

情報システム学的パタン・ランゲージの再発見

児玉公信[†] 水野忠則[‡]

ソフトウェア工学では、建築学のパタン・ランゲージをノウハウの記述形式という側面
のみに注目して、デザイン・パターンやソフトウェア・アーキテクチャ・パターンといった
大きな成果を産んだ。しかし、パタン・ランゲージのもう一つの側面である「たゆまざる
道」という継続的な活動に着目すると、これは、情報システムの要求記述や設計制約
を、組織学習として組み込んだ理想的な仕組みであることがわかる。本稿では、こうした
観点から、企業情報システム構築のプロセスを、「情報システムサイクル」として規定し、
これにパタン・ランゲージを組み込むことの意義と課題について議論する。

Rediscovery of Pattern Language from Information Systems Viewpoint

Kiminobu Kodama[†], Tadanori Mizuno[‡]

Software engineering yielded the big fruits of the *Design Patterns* and *POSA*, etc using
the Pattern Language of architectonics, but they only focused on the side of the format of
the pattern as know-how description. Focusing on another side of the Pattern Language,
"timeless way" as continuous activities, we can find that the Pattern Language is an ideal
mechanism to incorporate the requirements and the design constraint of information sys-
tems into itself as organization learning. In this paper, "The Information Systems Cycle"
is provided for the building process of the enterprise information systems from such a
viewpoint, and the meaning and the problem to incorporate the Pattern Language into the
cycle are discussed.

1. はじめに

近年の企業情報システム(Enterprise Information Systems)は、事業戦略と密接に関連している^{[1][2]}が、システムの開発、保守、運用に膨大な費用と時間がかかるなど、抱える課題は少なくない。特に、事業環境の変化に俊敏に変更できないことは、事業の根幹を揺るがしかねない重大な問題である。たとえば、生産管理の領域で、個別受注による少量生産の比率が増えているのに、大量生産向けの生産手配システムを使い続けているなどである。この不整合は不良在庫の増加といった形で表面化する。事業環境の変化に追従できない原因としては、変化の速度が速すぎて開発が追いつけない、あるいは開発費が回収できないということもあるが、多くは企業側の情報システム担当者が問題を認識できない、ベテランが定年退職して業務全体を知っている人がいないので判断できないなどの周辺的な要因が積み重なっている。

このように、企業情報システムの企画力、調整力が低下することは、日本の企業力を低下させることに

つながる。情報システムにトラブルが起きた場合の社会的影響も大きくなり、企業としての責任体制の確立が必須となっている。これまでに、企業、特に情報システム部門が主体性を持って情報システムの企画、運営ができるように、プログラムマネジメントのためのフェーズ計画立案と、概念データモデリング、ビジネスプロセスモデリング、要求記述の指導及び教育が行われてきた^[3]が、概してモデリング能力は満足できるレベルにならない。そのこと自体は折り込み済みであるとしても、表面的なモデルしか書かれないことによって、ビジネスシステムに対する本質的な議論や柔軟な発想が阻害されることのほうが問題であった。

多くの場合、Davis がその著書^[4]のまえがきで述べているように、顧客自身が形式的手法で要求文書を書くことは困難であると考えた方がよい。顧客はこれに時間と労力を使うのではなく、より本質的で効果的なビジネスの仕組みを構想すべきである。本稿では、このような状況を前提として、情報システム部門の主体的な要求導出およびプログラム管理の枠組みを提案したい。

[†] 株式会社エクサ, EXA Corporation

[‡] 静岡大学創造科学技術大学院, Graduate School of Science and Technology Shizuoka University

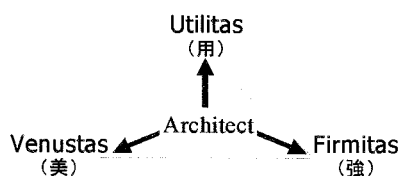


図 1. アーキテクトの役割

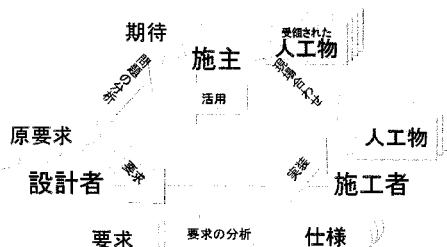


図 2. 情報システムサイクルモデル

2. 情報システムサイクル

企業情報システムの構築は、都市計画に基づく都市建設のあり方に類似している^[9]。現代の都市計画は、既存のインフラストラクチャを活かしつつ、住民との合意を形成しながら、街区の再開発を漸進的に行う。建築学、特に都市計画の知見から情報工学が学ぶことは多いだろう。

