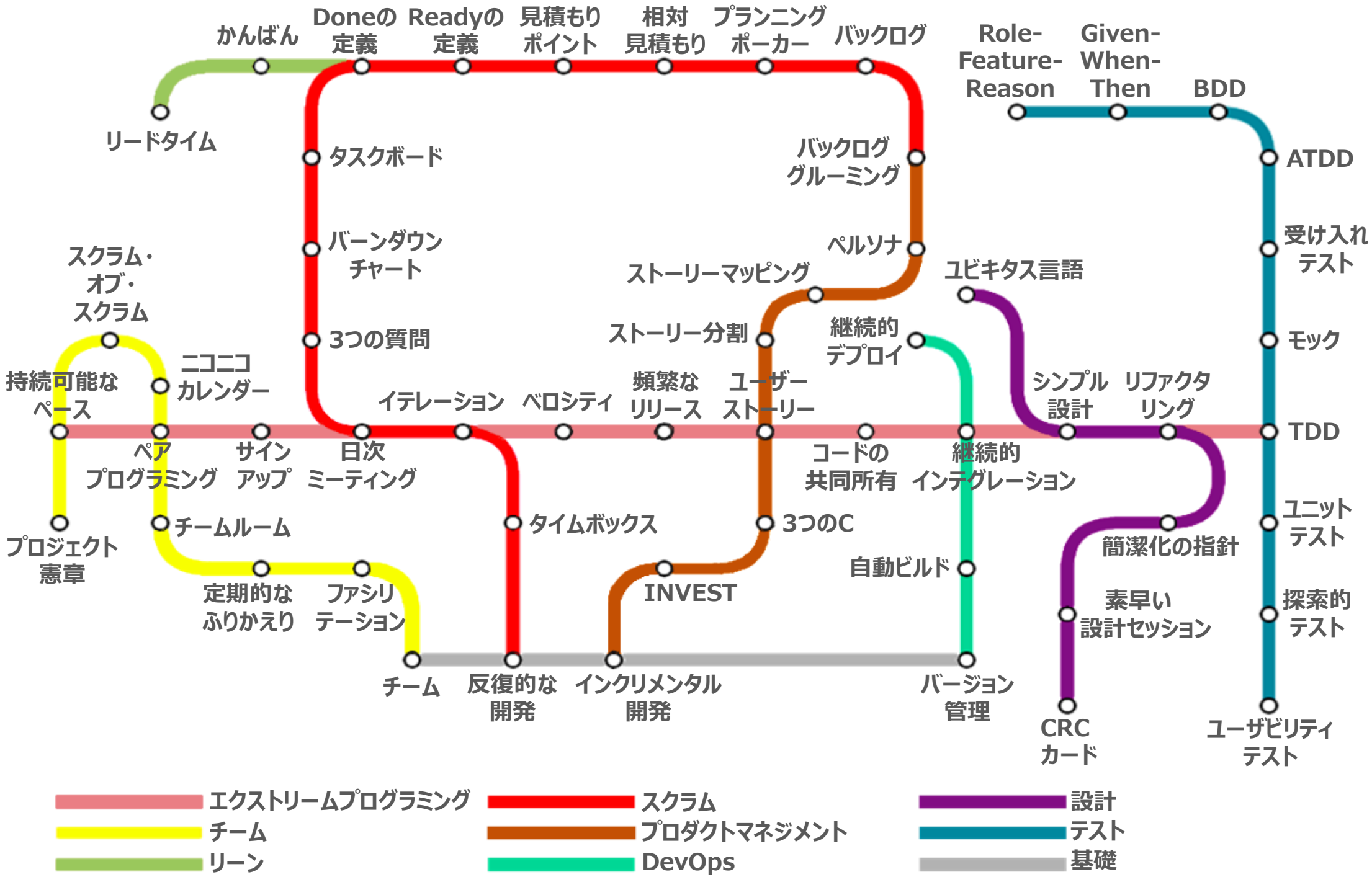
元資料：http://guide.agilealliance.org/subway.html



