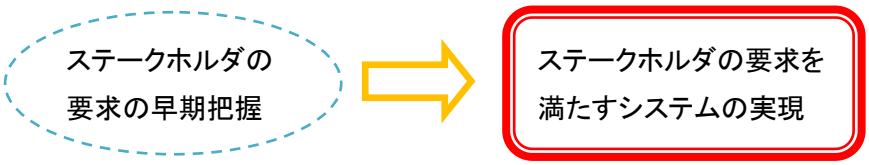
Weaving Together Requirements and Architectures
要求とアーキテクチャを同時に組み合わせる

説得力のある合理的な小前提(議論)は、「なぜステークホルダの要求の早期把握が結果として、彼らの期待を正当化するシステム（要求を満たすシステム）を導くのか」を証明する。



それと同時に、説得力のある議論は、それ以上に要求と制約を発見するための、システムの技術的な実現可能性を評価するための、代替設計の解決方法を見つけだすための原理を提供するために、ソフトウェアアーキテクチャの構造と早期把握を証明する。

ソフトウェア開発団体は、大抵開始点を選択肢—すなわち、アーキテクチャ要求の中から選ぶ。

これは、いつもウォーターフォールモデルが、開発のライフサイクルの中で、次の工程で使うために人為的に凍結された要求文書を生み出すことに終わる。

その代わり、このプロセスは、避けられない要求の変更と望ましい要求の変更に侵されないことによって、ユーザを限定したり、現場の開発者を不利にしたりするアーキテクチャの制約で、システムを開発する。

スパイラルライフサイクルモデルは、徐々に増加する開発プロセスを提供することによって、ウォーターフォールモデルの欠点に対処する。そして、開発者はたびたび徐々に増加する開発プロセスを提供することによる不安定な要求と財源を管理するために、プロジェクトのリスクを変更することを評価する。

細かい粒のスパイラルモデルでさえ、現代のソフトウェア開発の本質と必要なものの両方共の評価をもたらす。

そのようなライフサイクルは、変化する要求の存在の中で安定した、けれども適応もできるソフトウェアアーキテクチャを開発する必要があることを認識する。

このプロセスの要は、開発者がシステムの要求と同時にシステムのアーキテクチャを作ること、さらに開発者が彼らの研究をかわるがわる扱う（インタリーブする）ことである。

↓※見出し※↓

Twin Peaks intertwines software requirements and architectures
to achieve incremental development and speedy delivery.
ツインピークスは、徐々に増加する開発と迅速提供を成し遂げるために、
ソフトウェア要求とソフトウェアアーキテクチャを結び付ける。

THE TWIN PEAKS MODEL　ツインピークスモデル

多くのソフトウェア開発プロジェクトは、明確に定義された問題領域と、厳密な契約上の手続きに対して期待するので、要求仕様書と設計の問題に一斉に、そしてさらにそのように正当に取り組む。

要求の段階と設計の段階の分離を実現することは、大抵難しい。なぜなら、彼らの人工的な順序(= 開発モデル) が、開発者に強制的に、与えられた時間でどちらかの側面に焦点を当てさせるので。

実際には、候補アーキテクチャは、特定の要求を満たすことから、設計者を制限することができる。また、要求の選択は、設計者が選択するか、開発するかといったことでアーキテクチャに影響を与える。

工学に関するソフトウェア開発プロジェクトでの我々の経験に基づき、私と私の同僚は、スパイラルライフサイクルモデルの適応を用いる。

我々は内々に、我々が要求とアーキテクチャに付与する同等の状態を強調するために、このモデルを「ツインピークス」と呼ぶ。

このモデルは、要求仕様とアーキテクチャ仕様を同時に開発する。しかし、それ(ツインピークスモデルで開発すること)は、反復プロセスが図1で提案するように、徐々により詳細化された要求仕様と設計仕様を作る中で、「解決の構造と仕様」を「問題の構造と仕様」と区別するために継続する。

