

XDDP 導入を成功に導く処方せん 小冊子 (ver.3.0)

2023 年 5 月 26 日

派生開発推進協議会

T22 研究会



1. はじめに

XDDP を導入したことで大きな期待が得られた「XDDP の成功事例」が多く聞かれる一方で、XDDP を導入したが期待した効果が得られなかった「XDDP の導入の失敗事例」や、XDDP を繰り返すうちに効果が得られなくなった「XDDP の定着の失敗事例」が散見される。

そんななか、2014 年の派生開発カンファレンスにおいて、ワークショップ「XDDP はこうすれば定着失敗する」が開催された。XDDP 導入/定着において、どのような失敗事例があるのか。それら失敗事例の原因は何なのか。このワークショップでの議論を起点として、さらに研究を深めたいと希望する有志を中心に T22 研究会は発足した。

失敗事例には多くの人が共有できる「共感」(そうそう！ あるある！)が存在し、その共感からはより前向きな議論を引き出し易い。失敗事例にこそ多くの教訓を含んでいる。

-- 先人の失敗を繰り返さないための対策を T22 から発信したい --

そんな熱い想いのもと本研究は継続されている。

本研究が分析に用いた、現場で実際に起きた「失敗事例」は、清水吉男さんから T22 にご提供頂いた資料「XDDP 導入失敗事例」に記録された事例に基づいている。これは、清水さんご自身が 15 年にわたるコンサルティング経験を通して収集した 24 の失敗事例が記録されたものである。この資料なくして、本研究活動は起こりえなかった。清水さんは、闘病生活に入る直前まで T22 の研究会合には毎回参加下さり、数多くのアドバイスや考え方の指針を示してくださいました。改めて感謝申し上げるとともに、今後も清水さんの意志を引継ぎ、本研究活動を継続していく所存である。

本研究活動では、XDDP の導入/定着の失敗に至らしめた種々の原因を、25 の"病気"として体系化し、まず 10 の病気に対する処方せんをまとめ 2018 年に ver.1.0 として発行した。その後、派生開発カンファレンス 2019 に本小冊子の活用方法を考えるワークショップを行い、そこから得た情報と、残る 10 の病気について処方せんをまとめ 2021 年に ver.2.0 として発行した。そして今回、新たな視点での処方せん活用事例を追加した ver.3.0 を発行する。XDDP の導入/定着に取り組んでいる、あるいは導入を検討中の組織やプロジェクトが、本小冊子を活用することでいくつかのヒントや気づきを得ることができれば、幸いである。

本小冊子は、今後も段階的に改版していく予定である。今後の改定版をよりよいものとしていくために、本小冊子に対する感想や忌憚のないご意見など、切にお願い申し上げます次第である。

派生開発推進協議会 (AFFORDD) T22 研究会

派生開発カンファレンスの場に発表される事例は、中には小さな成功事例だったりするが、基本的に成功事例である。これまで毎回 250 名程の参加者があり、カンファレンスに参加することで成功するためのヒントが得られる。大きな組織での工夫もあれば、小さな開発組織での機敏な取り組み事例などもあって、参加者にとっては「なるほど、こんな方法があったのか」「こうすればいいんだ」「自分たちの考えている取り組み方とそれほど違ってはいない」といったヒントを持ち帰ることができる。実際、発表者の中には、以前のカンファレンスでは参加者だった人もいる。

しかしながら現実には、多くの組織で XDDP の導入に失敗しているケースがあると思われる。もちろん、それらのケースは派生開発カンファレンスの場に発表されることはないため、実態は把握しにくい。彼らとしては XDDP の本を読みながら取り組んだつもりでも、どこか違った解釈をしてしまった部分があるのかもしれない。あるいは、組織の文化などが邪魔をして XDDP を部分解釈してしまったかもしれない。あるいは、とんでもない数のバグを出していて、XDDP でどこまで減るのか不安だったため、従来の取り組みと重ねてしまったのかもしれない。この人たちにとっては、派生開発カンファレンスに参加しても、有効なヒントは得られないことになる。結局、派生開発推進協議会を設立した目的の半分は満たされていないことになる。

とは言え、失敗事例を集めて発表してもらおうというわけにはいかない。なぜなら、彼らは「成功していない」という現実をもっているが、「なぜ成功しなかったのか」を説明することはできない。誰かが、その失敗事例にたいして何を間違えたのかを指摘できなければ、単に失敗した人たちの慰め合いになってしまう。それならいっそのこと誰かが失敗事例を解説したほうが早いのではないか。これが、本書をまとめることにした動機である。

本書の多くは、私自身が 15 年間のコンサルティングの中で収集した失敗事例である。私がコンサルティングに入る前に、XDDP に取り組んでいたがなかなかうまく行かないので、やむなくコンサルティングを依頼してくるというケースも少なくなかった。

導入に失敗する中で、何とかヒントを得ようと派生開発カンファレンスに参加した人もいて、交換会の場で彼らの悩みを聞くこともあったし、外部のカンファレンスなどで講演したあとの交換会の場でも、同様の相談を受けることがあった。

本書では、それらの失敗事例を「XDDP の導入に失敗したケース」と、「XDDP の定着に失敗したケース」の 2 つに分けている。後者では、取り敢えずは現場の開発チームとしては XDDP の導入はなんとかできたものの、その後の他部門への展開や、継続させるところで失敗したケースを扱うことにした。

ここに示すのはそれぞれの失敗事例に対する一つの「対応策」にすぎない。その人が身につけてきた技術や習慣、その組織の文化によって、解決策は異なるかもしれない。それでも、ここに提案する対応策は十分に参考になるだろう。

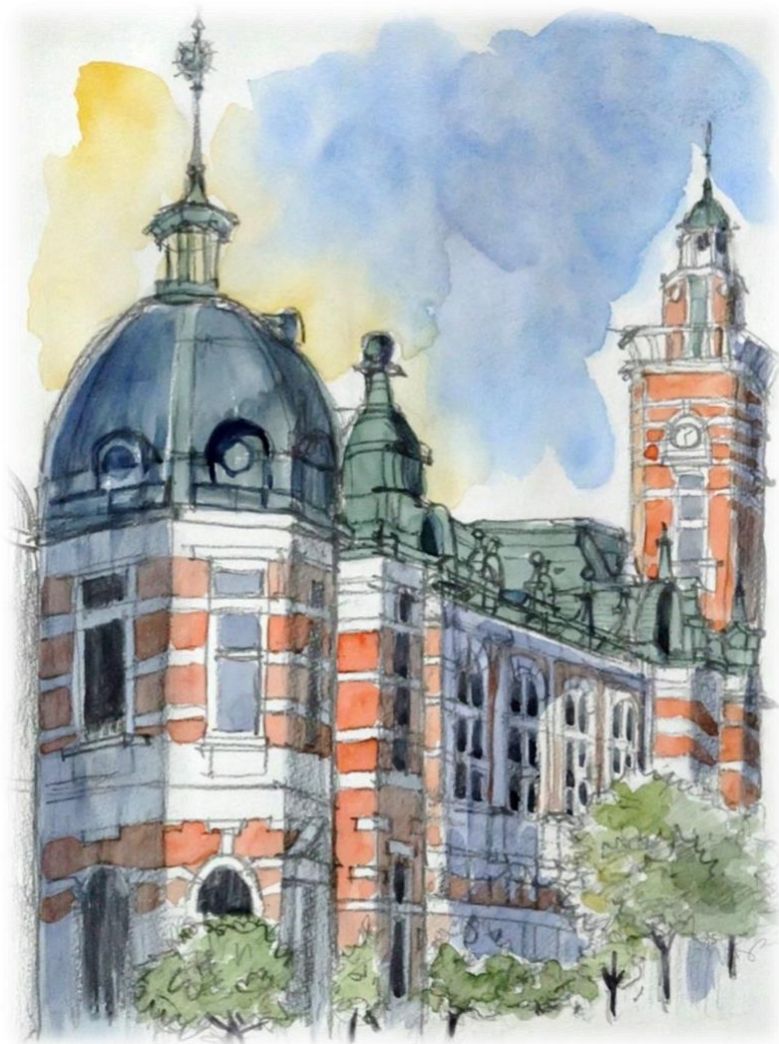
また、ここに示す失敗事例や対応策は、XDDP の導入に限らず、他の技法やツールの導入にも似たようなことが起きると考えられ、そこにも参考になるだろう。

清水 吉男

2. 目次

1. はじめに.....	2
2. 目次.....	4
3. 我々のアプローチ.....	6
3.1. 失敗事例を病気と捉える.....	6
3.2. 病気を治療する処方せん.....	7
3.3. 予防は生活習慣の改善.....	8
3.4. 処方せんの活用シーン.....	9
3.5. 処方せん活用による変化や効果.....	10
4. 治療の処方せんの探し方.....	11
4.1. XDDP の導入状況から探す.....	11
4.2. 該当する組織文化から探す.....	17
5. 治療の処方せん.....	20
#1 レビュー不全症候群：レビュー軽視症.....	21
#2 レビュー不全症候群：有識者依存症.....	22
#3 本質理解欠乏症候群：形式依存症.....	23
#4 ながら作業無自覚症.....	24
#5 トレーニング省略症.....	25
#7 標準プロセス呪縛症.....	26
#8 課題形成能力欠乏症.....	27
#11 プロセス化石化症.....	28
#12 本質理解欠乏症候群：偽装工夫無自覚症.....	29
#13 本質理解欠乏症候群：慢性ダイジェスト症.....	30
#14 本質理解欠乏症候群：我流症.....	31
#15 せっかち病 習得順序間違え.....	32
#16 対策重ね着症.....	33
#19 肯定眼欠如症.....	34
#20 プロセス設計意識欠乏症（導入失敗型）.....	35
#21 最前線至上症.....	36
#22 社内アドバイザー育成軽視症.....	37
#23 プロセス設計意識欠乏症（定着失敗型）.....	38
#24 ブルーカラー症.....	39
#25 先陣恐怖症.....	40
6. 予防の処方せん.....	41
価値観不良.....	42
企画不良.....	44
誤判断（開発者）.....	45
誤判断（共通）.....	46

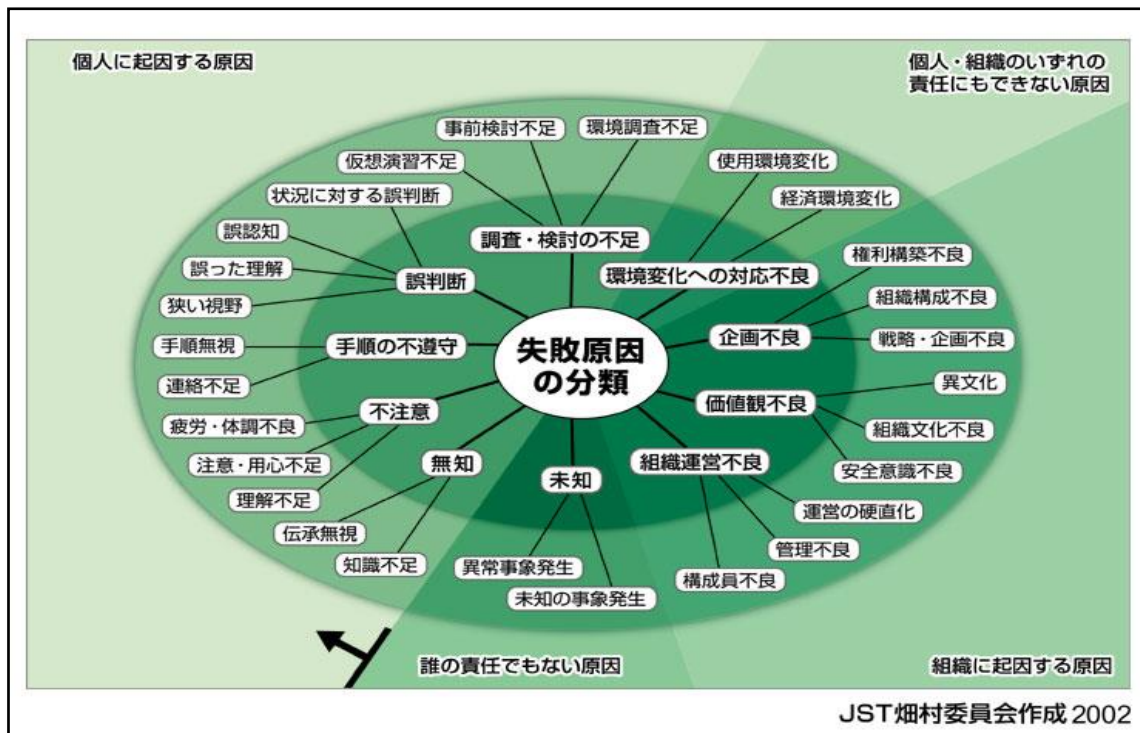
調査・検討の不足.....	47
無知.....	48
組織運営不良（管理者）.....	49
組織運営不良（共通）.....	50
環境変化への対応不良.....	51
付録 1.1 図『「生活習慣」が引き起こす病気と病気同士の関係』.....	52
付録 1.2 プロジェクトリスクマネジメント計画への適用.....	56



