システム理論的因果律モデル STAMP に基づくプロセス改善分析

日下部　茂
長崎県立大学
kusakabe@sun.ac.jp

荒木　啓二郎
九州大学
araki@ait.kyushu-u.ac.jp

要旨

プロセス改善のための分析に，システム理論的因果律モデル *STAMP* を用いるアプローチを提案する．継続的にプロセスを改善し最適化できる状態が目標であることを前提に，プロセスの状態を表現するモデルを用いて目標としている状態に向けた制御ができるかを分析する．提案手法の有効性については，これまでのプロセス改善のアプローチと同様のことが，提案するアプローチで系統的に導くことができるかという観点で評価する．

1. はじめに

ソフトウェアの開発効率の向上や品質の向上には要素技術だけでなく，開発プロセスも含めたライフサイクルプロセスが重要とされている．プロセスは，一旦構築すればそれで終わりということはなく，状況に応じてテーラリングや改善を継続的に行うことが重要である．本研究では，プロセスの状態を表現するモデルを用い目標としている状態に向けた制御ができているかを，システム理論的因果律モデルで分析するアプローチを提案する．

プロセス改善の一つのアプローチとして、最初は標準的なプロセスのプロセステンプレートを採用し，その定着後はそれをベースに継続的に改善を続けるものが考えられる．その際，プロセスデータを測定しながら改善を行う仕組みが埋め込まれたテンプレートを使っていれば，新しい技術の導入の得失をデータに基づいて判断しやすく，適切な導入が実現しやすいと考えられる．そのようなプロセスのテンプレート例として，個人レベルでは PSP(Personal Software Process)[1]，チームレベルでは TSP (Team Software Process)[2] といったものがある．このような考えのもと，我々は形式手法のような新しい技術の導入によるプロセスの改善についての取り組み行ってきた [3] [4] .

しかしながら，個人や組織の開発プロセスに，形式手法のような新しい技術や手法を取り入れ定着させるには，関係者を適切に動機づけることも重要とされている．このような観点から，動機づけ理論のうち，過程理論の一つを基礎として，動機づけプロセスの状態遷移モデルを提案し，新しい技術や手法の導入から定着に至るプロセスの分析に用いた研究もなされている [5].

本研究では，前述の関連研究のように状態を用いたモデルを用いるが期待と，実際の活動を行う側の実際には差があるという観点でモデルを考える (図 1 参照)．常に期待と実際の間にはギャップがあり，順調なのは適宜フィードバックを得ながら監視と制御を行いギャップを許容範囲に収めているからと考える．このギャップが大きくなり過ぎた結果，受容できない損失が生じることをアクシデントとしてモデル化する．開発プロセスに関して期待する状態と，実際の状態のギャップが大きく，アクシデントが起こりかねない状態をプロセス改善のハザードと考え，そのハザードを防ぐことに着目する。

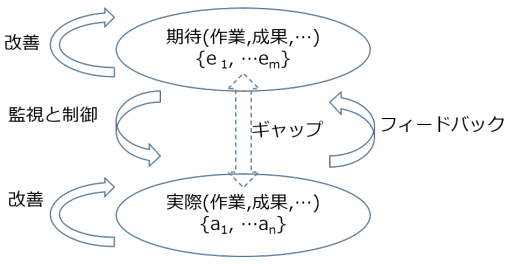

図 1. 期待と実際のギャップに対する監視と制御

前述の観点に基づきシステム理論的因果律モデル STAMP(Systems-Theoretic Accident Model and Pro-

cesses) [6] を用いてプロセス改善を分析するアプローチを提案する．STAMP を利用した分析を行うことで，体系的にプロセス上での脆弱な箇所をハザード要因として識別し，改善すべき点を明らかにすることができると考える．提案手法の有効性については，これまでのプロセス改善の事例と同様の改善が，提案の分析で系統的に導くことができるかという観点で評価する．

本稿の構成は以下の通りである．第 2 節で，システム理論的因果律モデル STAMP について説明する．第 3 節で，個人レベルのプロセス改善フレームワーク PSP を紹介し，第 4 節で PSP に STAMP にもとづいた分析を適用する例について論じる．第 5 節でまとめを行う．

