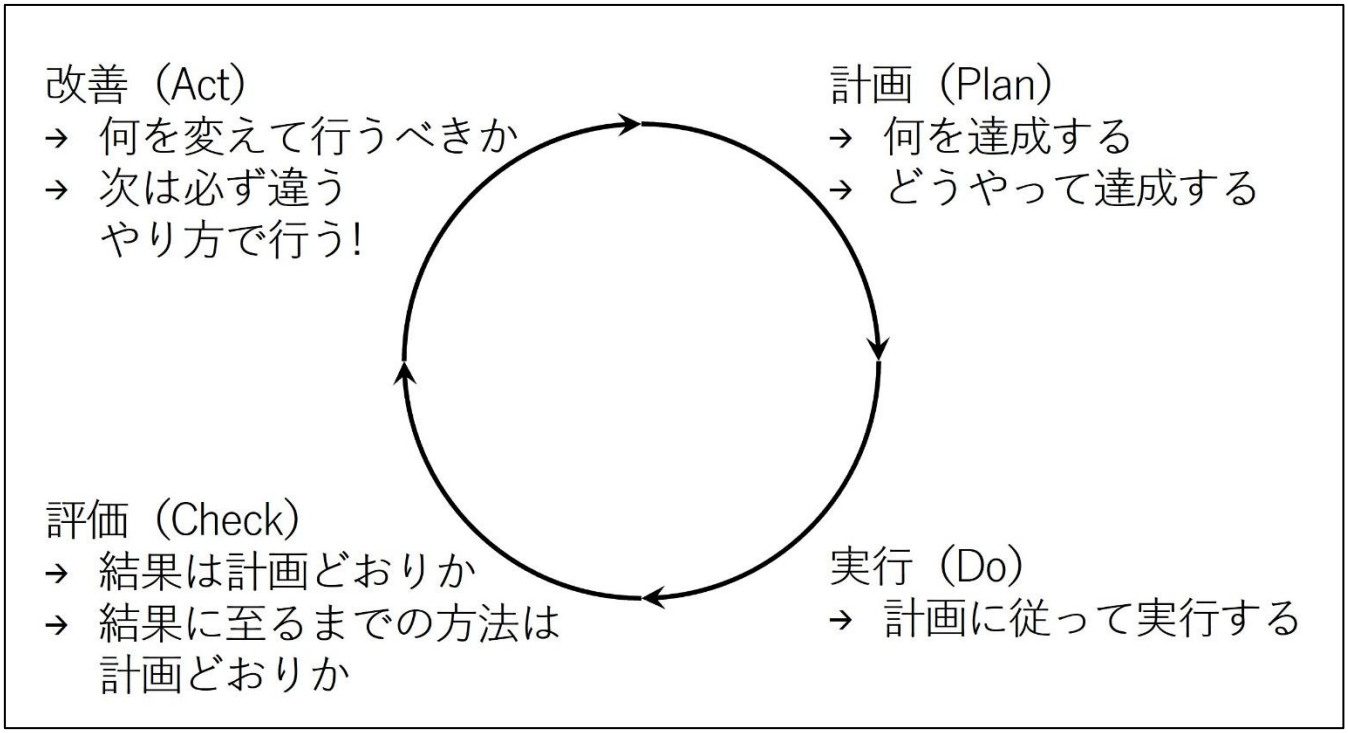
ニルス・マロトー

Predictable Projects

How to deliver the Right Results at the Right Time

予測可能なプロジェクト

適切なタイミングで適切な結果を出す方法



予測可能なプロジェクト

適切なタイミングで適切な結果を出す方法

1. プロジェクトの成果と納入日を予測することは可能ですか？

本書では、過度なストレスを抱えることなく、大幅に短い期間でプロジェクトを成功に導くための、極めて実践的なアプローチを解説します。同時に、いつ、何が完了するのかを驚くほどの精度で予測できるようになります。

不可能に聞こえるでしょうか？ 多くの人が、実際に試すまではそう思っていました。信じがたい話ですか？何しろ、あなたは長年プロジェクトに携わってきました。必要な資格も経験もすべて持っています。プロジェクトは予想以上に時間がかかるものだと誰もが知っているのに、一体何を見落としていたのでしょうか？突然、プロジェクトがより良い結果をはるかに速く出せるようになるなんて。私はほぼ必ず、「そんなことは不可能だ」「以前試したことがある」「聞いたことがある」「ここではうまくいかない」「私たちのプロジェクトは違う」「私たちはとてつもなく大きなシステムを構築している」「私はすでにこれらのことをやっている」といった抵抗の声を聞きます。これらの議論は十分に理解できます。なぜなら、人々は、大幅に短い時間でより良い結果を出すための技術が存在するとは想像できないからです。そうでなければ、彼らはすでにこれらの技術を使っているはずですよね？

素晴らしい成果を約束する理論的なプロセスは数多く存在しますが、それが現実世界で機能するかどうかは実践によってのみ明らかになります。本書で紹介するアプローチは、数十年にわたる研究と豊富な経験に基づき、国や文化、分野を問わず、大小さまざまなプロジェクトの現場で検証され、磨き上げられてきたものです。半信半疑であっても、この手法を実践した人々の生産性が数週間で約 30%も向上したという事実があるなら、少なくとも試してみる価値はあると思いませんか？「百聞は一見に如かず（プリンの味は食べてみなければわからない）」と言います。実際に味わう前に、その味を説明するのは容易ではありません。しかし、味見はすぐにできます。チームの生産性が 30%向上するということは、同じ人数、同じ時間で、約 40%も多くの価値を生み出せるようになるということを意味するのです。

本書では、まず背景について説明し、次に「進化型（Evo）プランニング」アプローチを構成する様々な要素について解説します。その後、Evo プランニングの最初の要素についてさらに詳しく掘り下げます。最後に、実践的な経験を積むために 5 週間の演習に取り組んでいただくことをお勧めします。信じられないかもしれませんが、このテキストを読んだだけでは想像もつかないほど、この手法には多くの重要なディテールが詰まっているからです。さらに詳しい背景や解説については、www.malotaux.eu をご覧ください。www.malotaux.eu/downloads では多数の小冊子やプレゼン資料を、www.malotaux.eu/conferences では動画を無料でダウンロードいただけます。著者は、読者の皆様からのご意見やご賛同、さらには反論を含め、幅広いご議論を歓迎しております。実際にこの手法を味わってみたい方は、ぜひコーチングをご依頼ください。メール：niels@malotaux.eu

2. 時間と予測可能性の重要性

プロジェクトの開始が一日遅れるごとに、完了も一日遅れます（図 1）。その結果、私たちや顧客は本来得られるはずの収益を失うことになります。収益はプロジェクトのコストをはるかに上回るはずなので、そうでなければそもそもプロジェクトを開始すべきではありません。あなたがこのプロジェクトをまだ始めていない唯一の正当な理由は、限られたリソースをより収益性の高いプロジェクトに費やしているからです。



図 1：プロジェクトのアイデアが生まれた瞬間から、タイマーが作動し始め

当初、プロジェクトは 1 日経過するごとに潜在的な価値を生み出しますが、次第に収穫逓減の領域に入り、1 日追加で費やしても期待されるリターンを下回るようになります。つまり、最適な利益を得るためには、納入日を適切に設計する必要があるということです。結局のところ、プロジェクトマネージャーは適切な結果を適切なタイミングで提供する責任を負いますが、適切なシステムやサービスを提供するのにかかる時間は、チームの*全員*が決定するのです。

エンジニアはしばしば、要件が「すべて」満たされるまでプロジェクトは終わらないと考えがちです。しかし、納入日も一つの要件であり、往々にして最も重要な要件であるにもかかわらず、なぜ他のすべての要件が、その最重要要件よりも優先して扱われるのでしょうか？すべてのステークホルダーからの要件はしばしば矛盾するため、設計プロセスは、相反する要件に対する最適な妥協点を見つけるために存在するのです。

ある通信会社のプロダクトマネージャーから聞いた話では、2 人が 4 人で 10 ヶ月かかるプロジェクトを分析していたそうです(図 2)。しかし、システムに 20 万ユーロを投資すれば開発期間を 4 ヶ月短縮できるものの、同社はそれほどの投資を行う用意がなかった。開発期間は 4 ヶ月短縮されたものの、プロジェクト費用はわずか 16 万ユーロしか削減されなかった。つまり、16 万ユーロを節約するために 20 万ユーロを費やす必要はなかったのです。

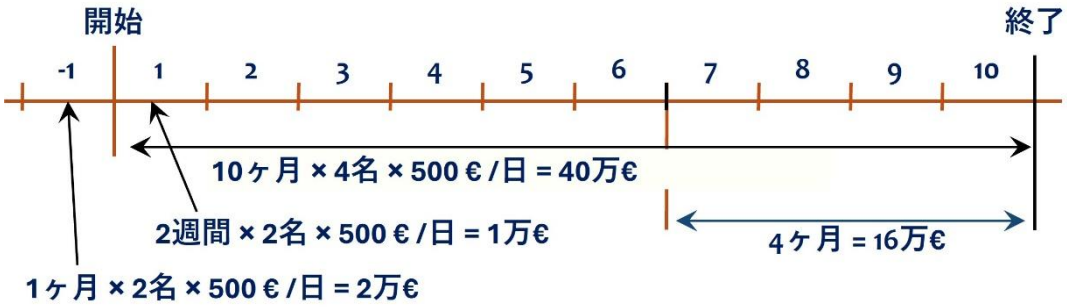


図 2：時間の重要性

アナリストたちは「この 20 万ユーロを投資すべき理由をより明確に説明するために、さらに 2 週間調査させてください」と述べた。プロダクトマネージャーは、プロジェクトはすぐに開始すべきであり、これ以上の追加分析に時間を浪費すべきではない、と不満を漏らしました。

私は彼に「この人たちの時給はいくらですか？」と尋ねました。プロジェクトに関わるほとんどの人と同じように、彼も知りませんでした。そこで私は「1 日 500 ユーロはどうですか？」と提案しました。彼はうなずきました。「つまり、彼らは約 2 万ユーロ（2 人、20 日間、500 ユーロ）を費やし、さらに調査するために約 1 万ユーロを追加で費やしたいということですか？」彼は「そうですが、これ以上時間を無駄にさせたくありません！」と言いました。私は「プロジェクトが完了したら、どのようなメリットが得られるのでしょうか？」と尋ねました。答えは「年間 1600 万ユーロの利益が見込まれます！」でした（通信業界は金のように貴重なのでしょうが、もちろんこれらの企業は私たちに教えてくれません）。「なるほど！つまり、追加の 2 週間のコストは 1 万ユーロではなく約 50 万ユーロであり、4 ヶ月を短縮するための投資は 20 万ユーロのコストではなく、むしろ約 500 万ユーロの追加利益を確保するものなのですね！」プロダクトマネージャーは事業部門の担当者の元へ駆け込み、「その 20 万ユーロを投資して、今すぐプロジェクトを始めましょう！」と訴えました。