3. 我々のアプローチ

3.1. 失敗事例を病氣と捉える

XDDP の導入や定着に失敗したときの問題の本質を捉えるには、失敗原因を体系的(構造的)に整理・分類する必要があると考えた。そこで失敗の原因を追究することで広く認識されている「失敗学」の中で失敗原因を階層的に分類することができる「原因まんだら」を整理・分類するツールとして利用した。当活動で利用した原因まんだらを図 3.1 に示す。



<http://www.sozogaku.com/fkd/img/genin.jpg>

図 3.1 : 原因まんだら

原因まんだらは、失敗原因を内側の第 1 レベル (L1) と外側の第 2 レベル (L2) の 2 階層でとらえる構造であり、この原因まんだらを使い、XDDP の導入や定着に失敗した原因を外側の第 2 レベルに割り当てて原因を分類した。なお、原因まんだらにある「個人」と「組織」は、問題の対象を具体的にするため、それぞれ、「開発者」と「管理者」に置き換えて使用することとした。しかし、処方せんを作成する過程において、「開発者」と「管理者」に当てはめるには無理があることが判明したため、「個人」と「組織」はあくまでも参考の情報とした。

3.2. 病気を治療する処方せん

次に、誰もが使いやすい失敗事例の対策にするためには、そこに示されている内容が分かりやすく納得できるものでなければならないと考え、失敗の事象を病気の発症ととらえる「ソフトウェア病理学」のアプローチを取り入れた。失敗した事象を病気の症状に置き換え「処方せん」として身近なものにすることで、問題が起こっている開発現場でも受け入れやすい改善策になるようにまとめた。

当活動で認定した病名とその病原の関連を図 3.2 に示す。

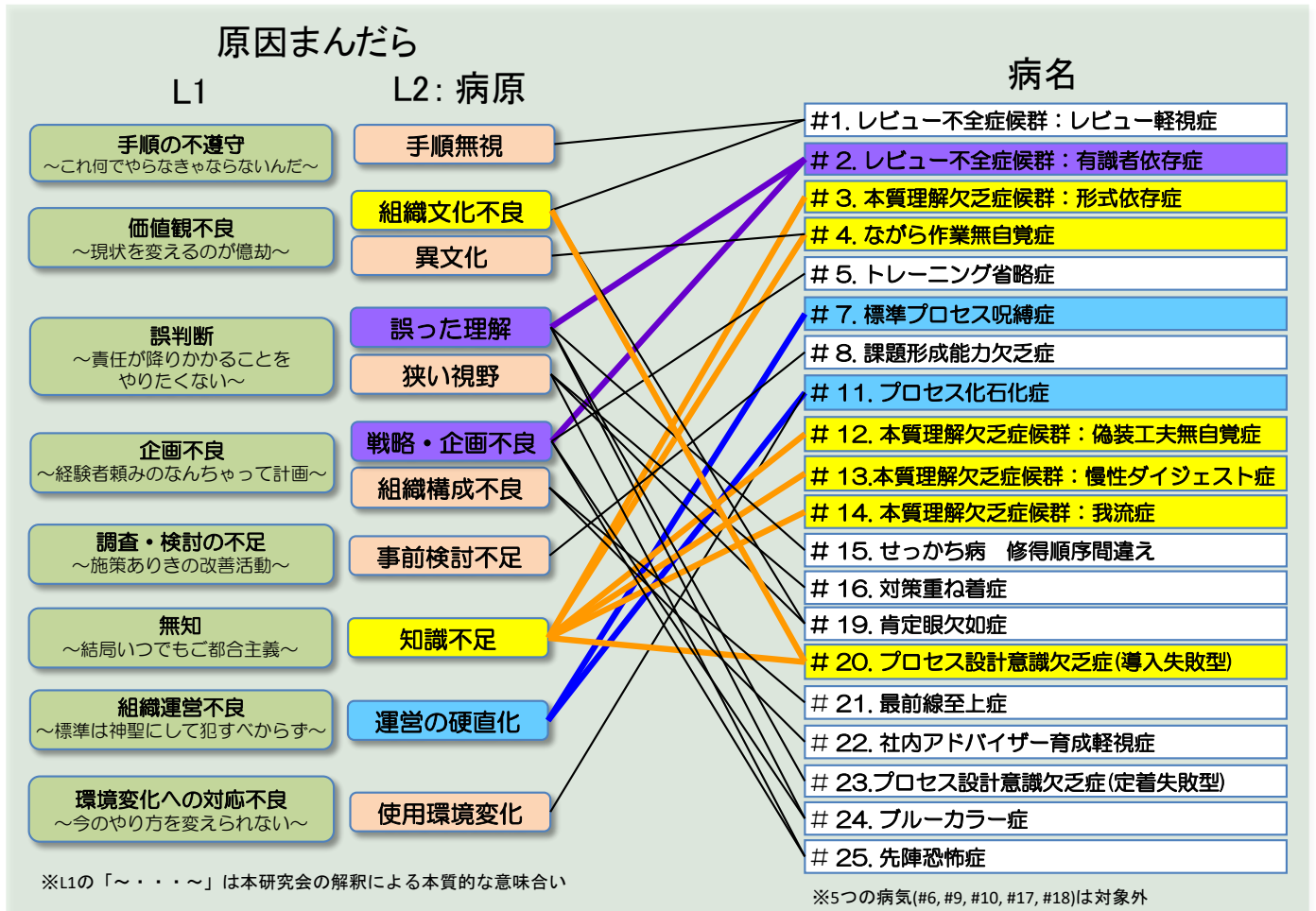


図 3.2 : 病名・病原関連図

ここでは、病原は原因まんだらの第 1 レベル(L1)、第 2 レベル(L2)に対応させている。

病名から見た場合、例えば、「#2 レビュー不完全症候群：有識者依存症」を見ると、「誤った理解」と「戦略企画不良」の 2 つの病原(L2)から発症する。一方、病原(L2)から見た場合、例えば「知識不足」に侵されると、「#3 本質理解欠乏症候群：形式依存症」など 6 つの病気(病名)を引き起こす。また、「運営の硬直化」に侵されると、「#7 標準プロセス呪縛症」など 2 つの病気(病名)を引き起こすことになる。このように、病名と病原は n 対 n の関係になることが解った。

※ 病名で欠番となっている #6, #9, #10, #17, #18 の 5 つの病気については、内容的に他の病気と同等とみなし処方箋の対象外(欠番)とした。

3.3. 予防は生活習慣の改善

発生している問題を分析して対応することは重要であるが、できれば問題が発生する前に未然に防止することはできないか考えた。そこで、共通の対策を検討するために再度、原因まんだらを利用した。病気(病名)の原因は原因まんだらの第2レベル(L2)で分類しているので、その上位にある第1レベル(L1)が複数の病気(病名)の共通点になる。この共通点の対策が予防になると考えた。

当活動で認定した病名とその病原と生活習慣の関連図を図3.3に示す。

例えば、病名「#2 レビュー不全症候群：有識者依存症」は「誤った理解」と「戦略企画不良」の2つの病原(L2)から発症する。「#16 対策重ね着症」は「狭い視野」の病原(L2)から発症する。「#2 レビュー不全症候群：有識者依存症」の病原である「誤った理解」と「#16 対策重ね着症」の病原である「狭い視野」の上位であるL1は「誤判断」になる。すなわち病気発症の本質に相当するものがL1であり、このL1を起こさないようにするための対応策を予防の処方せんとしてまとめた。

これら病原(L2)のもとであるL1は、言ってみれば生活習慣に相当するもので、このような生活習慣を改めなければ、いずれは対応する病原(L2)に感染して病気を発症することになる。予防法は生活習慣(L1)を改善するための処方せんである。

これらの病気の中で将来発症のリスクが考えられるもの、あるいは現時点で該当する生活習慣が認められるものについては、それらに対応する予防の処方せんを選択して実施することで、病原の感染や病気の発症を防ぐことができる。

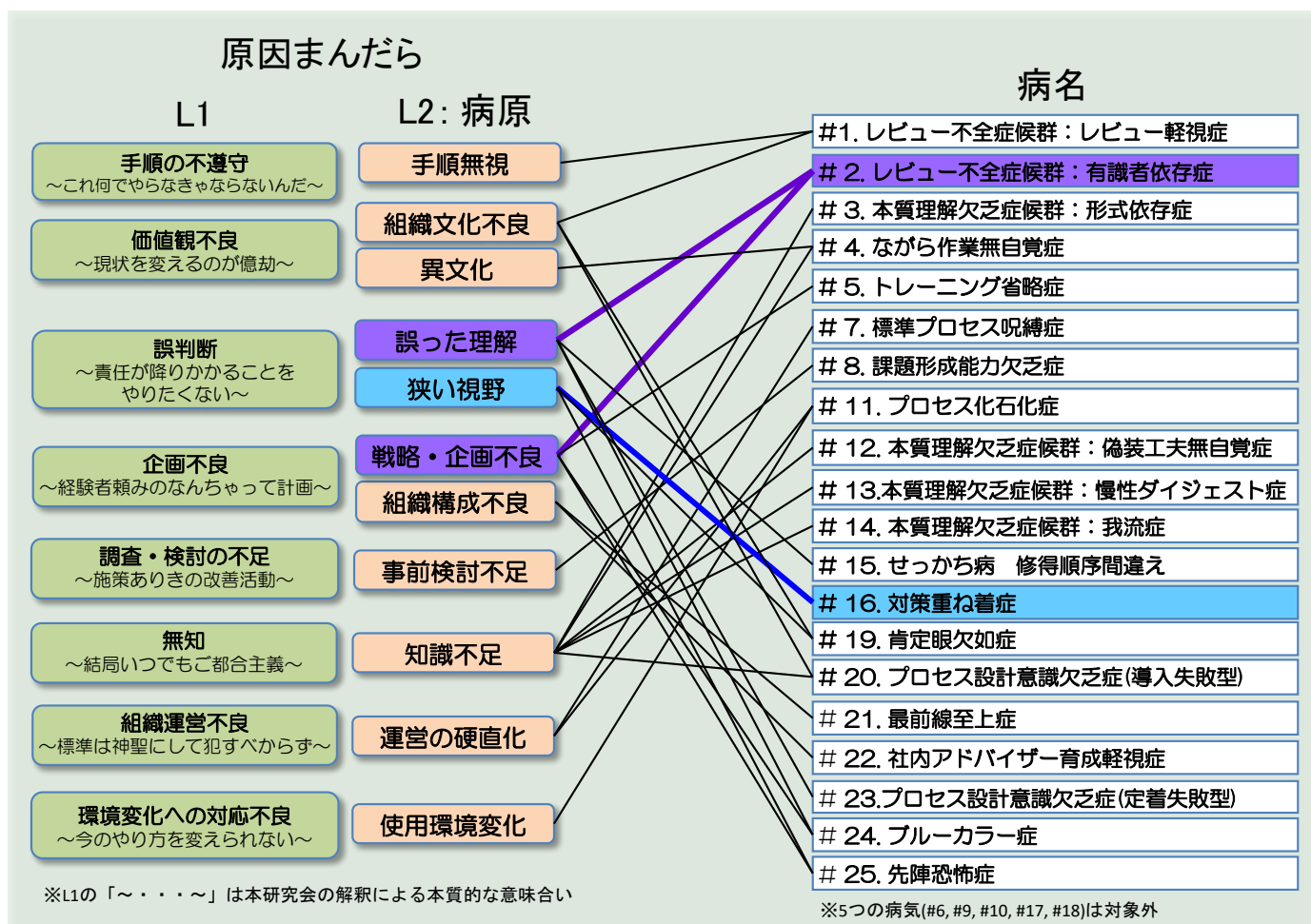


図 3.3 : 病名・病原・生活習慣関連図

3.4. 処方せんの活用シーン

下記に示すそれぞれの状況で、開発者と管理者それぞれの立場で処方箋が活用できる。

- ① 炎上中のプロジェクトにおける問題対応：治療の処方せん
 - ✓ 発生している問題と同様な症例を本小冊子の処方せんから探し、その治療法を処方する。
- ② XDDP の導入／定着に失敗した経験をもつプロジェクトの次回への対策（再チャレンジ）：治療の処方せん
 - ✓ 過去に XDDP の導入を試みたが、XDDP は使えない技術であると判断し導入をあきらめた組織や個人が、本小冊子の処方せんから真の原因に気づき、XDDP 導入の再チャレンジを決断するきっかけにする。
 - ✓ プロジェクトで発生した問題と同様の症例を本小冊子の処方せんから探し、失敗原因を確認したうえで、次回に向けた再発防止の策定を計画段階で実施する。
- ③ これから XDDP に取り組むプロジェクトでのリスク回避：予防の処方せん
 - ✓ 本小冊子が提示する失敗事例から、自プロジェクトでも起きる可能性があることと判断できる事例（＝リスク）をリストアップし、各失敗事例の要因となる「生活習慣」を改める「予防の処方せん」実施を導入計画に組み込む。