2. システム理論的因果律モデル STAMP

2.1. STAMP 概要

STAMP は MIT の Nancy Leveson 教授によって提唱されたもので，従来の解析的還元論や信頼性理論ではなくシステム理論に基づき，システムを構成するコンポーネント間の相互作用に着目した事故モデルである．STAMP のモデルは，安全制約，階層的なコントロールストラクチャ，プロセスモデルという三つの基本要素で構成されている．

- コントロールストラクチャ：システムの構成要素間の構造と，相互作用を表したもの
- プロセスモデル：コントロールする側がその対象プロセスをコントロールするアルゴリズムと対象プロセスを (抽象化して) 表現したもの
- 安全制約：安全のために守るべき制約

STAMP では，コントロールストラクチャとプロセスモデルに対して，システムの安全制約が正しく適用されているかどうかに着目する．STAMP のコントロールストラクチャとプロセスモデルの基本形を図 2 に示す．

コントロールストラクチャは，一般には上述の基本形を組み合わせ，システム内の複数階層の複数コンポーネントの構造とやり取りされる制御の指示やフィードバックなどの関係を表す．STAMP では，安全に関するコントロールストラクチャがシステムの安全制約を守れず，システムがハザード状態になることでアクシデントが発生すると考える．

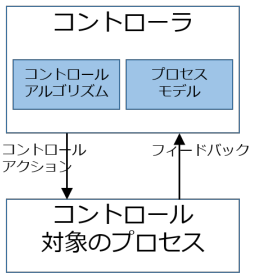

図 2. コントロールストラクチャ

STAMP では，システム内だけでなく，開発や運用のプロセスにおけるコントロールストラクチャも含めて統一的にモデル化できる．そのため，エンジニアリングの観点での要因分析や，人や組織間の関係，規則までを含めた要因分析などが可能である．図 3 にシステムの開発と運用におけるコントロールストラクチャの例を示す．

STAMP 自身はアクシデントを説明するモデルであるが，STAMP をベースとした解析の道具立てやプロセスは複数提案されている (図 4 参照)．STAMP を用いた安全制約の分析は，システムだけでなく，その開発プロセスや運用プロセスに対しても行うことができる．例えば，ソフトウェアの開発に関しては，ソフトウェアだけでなく，開発プロセスにも適用できる．ハザード分析手法 STPA (System - Theoretic Process Analysis)， 事故分析手法 CAST (Causal Analysis based on STAMP)， STPA-Sec (STPA for Security)， STECA (System-Theoretic Early Concept Analysis) といった分析法が提案されており，航空宇宙分野をはじめセキュリティや医療など様々な分野ですでに利用されている．

