連載「プロマネの現場から」

第２０６回　「good enough（グッド・イナフ)」という思想

蒼海憲治（大手 SI 企業・グループ会社・事業部長）

1980 年代に、個性的な歌声やキャラクター、ファッションで、「ガールズ・ジャスト・ワナ・ハヴ・ファン」「タイム・アフター・タイム」等のヒット曲を連発したシンディ・ローパーさんが、先月 4 月 19 日、6 年ぶり通算 15 度目の最後のジャパン・ツアーのため来日しました。大阪で 1 日、東京で 2 日の開催でしたが、Asue アリーナ大阪のチケットが取れ、ライブを聞くことができました。1983 年に、30 歳でソロデビューした遅咲きのシンディさんでしたが、現在 71 歳になった今も、その歌声は健在でした。

この日のライブは、「シー・バップ」から始まり、「グーニーズはグッド･イナフ」が続きました。この曲は同名の映画『グーニーズ』の主題歌ですが、当初の題名は「グッド･イナフ（good enough)」だったといわれています。

「good enough」は、「それで十分だよ」や「まあまあいいよ」という程度の意味ですが、完璧や最高レベルのものを表すものではありません。しかし、単なる妥協や手抜きをすることと異なります。その背景には、より現実的で効率的なアプローチや、精神的な負担の軽減といった意図があります。そのため、以前から、この「good enough」という考え方は、生き方やシステム構築においても、参考になるのではないか、と思っています。

今回は、「good enough」という考え方について、紹介してみたいと思います。

「good enough」とは、「完璧を目指さず、実用的・現実的なラインで満足する」ことを意味します。つまり、完璧を求めるよりも、必要最低限、あるいはそれ以上を満たしていれば OK とし、素早く行動し、次に進むことを重視する、という姿勢になります。

そもそも「good enough」という考え方が生まれた背景には、完璧を目指すことにより生じるデメリットがあります。

①時間がかかりすぎる
②コストがかかりすぎる
③機会を逃す
④ストレスが増え、心が疲弊する

そのため、ＶＵＣＡといわれ、先が見通せず、変化が速い社会においては、「80%できていればリリースして、改善しながら完成度を高めていく」ほうが、結果的に成功しやすい場面が多くなるという背景があります。

この「good enough」の核となる考え方には、以下のものがあります。

（１）目的達成志向

最も重要なのは、設定した目標を達成することです。完璧である必要はなく、目標を十分に満たすレベルであれば良いと考えます。

（２）効率性重視

完璧を追求するには、時間、労力、資源が無限にかかる可能性があります。

「good enough」においては、ある程度の水準に達したら、そこで区切りをつけて他の重要なタスクの効率を高めることができます。

（３）現実的妥協

世の中には、どれだけ努力しても完璧には到達できないものが多く存在します。また、完璧を追求することが必ずしも最良の結果をもたらすとは限りません。

「good enough」は、現実的な範囲で満足できるレベルを見極めることを意味します。

（４）ストレス軽減

常に完璧を目指すことは、大きなストレスや不安につながります。

「good enough」の考え方を取り入れることで、プレッシャーから解放され、精神的な余裕を持つことができます。

（５）バランス感覚

完璧主義は、準備に時間がかかりすぎたり、失敗を恐れて行動に移せなかったりする原因になることがあります。

「good enough」は、完璧さと不完全さのバランスを取ることで、ある程度のレベルで完了させ、次のステップに進むことを促します。

一方、「Good Enough」は、決して、手を抜くことや、品質を下げることを意味するものではありません。

「good enough」は、必要な質を確保した上での効率性を追求する考え方であり、最初から質を落とすことや努力をしないこととは異なります。

また、現状維持や現状に甘んじるのではなく、目標達成のために必要なレベルを見極める考え方です。

システム構築において、「good enough」の考え方が大切だと考える理由には、以下のものがあります。

（1）リソース（時間・コスト）の制約があるため

完璧を目指すと、設計や実装、テストなどに非常に多くの時間とコストがかかります。「good enough」な状態でリリースし、実際に使いながら改善していく方が、リソースを有効に使え効率的になります。