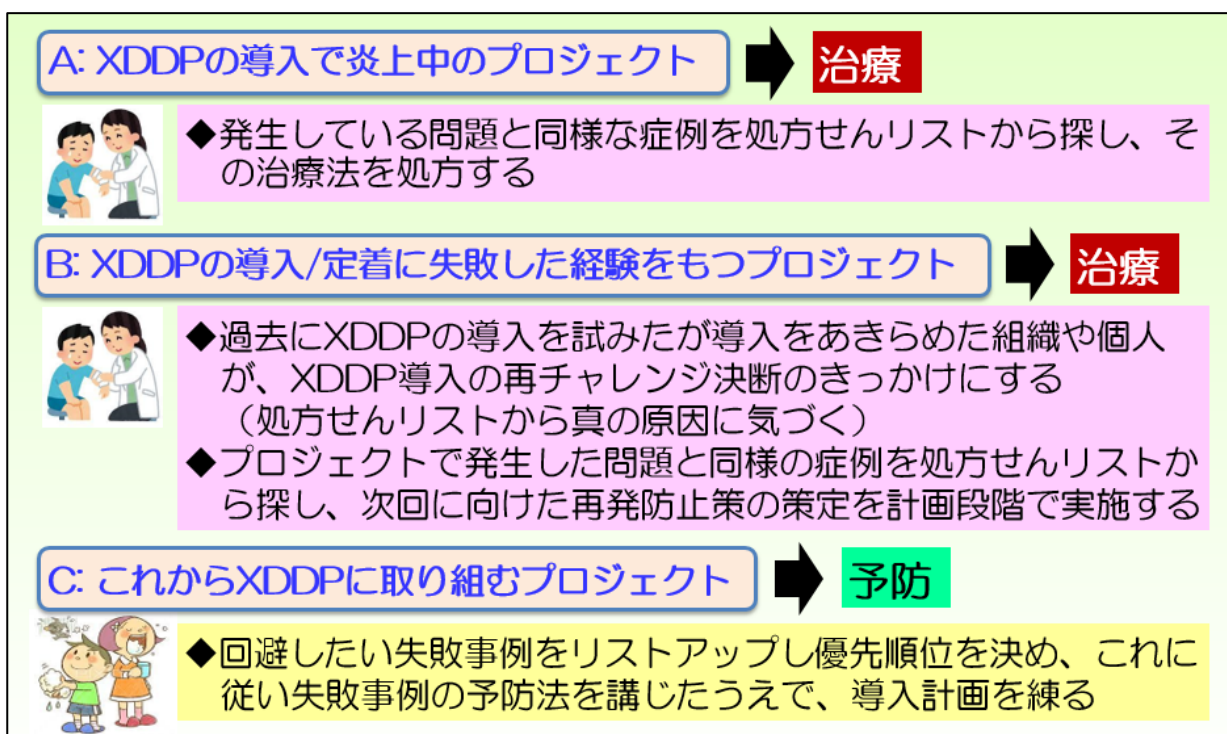


図 3.4：処方せんを活用する 3 つのシーン

3.5. 処方せん活用による変化や効果

下記に示すそれぞれの状況で、開発者と管理者それぞれの立場で改善による変化や効果を期待できる。

① 炎上中のプロジェクトにおける問題対応：治療の処方せん

- ✓ 問題解決までのスピードアップ。
- ✓ 問題に対する対策内容の質の向上。

② XDDP の導入／定着に失敗した経験をもつプロジェクトの次回への対策（再チャレンジ）：治療の処方せん

- ✓ XDDP の導入をあきらめた組織や個人が、導入の再チャレンジ案を計画する。
- ✓ 次回プロジェクトにおける再発防止策の質と検討効率のアップ。

③ これから XDDP に取り組むプロジェクトのリスク回避：予防

- ✓ XDDP に取り組んだ際の失敗事例の発生確率を低くする。

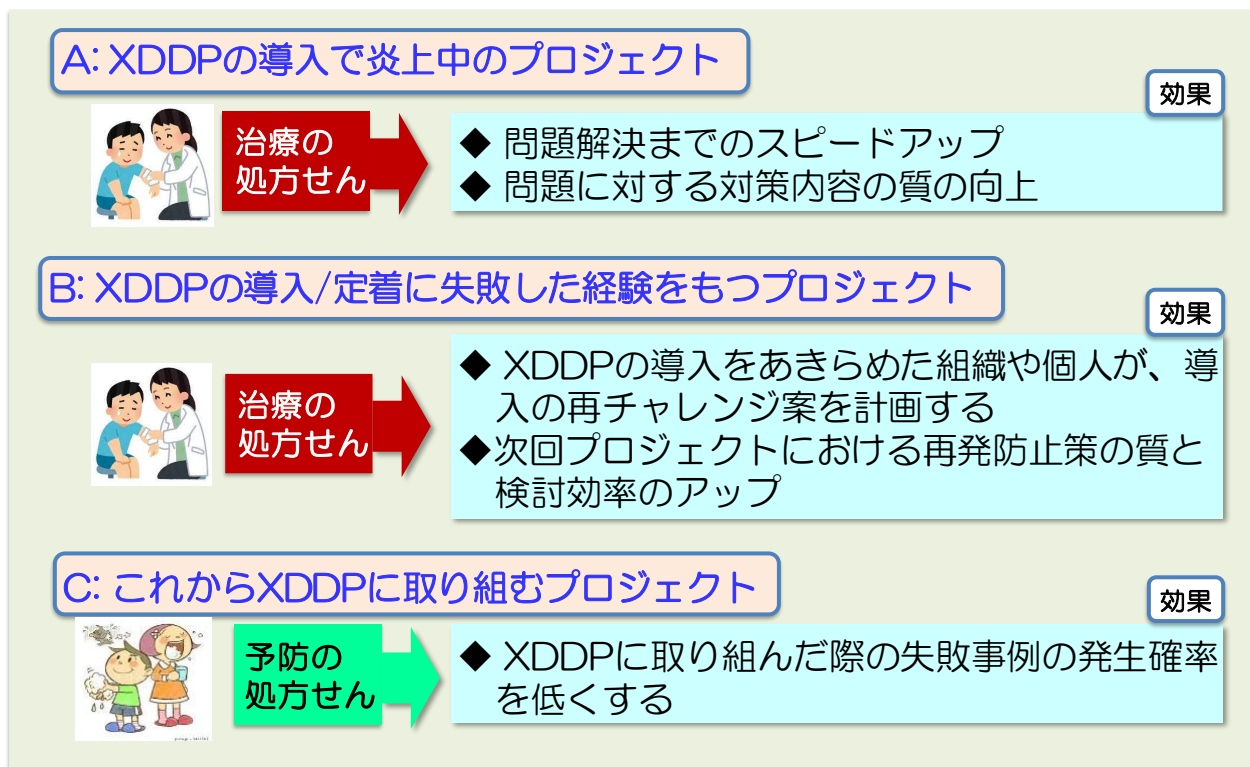


図 3.5：治療と予防の処方せん活用で期待できる効果

4 治療の処方せんの探し方

本小冊子から、組織やプロジェクトに適合する処方せんを探すうえでの入り口として

4.1. 「XDDP の導入状況から探す」

- a) 「これから導入する（企画段階）」
- b) 「現在導入活動中である」
- c) 「現在定着活動中である」
- d) 「導入・定着において過去に失敗した経験がある」

→b) 「現在導入活動中である」と c) 「現在定着活動中である」を参照して頂きたい。

4.2. 「該当する組織文化から探す」

を用意している。

本小冊子の利用者である開発者、管理者は、自分の組織やプロジェクトにおける上記 XDDP 導入状況や組織文化の観点から、該当する「病気」の症状や組織文化に対応する処方せんを参照して頂きたい。

4.1. XDDP の導入状況から探す

=====

a) これから導入する（企画段階）

=====

【開発者が注意すべき病気】

- #13 本質理解欠乏症候群：慢性ダイジェスト症 処方せん #13
- 推進役（開発者）が XDDP は良さそうだと判断しフォーマットだけを手に入れ開発現場に導入したが、誰も適切な指示ができないため現場がかえって混乱し期待した効果が得られなかった。
- ・既存のプロセスに XDDP の成果物を追加したため作業が増加した。
 - ・XDDP の成果物を最後に作成している。成果物の作成が形骸化している。
 - ・XDDP の指導者が不在のため誰も成果物のレビューができない。

【管理者が注意すべき病気】

- #5 トレーニング省略症 処方せん #5
- 1)XDDP のトレーニングなしでの導入により現場が混乱した。
- [混乱の例]
- ①開発期間が長くなった
 - ②開発プロセスが不明(不定)になった
 - ③バグが多くなった
- 2)XDDP を導入しても期待した効果が得られない
- 3)管理者は手法を紹介するだけで推進しない

■#7 標準プロセス呪縛症 処方せん #7

「組織の標準プロセス」の名の下に異なるプロセスが許容されない。

- ・組織として制定された開発プロセスが有る。
- ・制定された開発プロセスは、外部の機関（もしくは内部の有資格者）によって認定／確認されている。
- ・制定された開発プロセスは、ある規格で開発組織に必要とされたプロセス「標準プロセス」として組織の開発規定に記載されている。
- ・開発規定に開発プロセス（標準プロセス）遵守の記載がある。
- ・開発の要求や開発環境が変わってきているが、開発規定に記載されているため（標準）プロセスを変更／更新出来ない。

【開発者と管理者共通で注意すべき病気】

■#8 課題形成能力欠乏症 処方せん #8

現在派生開発を行っており変更量の割にバグが多発している。「XDDP が派生開発に有効」と情報を得たので導入してみたが想定したほどの効果が出ていない。

■#15 せっかち病 習得順序間違え 処方せん #15

とにかく XDDP さえ導入すれば派生開発はうまくいくと考え導入したが、下記の事象が発生し、状況が改善しなかった。

- ・適切な修正箇所を抜き出すためのソースコード解析ができていない。
- ・「変更3点セット」による変更箇所や変更内容、変更方法の特定がずれている。
- ・XDDP の成果物をレビューしても有効な指摘が得られない。

■#16 対策重ね着症 処方せん #16

XDDP の取組みだけでは不安になっていろいろな取り組みを重ねてしまった結果、工数がたりなくなりプロセスを省略するなど不完全な取り組みになる。もしくは期待する効果が出ないので取り組み自体をやめてしまう。

- ・セミナーやカンファレンス（・コンサルティング）で入手した改善策を手当り次第に取込んでしまう。
- ・1 施策では不安であるため入手した施策を全て実施（並行実施）してしまう。
- ・結果工数が足りなくなってしまう。
- ・そして必要なプロセスを省略してしまう。

=====

b) 現在導入活動中である

=====

【開発者がかかる病気】

■#3 本質理解欠乏症候群：形式依存症 処方せん #3

形から入って、その後も理解が深まらず、XDDP の変更 3 点セットを使っているが、期待した効果が出ていない。

XDDP 導入にあたって「形」から入ることは必ずしも悪くないが、そのままの状態が続くと、時間の経過とともに取り組みが形骸化して XDDP の効果が得られなくなってしまう。具体的には「なぜ変更要求と変更仕様を使い分けるのか?」、「なぜ【before / after】で表現するのか?」など「そうすることの本質」を認識しないまま変更 3 点セット作成を続けているため、効果が得られない。

■#12 本質理解欠乏症候群：偽装工夫無自覚症 処方せん #12

変更 3 点セットのフォームや構成をアレンジや改善の名目で変更を加えた結果、本来 XDDP で得られるはずの効果が得られなくなる。

【自己流アレンジの例】

- ・変更要求仕様書に「備考欄」を新たに設け、見つけた変更箇所と変更ソースコードを書き残しておく。ソースコード変更時はこれをカット&ペーストして”効率化”と称する。
- ・変更要求仕様書のフォームから「説明」欄を削除してしまう。
- ・変更要求仕様書の記述で「before 欄」、「after 欄」をテンプレート化してしまう。

■#13 本質理解欠乏症候群：慢性ダイジェスト症 処方せん #13

推進役（開発者）が XDDP は良さそうだ判断しフォーマットだけを手に入れ開発現場に導入したが、誰も適切な指示ができないため現場がかえって混乱し期待した効果が得られなかった。