プロジェクトのコーチングを始める際、私が最初に尋ねることの一つは、次のとおりです。

- 「（不必要な）遅延が 1 日発生した場合のコストはいくらですか？」
  人々はたいてい答えられません
- 「プロジェクトの 1 日あたりの費用はいくらですか？」人々は通常知りません
- 「一人当たりの 1 日あたりの費用はいくらですか？？」ほとんどの人はそのことすら知りません

時間のコストを知らなければ、人々はどのように設計上の意思決定を行うことができるでしょうか？正確な数値は重要ではありません。より良い意思決定を行うためには、妥当な数であればどれでも構いません。

かつて、私がプロジェクトでこれらの質問をしていたとき、上司がドアの小さな窓から通り過ぎるのが見えました。私はドアを開けて上司にこう言いました。「ボス、この人たちは一日遅れることの代償、プロジェクト全体のコスト、ましてや自分たちの経費のことなど全く理解していません。こんな人たちがどうやってまともな判断を下せるというのでしょうか？」。
彼は「私も知らないが、調べてみるよ！」と言った。
1 時間後、彼は戻ってきて、「1 人 1 日あたり 400 ユーロです」と言った。

プロジェクトのメリットは非常に大きいものでなければなりません。そうでなければ、別のプロジェクトに取り組むべきです。メリットが分からない場合は、プロジェクトのコストの約 10 倍のメリットを想定しておくことをお勧めしています。
この数値を使って、1 日遅れた場合のコストを計算しました。5 人 × 400 ユーロ × 10 = 20,000 ユーロ。これは通常、誰もが想像していたよりもはるかに大きな金額であり、より根拠に基づいた意思決定を行うための良い出発点となります。

3. 時間通りに到着するための 7 つの選択肢（あるいは錯覚）

多くのプロジェクトが納入日遅延に見舞われている現状を踏まえると、納入日を早める方法を知りたいというニーズは当然あるでしょう。プロジェクトを早く終わらせようと人々が試みる方法はいくつかありますが、そのほとんどは直感的には正しいように思えるものの、実際にはうまくいきません。この矛盾から、人々はプロジェクトの遅延は避けられないものだと考えてしまうのです。結局のところ、彼らは最善を尽くし（ほとんどの人がそうするでしょう！）、対策も講じ（直感的に正しい対策を講じたにもかかわらず）、うまくいかなかったのですから。
デミングは「最善を尽くすだけでは十分ではない」と言いました。まず何をすべきかを知り、最善を尽くすためのアプローチを持ち、そのアプローチを継続的に改善する仕組みを整え、それから最善を尽くすべきです。もちろん、実際に効果のある選択肢も存在します。

四つの欺瞞的な選択肢

まず、4 つの欺瞞的な選択肢を排除しましょう。欺瞞的な選択肢は、ほとんどすべてのプロジェクトで適用されているのを目にしますが、効果がなく、むしろ事態を悪化させる場合がほとんどです。人々がそこから学ばず、これらの選択肢を使い続けるのは驚くべきことです。

1. **希望的観測** – 運を天に任せる
   過去のプロジェクトが予想以上に時間がかかったのなら、プロジェクトの進め方を変えなければ、結果は以前と同じままです。今回は予定通りにプロジェクトが進むことをただ願うだけでは何の役にも立ちません。私たちはこれを「オーストリッチング（ダチョウのように砂に頭を突っ込み、現実から目を背けること）」と呼んでいます。マーフィーの法則が再び発動するのをただ待っているのと同じです。

2. **よし、気合で乗り切るぞ**
   時間は足りないけれど、やらなければならない。「よし、やろう！」何も問題が起こらなければ（そんなことはまずあり得ないけれど）、そして（すでに一生懸命やっているけれど）もう少し頑張れば…まあ、それは忘れてしまおう。本当に時間が足りないなら、それは実現しないだろう。

3. **残業** – 自分、顧客、上司を騙す行為
   週 40 時間の労働はすでにかなり疲れるものです。さらに長時間働けば、疲労が増し、ミスが増え、ミスを見つけて「修正」するために余分な時間を費やすことになりますが、その半分は修正できません。本人は一生懸命働いている（working hard）つもりでも、賢く働けて（working smart）いないため、結果的に生産量は減り、品質も低下します。しばらくすると、残業によって生じたミスを修正するためだけに、残業をすることになるでしょう。原則として、残業は絶対にしないようにしましょう。そうすれば、本当に必要な年に 1、2 回だけ残業するエネルギーが残ります。

4. **時間を追加する** – 納期を延期する
   納期を延長することも良い考えではありません。納期が遠ければ遠いほど、プロジェクト遅延の

リスクは高まります。これはパーキンソンの法則[1]と学生症候群[2]によるものです。新たな納期までにはほとんど進捗がなく、プロジェクトの納品がさらに遅れることになります。あらゆる最適化手法を用いても解決できない「9 母」のような状況でない限り、納期を延長するのは良い考えではありません （9 人の母親がいれば、1 か月で赤ちゃんを作れると考える人もいる)。納期までの限られた時間を最大限に活用するのが賢明です。納期が早ければ早いほど、その後の時間的余裕が生まれ、次に何をすべきかを考えるための柔軟性も高まります。

学生プロジェクトの途中で、私は学生たちに、きちんと動作するアプリケーションを完成させてプロジェクトを成功裏に終えられるか尋ねました。すると、1 分以内に 4 つの欺瞞的な選択肢が返ってきました。「いい予感がする！」とある学生は言いました（宿命論的）。「それに、成功させなければならないので、必ず実現させます！」（マッチョ）。「実現させるために、夜も週末も働いています」（残業）。「申し訳ありませんが、先週お約束したものはまだきちんと動作していません。来週には一部を納品します」（納期の変更）。最終的な結果：役に立たないアプリケーション、顧客は満足せず＝失敗！

5. 人員を増やす、リスクの高い選択肢…

典型的なやり方は、より少ない時間で物事を完了させるために、プロジェクトに人員を増やすことです。直感的に、私たちは時間と人員を交換できると感じ、図 3 に示すように、12 人月かかるプロジェクトを 2 人で 6 か月、4 人で 3 か月、または 6 人で 2 か月完了できると考えます。ブルックスは自著「人月の神話」の中で、これは誤謬であり、おとぎ話であり、神話であることを示しています。

ブルックスの法則（1975 年）：
遅れているプロジェクトに
人員を追加すると、
さらに遅れることになる

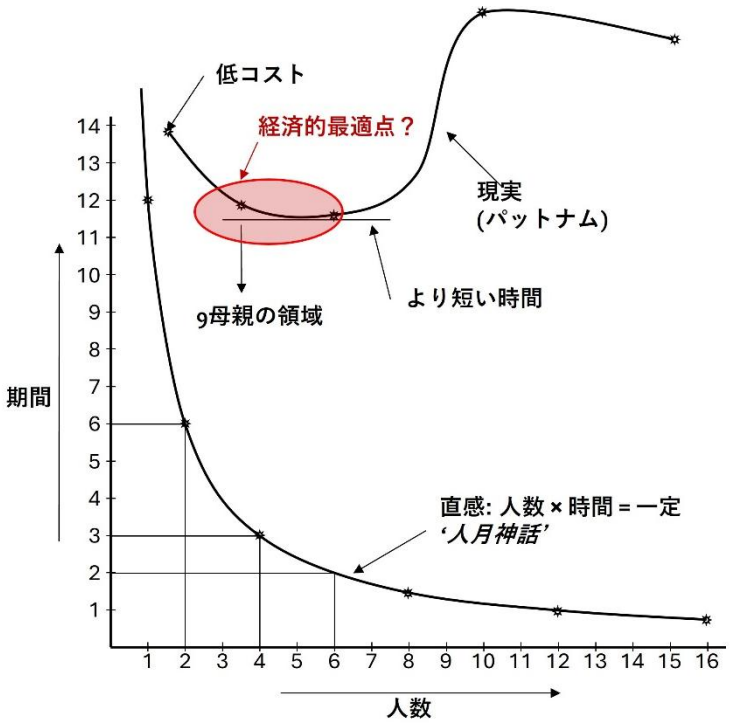


図 3：人月の神話とパトナムの研究結果

ブルックスの法則について初めて聞いたとき、私は、プロジェクトの終盤、つまり時間が迫っている段階で人員を追加すべきではないという意味だと解釈しました。これは、多くのプロジェクトにおいて、遅延は終盤になって初めて気づかれることが多いからであります。
時間がなくなりつつあると、追加の人員がプロジェクトについて学ぶ必要があり、貴重な時間が説明に費やされ、プロジェクトに利用できる時間がさらに減少します。

しかし、その影響はより複雑です。プロジェクト開始後数週間で進捗が遅れたとしても、人員を増やすことで納入日がさらに遅れる可能性があります。これは、人員が増えればコミュニケーションチャネルが増え、管理すべき人数も増

[1] 「仕事は、それを完了するために利用できる時間を埋めるように拡大する」 、あるいは私の訳では「人々は与えられた時間を使う」 。
パーキンソンは次のように述べている。「仕事は時間に対する要求において弾力性があるとすれば、行うべき仕事と、それに割り当てられるスタッフの規模との間に、ほとんど、あるいは全く関係がないことは明らかであります。」

[2] 「学生症候群」：試験直前のプレッシャーを本当に感じた時だけ、できるだけ遅く勉強を始める（E・ゴールドラットの説）。試験に合格するために勉強しなければならなかった時、3 週間前から一生懸命勉強しましたか？それとも、ほとんど試験前夜に勉強しましたか？正直に答えてください。

えるからです。一方で、プロジェクトマネージャーやシステムエンジニア／アーキテクトが管理できる人数には限界があり、その限界を超えるとボトルネックになりかねません。したがって、人員を増やすことは必ずしも効果的な解決策ではなく、非常にリスクの高い行動となる可能性があります。

あるプロジェクトで、実際の作業に当初見積もっていた時間の約 4 倍もの時間がかかっていることが判明しました。どうやら、見積もりの際に、学習やテストといった作業要素を多く見落としていたようです。そこで、生産性を向上させるために、人員を 5 人から 20 人に増やすことを試みました。しかし、生産性は 50%しか向上しませんでした。プロジェクト管理は、直感に反して人員を 20 人から 10 人に減らすことを決断するまでに約 10 ヶ月を要しました。そしてようやく人員削減を行ったところ、残った 10 人の純生産性は 20 人だった時と変わりませんでした。つまり、10 ヶ月もの間、生産性向上に全く貢献しない 10 人分の人件費を払い続けていたことになります。これは非常にコストのかかる試みでしたが、おそらく今後も多くの場所で同様のことが繰り返されるでしょう。