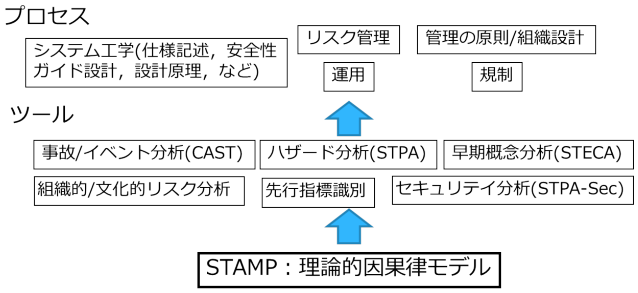

図 4. システム理論的因果律モデル STAMP とその活用プロセス

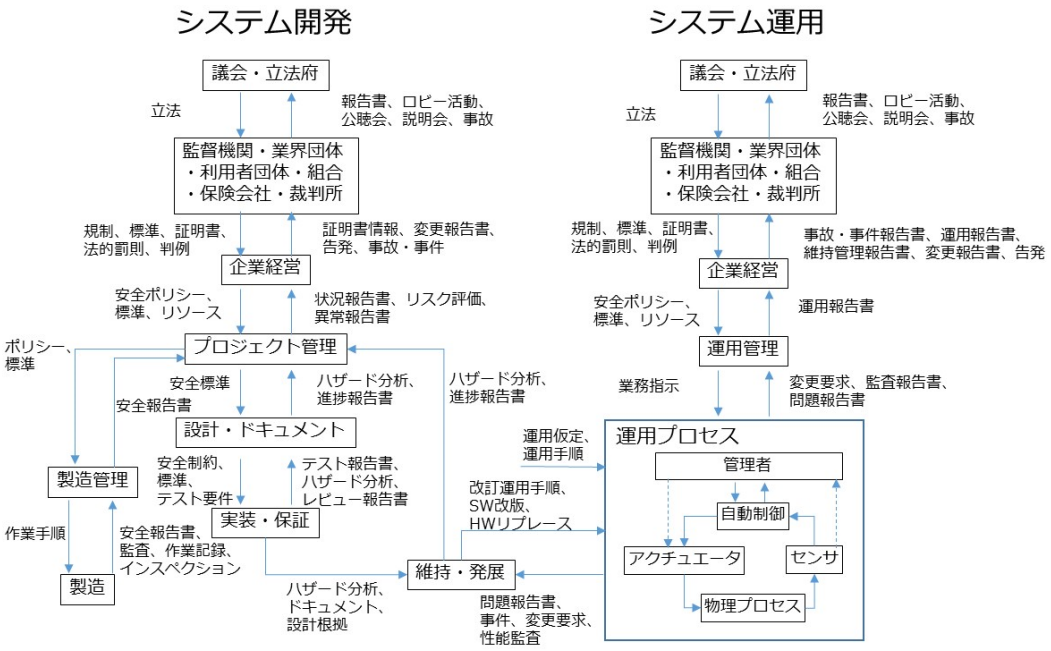

図 3. 開発と運用におけるコントロールストラクチャの例

ソフトウェア開発プロセスの場合，プロセスを設計する場合は STPA，実績データの分析に基づく改善の場合は CAST を適用するといった使い分けが考えられる．

2.2. STPA

STPA は，STAMP モデリングにおける代表的なハザード分析法で，典型的にはシステム開発の際にトップダウンで用いられる [7]．STAMP/STPA では，まず対象システムを理解するといった準備の後，回避すべきアクシデントとハザードを決めて STAMP によるモデリングを行い STPA の分析に着手する．STAMP モデルの前述の三つの要素を用い，コントロールを行う側とその対象プロセスとの間の相互作用において，安全制約が不適切であったり，守られない状態になったりするシナリオを中心に分析する．

以下に，STPA の手順の例を示す．実際には，ハザード分析を行う目的や段階の違いも含め，STAMP/STPA の具体的な適用法には様々なバリエーションがあり得る．

- 準備 1 ： アクシデント，ハザード，安全制約の識別
- 準備 2 ： コントロールストラクチャーの構築
- Step1：安全でないコントロールアクション（Unsafe Control Action：UCA）の抽出
- Step2 ：非安全なコントロールの原因の特定

最初に，アクシデント，ハザード，安全制約を決めるが，STAMP/STPA の特徴の一つとして，安全工学の技法を広い分野で適用できるようにとの意図の下、これらが定義されていることがある．プロセス改善の場合，プロセスに関する受容できない損失をアクシデントとすることで，STAMP/STPA が適用可能である．

アクシデント，ハザード，安全制約，コントロールストラクチャを定めたのち，コントローラからのアクションのうち以下の 4 つのタイプの非安全なコントロールアクションを識別する．

1. Not Providing causes hazard（与えられないとハザード）：安全のために必要なコントロールアクションが与えられないことがハザードにつながる．
2. Providing causes hazard（与えられるとハザード）：ハザードにつながる非安全なコントロールアクションが与えられる．
3. Too early/too late, wrong order causes hazard（早過ぎ，遅過ぎ，誤順序でハザード）：安全のための

