これから矢吹研究室---清水竜吾の発表を始めます．

私の研究内容はオープンソースにおけるソフトウェア開発の実態調査です．●

まず，背景としてこれまでのソフトウェア開発の現場では主に，ウォーターフォール型開発手法が採用されていましたが，近年ではアジャイル型開発手法が採用されることが多くなりました．アジャイル型開発手法にはスクラムやエクストリームプログラミングなど様々な開発手法がありますがこれらスクラムやエクストリームプログラミングはIPA独立行政法人-非ウォーターフォール型開発に関する調査に関する図のデータのようにシェア率約8割を占めています．また，スクラムやエクストリームプログラミングでは●テスト駆動開発手法という先にプログラムに必要な機能に対してのテストを作成し，そのテストが動作する必要最低限な機能の実装を短い工程で，計画・開発を繰り返し行い，コードを洗練させる開発手法がとられます．

●私は，ソフトウェア開発の現状を調べることにしました．ソフトウェア開発を調べるには様々な方法があります．本研究では，開発プロセスのテストの実装工程に着目しました．開発プロセスのテストの実装において，テストコードとメインコードのコード量の時間変化を調べることにしました．

これらを調べるためには，バージョン管理がされていて，ソースコードが公開されているソフトウェアでなければなりません．●しかしながら，MicrosoftOfficeなど，これらのソフトはソースコードを初めとする，開発プロセスを一般では，取得することができません．●そこで，オープンソースソフトウェアというソースコードがバージョン管理されていて，開発プロセスが一般でも取得することができるソフトウェアを使用することにしました．●また，オープンソースソフトウェアはGitHubやソースフォージというホスティングサービスで開発されています．オープンソースホスティングサービスでは，ソフトウェアがオープンソース開発されているので，バージョン管理されたソースコードなどの開発プロセスを一般でも取得ができるように開発されています．また，本研究ではGithubを使用して，研究することにします．

●何故Githubなのかと言うと，現状のソフトウェア開発と言うことで，流行のホスティングサービスを使用したかったためです．そこで，Googleトレンドと言うサービスを使用して，Googleでの検索クエリ数のデータを元に調べた結果，オープンソースホスティングサービスの中では，近年GitHubが人気でした．●また，GitHubはユーザー数やリポジトリ数を年々増やしています．2008年サービス開始当初は約四千百人のユーザー登録数だったところ，4年後の2012年では，約二百八十万人のユーザー登録数と大幅に増加しました．また，リポジトリ数は同期間で約三万八千件から約四百六十万件と大幅に増えています．近年ではリポジトリ数が千万件を突破しました．このように多くのユーザーが様々なソフトウェア開発プロジェクトをホスティングするGitHubで調査・解析することによってソフトウェア開発傾向を調べることができます．

●本研究で，テストのコードを調べる方法として，GitHubでホスティングされているソフトウェアにはGitと言う分散型バージョン管理システムで行われているため，パソコンにGitを入れることによって，Githubにてバージョン管理されているソフトウェア開発プロジェクトをロールバックと言う過去のバージョンに戻す操作をすることができるようになります．●そこで本研究では，調査するためにGitを使ってCloneと言う操作をすることにより，プロジェクトデータをローカル環境へ複製を行う必要があります．●複製したプロジェクトデータから履歴情報を取得し，取得した履歴情報にてロールバックを行い，ソースコード量を計測します．●しかし，履歴情報はたくさんあるので本研究では履歴情報すべてに対してロールバックを行いソースコード量の計測を行うコードカウントツールを開発しました．●また，コードカウントツールで取得したデータは自動でグラフ作成ツールにポストすることによって，テストコードとメインコードのコード量でグラフを作成し，可視化することができます．

●本研究ではGitHubにホスティングされている，表に記載されている人気度の高いソフトウェア開発の全37件に対して調査を行った結果，主に四つの傾向を見つけることができた．そこで，各傾向の代表的な物のグラフを使いパターンを説明する．

●パターン１は，メインコードよりテストコードのほうが多い傾向です．

これは緑色のメインコードよりも，青色のテストコードが多いグラフのことです．

●パターン２は，メインコードとテストコードが共に成長する傾向です．

これは，青いテストコードと緑色のメインコードが共に伸びているグラフのことです．

●パターン３は，テストコードが殆ど書かれていない傾向です．

これは，緑色のメインコードは伸びているが，青色のテストコードが全く伸びていないグラフのことです．

●パターン４は，テストコードが初期状態から殆ど変化がない傾向です．

これは，緑色のメインコードは変動しているものの，青色のテストコードはほとんど変動していないグラフのことです．

●本研究にて，全37件調べた結果，それぞれのパターンがこのように分布していました．

特に私が注目したのは三番目と四番目のパターンです．「テストコードが殆ど書かれていないものや，テストコードが初期状態から殆ど変化がないもの」だけで，約50％もの割合を占めていることがわかりました．テストが作られていないのがおよそ半分を占めているのは意外な結果でした．テスト駆動開発が大事だと言われているのに，人気のあるプロジェクトでも，テストはあまり重視されていないかもしれない．

●また，その他というのは印が付いているような急激なコード変化が見受けられました．今回の例として表示されているGruntの開発プロジェクトでは印が付いているところで，他のソフトウェアへ派生していました．

●まとめとしまして，本研究ではオープンソースソフトウェア開発のテスト工程に着目して，開発プロセスを調査しました．

開発プロセスを調査するためにバージョン管理システムを使いバージョンごとにコード量を調べ，コード量を可視化するツールを開発しました．

開発したツールを使用した結果，主に四つのパターンに分類することができました．

分類した結果，テスト駆動開発は必ずしも行われていないことがわかりました．

また，このような現実のソフトウェア開発プロセスを学ぶことで実際のソフトウェア開発のプロジェクトを学ぶことができました．

このような手法を使うことによって，プロジェクトの中の人しかわからなかったソフトウェア開発のプロジェクトマネジメントを一般でも知ることができます．

以上で発表を終わりにします．ご清聴ありがとうございました．