ALMA のソフトウェアテストの技術

国立天文台チリ観測所　中村京子

ALMA は、観測運用中の現在もソフトウェアの開発（機能追加、改良、バグ修正）を継続的に行っており、その一環であるソフトウェアテストも定期的に実行している。2015 年の技術シンポジウムにおいて、「ソフトウェアテストの紹介」と題して発表する機会を得たが、今回はその続きとして、ALMA のソフトウェアリリースの現状、テストの実際、そして使用しているソフトウェアツールについて簡単に紹介したい。

１．ソフトウェアテストとは

前々回のシンポジウムにおいても“テスト”と“デバッグ”の違いについて説明したが、ソフトウェアテストの部外者には両者の区別がつきにくいきらいがあるため、念を入れて、今回も定義を簡単に述べておく。

「ソフトウェアテスト」

・欠陥から発生する故障を発見すること(「JSTQB Foundation 第 3 版」)

・コンピュータのプログラムから仕様にない振舞または欠陥（バグ）を見つけ出す作業(Wikipedia)

「デバッグ」

・ソフトウェアの故障の原因を見つけて、分析して、取り除くプロセス(「JSTQB Foundation 第 3 版」)

・プログラム中の欠陥を修正する作業(Wikipedia)

つまり、ソフトウェアテストとは、製造業における検査や商品テストに相当するものである、とイメージすれば理解しやすいであろう。

２．ALMA のソフトウェアリリース

ALMA ソフトウェアは、開発が主体の時期には半年に一度のリリースであったが、2013 年からの本格的な観測運用開始を境に体制を変更し（2014 年 、1.5 ヶ月に一度のリリースとなった。その後さらに運用体制の見直しを行い、2016 年 10 月からは月毎リリースに変更されている。リリースはサブシステム単位で行われ、オンライン系（ONLINE)、共通基盤系（COMMON)、データベース系（ARCHIVE)、観測支援系（OBOPS)、観測準備系（OBSPREP)、観測データ処理系（SPT-PIPE）に分かれる。

月毎リリースとしたのは、アジャイルの手法（少数のソフトウェア変更を短周期で繰り返す）を取り入れたためである（ONLINE のみ二カ月周期とした)。各周期の開発終了時にテストが実施され、テストにパスしたもののみが、正規の ALMA ソフトウェアとしてリリースされる。開発部隊は、日本（NAOJ)、米国（NRAO)、欧州連合（ESO）そしてチリ（JAO）各所にある。テストは IRM（Integration and Release Management)という専門のグループが担当し、やはり四サイトに散らばるメンバーで構成されているが、主体はチリの JAO に置かれている。テストは目的の違いにより、以下の三フェーズに分かれる。

Phase A：実装機能がモジュール内で正しく動くことの確認（開発作業中に実施）

Phase B：実装機能の動作確認および既存機能の動作確認（3 週間　※ONLINE は 4 週間）

Phase C：実装機能が仕様通りであることの確認（1 週間）

３．Phase B テストの一連の流れ

ここでは、IRM で行っているテスト作業（Phase B）の概要を述べる。

① テスト対象確認

リリース毎に開発すべき機能や修正すべきバグが指定され（天文学者と開発者の両者が議論の上、決定する)、一機能一チケットとして管理システム JIRA に登録される。チケットにはステータスが表示され、このステータスにより開発段階やテスト状況が一目でわかるようになっている。また、どういう機能か、どのようにテストすべきか等も記述され、コメント欄を使って複数人で議論もできる。テスターは、ステータスが実装完了となったチケットをテストすることになる。

② ソフトウェアのビルド

開発の済んだソースは、ソフトウェアリポジトリ（従来 Subversion を使っていたが、本年 10 月から Git に移行した）に保管される。テスターは、ソースファイルをリポジトリから取得し、実行プログラムを生成（ビルド）する。