2.1. 建築学に学ぶ

ローマ時代の建築家 Vitruvius は、その著書^{[6][7]}の中で、アーキテクトは建築物に対して「用:機能」, 「美:外観」, 「強:構造」のトレードオフを同時に満足するよう働きかけるものと述べている(図 1)。

2.1.1. Zachman フレームワーク

情報システムの構築プロセスを、建築プロセスと対応づけて規定した先行研究として Zachman^[8]および Sowa and Zachman^[9]がある。そこでは、アーキテクトが施主および施工者との対話によって建築が進む様子が描かれている。

建築プロセスは、アーキテクトと施主との建物の用途、敷地面積などについてのやり取りに基づき、「Bubble charts」と呼ぶラフな間取り図を書いて基本的合意と信頼を得ることから始まる。次に、施主の視点で図面 (Architect's drawings) を書いて理解を取った後、施工者の視点で製品、つまり建物の詳細図面など (Architect's plans) を作成する。施工者は、これらの plans から、規制などの制約を加味して、施工方法 (Contractor's plans) を考える。これをさらに小分けして下請け用の図面と指示書 (Shop plans) を書いて渡し、建築作業が行われて完成する。

Zachman は、このようなプロセスと、航空機の製作および情報システムの製作とを対比し、それらの類似性を見た後に、情報システムの構築も、上の建築プロセスと文書化に沿った disciplined approach によって実施し、文書をデータ、プロセス、通信という3つ

の側面で記述することを提案した。これを、Sowa and Zachman は拡張して、文書記述の観点を 5W1H とすることを提案している。

2.1.2. 施主、設計者、施工者およびアーキテクト

Vitruvius によると、アーキテクトは設計者を兼ねながら、「用・美・強」の全体を扱う存在である。

情報システムの構築においても、施主、設計者、施工者、アーキテクトという役割が存在する。むしろ、Zachman が着目したように、アーキテクトという役割は、3 者からなる構造と、その間のプロセスによって相互に規定される。

ただし、Zachman は 1 件の建物の構築しか考えていない。先にも述べたように、現代の企業情報システムは複合的な都市計画として扱われなければならない。これは、企業情報システムの構築も、継続的に時間をかけて行われる漸進的なプロセスであることを意味する。そのプロセスと成果物は、企業の文化を体現したものである。

2.1.3. 情報システムと建築との違い

都市建設プロセスが情報システム構築プロセスのメタファとして使えるとしても、それらが同一であるということではない。その違いについて議論しておく。

情報システムが支援する対象は人間活動システム (Human Activity System; HAS)^[10]である。HAS はソフトなシステム、すなわち、問題に対する解があるとは限らないシステムである。あるところに働きかけても、また別のところで新たな問題が発生するかもしれない。こういう場合は、ステークホルダたちが合意することをもって仮の解とし、それを実行してその結果を評価するというサイクルを取らざるを得ない。情報システムの構築においても、このようなサイクルが有効となる。

構築プロセスに関わる役割についても、単純に対応づけられない。企業情報システムにおける施主とは、ここでは、企業全体の利益代表者として、いわゆる CIO やその補佐役を含む組織全体とみなす。

2.2. 情報システムサイクルモデル

Zachman がとらえた情報システムプロセスを、継続的に循環するサイクルとして再定義する。

2.2.1. 成果物の変換プロセス

情報システムの構築に参加する施主、設計者、施工者の関係と、その間に流れる成果物を図 2 のように捉える。成果物は参加者それぞれが果たす役割によって次々と変換されていく。これを「情報システムサイクル」モデルと呼ぶことにする。アーキテクトは設計者を兼ねるが、少なくとも 1 回のサイクルの完結に責任を負う。

