Veriserve Navigation

ソフトウェアと品質を考える

ベリサーブ アカデミック イニシアティブ 2019　特集

アジャイル開発とテスターの役割

～ビジネスとエンジニアの協働チームづくり～

平鍋 健児 氏

デジタルトランスフォーメーションの推進と政策展開

―2025年の崖を克服するための企業競争力の観点から―

田辺 雄史 氏　和泉 憲明 氏

2020 March Vol.19

ソフトウェアと品質を考える

ベリサーブアカデミックイニシアティブ2019特集

2019年に開催された
「ベリサーブ アカデミック イニシアティブ」では、
アジャイル開発やデジタルトランスフォーメーションの推進といった
トピックをはじめ、ベリサーブの技術講演もお送りいたしましたので、
今号でご紹介します。

Contents

アジャイル開発とテスターの役割

～ビジネスとエンジニアの協働チームづくり～

平鍋 健児氏

ひらなべ けんじ

株式会社永和システムマネジメント
代表取締役社長

株式会社永和システムマネジメント代表取締役社長、株式会社チェンジビジョンCTO、Scrum Inc. Japan取締役。
福井でソフトウェア受託開発を続けながら、アジャイル開発を推進し、UMLエディタastah*(旧JUDE)を開発。現在、国内外でアジャイル開発の普及に努める。ソフトウェアづくりの現場をより協調的に、創造的に、そしてなにより、楽しく変えたいと考えている。
著書『アジャイル開発とスクラム～顧客・技術・経営をつなぐ協調的ソフトウェア開発マネジメント』、翻訳『リーン開発の本質』など多数。

はじめに

海外でソフトウェア開発手法としてメインストリームになってきたアジャイル開発が、日本でも活発になってきました。現代にマッチしたこの手法が現れてきたビジネス的背景を説明し、アジャイル開発の要点とアジャイル開発の現場についてお話します。最後に、アジャイル開発における品質保証についても取り上げます。

1. なぜアジャイルか？

◆アジャイル開発が現れた背景

従来の日本の受託開発から見ていきます(図1)。

まず、「ビジネス」(事業・サービスを行う部門)が、「市場」を分析し、企画を作ります。企画を基に、開発したい要求をリストにします。それを「IT」(開発側)に発注します。要求リストを受け取った「IT」は、見積もりをして、納期までに開発、テストをし、納品します。「ビジネス」が、できあがった製品・サービスを使い

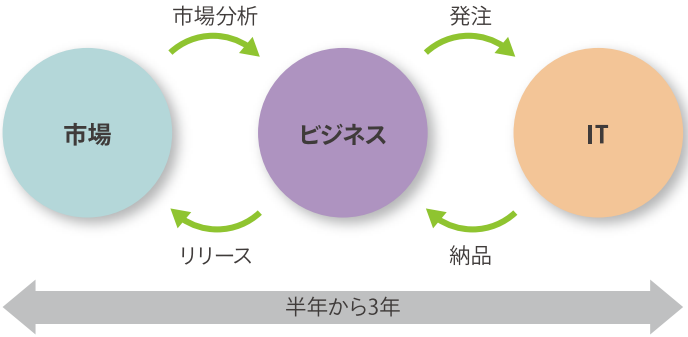


図1(出典:アジャイル開発とスクラム)

「市場」でサービスを展開します。

ここには、2つの大きな問題があります。

一つ目は、「市場」「ビジネス」「IT」間を往復する時間です。市場分析をしてから、ユーザーが使うまでの時間が半年以上と非常に長いことです。サービスが「市場」に出るタイミングが遅くなることは、サービスにとって致命的です。「市場」に出た時には、市場分析時よりユーザーの受けが悪くなっている、もっと進んだサービスが世に出ているといったことが起こります。「市場」は、将棋のように一手指したら、相手が一手指す、といったルールでは動いていません。10人が一挙に、駒を突っ込むような戦いが行われています。「一年間がんばって、やっとできました」では、「市場」のユーザーを逃してしまいます。それが一つ目の問題です。

二つ目の問題は、「ビジネス」と「IT」に、発注、納品という契約の壁があることです。

「ビジネス」は、最初にできるだけたくさんの開発要求を詰め込もうとします。後から要求を追加すると、最初に開発工数として見込まれていないため、「IT」から「それは別途お見積もりですね」と言われることがあります。一方、「IT」は開発要求が動かないように固定します。要求が変わると、後になればなるほど、開発上の制約が増え、納期までに要求を組み込むことが難しくなるためです。「IT」が、最終ユーザーや「市場」のことを考えずに要求(仕様書)に向かって、仕事していることも問題です。結果、「ビジネス」と「IT」が敵対関係になってしまうこともあります。

仕様書通りにシステムを作ることが成功ではありません。「IT」は「ビジネス」の一部であり、「ビジネス」と一体です。アジャイル開発では、「ビジネス」の成功が「IT」の成功であると定義して、その外側に「市場」があるという構造を考えます(図2)。

正しい要求は「市場」が持っています。サービスが使いやすい、使ってみて嬉しい、サービスに価値があるといったことは「市場」が決めます。「ビジネス」がいくらサービスのことを考えても正解がわからないということが起きます。

Webサービスを考えるとよくわかると思います。サービスを開発して、「市場」にリリースして、人気があれば、その機能を充実させます。人気が無ければ、その機能の開発をやめてしまいます。動くものを作って、「市場」に見ていただき、そのフィードバックを得て、次の作戦を考え、開発を行うというループを短期間に回します。

◆アジャイル開発の普及

2011年、起業家のエリック・リースが書いた「リーンスタートアップ」という本が出版されました。

本書に書かれている方法論に、サービス開発のループの話が出ています。最初にアイデアがあって、それを素早くサービスのプログラム(コード)にします。サービスができたら、すぐに使っていただき、データを取ります。取ったデータを分析して、

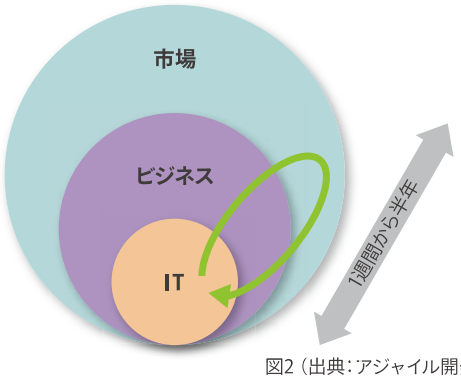


図2(出典:アジャイル開発とスクラム)

次のアイデアを決め、開発を行うというループです。この本が爆発的に売れたことで、「ビジネス」と「IT」という関係を一つのループの中に入れる考えが普及しました。これをきっかけに、価値の高いものから開発し、順次、リリースをするというアジャイル開発が一気に主流の開発手法になりました。2011年のアメリカの話です。

日本ではどうでしょうか？　現在、日本のアジャイル開発のブームは、第三の波が来ています（図3）。

最初の波は、アジャイル開発の本がたくさん出版され、エンジニアが盛り上がった2000年当初です。この時のブームは、ほとんどエンジニアの世界の出来事で、マネジメント層には受け入れられなかった時代です。例えば、アジャイル開発で実践されているテスト駆動開発について、マネジメント層からは、「テストコードを書くことで、工数が2倍になるじゃないか」と言われることもありました。実際には、すでにリリースした要求に影響が出ないように開発するには、開発中のシステムに影響が出ないことを確認しながら進めるためのテストコードが欠かせないものになります。

日本でアジャイル開発がはやり出したのは、Webのサービス企業からです。楽天、リクルート、ヤフー、クックパッドといった企業が社内にエンジニアを雇用し始めました。2010年ごろ、第二の波です。自社の外に開発を発注するのではなく内部のエンジニアと一緒にシステムを作っていきました。会社として最も重要なWebシステムですから、他人任せにせず、自社で開発を行います。そうすることで、発注、納品といった契約の壁も無くなります。日本にアジャイル開発が増えていきました。

そして、第三の波は、2016年、2017年、最近の話です。

ITの開発リソースを持っていないユーザー企業、例えば、デパートや製薬企業も積極的にアジャイル開発を取り入れ始めています。デジタルの力を使った新しいサービスを作りたいけれど、これまでの外部にシステム開発を発注するやり方では、市場に出すスピードが遅くなります。少し作って、すぐに使ってみるということが必要です。小さな開発チームにアイデアを与えて、うまくビジネスが作れるか、という実験が行われています。

例えばデパートの例です。モバイルアプリの会員となっていただいた方にどういう通知を送って、店舗に来て

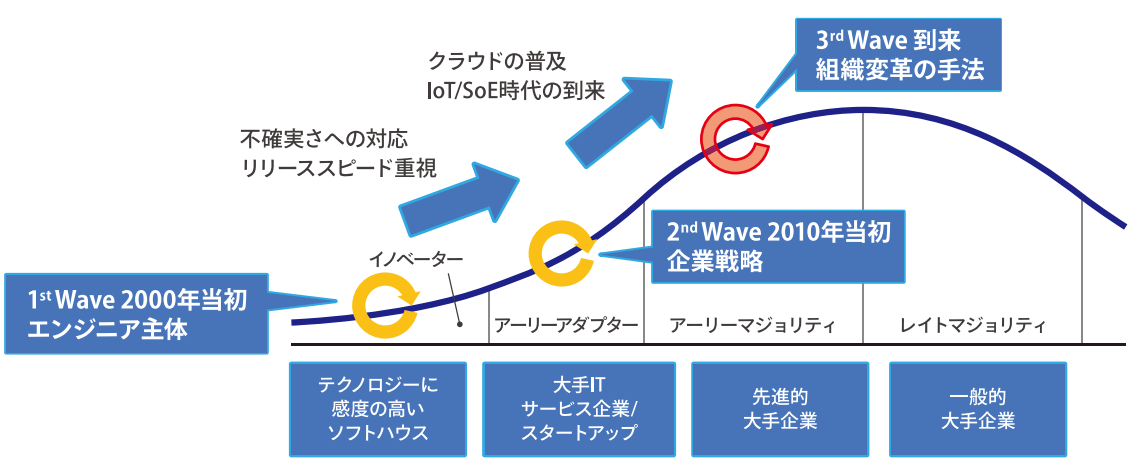


図3：アジャイル開発の普及の波

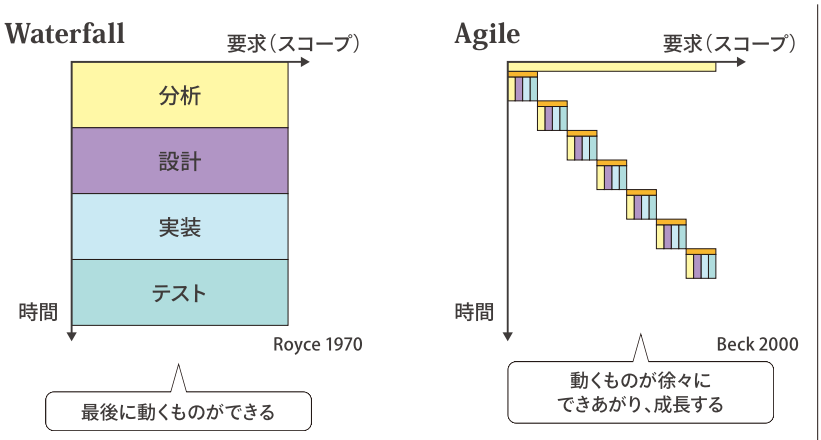


図4：プロセスとしてのAgile（出典：アジャイル開発とスクラム）

いただくのか。来店いただいたら、画像認識を使って年齢層と性別を判別して、その人にどういう通知を送り、どのお店に誘導するか。こういうサービスを考えるとき、きっちりした要求仕様書を書いて、システム開発を外注して、一年かけて開発するでしょうか？ベータユーザに試してもらいながら要求を考え、使いやすいものに仕上げていくというやり方が必要です。

2. アジャイルとは何か？

◆従来の開発手法とアジャイル開発

それでは、要求を素早く反映させるアジャイル開発の進め方について見ていきます（図4）。

従来の開発では、開発期間の最後に動くサービスができます。

例えば、一年後までにリリースしたい要求が365個あったとします。

これまでの開発手法では、全体を設計して、開発、最後にテストして、サービスをリリースしていました。これでは一年後までサービスがうまく稼働するかどうかわかりません。従来の開発手法をケーキ作りに例えると、ホールケーキを作るイメージです（図5）。一層作って、クリームを塗って、また一層作って、‥最後に全体にクリームを塗って、苺を乗せます。