③ 実行プログラムのインストール

ビルドに成功した（エラーのない）実行プログラムを、テスト環境(Red Hat Linux)にインストールする。

④ 事前検査（サニティチェック）

インストールしたプログラムが実際に使えるか、本格的なテストに入る前に簡単に検査する。

⑤ テスト

テスト結果が期待通りであれば、チケットのステータスをテスト終了に変更する。テスト結果が期待したものとは異なる場合、チケットのコメント欄を使って開発者とやりとりをする。開発者はコメントを見て、必要に応じてソースファイルを変更し、リポジトリに保管する。テスターは修正されたソースファイルを使ってビルドし、再テストする。

⑥ 回帰テスト（リグレッションテスト）

従前の機能が問題なく動いていることを確認する。何らかの問題が発見された場合、バグチケットを新規作成し、開発者はそれを確認してデバッグ作業を行い、修正したソースファイルをリポジトリに保管する。テスターは再びビルドを作成し、再テストを行う。バグ修正が最終的に確認されれば、チケットのステータスを変更して作業を終了する。

⑦ まとめ

今回のリリースのテスト結果をドキュメントにまとめる。

４．ソフトウェアツールの紹介

前節でソフトウェアリポジトリやチケット管理システムに言及したが、ここではそれらも含めて、主にテストチームが現在使用しているソフトウェアツールを紹介する。

サーバシステムの仮想化

VMware：物理的に一台のコンピュータ上に、仮想的に複数のコンピュータを立ち上げる。観測およびソフトウェアテスト環境で採用している。

Docker：アプリケーション実行に必要なすべてのファイル一つにまとめ、Linux のコンテナ上で実行する。ALMA が開発している様々な Web アプリケーションが対象である。

ソフトウェアリポジトリ

Subversion(SVN)：集中型バージョン管理システム。すべてのソフトウェアソースを保管する。ALMA

では wandisco を運用し、四サイトでサーバを同期運転していた。2017 年 9 月まで。

Git：分散型バージョン管理システム。2017 年 10 月から運用開始し、Web ベースのサービスである bitbucket を使用している。開発・テスト作業はブランチ上で行い、”master”には動作保証されたファイルのみ保管する。

ユーザ認証管理

LDAP：ユーザ情報管理データベース。観測およびソフトウェアシステムのユーザ認証に使用している。

CAS：シングルサインオンの認証機構。複数 Web アプリケーションのユーザ認証で運用している。

テストの自動化

Selenium：Web アプリケーションの自動テストツール。これに JUnit や TestNG ライブラリを組み合わせてテストスクリプトを作成する。

自動実行

Jenkins：継続的インテグレーションツール。ビルドの自動実行、テストの自動実行等に使用している。

５．今後の展望

自分が ALMA コンピューティングチームに加わった頃（2009 年）は、Linux システムは実マシン、SVN は ESO 本部のドイツだけにあり、LDAP の認証はまだ、テストはほぼ手動で行うという形で、上に挙げたツールのほとんどは使っていなかった（まだ世の中に出ていなかったものもある）。ところが、その後 8 年間で VM や Docker が実質的に標準化され、ソフトウェアリポジトリの分散管理、認証の集中管理、そして自動テスト技術が大きく伸びた。中でも自動テストは発展途上にあり、まだまだ伸びしろがある。

現在、自動テストは単純作業の繰り返しを効率化する程度である。高度なテストは人間が手動で行っており、現行、人が操作するテストに勝るものはない（機械が行うより多くのバグを見つけることができる）。ただ、最近とみに話題の AI 技術の進展により、人間を越えるインテリジェントなテストが自動で行えるようになる日が来るかもしれない。

参考文献

「JSTQB Foundation 第 3 版」大西建児他　翔泳社
「JUnit 実践入門」渡辺修司　技術評論社
「Docker 実践入門」中井悦司　技術評論社
「システムテスト自動化標準ガイド」マーク・フュースター、ドロシー・グラハム　翔泳社
「継続的インテグレーション入門」ポール・M・デュバル、スティーブ・M・マティアス、アンドリュー・グローバー　日経 BP 社