2.2.2. 期待、原要求、要求、仕様、人工物、運用

情報システムサイクルの開始のトリガは、施主による「期待」の表明である。これは目標状態の設定である。その期待に対してどのように事業を行うか、ステークホルダー間で合意を取って、大筋の要求を非形式的に記述したものを「原要求」と呼ぶ。原要求は設計者によって「要求」と呼ぶ形式的な記述に変換され、その内容について施主と合意する。要求記述は、施工者が設計者と打ち合わせて製造「仕様」に翻訳され、その内容について設計者と合意する。これをプログラムとデータ、シェルスクリプトなどの「人工物」に変換し、検査を経たのちに、施主に引き渡される。その際、軽微な修正が行われることがある。これを「現場合わせ」と呼ぶ。また、引き渡し後に行われる修正を「保守」と呼ぶ。施主は実装物を受け取って、実際に使うことで、「効果」を得る。

一般に、企業情報システムの構築は、部分的かつ段階的に行われるため、1 回のサイクルで施主の期待が満足されることは稀であって、このサイクルは継続的に繰り返される。また、この間に事業環境が変化することもあるので、このサイクルは事業が続く限り回り続ける。

2.2.3. ソフトウェアプロセスの位置づけ

ソフトウェアプロセスは、ここでは施工者が行う内部プロセスであると見なされる。内部プロセスも、分析、設計、製造というステップを踏むが、情報システムサイクル上ではそれを明示しない。

いわゆる RFP (Request for Proposal; 提案要請) はアウトソーシングの手続きであって、文書の内容を指しているのではない。一般には、施工をアウトソースする場合の要求記述であり、ソフトウェアプロセスとの最初の接点となる。ただし、RFP が、本稿でいう設計も含めたアウトソースまでも要請している場合があり、責任範囲の混乱の原因となっている。

2.2.4. 「情報システムサイクル」の意義

この情報システムサイクルは、新たにプロセスを設

計したものではなく、これまでに知られているソフトウェアプロセス^[27]のとらえ直しに過ぎない。しかし、このサイクルを認識することによって、特に原要求を記述することによって、ソフトウェアプロセスとの明確に分離し、施主側の主体的な活動を規定できる。

次に、情報システムサイクルにおける主要な作業について詳述する。

2.3. 施主による「期待」の表明

期待とは、施主が求める目標状態(ビジョン)の明示である。これには一切の設計を含めない。期待の表明は施主が自分自身の言葉として述べられる必要がある。このため、たとえば、物語として語られることも多い。これがうまく語られることで、施主チームのトラストビルディングが図られる。この1つの例が、日産自動車のスカイライン R30 の開発リーダー、桜井真一郎氏が語った「戦場ヶ原の稲妻」と呼ばれるストーリー^[14]である。

2.4. 原要求を作る

要求記述の前に原要求を記述する理由は、施主の言葉で要求を述べることによって、施主側のコミットメントを確保すること、継続的活動の主体であることを認識してもらうためである。アーキテクトはそのような言明を引き出すように働きかける。Zachman フレームワークの Bubble charts に相当する。

2.4.1. 原要求の位置づけ

原要求の記述内容は、「期待」と「要求」記述との間のどこかにある。どこにあるかは企業のビジネスの特質や情報システムの発達段階によって異なる。たとえば、システムの応答性がビジネスを左右するようなビジネスでは、性能要求や安全性要求までが原要求に含まれるだろうし、情報リテラシーが不十分な企業では、期待を機能要求として言い換えただけの原要求もありうる。また、設計者のその領域での経験によっても、原要求の内容は異なる。いずれの場合も設計者がそこから設計できる何らかの情報が含まれていなければならない。

原要求には、企業情報システム全体に対するものと、その部分である個別の業務システムに対するものがある。初期的には、これらは渾然一体となっているが、情報システムサイクルが回るに連れて、変化し、分化していく。

「グランドデザイン」という言葉がある。これが設計を意味するのではなく、施主の言葉で書かれた企業情報システムの全体構成、あるいはアプリケーションポートフォリオを指すのであれば、原要求の一部である。

2.4.2. これまでの原要求導出の方法論

これまでに知られている方法論のうち、システムに対する働きかけの循環的な修正を強調しているものは、Checkland らが提案するソフトシステム方法論^[10]である。このほかに、大手システムベンダから提供される問題分析手法がある。これらは、問題－原因ネットワークを書いたり、カウンセリングの手法や KJ 法と類似の手法を用いるなどして、巧みに合意形成を誘導する。この結果、機能要求が得られれば、それを原要求とすることができる。ただし、これらの手法にはプロセスの循環性を念頭に置いていないものもある。

