#Eclipse Che 历史及现状

##Eclipse Che的简介

Eclipse Che是一个基于Docker的云端集成开发环境。第一版发布于2014年，其名来源与它的主要开发工作进行的所在地——乌克兰的切尔卡瑟（Cherkasy）。Eclipse Che启动后可以通过浏览器进行访问，在服务其上搭建多个Workspace。可以为每个Workspace配置不同的语言支持，框架支持与开发工具支持。在每个Workspace中可以创作多个项目（Project），对于Workspace可以进行代码编写，版本控制，编译部署等操作。

经过多年的发展，Eclipse Che已经发展出了独立的开发产品Codenvy，并且支持多种开发语言、框架和开发部署、依赖管理以及版本控制工具。其中开发语言包括C、 C++、C#、 F#、 GO、 Java、 JavaScript、 PHP、 Python、 Ruby、 SQL以及TypeScript等； 框架包括AngularJS、 Docker、 .Net 2.0、 Kubernetes、 OpenShift以及Yeoman等；开发部署、依赖管理以及版本控制工具包括：Ant、 Bower、 Grunt、 Gulp、 Maven、 Npm、 Git、 Orion、 SSH以及Subversion。[https://en.wikipedia.org/wiki/Eclipse\_Che]

Eclipse Che在github上发布了开源项目，最早的标签是2016年4.X.X版本发布标签，之后一直以周为周期快速迭代，目前已经迭代到7.X.X版本的beta版，同时6.X.X版本也在更新维护，4.X.X，5.X.X版本基本不再更新维护，支持文档已删除。在进行本研究时，兼顾产品更新支持与稳定性，选择6.16.0版本作为研究对象，定义6.16.0为当前版本。同时在下文中，我通过回忆预研究阶段的内容与查找github上的历史记录等方式，列出了一些过去的功能，目的是在以后的研究中，若在网上搜到旧版本的相关资料时，可以快速绕开，避免不必要的麻烦。

#Eclipse Che的相关技术变更历史

##部署运行方式的变更

Eclipse Che的部署方式也随版本更新而更新。本人在其4.X.X版本时进行相关预研究，版本四支持以zip压缩包的的形式进行发布，需要安装Docker，解压之后通过脚本将相关组件挂载在到Docker上运行，从而启动产品实例。在版本四之后，随着Eclipse Che的Docker images的发布，zip压缩包的发布方法不再受到官方的支持，从而变为用户个性化定制的开发方法。即用户对产品进行部分修改之后将修改过的组件挂载到Docker上进行运行或调试。

而在运行5.X.X版本以及之后的原生的Eclipse Che产品实例时，需要通过命令行启动Docker，并程序从Docker镜像仓库中查找并拉取主要镜像，主要镜像在运行过程中会继续拉取其他需要的镜像，之后启动Eclipse Che实例。其中Eclipse Che的主要镜像既可以使用默认的最新版本，也可以指定某一版本。因此，用户在发布的时候可以考虑发布编译好的Docker镜像。此外，还可以将编译打包好的扩展部署到Maven远程仓库中，实现单个扩展的发布，或者将组件（assembly）打包好之后进行发布。6.X.X目前继续支持这种运行方式和发布方式。

除此之外，从5.X.X版本开始，Eclipse Che也开始支持多人模式。并且还加入了不经网页端，而通过Restful API对Workspace进行直接操作的功能。

##Eclipse Che 插件开发的技术变更

在4.X.X版本时期，项目本身并没有提供特殊的插件开发方法，开发者可以通过直接在源代码项目中plugins模块下添加代码以及在适当的位置注册插件和扩展从而达到改进Che功能的目的。

在5.X.X版本时期，该项目通过名为che-dev的Docker镜像提供了archetype服务。即可以通过运行该镜像生成一些简单的扩展模板，通过修改之后即可进行编译，加入现有的组件中生成新的产品实例。但是这种开发方法在6.X.X版本中不再继续支持。[https://github.com/eclipse/che-archetypes/tree/che6]

在6.X.X版本中，Eclipse Che本身提供了带有Eclipse Che开发环境的Workspace模板，并提供了一些Eclipse Che扩展的项目模板，就像上一版本中的archetype一样，不同的是这次可以在Eclipse Che产品中进行，即Eclipse Che可以编译自身，并部署调试运行。

#Eclipse Che开源项目结构简介

Eclipse使用Maven进行项目依赖与编译管理，Che项目下也分为多个Maven模块，下文将介绍几个主要的Maven模块。

##dashboard

负责Eclipse Che实例启动时的起始控制界面，显示关于添加，删除，配置Workspace的相关选项。由于该模块需要额外的环境配置，一般建议跳过该模块的编译，直接从Maven模块中引用已经编译好的代码库。

##plugins

该模块包含了Eclipse Che的主要扩展，一个扩展中包括IDE插件，server插件，前后端共享插件中的一个或多个。各个插件在上层pom.xml中的dependencyManagement标签下注册，并在对应的assembly中注册，等待被编译到产品中。

##assembly

assembly是Eclipse项目下负责组织编译的模块，通过assembly模块下的assembly-main模块可以生成完整的Eclipse Che产品文件，而具体各个部分的编译会依赖其他assembly来进行：如包含前端代码和gwt插件，负责界面显示的assembly-ide;包含各种以jar包形式存在的，对Workspace进行操作的插件的assembly-wsagent-war； 用于將各个插件打包进Tomcat并添加配置文件的assembly-wsagent-server；以及包含核心代码和Workspace API的assembly-wsmaster-war。通过上述模块，最后在assembly-main的target下面生成了可以被挂载到Docker镜像下进行调试和运行的产品实例。

#与Eclipse Che有关的其它技术

##Docker

Docker是一款开源应用容器引擎，为开发人员提供用于构建，发布和运行的分布式应用平台。[[1]刘国乐,余彦峰.浅析Docker容器技术[J].保密科学技术,2017(10):26-30.] Docker主要包括Docker镜像，Docker容器和Docker仓库。Docker镜像相当与一个只读的程序模板，具有完整的运行环境和程序包。Docker容器相当与一个镜像的实例或者镜像上的可读写层，依赖于镜像，可以存储数据，可以随时启动，停止和移除。Docker仓库用于存储镜像，分为共有和私有。[[1]李娜.Docker容器技术的发展及应用研究[J].数字技术与应用,2018,36(11):95-96.]Eclipse Che的默认Workspace功能模块会以Docker镜像的形式存储在Docker仓库中，而Workspace实例在运行时会以Docker容器的形式运行，停止后会被写入挂载的数据文件夹中。

##Maven

Maven ([]

#本章小结

本章前几小结介绍了Eclipse Che的发展历史，技术变更和现行主要模块。主要目的是为了提示读者在搜集资料时不会被过时的信息所迷惑。同时为在后续版本若出现与现在版本不一样的行为时，及时识别出是版本问题并快速剔除过时内容提供帮助。之后用一节介绍了Eclipse所用的其他技术，作为对上文的补充。