ものであり得るコントロールアクションが，遅すぎて，早すぎて，または順序通りに与えられないことでハザードにつながる．

4. Stopping too soon/applying too long causes hazard（早過ぎる停止，長過ぎる適用でハザード）: (連続的，または非離散的なコントロールアクションにおいて) 安全のためのコントロールアクションの停止が早すぎる，もしくは適用が長すぎることがハザードにつながる．

ハザードにつながる 5 番目のタイプとして，必要なコントロールアクションが与えられたけれども実行されない，という場合もある．

非安全なコントロールを識別したのち，それにつながるシナリオを考え原因を識別する．安全制約を保つためのコントロールループやその構成要素を確認し，以下のような点を分析する．

1. プロセスモデルが正しくなくなってしまう原因も含め，どのようにして非安全なコントロール，ひいてはハザードにつながるかを分析する．
2. 必要なコントロールアクションが与えられたにもかかわらず，適切に実行されない原因を分析する．この分析結果は，(FTA, HAZOP, FMEA といった) 既存の分析の結果を含み得る．

図 5 はコントロールループにおいて，齟齬の原因となりうるものの例である

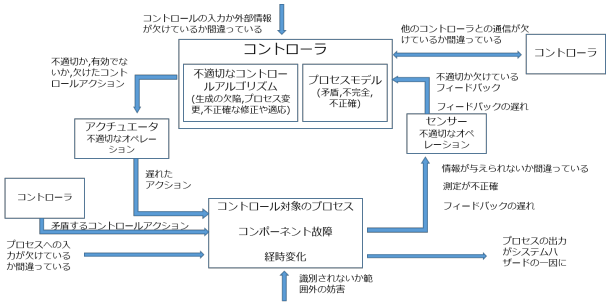

図 5. Causal Factor の例

3. プロセス改善フレームワーク PSP

ここでは個人レベルのプロセス改善フレームワーク PSP を簡単に紹介する．PSP は個人レベルのプロセスを対象としているが，CMM (CMMI の前身) に基づいて開発され，レベル 5 相当のものとして設計されている．そのトレーニングコースには複数のものがあるが，いずれも段階的にプロセス改善の知識とスキルを獲得する．図 6 に 8 つの課題に取り組むコースの概要を示す．

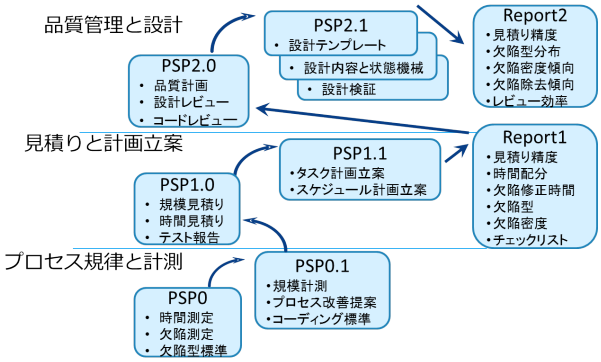

図 6. PSP トレーニングコースでの段階的な知識とスキルの獲得

知識とスキルに関する講義を受け、対応する演習を行う。その規模や欠陥等のプロセスの履歴データに基づき，プロセスに関する自身の問題点を定量的に分析する．分析結果に基づき，プロセス改善提案を行い，引き続く演習の実績により改善効果を確認する．受講者は，このような一連の講義と演習とを通じて，高品質ソフトウェア開発のための継続的な改善に必要とされる，プロセスの知識とスキルの定着を目指す．

このような段階的なプロセス改善を，STAMP モデリングに当てはめると，図 2 のコントローラ中のプロセスモデルに含まれる状態変数や，それらを監視・制御するアクション数が増え、アクションを決定できるアルゴリズムが高度化することに対応する。

4. PSP への STAMP 適用

4.1. PSP のインストラクタ向け分析

ここでは，PSP トレーニングコースのインストラクタが適切に受講者を指導するための分析を STAMP モデリングを用いて系統的に行うことについて議論する．各開発者個人が PSP トレーニングコースを受講し，必要な知識とスキルが定着した後は，図 1 でモデル化される