2.4.3. 原要求に書かれるもの

根来^[12]によると SSM の 5,6 番目のステップで、概念モデル中の活動と現実の活動を比較して、改善案を作成することになっている。これが原要求の1つの表現である。

IEEE Std 1362^[13]では Concept of Operation Document (構想書)に書くべき内容を定めている。現状の記述から始まって、変更の必要性、新システムの構想、運用のシナリオ、新システムのインパクト、効果、注意事項を書く。原要求の記述としては、実質的に設計を行っている部分は不要である。

2.4.4. 原要求の早期テスト

記述された原要求について、合意の存在を確認し、言い放して終わらせないために、できるだけ早期にテストする。ソフトウェアのテストではないことに注意すること。原要求のテストは、専門家によるレビューだけではなく、施主自身およびステークホルダが納得できるように、わかりやすい形式、たとえば、ユーザシナリオ、デモビデオ、絵コンテなどを用いて行う。一部の内容に関して、プロトタイプを作るのもよい。

2.5. 設計

設計とは、原要求に対する解決案を要求記述として変換する作業である。要求記述は、建築の領域で言う意匠設計と構造設計の一部に当たる。変換作業自体は、設計者の高度な発明であり、他者の干渉を受けない。

企業情報システムの要求記述は、業務システムのラフスケッチであり、全体構想、概念データモデル、ビジネスプロセスモデル、概念ユースケースなどからなる。これは施工者に対するインプットとなり、施工上の都合、より安価で速い製造方法などのフィードバックを受けて、多少修正される。

こうして完成した要求記述も、早期にテストされる。これは、施主に対する、一般的説明、一部のプロトタ

イプ、デモによって行われる。施工者は、この要求記述に基づいて製造にかかる費用を見積もる。

要求記述のテンプレートとして、IEEE Std 830^[14]や Volere アプローチ^[15]があるが、本来、ソフトウェアの要求記述(SRS)であり、本稿で言う要求記述としては、製造仕様の記述が多いように思う。

3. パタン・ランゲージの再発見

情報システムサイクルモデルの左側、原要求と要求の導出について、Alexander のパタン・ランゲージが参考になる。

3.1. 活動としてのパタン・ランゲージ

パタン・ランゲージの考え方がソフトウェア工学に与えた影響は大きい。その偉大な成果である「デザインパターン」^[21]やそれに続く多くのパターンカタログ、そしてそれらを生み出す原動力となった PLoP の活動^[22]に敬意を表する。これらが、ソフトウェア製造における品質の向上に果たした役割は大きい。

しかし、これまでの情報技術に関わるパタン・ランゲージの活用は、ノウハウ記述としてのパタンの側面だけを取り上げていた。元々のパタン・ランゲージとは、「たゆまざる建築の道」を行く「活動」なのである。これについて、再度確認しておく。

3.1.1. ルール

建築プロジェクト開始の前に、パタン活動を支える原則的なルールが制定される。ルールとは、プロジェクトの遂行において、さまざまな活動の原則を示したものである。

「オレゴン大学の実験」では、①有機的秩序、②参加、③漸進的成長、④“パタン”、⑤診断、⑥調整となっている。「パタンランゲージによる住宅の建設」では、①アーキテクトビルダ、②ビルダーズヤード、③共有地の共同設計、④個々の住宅のレイアウト、⑤一步一步の施工、⑥コストコントロール、⑦プロセスの人的なリズム、となっている。

3.1.2. 建設委員会

「オレゴン大学の実験」では、パタン・ランゲージの制定、改訂を諮る組織が設けられた。これは、施主(大学)、利用者(学生)、アーキテクトの三者からなる。施工者は含まれない。ほかの Alexander のプロジェクトで、建設委員会に相当する組織を正式に設けたことの記述はないが、施主や利用者を巻き込んだ活動になっている。建設委員会は、情報システムサイクルの左上に、利用者を入れたもう1つの三角形として表される(図4左参照)。