- ・既存のプロセスに XDDP の成果物を追加したため作業が増加した。
- ・XDDP の成果物を最後に作成している。成果物の作成が形骸化している。
- ・XDDP の指導者が不在のため誰も成果物をレビューができない。

【管理者がかかる病気】

■#5 トレーニング省略症 処方せん #5

1)XDDP のトレーニングなしでの導入により現場が混乱した。

[混乱の例]

- ①開発期間が長くなった。
- ②開発プロセスが不明(不定)になった。
- ③バグが多くなった。

2)XDDP を導入しても期待した効果が得られない。

3)管理者は手法を紹介するだけで推進しない。

■#7 標準プロセス呪縛症 処方せん #7

「組織の標準プロセス」の名の下に異なるプロセスが許容されない。

- ・組織として制定された開発プロセスが有る。
- ・制定された開発プロセスは、外部の機関（もしくは内部の有資格者）によって認定／確認されている。
- ・制定された開発プロセスは、ある規格で開発組織に必要とされたプロセス「標準プロセス」として組織の開発規定に記載されている。
- ・開発規定に開発プロセス（標準プロセス）遵守の記載がある。
- ・開発の要求や開発環境が変わってきているが、開発規定に記載されているため（標準）プロセスを変更／更新出来ない。

【開発者と管理者がいっしょにかかる病気】

■#1 レビュー不全症候群：レビュー軽視症 処方せん #1

- ・変更3点セットを書いているのにレビューを実施しようとしない。また実施しても誰もレビューに参加しない。
- ・時間がないことを理由にレビューを実施しない。

■#2 レビュー不全症候群：有識者依存症 処方せん #2

レビューに存在しない“有識者”を求め、レビュー確認漏れが減らない。

- ・レビューで有効な指摘ができない理由を有識者不在としている。
- ・レビューの問題を有識者不在とし、それ以上の深堀ができない。
- ・レビューで問題の指摘漏れが繰り返し発生する。

■#4 ながら作業無自覚症 処方せん #4

変更箇所を見つけ次第ソースを変更するやり方を変えられず、変更3点セット作成に取り組めない。

- ・実装期間に、設計書に書かれていないことを穴埋めする作業が習慣化している。
- ・要求の仕様化が不完全な状態で設計・実装へと進み、その穴埋め作業はコードを実装しながら実施することが、納期を守るうえで当然なやり方であると信じている。

■#8 課題形成能力欠乏症 処方せん #8

現在派生開発を行っており変更量の割にバグが多発している。「XDDP が派生開発に有効」と情報を得たので導入してみたが想定したほどの効果が出ていない

■#11 プロセス化石化症 処方せん #11

自組織のプロセスの“特殊性”を強調して XDDP を自組織のプロセスに合わせようとした結果、従来のプロセスが残ったままで XDDP の本質から外れる。

- ・ XDDP 導入のためには、自組織の従来のプロセスを変更する必要があるにもかかわらず、従来から慣例的に実施してきている慣れ親しんだプロセスなので、変えようとしなない。その結果、XDDP のプロセスを本質から歪めたかたちで導入してしまった。
- ・ 自組織にとって、もはや遵守することの意味を特定できないプロセスになっているにもかかわらず、慣例的に継続されてきているものなので簡単に外すことができない。これが XDDP のプロセス導入を阻害する。

■#14 本質理解欠乏症候群：我流症 処方せん #14

自分たちの旧来のやり方（都合）に合わせて変更要求仕様書と変更設計書を作成してしまい、導入しても効果が出ない。

変更要求仕様書と変更設計書の作成が我流の書き方になってしまっている。

■#15 せっかち病 習得順序間違え 処方せん #15

とにかく XDDP さえ導入すれば派生開発はうまくいくと考え導入したが、下記の事象が発生し、状況が改善しなかった。

- ・ 適切な修正箇所を抜き出すためのソースコード解析ができていない。
- ・ 「変更3点セット」による変更箇所や変更内容、変更方法の特定がずれている。
- ・ XDDP の成果物をレビューしても有効な指摘が得られない。

■#16 対策重ね着症 処方せん #16

XDDP の取組みだけでは不安になっていろいろな取り組みを重ねてしまった結果、工数がたりなくなりプロセスを省略するなど不完全な取組になる。もしくは期待する効果が出ないので取り組み自体をやめてしまう。

- ・ セミナーやカンファレンス（・コンサルティング）で入手した改善策を手当り次第に取込んでしまう。
- ・ 1 施策では不安であるため入手した施策を全て実施（並行実施）してしまう。
- ・ 結果工数が足りなくなってしまう。
- ・ そして必要なプロセスを省略してしまう。

■#20 プロセス設計意識欠乏症（導入失敗型） 処方せん #20

開発現場では組織に存在する標準プロセスを変更もせず利用している。組織の課題を解決するために、XDDP を導入しようとするが、過去から慣れ親しんだ標準プロセスに執着してしまい XDDP の導入はできなかった。

- ・ 慣れ親しんだプロセスをいつも使っている。
- ・ XDDP の成果物をそのプロセスに当てはめてしまった。

=====

c) 現在定着活動中である

=====

【管理者がかかるときの病気】

■#21 最前線至上症 処方せん #21

- ・ XDDP で成果をあげた有識者を炎上したプロジェクトに応援させるため、後続プロジェクトで XDDP に取り組んでも、相談相手がいない。
- ・ XDDP の有識者が増えない。
- ・ 現場で問題を認識しても相談できる人/組織がないため、有効な対策が実行されず、問題が収束しない。
- ・ 問題に対する技術解を提供できる人/組織がない。

■#22 社内アドバイザー育成軽視症 処方せん #22

- ・ 社外コンサルタントのサポートがあるときは XDDP の導入に成功したが、その後サポートが終了すると従来のやり方に戻ってしまった。
- ・ 社外コンサルタントに代わる社内アドバイザーがおらず、XDDP が組織に定着しない。

【開発者と管理者がいっしょにかかるときの病気】

■#19 肯定眼欠如症 処方せん #19

物事の捉え方がいつも否定的であり、他者の成功事例や新たな設計手法などから優れた点を見出し、改善につなげる活動が行われない。

■#23 プロセス設計意識欠乏症（定着失敗型） 処方せん #23

XDDP を導入した後にプロセスを固定化してしまった。そのため、市場の要求が変わってもプロセスを変更（設計）しなくなり、開発に影響が出ている。

XDDP を導入した後に、

- ・ 作成したプロセスを変更していない。
- ・ プロセスで想定していない作業が必要なときがある。
- ・ 今のプロセスが合っていないのではないかと感じる。

■#24 ブルーカラー症 処方せん #24

新たな手法への取り組みを行わない。

また、行おうとしない。その結果、開発者の技術力が上がらない。

<管理者>

- ・ 業務をルーチンワーク化することで難易度を下げ、誰でもできることを求める

<開発者>

・ 与えられた仕事をこなすことはできるが、状況に応じて自ら判断して仕事を進めることができない。

- #25 先陣恐怖症 処方せん #25
 - ・開発技術・設計技術における現状のやり方に問題があると認識していても「身近な成功事例がない」という理由だけで新たな技術の取り組みを行わない／行おうとしない／認めない／評価しない。
 - ・新しいことに取り組まないため、開発者の技術力が上がらずモチベーションが低下してしまい、次第に疲弊していく。

=====

d) 導入・定着において過去に失敗した経験がある

=====

【導入の失敗の原因】

- b) 「現在導入活動中である」を参照して頂きたい。

【定着の失敗の原因】

- c) 「現在定着活動中である」を参照して頂きたい。

4.2. 該当する組織文化から探す

【新しい技術や手法に取り組むことが多い組織】

【セミナーや講習会で扱った技術や手法に取り組むことが多い組織】

- #5 トレーニング省略症 処方せん #5
- #8 課題形成能力欠乏症 処方せん #8
- #13 慢性ダイジェスト症 処方せん #13
- #16 対策重ね着症 処方せん #16

【プロセスが長い期間固定化されている組織】

【同じ組織に長期間（5年以上）所属している人が居る組織】

- #11 プロセス化石化症 処方せん #11
- #20 プロセス設計意識欠乏症（導入失敗型） 処方せん #20
- #23 プロセス設計意識欠乏症（定着失敗型） 処方せん #23

【設計よりも製造工程に重点を置いている組織】

【手戻りの多くが製造工程やテスト工程で発生する組織】

【多くの障害がテスト工程で発覚する組織】

- #1 レビュー不全症：レビュー軽視症 処方せん #1
- #4 ながら作業無自覚症 処方せん #4

【ドキュメントが陳腐化しているため知識が属人化している組織】

【有識者を常に頼りにしてしまうことが多い組織】

- #2 レビュー不全症候群：有識者依存症 処方せん #2

【新しい技術や技法よりも過去の経験を重視してしまう組織】

■#3 本質理解欠乏症：形式依存症 処方せん #3

【標準化や組織内のルールが長期間変更されない組織】

■#7 標準プロセス呪縛症 処方せん #7

【改善活動が盛んであり、改善に積極的な組織】

■#12 本質理解欠乏症候群：偽装工夫無自覚症 処方せん #12

【自分の会社や組織流にアレンジすることに価値を置きたがる組織】

■#14 本質理解欠乏症候群：我流症 処方せん #14

【新しい技術や技法の習得に消極的な組織】

■#15 せっかち病 習得順序間違え 処方せん #15

■#24 ブルーカラー症 処方せん #24

■#25 先陣恐怖症 処方せん #25

【減点法的な評価尺度が根付いている組織】

■#19 肯定眼欠如症 処方せん #19

【すぐに成果を求める傾向が強い組織】

■#5 トレーニング省略症 処方せん #5

■#15 せっかち病 習得順序間違え 処方せん #15

■#19 肯定眼欠如症 処方せん #19

■#22 社内アドバイザー育成軽視症 処方せん #22

■#25 先陣恐怖症 処方せん #25

【経験者の知見による後方支援体制がない組織】

■#21 最前線至上症 処方せん #21

5. 治療の処方せん



治療の処方せん

病名	#1 レビュー不全症候群：レビュー軽視症		
発症	開発者 管理者	病原	開発者：手順無視 管理者：組織文化不良
症状	<ul style="list-style-type: none"> ・変更3点セットを書いているのにレビューを実施しようとしない。また実施しても誰もレビューに参加しない。 ・時間がないことを理由にレビューを実施しない。 	発症原因	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> ・レビューの効果を理解できていない。 <p><開発者></p> <ul style="list-style-type: none"> ・変更3点セットの目的は、視点の異なったレビュー機会を設けて当人の気付かない問題をチーム力で発見することであることを理解していない。 <p><管理者></p> <ul style="list-style-type: none"> ・もともと一人プロジェクトのスタイルが組織に定着していて、他人の成果物をレビューする文化がない。
治療法	<p><開発者・管理者共通></p> <p>XDDPの変更3点セットは、レビューで合理的に問題を発見することを目的に構成されており、レビューを行わなければ変更3点セットを記述すること自体意味がないことを理解したうえで、レビューを必ず実施する。(レビューはXDDPの生命線である)</p> <p>変更3点セットのレビューを通して、変更要求仕様、変更箇所、変更方法における問題点を実装前に気付けることを実感する。</p>		
合併症	なし		

治療の処方せん

病名		#2 レビュー不全症候群：有識者依存症	
発症	開発者 管理者	病原	開発者：誤った理解 管理者：戦略・企画不良
症状	<p>レビューに存在しない“有識者”を求め、レビュー確認漏れが減らない。</p> <ul style="list-style-type: none"> レビューで有効な指摘ができない理由を有識者不在としている。 レビューの問題を有識者不在とし、それ以上の深堀ができない。 レビューで問題の指摘漏れが繰り返し発生する。 	発症原因	<p><開発者></p> <ul style="list-style-type: none"> 開始前にレビューに必要な知識(レビューアの知識)を考えていない。 問題の原因分析で、「レビュープロセス」まで特定したが、レビューの進め方、成果物、レビューアのいずれの問題なのか追求していない。 <p><管理者></p> <ul style="list-style-type: none"> 有識者を引っ張ってこない。 開発者に有識者がいないことを言い訳にできる環境にしている。 有識者がいないことを受け入れている。 人を育てていない。
治療法	<p><開発者></p> <ul style="list-style-type: none"> “有識者”でないとできないと決めつけるのではなく、そこで確保できる人たちで、レビューできる範囲を確認し押さえていく。 レビュー範囲を手分けしたりして少しずつ対応できる範囲を広げることで、“有識者”に依存しない状態にする。 レビュー漏れなどのエラーに対して具体的に原因と対応策を検討することによって、成果物の構成や書き方により解決できることと、レビューアのスキルが必要なことを区別する。 <p><管理者></p> <ul style="list-style-type: none"> レビューに必要な知識を持った人を引っ張ってくる。 人を育てる。 ドメイン知識を周知できる仕組みを作る。 <p>レビューのやり方を工夫する。</p>		
合併症	「#1 レビュー不全症候群：レビュー軽視症」「#11 プロセス化石化症」を発症する恐れがある。		

