第5章　ソフトウェアエンジニアリングの人間的側面

*IEEE Software*のジャーナル特別編において、ゲスト執筆者らが次のような洞察[deS09]を示している。

> ソフトウェアエンジニリングは、ソフトウェア開発プロセスとプロダクトの双方を向上させるために作られ、関連する技術、ツール、手法が多数存在する。次々と新しい技術が登場し、使うに値する結果をもたらしている。

> しかし、ソフトウェアエンジニアリングは効率の悪い活動を改善する策として使われるだけではない。 ソフトウェアは人によって作られ、そして使われ、人々の相互関係を支援する。人の性格や態度、協調がソフトウェア開発の中心となる。

> スキルやモチベーションを欠いたチームが成功することは稀である。

5.1 ソフトウェアエンジニアの特徴

ここまで来ていきなりだが、あなたはソフトウェアエンジニアになりたいだろうか。エンジニアは技術的に熟達する必要がある。また、問題を理解して有効な解決策を設計し、ソフトウェアを構築してできる限り高い品質のプロダクトを目指してテストもする。そのためこれらのスキルを身につける必要がある。変更を管理し、ステークホルダとコミュニケーションを行い、適切なツールを必要に応じて使用する。以降の章では多数の紙面を割いてこれらの内容を記述する。

これら以外にも重要なものがある。優れたソフトウェアエンジニアであるための人間的側面である。Erasmusは、ソフトウェアエンジニアにおける7つの「プロフェッショナル（superprofessional）」な特性について言及[Era09]している。

優れたソフトウェアエンジニアは、**責任感を持っている**。この責任感とは、同僚やステークホルダ、マネージャに対して成果を約束し、成果の達成へ向けて最善の努力をすることを意味する。

優れたソフトウェアエンジニアは、**感受性が鋭い**。既存のソフトウェアソリューションの変更を求めるステークホルダやチームメンバーのニーズを感じ取ることができる。また、そのプロジェクトを担当するマネージャのニーズも把握している。優れたエンジニアは、業務環境を観察し、その環境に自分の行動をあわせることができる。

優れたソフトウェアエンジニアは、**包み隠すことなく正直である**。設計に問題を発見したのであれば、正直かつ前向きにその問題を指摘する。スケジュールやフィーチャ、性能、他のプロダクトやプロジェクトの状況に対して事実を歪めるよう求められた場合でも、正直かつ現実的な判断を下す。

優れたソフトウェアエンジニアは、**プレッシャーの強い状況でも実力が発揮できる**。ソフトウェアエンジニアは常にカオスの縁に立っている。要件変更や優先順位の変更、ステークホルダや同僚たちの要望、威圧するマネージャ等さまざまなプレッシャーが存在している。優れたエンジニアは、プレッシャーと上手く付き合い、成果を出せる状態を保つ。

優れたソフトウェアエンジニアは、**公正さへの意識が強い**。仲間たちとの信頼を分かち合い、利害衝突を避け、他の人の業務を邪魔しない。

優れたソフトウェアエンジニアは、**細やかな注意を払う**。完璧さに拘るということではない。プロジェクトやプロダクトに影響する性能やコスト、品質等の広い基準に照らし合わせて日々の技術的判断を慎重に行うという意味である。

つまり、優れたソフトウェアエンジニアは現実的である。ソフトウェアエンジニアリングとは、特定の環境において役に立つ規律であることを優れたエンジニアは理解している。

5.2 ソフトウェアエンジニアリングの心理学

```
図5.1 ソフトウェアエンジニアリングにおける階層行動モデル [Cur90を参考とした]
Software ソフトウェア
Problem 問題

Individual 個人の層
Team チームの層
Project プロジェクトの層
Company 企業の層
Business Milieu ビジネス環境の層

Cognition and Motivation 問題の認識と解決へのモチベーション
Group Dynamics 集団力学
Organizatinal Behavior 組織行動論
```

Bill CurtisとDiane Walsは、ソフトウェアエンジニアリングの心理学への影響が強い論文にて、ソフトウェア開発における階層行動モデル（図5.1参照）[Cur90]を提案している。ソフトウェアエンジニアリングの心理学における「個人の層」では、解決すべき問題の認識、その問題を解決するための問題解決スキル、モデル上の外層に起因する制約の下で解決を完遂するまでのモチベーションに焦点をあてる。 対して「チームの層」および「プロジェクトの層」では、集団力学（group dynamics）の要素が支配的になる。この層では、チーム体制、コミュニケーションのような人間的要素が成功のカギとなる。各チームメンバーのスキル同様に、グループ内コミュニケーション、協調や連携が重要なのである。さらに外側の層では、組織行動論（Organizatinal Behavior）の要素がビジネス環境に対する企業の方策や対応を決める。