2-3 玄関道

・正門より内境界に向けて玄関道がある。玄関道の両側には壁か樹木が並び非常に静かである。

2-5 中央広場

・第2の門の内側には、中央広場がある。この広場は大講堂とともに構成され、講堂の正面は庭に向かっていて。

2-6 大切な中心

・中央広場の先に、第3の門を通り抜けると高校と大学の最も大切な中心がある。ここは幾重もの層を通り抜けて到達できる場であり、そこには静かさがある。

2-8 大切な中心から伸びる腕

・この中心から枝分かれして、個別・分離した大学・高校部分が始まる。

図3 パタン・ランゲージによる原要求記述の例(一部)

3.1.3. 原要求を物語る

「あるパタン・ランゲージ」の使い方は、253 のパタンの中から、建築しようとするテーマに沿って、語り始めのパタンを探し、前後のパタンを参考にして、パタンをいくつか取り出して、必要に応じて変更したものを、そのプロジェクトで使う「その(the)パタン・ランゲージ」とする。これは1つの物語であり、それを語る順序が設計の順序に対応するとされる。

どのように語るかについての具体例が5書には載っていないので、Alexander が手がけた日本の盈進学園東野高校の例^[24]を図3に示す。ここで、下線部分がパタンであり、これに対応するパタン集が別に作られる。図3の記述を読むと、建物の中を歩いて行く(walk-through)ように書かれ、これから出来上がる学校のイメージがありありと浮かぶようである。同高校のプロジェクトでは、このような原要求が建設委員会の手で作られた。

このように、パタンは、施主、利用者、設計者の間で通用する技術語彙である。「玄関道」といえば、3者が直ちに理解する。誤解があれば修正する。

3.1.4. 現場でのイメージアップ

このほかに、Alexander は建設現場に行きつて棒を立てて間取りを決めたり、紙でドアのイメージを書い

て貼ったりして、現場で原要求を具体化する手段をとっている。

3.2. パタン・ランゲージのその他の側面

このほかにも、パタン・ランゲージの活動に関して、特筆すべき特性がいくつかある。

3.2.1. アンチ・マスタプラン

建築物は100年、200年の時を越えて存在する。それを現段階で基本計画(マスタプラン)を立てて実行することには意味がない。前述のように、現代的都市計画においては、現状の都市基盤(インフラ)を活かしながら、住民の合意を得て、漸進的に街区を再開発していくしかない。

トップダウンの構想は必要だが、建設は小規模に、現場に根ざした活動としてボトムアップで行われる。これが同じパタン・ランゲージに従って行われる限り、都市は全体として有機的秩序を保つ。

3.2.2. デザインコード

パタン・ランゲージは設計者の自由度を制約するデザインコードの意味を持つ。自由度は制約されても、設計者が変換作業で用いる作業空間が狭められることで、設計効率は上がると期待される。異なる設計者が設計しても、景観上の秩序は守られる。

3.2.3. XP との共通性

Extreme Programming^[23]では、12のルールが定められている。これがパタン・ランゲージのルールに相当する。XPの提唱者の一人であるBeckは、ソフトウェアにパタンのアイデアを持ち込んだ張本人である。そこではパタンのノウハウ記述の側面だけを取り上げたが、XPでは、ルールと活動の側面、施主の参加を取り上げることで、最終的にパタン・ランゲージ全体をソフトウェア工学にもたらしたと言える。

3.2.4. 学習する組織

パタン・ランゲージがプロジェクトからのフィードバックを受けて、追加修正されるとしたことで、その都市建設における良いパタンが蓄積され、長い時間の中で引き継がれていく。都市自身が学習していくのである。それが「無名の質」なのであろう。特に、業務領域の原則、業界の掟など、明示されにくい知識も蓄積されていくことが期待される。

3.3. パタン・ランゲージの利点と欠点

長い時間をかけながら、有機的秩序を保って作られていく都市にあって、住民が生き生きと生きることこそ、Alexander が見た都市の質であろう。反面、非現代的な都市が作られるかも知れない。これを欠点と見るかどうかは意見が分かれる。