ツインピークスモデルは、バリー・ベームによって識別された3つの管理に関わることに対処する。

●私は、私がそれを理解する時、それを識別するだろう（IKIWISI）

要求は大抵、ユーザがモデルまたはプロトタイプについて反応を提供したり、見てみたりするための良い機会をもった後にのみ、明らかになる。

ツインピークスモデルはユーザに明確に、インクリメンタル開発と、それに伴うリスク管理を可能にして、早期解決の領域を探索することを可能にする。

●大量生産される市販のソフトウェア（COTS）

だんだん、ソフトウェア開発は実のところ、既存の市販のソフトウェアパッケージから、望ましい要求を選択する過程と、特定する過程である。

ツインピークスを使い、開発者は迅速かつ段階的に、市販の製品を使うことで要求とアーキテクチャを特定できる。

開発者は、既存の COTS ソリューション（解決法）を適応させるために、すぐに主要なアーキテクチャの決断事項を作ったり、選択を狭めたりすることによって、利益を得る。

●急激な変化

変更を管理することは、ソフトウェア開発とプロジェクト管理で、根本的な問題である。

細かい粒の開発に焦点を当てるので、ツインピークスは、我々が思いつくような変化を受け入れようとする。

ソフトウェアシステムの中心となる要求(core requirements)を分析し、特定することは、変化する要求の中で、安定したソフトウェアアーキテクチャを開発するために、必要不可欠である。

そのような状況でソフトウェアシステムを開発することは、異なる開発プロセスを考慮することを必要とする。

●IKIWISI に取り組む(対処する)ことは、いつもより早く設計と実装を開始することを意味する。;

●COTS は、要求仕様化の初期段階で、再利用することを必要とする。;

●迅速な変化に適応する間に、競合を維持すること(生産性を維持すること)は、我々に、より早く全ての開発タスクを実行することを要求する。

Figure1. 図1

ツインピークスモデルは、徐々により詳細化された要求仕様とアーキテクチャの仕様を同時に開発する。

これは、ポール・ワードとスティーブン・メラーの『リアルタイムのために構造化された開発(Structured Development for Real-Time)』ではじめて出版されたモデルの適応です。

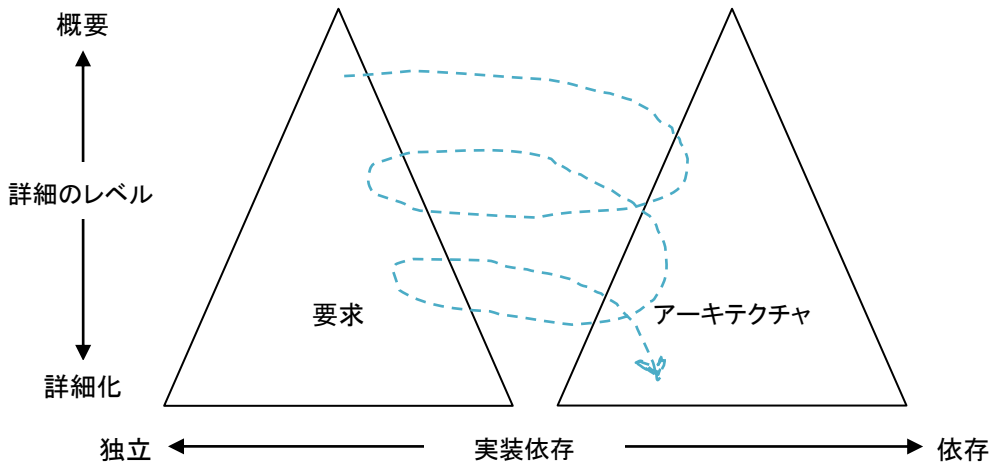


図1.ツインピークスモデル

※ポール・ワードとスティーブン・メラーの

『リアルタイムのために構造化された開発(Structured Development for Real-Time)』

◆要約◆(間違っているかもしれない)参照↓

http://www.lybrary.com/structured-development-realtime-systems-volume-introduction-tools-p-6056.html

組み込みシステムはリアルタイムの反応が不可欠であり、反応が適切な時間に起きないと、人命に関わる可能性がある。しかし、現在のコンピュータシステムは、現実問題を理解せず、コンピュータシステム自体の難しさに注目する傾向にある。本来ならば、モデリングツールや技術を、リアルタイムに開発環境などに適合させることが必要である。