 翻訳：Agile Japan 2016実行委員会 2015/11/17版

チーム
アジャイルにおけるチームは、一プロジェクトにほぼ100%参加しているメンバーで構成されている。

反復的な開発
アジャイル開発は通常反復型であり、ソフトウェア開発の各工程を繰り返し実行する。

インクリメンタル開発
ユーザーから見て分かる単位で、製品に機能を順次(インクリメンタルに)追加していく。

バージョン管理
一般的なソフトウェア開発でも既に浸透している手法ではあるが、アジャイル開発では特にチームの指針を明確にすることが重要である。

持続可能なペース
チームはいつまでも続けることができる仕事のペースを目標とする。

ペアプログラミング
二人ペアになって、１つのマシン(画面、キーボード、マウス)でプログラミングする。

サインアップ
仕事やタスクをマネージャーが「アサイン」するのではなく開発者が「サインアップ」する。

日次ミーティング
毎日同じ時間にチーム全員でミーティングを行い、共同作業のために重要な情報を共有する。日本では「朝会」と呼ばれて親しまれている。

イテレーション
アジャイル開発の繰り返し単位となる期間。タイムボックス。スクラムではスプリントと呼ぶ。

ベロシティ
イテレーション内でチームが実装できたユーザーストーリーの見積もりポイント合計。

頻繁なリリース
最低でも数回のイテレーションごとにリリースし、ユーザーからのフィードバックを得る。

ユーザーストーリー
アジャイル開発における開発対象の機能単位。ユーザーから見てその価値が見えるものでなければならない。

コードの共同所有
チームメンバー全員が、すべてのコードに対して必要に応じて修正することを許可するだけでなく、むしろ積極的に推奨する。

継続的インテグレーション
リリース可能な製品を常に提供可能にするため、繰り返し頻繁に実行可能なインテグレーション手順を整備する。

シンプル設計
基本的な設計方針として「シンプル」を心がけ、設計する際にそのメリットとコストを検討する。

リファクタリング
既存のプログラムの外部から見た動作を維持しつつ、その内部構造を改善することである。

TDD（テスト駆動開発）
まずユニットテストを書き、パスするシンプルなコードを実装し、リファクタリングを行うことを繰り返す開発手法。

プロジェクト憲章
プロジェクトの目的・スコープ・概要を簡潔にまとめ、チームルームに貼り出しておく。

スクラム・オブ・スクラム
スクラムにおける、複数チーム間で連携するための日次ミーティングの手法。

ニコニコカレンダー
チームメンバーが帰り際に、部屋の壁に貼られたカレンダーにその日の気分を手書きの顔の絵や色付きのツールで表すことで、チームやメンバーの気分の変化のパターンが表れる。（日本発のプラクティス）

チームルーム
チームが同席するためのスペースであり、さまざまな貼り物によって情報が可視化、共有されている。

定期的なふりかえり
チームは通常、イテレーションごとにチーム活動を振り返り、その後の改善につなげる。

ファシリテーション
ファシリテーターは、ミーティングの司会役であり、効果的なグループディスカッションになるように場づくりをする。

リードタイム
ある要求をユーザーストーリーとして定義してから製品に組み込まれるまでの経過時間である。「かんばん」を導入しているチームは、ベロシティ向上よりも、リードタイム短縮を目指す。

かんばん
仕掛り制限を持つ「かんばんボード」を用いて、ベロシティの代わりにリードタイムなどを重視し、流れを妨げている作業をチーム全員で助ける。

Doneの定義
ユーザーストーリー(追加機能)には、完了時に満たされなければならない基準(何をもって完了とするか)が決まっていて、チームが合意している。

タイムボックス
作業時間を変更できない固定期間に区切り、その期間単位で仕事を行うこと。

3つの質問
日次ミーティング(朝会)では「昨日やったこと」「今日やること」「現在の課題」について述べること。

バーンダウンチャート
縦軸に残作業量、横軸に作業期間を取るグラフで、チームの進捗状況を可視化する。

タスクボード
ホワイトボードや壁を使用して、作業を書いた付箋などを現在の状況に合わせた（「仕掛中」のような）列に貼ることで進捗を可視化し、共有する。

Readyの定義
着手前に満たさなければならない、ユーザーストーリーの準備完了基準。(Doneの定義参照)