パットナムは、約 500 件のプロジェクトの測定結果からブルックスの法則を裏付けています（図 3）。彼は、プロジェクトを 2 人または 3 人で実施するとプロジェクトコストが最小化されることを発見しました。一方、5 人から 7 人で実施すると、人員が 2 倍になってもプロジェクト期間が 20%短縮されるだけなので、コストは高くなりますが、最短のプロジェクト期間を実現できます。市場投入までの時間は経済的に非常に大きな価値を持つことが多いため、これはおそらく依然として経済的に最適な状態と言えるでしょう。

さらに多くの人員を追加すると、プロジェクトは過剰なコストで長引くことになります。並列化できないクリティカルパスが存在するため、プロジェクト期間は恣意的に短縮できないようです。誰もプロジェクトを完了できない期間を「 9 人の母親の領域」と呼びます。何百人、あるいは何千人もの人が協力して取り組む巨大プロジェクトは、どのように成功できるのでしょうか？多くの場合、成功しません。期待よりも少ない成果物しか得られず、納入日も遅れ、多くのプロジェクトは失敗に終わります。莫大な時間と費用をかけ、通常は大幅な遅延を伴う航空機を設計できる企業は、世界にわずかしか残っていません。ブルックスの法則は自分には関係ないと思っているなら、用心した方がいいでしょう。必ず影響します！ブルックスの法則を回避する唯一の方法は、並列作業が可能で、定期的に結果を適切に同期できる多数の小規模チームで作業することです。私たちはこの法則を無視すべきではありません。それに対処する方法を学ばなければなりません。

6. **時間を節約** – 常に機能対策・手軽な成果（Low-hanging fruit）

幸いなことに、結果に悪影響を与えることなく時間を節約する方法は存在します。ここで紹介する進化型プロジェクト計画アプローチは、これらの手法を統合し、定期的に適用することで、最短時間で最良の解決策を実現します。

時間を節約する方法には、多くの側面があります。

- **何**（なぜ、誰のために）**をするかの効率性を向上させる**
  後になって不要だと判明するようなことはしない。特に目標が明確でない場合、人は必要以上に多くのことをする傾向があるため、不要なことをしない機会は十分にあります。このプロセスを制御するために、ビジネスケース、ステークホルダー管理、継続的な要件管理を使用します（このテキストでは説明されていません）。毎週、実行する前に、何をするか、何をしないかを決定します。これにより、必要なことをより良く実行し、それでも納入日を守るために使用できる時間が大幅に節約されます。
- **業務の効率性を向上させる**：やり方を変える
  これはいくつかの側面で効果を発揮します。
  - **製品**
    適切かつ最も効率的なソリューションの選択。選択されたソリューションは、製品の性能とコスト、そしてプロジェクトの時間とコストを左右するため、単に最初に思いついたソリューションではなく、最適な妥協点であるべきです。私たちは、適切なステークホルダーと短いフィードバックサイクルを用いて、要件と前提条件を確認します。
  - **プロジェクト**
    これまでと同じやり方をすぐに実行するのではなく、まず代替となるより効率的な方法を検

討すれば、おそらく同じことをより短時間で実現できるでしょう。私たちは製品を設計するだけでなく、仕事のやり方も継続的に設計し、再設計しています。

- **継続的な改善と予防**
  物事をより良く行う方法、そして悪い傾向を克服する方法を、積極的に継続的に学びます。PDCA サイクル[3]（デミングサイクルとも呼ばれます）を迅速かつ頻繁に実施し、製品、プロジェクト、プロセスを積極的に改善します。早期レビューを活用して、悪い傾向が成果物にさらに悪影響を及ぼす前に認識し、対処します。また、欠陥ゼロを目指す姿勢を貫きます。なぜなら、それが欠陥ゼロを実現する唯一の方法だからです[4]。

- **物事を正しい順序で行う**
  適切なタイミングで適切な順序で作業を行うことが重要です。お互いを待ったり、間違った順序で実行したタスクのやり直し、同期の問題によって多くの時間が無駄になっています。積極的に同期を取り（www.malotaux.eu/activesync）、作業の順序を設計しましょう。

これらの要素はすべて、大幅な時間短縮につながります。もちろん、これらの時間短縮策を適用することで、たとえやるべきことが利用可能な時間に十分収まると思っていても、より良い結果をより速く得ることができます。予期せぬ問題に対処するために、後で節約した時間が必要になるかもしれません。そうすることで、言い訳をする必要もなく、時間通りに作業を完了できます。最後に最適化するだけでは、最初に失った時間を取り戻すことはできません。また、最後に最適化するということは、そもそも最適化する時間があまり残されていないことを意味します。

新築の建物で左官職人が壁に漆喰を塗っている場面を想像してみてください。次に電気技師がやってきて、電気配線を通すために壁の漆喰に溝を掘ります。左官職人が戻ってきて壁を補修します。その後、配管工がやってきて、水道管を通すために漆喰と電気配線を切断します。電気技師と左官職人が戻ってきて、配線と漆喰を補修します。もし彼らが作業を始める前に計画のタイムラインを作成していたら、作業の順番が簡単に分かり、同じ作業を繰り返すこともなかったでしょう。彼らは愚かではありません。ただ、考えなかっただけなのです。



左官職人が仕事を終えた後、電気工事士がつけた溝が残る

7. **プロジェクトの中止**
   もちろん、投資に対してプラスのリターンが得られないことが早くわかればわかるほど、予算の 3 倍、あるいはそれ以上の費用を費やした後ではなく、プロジェクトを早く中止することができます。

4. 目標

あらゆるプロジェクトの普遍的な目標として、私たちは以下を使用します。

適切な結果を適切なタイミングで提供し、時間をできる限り無駄にしない（＝効率的に）

さらに簡潔に言えば、「**品質を期日通りに届ける**（Quality on Time）」ことです。

より正式には、あらゆるプロジェクトの最上位要件として以下を使用します:

顧客に提供する

- **必要なものは**
  通常、彼らが要求したものとは異な
- **必要なときに**
  彼らが要求したよりも早い場合も遅い場合もあります
- **満足してもらえるように**
  顧客に支払い意欲を持たせる

[3] 第 6 章「Plan・Do・Check・Act（PDCA サイクル）」を参照してください。www.malotaux.eu/pdca
[4] 「ゼロ・ディフェクト」については、www.malotaux.eu/zerodefects を参照してください。

- **それがない場合よりも成功できるように**
  顧客が成功しなければ支払うことはできません。
  以前より明確に成功していると言えないのであれば、なぜ支払う必要があるのでしょうか?
  成功は最終的にはシステムのユーザーによって生み出されることを忘れないでください。私たちのプロジェクトと顧客は、ユーザーが成功を生み出すための条件を提供するだけです。

顧客に提供する
- 必要なものを
- 必要なときに
- 満足してもらえるように
- それがない場合よりも成功できるように

制約されるもの（win-win）
- 顧客が支払える範囲
- 双方にとって利益があり
- かつ満足できる形で提供できるもの
- 妥当な期間内に

制約されるもの（win-win）

- **顧客が支払える範囲**
  顧客が「望むもの」を彼ら自身が支払えない場合、そのまま開発を始めても、結局は失敗に終わる可能性が高いです。
- **双方にとって利益があり、かつ満足できる形で提供できるもの、**
  それは win-win でなければなりません：顧客は王様ですが、私たちは奴隷ではありません
- **妥当な期間内に**
  奇跡が起こるには、より多くの時間がかかります

本来の業務に集中することで、時間の節約につながります。プロジェクトで行うすべての作業はこの最上位の要件を満たすものでなければならず、そうでなければ無駄です。無駄を生み出すために、わざわざ時間を無駄にしたい人などいるでしょうか？

5. 事前省察、先見性、そして予防

もちろん、あなたのプロジェクトは他とは異なりますが、多くのプロジェクトは適切な結果を適切なタイミングで提供できないという問題を抱えています。コブのパラドックス(1995)はこう述べています:「プロジェクトが失敗する理由は分かっている。失敗を防ぐ方法も分かっている。それなのに、なぜプロジェクトは失敗し続けるのか？」どうやら、多くの人は失敗を防ぐ方法を知らないようです。

適切なものを適切なタイミングで届けるために私たちがしなければならないことの多くは、直感に反するものです。直感は、日常的な状況において自動的に反応するように私たちを導きます。もし直感が完璧であれば、私たちの行動も完璧になるでしょう。しかし、私たちの行動すべてが完璧というわけではないので、直感は時に私たちを間違った方向へ導くことがあるようです。直感は非常に強力なメカニズムであり、それに逆らうのは困難です。しかし、直感が時に非生産的な決定を下させ、積極性を阻害することを認識し（そして認める）ことができれば、私たちはそれに対して積極的に行動を起こすことができるのです。

アルバート・アインシュタイン（1879-1955）はこう言ったようだが、ベンジャミン・フランクリン（1706-1790）はもっと前にこう言っていた

狂気とは、同じことを何度も繰り返し、結果が違うことを期待することです

これまでと同じやり方で仕事を続けていたら、同じ過ちを繰り返し、プロジェクトの納入日遅延は依然として続くでしょう。仕事のやり方を変えて初めて、結果は変わるかもしれません。

「後知恵は容易」であるため、私たちはしばしばそれを利用して過去の行動を振り返り、そこから学ぶことができます。例えば、今となっては間違っていた、不必要だった、あるいは無駄だったとわかるようなことを、当時は避けることができたのではないか？あるいは、もっと効率的にできたのではないか？といったことです。しかし、振り返りによって失われた時間を取り戻すことはできません。時間はすでに過ぎ去っており、二度と取り戻すことはできないのです。

先見の明を持つことは容易ではありませんが、先見の明があれば、後々の行動が誤りであったり、無駄であったりする可能性を想像し、それを行わないという決断を下すことができます。また、単に始めて最善を尽くすよりも効率的な方法を想像することもできます。

振り返り（Reflection）は後知恵による学習のためのものです。しかし、学んだことを事前省察（Preflection）として積極的に活用しなければ無意味です。事前振り返りがあってこそ、私たちは未来を予見し、貴重な時間を無駄にすることを防ぐことができる「振り返り（Reflection）」という言葉があるのに、それよりもはるかに重要な「事前省察（Preflection）」という言葉がないのは、不思議ではないでしょうか？