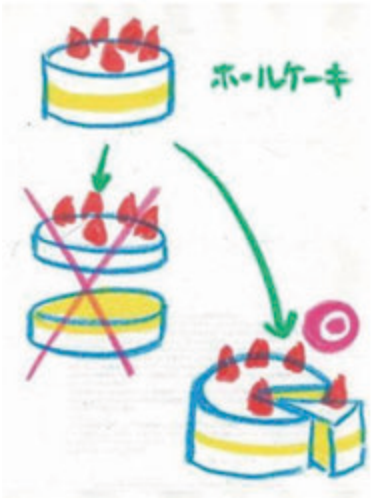


図5：ケーキ作りと開発手法の違い
（出典：アジャイル開発とスクラム）

アジャイル開発では、まず365個の優先順を決めます。そして、例えば一週間という期間で、優先順のトップ、7個なら7個の要求をまとめて開発をします。一週間で7個の要求を実際に使えるところまで開発して、ユーザーに見ていただきます。すぐに食べられるショートケーキを一つ一つ作るイメージです。

ユーザーからフィードバックをいただき、追加要求があれば全体の要求リストに加え、再度、要求を並べ替え、優先順の高い要求を次の一週間で開発します。従来の開発手法では、開発フェーズの後ろの方で「新たな要求を追加したい」という話が出た場合、開発側にとって悪いニュースでした。すでに全体の設計が終わっていますし、追加された要求がシステムにどう影響するかわからないためです。アジャイル開発では、いま一番フレッシュな情報に基づく要求なら、その要求を入れるべきだと考えます。しかし、これは簡単ではありません。

ショートケーキに例えると、一つのショートケーキの隣に、もう一つショートケーキを作って、2つがうまく合わせられるでしょうか。アジャイル開発では、ショートケーキがうまく合わせられるよう、まずは、

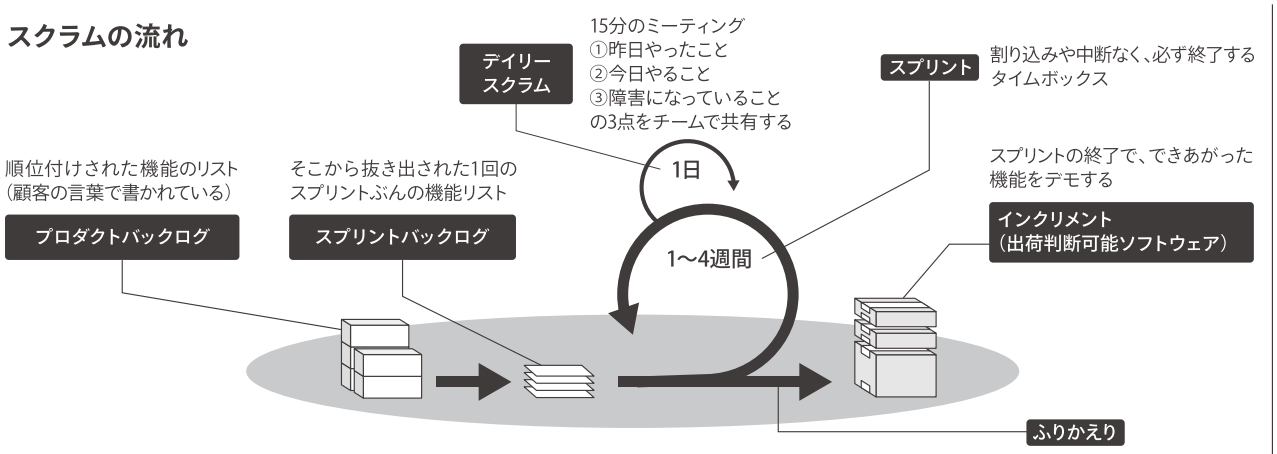


図6：スクラムの流れ（出典：アジャイル開発とスクラム）

ショートケーキの足場（テストコード）をケーキの周りに作ります。テストコードにより、動作が変わらないことを確認しつつ、共通の処理となる内部構造を変えながら、次の要求を開発します。

実際に、私が社内のプロジェクトを調べてみたところ、ショートケーキ（製品のコード）に対し、足場（テストコード）が1.5倍ありました。フレッシュな要求を受け入れながら、変化に強い構造にしていくには、それだけテストが重要になります。

◆アジャイル開発手法のスクラム

アジャイル開発には、さまざまな手法がありますが、これからアジャイル開発に取り組むには、軽量で最も普及している「スクラム」（世界のアジャイル開発の7割で取り入れられている）からやってみるのがいいでしょう（図6）。

例えば、サービスに対する365個の要求（プロダクトバックログ）があったとします。リリース日と次のリリース日までの間隔（スプリント）を決めます。仮にここでは一週間とします。[*1]スプリント内で開発可能で、重要と思う機能を取り出します（スプリントバックログ）。7個取り出したら、7個を一週間で作ります。一週間のスプリントが終わったら、動くソフトウェア（インクリメント）ができます。日々の開発に関わる作業、問題点は朝会（デイリースクラム）で共有し、一週間の終わりにふりかえりをします。うまくいったこと、できなかったこと、次にトライすることなどを洗い出して、開発プロセスを自分たちで改善しながら進めます。ただし、スクラムのガイドブックである「スクラムガイド(日本語版)」は、わずか18ページ、チームメンバーの役割、プログラムなどの作成物、会議体などのイベントが決められているだけです。これだけでうまく開発できるかというと、そんなことはありません。スクラムガイドには、テストのやり方も、レビューの仕方も書いてありません。他にもアジャイル開発手法は、たくさんあるので、その中で必要と思われるプラクティス（※）をスクラムに盛り込んで実践するとよいでしょう。

*1：一週間と決めたら、一週間、一週間、一週間、…と続けていきます。

※プラクティスの例

ソーシャルプラクティス：
タスクかんばん
バーンダウンチャート

技術プラクティス：
継続的インテグレーション
テスト駆動開発
リファクタリング
ペアプログラミング

3. アジャイル開発の現場

アジャイル開発のことが少し理解できたら、次にアジャイル開発の現場について見てみましょう。

私はプロジェクトが始まる時、「プロジェクトに一つ、野球のスコアボードを作ってください」という話をします。関係者全員が同じ情報（点数、審判、ストライク・ボール・アウトカウント）を共有するということです。

お客様、上司、品証部門、どのような立場の方でも、全員が同じ状況を把握できることが重要です。これまでの開発プロジェクトでは、「今回の報告はここまでに止めておこう」「ちょっと言いにくいから、こういう言い方にしよう」といったことをやりがちですが、これではいけません。必ず透明性を保ちます。

例えば、右上の写真は、タスクカンバンと言います（写真1）。

写真1：タスクカンバンの例 その1（出典：アジャイル開発とスクラム）

開発メンバー全員で、今週のやること（要求）を洗い出します。直近の要求を作業項目に分解して付箋に書き出します。作業項目が書かれた付箋を左端の「ToDo（やること）」に貼ります。朝会で、「やります」と言った人が作業項目の付箋を「Doing（仕掛かり中）」に動かします。作業が終わったら右端の「Done（完了）」に付箋を動かすということを一週間やります。左端にあったものが、一週間で全部右端に行けば、めでたく終了となります。非常にわかりやすいですね。

もう一つの例を見てみましょう。

写真2：タスクカンバンの例 その2
（出典：http://leansoftwareengineering.com/2007/10/27/kanban-bootstrap/）

こちらの例には、真ん中に並ぶ付箋に小さなオレンジ色の付箋が貼ってあります（写真2）。このチームには、在宅勤務の方がいて、在宅勤務の方には連絡が取りづらいそうです。オレンジ色の付箋には、社内にバディという作業を共有している方がいて、作業が止まらないようにその方の名前が書いてあるそうです。この付箋は、この現場の工夫です。

私は大きな会社に行った時、「アジャイル開発の全社標準を作りたいので、協力してください」と相談を受けることがありますが、「そういうことはやめてください」と言うようにしています。例にあるオレンジ色の付箋は全社標準からは出てきません。実際に手を動かして困っている人が、何かを発見して、自分たちで自らの開発プロセスの中に組み入れていく。これがアジャイル開発にとって大事なことです。

どんどん工夫が起こっていることが、アジャイル開発がうまくいっているかどうかの指標になります。

4. 品質保証

最後に、アジャイル開発における品質保証の話をします。

従来の開発では、開発したサービス品質の担保はリリース直前に行われます。

アジャイル開発では、日々、要件や発生した問題の確認を当事者同士で行います。確認の過程を知らずに途中からプロジェクトに参加しても、何が起きているのかわかりません。日々の開発の中で、品質保証の話をしていく必要があります。品質保証の担当者は、サービス開発の最初からプロジェクトに入るべきです。アジャイル開発では、開発するシステムを要求変更に強くするため、テストコードの割合が大きくなります。品質保証の立場で、どういう観点でテストし、どういうテストの仕組みを構築していくか、テストしやすい構造をどう作るのか、最初から話し合っておかなければいけません。サービス品質向上の視点からのシナリオテストにどのようにテスト観点を入れるのか考えなければいけませんし、システム発注者側の受け入れテストだけでなく、開発のループにテストの仕組みを入れていく活動も必要です。アジャイル開発では、継続的にテストをし続けないと要求を受け入れにくくなり、素早くリリースできなくなります。

アーキテクチャ側のリスク、後から入ってきて困る要件は他にもあります。セキュリティの話を後にしても全体に対してのインパクトが大きすぎて取り扱えません。セキュリティが重要になるのであれば、そういう要求項目は、なるべく最初に話す必要があります。

そして、大事なことは、「私はQA（品質保証担当）／テスターだから関係ない」といった消極的な関わり方はではいけないということです。役割を超えて、関係者全員で品質に対して一緒に考える姿勢が重要です。バグの収束曲線を見て、リリース判定を行うのではなく、最初から品質保証の観点をどのように設計に入れ込むか考えることが、今のアジャイル開発の考え方です。

おわりに

アジャイル開発手法の一つ「スクラム」は、実は日本から1986年に出された論文[*2]が元になっています。そこには、いろんな知見を持った人たちと一緒になって開発を進めていきましょうということが書かれています。バトンリレーではなく、ラグビーのように全員で押し進めていくことから、開発手法を「スクラム」と名付けたそうです。

本稿をお読みの方の皆さまは、さまざまなお立場でお仕事をされていると思います。この講演内容が、関係者一丸となって、開発を進めるということを考えるきっかけになれば幸いです。

本稿の図は、こちらから出典しています。

講演内容を含むアジャイル開発とスクラムを、体系的に、かつ、多くの実践事例を交えて解説しています。

開発において「スクラム」という言葉を発明した経営学の野中郁次郎先生と講演者の共著によって、知識創造モデルの文脈でも考察を加え、組織論としてのスクラムにもアプローチしている書籍です。

*2:「The new new product development game」
https://www.agilepractice.eu/wp-content/uploads/2016/09/Product-Development-Scrum-1986.pdf

本記事の内容に関するご質問、お問い合わせは、ベリサーブ 広報・マーケティング部 Email:verinavi@veriserve.co.jp までお寄せください。

デジタルトランスフォーメーションの推進と政策展開

－2025年の崖を克服するための企業競争力の観点から－

田辺 雄史氏
たなべ たけふみ

経済産業省 商務情報政策局
情報産業課
ソフトウェア・情報サービス戦略室長

1997年早稲田大学大学院理工学研究科修了後、通商産業省（現経済産業省）に入省。2000年以降内閣官房、経産省、IPA等において、サイバーセキュリティ政策、IT政策に長年従事。2017年よりIPA産業サイバーセキュリティセンターの立上げ・運営を陣頭指揮。このほか、米国大学院への留学、JETROデュッセルドルフ、在オーストラリア日本大使館への赴任等、幅広い海外経験を経て、2019年より現職。米国公認会計士。

和泉 憲明氏
いずみ のりあき

経済産業省
商務情報政策局 情報産業課
ソフトウェア産業戦略企画官

静岡大学情報学部助手、産業技術総合研究所（産総研）サイバーアシスト研究センター研究員、産総研情報技術研究部門・上級主任研究員などを経て2017年8月より現職。博士（工学）（慶應義塾大学）。その他、これまで、東京大学大学院・非常勤講師、北陸先端科学技術大学院大学・非常勤講師、大阪府立大学・文書解析・知識科学研究所・研究員、先端IT活用推進コンソーシアム（AITC）顧問などを兼務。