治療の処方せん

病名		#3 本質理解欠乏症候群：形式依存症	
発症	開発者	病原	開発者：知識不足
症状	<p>形から入って、その後も理解が深まらず、XDDPの変更3点セットを使っているが、期待した効果が出ていない。</p> <p>XDDP 導入にあたって「形」から入ることは必ずしも悪くないが、そのままの状態が続くと、時間の経過とともに取り組みが形骸化して XDDP の効果が得られなくなってしまう。具体的には「なぜ変更要求と変更仕様を使い分けるのか?」、「なぜ【before / after】で表現するのか?」など「そうすることの本質」を認識しないまま変更3点セット作成を続けているため、効果が得られない。</p>	発症原因	<p><開発者></p> <ul style="list-style-type: none"> ・XDDP が表面的な理解にとどまっており、本質的な部分、思想や考え方が理解できていない。 ・疑問/問題に突き当たった時に「原典」に戻るという発想がなく、安直にこれまでの経験のなかから対応策を引き出だしてしまう。 ・「原典」を読み解く努力が不足している。 <p>原典：参考文献[1], [2]</p>
治療法	<p><開発者></p> <ul style="list-style-type: none"> ・取り組みのなかで「なぜそうするの?」と疑問に思うことがあれば、その場で疑問を書き出しておく。 ・それらの疑問に対して「こういうことを明らかにしたいから」とか「こういう効果があるから」などと自分で答えてみる。 ・疑問について「原典」に立ち戻って何度も読み返し、他のメンバとの議論を通して自分たちの理解を確認する。 ・自分たちの理解を実践し、効果/結果を確かめる。結果が自分たちの理解の証明になる。 		
合併症	「#14 本質理解欠乏症候群：我流症」を発症する恐れがある。		

治療の処方せん

病名		#4 ながら作業無自覚症	
発症	開発者 管理者	病原	開発者：知識不足 管理者：異文化
症状	<p>変更箇所を見つけ次第ソース変更するやり方を変えられず、変更3点セット作成に取り組めない。</p> <ul style="list-style-type: none"> ・実装期間に、設計書に書かれていないことを穴埋めする作業が習慣化している。 ・要求の仕様化が不完全な状態で設計・実装へと進み、その穴埋め作業はコードを実装しながら実施することが、納期を守るうえで当然なやり方であると信じている。 	発症原因	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> ・変更3点セット作成による工数増の不安を払拭できない。 ・そもそもなぜながら作業が問題なのかを知らない。 ・見つけ次第ソース変更するやり方が組織の文化として定着しており、かつそのやり方が最も効率的であると認識している。
治療法	<p><開発者・管理者共通></p> <p>実装と同時並行で仕様化や設計を進めることが効率的であるという、組織や個人に根付いた「ながら作業」習慣の問題点に気づき、これを改める。</p> <p>■教育的アプローチ</p> <ul style="list-style-type: none"> ・XDDPの「ながら作業」に対する考え方、XDDPが推奨している一気にコード修正を盛り込む理由について教育し、ながら作業の効率の悪さに気づく。 ・XDDP教育におけるUSDM演習を通して、USDMのレビューを実施しているときに、この指摘や気づきにもとづく修正は、従来は設計や実装中に実施していたことであることを体験し実感する。 <p>■実践的アプローチ</p> <ul style="list-style-type: none"> ・現状のやり方をPFDで表し、どれだけの「ながら」が発生しているかを知る。 ・実装工程におけるコード生産性を計測し、ながら作業の実態を定量的に把握する。これを起点として、XDDPのプロセスを導入することで実装中の「ながら」が減り、結果的にコード生産性が上がっていくことを気づく。 <p>※コードを触っていると仕事をしている気分になっているものだ。ながら作業であっても、ソースコードをバリバリ変更する人が優秀だとする風土はないだろうか。</p>		
合併症	なし		

治療の処方せん

病名		#5 トレーニング省略症	
発症	管理者	病原	管理者：戦略・企画不良
症状	<p>1) XDDP のトレーニングなしの導入により現場が混乱した。 [混乱の例]</p> <p>①開発期間が長くなった。 ②開発プロセスが不明(不定)になった。 ③バグが多くなった。</p> <p>2) XDDP を導入しても期待した効果が得られない。</p> <p>3) 管理者は手法を紹介するだけで推進しない。</p>	発症原因	<p><管理者></p> <ul style="list-style-type: none"> ・組織の XDDP に対する理解が不十分なまま XDDP を導入し実践した。 <ul style="list-style-type: none"> - 開発者が XDDP を理解する／させるためのトレーニング等、XDDP を導入するための準備を怠った。 ・XDDP の導入を開発者に丸投げした。 <ul style="list-style-type: none"> - XDDP の導入には準備が必要なことを理解していない、理解しようとししない。 - XDDP を紹介するだけで、自ら手法を理解しようとししない。 ・トレーニングなしに新たな開発手法を導入することの危険性を認識していない。 ・トレーニングの必要性が分かっていない。 ・管理者が実践に移るまでの踏むべきステップを検討していない。
治療法	<p><管理者></p> <ul style="list-style-type: none"> ・トレーニングをしないということは、自転車の乗り方の本を読んだだけでいきなり公道に出るようなものである。実際に乗れるようになるためには練習が必要である。 ・XDDP を導入するには、XDDP の原典を読み込んだうえでトレーニングを含めた導入計画を作成する。導入計画を作成する際には、先行チーム(パイロット)を作り XDDP を用いた開発を複数回実践する。この実践した結果をメンバー間で意見交換して導入時の資料の見直しを行い、徐々に適用プロジェクトを広げていくことを推奨する。 <p>導入計画には以下のような内容を記載する。</p> <p>① 課題共有 → 管理者も含め組織の課題と XDDP の有効性を組織で共有する。</p> <p>② 適用準備 (トレーニング) → XDDP 勉強会と XDDP 導入後のプロセスを検討する。さらに他社事例やセミナーなどを利用して啓蒙活動を行う。</p> <p>③ パイロット → 短期のプロジェクト、または一部サブシステムのみを対象として XDDP を適用する。 このとき実施計画書を作成し支援体制や評価項目を決める。 なお、完了後は結果を評価・分析し展開計画に盛り込む。</p> <p>④ 展開計画 → パイロットプロジェクトの結果を基に、XDDP の展開計画を作成する。</p> <p><導入計画の参考情報></p> <ul style="list-style-type: none"> ・派生開発カンファレンス 2012 「XDDP 導入に向けた組織的アプローチの紹介 ～モデルベース (CMM) から問題解決型へ、そして XDDP～」 ・派生開発カンファレンス 2015 「XDDP の組織的導入へのアプローチとノウハウ ～2012 年カンファレンス報告から「その後」～」 		
合併症	なし		

治療の処方せん

病名		#7 標準プロセス呪縛症	
発症	管理者	病原	管理者：運営の硬直化
症状	<p>「組織の標準プロセス」の名の下に異なるプロセスが許容されない。</p> <ul style="list-style-type: none"> ・組織として制定された開発プロセスが有る。 ・制定された開発プロセスは、外部の機関（もしくは内部の有資格者）によって認定／確認されている。 ・制定された開発プロセスは、ある規格で開発組織に必要とされたプロセス「標準プロセス」として組織の開発規定に記載されている。 ・開発規定に開発プロセス（標準プロセス）遵守の記載がある。 ・開発の要求や開発環境が変わってきているが、開発規定に記載されているため（標準）プロセスを変更／更新出来ない。 	発症原因	<p><管理者></p> <ul style="list-style-type: none"> ・標準プロセスを安定化（固定化ではなく）することは必要だが、プロジェクトの特性や市場や開発環境の変化によってプロセスを変化（改善）させるべきであるが、その意識が欠けている。 ・標準プロセスは変えては行けないと思い込んでおり、異なるプロセスが許容されない。 ・プロセスを設計/検証出来る技術を持たない。 ・標準プロセスを変えたときの混乱を収める技術を持たない。 ・変化させたプロセスをシミュレーション（疑似体験）しない（技術がない）。 ・（80～90年代に）プロセスを変化させて市場の要求に応えることを学習しなかったことが尾を引いている。
治療法	<p><管理者> 変化する市場の要求に応え続けられる組織となる。 そのために、</p> <ul style="list-style-type: none"> ・全てのプロジェクトに既成のプロセスを適用するのではなく、それぞれの要求に応じてそれを実現する最適なプロセスに変化させることで市場の要求に応えることが出来ることを組織レベルで理解し実施する。 ・変化を受け入れられない考え方が入り込んだ理由を調べる。 ・標準プロセスとテーラリングの意味を正しく理解する ・発生したバグについて分析を行い、プロセスに起因するバグに対しバグを生じさせないプロセスを検討し標準プロセスを変化させることを促す。 <p>※標準プロセスにこだわっていることに起因するバグを見つけて変化を促す（バグは、プロセスに起因する→プロセスで解消する）。</p> <ul style="list-style-type: none"> ・要求に合わせてプロセスを自在に設計するために、現状のプロセスを表現できる PFD のような技術を習得する。 ・最適化されたプロセスを正しく評価・認定できる仕組みを導入する。 		
合併症	<p>「#11 プロセス化石化症」「#16 対策重ね着症」「#24 ブルーカラー症」を発症する恐れがある。</p>		

治療の処方せん

病名		#8 課題形成能力欠乏症	
発症	開発者 管理者	病原	開発者・管理者：事前検討の不足
症状	<p>現在派生開発を行っており変更量の割にバグが多発している。</p> <p>「XDDP が派生開発に有効」と情報を得たので導入してみたが想定したほどの効果が出ていない。</p>	発症原因	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> ・XDDP で何を解決するのかを明らかにしないまま XDDP 導入に取り組み、導入自体を解決策としてしまった。 ・解決すべき課題と対策の整合が検討されていない。 ・発生している問題の整理・分析を行っていない。 ・分析結果がないので解決しなければならない真の課題が形成できていない。 ・解決課題が明確でないため対策の検討がされていない。 <p>この状態で XDDP を導入してしまった。</p> <ul style="list-style-type: none"> ・XDDP について正しく理解していないため、変更3点セットの役割と効果や必要な書き方を習得しないまま、作成することが目的になっている。 ・変更3点セットで何が解決できるのか理解できていない。 ・変更3点セットを正しく書いたりレビューしたりすることで問題や間違いに気づくことを理解していない。
治療法	<p><開発者・管理者共通></p> <p>現場で起きている問題を整理分析して、XDDP で解決できそうな問題を把握し実施する。</p> <ul style="list-style-type: none"> ・現状を表すデータを収集する。 ・問題となるものを整理(・分類)する。 ・整理したものについて原因分析を行い、問題を解決するための課題を抽出する。 ・課題に対して対策を検討する。 ・検討した対策の中から XDDP 導入で解決されると予想されるものを抽出し対応づけ実施する。 ・正しく対応付けするために <ul style="list-style-type: none"> ・XDDP(USDM・PFDを含む)の本質(思想や考え方)を理解する。 ・変更3点セットの役割と効果を理解する。 <p>※変更3点セットを書くときに気づく問題やレビューで気づく間違いは、いずれも気づこうとしなければ得られない。</p> <ul style="list-style-type: none"> ・変更3点セットに必要な書き方を理解し習熟する。 <p>参考:</p> <ul style="list-style-type: none"> ・プロセスに起因するバグに対しては、バグを生じさせないプロセスを検討しプロセスを変化させることを促す。 ・既存の(標準)プロセスにこだわっていることに起因するバグは、プロセスを変化させ対応する。(バグはプロセスに起因する → プロセスで解消する。) 		
合併症	なし		