これは、PDCA サイクル（Plan・Do・Check・Act サイクル）またはデミングサイクルで使用されます。

6. Plan-Do-Check-Act

進化型計画の基本の一つは、継続的な学習と改善のための、昔ながらの Plan・Do・Check・Act サイクルサイクル（PDCA サイクル、またはデミングサイクル）です。

- Do（実行）は問題ではない。私たちは常に「する」のです
- Plan （計画）は多かれ少なかれ実行されるが、通常は不十分であります
- Check（評価）と Act（改善）については、忙しすぎて次の Do（実行）に移りたがるため、時間が割かれません。

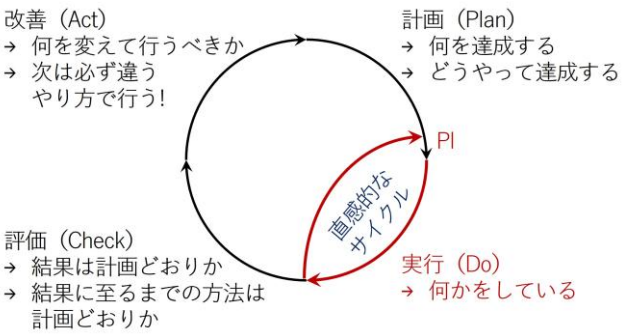


図 4：直感的なサイクル：Pl-Do-Pl-Do

直感とは、過去の経験に基づいて人々が状況に自動的に反応する方法です。私たちの潜在意識は、状況に対処する方法を提案し、私たちは以前と同じようにすぐにそれを実行します。これを直感サイクル、または「Pl-Do-Pl-Do」と呼び、図 4 に示します。明確な「Plan」がないため、単に「Pl」と表記します。

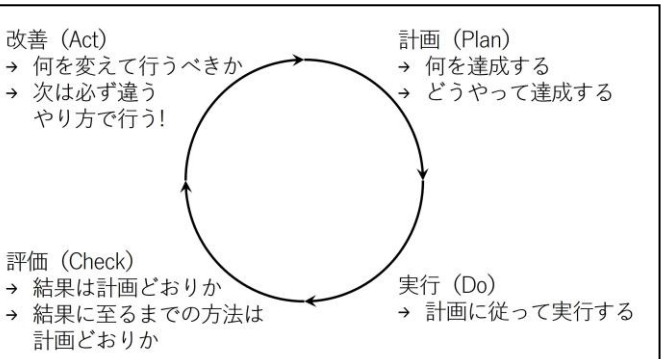


図 5：PDCA サイクルまたはデミングサイクル

Pl-Do サイクルに従う代わりに、図 5 に示すように、Plan-Do-Check-Act サイクルをより効果的に積極的に使用する方法を見ていきましょう。

- まず Plan を立てますが、それは 2 つの側面から成り立っています。
  - 達成目標（製品）
  - それを最も効果的に実現する*方法*（プロジェクト）
- 次に、計画通りに実行（Do）します。これが最初の落とし穴です。なぜなら、計画は実行可能でなければならず、計画に従わなければならないからです。
  これが事実だと仮定しましょう：
- Check または Study フェーズに進み、以下の点を分析します。
  - 達成した成果は計画通りか？有効性（effectiveness）を考える
  - 方法は計画通りだったか？　効率性（efficiency）を考慮して
  - もしそうなら：次回はもっとうまくできるだろうか？
  - いいえの場合：次回はもっとうまくできるでしょうか？
- そして、Act フェーズで、次回はどのように行動を変えるかを決定します。なぜなら、次回も何も変えなければ、結果は同じになるからです。

私たちは一度に非常に小さなステップを踏むことで、何が明らかに優れていて何がそうでないかを学びます。こうした小さなステップを迅速かつ頻繁に行うことで、私たちが何をしているか（製品）、どのように行っているか（プロジェクト）、そしてそれらすべてをどのように組織しているか（プロセス）を迅速かつ継続的に改善していきます。Act（改善）フェーズでは、私たちの作業内容やその進め方に変化（ミューテーション）を取り入れるため、これを「進化的（Evolutionary）アプローチ」と呼んでいます。

また、“evolutionary” は非常に長い単語なので、既知の代替手段よりもうまく機能するものすべてに対して、『Evo』というラベルを使います。

人々は何がうまくいっていないかをよく理解していて、「そうすべきだけど…」と簡単に言う。「経営陣が許可しない」「それは不可能だ」「以前に試したことがある」など。
こうした「そうだけど…」という言い訳を聞いたら、私はたいていこう提案する。「あなたは Check フェーズで立ち止まっています。それが本当に問題なら、 「私たちはそれについてどうすべきか、そしてどうできるのか？」と問いかけると、相手はたいていすぐに解決策を提案してくれる。私たちが直面している問題は、本当の問題ではない。本当の問題は、それに対して何か行動を起こさなければ

ならないということだ。Check フェーズにとどまるのではなく、Act フェーズへと進むのだ。 「私たちはそれについてどうするつもりなのか？」と。できないことを嘆くのではなく、できることを考える。そうすれば、思っていたよりもずっと多くのことが可能だと気づくだろう。

7. 迅速なフィードバックのための評価

多くの組織では、プロジェクトの終了時、またはプロジェクトの各段階の終了時にプロジェクト評価を実施することを義務付けています。これを「レトロスペクティブ（振り返り）」、あるいは「ポストモーテム（事後分析）」と呼ぶ組織もあります。しかし、実際に評価を実施するプロジェクトはごくわずかです。なぜなら、こうした評価は必ずしも成果向上につながるとは限らないと考えているからです。なぜでしょうか？

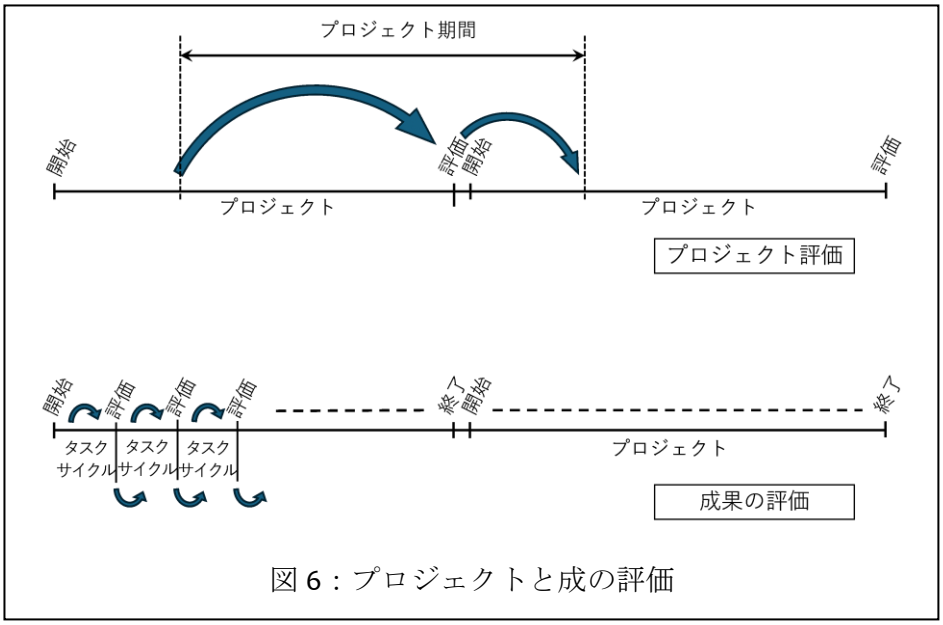


図 6：プロジェクトと成の評価

1 年間のプロジェクトを考えてみましょう（図 6）。人々は、何がうまくいかなかったのか、何がうまくいったのか（多くのことは偶然うまくいきます）、そしてその理由を、1 年前まで遡って評価しなければなりません。プロジェクト開始から 3 か月後に何かが起こった場合、9 か月後にそれを思い出し、どうすればもっとうまくできるかを考えなければなりません。次のプロジェクト開始から 3 か月後には、何か違うことを、できればもっとうまくやろうとしていたことを思い出さなければなりません。結果として、前のプロジェクトは、プロジェクト終了後に初めて物事をより良くする方法を考え出したため、何のメリットも得られず、次のプロジェクトも、評価の決定をタイムリーに思い出すことができず、まだ直感に染み付いていないため、メリットが得られません。

原理は正しい：PDCA サイクルと同様に、まずプロジェクトを計画し、次に実行し、その後、改善できた点を確認し、改善フェーズで次回はどのように行うかを決定する。しかし、実際にはうまくいかない。時間枠は人間の能力に合わせて調整されていません。
原理的には正しいこのプロセスを、実際に私たちにとって有効なものにするにはどうすればよいのだろうか？

毎週評価を行うのはどうでしょうか？先週の出来事を覚えておくだけでも大変です。4 日前の夕食に何を食べたか覚えていますか？ほとんどの人は覚えていません（まあ、料理人でもない限り、覚えておく必要はありませんが）。

毎週評価を行うようになれば、 PDCA サイクルが効果を発揮し始めます。

- 過去 1 週間の良かった点と悪かった点を意識的に思い出すようにすれば、週末までに評価することはそれほど難しくありません。問題が発生した際にホワイトボードや付箋を使って書き留め、後で評価することを忘れないようにする人もいます。また、すぐに改善できることを繰り返さないように、すぐに評価する人もいます。しばらくすると、それは生活の一部となるのです。
- 1 週間ではあまり多くのことは起こらないので、評価に長い時間をかける必要はありません（どうしても必要な場合を除いて）。
- ほとんどの作業は 1 週間以上かかるため、改善策を翌週にすぐに試すことができます。1 週間後には、結果が改善したかどうかを確認できます。改善した場合は、さらに改善できるかどうか、改善すべきかどうかを検討します。改善しなかった場合は、さらに改善できるかどうかを検討します。
- 実際に翌週にはやり方を変えることで、新しい働き方がすぐに身につき、後で再び必要になったときに直感的に思い出せるようになるのです。

これで評価方法は人間の能力とニーズに合わせて調整され、私たちにとって有益なものとなり始めた。

8. 進化型計画

進化型（Evo）計画は、効率性と有効性を継続的に向上させ、誰も必要としないものに時間を浪費せず、可能な限り最短時間でより成功裏に事業を完了させることを目的としています。

物事を管理しやすくするために、私たちは業務をいくつかのレベルで組織化しています。

- 週ごとの**タスクサイクルで**、作業方法を最適化し、効率性を向上させます。
- 隔週（2 週間ごと）の**納入サイクルで**、納入内容を最適化し、効果を高めます。
- **タイムライン**：
  - 私たちが今のやり方で働き続けたらどうなるだろうか
  - 期待通りの結果を期日までに得られない場合の対処法、
    必ず期日までに完了させること、言い訳は不要