はじめに

経済産業省が2018年9月7日に発表した「DXレポート ～ITシステム「2025年の崖」克服とDXの本格的な展開～」は、各方面で反響を呼んでいます。過去に関与した論文や解説、レポート等と比較しても、本書はその中でもずば抜けてメディアやネットニュースなどでの引用が多く、関心の高さがうかがえます。

1. デジタルトランスフォーメーションとは

デジタルトランスフォーメーション（DX）とは何かという点については、さまざまなメディアや有識者がそれぞれの考え方で説明していますが、どうも言葉が一人歩きしている傾向があり、具体の議論が希薄になっていると感じていました。そこで我々はDX推進政策の共通理解として、以下の様にDXを定義しました。

> 企業がビジネス環境の激しい変化に対応し、データとデジタル技術を活用して、顧客や社会のニーズを元に、製品やサービス、ビジネスモデルを変革するとともに、業務そのものや、組織、プロセス、企業文化・風土を変革し、競争上の優位を確立すること

▶ 欧米の事例

かつて北米のプラットフォーマー本社でヒアリングをした際、彼らが強調していたのは、何を作るか、どんなサービスを提供するかの判断基準は、すべて「顧客体験の最適化」にあるということでした。同社はさまざまな領域で事業を展開していますが、そのすべてを横断して顧客体験の最適化に挑戦し続けているというのです。彼らは「Innovation at the edge＝顧客接点におけるイノベーション」という言い方をしていましたが、顧客体験をより良くするためのイノベーションを重ねることが、現在の成功を導く要因であったと考えています。

顧客接点の最適化の好例が、ある北米の航空会社です。この会社のスマホアプリには、大変興味深い機能があります。搭乗前にゲート前で待っている時に、アプリ上にフードコートのメニューが表示されます。ここで例えばハンバーガーを注文しておくと、搭乗後に自分の座席まで熱々のハンバーガーが運ばれてきます。実際に片道2時間半搭乗したのですが、通常の機内サービスはお世辞にも品質が高いとは言えないものの、このサービスは座席クラスに関係なく誰でも受けられるのです。

このサービスにはもうひとつ、重要なメリットがあります。一般にはあまり知られていませんが、航空会社の生産性向上の一番の近道は、機体を軽くして燃料費を抑制することだそうです。このサービスなら事前に注文が入った食事のみ積み込めば済むので、機内に厨房設備が不要になり、その分機体を軽くできるわけです。

また、機内では最新の映画が見放題というサービスがあるのですが、同社の座席には他の航空会社のようなディスプレイがありません。このことは乗客には告知済みで、各々が私物の端末でWi-Fi経由で映像サービスを受ける仕組みになっています。これも顧客サービスの充実を図りながら、機体を軽くする方策のひとつです。

この2つのサービスでは、技術的に目新しい点は何もありません。しかし、ソフトウェアへの投資→ハードウェアのダウンサイジング→顧客体験の向上という、デジタル技術の活用における理想的なステップが具体化された非常によくできたモデルと言えます。私見ですが、こういった発想はソフトウェア技術者のものではないのではないかと感じています。技術の向上や高度化は大切ですが、それが顧客満足に結びついているかどうかの方が重要なのです。

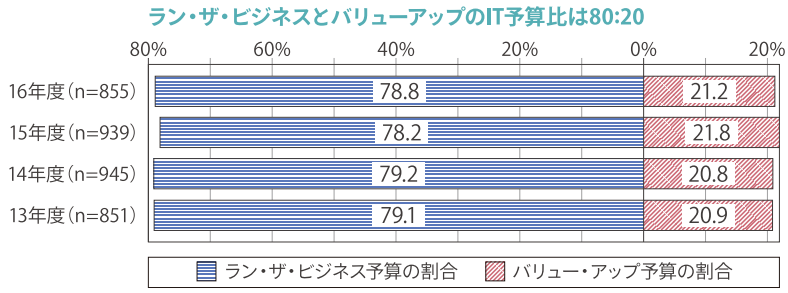
▶ 日本の事例

宮崎大学医学部附属病院では、汎用のAndroidスマホをベースとした電子カルテ端末を導入、看護師に常時携帯させています。その基本動作は、QRコードをかざすだけ。患者認証から始まり、点滴、注射、食事など、医師が指示するすべてのケアをQRコードの読み取りで実施する仕組みです。通常の病院業務では、患者に対して行われたケア情報をナースステーションでカルテに入力するのが定番の作業ですが、これがほぼ不要になりました。

この結果、同院で何が起こったか。看護師がスマホの充電時以外に、ナースステーションに戻る必要がなくなったのです。つまり、より長い時間患者のそばに寄り添い、そのケアに専念できるようになったわけです。さらに、業務負荷が軽減されたために残業も少なくなり、離職率が大幅に改善されました。顧客満足度の向上と働き方改革が同時に実現できたことになります。

DXがもたらすイノベーションは、大企業だけのものではありません。京都の宇治市にあるHILLTOP株式会社は、以前は典型的な町工場で、機械加工による自動車部品などの大量生産が主な事業でした。それが経営者の強い意志のもと最新のNC工作機を導入、3D-CADを使った多品種単品生産による試作品開発へと方向転換しました。業務の主体はプログラミングになり、製品の8割は生産数1～2個の小ロット試作品です。その品質が評判を呼び、海外からの発注も増加した結果、日本で組んだプログラムを元に現地で加工・納品を行うアメリカ支社をつくり、現在ではNASAからも発注が来るグローバル企業になったのです。

この事例であらためて感じたのは、DXの実現には経営トップのリーダーシップ、それも自らが積極的に変わろうとする意志が不可欠であるということです。技術はもちろん大事なのですが、それだけで成し得ることはもはや限られていて、より重要なのはそれを使って何をするのか、顧客にどんな価値をもたらすことができるのか、ということです。これを決められるのは経営者自身しかいないのです。



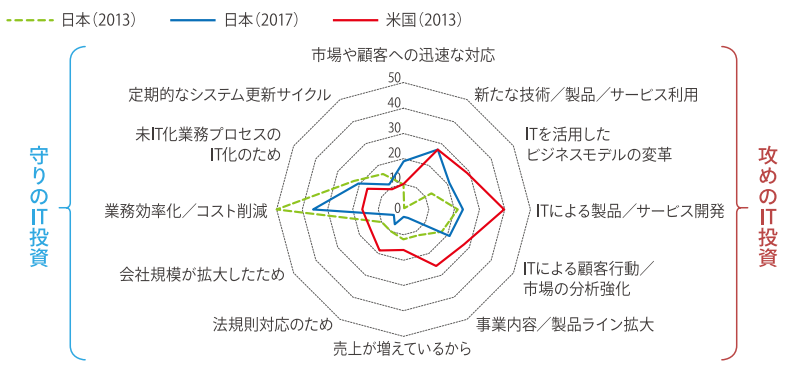
（出典）一般社団法人日本情報システム・ユーザー協会「企業IT動向調査報告書 2017」
図1：ラン・ザ・ビジネスとバリューアップのIT予算比

2. DXレポートと2025年の崖

ここまでDXを実現した企業の実例について説明してきましたが、一般的な日本企業の現状を実際のデータを元に見てみましょう。

現在、中国におけるIT分野の平均成長率は15%、米国では6%という統計が出ています。これに対し、日本の成長率は1%に過ぎません。この差はどこから来るのか。実は、日本企業のIT予算の総額自体は米中と比べても遜色はないのですが、問題はそれが何に使われているのかということです。

図1は過去4年間にわたる日本企業のIT予算の使い道を示したものですが、実にその80%が現行ビジネスの維持・運営（ラン・ザ・ビジネス）に使われており、バリューアップ予算、つまり戦略的なIT投資は20%に過ぎません。この20%にしても、もしも軽減税率や制度改正の対応に使われているとしたら、投資と呼べるようなことはほとんど行われていない可能性すらあります。



（出典）一般社団法人 電子情報技術産業協会「2017年国内企業の「IT経営」に関する調査」（2018年1月）より
図2：IT投資における日米比較

また、図2は日米のIT投資の詳細比較です。米国では製品やサービスの開発、顧客の行動分析など(攻め)に多くを投資しているのに対し、日本は業務効率化やコスト削減(守り)に予算を使っています。政府がいくらAI革命とかIoT革命とか叫んでも、民間企業は現行システムの維持・運営にしかお金を使っていないわけです。

▶ レガシーシステムと2025年の崖

維持・運営にこれほどの予算が必要となる原因は、システムのレガシー化です。レガシーシステムが恐ろしいのは、その事実が発覚しにくい点にあります。ユーザ企業は現行のビジネスを運営する上で不具合が起こらない限り、レガシーを自覚できません。また、ある企業のシステムが1つのベンダーによって構築・保守されているケースは少ないため、外部からも全体を俯瞰できずブラックボックスになってしまいます。

一方で、仮にIT部門がシステム刷新を提案しようとしても、誰も困っていないのに何のための予算なのだと経営陣から言われると、なかなかプロジェクトは進みません。競争の舞台がデジタル経済へ移る中、環境の変化や新たなデータの利活用に対応できるようシステムも改修しなければならないのに、問題が先送りされてしまいます。

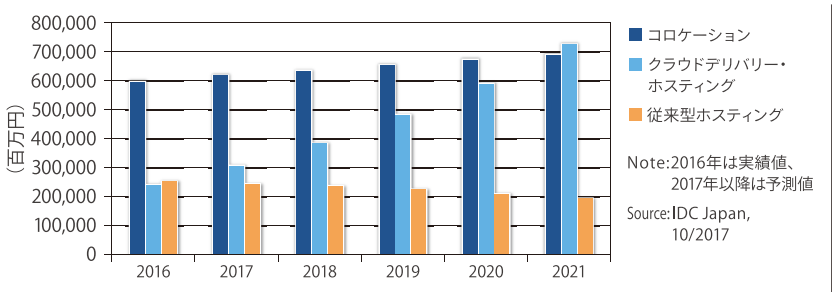


図3:国内におけるデータセンタービジネスの年間平均成長率

このまま放置すれば、2025年には人材不足やERPパッケージの保守終了などにより、保守費がさらに高騰するうえ、旧技術に精通した人材も枯渇してトラブル対策さえ困難になります。当然、ビジネス環境の変化にも対応できず、まさに崖から落ちるようにデジタル競争の敗者となる未来が待っている。これが2025年の崖で、最大で年間12兆円(現在の約3倍)の経済損失が生じる可能性があります。

これを克服するため、経営レベルでの危機感の醸成とDXの展開を促進する具体的な方策を提示するべく作成したのが、DXレポートというわけです。

3. クラウドの加速と技術動向

ここからは、いまグローバルでどの様な変化が起こっているのか、それによって技術者の成すべきことや評価基準がどう変わっていくのかについての情報提供をしたいと思います。

図3は、国内におけるデータセンタービジネスの年間平均成長率を示したもので、8.1%と好調な延びを見せています。一方、図4(次ページ)で示す通り、国内のサーバー出荷台数は7.8%減っています。ほぼ両者の相殺で数字は一致しているように見えますが、データセンターにはサーバーが納入されるわけですから本来ならこちらも8%程度増えるべきで、実際には国内出荷は16%減少していることになります。

このカラクリは実は簡単で、図3のグラフで大きく伸びているのはクラウド型ホスティング(■部)ですが、この大部分はAmazonやGoogleなどの海外メガクラウドベンダーが占めています。彼らはサーバーのパーツを独自の仕様のものを個別に調達・輸入し、現地で組み立てて自社のクラウドサービス用に設置していると考えられており、その分は出荷台数にカウントされていない可能性があります。しかも、国内の事業者とは比較にならないほどの大規模なデータセンターの建設ラッシュを、日本を