ような継続的な改善活動を個人レベルで行うことが想定される．しかしながら，トレーニング中は，図 7 のような入れ子のモデルとなる．受講者が各ステップでの知識とスキルの確実な習得を目指した活動を行うと同時に，インストラクタは受講者を外側から観測し各ステップでの確実な知識とスキルの習得を促進する指導を行う必要がある．

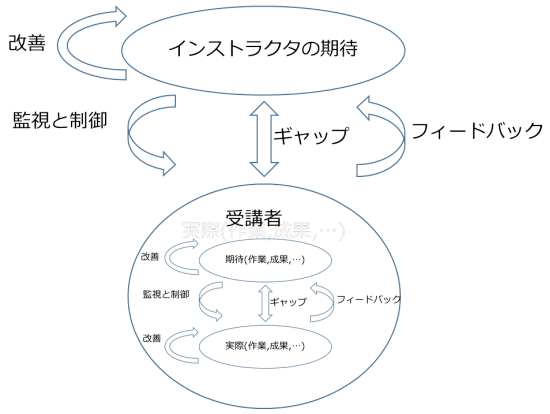

図 7. PSP トレーニングコースにおけるインストラクタの期待と受講者の実際

これを図 2 のような STAMP のモデリングに当てはめて考えると，インストラクタは，トレーニングの進捗に応じた受講者の期待モデルを持つと同時に，受講者の実際の状態を適切に把握するフィードバックを得る仕組みと，指導のアクションが出せるコントロールアルゴリズムを持つ必要がある．このような STAMP モデルに STPA 分析を行うことで，PSP トレーニングでのインストラクタの振る舞いに関するハザード分析が可能となる．

例えば図 8 に示すようなフェーズ構成を持つ PSP2.x では、インストラクタが受講者の状態のフィードバックを得て指導を行うタイミングは、典型的には計画フェーズと事後分析フェーズが想定されている．その際のインストラクタの振る舞いに対して STPA 分析を行うことで，指導方法を検討する際の系統的な分析ができる．指導の失敗事例に対して STAMP モデリングを行い失敗の原因について CAST 事後分析を行うことで，失敗事例を系統的に分析できる．

文献 [5] では，動機づけに着目した状態と操作により動機づけプロセスを定式化し，PSP 教育手法の改善への応用を論じているが，動機づけ失敗も含めたシナリオを系統的に分析できているわけではない．我々のアプローチでのプロセス改善に関する状態変数の集合に，動機づけに関する状態変数を加えることで、動機づけも含めてたモデルで分析できる．トレーニングコースの受講者は，通常は研究室やプロジェクトに属しているなどで、動機づけに関する他のコントローラが存在していることが多い．図 5 のように，受講者にとっての複数のコントローラを明示的に分析に加えることで、インストラクタ以外からの動機づけへの影響も含め系統的に分析が可能になる。

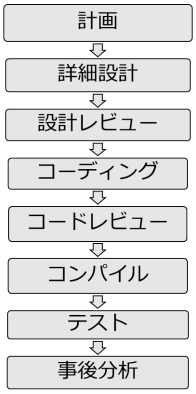

図 8. PSP2.x でのフェーズ

4.2 STAMP による形式手法を用いたプロセス改善分析

PSP では，プロセスデータに基づく改善を可能とするような測定の仕組みが埋め込まれており、履歴データに基づいてプロセスをハザードに導くシナリオを分析できる．例えば、主要な履歴データの一つである欠陥の記録には、自由記述的な欠陥の説明に加え、欠陥のタイプ、混入したフェーズ、検出・修正したフェーズといった事前に定められた区分に基づく記述も含まれる。このような欠陥の記録に基づいて、望ましくない欠陥が生じている状態をハザードとして，混入が特に多いフェーズや、混入から検出までのフェーズ間距離が長い欠陥のタイプといった観点で分析に着手できる。

PSP の各フェーズの進行手順には、開始基準や終了基準が設けられているが、それらが適切に守られていないことがハザードに結びついている可能性がある。例えば、特定のフェーズで欠陥の混入が多発するというハザード