治療の処方せん

病名		#11 プロセス化石化症	
発症	開発者 管理者	病原	開発者：運営の硬直化 管理者：使用環境変化
症状	<p>自組織のプロセスの“特殊性”を強調して XDDP を自組織のプロセスに合わせようとした結果、従来のプロセスが残ったままで XDDP の本質から外れる。</p> <ul style="list-style-type: none"> ・ XDDP 導入のためには、自組織の従来のプロセスを変更する必要があるにもかかわらず、従来から慣例的に実施してきている慣れ親しんだプロセスなので、変えようとしなない。その結果、XDDP のプロセスを本質から歪めたかたちで導入してしまった。 ・ 自組織にとって、もはや遵守することの意味を特定できないプロセスになっているにもかかわらず、慣例的に継続されてきているものなので簡単に外すことができない。 	発症原因	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> ・ 既存プロセスが PFD のようなツールで表現（見える化）されていない。 ・ ソフト開発で要求されている環境変化に現状のプロセスがあっていないことを認識できていない。
治療法	<p><開発者・管理者共通></p> <p>プロセスが長年固定化されることの弊害を問題として捉え、合理的なプロセスを設計する必要性に気づく。</p> <p>■教育的アプローチ</p> <ul style="list-style-type: none"> ・ プロセスとその成果物を表現する技術（PFD・DFD など）、プロセスデザイン・プロセスシミュレーションの考え方と得られる効果について教育し、「プロセスは順守するもの」から「要求に応えるためにデザインするもの」への意識改革を行う <p>■実践的アプローチ</p> <ul style="list-style-type: none"> ・ 直近のプロジェクトについて、そのプロジェクトで要求（機能要求以外も含む）されている成果物（最終成果物）を得るためのプロセスを PFD や DFD で表現する。 ・ 固定化しているのはプロセスフレームワークであって、詳細のプロセスは暗黙知で実施しているケースがないか、確認する。 ・ プロジェクト計画作成時に、各サブシステム単位で今回のプロジェクトにおける最適なプロセスデザインを実施することをルール化する。あるいは、この活動自体を組織の改善活動として据え付ける。 <p>※いつまでも化石のように残るプロセスの代表格は、品質問題を起こした際に策定した再発防止プロセス。プロセスは生ものであると認識し、賞味期限があるものとする。設定した「賞味期限」を迎えたら、そのプロセスを継続するかどうかの判断、あるいはプロセス自体のエンハンスを検討するなど、プロセスを化石化させない取り組みをルール化する。</p>		
合併症	「#7 標準プロセス呪縛症」を発症する恐れがある。		

治療の処方せん

病名	#12 本質理解欠乏症候群：偽装工夫無自覚症		
発症	開発者	病原	開発者：知識不足
症状	<p>変更3点セットのフォームや構成をアレンジや改善の名目で変更を加えた結果、本来 XDDP で得られるはずの効果が得られなくなる。</p> <p>【自己流アレンジの例】</p> <ul style="list-style-type: none"> ・変更要求仕様書に「備考欄」を新たに設け、見つけた変更箇所と変更ソースコードを書き残しておく。ソースコード変更時はこれをカット&ペーストして”効率化”と称する。 ・変更要求仕様書のフォームから「説明」欄を削除してしまう。 ・変更要求仕様書の記述で「before 欄」、「after 欄」をテンプレート化してしまう 	発症原因	<p><開発者></p> <ul style="list-style-type: none"> ・XDDP に取り組みだしたが思うようにはかどらず、従来の経験に基づいた（慣れた）方法に基づく打開策を講じようとする。（結果として XDDP の背骨（本質）が崩れてしまう。） ・「なぜそうするのか」「なぜこれが必要なのか」を理解していないため、その取り組みへの必要性を感じない。（やがて「私（たち）にとっては不要」と判断し省略してしまう。）
治療法	<p><開発者></p> <p>【治療上の注意】</p> <p>この病気の特徴は、アレンジや改善を加えている本人は偽装とは思っておらず、あくまでも良かれと思って「工夫」しているつもりでいるという点にある。やり易い方法に安易に「変更しない・させない」考え方やプロセスを設けることが重要となる。</p> <ul style="list-style-type: none"> ・XDDP の技術を十分に修得していない時期には、原則自己流アレンジや改善は行わず、効果を実感できるまで、原典に書かれている事をそのまま実行することを基本とする。 「守・破・離」の考え方を開発者間で共有する。 ・「工夫」としてアレンジや改善を行う際には、都度、それは何を目的とした変更なのか考え方を明らかにし、それが XDDP の本質を崩していないことを確認したうえでアレンジを実行する。 <p>【本質を崩していないことの確認方法】</p> <ol style="list-style-type: none"> a) XDDP に詳しい有識者に都度確認する。 b) 有識者がいない場合は、原典に立ち戻ったうえでチームメンバーどうしの議論で確認する。 		
合併症	「#13 本質理解欠乏症候群：慢性ダイジェスト症」「#14 本質理解欠乏症候群：我流症」を発症する恐れがある。		

治療の処方せん

病名	#13 本質理解欠乏症候群：慢性ダイジェスト症		
発症	開発者	病原	開発者：知識不足
症状	<p>推進役（開発者）が XDDP は良さそうだと判断しフォーマットだけを手に入れ開発現場に導入したが、誰も適切な指示ができなため現場がかえって混乱し期待した効果が得られなかった。</p> <ul style="list-style-type: none"> 既存のプロセスに XDDP の成果物を追加したため作業が増加した。 XDDP の成果物を最後に作成している。成果物の作成が形骸化している。 XDDP の指導者が不在のため誰も成果物をレビューができない。 	発症原因	<p><開発者> 推進役自身も含め、誰も XDDP の本質的な部分（思想や考え方）を理解しないまま導入してしまった。</p> <ul style="list-style-type: none"> 現行プロセスを変更せずに入手したフォーマットを成果物として現行プロセスに追加してしまった。 開発現場に向けて現状の課題と XDDP の導入による効果など、現行プロセスを変更することで開発現場の課題が解決できることを正しく伝えることができなかった。 推進役は開発現場のメンバや上司の協力を得ることなく独自で導入しようとした。
治療法	<p><開発者></p> <ul style="list-style-type: none"> 推進役（開発者）が XDDP の本質（思想や考え方）を理解して組織に浸透させる必要がある。 推進役は XDDP のセミナーへの参加等を通して本質を理解し、現状の開発現場の課題を整理して XDDP で解決できる課題を把握する。 推進役は、開発現場で改善意欲の高いメンバと共に XDDP の導入により効果が期待できそうな組織に適用する。まずは、改善意欲の高いメンバと実践を繰り返し、徐々に適用する組織を広げていく。 推進役は、成果が出た時点で上司に報告する。上司の承認を得たうえで、組織全体への導入計画を立案し、先行で導入を経験したメンバを中心に導入に着手する。 上司からの支援を継続的に得るために、上司にも XDDP のセミナーへの参加等を通じて XDDP の本質を理解してもらう。 		
合併症	「#12 本質理解欠乏症候群：偽装工夫無自覚症」「#14 本質理解欠乏症候群：我流症」を発症する恐れがある。		

治療の処方せん

病名		#14 本質理解欠乏症候群：我流症	
発症	開発者 管理者	病原	開発者：知識不足 管理者：戦略・企画不良
症状	自分たちの旧来のやり方（都合）に合わせて変更要求仕様書と変更設計書を作成してしまい、導入しても効果が出ない。 変更要求仕様書と変更設計書の作成が我流の書き方になってしまっている。	発症原因	<p><開発者></p> <ul style="list-style-type: none"> ・我流で判断するためには判断の対象となる領域に対しての十分な知識が必要であることが前提であることを知らない。 ・XDDP に関する知識が不十分な状態で、従来のやり方に合わせて変更要求仕様書と変更設計書を作成している。 ・すでにエンジニアリング技術として確立されている技法があるにも関わらず理解せずに従来の慣れたやり方で実施してしまう。 ・1つのパターンとして、管理者から XDDP 導入の命令を受け、開発者が納得しないまま取り組んでいる。 <p><管理者></p> <ul style="list-style-type: none"> ・XDDP を導入する目的、目標を十分検討し、啓蒙活動等を行っていないため、開発者が今までの開発手法と混同して使用している。
治療法	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> ・狙い通りの成果が上がっていないことに気付いたら、現在の『取り組み』を整理した上で「原典」に立ち戻る。（「原典」の意図を読み取る） ・その結果、自分たちの『取り組み』のどこが「原典」の意図するところから外れているかを見つける。 ・「原典」から外れているところに対し、どのような改善をすれば「原典」の意図に近づけるかを検討する。 <p>旧来のやり方からいったん離れて「原典」の通り実施する。</p> <p>原典：参考文献[1], [2]</p>		
合併症	「#3 本質理解欠乏症候群：形式依存症」「#12 本質理解欠乏症候群：偽装工夫無自覚症」を発症する恐れがある。		

治療の処方せん

病名		#15 せっかち病 習得順序間違い	
発症	開発者 管理者	病原	開発者・管理者：誤った理解
症状	<p>とにかく XDDP さえ導入すれば派生開発はうまくいくと考え導入したが、下記の事象が発生し、状況が改善しなかった</p> <ul style="list-style-type: none"> 適切な修正箇所を抜き出すためのソースコード解析ができていない 「変更3点セット」による変更箇所や変更内容、変更方法の特定がずれている XDDP の成果物をレビューしても有効な指摘が得られない 	発症原因	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> XDDP を導入するためには、その前提となる必要な技術や習得順序があることを理解していなかった。 変更には設計技術が必要ないと思っている。 <p><管理者></p> <ul style="list-style-type: none"> XDDP の「変更3点セット」さえ導入すれば成果が出せると考えた。 <p><開発者></p> <ul style="list-style-type: none"> 設計に必要な基本的な技術を持っていない
治療法	<p><開発者・管理者共通></p> <p>XDDP を導入するには組織やメンバがソフトウェアエンジニアリングに関する基本的な知識や技術を習得していることが前提である。</p> <p>例えば、ソースコードを読み解くにも設計技術が必要である。</p> <p>前提となる主な知識や技術には以下のものがある。</p> <ul style="list-style-type: none"> スペックアウト（ソースコードを読み解く）技術 仕様書、設計書を書く技術 レビューとテストに関する技術 <p>など</p> <p>また、XDDP を支える技術として USDM（要求を表現する技術）と PFD（プロセスを設計する技術）がある。</p> <p>これらの組織に必要な技術の教育計画を作成し教育を行う。</p>		
合併症	「#13 本質理解欠乏症候群：慢性ダイジェスト症」を発症する恐れがある。		

治療の処方せん

病名		#16 対策重ね着症	
発症	開発者 管理者	病原	開発者・管理者：狭い視野
症状	<p>XDDP の取組みだけでは不安になっていろいろな取組みを重ねてしまった結果、工数がたりなくなりプロセスを省略するなど不完全な取組みになる。 もしくは期待する効果が出ないので取組み自体をやめてしまう。</p> <ul style="list-style-type: none"> ・セミナーやカンファレンス（コンサルティング）で入手した改善策を手当り次第に取込んでしまう。 ・1 施策では不安であるため入手した施策を全て実施（並行実施）してしまう。 ・結果工数が足りなくなってしまう。 ・そして必要なプロセスを省略してしまう 	発症原因	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> ・XDDP によって今起きている問題の何を改善したいのかが分かっていない。 ・今おきている問題の分析が正しく出来ていない ・XDDP を正しく理解していない。そのため XDDP に依って問題の何がどのように改善されるか理解出来ない。 ・発生している問題がどこまで（どの部分まで）改善されるか理解出来ない。
治療法	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> ・管理者は、XDDP で改善できる問題は何なのかを組織レベルで明確にする。 ・各開発者自身はそれを理解し納得する。 ・そのためには、管理者と開発者が協業で派生開発の現場で起きている問題の原因分析を行い、 <ol style="list-style-type: none"> a) 問題が現状のプロセスに起因している b) XDDP とはどんなものか、何が解決出来るのかを理解する c) 当該問題の改善手段として XDDP が有効であることを両者が納得・確認したうえで XDDP の導入を進める。 		
合併症	「#11 プロセス化石化症」を発症する恐れがある。		

治療の処方せん

病名	#19 肯定眼欠如症		
発症	開発者 管理者	病原	開発者：狭い視野 管理者：組織文化不良
症状	物事の捉え方がいつも否定的であり、他者の成功事例や新たな設計手法などから優れた点を見出し、改善につなげる活動が行われない。	発症原因	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> 減点法的な評価尺度が組織文化として根付いており、悪いところのみを取り上げてそこを改善することが目標達成の手段であるという考えに疑いを持っていない。(問題や課題に対してのみ焦点をあてる組織のレビュー文化)したがって、他のPJの良いところを取り込む意識も持っていない。
治療法	<p><開発者・管理者共通> レビューにおいては、否定から入るのではなく肯定から入る習慣を身につける。</p> <p><開発者> レビュー手法として、アジャイル開発で推奨されている KPT を導入してみるの是一种の方法である。(KPT ではまず Keep すべき良い点の整理から入る)</p> <p><管理者> 減点法的な評価尺度を持つ組織</p> <ul style="list-style-type: none"> レビューにおいては、まずは良かったこと・優れていると思われる点に焦点を当てて評価し、これを次にむけての組織のアンカーボルトとしていく改善の流れを定着させる。 		
合併症	なし		