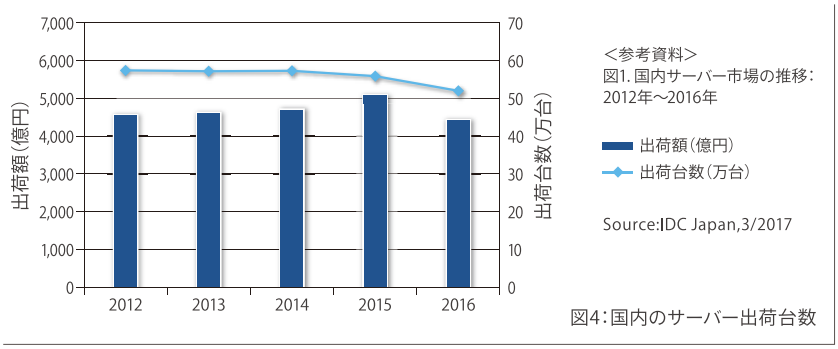


図4：国内のサーバー出荷台数

含む世界各地で続けています。

だから、データセンター市場が拡大しているにも関わらず、サーバーは売れないと考えられます。

こうしたクラウド化の加速が技術者にとって何を意味するか。新しいプロジェクトの立ち上げにあたって、今までのようにサーバールームを設置して、電源や空調を用意するなどという作業は消滅し、ボタン一つで契約完了します。そうなれば、誰もサーバーを買う必要はなく、そのメンテナンス要員という職もやがて無くなります。ビジネスの主戦場がクラウドに移行するなら、技術者もソフトウェアも、それに合わせてアップデートしていかなければならないのです。

4. DXの実現シナリオ

こうした海外の巨人たちを前に、我が国企業はどうするべきか。成長する企業とはどういうものかを明らかにし、そのためにどんな取り組みをすべきかを、DXレポートの中では実現シナリオとして提示しています※。

2020年まではオリンピックの恩恵もあり、好景気が続くでしょう。この収益が好調な時期を準備期間として、変革のための計画を立案します。その上で、その後5年間をかけてDX実現のための集中的なシステム刷新と経営改革を実行していきます。先に挙げたIT予算で言えば、ラン・ザ・ビジネスとバリューアップの比率を、現在の8:2から欧米並みの6:4くらいにまで変えていく。その他、人材の確保や育成などにも具体的な方策と目標を定め、実行していくことで、2025年には「あらゆるユーザ企業がデジタル企業に変革する」というDXのゴールへの到達を目指します。

▶ DX推進ガイドライン

DXレポートの発表後、企業にヒアリングをしてみると、DXの失敗事例にも一定の法則があることがわかりました。典型的なのが、経営側とIT部門の認識の不一致です。経営側はクラウドやAI、IoTなどの重要性だけは理解しているものの、それで何をするのかという明確なビジョンがない。こうした場合、IT部門が自らアクションを起こし改革を牽引して経営に貢献するというのが理想なのですが、現実的にはなかなか難しい。その結果、経営側はIT部門が改革に消極的であるとみなし、両者に反目が生まれることになってしまいます。これを打開すべく、2018年12月12日に公開したのが「DX推進ガイドライン」です（図5）。

ガイドラインは、「(1)DX推進のための経営のあり方、仕組み」と、「(2)DXを実現する上で基盤となるITシステムの構築」の2つから構成されています。前者では、経営側が明確なビジョンを持ち、それによってどんな価値を生み出すかを提示しているか、また、変革に対する強いコミットメントがあるか。後者は、全社的な体制・仕組みを整えているか、また、システム刷新そのものではなく、ビジネスへの貢献度で評価する仕組みができているかなどを主眼としています。

※：DX レポート（サマリー）- 経済産業省
https://www.meti.go.jp/press/2018/09/20180907010/20180907010-1.pdf

▶ DX推進指標

これをさらに推し進め、2019年7月31日に公表したのが「DX推進指標」です。先の事例にも見られる通り、DXは技術だけの話ではなく、それを使って顧客視点での新たな価値を創出していくことにあります。そのためにはシステムだけでなくビジネスモデルや企業文化などの変革も求められるわけで、これには経営者の意識改革が欠かせません。

しかし多くの企業経営者はどんな価値を創出するかではなく、「AIを使って何かできないか」といった曖昧な指示をIT部門に丸投げしている。また、社内で危機感が共有されていないため、経営としての仕組みの構築が伴わないといった事態になっています。「DX推進指標」では、簡単な質問による自己診断などを交えて、経営者・事業部門・DX推進部門・情報システム部門など、DXに関わる関係者がどう連携すればよいのかを明確にするための指針を提示しました。

「DX推進指標」の目的を簡単に言うと、IT部門からの意見具申が難しいのであれば、政府や役所が代わりにそれを言いましょう、ということです。皆さんは是非これを利用して、政府がこんなこと言っていますよ、どうしましょうかと、経営側にぶつけていただきたい。経営側がこれを見て、自分にはビジョンも危機感もないのかと自問自答すれば、皆さんとの間で会話が始まり、アクションを起こすきっかけが生まれるはずです。

「DX推進ガイドライン」の策定（2018年12月12日公表）

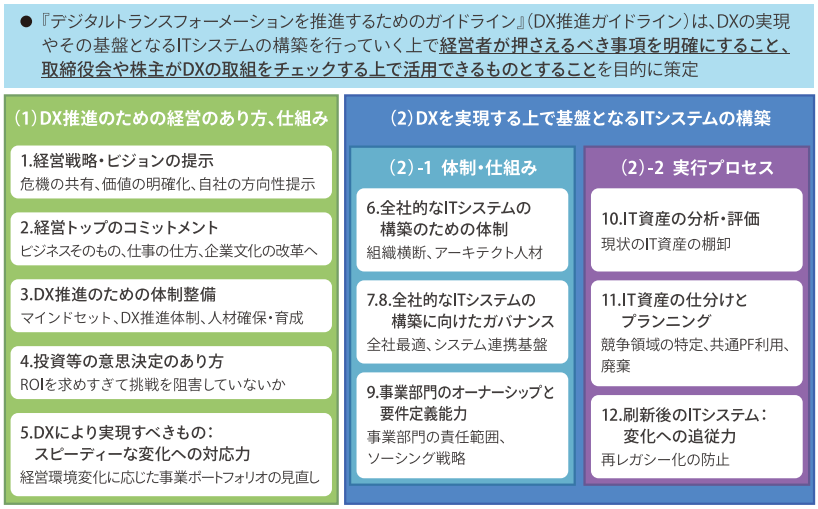


図5：DX推進ガイドライン

おわりに

デジタル時代における国際競争の「第1幕」は、主に検索・コミュニケーション・消費といった行為をサイバー空間で展開するための戦いで、GAFAに代表される海外企業が先行したのは事実です。

しかし、顧客接点での価値提供を突き詰めていけば、勝負の舞台はよりフィジカルな空間へと向かっていくでしょう。AIなどで分析したデータを、モビリティサービスや健康・医療・介護、農業といった現場でどのように活用し、顧客満足に結びつけるかが競争の「第2幕」です。カスタマーサービスを向上させていくという点では、日本人の素養である「おもてなし」の精神が強みを発揮するはずで、我が国企業が優位に立てる勝負所は数限りなくあるのです。

政府省庁というのは基本的には規制官庁ですが、我々経済産業省は唯一の「規制緩和官庁」として、市場に対してどんなことができるのかを常に模索しています。国際競争の「第2幕」に勝利するための基盤整備に向け、次なる政策展開を推進していきたいと考えています。

本記事の内容に関するご質問、お問い合わせは、ベリサーブ 広報・マーケティング部 Email:verinavi@veriserve.co.jp までお寄せください。

システムテスト自動化を開始する際に考慮すべき5W1H

奥村 哲郎

おくむら てつろう

株式会社ベリサーブ
製造システム事業部 第1ビジネスユニット

2012年、東京工業大学大学院修了後、独立系SIerにてQAエンジニアとして、テスト設計・実装やSEPGを担当。この間、テストチームの立ち上げ、テスト自動化にも携わる。
2017年ベリサーブに入社後は、車載器の開発管理を経て、現在は組込機器のテスト自動化に携わる傍ら、社外でのテスト自動化振興にも取り組んでいる。SeleniumConf Tokyo実行委員。

はじめに

「テスト自動化」という言葉が、ここ3~4年で業種や分野を超えてさまざまな開発現場で聞かれるようになりました。このような中、過大な期待や幻想を抱いたまま自動化に突き進んでしまうプロジェクトや、自動化という手段が目的化してしまっていることがあります。英文法における5W1Hの枠組みに沿って、システムテスト自動化を成功に近づけるために考慮すべきことを紹介します。

講演の背景・目的

テスト自動化に関わっていると、「テスト自動化」という言葉の意味が、広く使われ過ぎていることに気付きます。また、テストが勝手に行われるといった魔法の言葉のような印象を受けることもあります。そして、どうやって自動化すればよいのか、どういうツールを使うのか、といった手段が先に議論されることに危機感を覚えます。

こうした状況に対して、本稿は2つの目的を持って進めます。

- テスト自動化に計画が必要であることを理解する
- 計画を立てる上で必要な情報、考慮すべきことを把握する

そして、考慮すべきことを5W1Hという形で提案します。

Why: 目的
What/Where: テスト対象、テスト範囲
Who: 作成担当者、運営担当者

When: 時期、実施タイミング
How: 導入手法、アーキテクチャ、
開発プロセス、ツール

※本稿では、特に断りがない限り、「テスト自動化」をシステムテストの実行自動化として、話を進めます。

テスト自動化における 5W1H とは

5W1Hで考える内容について、一つずつ、確認していきましょう。

Why:目的

テストを自動化する目的は、何でしょうか?

それを考える前に、手動テストと自動テストが得意なこと、苦手なことを整理します。

手動テストと自動テストの特徴を整理したのが、下表です。

手動テストの一番大きな特徴は、テストケースに書いていない事象に気付くことができる点です。

例えば、組み込み系のシステム開発のテスト中、一瞬だけディスプレイに何かが映り込んだ、ということがあったとします。人間であれば、テストケースに無いことでも、おかしいと判断して、追加で確認することができます。これは自動テストに無い手動テストの強みです。

一方、自動テストにどういう強みがあるかを示したのが表の右側になります。

大きな特徴は、繰り返し作業、連続作業が得意であることです。

例えば、

・48時間連続で実行し続ける
・200回連続で繰り返し実行する

といったことです。

人間が間違えやすいことも自動テストに向いています。正確に100回ボタンを押してくださいといった場合、人間だとカウントを間違えることもあります。しかし、自動化することで、正確にカウントして実行することが簡単に実現できます。

ここまでが手動テスト、自動テストの特徴です。

手動テスト	自動テスト
テストケースに書いていなくても気付きがある ・なんとなく遅い ・色がおかしい ・一瞬何か映らなかった?	**繰り返し、正確な作業が得意** ・安定化試験、再現試験 ・MAX 試験、設定初期値確認試験
テストする人のスキルに依存する ・ドメイン知識 ・過去に検出した不具合	**人が間違えやすい操作手順の難しいテストが得意** ・割り込み試験、シミュレータ試験
臨機応変に対応できる ・既知不具合を回避する ・途中からやり直す	**休みなしにテストできる**
	書いてあることしかテストできない

表:手動テストと自動テストの特徴[*1]

*1: 自動テスト再入門(JaSST '19 Tokyo)ベリサーブ 藤田真広　http://www.jasst.jp/symposium/jasst19tokyo/pdf/E6.pdf
初めての自動テスト: Webシステムのための自動テスト基礎　Jonathan Rasmusson (著), 玉川 紘子 (翻訳),オライリー・ジャパン, 2017

実際にテスト自動化を行う例を示したのがこちらです。

- アプリの起動・終了処理をデプロイ毎に確認すること
- ユーザがよく使用する手順の操作を確認すること
- お金周りに関するテスト
- システムの安定性を見るためのテスト（連続稼働テスト、同時アクセステスト、負荷テスト）
- 組合せが爆発的に多いテスト（金融、自動車）

自動化するテストには、大きく2つの種類があります。

一つ目は高頻度で確認したいことです。例えば、お金周りは頻繁に確認することになるから、ここだけは自動化してほしい。そういったニーズは多くあります。

二つ目は、自動化したほうが楽になることです。

例えば、連続で動かす必要がある、同時にアクセスしないといけない、といった安定性などの非機能要件に関わることが挙げられます。また、テストケースとなる条件の組み合わせが爆発的に多く、手動でやるにはどうしても人手が足りない、といったテストは自動化されることが多いと思います。

ここまでが自動化の特徴と、なぜテストを自動化するかというお話でした。

What/Where:テスト対象、テスト範囲

テスト対象の話です。

必ず、「やりたいこと」「できればやりたいこと」「やらないこと」を決めましょう。システムテストを全て自動化しようとすることは、私のこれまでの経験上、うまくいったプロジェクトはありません。システムテストの自動化は、テスト対象を広げようとすると、あるところから急激にコストが増加します。例えば、電源をON/OFFするなど、物理的な操作が発生するといった人間が操作しないとできないことを無理やり、実現しようと考えた場合です。

盲目的に全てを自動化し始めるとうまくいきません。

「やりたいこと」「できればやりたいこと」「やらないこと」を仮にでも良いので決めてから取り組むようにしましょう。

Who:作成担当者、運営担当者

Whoで考えることは3点あります。

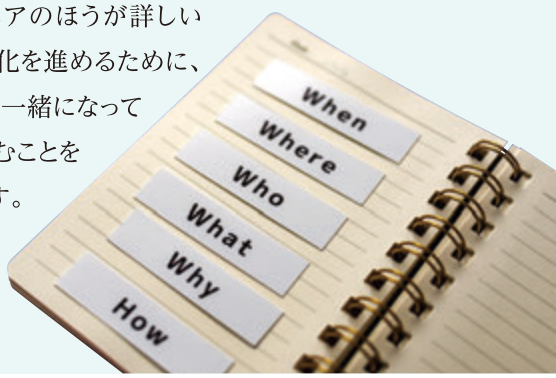
- 誰が作成・構築するのか
- 誰が運用・保守するのか
- 誰と協力する必要があるのか

（1）誰が作成・構築するのか:自社でやるか、ベンダーに任せるか?

