 bug，但是热度还很高的，有很多人会提出相关请求。

README 文件，贡献者指南，Code of Conduct 及其他可能有的相关文档

README 中主要介绍了 dep 是一个 Go 的依赖性管理工具，以及在不同操作系统下的安装流程。

贡献者指南可以参考：https://golang.github.io/dep/,其主要介绍如何使用 dep 进行实际操作，而参考资料则是对特定主题的更深入研究。该文档包含 Guides、References，其中在 References 中有个 FAQ，这里主要是问题与作者的回答，其中就有 dep 是否能取代 Go get？等含金量高和值得让人深思的问题。

Code of Conduct 主要包含了 Pledge、Standards、Responsibilities、Scope、Enforcement、Attribution。就是对参与者提出的一些要求，其目的就是共同维护一个积极的环境，供大家分享学习。

项目归档原因分析

根据该项目的内容是关于 Go 的，而 Go 语言在国内近几年是非常火热的。根据其每月的新增 star，可以发现该项目在 17-18 年是很受国内外欢迎的，正值高峰,；从每个月打开和关闭的 Issue 来看，项目前期有不完善的地方，但是随着技术的发展和关注度的提高，这些问题基本在后期都得到了解决，说明该技术已经趋向于成熟。从每个月打开和合入 PR 的个数看出，从前期来看，贡献者贡献的频率是很高的，说明项目在不断得到完善，后期基本趋向于 0，说明已经趋于完善；从仓库中活跃的不同开发者总数来看，平均每个月都在 100 人左右，说明该项目的符合很多开发者的需求，该项目很有前景和意义。

2017 年 1 月 Go 团队发布 Dep，作为准官方试验标准，2018 年 8 月 Go 1.11 发布 Modules 作为官方试验，2019 年 2 月 Go 1.12 发布 Modules 默认为 auto，此后 dep 被淘汰，2019 年 9 月 Go 1.13 版本默认开启 Go Mod 模式。

项目归档后可能产生的影响（对开发者和用户）

对开发者而言，对 Go 语言的开发提供了很大的帮助，是个比较成功和有意义有价值的项目；对于用户，体验感是最好的，因为项目是完善的，使用方面也不会出现说明太大的问题。

表述你对开源项目如何可持续发展的理解

首先一个开源项目想要得到可持续发展，其本身就需要有价值，自身有价值，才能让企业找到开源项目的盈利点，并且其涉及的技术在该领域有领先优势，这样才能吸引更多的开发者的关注，从而更加完善项目。其次除了公开代码外，还需要花很长的时间培养和经营，比如面对 Issue，要做到快速的反馈，考虑其他用户所提出的意见，并对项目的不足及时完善。当然还有一些不可控因素，开源项目的贡献主要都是开发者，程序员贡献的，但是考虑到国内程序员的工作时间和工作环境，很多人并没有时间参与开源贡献；另外在招聘岗位

上，也很少有开源工程师的招聘，就会有很多人不了解计算机中的一个分支，事实上，开源是个很有前景的方向。总之，我相信国内的开源生态会慢慢成熟起来的。目前越来越多的专注做开源的公司发展起来，也预示着一种开源趋势。