起源，背景

git（ギット[2][3][4]）は、プログラムのソースコードなどの変更履歴を記録・追跡するための分散型バージョン管理システムである。Linuxカーネルのソースコード管理に用いるためにリーナス・トーバルズによって開発され、それ以降ほかの多くのプロジェクトで採用されている。Linuxカーネルのような巨大プロジェクトにも対応できるように、動作速度に重点が置かれている。現在のメンテナンスは濱野純 (Junio C Hamano) が担当している。

gitでは、各ユーザのワーキングディレクトリに、全履歴を含んだリポジトリの完全な複製が作られる。したがって、ネットワークにアクセスできないなどの理由で中心リポジトリにアクセスできない環境でも、履歴の調査や変更の記録といったほとんどの作業を行うことができる。これが「分散型」と呼ばれる理由である。

<https://ja.wikipedia.org/wiki/Git>

GitHub（ギットハブ）はソフトウェア開発プロジェクトのための共有ウェブサービスであり、Gitバージョン管理システムを使用する。 Ruby on RailsおよびErlangで記述されており、GitHub社によって保守されている。 主な開発者はChris Wanstrath、PJ Hyett、Tom Preston-Wernerである。 GitHub商用プランおよびオープンソースプロジェクト向けの無料アカウントを提供している。 2009年のユーザー調査によると、GitHubは最もポピュラーなGitホスティングサイトとなった[3]。

<https://ja.wikipedia.org/wiki/GitHub>

バージョン管理システム（バージョンかんりシステム）とは、コンピュータ上で作成、編集されるファイルの変更履歴を管理するためのシステム。特にソフトウェア開発においてソースコードの管理に用いられることが多い。

<https://ja.wikipedia.org/wiki/バージョン管理システム>

リポジトリ (英: repository) とは、情報工学において、仕様・デザイン・ソースコード・テスト情報・インシデント情報など、システムの開発プロジェクトに関連するデータの一元的な貯蔵庫を意味する。日本語でレポジトリと表記される場合もある。一種のデータベースであり、ソフトウェア開発および保守における各工程の様々な情報を一元管理する。

https://ja.wikipedia.org/wiki/%E3%83%AA%E3%83%9D%E3%82%B8%E3%83%88%E3%83%AA

使用状況，位置づけ（具体例）

<https://osdn.jp/magazine/09/03/16/0831212>

gitコマンド集

<http://www.backlog.jp/git-guide/>

サルでも分かる

<https://www.ibm.com/developerworks/jp/devops/library/d-learn-workings-git/>

IBMのガチ目の解説

<http://uxmilk.jp/14664>

スライド

<http://qiita.com/YusukeHosonuma/items/14c59f3878d640a401a1>

SVNとの比較

ローカルリポジトリが使える

ブランチが切りやすい

<http://d.hatena.ne.jp/hyoshiok/20140726/p1>

歴史

<http://www.atmarkit.co.jp/ait/articles/1210/19/news025.html>

github解説

<https://ja.wikipedia.org/wiki/BitKeeper>

bitkeeper

<https://ja.wikipedia.org/wiki/%E3%82%B3%E3%83%BC%E3%83%89%E3%83%AC%E3%83%93%E3%83%A5%E3%83%BC>

コードレビュー

<https://ja.wikipedia.org/wiki/Apache_Subversion>

SVN

<https://ja.wikipedia.org/wiki/Concurrent_Versions_System>

CVS

<https://ja.wikipedia.org/wiki/Revision_Control_System>

RCS

<https://ja.wikipedia.org/wiki/WebDAV>

WebDAV

<https://ja.wikipedia.org/wiki/Secure_Shell>

SSH

マイルストーン

課題にマイルストーンを設定することで、リリース計画を立てることができます。

どのリリースで課題を対応するか、マイルストーンで管理します。例えば、現在バージョン0.9までリリースをおこなった時点で、課題が3つ残っているとします。

このうち、「SAMPLE-5」「SAMPLE-6」はバージョン1.0で対応し、「SAMPLE-7」は1.1まで対応を延ばすというリリース計画を立てた場合の、マイルストーンの設定は右のようになります。