これらの要素が組み合わさることで、個々の業務がより効果的、効率的、かつ予測可能になり、結果としてプロジェクト全体の効果、効率性、予測可能性も向上する。

1. タスクサイクル

タスクサイクルでは、作業を整理します。正しいことを、正しい順序で、適切な詳細レベルで行っているかを確認します。見積もり、計画、追跡能力を最適化し、将来をより正確に予測します。優先度の高いタスクのみを選択し、優先度の低いタスクや未定義のタスクは決して行いません。これにより、効率（＝最短時間での有効性）が向上します。

実際には、利用可能な時間の 3 分の 2 を計画に充て、残りの 3 分の 1 で、互いに助け合ったり、プロジェクト会議に出席したり、その他やらなければならない様々な用事をこなしたりといった、ちょっとした用事を済ませます。もし利用可能な時間の 100%を計画に充ててしまうと、結局は他の用事もすべてこなさなければならず、計画したことを達成できないまま終わってしまうのです（心当たりはありませんか？）。

タスクサイクルは通常 1 週間ですが、特別な場合はそれより短くなることもあります。各タスクサイクルでは、何をするのが最も重要か、利用可能な時間はどれくらいか、それを適切に行うのにどれくらいの時間が必要か（適切に行うとはどういうことかを定義します）、そして利用可能な時間で何ができるかを決定します。また、必要でないか、または行う時間がないため、このサイクルでは行わ*ないことも決定します。こうすることで、*できることに全力を注ぐことができ、よりリラックスして生産性を高めることができます。マネージャーの中には、利用可能な総時間の 2/3 しか計画しないと、人々があまり仕事をしないのではないかと心配する人もいます。実際には、人々はより多くの仕事をしていることがわかります。これは、直感に反する典型的な例です。

2. 納入サイクル

デリバリーサイクルでは、選定されたステークホルダーに提供する成果物を整理します。適切なものを、適切な順序で、適切な詳細レベルで提供できているかを確認します。要件を最適化し、前提条件を検証することで、効率性を向上させます。

デリバリーサイクルは通常 2 週間以内です。Evo の初心者は、ほぼ例外なく、短いデリバリーサイクルに苦労します。彼らは、それが不可能だと考えます。しかし実際には、例外なく*常に*可能です。ただ、少し練習が必要なだけです。サイクルが短い重要な理由の 1 つは、後で不必要だったり、間違っていたりして貴重な時間を無駄にする可能性のある多くの作業を行う前に、私たち（およびステークホルダー）の前提を検証したいからです。短いデリバリーサイクルは、リスクとコストを最小限に抑えるのに役立ちます。

デリバリーに関するよくある誤解は、常にユーザーや顧客にデリバリーしなければならないというものです。しかし実際には、ユーザーや顧客、私たち自身、あるいはその間のあらゆるステークホルダーなど、あらゆるステークホルダーにデリバリーすることができます。これにより、デリバリーの定義が容易になります。ただし、最適なフィードバックを得るためには、常にデリバリーを最適化する必要があります。つまり、何が正しく行われているか、何がまだ間違っているかを確認する必要があります。したがって、すべてのデリバリーについて、「*何を*、*誰に*、なぜデリバリーすべきか?」という質問を自問します。

3. タイムライン

多くのプロジェクトでは、やらなければならないと思われる作業はすべて細かく分割され（これをワークパッケージと呼ぶ人もいますが、私たちは「作業のまとまり（チャンク）」と呼んでいます）、各部分の見積もりが行われ、それらの見積もりを合計して、作業に必要な労力の見積もりを算出します。次に、この見積もりを利用可能な人員で割り、作業期間の見積もりを算出します。相互依存関係を評価し、予備費を加えた後、この期間がプロジェクトの期間として提示されます（図 7）。

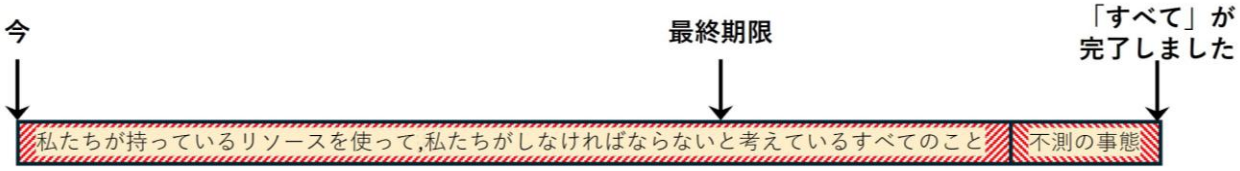


図 7：標準的なアプローチ：必要なことは何でもするが、多くの場合、それでは手遅れになる

問題は、多くの場合、必要な期限がそれよりも早いということです。予定納期と見積納期のずれは、さらなる議論を招き、時間の浪費につながります。一方、要求される納期は変わらないため、プロジェクトに使える時間はどんどん少なくなっていきます。

納入日は他のすべての要件と同様に重要な要件であるため、他のすべての要件と同様に真剣に扱われなければなりません。タイムラインでは、納入日を真剣に扱い、必ず期日を守ります。納期を守ることが本当に不可能な場合は、最終週まで待ってからお客様に伝えるようなことはしません。もし期日を守ることが不可能な場合は、事実に基づいて、非常に早い段階でご説明します。なぜなら、本当に不可能であれば、もっと早い段階で把握しているからです。納入日を守ることが可能であれば、あらゆる時間短縮技術を駆使して、いつ何を納品できるかを最適化し、納品します。

タイムラインは、プログラム、プロジェクト、サブプロジェクト、納品物、さらにはタスクサイクルなど、あらゆる規模で使用できます。手法は常に同じです。まず、達成する必要があると思われることを見積もります、次に顧客やその他の適切なステークホルダーとタイムラインについて話し合い、します（図 8）。

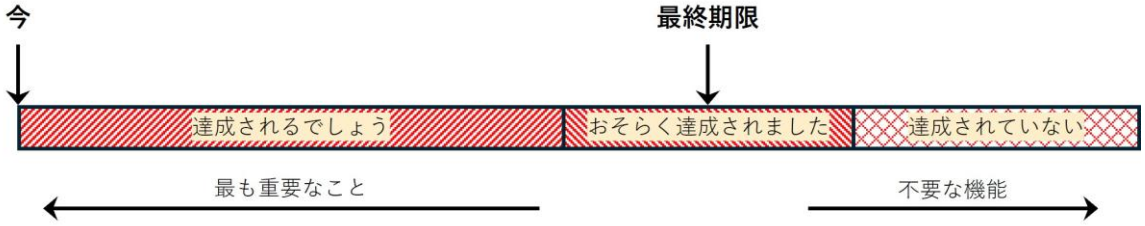


図 8：基本タイムライン：確実に実施されること、実施されないこと、そして現状のまま作業を続けた場合に実施される可能性のあること

説明方法：

- 最終期限には必ず行われること
- 必ず行われないこと
- 何が起こり得たか（結局のところ、予測は正確な科学ではない）

確実に起こるであろう事柄が成功を保証するのに十分でないならば、時間と金の無駄を避けるためにも、すぐに中止するのが最善だろう。計画は厳密な優先順位に従って行うので、最終期限までに少なくとも最も重要なことは完了しているはずです。市場投入までの時間が重要な場合、顧客は余計な機能には関心を持ちません。優先順位は非常に動的に変化する可能性があるため、何をいつ行うかの順序を常に再検討する必要があります。80%の完成でもきちんと終わっている方が、100%の出来でも未完成よりいい。

4. ホライズン（計画の見通し期間）を設定する

プロジェクト全体の所要期間が 10 週間を超える場合、タイムライン上で約 10 週間をホライズンとして設定します。これは、それ以上の期間を現実的に管理することが困難であるためです。10 週間という期間は、管理可能な範囲と、最適なフィードバックを得るために成果物の提出順序を最適化するのに十分な長さとの間で、良い妥協点となります。ホライズンの先のことも忘れてはいませんが、当面は目の前の 10 週間に集中します。

5. 納入サイクル

今後 10 週間にわたり、それぞれ最大 2 週間の納入サイクルを複数回計画し（図 9）、その際に「何を、誰に、なぜ納入するのか？」という問いを自らに投げかけます。

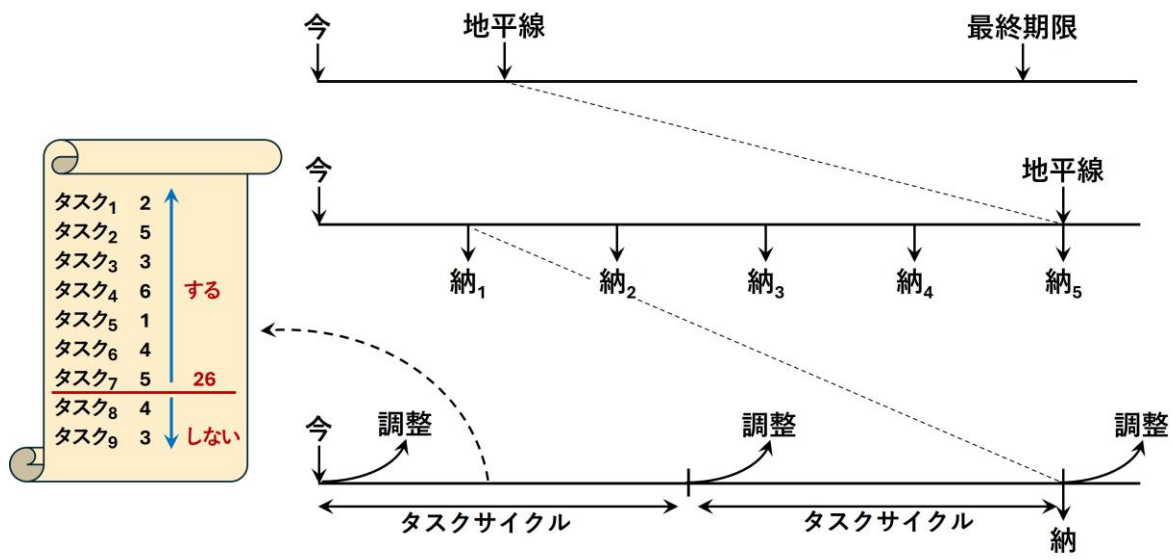


図 9：タイムライン：まず、何が起こるべきかをトップダウンで分析し、次にボトムアップで調整して実際に何が起こるかを把握し、そして、その結果に対する失望感を、それに対処するための行動へと繋げます。