5.3 ソフトウェアチーム

古くから親しまれる書籍*Peopleware*内で、Tom DeMarcoとTim Listerはソフトウェアチームについて次のように語る。[DeM98]

> 一緒に仕事をするグループをすべて「チーム」と呼ぶように、ビジネスの世界ではチームという言葉をかなりずさんに扱っている。しかし、そのほとんどはチームとはいえないものだ。共通の目標も、特徴あるチームの精神もないからである。「ゼリー状[訳注1]」と呼ぶべき現象が見られないのだ。

> ゼリー状になったチームは、部分的な最適でなく全体の最適に向けて、非常によくまとまったグループになる。〈中略〉いったんチームがゼリー状になり始めると、成功の可能性が飛躍的に向上する。チームは成功へ向けて止まらなくなる。〈中略〉チームを従来の手法で管理する必要はなく、動機づけも不要となる。弾みがつくのだ。

[訳注1] ゼリーは食べ物のゼリーを考えてもらってよい。個人ではなくチームとしてまとまり、1人が欠けたとしてもチームが変わらず動くことができるような状況を指している。

DeMarcoとListerは、ゼリー状になったチームは平均よりも明らかに生産性が高く、成功に向けた意欲も高いと主張している。メンバーは共通の目標、共通の文化、そして多くの場合エリート意識を共有している。エリート意識こそが彼ら/彼女らを際立たせる原点である。

チームをゼリー状にするための絶対確実な方法は存在しないが、優れたソフトウェアチームには共通性がある[1]。Miguel Carrascoは、優れたソフトウェアチームは**目的意識**を構築することが必要であると提案している[Car08]。各メンバーが自分のスキルや貢献に価値があると感じる**関与意識**も、優れたチームには必要である。

[1] Bruce Tuckmanは、成功を収めるチームが成果を出せるようになるまでの4段階のモデルを示した。リンク：https://en.wikipedia.org/wiki/Tuckman%27s_stages_of_group_development

優れたチームは**信頼意識**をメンバー間で育む。そのため、チーム内のソフトウェアエンジニアは、同僚やマネージャのスキルや適性を認めることができる。また、メンバーに**改善意識**を奨励する。チーム全体で、ソフトウェアエンジニアリングのアプローチを一定の間隔でふりかえり、業務改善の方法を追い求めつづける。

最後に、優れたチームにはさまざまなスキルをもつメンバーが揃っているものである。つまり**多様性**をもつ。例えば、ステークホルダのニーズをうまく捉えるメンバーが、そうでない技術者を支える等、補完しあえるチームである。

しかし、すべてのチームが優れておりゼリー状であるとは限らない。多くのチームがJackmanの呼ぶ「チームの毒（team toxicity）」に悩まされている[Jac98]。彼女は「潜在的に毒を生み出してしまう環境」について次の5つの毒を定義している。

> 1. 無秩序な仕事環境
> 2. チームメンバー間に摩擦を生じさせるほどに高い欲求不満
> 3. 断片的で不適切なソフトウェアプロセス
> 4. チームやメンバーの不明瞭な役割の設定
> 5. 繰り返し露見される失敗

仕事環境が無秩序になることを回避するためには、プロジェクトマネージャはチームメンバーが作業するために必要な情報すべてへアクセスできるようすべきである。また、一度確定されたゴールや目的は、やむを得ない場合を除いて変更すべきではない。チームメンバーそれぞれに意思決定の権限を与えることにより、欲求不満の発生を回避できる。不必要で厄介な作業が発生したり、断片的で使えない成果物を作成してしまうようなプロセスの存在は不適切であるが、開発するプロダクトとそれに携わるメンバーの属性を理解し、チーム自身でプロセスを選択させることで防ぐことができる。チームは説明責任を果たすことができる仕組みを構築し、メンバーが失敗した場合のリカバリーに向けたアプローチ方法を定義する必要がある。テクニカルレビュー[2]は、説明責任を果たすためのよい方法である。最後に、失敗しそうな雰囲気を回避するポイントは、チーム主導のフィードバックならびに問題解決のノウハウを確立することである。