見積もりポイント
見積もりには日や時間を単位とした数字ではなく、ストーリーポイントといった「相対見積もり」による無名単位の数値を使用する。

相対見積もり
タスクやユーザーストーリーを、同じような難易度のものを基準に相対的に見積もること。(見積もりポイント参照)

プランニングポーカー
アジャイルチームで行われる、遊び要素を含んだ合議的な見積もり手法である。

バックログ
チームが実現させるべき機能や技術タスクが優先順に並んだリストで、いつも最新の状態に更新される。

バックロググルーミング
製品バックログを常に最新状態になるように手入れ（グルーミング）する。

INVEST
ユーザーストーリーのよしあしを評価する基準。Independent(独立) / Negotiable(交渉可能) / Valuable(ユーザー価値を持つ) / Estimable(見積可能) / Small(小さい) / Testable(テスト可能)

3つのC
ユーザーストーリーの3要素。カード(Card)、会話(Conversation)、確認(Confirmation)。

ストーリー分割
大きなユーザーストーリーを、一イテレーション内で実現できる程度に小さく、かつビジネス価値は保持するように分割すること。

ストーリーマッピング
ユーザーストーリーを、横軸にユーザーのふるまい順、縦軸にそのふるまいを実現するために詳細化した順に並べ、製品の全体像を可視化する。

ペルソナ
製品のユーザー像。歳や趣味まで具体的に想定し、写真やイラストなどを使って共感できる対象にする。

自動ビルド
自動化されたビルドプロセス。ビルドにはコンパイルだけでなく、自動テスト、パッケージングやデプロイが含まれる。

継続的デプロイ
繰り返し頻繁にインテグレーションし、デプロイまで行う。(継続的インテグレーション参照)

CRCカード
オブジェクト指向設計をする際、カードを使って開発者がオブジェクトを「演じる」ことで、そのクラスの責務や協調関係を見つける手法。

素早い設計セッション
複数人の開発者が集まって、ホワイトボードなどで素早く設計についてディスカッションする。

簡潔化の指針
ソースコードが「十分簡潔であること」を判定するための指針、優先順位である。

ユビキタス言語
要件・設計・コード、あらゆる部分でビジネス領域の語彙を使用する。

ユーザビリティテスト
ユーザーに実際にソフトウェアを使用してもらい、その行動を観察するテスト手法。

探索的テスト
テストの設計・実行・結果解析を決まった順序で行うのではなく、探索的に行う手法。テスターの創造性をより活かす。

ユニットテスト
ユニットテストをつかって製品のソースコードの小さな部分を動作させ、その実行結果を確認する。

モック
ユニットテストをする際に、テストに必要な偽の呼び出し先オブジェクト(モックと呼ばれる)を作成する手法。

受け入れテスト
ソフトウェア製品の振る舞いを記述するテスト。使用例や利用シナリオ。

ATDD（受け入れテスト駆動開発）
受け入れテストを作ってから、そのテストにパスする製品機能を実装する手法。

BDD（ふるまい駆動開発）
TDDとATDDを統合し改良したもので、ユーザーストーリーから抽出する「ふるまい」を記述して開発を進める。

Given-When-Then
ユーザーストーリーのための受け入れテストを書く際に利用するテンプレート：「Given(この状態で)、When(こう行動したら)、Then(この結果になる)」

Role-Feature-Reason
ユーザーストーリーを書く時のテンプレート：「Role(役割)として、Feature(機能)が欲しい、Reason(理由)だからだ」

 翻訳：Agile Japan 2016実行委員会　2015/11/17版