しかし、「4階建ての制限」^[17]というパタンが、高層

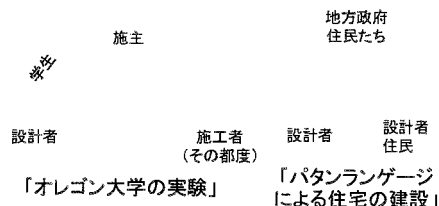


図4 建設委員会

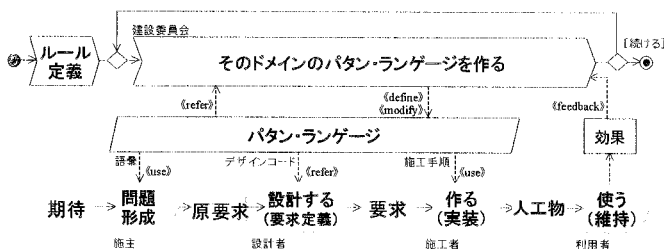


図5 パタン・ランゲージによる情報システムサイクル

建築の非人間性を指摘していたとしても、高層建築が良いと建設委員会が判断すれば、そのようなパタンを作ればよいだけである。ランゲージの内容とランゲージの活動とは別物であり、後者については、継続的な合意形成の手法として、その有効性を試みる価値はある。

4. 情報システムパタン・ランゲージの構想

上で整理したパタン・ランゲージの考え方に基づいて、情報システムサイクルにおける原要求の導出方法と記述方法について私案を述べる。なお、付録に小さな事例を載せた。

4.1. 情報システム版パタン・ランゲージ

情報システムサイクルに、パタン・ランゲージを導入した場合の、作業の流れを Eriksson and Penker^[25]のプロセス図を使って図5に示す。

4.1.1. たゆまざる道

パタン・ランゲージを作り、維持するのは建設委員会であり、これを継続的に行う。これが図5の上半分のサイクルである。個別の業務システムの開発では、図5の下半分に示すように、既存のパタン・ランゲージをデザインコードとして参照し、施主は個別の原要求を、設計者は要求を作り、施工者は実装する。利用者は実装物を実際に運用した上で、設計の良否、実装の良否を評価し、建設委員会はその評価を受けて、必要に応じてパタン・ランゲージに修正を加える。これによって、継続的に企業システム全体の質が改良されていく。

4.1.2. PMO

施主、実質的には PMO (Program Management Office) は、個別ソフトウェア実装案件のリリースタイミングの調整を行う。この調整をプログラム管理という。これが企業情報システムの成長速度を左右する。

4.1.3. 要求管理

原要求がパタンによって語られることによって、原要求と要求の対応づけが明確になる。それ以降の工程でも同様にトレーサビリティが確保される。変更要求も同様に扱える。

4.2. 設計から引き渡しまで

リリースされた個別計画ごとに、設計から引き渡しまでを行う。この流れは、部分的に同時並行的に進むこともある。

4.2.1. ソフトウェアサイクル

情報システムサイクルでは、内部プロセスとしてまとめて扱ったソフトウェアプロセスだが、これが企業の要員自身によって行われることもあるが、一般にはソフトウェアベンダの作業である。

施主の立場から見ると、この作業は個別の計画ごとに、通常1回しか行われぬが、ソフトウェアベンダは類似の作業を他の施主に対して繰り返して行っており、その知見の蓄積もある。したがって、ここにはベンダ側の論理に基づくフィードバックループがある。これは現在では、ソフトウェアプロダクトラインアプローチ^[26]として議論されている。これをソフトウェアサイクルと呼ぶことにする。このイメージを図6のプロセス図に示す。

4.2.2. 斜交する2つのサイクル

このように施主の情報システムサイクルと施工者のソフトウェアサイクルとは斜交している(図7)と考えると良い。この交点で、要求の受け渡し(施工者の立場からは「要件定義」と呼ばれる)と人工物の受け渡しまでが行われる。

強調したい点は、ソフトウェアサイクルの直接の上流に、要求、原要求、期待は存在しないことである。したがって、要求工学には、ソフトウェア工学の価値観とは異なる価値観に基づくアプローチが必要となるだろう。こうした観点からの議論を期待する。