サービスを持つ企業が、自社でノウハウをためながらテストの専門家がいるベンダーなどと一緒に自動化を進めるとよいでしょう。筆者の経験上、サービスを持つ企業のエンジニアが自動テストでできること、自動化が難しい理由をしっかり把握しておかないと自動化を進める効率が悪いと感じます。しかし、自動化の取り組み経験がないまま自力でやろうとすると手探りになってしまい、これも効率がよくありません。日常の業務に追われ自動化に注力できないこともあるでしょう。

テスト対象、例えば、皆さんが関わっているシステムを考えてみましょう。

同じ組み込み系のテストであっても顧客層、開発手法、使っているフレームワークの構成など、さまざまな理由で自動化が難しいポイントは異なります。こうしたポイントは、特にサービスを持つ企業のエンジニアのほうが詳しいので、うまく自動化を進めるために、テストの専門家と一緒になって自動化に取り組むことをお勧めしています。



(2)誰が運用・保守するのか

自動化したらそれで終わりということはありません。

例えば、仕様変更により画面遷移が変わった場合、自動テストもそれに合わせて修正する必要があります。また、テストを自動実行した後に失敗するテストケースがあった場合、不具合が起きているのか、テストコード側が誤っているのかを判断するのは人間です。

こうした自動テストの運用・保守は、誰が実施するかを決める必要があります。

(3)誰と協力する必要があるのか

三つ目は、誰と協力して進めるかという点です。

特に、CI/CD環境(Continuous Integration:継続的インテグレーション , Continuous Delivery:継続的デリバリー)を構築するインフラ担当との協力は必須です。

自動テストを作ったけれどもテストが実行されない、必要なタイミングで動かないということが起きます。夜中に実行するように準備していても、サーバーメンテナンス中で実行されないことはよくある話です。実行環境への自動テストの組み込みを依頼すること、実行環境が利用可能な時間を調整することが必要になります。

インフラ担当以外にも、自動テストを組み込みやすくするためのプログラム本体の実装の工夫、APIの開発など、開発チームとの協力も大事です。

When:時期、実施タイミング

テスト対象のシステムが、どのタイミングで自動テストが必要になるのか、という話をします。

顧客やシステム開発側の要望の一つとして、必ず実施タイミングを確認します。そのタイミングに対して計画を立てなければいけません。自動テストの場合、一度、テストを作って実行できるようになれば、その後、同じテストの実行には、ほとんど工数がかかりません。しかし、最初に、自動テストを作るところに時間、工数がかかります。また、手動でテストできないものは、自動化できません。自動テストを作り始める時期を決めるため、まず、手動でテストできるようになる時期を確認しましょう。

How:導入手法、アーキテクチャ、開発プロセス、ツール

まずは、どこまで現実的に自動化ができるのか検討します。

対象となるシステムによって異なるので、ただ一つの正解はありません。実際にテストケースを手動でテストしてみて、自動化できるところを検討していただくとよいでしょう。

自動化の検討が終われば、ツールの選定です。

操作を録画するように記録して、操作を再実行するキャプチャ&リプレイツールがよいのか、ソースコードのようなテスト用のコードを書いたほうがよいのか、というところでツールの種類は大きく2つに分かれます。

有償のツールもあれば、無償のツールもあります。大事なことは、やりたいことが実現できるかということです。試しにツールを使ってみるとよいでしょう。

もう一つ、How muchということにも少し触れておきます。

テストのコストダウンの話です。開発を繰り返す中で、確かに2回目以降のリリースでは、既存部分のテストの工数は少なくなります。実際に削減したコストに関するレポートもありますので、ご参考ください。[*2]しかし、本来求められているのは、テストのコストダウンではなくて、開発全体のコストダウンではないでしょうか。自動化したことで、これまで手動テストをしていた人が、他の活動をできるようになり、より開発に注力できるのではないかと思います。目先のコストダウンにとらわれ過ぎないことが大事です。

---

*2: 東芝レビュー 73巻3号(2018年5月)「ソフトウェアのテスト工数・期間を削減するためのシステムテスト自動化技術」
https://www.toshiba.co.jp/tech/review/2018/03/73_03pdf/a10.pdf

ケースで考えてみましょう

私が実際に携わっているシステムは、お客様の情報であるためお話することはできません。書籍のケーススタディを参考に5W1Hをどのように考えていくか見ていきましょう。参考にした書籍[*3]から、保険の販売員が使う保険契約の申し込みシステムを取り上げます。

<書籍からの引用>

システム:保険系(販売員が使う保険契約の申込みシステム)
自動化したい内容:手動の回帰テストを自動化

<書籍情報だと検討に不十分のため、追加した情報>

システムテストケース:900件と仮定
開発期間:1年間
テスト対象:Webシステム(タブレット上でブラウザ操作)
自動テスト開始時期:商用リリース後(毎月不具合修正、機能追加が入る)
組織:テスト自動化を行うのは社内では初めて。開発は大部分を委託

< Why >

なぜ自動化するのか、というところから考えます。

サンプルケースに記述がありますが、回帰テストのコスト削減を目的とします。さらに、これまで手動テストを行っていたメンバーを機能追加、不具合修正にアサインするという目的にします。

< What/Where >

自動化するスコープを決めましょう。サンプルケースにはWebブラウザから実行すると記述されています。ブラウザ操作の回帰テストを自動化することにします。Webブラウザのテストでは、どのブラウザで対応するのか、どのバージョンをやりたいのかという話が出ます。ここでは、Chrome、Internet Explorerのリリース時の最新バージョンのみと決めます。

< Who >

自動テストを誰が運用・保守していくか考えましょう。

自動テストを作成する人が必要です。<Why>で決めた手動テストを行っていた既存のテストメンバーに加え、自動のテストを作る人を新たにアサインします。自動テストの保守は、自動テスト作成メンバーの一部が同時に担います。

今まで実施していた手動テストは、既存のテストメンバーに任せたいですが、既存のテストメンバーは、すでに自動のテスト作成をやっているので、必要があれば、臨時で手動テストを行っていただく人を確保する必要があります。ベンダーに要員追加を依頼することも考えます。

< When >

いつ自動化するのかといった話です。

すでにシステムができあがっているので、すぐに自動化に着手できます。

いつ自動テストを実行するかという話ですが、毎月、不具合修正、機能追加のリリースがあるので、そのタイミングを目指します。ただ最初に自動テストを実装するには時間がかかります。例えば、最初の3ヶ月で自動化を予定している全体の10%を自動化することを目標にします。まずは目標を決めることが大事です。決めてしまえば、それに対して効率よくできたのか、問題があって進められなかったのかということがわかるようになります。まず、目標値を決めます。その後、実際に自動テストを運用して、運用の結果を見て、計画を見直していただきたいと思います。

*3:『Experiences of Test Automation』Dorothy Graham(著),Mark Fewster(著), Addison-Wesley Professional, 2012

＜ How ＞

ツールとしては、キャプチャ&リプレイツールを採用します。

開発は多くの部分を委託しているという記述があります。スクリプトを作るというのは難しいと想像します。今回は、キャプチャ&リプレイツールを採用します。

以上が、サンプルケースを基に大まかに考えた5W1Hでした。

今後のテスト自動化

最近では、AIを使ったテスト自動化ツールが出てきています。

いろいろ種類はありますが、システムの変更を検知し、テストを自動修復（セルフヒーリング）する機能を持ったツールが増えてきている印象です。こうしたツールが徐々に使えるようになることで自動テストのメンテナンス工数が減っていくだろうと、私たちも、現在、見守っているところです。

ただ実際に自動修復された内容、具体的には、あるボタンの変更があったので、別のボタンを押すように変更した、という修復がされた時、それが正しいかどうか、人間が判断する部分は残ります。今後、そうした判断も、AIに任せられるかもしれませんが、まだ人間の判断が必要な状況は続くと思っています。

まとめ

お伝えしたいメッセージは3つあります。

一つ目は、「テスト自動化をする目的を明確にする」ということです。明確な目的がないままトップダウンからの指示だけで自動化を進めて失敗した例をたくさん見てきました。失敗しないために、まず目的を明確にしてください。

二つ目は、「費用対効果を高めるためにPDCAを細かく回す」ということです。やってみて、何かがわかって、次のアクションを決めて、進めていくことが大事です。システムとの親和性、想定コストとの差異を観察し、範囲やスケジュールを変化させていくことが必要です。

三つ目は、「組織、システム、開発状況によって、最適な自動化の進め方は異なる」ということです。他社、他のプロジェクトの前例やアンチパターンに振り回されず、自分たちのノウハウをためていくことが大事です。

サービスを持つ企業のエンジニアの方と、我々のような自動化のプロが手を携えて、試行錯誤して進めていければと思います。

本記事の内容に関するご質問、お問い合わせは、**ベリサーブ 広報・マーケティング部** Email:verinavi@veriserve.co.jp までお寄せください。

テキストマイニングによるテスト分析、リスクベースドテストへの応用

堀川 透陽

ほりかわ とうよう

株式会社ベリサーブ
西日本事業部 第3ビジネスユニット

2007年よりソフトウェアテスト業務に携わり、2017年、ベリサーブに入社。
組込み開発におけるテスト業務を中心に、プロジェクトリーダーやテストチームの立ち上げを担当。近年は検証コンサルタントとして、テストプロセスの改善を行う傍ら、TPI NEXTやODC分析などの研究会にも参加。VPI技術者認定。JaSST Kansai実行委員。

はじめに

テキストマイニングは、文章を品詞レベルで分解し、情報・知識を得る技術です。

これを統計解析の手法と合わせることで、さらにデータの傾向や偏りをつかむことができます。この技術をテスト活動の上流工程で行うテスト分析やリスクベースドテストにどのように応用していくか、実践事例を交えながら解説していきます。

前半では、テキストマイニングの説明、テキストマイニングの活用例、テスト活動におけるテキストマイニングの親和性について、後半は、実際にプロジェクトの中でテキストマイニングを使ったテスト分析例、リスクベースドテストへの応用について紹介します。

1. テキストマイニング

○ テキストマイニングとは

テキストマイニングは、データマイニング手法の一つです。

データマイニングは、データからさまざまな統計解析手法やパターン認識、AIを用いた情報の取り出しなどを行う技術の総称です。テキストマイニングは、その中で、テキストデータを分析対象として、文章を品詞レベルで分解し、出現頻度や言葉のつながりから情報を取り出す技術です。

