アジャイル時代のアーキテクチャ設計

野田　夏子†

現代は，環境や技術の変化が激しく，その中で迅速な開発が求められるアジャイル時代と言える．このような状況において，設計は依然として重要であるが，より迅速な開発のためには，様々な課題がある．本ワークショップにおいては，アジャイル時代にあってソフトウェアの骨格構造であるアーキテクチャの設計はどうあるべきかを考えたい．

Short Note on Architecture Design in Agile Age

Natsuko Noda†

This paper describes a position statement for the workshop (Winder Workshop 2016 in Zushi). The authors want to discuss issues of software architecture design in this “agile age.”

1. はじめに

変化の激しい現代社会に適合するため，ソフトウェアの開発の世界でアジャイル開発ということが言われ始めて久しい．現在，米国ではその適用が半数を超えているとも聞く．日本ではまだ十分な普及には至っていないものの様々な適用事例も報告されており，今後もさらに適用が増えることが予想される．まさに，アジャイル時代と言えよう．

アジャイル開発においては，ドキュメントよりも動くソフトウェアを，と言われることから，設計よりコーディングが大事と思われてしまったりする．しかし，アジャイルソフトウェア開発宣言[1]でも，ドキュメントよりも動くソフトウェアを，と謳っているのであり，決して設計よりソースコード(あるいはコーディング，実装)を，と言っているのではない．現に，アジャイルソフトウェアの 12 の原則[1]でも，「技術的卓越性と優れた設計に対する不断の注意が機敏さを高めます」と述べられており，設計を重視していることがわかる．

しかし，従来的な設計技術を従来通りに適用しているだけでは，やはりソフトウェア開発全体をアジャイルに行うことはできない．本稿では，近年注目されているソフトウェア開発技術とアジャイル開発との関係を考える．特に，ソフトウェアの骨格構造であるソフトウェアアーキテクチャ(以降，アーキテクチャと呼ぶ)の設計について，アジャイル時代にどうあるべきかを考えたい．

†芝浦工業大学
Shibaura Institute of Technology

2. ソフトウェア開発における変化

ソフトウェアにおいて，その要求は常に変化し得る．これはソフトウェアの本質であるとも言え，ソフトウェア開発の難しさの一因でもある．アジャイル開発は，こうした要求の変化に柔軟に対応しながら，迅速な開発を実現することを目指している．要求が変化すれば，当然その要求を実現するソフトウェア自体が変化しなければならない．設計技術とアジャイルの関わりを考える前に，このソフトウェアの変化について，考えておきたい．

ソフトウェアの変化は，いくつかの軸から整理することができる．ひとつは機能とデータである．要求が変われば，ソフトウェアの機能に変更が起こる．ある機能を修正したり，削除したり，別の機能を追加したりしなければならない．また，機能だけでなく，扱うデータが変わる場合もある．データ種類が変わる場合もあるし，ある種類のデータに関して取り扱う量が変わる場合もある．

別の軸として，時間と空間が挙げられる．ソフトウェアの変化は，当然時間の流れの中で起こる．時間はソフトウェアの変化の要因であるが，扱うデータ等の存続時間が変化するなど，ソフトウェアが扱う対象の変化としても働く．一方，空間の変化については，そのソフトウェアがどこで稼働することを期待されるのか，その稼働空間の変化もあるし，扱うデータの存在空間の変化もある．現代では，稼働環境も対象環境も広がる傾向にあり，スケーラビリティの問題として顕在化する場合もある．

アジャイル時代における設計技術について考える際には，要求の変化とソフトウェア自体の変化について，その性質を整理することが必要と考える．

3. アジャイルと設計技術

本章では，モデル駆動開発とソフトウェアプロダクトライン開発(以降，プロダクトライン開発と呼ぶ)を取り上げ，アジャイルの視点から考える．

3.1. モデル駆動開発

モデル駆動開発は，モデルを変換することによりソフトウェアを開発するアプローチである．典型的な適用例としては，設計モデルを変換することによりコードを自動生成することが挙げられる．