4.2.3. スパイラルモデルとの対比

Boehm のスパイラルモデル^[27]は、情報システムの

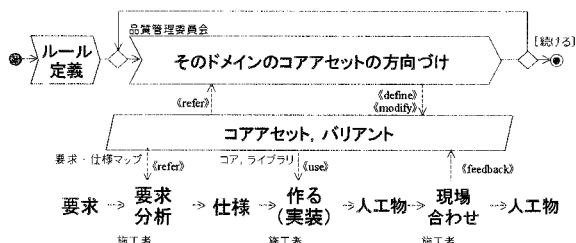


図6 プロダクトラインを組み込んだソフトウェアサイクル

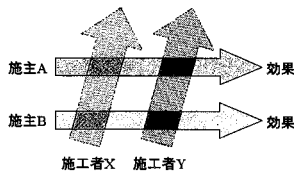


図7 情報システムサイクルとソフトウェアサイクル

構想から始まって、概略設計、詳細設計、実装と4回ラウンドすることで完成することになっている。その間、リスクをヘッジするためにプロトタイピングを行うことが強調されている。これは、その時代性を反映しているが、構想や設計内容を早期にテストすることの有効性を述べている。また、ラウンドごとに計画の調整を行う点も特徴的である。

ただし、スパイラルモデルは、Boehm自身が言うようにソフトウェアプロセスのモデルである。そこには、構想と要求記述が含まれていて、施主、設計者、施工者とが分離されていない時代を反映している。

5. 情報システムパタン・ランゲージの実現に向けて

具体的に、情報システムのパタン・ランゲージの活動を行うに当たり、Alexander が自ら行ったように、この領域の参照モデルとなる「あるパタン・ランゲージ」があった方がよい。現在、この編纂に取りかかっている。

情報システムのパタン・ランゲージは次のような手順で作る。個別開発ケースの要求記述を集めて原要求を設計順に述べるように語り直す。そのときに使われた技術語彙を取り出してパタンフォーマットに書き起こす。収集されたパタンの間で、重複やギャップがあれば、パタン記述を修正・追加する。最後に、全体の構成が、企業全体に関わるパタンから、小さな情報システムのプラクティスに至るまで通して書かれるように、再度、重複やギャップを調整する。

現在の構想では、情報システムのパタン・ランゲージの構成は、「社会」、「組織」、「機能」、「アスペクト」となっている。これは、今後の編纂作業によって変わる可能性がある。

また、パタン・ランゲージの活動の原動力となるルールは、個々の企業が定めるものであるが、参照モデルとして、「オレゴン大学の実験」のそれを用いることを想定している。

6. おわりに

企業情報システムの構築は、1980年代の前半までは、自前で作るのが普通であった。1980年代の後半からは、情報システムが経営戦略の一翼を担うようになって、巨大化、複雑化し、プロフェッショナルが代わって作る時代になった。そして多くの企業が情報子会社を作ることになった。このとき、発注者と受注者の関係が生まれたが、本来発注者が行うべき情報システムの計画、期待と原要求の表明までもが、受注者の仕事にされてしまった。発注者が手放してしまったそれらの責任はやがて失われ、1990年代後半からは、自称「企業情報システム」がパッケージ化されて販売される状態になった。しかし、施主の主体性が失われたパッケージの導入は大した効果を生み出さない。逆に、ソフトウェアベンダの言う「ベストプラクティス」の文化が流入してしまう。

Zachman のフレームワークを契機とする Enterprise Architecture (EA) のアイデアは、本来、情報システムの計画、実施、運用の主権を施主に取り戻す活動であった。本稿は、この活動のための具体的なプロセス、すなわち「たゆまざる道」への初期的な提案のつもりである。

参考文献

- [1] 経営システム研究会(編), 「NTT ドコモ リアルタイムマネジメントへの挑戦」, 日刊工業新聞社, 2004
- [2] 経営情報学会システム統合特設研究部会(編), 「成功に導くシステム統合の論点」, 日科技連, 2005
- [3] 手島歩三, 「悩み深い情報システム部門の再生策」, 日経コンピュータ, 2002/08/26 号, 日経BP, pp. 172-177
- [4] Davis, A. M., *Just Enough Requirements Management: Where Software Development Meets Marketing*, Dorset House Publ. NY, 2005
- [5] 南波幸雄, 「企業情報システムにおけるシステム統合と都市計画アプローチ」, in 「システム統合の論点」, 日科技連, 2005, pp. 79-92
- [6] O'Gorman, *ABC of Architecture*, PENN, 1998
- [7] Vitruvius 著, 森田慶一訳, 「ウィトルーウィウス建築書」, 東海大学出版会, 1979
- [8] Zachman, J. A., *A Framework for information systems architecture*, IBM SYSTEMS JOURNAL,