⇒このことから、このモデルは、要求とアーキテクチャの問題を理解し、それぞれ適合できる技術や環境を考え、適合することを考えたモデルであると考えられる。

BUILDING MODULAR SOFTWARE INCREMENTALLY

モジュール式のソフトウェアを増加的に作り上げること

明確に定義されたコンポーネントインタフェース(構成要素界面)でソフトウェアを作り上げることは、効果的な再利用と保守のための良い機会を提供する。

それは曖昧であるが、コンポーネントベース開発が提案する方法は、開発プロセスに適合する。

1つの提案は、要求とアーキテクチャとデザインパターンの活用を考慮することである。

ソフトウェア設計団体は、すでに実装の範囲を表現するために、デザインパターンを特定した。

ソフトウェアアーキテクチャ団体は、様々な全体要求に対処するために、安定したアーキテクチャスタイルを特定する。

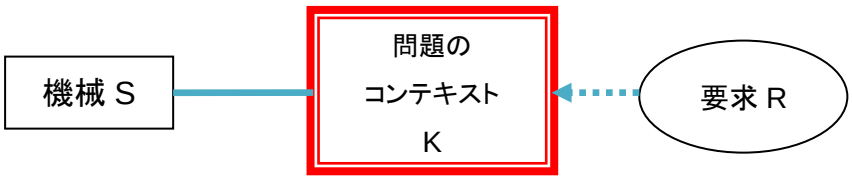
要求工学団体は、解が存在するための問題を特定するために、マイケル・ジャクソンの問題フレームとマーティン・ファウラーのアナリシスパターンの使用の普及を促進してきた。

※問題フレーム参考：http://www.bcm.co.jp/site/2005/2005-09/05-yokyu-09/05-yokyu-09.html

【問題フレーム】問題を識別し、構造化するための分析手法。

・外部要求 R ： 問題世界の望ましいふるまいや性質に関する明示的な要求
・現実世界を構成する問題領域の性質 K ： 世界に関する現象についての記述
・ソフトウェア開発の対象となる機械の仕様 S ：
　実現する機会が、問題領域 K に対して動作する時、外部要求 R が成立する必要がある。

を考える。この分離によって、ソリューションに対する内部要求と外部要求に分類することができる
これにより、外部要求（顧客からの要求）と、内部要求（仕様）の関係を記述（K により）できるので
顧客と開発者のギャップを埋める手段として有効であると考えられている。



問題フレームの構成要素

それらの異なったパターン（デザインパターンとジャクソンとマーティンの3つのパターン）を結び付ける関係は何だろうか？

図2は、コンポーネントベース開発のための始点として、我々が要求と設計とアーキテクチャのパターンを扱うことができることを提案する。

例えば、与えられた固定アーキテクチャは、「我々が対処できる問題の種類」と「我々が開発する可能な設計」を制限することができ、さらに固定した要求は、候補アーキテクチャと設計上の選択を制限することができる。

要求定義工学の視点から、一刻も早く問題フレームを使用して、満足できる問題構造決定(problem structuring)を実現することは、必要不可欠である。

もし、既存のアーキテクチャが、どの程度開発者が問題を構造化するかといったことに影響を与えることができるならば、問題フレームは、既存のアーキテクチャデザインから、リバースエンジニアリングされることを必要とするかもしれない。

Figure2.　図2

同様の注意を受け取る、要求とアーキテクチャと設計の一員で、ソフトウェア開発領域の一部。(Part of the software-development terrain, with requirements, architecture, and design receiving similar attention.)

それぞれのパターンが開発したシステムコンポーネントの種類と、それらの間の関係に影響を与えることは、開発者が採用するプロセスの種類の主な決定要因である。

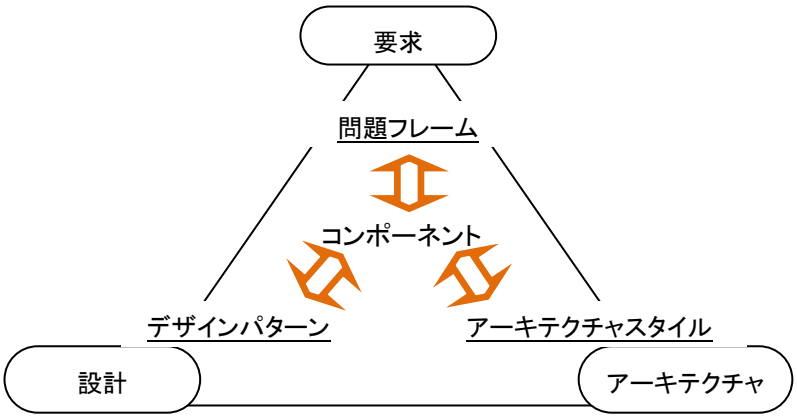


図2.それぞれの関係図

WEAVING THE DEVELOPMENT PROCESS その開発を作る過程

ツインピークスは Kent Beck のエクストリームプログラミング(以後 XP)と、早期に、反復して実装の可能性(要求に対して実現できるか)を探査するゴールのような、多くの共通点を共有する。