治療の処方せん

病名	#20 プロセス設計意識欠乏症（導入失敗型）		
発症	開発者 管理者	病原	管理者：組織文化不良 開発者：知識不足
症状	開発現場では組織に存在する標準プロセスを変更もせずに利用している。組織の課題を解決するために、XDDPを導入しようとするが、過去から慣れ親しんだ標準プロセスに執着してしまいXDDPの導入はできなかった。 ・慣れ親しんだプロセスをいつも使っている。 ・XDDPの成果物をそのプロセスに当てはめてしまった。	発症原因	<p><管理者></p> <ul style="list-style-type: none"> ・組織全体が標準プロセスは守るもの、順守するものと考えているため、標準プロセスは変えるものではないと考えている。 ・障害は属人性の問題でありプロセスで解決できると思っていない。 ・組織全体がプロセスを設計する意識が希薄。 <p><開発者></p> <ul style="list-style-type: none"> ・標準プロセスを過去に教えられたまま使っている。 ・プロセスは設計するものと考えていない。
治療法	<p><開発者></p> <ul style="list-style-type: none"> ・プロセスを設計する技術を身に付け、プロセスは要件より変更するものと理解する。 ・新たなプロジェクトを立ち上げる際に、プロセスをPFDで作成し、今回のプロジェクトで変更すべき箇所が無いか検証する。 <p><管理者></p> <ul style="list-style-type: none"> ・いままで慣れ親しんだプロセスが最善なプロセスではないことを理解し、プロセスを設計する技術の必要性に気づき標準プロセスの改善に取り組む。 ・開発者にプロセスを設計する技術（PFD）の教育を受講させ、XDDPを導入したプロセスをPFDで図解化する。図解化したプロセスをQCDの観点で非効率な部分が無いか検証（シミュレーション）する。 ・新たに作成したプロセスを一部の組織に適用しQCDの観点で問題が無いか確認する。その後、すべての組織で適用を開始する。 		
合併症	なし		

治療の処方せん

病名	#21 最前線至上症		
発症	管理者	病原	管理者：組織構成不良
症状	<ul style="list-style-type: none"> ・現場で問題を認識しても相談できる人/組織がないため、有効な対策が実行されず、問題が収束しない。 ・問題に対する技術解を提供できる人/組織がない。 ・XDDP で成果をあげた有識者を炎上したプロジェクトに応援させるため、後続プロジェクトで XDDP に取り組んでも、相談相手がいない。 ・XDDP の有識者が増えない 	発症原因	<p><管理者></p> <ul style="list-style-type: none"> ・現場から相談を受けてアドバイスする人や組織を作らなかった。 ・後方支援に優れた人（有識者）を配置する発想が組織にない。 ・現場で先行する人が後から来る人に教えることの有効性を認識していない。
治療法	<p><管理者></p> <p>1. 工数確保（短期策）</p> <p>→最前線支援に投入した有識者に直接問題を相談できる場を持ち現状の問題を少しでも解決する。</p> <p>そのために管理職は</p> <ul style="list-style-type: none"> ・最前線支援に投入した有識者と直接会話（リモート可）できる時間を確保する。 ・最前線支援に投入した有識者の次に理解しているメンバをまとめ役にして、現場で発生している問題を整理させる。 ・まとめ役が整理した問題を有識者と一緒に解決策を考え、その結果を現場に説明させる。 <p>※この説明には、できれば最前線支援に投入された有識者も同席できるようにする。</p> <ul style="list-style-type: none"> ・このような相談を週1回もしくは隔週1回実施できるように調整をする。 <p>2. 体制作り（中長期策）</p> <ul style="list-style-type: none"> ・有識者の知見を、組織に広く貢献できる仕組みや体制を構築する ・有識者が次の有識者を育てる風土を作る 		
合併症	「#2 レビュー不全症候群：有識者依存症」を発症する恐れがある。		

治療の処方せん

病名		#22 社内アドバイザー育成軽視症	
発症	管理者	病原	管理者：組織構成不良
症状	<ul style="list-style-type: none"> ・ 社外コンサルタントのサポートがあるときは XDDP の導入に成功したが、その後サポートが終了すると従来のやり方に戻ってしまった。 ・ 社外コンサルタントに代わる社内アドバイザーがおらず、XDDP が組織に定着しない。 	発症原因	<p><管理者></p> <ul style="list-style-type: none"> ・ XDDP を組織内で定着・推進するための取り組み検討が不十分である。 <ul style="list-style-type: none"> a) 現場の人が習得すれば、自然と全体への水平展開ができると勘違いしている。社内アドバイザーの有効性を理解しない/できない。 b) 社外コンサルタントに頼るばかりで社員育成がおろそかになり、社内アドバイザーを育てていない。(社外コンサルタントはあくまでも「便宜」として利用するものである。『便宜に生きて便宜に死す』という考え方を知らない。)
治療法	<p><管理者></p> <ul style="list-style-type: none"> ・ 導入して完了ではなく、XDDP によって狙った効果が継続的に得られる (=定着) ことが「狙い」であると認識を改める。 ・ XDDP の本質を理解して指導できる社内アドバイザーの重要性を理解し、社内アドバイザーの育成プランを立てる。 XDDP 社外コンサルタントがいるうちに、XDDP 社内アドバイザーを育てる。 ・ 育成した社内アドバイザーを核として、組織に水平展開する体制を構築する。 		
合併症	「#21 最前線至上症」を発症する恐れがある。		

治療の処方せん

病名	#23 プロセス設計意識欠乏症（定着失敗型）		
発症	開発者 管理者	病原	開発者・管理者：誤判断・誤った理解
症状	<p>XDDP を導入した後にプロセスを固定化してしまった。そのため、市場の要求が変わってもプロセスを変更（設計）しなくなり、開発に影響が出ている。</p> <p>XDDP を導入した後に、</p> <ul style="list-style-type: none"> ・作成したプロセスを変更していない。 ・プロセスで想定していない作業が必要となるときがある。 ・今のプロセスが合っていないのではないかと感じる。 	発症原因	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> ・「プロセスは固定化することが標準化である」と思い込んでいるため、プロセスを変更していない。 ・ツール等を導入すると時間の経過とともに不要となったとしてもツールを使うことが当たり前と思い込んでしまう。 ・XDDP を導入できたつもりだが、導入できていない。プロセスを変更し続けることがXDDP の定着であることを理解していない。
治療法	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> ・XDDP の定着とはプロセスを固定することではなく、プロセスを改善し続けることであると認識しましょう。プロセスを改善するためにまずは現在のプロセスを PFD で表現することが大切です。そして、作成した PFD をもとに検討（シミュレーション）を行い、新たなプロセスを定義してプロジェクトを進めましょう。 ・プロセスの概要はほぼ固定として、詳細なプロセス（PFD の第二階層など）以下は毎回プロセスを設計（定義）するというルールを運用することも有効です。さまざまな要求に応じたプロセスがあるはずで、前回の要求と今回の要求を比較することで変化点を抽出し、プロセスを変化させていくことでプロセス設計を継続し続けることができます。 ・「標準プロセス」にはプロセスを見直し、設計し続ける工程を含めることをお勧めします。 		
合併症	「#11 プロセス化石化症」を発症する恐れがある。		

治療の処方せん

病名		#24 ブルーカラー症	
発症	開発者 管理者	病原	管理者：戦略・企画不良 開発者：誤った理解
症状	<ul style="list-style-type: none"> ・新たな手法への取り組みを行わない。また、行おうとしない。その結果、開発者の技術力が上がらない。 <管理者> <ul style="list-style-type: none"> ・業務をルーチンワーク化することで難易度を下げ、誰でもできることを求める <開発者> <ul style="list-style-type: none"> ・与えられた仕事をこなすことはできるが、状況に応じて自ら判断して仕事を進めることができない。 	発症原因	<開発者> <ul style="list-style-type: none"> ・開発者はやりがいに乏しい作業を長期的に強いられることにより思考停止に陥り、モチベーションや技術力の慢性的な低下を招いてしまう。 <管理者> <ul style="list-style-type: none"> ・知的労働と単純作業の違いが明確に認識されていない。 ・組織に人材育成の観点がない（人材戦略がない）
治療法	<開発者・管理者共通> <ul style="list-style-type: none"> ・「(考えなくても・誰でも) このとおりにやればOK」といったような、プロセスや手法の固定化・ルーチンワーク化を行わず、技術や環境の変化に応じてプロセスや手法を変化させる。 ・モチベーションや技術力の向上のために「新しい技術」や「不足している技術」の修得にかかる時間を確保する。 ※XDDPの真の狙いは、XDDPで派生開発の現場を合理化して技術者がキャリアを磨く時間を確保することにある		
合併症	「#7 標準プロセス呪縛症」を発症する恐れがある。		

治療の処方せん

病名		#25 先陣恐怖症	
発症	開発者 管理者	病原	開発者：狭い視野 管理者：戦略・企画不良
症状	<ul style="list-style-type: none"> 開発技術・設計技術における現状のやり方に問題があると認識していても「身近な成功事例がない」という理由だけで新たな技術の取り組みを行わない／行おうとしない／認めない／評価しない。 新しいことに取り組まないため、開発者の技術力が上がらずモチベーションが低下してしまい、次第に疲弊していく。 	発症原因	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> リスクを取って、新しいことに挑戦する勇気がない。 <p><開発者></p> <ul style="list-style-type: none"> 他者の後追いのみに注力しているため、新しい技術を導入する際の取り組みがイメージできない。 <p><管理者></p> <ul style="list-style-type: none"> 失敗による余分なコストがかかることを過度に恐れている。その結果、失敗の可能性がある新規取り組みを行わず、既に成功している事例を後追いで取り組むことに終始している。 教育といった必要な投資まで削減してしまった。
治療法	<p><開発者・管理者共通></p> <ul style="list-style-type: none"> 失敗しても大きな影響にならない範囲でパイロット的な導入を行い、体験して学ぶサイクルを回して実績を積み上げ、失敗に対する恐怖心を取り除く。 <p><管理者></p> <ul style="list-style-type: none"> リスクマネジメントを行い、成功・成長するためにリスクを取るという考え方を身につける 		
合併症	「#24 ブルーカラー症」を発症する恐れがある。		

6. 予防の処方せん



予 防 の 処 方 せ ん

生活習慣	分類	価値観不良 異文化の理解ができていない・組織不良・安全意識不良など
	説明	組織が持つルールへの遵守に囚われている、自らの価値観を疑おうとしない
原因となる行動パターン		エンジニアとしての自覚と基礎知識修得に対する取組みが消極的。標準プロセスを改善していくという意識の欠落
発症者		管理者
予防可能な病名		<ul style="list-style-type: none"> ・#1 レビュー不全症候群：レビュー軽視症 ・#4 ながら作業無自覚症 ・#19 肯定眼欠如症 ・#20 プロセス設計意識欠乏症（導入失敗型）
予防治法		<p><共通></p> <ul style="list-style-type: none"> ・新たな手法/技法を手に入れるためには、ソフトウェアエンジニアリングの理解は不可欠であることを認識しましょう。（基礎知識がないのに、応用知識は理解できない） <p>※「基礎」には、現状の作業（プロセス）を表現することが含まれます。</p> <ul style="list-style-type: none"> ・標準プロセスは要件や時代の変化と共に変化するものと認識しましょう。 <p><#1 レビュー不全症候群：レビュー軽視症></p> <ul style="list-style-type: none"> ・短い開発期間と「部分理解」の制約のなかで作業を強いられる派生開発においては、レビューはまさに“生命線”であることを理解しましょう。 ・変更3点セットを作成する最大の目的は、視点の異なったレビュー機会を設けて、自分の気付かない問題を組織のちからで早期に発見することが狙いです。 ・成果物をレビューするなかで成果物の構成や書き方が工夫された箇所を見つけて褒める（肯定する）ことも大事です。レビューは組織の文化を変えるための重要な布石になることを理解しましょう。 ・レビューアの育成も導入計画に入れておくことをお勧めします。 <p><#4 ながら作業無自覚症></p> <ul style="list-style-type: none"> ・管理者自身がエンジニアとしての自覚を持つことが重要です。エンジニアであるということは、今より良くなる何かを求める探求心を持つということです。 ・ソフトウェアエンジニアのブルーカラー化（言われたことしかやらない）を防ぐためにはどうしたらよいか、管理者としての考えや取組みを持つことを心がけましょう。（知識労働者たれ） <p><#19 肯定眼欠如症></p> <ul style="list-style-type: none"> ・新たな手法/技法、成功事例導入についての議論の場や、異なる価値観がぶつかり合うレビューの場においては、否定から入るのではなく、それらをまずは肯定的な眼で捉えてみようとする意識を持ちましょう。

<#20 プロセス設計意識欠乏症（導入失敗型）>

- ・標準プロセスは組織で定期的に見直しをする習慣を持ちましょう。見直しする場合は、プロジェクトでテーラリングした情報を記録しておくことをお勧めします。この記録した情報が見直しの INPUT になります。
- ・プロジェクト計画時にリーダーが作成したプロセスを「機能的合理性（品質の観点）」「経済的合理性（生産性の観点）」の観点でレビューしましょう。プロセスをレビューすることにより導入した目的をリーダー伝えることができます。
- ・プロジェクト進行中に課題が発生した時にプロセスを見直し、プロセスに改善すべき点がないか検討してみましょう。