納品はステークホルダーからのフィードバックを得るためのものです。要件に対する私たち（そして彼ら）の認識は完璧ではなく、多くの前提が間違っている可能性があることを謙虚に認めます。そのため、コミュニケーションとフィードバックが必要です。私たちは、 ステークホルダーが熱心に待っていなければ、十分な速さでフィードバックを得ることができません。適切なステークホルダーが熱心に待っていない場合は、関心がないか、他のステークホルダーのために活動した方が良いか。今すぐフィードバックが必要な場合は、いわゆる「ジューシービット」を提供することで、彼らの期待感を高めます。ジューシービットに高い優先順位を与えるのはなぜでしょうか？適切なフィードバックが得られなければ、誤った前提に基づいて作業を進めてしまい、結果として物事を間違った方向に進め、後々遅延を招く可能性があります。したがって、ステークホルダーにジューシービットを提供して、彼らが待ち望むようにすることで、私たちが切実に必要としているフィードバックを得られるのであれば、これは高い優先順位を持つべきでしょう。

6. タスクサイクル

作業をデリバリー（それぞれがホライズンでもある）に分割したら、最初のいくつかのデリバリーに集中し、これらのデリバリーを作成するために実際に行う必要がある作業を定義します。この作業は、1 週間のタスクサイクルに編成されます。タスクサイクルでは、正味の作業時間で見積もられたタスクを定義します。作業は計画可能な作業時間で計画され、デフォルトでは総利用可能時間の 2/3（40 時間週の場合は約 26 時間）となり、メール、電話、計画、小さな割り込みなど、計画不可能なプロジェクト活動はすべて、総利用可能時間の残りの 1/3 に制限されます。各自が何をするか、どの作業を共同で行うか、作業を適切に行うためにどれくらいの時間が必要か、過負荷にならないようにするか、作業の同期を取って全体の期間を最小限に抑えるかを決定します。

7. 上から下へ、下から上へ （トップダウンとボトムアップ）

最初の週にやらなければならない作業を見積もったことで、タイムライン上で見積もりを調整するための最初の指標が得られました。最初の週のタスクがその週にやるべきことの約半分しか達成できないとすれば、現状のまま作業を続けると、つまり何も対策を講じなければ、プロジェクトには 2 倍の時間がかかると推測できます。最初の週の見積もりデータは最初は弱い証拠に見えるかもしれませんが、すでに私たちの見積もりが楽観的すぎる可能性を示しています。この証拠を無視するのは危険です。「一時的な原因」に関する言い訳はこれまで何度も聞いてきました。その後、必ず別の「一時的な原因」が現れたものです。

1 週間後、最初の週の実際の結果が判明すれば、プロジェクトが実際にどれくらいの期間を要するのかをより正確に予測でき、それに応じて規模を調整するためのより具体的な数値が得られます。毎週、トップダウンとボトムアップの両方のアプローチを用いて情報を収集し続け、それに基づいて、最終期限（Fatal Date）までに、あるいはそれよりも早く達成できるであろうことについての理解を調整します。このようにして、タイムライン作成プロセスは遅延のリスクに関する早期警告を提供します。警告を早く受け取るほど、それに対処するための時間的余裕が生まれます。実際のタイムライン作成プロセスはもう少し複雑になる場合もありますが、基本的な流れは上記のとおりです。

失敗は許されません。潜在的な失敗を示す警告信号を早く受け取るほど、それを防ぐための対策を早く開始できます。

9. Evo の実践例：タスクサイクル計画

これ以上長くなるのを避けるため、プロジェクトを進化型計画に移行するための最初のステップについて、もう少し詳しく説明しましょう。

実際、TaskCycle プランニングの結果は、プロジェクトに関わる全員の ToDo リストになります。これは、私たちが普段作成しているような ToDo リストとは異なります。私たちの ToDo リストは、作業に取りかかる前に、実現可能性（利用可能な時間に収まるか？）、優先順位（他の何よりも重要か？）、同期性（作業の順序や他の人の作業と合致するか？）、その他あらゆる有用な側面についてチェックされます。作業に取りかかる前に、何をするか、何をしないかを最適化できます。作業が終わってからでは、事前にチェックしなかったために時間を無駄にしたと嘆くしかありません。計画したことは実行可能なので、最初の週からストレスがなくなると言われています。集中して作業に取り組み、なぜこの作業をするのか、なぜ他の作業をしないのかが明確になります。生産性が飛躍的に向上します。

週ごとのタスクサイクルの開始時に、私たちは以下のことを行います。

1. **まず、このタスクサイクルで何を達成すべきかを決定します。**
   明確な目標を持つことで、本当に必要なことと（同様に重要なこととして）必要 *でないこと*に集中できます。

2. **このタスクサイクル中にこのプロジェクトに利用可能な作業時間の合計を決定します。**
   何をする必要があるかを考える前に、まず利用可能な時間を明確にしましょう。必要な作業に要する時間が利用可能な時間より長い場合、**実際に持っているよりも多くの時間を割り当ててしまい、**自分を欺き、計画どおりに目標を達成できないという結果になりがちです。原則は、言い訳をせずに計画したことを完了することです。利用可能な総時間から、歯医者やトレーニングなど、すでに予定されているすべての時間を差し引きます。

3. **この総利用可能時間を以下のように分割します。**
   - 利用可能な計画可能なの時間（デフォルトは総利用可能時間の約 2/3）
   - 利用可能な計画不可能なの時間（デフォルト：総利用可能時間の約 1/3）

   週 40 時間労働の場合、通常は計画可能な時間を 26 時間、計画不可能な時間を 14 時間とします。
   多くのプロジェクトにおいて、これは現実的な数値であることが証明されています。もしそうでない場合は、ご自身にとって適切な別の数値を使用してください。まずは 26 時間から始めて、様子を見てください。
   計画しなければ実行されないタスクのみを計画します。計画に含まれていれば、タスクは必ず完了します。計画段階で「完全に完了する」とはどういうことかを定義し、それに基づいて必要な時間

を算出します。計画段階で「完全に完了」の意味を定義し、その定義に基づいて所要時間を見積もります。

利用可能な時間すべてを計画に組み込もうとすると、毎週必ず失敗してしまうでしょう。なぜなら、トイレに行く、コーヒーを飲む、互いに助け合う、計画を立てるなど、計画外の行動は必ず起こるからです。計画外の行動には、利用可能な時間の最大 1/3 の時間枠のみを割り当て、残りの 2/3 は計画しなければ起こらないような行動に充てられるようにしています。

4. **このサイクルのタスクを定義する**

   *現時点*で最も重要なタスクを見つけ、目標達成に貢献することに集中してください。緊急性の低いタスクに時間を費やさないでください。現在のタスクから得られる情報に基づいて、今後のタスクの定義が変わる可能性があるため、あまり先の計画を立てすぎないでください。タスクで取り組むべき内容を明確にするために、成果物の定義を活用してください。

5. **各タスクを完全に完了するために必要な作業時間を推定する**

   私たちは常に正味の作業時間を見積もっています。日数で見積もると、通常はリードタイム（タスクの開始から完了までの時間）が算出されます。時間で見積もると、作業時間、つまりタスクを完了するために必要な正味の時間をすぐに算出できるようになります。作業時間とリードタイムを分けて考える理由は、変動の原因が異なるからです。作業時間の見積もりが間違っている場合は、複雑性の評価の問題です。計画よりも時間が足りない場合は、時間管理の問題です。これらを分けて考えることで、私たちは学ぶことができます。

   タスクの所要時間を決めることができるのは、実際にタスクを実行する本人だけです。他の人は、見積もり担当者の心理に影響を与えるため、ヒントを与えることさえ許されません。見積もりに同意できない場合は、見積もり時間そのものではなく、タスクの内容（またはあなたが内容と認識しているもの）に異議を唱えることができます。最終的に、タスクの要件について合意が得られたら、実行者がどれだけの時間が必要かを決定します。そうでなければ、成功へのコミットメントが生まれません。コミットメントがなければ、たとえ失敗したとしても、苦痛を感じることはありません。苦痛がなければ、何も学べないのです。

6. **約 6 時間以上のタスクをより小さなタスクに分割する**

   作業を管理しやすい部分に分割します。見積もりは厳密な科学ではないため、見積もりには多少のばらつきが生じます。正確な見積もり時間に縛られることはありません。私たちが縛られるのは結果だけです。つまり、週末までに約束した作業がすべて完了することです。推定値のわずかな違いは気にしないでください。複数のタスクがあり、*平均*見積もり時間が十分に正確であれば、ばらつきは相殺されます。26 時間かかるような大規模なタスクの場合は、見積もりがより難しくなり、平均化のトリックももはや役に立ちません。それに、1 日にできる作業は 6 時間程度が限界です（利用可能な時間の 2/3！）。

7. **計画可能な時間を最も重要なタスクで埋める**

   優先順位の低いタスクは絶対に選ばないでください。計画した時間は常に正確に使いましょう。常に、計画可能な時間を正確に使い切るようにしましょう。時間以上に計画を立てても、実行できないことは既に分かっています。計画していない時間は、いずれ消えてしまいます。

8. **これらが本当に最も重要なタスクであることを確認し、その作業が実行可能かつ実行されると確信してください。**

   - 疑念はあなたの決意を損なうので、必ず実行できるようにしてください
   - このサイクルにおけるタスクリストを受け入れるということは、自分自身とチームに対する責任を受け入れることを意味するということを認識してください。サイクル終了時には、これらのタスクは完全に完了しているはずです。言い訳は一切不要です。

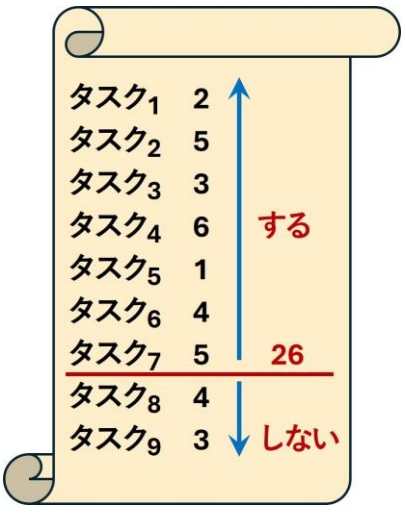


図 10：タスクリスト

この時点で、完了するタスクのリストが作成されます（図 10）。他のタスクが完了*しない*という結果を受け入れられない場合は、何らかの対策を講じてください。例えば、次のようなことができます。

- 優先順位を再考して。
- 追加のサポートを依頼し、一部のタスクを他の人に任せましょう。ただし、タスクを他の人に任せるには時間がかかる場合があることを覚えておいてください。やはり、この時間を考慮して計画を立てる必要があります。
- 他に選択肢がないなら、現実を受け入れましょう。あり得ないことが起こることを期待しても、避けられない事態を先延ばしにするだけです。行動を起こすのが遅くなればなるほど、残された時間は少なくなります。現実から目を背けてはいけません。Evo では、砂の中から頭を出して、積極的に課題に立ち向かいます