- Vol. 26(3), 1987. Reprinted Vol. 38(2,3), 1999
- [9] Sowa, J. F. and Zachman, J. A., Extending and formalizing the framework for information systems architecture, *IBM SYSTEMS JOURNAL*, Vol. 31(3), 1992.
- [10] Checkland, P., *Systems Thinking, Systems Practice*, John Wiley & Sons, 1981. 高原康彦ほか監訳, 「新しいシステムアプローチ」, オーム社, 1985
- [11] <http://www.motorfan.jp/modules/xf6section/article-17.html>
- [12] 根来龍之, 「SSM 入門テキスト」(未公刊), 1994年6月
- [13] IEEE 1632, *Guide for Information Technology – System Definition -- Concept of Operation Document*, IEEE, 1998
- [14] IEEE 830, *Recommended Practice for Software Requirements Specifications--Description*, IEEE, 1998
- [15] Robertson, S. and Robertson, J., *Mastering the Requirements Process, 2nd Ed.*, Addison-Wesley, 1999
- [16] Alexander, C., *The Timeless Way of Building*, Oxford Univ. Press, 1979. 平田翰那訳, 「時を越えた建築の道」, 鹿島出版, 1993
- [17] Alexander, C. et al, *A Pattern Language*, Oxford Univ. Press, 1977. 平田翰那訳, 「パタン・ランゲージ」, 鹿島出版, 1984.
- [18] Alexander, C., *The Oregon Experiment*, Oxford Univ. Press, 1975. 宮本雅明訳, 「オレゴン大学の実験」, 鹿島出版, 1977.
- [19] Alexander, C. et al, *The Production of House*, Oxford Univ. Press, 1985. 中埜博監訳, 「パタンランゲージによる住宅の建設」, 鹿島出版, 1991.
- [20] Alexander, C., *A New Theory of Urban Design*, Oxford Univ. Press, 1987. 難波和彦訳, 「まちづくりの新しい理論」, 鹿島出版, 1989.
- [21] Gamma, E., et al, *Design Patterns*, Reading, MA, Addison-Wesley, 1995. 本位田真一ほか監訳, 「デザインパターン(改訂版)」, ソフトバンクパブリッシング, 1999
- [22] Hillside Group, URL <http://hillside.net/>
- [23] Beck, K. et al, *Extreme Programming Explained: Embrace Change*, Addison-Wesley, 1999
- [24] 羽生田栄一, personal communication, 2007

- [25] Eriksson, H. and Penker, M., *Business Modeling with UML*, Addison-Wesley, 2000. 本位田真一ほか監訳, 「UMLによるビジネスモデリング」, ソフトバンククリエイティブ, 2002
- [26] Sugumaran, V., Park, S. and Kang, K. C., *Software Product Line Engineering*, *CACM*, Vol.49(12), 2006, pp. 29-32
- [27] Boehm, B. W., *A Spiral Model of Software Development and Enhancement*, *COMPUTER*, IEEE, 1988, pp. 61-72

付録

「期待」と「原要求」記述の小さな事例を示す。施主は、ある業務システムの運用管理者である。

期待

ビジネスのスピード, 信頼性, 継続性を保証すること。

- (1)ユーザが直接使うプログラムはサクサク動いて欲しい,
- (2)経営環境の変化に合わせてプログラムもその場で変更したい,
- (3)過去, 現在のデータが失われないように保証したい。

原要求

(1)サクサク動くプログラム

取引件数, 接続ユーザ数にかかわらず, 一定の応答時間が保証されている。

(2)改修しやすいプログラム

どのプログラムのどこをどう改修するかが分かっている, 誰でも改修できる。

(3)データが失われない保証

大量データの確実で高速なバックアップコピーが確保できる。大量データの確実で高速なリストアが実施できる。

パタン・ランゲージ(一部)

2 誰でも改修

背景

設計書の精度が悪い。設計者も既に不在である。

問題

特定のベンダしか改修できず, 時間と費用がかかる。

解決策

現行システムを作り直す。その際, 合意された開発手法に従って, 改修向けのドキュメントを作る。その内容は...

注意事項

現行データの移行方法を検討すること。「データの移行」パタンを参照のこと。