従来，モデルは，開発プロセス中のある行程において，特に設計においてその結果を記述するドキュメンテーションのために用いられてきた．UML を使って設計したモデルを設計文書に記述するといった使い方である．この使い方においては，モデルはドキュメンテーションであり，動くソフトウェアではない．しかし，モデル駆動開発においては，変換によりソフトウェアを開発するため，モデルを作成することは単なるドキュメンテーションではなく，ソースコード作成に匹敵する作業となる．要求の変化に対しても，モデルを変更することにより最終的なソフトウェアへと変化を反映させることができる．モデル駆動開発の出現によって，モデルをアジャイル開発に組み込むことが可能になったとも言えよう．

その際に課題となるのが，モデル変換である．要求の変化が大き過ぎて，ソフトウェアの根本の構造も変化するような場合，既存のモデル変換では対応できないことも起こり得る．設計したモデルをモデル変換で変換するという枠組みで考えるのではなく，モデル変換自体も設計対象として考える等が必要になるかもしれない．

3.2. プロダクトライン開発

プロダクトライン開発は，体系的な再利用の形態であり，特定のマーケットやミッションのためにコア資産と呼ばれる共通の再利用資産を利用して，個々のソフトウェアプロダクトを開発する．

プロダクトライン開発は，将来開発されるだろうものも含めた複数のソフトウェアプロダクトを対象にしてコア資産を開発する．将来の変化に備える開発とも言える．短いサイクルを繰り返すことで変化の都度に対応するアジャイル開発に対し，プロダクトライン開発ではある程度変化を見越して備えることで，それぞれ異なる部分を含んだプロダクトをひとつひとつ開発するより短い期間で開発する．プロダクトライン開発は「重い」開発の典型であり，「軽い」開発であるアジャイル開発の対極にあるようにも思えるが，異なる方法で変化への対応という同じ目的にアプローチしているとも考えられる．また，プロダクトライン開発にしろアジャイル開発にしろ，変化が大き過ぎてソフトウェア構造が大きく変化する場合には，変化への追随が難しくなるという課題も共有する．

4. アジャイルとアーキテクチャ設計

前章で，ソフトウェア構造が大きく変化する場合の課題を述べた．つまり，要求の変化によっては，アーキテクチャの変更が求められ，それは従来対応が困難であると考えられてきた．アーキテクチャは「固い」構造であり，その設計は「重い」プロセスと捉えられるからである．

アジャイルとアーキテクチャも，アジャイルは軽く，柔軟であり，アーキテクチャは重く，固いものであり，対立的なものと捉えられてきた．しかし，近年，アジャイル開発においても，ソフトウェアの骨格構造を適切に設計することは重要であり，アジャイルの迅速な開発においてどのようにしてアーキテクチャ設計を行うべきかが研究されるようになってきた[1]．これらの研究は，どちらかというと開発プロセスの側面から，アーキテクチャの設計や管理とアジャイル開発の整合をどのように取っていくかに焦点を当てたものが多い．

一方，著者らは，アジャイル時代にあって，より柔軟なアーキテクチャはどうあるべきか，その構造を検討している．筆者らは，従来固いものとされてきたアーキテクチャを柔軟なものとするために，骨格構造とアプリケーション構造の依存関係を，アスペクト指向技術を用いて逆転することを考えている[3]．まだ基本的なアイディアのレベルであるが，2章で述べたようなソフトウェアの変化について，その性質を分析しながら，どのような変化については依存関係の逆転が効果的であるのか検討する等し，より柔軟なアーキテクチャの設計を目指している．

5. おわりに

本稿では，近年注目されるソフトウェア開発技術とアジャイルの関係について述べた．ワークショップでは，ソフトウェアの変化に対して設計はどうあるべき，どのような設計技術が有効かを議論したい．

参考文献

[1] http://www.agilemanifesto.org/iso/ja/
[2] Babar, A. M. et.al, Agile Software Architecture, Morgan Kaufmann, 2013
[3] 野田 夏子，岸 知二，アスペクト指向を利用したアーキテクチャ設計に関する考察，情報処理学会ソフトウェア工学研究会報告, 2014(17) 1-3, 2014.