（2）要件は常に変化するため

初期段階で完璧な設計を目指しても、ユーザーのニーズやビジネス環境は変化します。変化に柔軟に対応するには、初めから「完璧」を追求するよりも、「good enough」な「現実的に使える」システムを目指し、継続的に見直す方が理にかなっています。

（3）ユーザーフィードバックを早く得たいため

「good enough」な状態で早めにリリースすることで、実際のユーザーからのフィードバックを得られます。これにより、机上の空論ではなく、実際の利用状況に基づいた改善をすることができます。

（4）パレートの法則

多くの場合、機能の 80%は、全体の 20%の努力で実装できます。残りの 20%の機能を完璧に仕上げるには、非常に大きな労力が必要になります。「good enough」では、この 80%に焦点を当て、最大限の効果を最小限の労力で得ようとするアプローチです。

（5）「金メッキ」「過剰設計」を防ぐ

完璧を追い求めると、実際には不要な機能や複雑な構成になりがちです。これは保守性を下げ、トラブルの原因にもなります。
「good enough」を意識することで、シンプルで実用的なシステムが作れます。

システム構築における「good enough」の適用した具体的な事例には、以下のようなものがあります。

（1）MVP（Minimum Viable Product）開発

新しいプロダクトやソリューションを開発するとき、「すべての理想的な機能を最初から作る」のではなく、まず「一番大事な機能」だけに絞って動く最低限のバージョン（MVP）をリリースします。そうすることで、実際のユーザーからフィードバックを得て、機能追加や改善を続けることができます。

（2）要件定義フェーズでのスコープ限定

要件定義フェーズにおいて、顧客から多くの要望があり、納期・品質・予算の面からみて、一度に取り込めないことが明白なときがあります。そのような場合、「必要十分な要件に絞って合意し、完璧を目指さずまず動くものを作る」必要があります。

（3）システム移行時の段階的リリース

旧システムから新システムへの移行する場合、いきなり全社一斉リリースするのではなく、一部部署だけ先にリリースし、そこで、課題や問題を洗い出してから全体展開をする。最初の段階では多少のバグや不便があっても、業務の成立性の見極めを行い、移行判定を行う。

以上、よいことずくめにみえる「good enough」ですが、上手くいくことばかりではないと思っています。プロジェクトメンバーの意識が、通常の品質保証プロセスの遵守する意識と異なり、「どうせ good enough だから」と思ってしまうことです。その結果、本来守るべきルール（作業プロセス、品質基準、セキュリティ、法令遵守など）を無視・軽視して、「手抜き」「品質低下」「セキュリテリスク」「コンプライアンスリスク」が生じるリスクがあります。

そのため、
（1）目的を満たしているか
　必須の要件や機能は満たしているか
（2）リスクは許容範囲か
　致命的なバグや法律違反を侵していないか
（3）次に進める状態になっている
　次フェーズの開始条件が整っているか
（4）顧客やチームからフィードバックが得られるか
（5）コストパフォーマンスは最適か

これらを「good enough」でタスクを続けることの判断基準とし、作業の妥当性や継続可否の判断を行う必要があります。

最後に、シンディさんと日本との関係は深いのですが、一番印象的な出来事は、東日本大震災時のエピソードになります。２０１１年３月１１日の地震発生した時、ジャパン・ツアーのための来日の飛行機の中でしたが、シンディさんの乗った飛行機は地震閉鎖のため成田空港に降りられず、横田基地に緊急着陸します。他の来日アーティストが日本公演を中止しましたが、シンディさんは日本に残り、予定通りライブを敢行し、会場では、被災者の募金活動もされました。日本のファンに寄り添い、力を与えたいという気持ちが強く伝わりました。今回で、ワールドツアーとしては最後になるのだと思いますが、またいつか、シンディさんの歌声を聞いてみたいと思っています。