ツインピークスはソフトウェア開発の初期段階の活動で――要求とアーキテクチャに焦点を当てるという点において――XP を補足する。

これは潜在的にしばしば XP の弱点であると公言されているスケールのいくつかの論点に対処する。

要求の早期理解とアーキテクチャの選択は大規模なシステムとプロジェクトを管理する鍵である。

もちろん、それ自身で要求とアーキテクチャに焦点を当てることはスケーラビリティの達成に十

分ではない。

モジュール性と反復もまた、極めて重要である。

ツインピークスは本質的に反復され、十分に理解されたパターンから派生した、試されたコンポーネントとテスト済みのコンポーネント組み合わせることは、大規模システムの増分の開発を容易にすることができる。

全部のソフトウェア開発プロセスの結果として、必然的に解決のための問題からより複雑な経路を取る。

要求と設計の間の概念の違いは現在ずっと良い理解と統一がされているのも関わらず、問題の世界と解決の世界の間の移行プロセスは同様に認識されていない。

(Michael GoedickeとBasher Nuseibeh,“要求と設計の間の過程の道”,Proc.第２回世界研究会　統合された設計とプロセス技術,SDPS,オースティン,テキサス,1996,pp.176-177)

※見出し※

そのツインピークスモデルはソフトウェア開発での多くの既存の実行の状態を表す(しかし黙示的に)。

改良された分析と厳しい時間と予算制約の中で高品質なシステムを生産する必要のある計画を組み合わせたので、研究者と開発者は競争市場の中で早急な開発をさせるようなプロセスを開発するためにもがいている。

より丈夫で現実的な開発プロセスは要求工学者とシステム設計者との両方に彼らが生産したいと願う成果物を表すため、共にそして反復して働くことを許す。

このプロセスは開発者にアーキテクチャの制約を考慮することを通じて、より問題を理解させ、そして彼らは要求に基づいたアーキテクチャを開発・適応することができる。

たくさんの難しい疑問は未解決のままである:

- ●どのようなソフトウェアアーキテクチャ(またはアーキテクチャの様式)が変化する要求の存在下で安定していて、そしてわたしたちはどのようにそれらを選択するのか？
- ●どのような要求の集合が他のものより安定していて、そしてわたしたちはどのようにそれらを特定するのか？
- ●システムはどのような変化の種類をそれらの存続期間で経験しそうであり、そして私たちはどのようにこれらの変化の影響を最小限にするために要求とアーキテクチャ(そしてそれらの開発プロセス)を管理するのか？

この疑問への答えは、プロダクトラインとプロダクトファミリ,COTS システム,レガシーシステムを含む、鍵となる新しいソフトウェア開発コンテキストに影響を与えるだろう。

- ●プロダクトラインとプロダクトファミリ、そしてそれは変化する要求を許容する安定したアーキテクチャを必要とする;
- ●COTS システム、そしてそれは(最初からシステム要求を開発することに反対することとして)要求のために既存のアーキテクチャを特定し、適合させることを要求する;そして、

●レガシーシステム、そしてそれは要求仕様書に、既存のシステムの制約を扱うことができる。

ツインピークスの特徴を具体化(統一)するプロセスは避けられない要求の不安定さに直面してアーキテクチャの安定の必要性に取り組む上での最初のステップである。

高速で、段階的に増加する提供を容易にする開発プロセスは、要求の鍵として、次第に市場に出すまでの時間を短くするとともに、早く開発される必要のあるソフトウェアシステムのための本質的要素である。

そのツインピークスモデルは多くの存在するソフトウェア開発中での実行上の状態を表す(しかし黙示的に)。

その進化的な開発の中で、研究を受け入れることに基づいている一方で、そのソフトウェア開発コミュニティはそのようなモデルが受け入れられる実践を表すことだとまだ認めていない。

Basher Nuseibeh はイギリスのコンピュータに関する通信制大学の教授で、そしてロンドンのインペリアルカレッジのコンピュータに関する学部でシステム要求工学のためのセンターの指導者である。