の場合、フェーズ内での作業内容の問題に加えて、開始基準やその確認法といった問題で、本来はまだ開始できる状態にない作業に着手している可能性がある。特定の欠陥タイプに、混入から検出までのフェーズ間距離が長いものが多い場合、そのパス上のフェーズの終了基準の内容や確認法に、該当欠陥タイプ固有の問題があり、本来は検出されるべきところで欠陥が検出できずにすり抜けている可能性がある。

このような状況を STAMP のコントロールストラクチャに対応付け、ハザードが発生するシナリオを分析し、例えば以下のように形式手法導入の有効性を分析できる。

- 作業に関するアクションの結果である成果物に着目し、フェーズ間で受け渡される作業成果物の曖昧さの影響の伝搬を防止するため、形式的な仕様記述言語で作業成果物を記述する.
- 作業の進行を適切に制御する観点で、開始・終了条件の厳密な判定に焦点を当てる。あるフェーズで準備が不十分なまま作業が開始されているのは、前フェーズの終了基準の判定が不適切であった可能性も考慮する。問題が顕在化しているフェーズの前フェーズの終了基準を厳密に判定するために形式的な仕様記述の導入を検討する。

5. おわりに

システム理論的因果律モデルを用いてプロセス改善を分析するアプローチを提案した．本稿では主に個人レベルのプロセス改善を対象に、アドホックに行ったこれまでの事例に，提案手法を適用し，同様の改善をより系統的に導くことができる見通しを得た．これらの結果をふまえ，プロセスのテーラリングや改善についての系統的に取り組を発展させていくことができると考える．

本稿では主に個人レベルのプロセス改善を論じたが、STAMP は解析的還元論で説明できない創発的な特性も対象にできるため、チームレベルのプロセス改善も分析できる。図 3 に示すように、STAMP では，人や組織間の関係，規則まで含めた要因分析も可能であるため、CMMI のような組織レベルの分析も対象とできる。これまでの研究で、プロセス改善モデル CMMI-DEV[8] のモデル要素を用いた分析で，形式手法のような要素技術を起点にしたアプローチであっても，直接関係する技術的なプロセス領域以外の様々なプロセス領域との関係も考慮する必要があることが確認できている [9]．これらの結果をふまえ，組織レベルでの様々なプロセス領域に属するプラクティスや作業成果物の間の相互作用を STAMP モデリングを用いて分析する研究を今後行う予定である．

参考文献

[1] Humphrey, W. S.: PSP: A Self-improvement Process For Software Engineers, Addison-Wesley Pub, 2005.

[2] Team Software Process, http://www.sei.cmu.edu/tsp/

[3] 小材健，日下部茂，大森洋一，荒木啓二郎：測定可能な個人プロセスを対象とした形式手法導入に関する提案，情報処理学会ソフトウェア工学研究会研究報告，2009-SE-163-21 pp.17-24, 2009.

[4] 山田真也，日下部茂，大森洋一，荒木啓二郎：ソフトウェア開発チームプロセス演習 TSPi へのモデル指向形式手法 VDM の導入事例，ソフトウェアシンポジウム SS2012 予稿集, 2012.

[5] 梅田 政信, 片峯 恵一, 石橋 慶一, 橋本 正明, 吉田 隆一,“ソフトウェアプロセス教育における動機づけプロセスの定式化と教育改善への応用,”信学技報, vol. 113, no. 71, KBSE2013-10, pp. 55-60, 2013 年.

[6] Nancy Leveson, Engineering a Safer World, MIT press, 2012.

[7] An STPA Primer, psas.scripts.mit.edu/home/wp-content/uploads/2015/06/STPA-Primer-v1.pdf

[8] CMMI, http://cmmiinstitute.com/

[9] 日下部茂，林信宏，大森洋一，荒木啓二郎，ソフトウェア開発プロセス改善モデル CMMI-DEV の関連プロセス領域ネットワークの中心性分析，日本ソフトウェア科学会 コンピュータソフトウェア, Vol. 32 (2015) No. 3 p. 3_126-3_136