テキストマイニングのさまざまな解析手法について見ていきましょう。

まず、「形態素解析」と「構文解析」について説明します。

「形態素解析」は、文章を品詞レベルで分解します。分解したものを次に「構文解析」して文章の構造から意味のつながりを解析します。

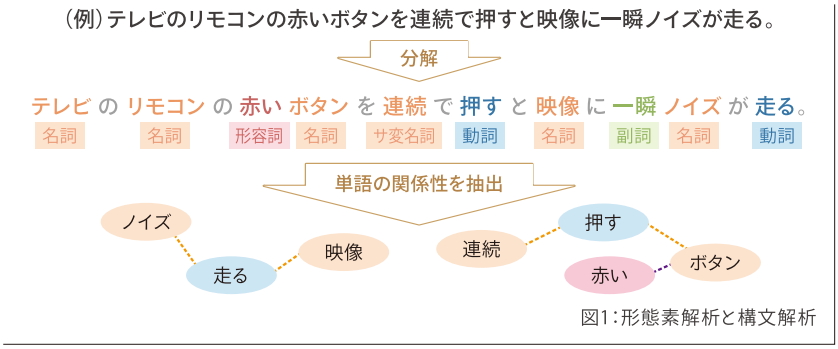


図1：形態素解析と構文解析

例文を見てみましょう。

「テレビのリモコンの赤いボタンを連続で押すと映像に一瞬ノイズが走る。」

こちらを「形態素解析」で分解します。

「テレビ」「リモコン」「ボタン」は名詞です。他に「押す」「走る」といった動詞があります。副詞、形容詞といった品詞も分解されて出てきます。「構文解析」により、意味のつながりを可視化します（図1）。

テスト活動で分析を行うと、名詞は機能名であることが多く、動詞はその機能に対して、どうアクションするか、名詞、動詞を修飾する形容詞、副詞というのは、アクションの条件として現れます。テキストマイニングツールを使い、単語の関係性を抽出してこれをテストに活用します。

他の解析手法についても、見てみましょう。

テキストマイニングでよく見かけるデータ解析手法として、さらに3つ紹介します（後半で、事例として取り上げます）。

「共起ネットワーク」は、語同士のつながりの多さ、つながりの深さをネットワーク図にして可視化します。

「対応分析」は、相関関係を把握する散布図を用いて可視化して、データの特徴をつかみます。

「クラスタ分析」は、似た性質の単語をグルーピングします。

テキストマイニングの解析手法は、他にもたくさんあります。

簡単に可視化できるため、どれを使うか迷いますが、大事なことはテキストマイニングを使って何を得たいのか、解析の目的に合わせて手法を選択することです。とりあえず可視化する、ということでもよいですが、すぐに有益な情報が得られるわけではありません。あらかじめ仮説を立てて、目的に合ったデータ解析手法を用いるほうが、結果としては早く自分の仮説を立証できます。

○ テキストマイニングの活用例

「形態素解析」は、大手ショッピングサイトでも見ることができます。

商品のカスタマーレビューを見ると、関連トピックのレビューを読む、という表示があって、関連語が並んでいます。単純な名詞だけでなく、動詞が混じっていることもあります。“テレビ”のレビューの場合、「初期不良」「電源を入れ」「リモコンの反応」などが表示されます。表示された語をクリックすると、その内容に関連したレビューが表示される仕組みです。

その他にも、テキストマイニングは、いろいろなジャンルで利用されています。

金融分野では、マーケット情報をテキストマイニングで解析して為替相場を予測するというサービスがあります。

特許調査では、他社の技術・出願動向を効率よく把握するツールとして使われています。非常に多くの件数を確認する必要があるので、効率的に早く特徴を把握したいということでしょう。

コールセンターでは、お客様への対応記録から顧客のニーズを引き出し、お客様対応の改善のために利用されています。

○テキストマイニングを行う理由

なぜテスト活動（特にテスト設計）で、テキストマイニングを利用したほうがよいのでしょう。

一つは、分析の効率化です。先ほどの大手ショッピングサイトのカスタマーレビューの例では、100件、200件程度ですが、数年分、数千件以上のデータもすぐに一次解析できます。

もう一つは、客観的、俯瞰的にデータを見ることができるためです。

特に、ベテランの技術者になればなるほど、自分のパターンに固執して、そこから離れた思考が持てなくなることがあります。解析手法を用いて可視化することで、自分が思いつかないパターンが見えることがあります。プラスアルファの思考力を手に入れることができると言ってもいいでしょう。積極的にテキストマイニングを使っていただきたい理由はここにあります。

○テストとテキストマイニングの親和性

テスト設計にテキストマイニング手法を用いることは本当に有効なのでしょうか?表1は、開発工程を含めて、テキストマイニングによる分析がどの工程に適しているのかを表にしたものです。各工程の成果となるドキュメントが、文章で記述されているものは、テキストマイニングの分析対象となります。

開発工程	ドキュメント	テキストデータの度合い
ビジネス要求／ユーザ要求	ビジネス要求定義書 ユーザ要求仕様書	○：文章で記述される
要件定義	要件定義書	○：文章で記述される
基本設計	業務フロー図、機能一覧、 ER図、CRUD図、画面レイアウト	×：文章から記号化・モデル化
詳細設計	詳細設計書 ※基本設計を実現するための記述	△：簡潔な文章
コーディング	コード	×：単語の集合、コメント程度

開発工程の上流のドキュメントが解析対象となり得る

表1：開発工程とテキストマイニングの親和性

開発工程が進んでいくと、成果となるドキュメントの内容が文章から少し遠ざかっていきます。「基本設計」では、業務フロー図、機能一覧、ER図などが出てきて、記号化、モデル化されていきます。「詳細設計」で基本設計をやや文章化しますが、簡潔な文章になっていきます。

「ビジネス／ユーザ要求」、「要件定義」の分析がテキストマイニングと親和性の高いところ、つまり解析対象と言えます。

次にテスト工程のテキストマイニングとの親和性を見てみましょう（表2）。

「テスト設計」工程の成果物は、テスト設計書、テストケースです。同工程の成果物を作るための分析なので、分析対象外とします。「テスト実行」では、成果物として不具合報告書が出てきます。文章で記述されるので、分析に適しています。不具合報告ではなく、ユーザビリティテストでも、いろんなご意見をいただくことができます。選択式のアンケート

開発工程	ドキュメント	テキストデータの度合い
テスト設計	テスト設計書 テストケース	△：文章でも記述されるが モデル化、自動化が進んでいる
テスト実行	不具合報告	○：文章で記述される
	ユーザビリティテストにおける レビュー記録	○：文章で記述される
その他	**ドキュメント**	**テキストデータの度合い**
リリース後	障害報告、クレーム報告 アンケート調査、口コミ・レビュー	◎：文章で記述される

**テスト工程では、実行による記録が分析・解析対象となる
また、リリース後のドキュメントはテキストデータが多い**

表2：テスト工程と以降のテキストマイニングの親和性

である場合もありますが、文章で記載されるところも多いので、分析可能としています。

そして、テスト工程が終わって、「リリース後」に出てくる障害報告、クレーム報告、顧客満足度調査、口コミレビューも文章で記述されています。ここが最もテキストマイニングが適している工程と考えます。一度リリースした後のユーザの声です。これらのドキュメントを解析対象とすることで、市場不具合の少ない品質を目指したテスト設計ができる可能性が高いと考えていますので、二重丸にしました。

実際、この後に紹介する実践事例でも障害報告やクレーム報告といったリリース後に得られたお客様の声を中心にテストにフィードバックしたアプローチをしています。

2. テスト分析の実践事例

○ テスト分析 ～リスクベースドテストへの応用

テスト分析の実践事例を見ていきましょう。

取り上げる事例の開発・テスト状況を説明します。開発モデルは、ウォーターフォール型、新規開発ではない派生開発でした。テストチームは、結合テストから参画しています。テスト要員、テスト時間は少ない状況でした。お客様の悩みは、市場リリース後の製品にトラブルを多く抱えていることでした。また、市場不具合の情報は持っていましたが、開発にフィードバックできていませんでした。

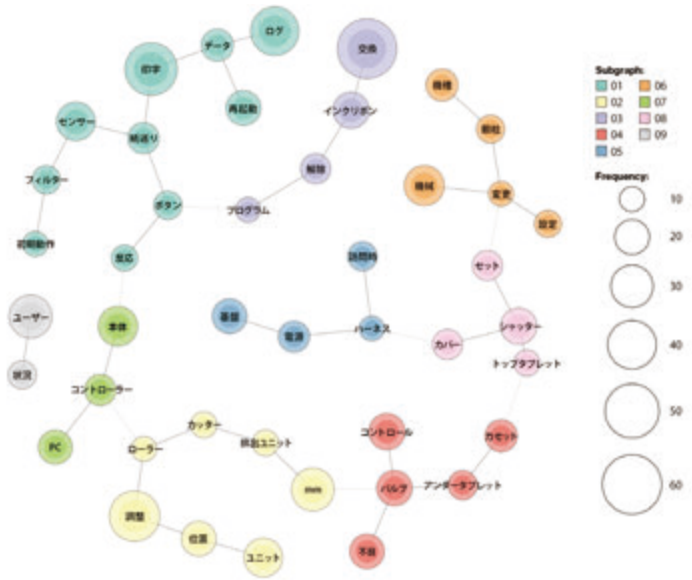
市場不具合を減らすために、市場リリース後のトラブルに着目して分析を行い、テストチームが担当する結合テストに、まずフィードバックしようと考えました。

市場不具合に関して、すでに管理されていた情報がありました。

「①コールセンターが受けたお客様からの問い合わせ内容」「②サポートエンジニアが現場で確認した事象」「③問題事象に対する開発側の考察」「④（考察を得て）お客様に行った対応内容」これらの情報が、すべてテキストデータという形で記録されていました。

テキストデータであるため、テキストマイニングによる解析が有効だと判断しました。さらにそのデータは数年分、開発にフィードバックされていないまま眠っていました。これだけの情報があれば、市場不具合をなくすことができると確信して、

テキストマイニングツール「KH Coder」を用いた共起ネットワーク図



・円の大きさ＝出現頻度
・線＝語の繋がり
・色＝語のグルーピング

数年分の不具合データから、問題事象が発生する語(機能・動作・条件)の組み合わせを、客観的に可視化できた

図2：共起ネットワーク

テキストマイニングを行いました。少ないテストリソースで、効率的なテストを行うために、市場不具合を解析して、問題の発生しやすい機能・条件を絞り込もうというテスト戦略です。市場不具合はどのような機能のつながりから出ているか、「共起ネットワーク」により可視化したものが前ページの図2です。

「共起ネットワーク」では、円の大きさは出現頻度・多さを示しています。

多く登場する語は、丸が大きいということになります。線は、語のつながりを示します。色が、語のグルーピングを示しています。色が異なり、線のつながっていないものは、何も関係がありません。配置にも意味はありません。

数年分の不具合データから問題事象が発生する語、機能・動作・条件の組み合わせをまず可視化しました。結合テストにフィードバックする分析については、特に「③問題事象に対する開発側の考察」をテキストマイニングしました。

開発工程に近い情報のほうが、テスト工程後半のシステムテストより、結合テストへのフィードバックがしやすいと考えたためです。「①コールセンターが受けたお客様からの問い合わせ内容」の分析結果は、システムテスト工程のシナリオテストに用いました。

グルーピングした情報をそのまま使うこともできますが、さらにリスクを特定したいと考え着目したものが、中心性と呼ばれる概念です。中心性は、ネットワークの濃度、重要さを測る指標です。中心性が高い語、色の濃い部分をリスクの高いものとして使っていきます。中心性には、いくつかの概念があります（図3）。

“次数”中心性は、隣接するノードの多さを示しています。人間関係なら、友達の多さを示したものです。真ん中の色の濃い“シャッター”という語、右下で色の濃い“排出ユニット”という語は次数中心性が高いと判断できます。