チームはひとまず「マイルストーンが1.0の課題のみ」に集中して、プロジェクトを進めていきます。

http://www.backlog.jp/help/usersguide/version-milestone/userguide106.html

<https://ja.wikipedia.org/wiki/%E3%82%A2%E3%82%B8%E3%83%A3%E3%82%A4%E3%83%AB%E3%82%BD%E3%83%95%E3%83%88%E3%82%A6%E3%82%A7%E3%82%A2%E9%96%8B%E7%99%BA>

アジャイル開発

<https://ja.wikipedia.org/wiki/%E3%83%81%E3%82%B1%E3%83%83%E3%83%88%E9%A7%86%E5%8B%95%E9%96%8B%E7%99%BA>

チケット駆動開発

<https://ja.wikipedia.org/wiki/Google_Developers>

google code

ソフトウェア構成管理（ソフトウェアこうせいかんり、英: software configuration management、SCM）とはソフトウェア開発プロジェクトをその成果物を通して制御・管理する方法論である。ソースコードや文書などの成果物の変更履歴を管理し、製品のバージョンやリビジョンに個々の成果物のどのバージョンが対応しているかを識別し、任意のバージョンの製品を再現可能とする。

バージョン管理システムは SCM のためのツールであるが、バージョン管理システムそのものを SCM と呼ぶこともある。しかし、一般にソフトウェア構成管理はバージョン管理とは等価ではなく、バージョン管理を制御するマネジメント的要素が含まれる。

<https://ja.wikipedia.org/wiki/%E3%82%BD%E3%83%95%E3%83%88%E3%82%A6%E3%82%A7%E3%82%A2%E6%A7%8B%E6%88%90%E7%AE%A1%E7%90%86>

<https://en.wikipedia.org/wiki/Monotone_(software)>

monotone

<https://www.google.co.jp/trends/explore#q=git%2C%20SVN%2C%20subversion%2C%20BitBucket%2C%20GitHub&cmpt=q&tz=Etc%2FGMT-9>

検索数の推移によるトレンドの推移

<https://ja.wikipedia.org/wiki/Bitbucket>

BitBucket

<http://sue445.hatenablog.com/entry/2012/12/02/000132>

BitBucketとgitの比較

<https://ja.wikipedia.org/wiki/PostgreSQL>

PostgreSQL

<http://catcher-in-the-tech.net/806/>

マージの強み

先の展望，自分の考察

GitHubの発展（SNS要素の発展）

GitHubがソーシャルコーディングを広めているが，現状はボランティア的で，自らのコミットが誰かの開発に貢献しているという満足感や，GUIによって刺激されコミットが楽しくなるような仕組みなどが参加している開発者の主なモチベーションとなっている．しかしながら，それでは限界があると考える．したがってこれ以降は加えてコマーシャルベースのソーシャルコーディングが展開されると思う．利用者から使用料を徴収するのはもちろん，デバッグのコミットをすればその見返りとして報酬がもらえるような仕組みが台頭すると考える．

これで，プログラム開発者からソーシャルコーディングでデバッグを行ってくれるユーザーに対してお金が流れる仕組みができると思う．

Assembla

SVNクローン（gitではない）

http://d.hatena.ne.jp/kunikiya/20120311/1331472055

https://en.wikipedia.org/wiki/Assembla

GitLab

GitHubクローンをうたっている

http://knowledge.sakura.ad.jp/tech/3005/

Phabricator

シェル上から直接利用？

git,mercurial,svnと統合されている

https://ja.wikipedia.org/wiki/Phabricator

http://d.hatena.ne.jp/hiroe\_orz17/20130118/1358470760

gitBREAK

無料でプライベートリポジトリ

http://d.hatena.ne.jp/satox/20130612/1370961213

GitBucket

GitHubクローン

http://www.softantenna.com/wp/software/git-bucket/

gitの仕組み，内部処理

http://koseki.hatenablog.com/entry/2014/04/22/inside-git-1