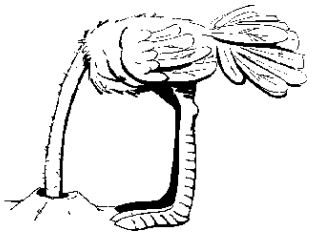


図 11：砂に頭を突っ込んで現実から目を背けるのは危険です

私は 3 人チームを監督していました。そのうちの一人が、翌週に 60 時間働く計画を立てた。私は彼に、週末までに仕事を終えるのはほぼ不可能だろうと伝えました。
「ええ、でも、やらなきゃいけないんです！」
私は、毎週そんなに長時間働くことで彼が病気になるのを同僚は望まないだろうと提案しました。
彼らはうなずきました。
長い議論の後、彼は折れて、（利用可能な総労働時間 40 時間を基に）その週の労働時間を 26 時間と計画しました。数週間後、彼は私にこう言いました。「ニルス、アドバイスをありがとう。あんなに長時間働いていたなんて、私は正気じゃなかった。今は毎週 26 時間の純労働時間を計画していて、以前よりもずっと多くの仕事をこなせています。」
パーキンソンの法則を覚えておきましょう！

9.1. 次のタスクサイクルを計画する際は、以下の手順に従ってください：Check, Act, Plan

9. 計画された作業はすべて完了したか？タスクが完了しなかった場合、分析します。

- 作業に時間を費やしたにもかかわらず、作業が完了しなかったのでしょうか？
  これは、*労力見積もり*の問題です。原因を考察し、次回はより正確な見積もりができるように学びましょう。作業を始める前に何を*考えていましたか*？そして、実際に作業を終えた後、何が分かったでしょうか？
- 作業に
  時間を費やさなかったため、作業が完了しなかったのでしょう*か？*これは*時間管理*の問題です
  - 気が散ったり、邪魔が入ったりしすぎる
  - 他の（見積もりが不十分な？）タスクに時間をかけすぎている
  - 計画外の作業に時間をかけすぎている － 計画外：行わないか、割り込みプロセスを使用してください www.malotaux.eu/interrupts

  原因について話し合い、時間管理の習慣をどのように変えるかを決めましょう。注：見積もり時間と実際の時間を比較して学習した後、これらのデータは学習のために消費され、その価値は失われます。

最初の数回のサイクルではすべてのタスクが完了しないことは想定内ですが、そこから得られる学びは、数週間以内に計画通りにすべてのタスクサイクルを完了するのに役立つはずです。

10. 余波に対処した後、未完了のタスクを完了させる。

- 「失敗」の失望感を、次回に備えるための直感力に蓄えましょう。これがコミットメントが非常に重要な理由です。コミットメントがあってこそ、私たちは真の失望感を味わうことができるのです。より良い予測を立てるためには、適切な心理学を用いて直感力を適切に養う必要があります。これは予測そのもののためではなく、予測精度を高め、成功へと導くためです。
- やり残した作業を完了させるための新しいタスクを定義してください。これらのタスクを候補タスクリストに追加してください。適切な時期にリストに表示されます。次のサイクルですぐ

に表示されないのは、より重要な作業があるためです。その場合は、適切なタイミングで作業を中断したと考えられます。
- 結果を受け止めた後、タスクの完了を宣言してください。過去には何もできないため、このタスクにはもう取り組むことができません。

11. **それでは、次のサイクルに向けたタスクの計画を続けましょう。**
    次のタスクサイクルを計画する際には、学んだことを必ず活用してください！

9.2. 週ごとの 3 ステッププロセス

多くのプロジェクトでは、毎週チームミーティングが行われています。私たちはこれらのミーティングにいくつかの欠点があることに気づき、得られた経験に基づいて、最終的に週ごとの 3 ステッププロセスにたどり着きました。このプロセスは、Evo プランニングの成功に不可欠な要素となっています。このプロセスにより、Evo プランニングの準備に費やす時間を最小限に抑え、最適化することができます。同時に、プロジェクト員間のコミュニケーションも大幅に向上します。

手順は以下のとおりです。

1. **個人の準備**
   この段階では、各チームメンバーはそれぞれが単独でできることを行います。
   - 現在のタスクを完了する
   - 次にどの成果に貢献すべきかを判断する
   - それらの目標達成に貢献する最も重要なタスクを特定する
   - これらのタスクに必要な時間を見積もってください
   - 来週、プロジェクトにどれくらいの時間が使えるかを判断してください

   プロジェクトマネージャー/システム エンジニア/アーキテクト/コーチ/ピア（次を参照）は、最も重要だと思うタスク、これらのタスクにかかる時間 （各タスクの内容に対する自身の認識と個人の能力に基づく)、およびチーム内の誰かとどの程度の協力が必要になると予想されるかをチームのために準備します。
2. **1 対 1: プロジェクト マネージャー/システム エンジニア/アーキテクト/コーチ/ピアによる調整とコーチング**
   このステップでは、各チームメンバーがコーチまたはチームメイトと個別に（1 対 1 で）面談を行います。この面談では、各自の準備結果について調整を行います。
   - 私たちは先週起こったことを観察します。まだ意図を達成できていない人々への支援を行います
   - 個人とプロジェクトマネジメント[5]が重要だと考えるタスクを比較します。意見の相違がある場合は、合意に至るまで話し合います
   - 見積もりに基づいて、これらのタスクすべてを完了できるかどうかの実現可能性を確認します
   - 次のサイクルに向けたタスクセットに満足するまで繰り返し作業を行い、実際のコミットメントを確認します。これで、次のサイクルに向けた作業パッケージが完成しました

   メモ帳、スプレッドシート、または Web ベースのツール（図 12 など）を使用します。
   対面での会議、たとえ 1 対 1 のミーティングであっても、全員が同じものを見て、同じことを話していることを確認するためにプロジェクターを使用します。各自がメモを取ると、それぞれ違う内容を書き込んでしまいます。

   他の人はあなたが書いたものを見ることができないため、あなたが書いた内容について疑問があっても、訂正することができません。さらに、メモを取っている間は、話されている内容を聞き逃してしまう可能性があります。

   各自でタスクの説明と見積もりを書き留めます。そして、書いた内容をどのように調整し、その意味を真に理解していくかを見ていきます。これは、参加者の責任感を大きく高めます。他の参加者は、書かれた内容が自分の頭の中にある内容と一致しているかどうかを観察し、話し合います

   開発者、プロジェクトマネージャー、システムエンジニア／アーキテクトは皆、こうした 1 対 1 のミーティングが Evo Planning のアプローチにおける最も強力な要素の一つであると口を揃えて言

[5] アジャイル開発者は一般的に「プロジェクト」や「プロジェクトマネージャー」といった用語を嫌います。「プロジェクト」を「仕事」に、「プロジェクトマネージャー」を「チームメンバー」に置き換えることができます☺。

います。そして、チームミーティングでは、これまで以上に個々のメンバーの実際の作業状況をより明確に把握できるようになったのです。

3. **チームミーティング:**
   チームとのすり合わせ（同期）
   このステップでは、すべての 1 対 1 の面談が終了した後、チーム全体でミーティングを行います。このミーティングでは、全員の協力が本当に必要なことを行います。

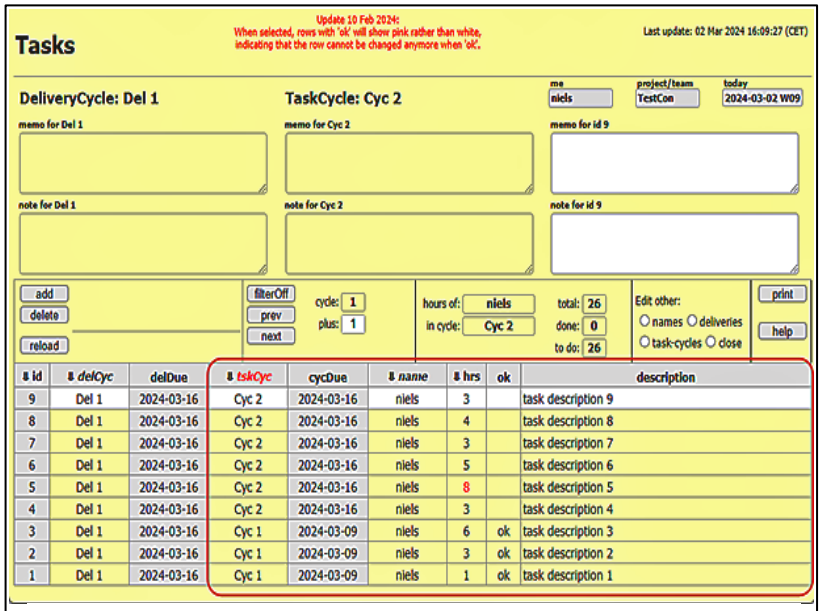


図 12：Web ベースの Evo タスク管理ツール赤い枠で囲まれた部分が本質的な機能です。その他の部分は「装飾」であり、より良い作業に役立つ場合にのみ使用してください

   - プロジェクター画面または共有画面にタスクが表示されたら、参加者は来週の自分のタスクを声に出して読み上げてください。これにより、いわゆる相乗効果が生まれます。例えば、「それを実行するなら、話し合う必要があります…」とか、「それはできません、なぜなら…」といったように、何かを見落としていたことに気づくのです。こうして、どう対処するかを話し合い、それに応じて計画を変更することができます。メリットは、計画を一緒に作成する必要がなくなり、調整するだけで済むことです。これにより、時間を節約できます。また、作業に取りかかる前に非常に効率的なコミュニケーションが大幅に増加し、互いに協力し合い、間違ったことに時間を費やすことを避けることができます。
   - 1 対 1 のミーティング中にグループにとって重要な情報が出てきた場合、それをチームミーティングでの議論のために保留することができます。従来のチームミーティングでは、最初に出てきた話題に多くの時間を費やしてしまい、後から出てくる可能性のあるより重要な話題に時間を割くことができなくなることがよくあります。Evo のチームミーティングでは、皆で話し合うべき最も重要な話題を事前に選択します
   - 学ぶため、そして交流するため

数週間後には、計画プロセス全体にかかる時間は一人あたり約 1 時間になりました。準備に 20 分、1 対 1 のミーティングに 20 分、チームミーティングに 20 分です。以前より話し合う時間が減ったのでしょうか？いいえ、そうではなく、適切なことを効果的かつ効率的に話し合えるようになっただけです。もちろん、プロジェクトマネージャーやシステムエンジニアはチーム全体と面談するためにもっと時間が必要になるかもしれませんが、ここで彼らがやっていることは、本来ずっとやるべきだったことなのです。