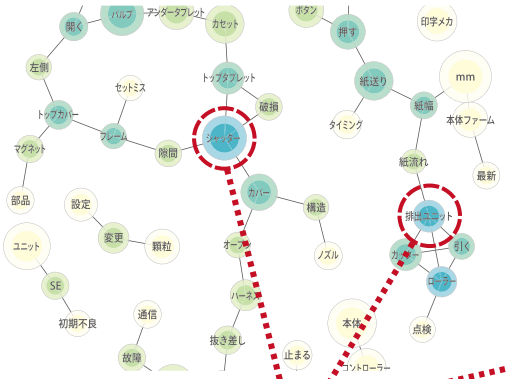
共起ネットワーク図からリスクとなる語を特定するために、中心性に着目する

※中心性：ネットワークのノード(語)やリンクから重要さを測る指標

共起ネットワーク図から中心性を可視化

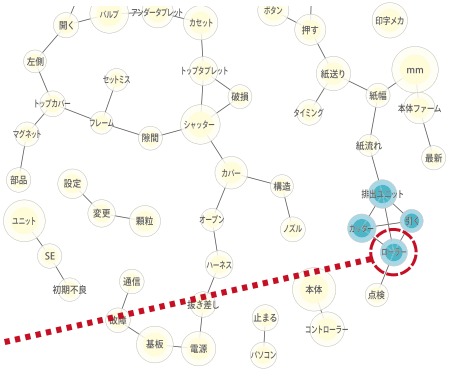
次数中心性

隣接するノードの多さ



固有ベクトル中心性

中心性の高いノードの連結度合い



中心性の高い語＝リスクの高い機能・動作・条件とし、テストにインプットする

図3：次数中心性と固有ベクトル中心性

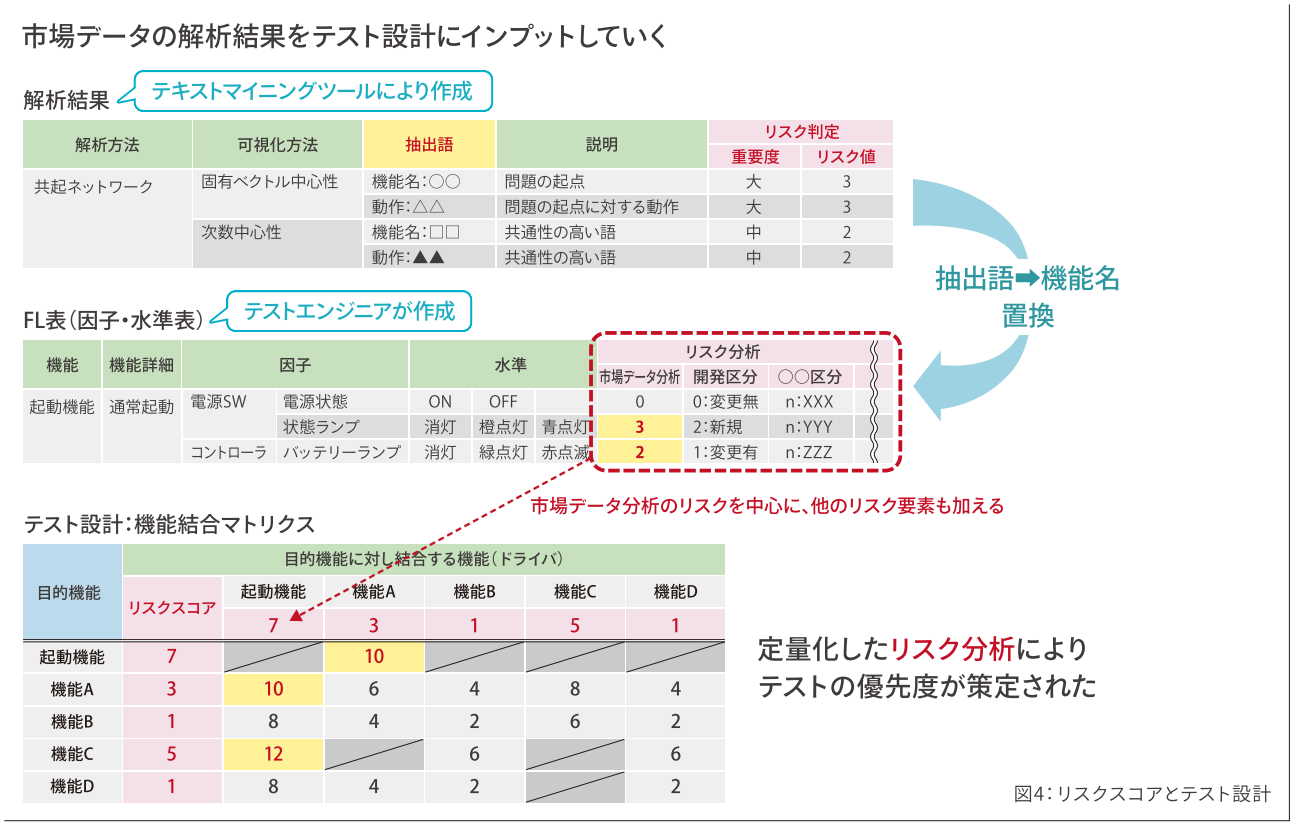


図4:リスクスコアとテスト設計

もう一つ、“固有ベクトル”中心性です。中心性が高い濃度の連結度合いを示しています。

人間関係に例えると、友達が多い人とたくさん知り合っている人を示しています。自分に友達が多くなくても、友人に多くの友達がいる場合、中心性が高くなります。次数“中心性”、固有ベクトル“中心性”という同じ中心性という言葉であっても、異なる語が強調されて表示されます。そのため、一つの“中心性”ではなく、複数の“中心性”からリスクを捉えていこうと考えました。今回は、2つの“中心性”の高い語をリスクの高い機能として、テスト設計にインプットをしようと決めました(図4)。先ほど抽出したリスクが高いと判断した言葉を抜き出します(図の解析結果)。

定量化するとわかりやすいので、リスクスコアとして、重要度をつけます。

解析結果の抽出語に関連する機能名を、テスト設計で用いるようなFL表(因子・水準表)と言われるテストエンジニアが作成する資料に反映させます。FL表からリスクスコアが出るので、こちらを基に機能結合マトリクスに展開します。

結合テスト以降を担当していて、目的機能に対する機能と機能間結合を見たかったので、リスクの高い機能同士を組み合わせて機能結合のリスクスコアを出しました。

この例では、機能結合マトリクスの黄色部分はスコアが高く、優先度が高いところです。

「起動機能」－「機能A」、「起動機能」－「機能C」が、テストの優先度が特に高い組み合わせとなります。このように優先度を明確にすることで、テストリソースをどこに当てはめるかを考えるときに、優先度が高いものからテストケースを選ぶことができます。

○テキストマイニングのさらなる応用

テキストマイニングのさらなる応用として、「対応分析」「クラスタ分析」について、紹介します。

「対応分析」は、全体の特徴をつかみます。実際に、厚生労働省のサイトの『毎月勤労統計調査オンラインシステム更改及び運用・保守に係る業務一式 要件定義書』を使って、対応分析を行いました(図5)。

要件定義書全体をテキストマイニングで解析して、語同士の距離感と語とカテゴリ(章)の距離感を座標にした図となります。赤四角は、カテゴリ(章)です。カテゴリに近い言葉ほど、特徴的な言葉を示しています。カテゴリに属さない原点に近い語は、普遍的な非特徴的な言葉です。

こちらの例だと、“毎勤システム”という言葉がありますが、毎勤システムの要件定義書なので、当たり前の語で特徴がありません。

「印刷」という言葉は、「3.4帳票に関する事項」のカテゴリの中で、特徴的な語であることがわかります。

次に「クラスタ分析」を紹介します。

「クラスタ分析」は、似た性質の単語をまとめます。仕様書を解析して、テストパターンを生成したいときに利用できます。こちらは仕様書全体から、出現パターンの似た語をグルーピングしています(図6)。

色がグルーピングを示しています。トーナメント表のような形をしていますが、ここから、赤のグループは何をしているのかを推測すると、赤は全国調査票と地方調査票の項目に関するものをまとめられていることがわかります。このクラスタ分析の結果を基にテストケースにない探索的テストアプローチができるのではないかと考えています。

なぜ、探索的テストアプローチができるのでしょうか。

通常、テストを設計するときには、機能単位で考えて、テスト項目を抽出しているでしょう。A機能であればA機能の設計書に書いてあることから、テスト設計をします。「クラスタ分析」によるグルーピングにより、全体からパターンを作ることで、機能単位では考えられない、プラスアルファの視点が得られることがあります。

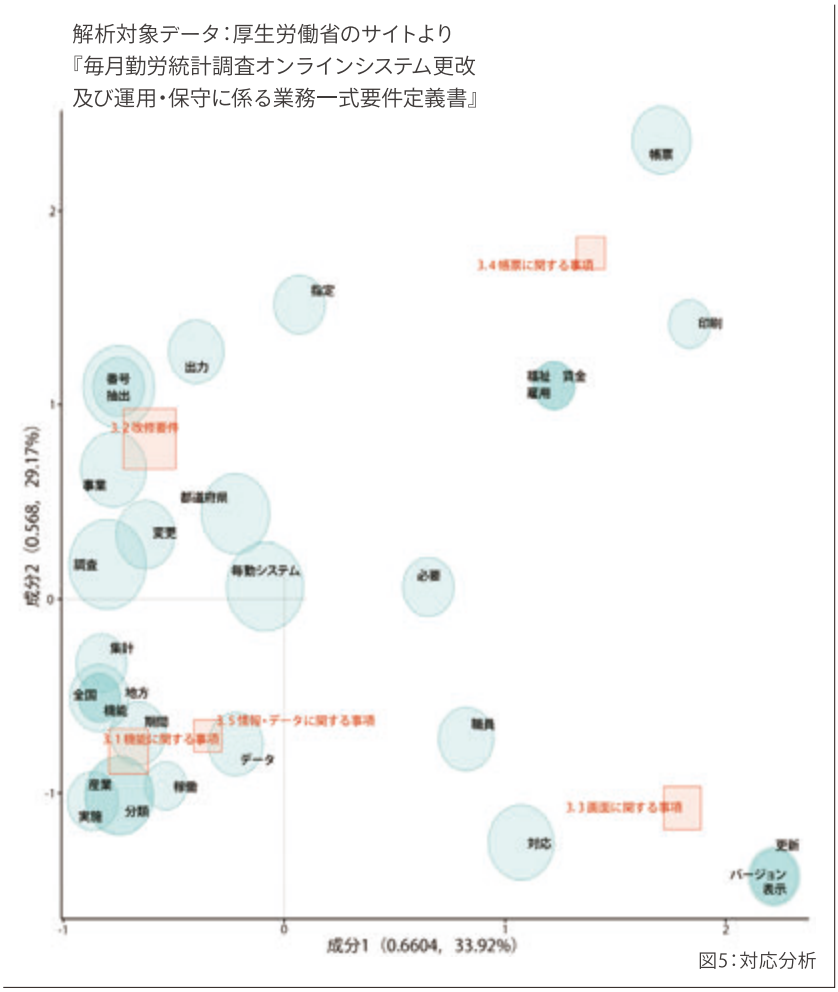
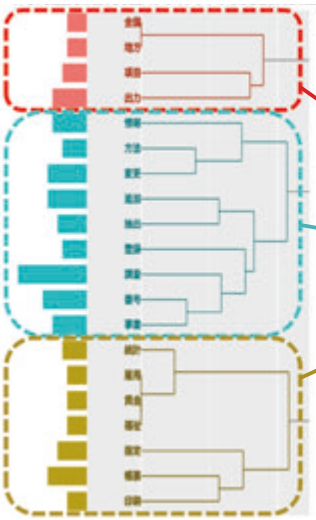


図5:対応分析

解析対象データ：厚生労働省のサイトより
『毎月勤労統計調査オンラインシステム更改及び運用・保守に係る業務一式要件定義書』



仕様書全体から、出現パターンの似た語がグルーピングされる

それぞれのクラスタのテーマは何か考察・推測する

- 全国調査票と地方調査票の項目出力
- 調査票及び事業所情報の登録・追加・変更
- 雇用・賃金福祉統計室の帳票印刷

従来の仕様書単体からの設計とは異なる探索的テストアプローチ

図6：クラスタ分析

おわりに

テキストマイニングは、開発の上流工程、要求、要件分析と市場リリース後のデータに対して解析の親和性が高く、テスト分析に活用できます。本稿では、実践事例として、「共起ネットワーク」を使用して市場不具合を導く語のつながりを可視化しました。特に、中心性に着目して、リスクの高い機能・動作・条件を導いて、リスクベースドテストにつなげる方法を紹介しました。さらに要求分析として、「対応分析」を用いたり、「クラスタ分析」により仕様書全体から探索的テストアプローチを考えたり、さらなる活用が期待できると考えています。

テキストマイニングは、データ分析の効率化、客観的な分析をして、これまでにない気付き、プラスアルファの思考力を得られるということをお伝えしてきました。本稿を読んで、ご興味が出てきた方は、ぜひ、テスト活動でテキストマイニングを活用してみてください。

本記事の内容に関するご質問、お問い合わせは、**ベリサーブ 広報・マーケティング部** Email:verinavi@veriserve.co.jp までお寄せください。