[2] テクニカルレビューについては第16章にて詳細を記載する。

Jackmanの呼ぶこれら5つの毒に加えて、チームはメンバーそれぞれの特性の違いにも悩まされることがある。直感的に情報収集し、共通性が見えづらい事実から幅広い概念をまとめあげていく者もいれば、データを集めて整理し、順を追って情報を処理する者もいる。論理的かつ順序だった論拠があると意思決定を行いやすいチームもあれば、「直感的」に意思決定をしたいチームもある。マイルストンの日付よりも大幅に前倒しで作業を完了できるように一生懸命作業し、締切日が差し迫ってくるストレスから逃れる者もいれば、期日ぎりぎりの土壇場で追い込みをかける者もいる。本節で示される文献を参考に人の違いを認識することは、ゼリー状のチームを構築するために役立つだろう。

5.4 チーム体制

「最良の」チーム体制は、組織のマネジメント方法、チームの人数やスキル、取り組む問題の難易度よって変わる。Manteiは、ソフトウェアエンジニアリングチーム体制の計画時に考慮すべき7つの要素を挙げている[Man81]。

- 開発にて解決すべき問題の難しさ
- コード行数あるいはファンクションポイント[3]で示されるソフトウェアの規模
- チームで作業する期間（チームの寿命）
- 問題が細分化できる度合い
- システムに要求されている品質や信頼性
- 納期の厳しさ
- プロジェクトで求められるコミュニケーションの度合い

[3] コード行数（LOC：Line of Code）やファンクションポイントはプログラム規模を数値化する。これらについて第24章にて議論する。

ソフトウェアプロジェクトにおける疫病のような問題の解毒剤として、10年以上前からアジャイルソフトウェア開発（第3章参照）が提案されている。アジャイル開発では顧客満足、早期かつインクリメンタルなソフトウェアリリース、少人数でモチベーションの高いプロジェクトチーム、形式ばらない方法、最小限の開発成果物、開発すべてをシンプルにすることを推奨している。

アジャイルチームと呼ばれる少人数でモチベーションの高いプロジェクトチームは、本書5.3節で論じてきた優れたソフトウェアプロジェクトチームの特徴を多数備えている。しかも、問題を発生させるチームの毒の多くを回避する。アジャイルの理念では、個人（チームメンバー）の高い能力とメンバー同士の協力がチームの重要な成功要素であることが強調されている。高いパフォーマンスを出すチームが想像的なエネルギーを発揮できるようにすることはソフトウェアエンジニアリング組織の中心的なゴールとなる。

CockburnとHighsmithが次のように記している[Coc01a]。皆、同意する内容であろう。

> 優れたソフトウェアエンジニアは、どのようなプロセスでも業務を行うことができる。対して、能力が低ければ、どのようなプロセスでも問題が起きるであろう。「人はプロセスに勝る」のである。しかしながら、不明確なプロセスおよび不十分なサポートの下では、優れたソフトウェアエンジニアであっても活動を妨げられてしまうものである。

チームメンバーの能力を活用し、プロジェクトにて有効な協力体制を築くために、アジャイルチームは自己組織化（self-organizing）されている。自己組織化されたチームは、1つのチーム体制を維持する必要はなく、エンジニアリングにおける問題解決方法の進化や開発環境の変化に応じて体制を変化していく。

チームメンバーとプロジェクトにおけるすべてのステークホルダとのコミュニケーションは必要不可欠である。顧客側の代表がアジャイルチームの一員という場合もある。この顧客を含めた体制は、進化するプロダクトにおけるタイムリーかつ頻繁なフィードバックという利点をもたらすだけではなく、開発者とステークホルダ双方の尊敬を育む。

5.5 ソーシャルメディアによる影響

Eメール、テキストメッセージ、ビデオ会議はソフトウェアエンジニアリング業務において当たり前となった。ただ、これらを用いたコミュニケーションは、対面でのやり取りの代替品や補足にすぎない。しかし、ソーシャルメディアにはそれ以上の価値がある。

Begelらはソフトウェアエンジニアリングにおけるソーシャルメディアの成長とその適用状況について、次のように記している[Beg10]。

> ソフトウェア開発に関する社会的なプロセスは、〈中略〉同一のゴールを共有でき、お互いを支えあうスキルをもつ人を発見しつなぐ能力により大きく左右される。この能力は、チームメンバーのコミュニケーションやチーム体制作りといった、ソフトウェアのライフサイクル全体における協調や連携に役立つ。これらに貢献するソーシャルメディアが市場にて成功を収めているのである。

ある意味、「つながり（connection）」は対面のコミュニケーションと同じように重要である。ソーシャルメディアの価値は、チーム規模が増大するにつれて高まる。チームが地理的に分散した際にその価値はさらに高まる。