先日、あるシステムエンジニアにこう尋ねました。「これだけ計画やスケジュール調整をしていると、システム開発そのものの監督といった、本来のシステムエンジニアとしての業務に十分な時間を確保できるのでしょうか？」すると、彼はこう答えました。「はい。問題ありません。」

10. 結論とご招待

私は、進化型計画アプローチのいくつかの理由と背景、原則、そしてその一要素について説明しようと試みました。このアプローチは、コミュニケーションを強化し、問題が発生する前に未然に防ぎ、業務の有効性と効率性を向上させ、生産性と予測可能性の向上につながります。

これはあなたをジレンマに陥れるかもしれません。プロジェクトをより生産的で予測可能なものにできると分かった今、それを実行しない言い訳はもうありません。もしかしたら、この文章の最後にではなく、冒頭で警告しておくべきだったかもしれません☺。

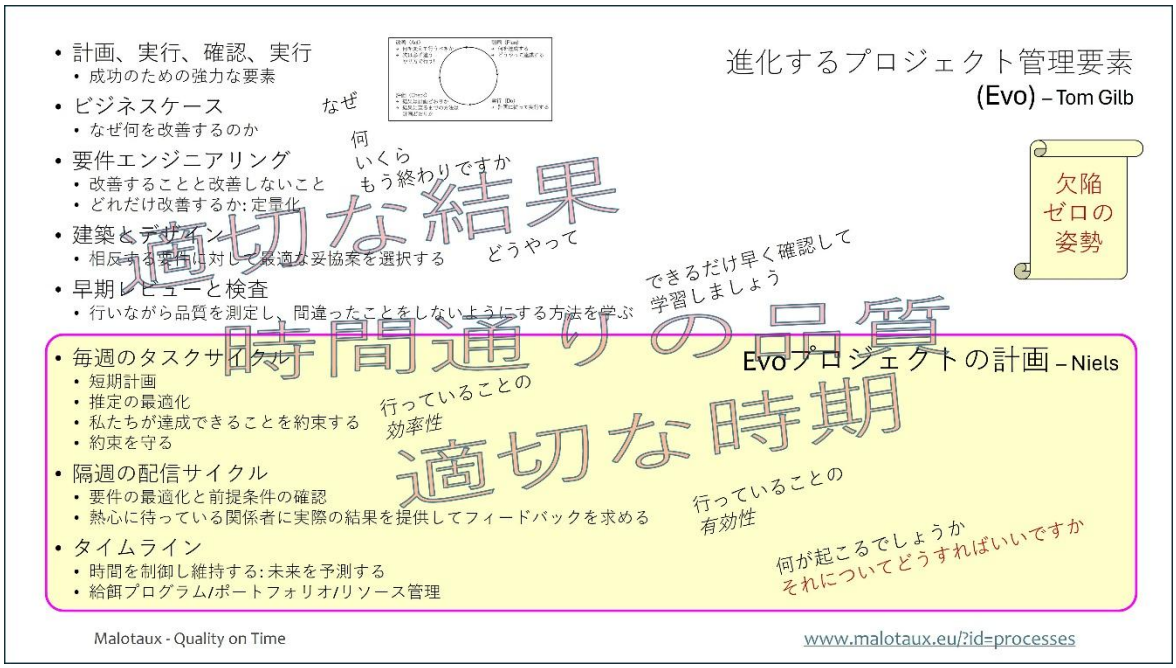
百聞は一見に如かず

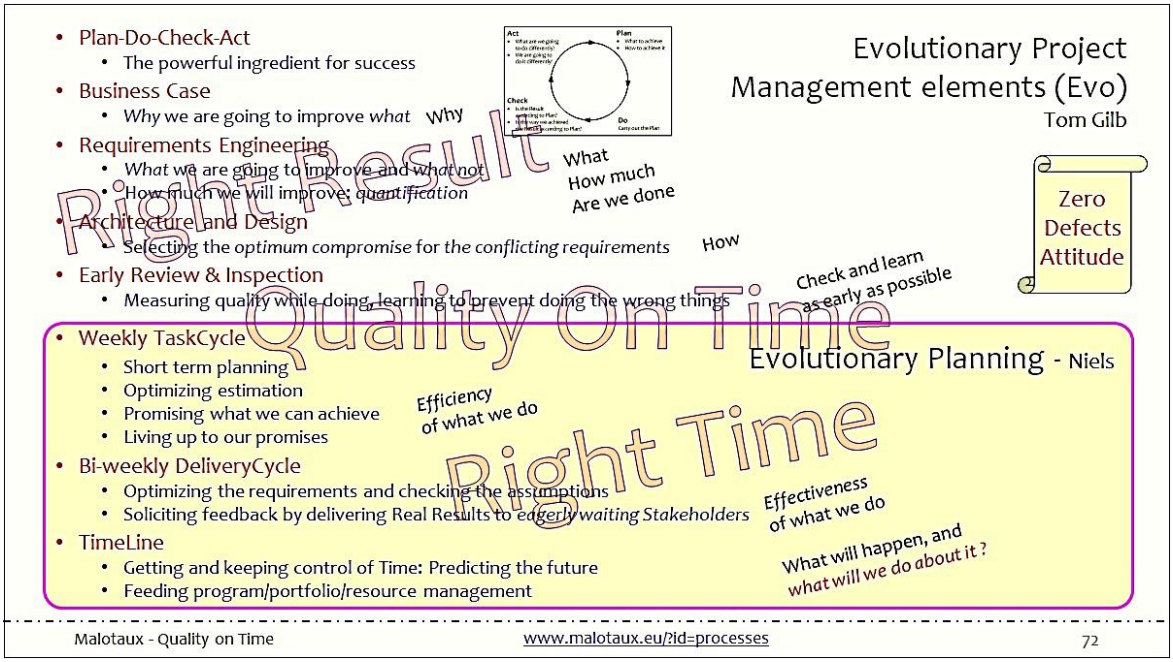
ぜひ私と一緒に 5 週間のエクササイズに取り組んでみませんか？実際に体験していただき、この方法がどのようにあなたの業務効率、効果、そして予測力を向上させるのかを実感していただけるでしょう。これまで多くの方々とこのエクササイズを行ってきた経験から、最初の数週間はきっと大変だと感じると思います。自分の現在の仕事のやり方を、今はまだ想像もできないような形で直視することになるでしょう。実際、最初は「かなり辛い」と感じる方もいらっしゃいますが、その後すぐに昇進された方も少なくありません。5 週間後には、この方法を実際に機能させるための多くの詳細について、より深く理解できるようになるでしょう。また、この方法があなたのプロジェクトや組織にどのように役立つのか、実感していただけるはずです。

このご案内により、エクササイズへの参加希望が殺到する可能性があります。そのため、誠に恐縮ですが、お申し込みを選考・お断りさせていただく場合や、キャンセル待ちとさせていただく場合がございます。参加の可能性を高めるためには、なぜこの演習があなたにとって、そして私があなたを支援することにとって興味深いのかを説明していただくことをお勧めします。

www.malotaux.eu/contact　または niels@malotaux.eu。

第 9 章の手順から始めて、何が起こるか見てみましょう。





必要に応じて、多くの要素から選択できます

ニルス・マロトー

Predictable Projects

How to deliver the Right Results at the Right Time

本書で説明するこのアプローチを用いることで、ストレスを最小限に抑えながら、より短い期間でプロジェクトを成功に導くことができ、同時に、いつ何が達成できるかを驚くほど正確に予測することが可能になります。

不可能に聞こえますか？多くの人が試す前にそう思っていました。信じられない話でしょうか？何しろ、あなたは長年プロジェクトに携わってきました。必要な資格も経験もすべて持っています。プロジェクトは予想以上に時間がかかるものだと誰もが知っているのに、一体何を見落としていたのでしょうか？突然、プロジェクトがより良い結果をはるかに速く出せるようになるなんて。私はほぼ必ず、「そんなことは不可能だ」「以前試したことがある」「聞いたことがある」「ここではうまくいかない」「私たちのプロジェクトは違う」「私たちはとてつもなく大きなシステムを構築している」「私はすでにこれらのことをやっている」といった抵抗の声を聞きます。これらの議論は十分に理解できます。なぜなら、人々は、大幅に短い時間でより良い結果を出すための技術が存在するとは想像できないからです。そうでなければ、彼らはすでにこれらの技術を使っているはずですよね？

ニルス・マロトーは、プロジェクトパフォーマンスの最適化を専門とする独立系プロジェクトコーチです。1974 年以来、デルフト工科大学、オランダ陸軍、フィリップス・エレクトロニクスで電子ハードウェアおよびソフトウェアシステムの設計に携わり、その後 20 年間は自身のシステム設計会社を率いました。1998 年以来、顧客が必要とするものを必要な時に提供し、顧客の成功を可能にする「品質を期日通りに提供する」プロジェクト支援に専門知識を注いでいます。この目的のために、ニルスは進化型プロジェクトマネジメント（Evo）手法、要求工学、レビューおよび検査技術を効果的に教えるためのアプローチを開発しました。2001 年以来、オランダ、ベルギー、中国、ドイツ、インド、アイルランド、イスラエル、日本、ルーマニア、南アフリカ、セルビア、英国、米国など 40 以上の組織で 400 以上のプロジェクトを指導・コーチングし、実際のプロジェクトにおいてどの手法がより効果的で、どの手法が効果的でないかについて豊富な経験を積んできました。彼はカンファレンスで頻繁に講演を行っています。詳細は www.malotaux.eu/conferences をご覧ください 。

www.malotaux.eu/booklets でご覧いただけます:

1. 進化型プロジェクト管理手法
2. 進化的手法によって品質がどのように保証されるか
3. プロジェクトの成功に対するテストの貢献度を最適化する

3ª より良い結果を得るための品質保証の最適化（3 と同じだが、今回はソフトウェア以外のプロジェクト向け）

4. 設計段階からプロジェクトリスクを管理する
5. タイムライン：プロジェクトを管理し、コントロールし続ける
6. 人間の行動を認識し理解する
7. 進化計画（小冊子 5「タイムライン」に似ていますが、順序が異なり、予測可能性が向上しています）
8. 助けて！品質管理に問題が発生しました！
9. 予測可能なプロジェクト ― 適切な結果を適切なタイミングで実現する方法（本書）

N R Malotaux
Consultancy
Niels R. Malotaux
phone +31-655 753 604
mail niels@malotaux.eu
web www.malotaux.eu

Originally published in SyEn newsletter #50: www.ppi-int.com/newsletter/SyEN-050.php, 2012
Version 1.5 (slightly updated) - 6 Februari 2026
Japanese version: 22 July 2026 - 日本語版：2026年7月22日