はじめに

ソフトウェア開発の大規模化と複雑さの増加は近年ますますそのペースを速めていく傾向にあります。開発と対をなすソフトウェアテストの現場においても、大規模化とテスト開発の複雑さは増していく一方にあります。さまざまなエンドユーザの利用状況に合わせて設計されたテストケースは、時に数万、数十万という規模に及び、これらのテストの開発、進捗状況を適切に取り扱うため、テストチームは日々スプレッドシートと格闘しています。「テスト資産やテスト結果を、何らかの形でデータベースを使って管理できれば…」そんな想いから開発が始まったベリサーブのクラウドサービス、QualityForwardが描く理想のテスト管理を、この場を借りてご紹介いたします。

日々のテストにチームワークを！
テスト管理クラウドサービス「QualityForward」のご紹介

テスト管理ツール とは

ソフトウェアテストの現場ではさまざまなツールが活用されますが、その中でもテスト管理ツールには主に以下のような役割が期待されます(JSTQB foundation level シラバスより)。

- テスト実行や欠陥の追跡
- 定量的分析やテスト対象のレポート
- テスト対象から要求仕様書への追跡

テスト管理者は、これらの役割から得られる追跡情報やレポートと開発プロジェクト側の状況を総合的に判断して、テストプロセスをハンドリングします。例えば、テスト実行状況に合わせてバグ曲線がある程度増加傾向にあればテストが着実な成果を上げていると見ますが、バグが出すぎている場合、バグの出現が偏っているモジュールのテストを一時中断して開発側と対策を検討しますし、バグが少なすぎた場合、テスト設計の品質に問題は無かったかを再検証する必要があります。また、テスト管理ツールに投入されたテスト資産(テストケース、テストデータ、テスト結果データなど)の定量データを分析することで、次にどのようなテストを実施すべきかを判断します。

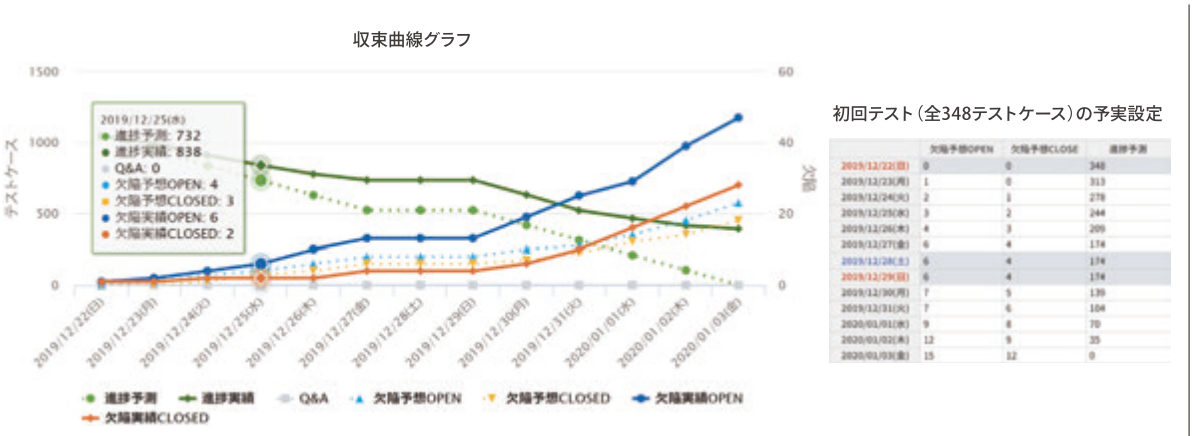


図1 テストケース消化、およびバグのオープンクローズの予実曲線

QualityForward では？

QualityForwardでは、通常のバーンダウンチャート（+バグ曲線）に加えて、個別のテストサイクルにおいてテスト管理者が知りたい①進捗、②テスト結果状況、③期日に対する遅延・前倒し の3つの観点をダッシュボードで一覧できる機能を備えています。極端な話、テスト実行フェーズにおけるこの3つの観点については、テスト管理者は朝出社してダッシュボードを開くだけで把握が可能になります（図1、図2）。

オンラインでテスト結果を入力し、自動的に集計される仕組みは、特に大規模、遠隔、並列のいずれかの要素を持つテストプロジェクトにおいてその威力を発揮します。数十人規模でテスト実行が行われるプロジェクトの場合、その数十人の日々のテスト結果を取りまとめ、状況を判断し、適切なコントロールを行うことが、テスト実行フェーズにおけるテストマネージャの第一の責務となります。しかし「数十人のテスト結果を取りまとめる」こと自体が、現場のテストマネージャにとって物理的、心理的に大きな負担になっているケースが多くみられます。QualityForwardをはじめとする、クラウドテスト管理ツールを利用したサービスを導入することで、テストマネージャはその双方の負担から解放され、本来行うべきコントロールに集中できることも、単純な工数

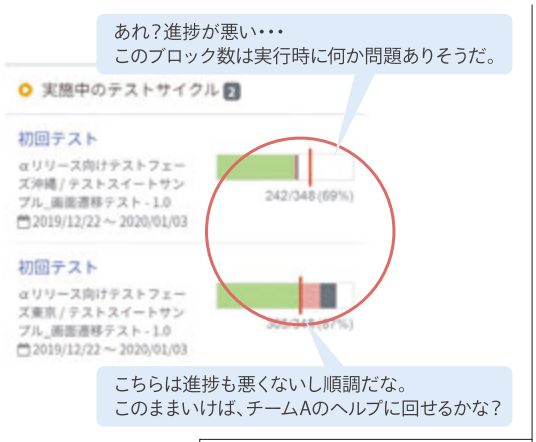


図2 テスト結果成分および進捗バー

削減に限定されない、サービスの大きなメリットになります。

日々のテストにチームワークを

前述の3つの観点（①進捗、②テスト結果状況、③期日に対する遅延・前倒し）についての進捗管理機能は、テスト結果を投入する画面にも全く同じものが表示されており、同じテストサイクルを実施している全員にリアルタイムで共有されます。これにより、テストマネージャの指摘を待たずに、テストチーム全員が、自分たちのテスト実行が遅れているのか？ 前倒しなのか？ を常に理解しながらテストを進めることが可能になります（次ページ図3）。

図3 テストの進捗状況をテスト実行画面の下部にも表示

テスト実行フェーズは、開発プロセスの中で最もボリュームの多い情報を扱うと言っても過言ではありません。多くの開発プロジェクトにおいて、実装された後の仕様や、ステークホルダーの判断がすべて明文化されているわけではないので、おのずとその情報はテスト結果を判定するためにヒアリングを重ねるテストリーダーやテストチームに蓄積することになります。

QualityForwardでは、テストフェーズにおいてテストの現場に溢れる暗黙の仕様をテストチーム自身が更新し、明文化された情報として扱うための「チームWiki」を搭載しています。

図4 QualityForward自身の明文化されていない仕様をテストチームがまとめている例

誰かの頭の中にある状態をできる限りなくし、仕様理解をチーム全体のナレッジとすることが、テスト設計やテスト実行フェーズの品質向上に寄与します。そして、そのナレッジベースはテストチーム自身が日々触れる場所にあるべきです(図4)。

テストケース消化数、それ以外の進捗尺度

テストの本当の意味での進捗は、テストケースの消化数だけではなく、設計されたテストの観点やテスト要求のカバー率など、多元的に確認する必要があります。QualityForwardでは、投入されたテスト資産のテスト観点ごとのシェアや、そのシェアごとのテストPASS率などから色分けされた「カバレッジパネル」をご提供しています。このテスト観点ごとの

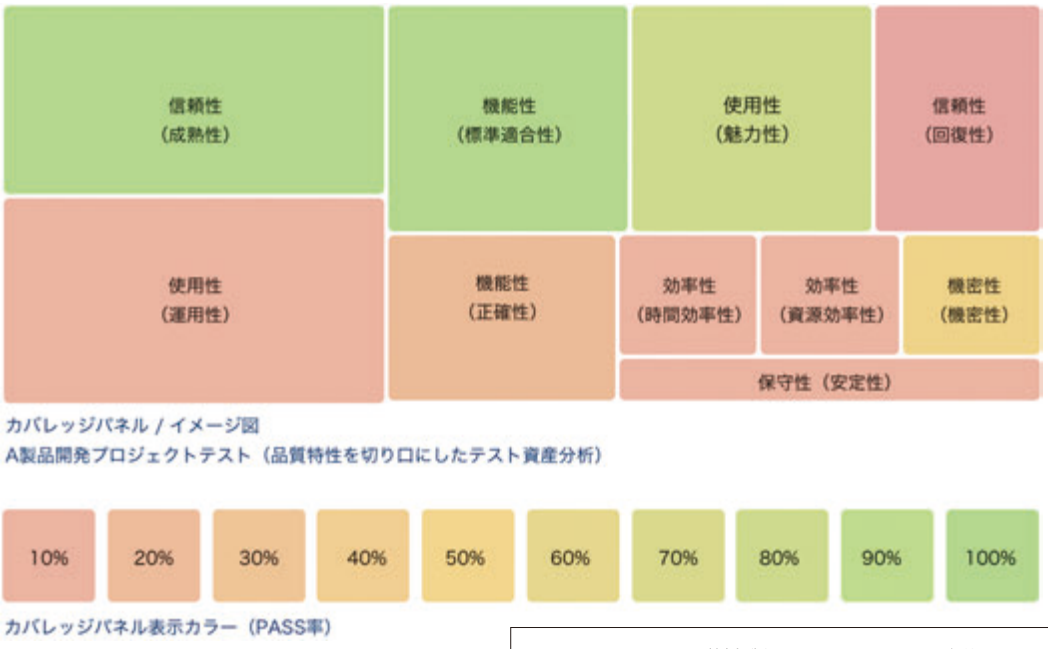


図5 カバレッジパネル（対象製品はQualityForward 自体のテスト初期）

シェアと、PASS率を見ることで、テストケース消化率に加えてさらに意味のあるテスト設計、およびテスト実行の進捗管理を可能にします。また、テスト管理者は事前に計画したテスト観点ごとのシェアと、実際に開発されたテストスイートの成分分析結果を比べることで、本当に実施したかったテストになっているのか、一目でわかるようになります。また、このカバレッジパネルは、テスト要件ごと、機能ごと、など、スプレッドシート形式の任意の軸を列として指定できます（図5）。

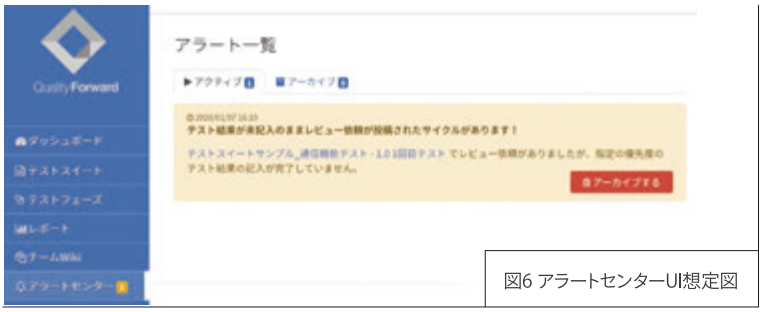


図6 アラートセンターUI想定図

これからのQualityForward

QualityForwardはおよそ2週間に一度のペースでミドルウェアの最適化、サービスの内部、外部の効率化、エンドユーザ様のご要望にお応えした機能のアップデートを行っています。今冬には、主に数十人、数百人規模のテストチームを擁する管理者が日々遭遇する実行管理上のリスクをQualityForwardが自動的に感知し、アラートする機能「アラートセンター」のリリースを予定しております（図6）。

クラウド技術と先進のテスト技術を融合し、テストチームを支援するQualityFowardを、ぜひ一度お試しいただければ幸いです。

お問い合わせ先 株式会社ベリサーブ
TEL：050-3640-8250（ソリューション推進部） 052-325-5010（中部支社） 06-6223-6110（西日本支社） URL：https://www.veriserve.co.jp/

品質を創造する企業

Veriserve Navigation 2020年3月号（ベリサーブナビゲーション）
編集・発行 | 株式会社ベリサーブ　東京都千代田区神田三崎町3-1-16 神保町北東急ビル9F
お問い合わせ | 広報・マーケティング部　TEL：050-3640-8194　MAIL：verinavi@veriserve.co.jp