FacebookやLinkedIn、SlackやTwitter等の「ソーシャルネットワーキングサービス（SNS）」では、ソフトウェア開発者とその関係する技術者の間接的なつながりを活用できる。SNS上の「友達」という仕組みは、とあるアプリケーションドメインにおける問題解決への専門知識をもっている「友達の友達」との出会いにつながる。SNSの仕組みは、組織内の非公開ネットワーク内でも使われることがある。

ソフトウェアエンジニアリングにソーシャルメディアを活用する際には、プライバシーとセキュリティは見逃すことができない重要な問題である。ソフトウェアエンジニアにおける業務の多くは雇用主に所有権があり、情報開示が契約違反の場合がある。このため、ソーシャルメディアの利点と非公開情報の開示のリスクについてトレードオフを行い、よく考えるべきだ。

5.6 グローバルチーム

ソフトウェアのドメインにおいて、グローバル化は国境を越えてプロダクトやサービスを輸出入する以上の意味がある。この数十年間において、複数の国に跨るチームが生み出すソフトウェアプロダクトが増え続けている。グローバルソフトウェア開発（GSD：Global Software Development）チームはコラボレーション（連携）やコーディネーション（調整）、コミュニケーション、意思決定において独特の取り組みを行っている。GSDチームにおけるこれらの取り組みはチームの体制に影響を受ける。グルーバルなソフトウェアチームの意思決定は、次の4つの要素[Gar10]により、単国でのソフトウェアチームにおける意思決定よりも複雑である。

- 問題の複雑さ
- 決定事項における不確実性やリスク
- 予期せぬ結果の法則（例：ある作業の決定が別プロジェクトの目標に意図しない影響を及ぼす）
- 今後の方向性に影響するような問題に対する複数の見解

```
図5.2 GSDチームに影響を与える要素
Distance  距離
Communication コミュニケーション
Collaboration  コラボレーション（連携）
Coodination  コーディネーション（調整）
Barriers and Complexity  文化の違いによる障壁や複雑さ

Introduces  もたらす
Attenuate  減衰させる
Reduces  削減する

Complicates  複雑にする
Enhances  拡大させる
Improves  より良くする
Accentuates the need for  必要性を際立たせる
```

GSDチームにおけるコラボレーション、コーディネーションそしてコミュニケーションへの取り組みは、意思決定に多大な影響を与える。図5.2にGSDチームの活動における距離の影響を示す。距離があることでコミュニケーションが複雑となり、コーディネーションの必要性を際立たせる。また、距離が離れることにより、各チーム文化の違いによる障壁や複雑さが発生する。距離がSN比（信号対雑音比）を減衰させるように、障壁がコミュニケーションを減衰させてしまう。この減衰により、プロジェクトが不安定になってしまう。

5.7 まとめ

優れたソフトウェアエンジニアは、高い技術力をもつ。加えて、成果への責任を持ち、同僚の必要性を知り、正直にプロダクトやプロジェクトを評価する。また、プレッシャーに強く、同僚たちを公平に扱い、細やかな注意を払うことができる。

ソフトウェアエンジニアリングの心理学では、「個人の層」における問題認識やモチベーション、「チームの層」における集団力学、企業における組織行動論を紹介した。優れた（ゼリー状の）ソフトウェアチームは、平均より生産的でモチベーションが高い。優れたチームは、目的意識、関与意識、信頼意識、改善意識をもつ。加えて、無秩序で欲求不満の高い環境、不適切なソフトウェアプロセス、チームの不明瞭な役割、繰り返し露見される失敗といった「チームの毒」を回避する。

アジャイルチームはアジャイルの哲学を支持し、固定的な役割や外部からのマネジメントをしている従来のソフトウェアチームより自律（自己組織化）していることが多い。アジャイルチームはコミュニケーション、シンプルさ、フィードバック、勇気、尊重の5つの価値を重視する。

ソフトウェアプロジェクトの多くで、チームのコミュニケーションや連携の強化を図れるソーシャルメディアは不可欠となった。グローバルソフトウェア開発では、距離があることでコミュニケーションへの障壁が発生しやすいため、ソーシャルメディアとリモートコミュニケーションは特に役立つ。

問題と考察のポイント

5.1 優れたソフトウェア開発者を観察および調査し、共通的な人格的特徴を3つ挙げよ。

5.2 「包み隠すことなく正直」になるため、どのようにふるまうべきか。侮辱や攻撃的と（他の人に）認識されないか。考えを述べよ。

5.3 「人為的な境界」はチームのコミュニケーションを減衰させる。どのようにその境界が作られてしまうのか。

5.4 図5.2について、なぜ距離がコミュニケーションを複雑にするのか。距離により調整が必要であることをなぜ強調するのか。なぜある種の障壁や複雑さが距離によってもたらされるのか。