予 防 の 処 方 せ ん

生活習慣	分類	企画不良 組織構成の不良・戦略や企画の不良など
	説明	企画や計画そのものに問題があっても、十分な準備・検討もせずに取り組んでしまうため、問題が発生すると経験者を頼ってしまう。経験者がいないと経験者をでっちあげる。 企画や計画を立てずに施策の実施を現場に丸投げするケースもある。
原因となる行動パターン		<ul style="list-style-type: none"> ・計画を重要視していないため、計画時にレビューを実施していない。 ・まずは実際に作業して問題が発生してから対応を考えている。 ・施策の実施に関わろうとせず現場任せにする。
発症者		管理者
予防可能な病名		<ul style="list-style-type: none"> ・#2 レビュー不全症候群：有識者依存症 ・#5 トレーニング省略症 ・#14 本質理解欠乏症候群：我流症 ・#21 最前線至上症 ・#22 社内アドバイザー育成軽視症 ・#24 ブルーカラー症 ・#25 先陣恐怖症
予防法		<共通> <ul style="list-style-type: none"> ・原典に立ち返ることを習慣化しましょう。 <#2 レビュー不全症候群：有識者依存症、#5 トレーニング省略症、#14 本質理解欠乏症候群：我流症、#22 社内アドバイザー育成軽視症> <ul style="list-style-type: none"> ・人材を育てる組織文化を持ちましょう。 ・現場ごとの課題を明確にし、組織的に改善を準備・検討・推進する仕組みを持ちましょう。 ・現状の状態を定量的な数値で把握しましょう。数値に基づき何をどう改善していくかを明確にし、実施しましょう。 ・コンピテンシーアセスメントを導入し、組織として中長期的な人材育成戦略を立案・遂行しましょう。 ・現状維持は衰退であることを理解し、技術の動向に注目し、組織にとって有用な技術を取り入れる準備やトレーニングを心がけましょう <#21 最前線至上症> <ul style="list-style-type: none"> ・XDDP 経験者や、能力のある優秀な人材が持つノウハウを組織に広く展開できる仕組みや組織構成を考えましょう。 <#24 ブルーカラー症、#25 先陣恐怖症> <ul style="list-style-type: none"> ・プロジェクトや組織運営において、リスクマネジメント（リスクの抽出/対応策の立案/実施）を定着させましょう。

予 防 の 処 方 せ ん

生活習慣	分類	誤判断（開発者） 狭い視野・誤った理解・間違った認識・状況に対する誤判断
	説明	発生した事象から対策を考えるため、根本原因が特定できず同じ事象が繰り返し発生してしまう。
原因となる行動パターン		根本原因を特定する考えを持っていない。エンジニアとしての視野が狭く、技術や効果を正しく理解しようとしなない。
発症者		開発者
予防可能な病名		<ul style="list-style-type: none"> ・#2 レビュー不全症候群：有識者依存症 ・#19 肯定眼欠如症 ・#24 ブルーカラー症 ・#25 先陣恐怖症
予防法		<p><#2 レビュー不全症候群：有識者依存症></p> <ul style="list-style-type: none"> ・発生した事象から解決すべき問題と真の原因を追究する組織文化を持ちましょう。 ・問題分析の方法を1つ以上持ちましょう。 ・対策を実施した際の効果を定量的/定性的に把握する仕組みを持ちましょう。 ・全ての問題を一挙に解決するのではなく、解決すべき範囲および対応順位を明確にし、段階的な対策導入も検討しましょう。 <p><#19 肯定眼欠如症></p> <ul style="list-style-type: none"> ・できたこと・良くなったことを大事にしましょう ・導入した手法や技術の効果をレビューする初期の段階では、狙いどおりの効果が表れなかったり、運用や活動面でうまくいかなかったりすることが起こりえます。しかし、マイナスの部分のみに注力するのではなく、まず何ができたのか、前よりもここが良くなったという点に眼を向けてそこを伸ばすことを重視したレビューを心がけましょう。 <p><#24 ブルーカラー症、#25 先陣恐怖症></p> <ul style="list-style-type: none"> ・新しい技術を導入する際には、小さく実施し、経験から学んだ改善点を素早く積み重ねていくサイクルが効果的であることを認識しましょう。

予 防 の 処 方 せ ん

生活習慣	分類	誤判断（共通） 狭い視野・誤った理解・間違った認識・状況に対する誤判断
	説明	導入する対策プロセスがどのようなものか正しく理解していないために使い方の判断を誤る。
原因となる行動パターン		新技術導入に際しての PDCA が回せていない。
発症者		開発者・管理者共通
予防可能な病名	<ul style="list-style-type: none"> ・#15 せっかち病 習得順序間違え ・#16 対策重ね着症 ・#23 プロセス設計意識欠乏症（定着失敗型） 	
予防法	<p><#15 せっかち病 習得順序間違え></p> <ul style="list-style-type: none"> ・焦って導入しても現場が混乱してしまいます。導入前に「落ち着いて」導入の目的や効果を組織全体で考え、導入計画を作成しましょう。 ・導入するためには組織やメンバにはソフトウェアエンジニアリングに関する基本的な知識や技術を身に付けている必要があります。 <p><#16 対策重ね着症></p> <ul style="list-style-type: none"> ・発生した事象から真の原因を追究する組織文化を醸成しましょう。 ・対策を検討する場合、その対策を実施した際の効果を定量化（または定量化）しておきましょう。 ・全ての問題を一挙に解決するのではなく、まずは効果が期待できそうな部分（XDDPの一部ではなく開発プロジェクトの一部）から段階的な対策（導入など）も検討しましょう。 ・プロジェクトの終了時に、効果の結果を定量化（定量化）したものと比較（効果を確認）しましょう。 <p><#23 プロセス設計意識欠乏症（定着失敗型）></p> <ul style="list-style-type: none"> ・PFD の教育を行い、自分の作業を表現する技術を広めましょう。その後、PFD で自分の仕事を表現し、仲間同士で意見を言う情報交換会を定期的で開催することがおすすめです。 ・プロジェクト開始時に PFD を書き、終了時に PFD の振り返りを行いましょう。 ・肯定眼（※）を持って他の人の PFD レビューを繰り返したり、自分の PFD レビューを受けたりすることで、プロセスを改善させていくことを身につけましょう。 <p>※「肯定眼」とは他人の意見・行動を肯定的に捉えて、良い点を見つけていくことです。</p>	

予 防 の 処 方 せ ん

生活習慣	分類	調査・検討の不足 事前検討不足・環境調査不足・仮想演習不足など
	説明	決定に至るまでに、十分な検討を行うこと無しに対策に取り組んでしまう。
原因となる行動パターン	解決すべき課題と対策の整合が検討されない。 (実施(導入)する対策で、何を解決するのか明らかにしないまま対策を実施してしまう)	
発症者	開発者・管理者共通	
予防可能な病名	・#8 課題形成能力欠乏症	
予防法	<p><#8 課題形成能力欠乏症> XDDP を導入する前に現場の問題を整理して、XDDP で解決できそうな問題を把握できるようになりましょう。 そのために、次の事柄を組織として理解し実現しましょう。</p> <ul style="list-style-type: none"> ・問題の整理分析結果から課題を抽出し対策を検討する習慣を組織で理解し、認めるようにしましょう。 ・XDDP の本質および XDDP における変更 3 点セットの作成・活用で何が得られるのかを理解し、正しく活用できるスキルを身につけましょう。 ・XDDP で解決できるもの、そうでないものがあることを理解し、解決できないものについてもその解決策を検討しましょう。 ・プロセスを表記する技術を身につけ、 <ol style="list-style-type: none"> 1) 現在実施しているプロジェクトのプロセスを表現してみましょう。 2) これから実施するプロジェクトの最終成果物を得るためのプロセスを設計してみましょう 3) 表記されたプロセスをレビューする技術を身につけましょう。 ・プロセスに問題があると、バグの発生・スケジュール遅延が起こることを理解しましょう 	

予 防 の 処 方 せ ん

生活習慣	分類	無知 伝承無視・知識不足など
	説明	必要な知識、前提となる知識を理解することもなく、業務に取り組んでしまう。
原因となる行動パターン	<ul style="list-style-type: none"> ・影響の分からないまま、定められた手順を変更してしまう。 ・ものごとを本質から理解しようとする取り組みを行おうとしない。 ・新しい事に取り組むが成果がでないため、本質の理解が不足している状態でアレンジを加えようとしたり、周囲に受け入れてもらえるような方法を安易に選択したり行動する。 	
発症者	開発者	
予防可能な病名	<ul style="list-style-type: none"> ・#3 本質理解欠乏症候群：形式依存症 ・#12 本質理解欠乏症候群：偽装工夫無自覚症 ・#13 本質理解欠乏症候群：慢性ダイジェスト症 ・#14 本質理解欠乏症候群：我流症 	
予防法	<p><共通></p> <ul style="list-style-type: none"> ・XDDP を導入するための企画を検討する前に、XDDP の本質（思想や考え方）を理解することが重要です。この本質を理解するために原典を読み本質を正しく理解しましょう。また、本質を理解するためにセミナーや導入された方の話を聞くこともお勧めします。 <p><#3 本質理解欠乏症候群：形式依存症></p> <ul style="list-style-type: none"> ・「原典」を繰り返して読み返すことで、原典を読み解く力「読解力」を身に着けましょう。 <p><#12 本質理解欠乏症候群：偽装工夫無自覚症></p> <ul style="list-style-type: none"> ・XDDP に対する理解や経験が浅い状態での「工夫」は、実は自分たちが「やり易い」従来のやり方に近づけている場合が多く、その「工夫」と称しているものが XDDP で効果を出すために不可欠な本質を崩す「改悪」になることには気付けないものである、ということを認識しましょう。 ・XDDP のプロセスや技術要素において、「XDDP では、なぜそうするのか？」を正しく理解できない箇所へのアレンジは避けましょう。 <p><#13 本質理解欠乏症候群：慢性ダイジェスト症></p> <ul style="list-style-type: none"> ・XDDP 導入の目的を明確にして開発現場のメンバと上司にも導入を支援してもらいましょう。 <p><#14 本質理解欠乏症候群：我流症></p> <ul style="list-style-type: none"> ・エンジニアリング技術として確立されている技法への知識を手に入れましょう。 ・XDDP の取り組みを変更する際には「原典」の意図と外れていないか、その変更はどのような影響を与えるか、を意識して検討しましょう。もし「原典」の意図との差異が説明できない場合には変更自体を行わないようにするか、再検討しましょう。 	

予 防 の 処 方 せ ん

生活習慣	分類	組織運営不良（管理者） 運営の硬直化・管理の不良・構成員の不良など
	説明	周囲の要求が組織運営に変更を求めても、組織の独自性を強調し組織の運営を変更しようとししない。
原因となる行動パターン	現在の標準プロセスが組織に合致していると思い込んでいるため、標準プロセス以外を受け入れようとししない。	
発症者	管理者	
予防可能な病名	・#7 標準プロセス呪縛症	
予防法	<p><#7 標準プロセス呪縛症> XDDP を導入する前に次の事柄を組織として理解し実現しましょう。</p> <ul style="list-style-type: none"> ・要求/要求管理を行う仕組みやプロセスを獲得しましょう。 ・組織標準のプロセスがある場合、プロセスを今回の要求に合わせてテーラリングするという考え方が組織で理解され、認められるようにしましょう。 ・自組織の特殊性を明確にするために PFD のような技術で自組織の現行プロセスを明確にしましょう。 ・また問題プロセスや成果物も特定しましょう。 ・XDDP の真の目的「今回の要求をプログラムのデグレードを起こさずに最小の投資で実現し余った時間を自身のスキル向上に使う」が組織内で共有されるようにしましょう。（不当応援は排除してください） ・自身のスキル向上のみならず、組織が求める方向へのスキル習得に使いましょう。 	

予 防 の 処 方 せ ん

生活習慣	分類	組織運営不良（共通） 運営の硬直化・管理の不良・構成員の不良など
	説明	組織のルールは遵守第一、変化に応じた改善・改革活動を回していない。
原因となる行動パターン		標準プロセスを変えない、標準プロセス以外のものを受け入れない。
発症者		管理者
予防可能な病名	・#11 プロセス化石化症	
予防法	<p><#11 プロセス化石化症></p> <ul style="list-style-type: none"> ・組織で要求/要件管理を行う仕組みやプロセスを持ちましょう。 ・組織標準のプロセスがある場合、プロセスをテーラリングするという考え方が組織で理解され、認められているようにしましょう。 ・自組織の特殊性を明確にするために PFD で自組織のプロセスを明確にしましょう。また問題プロセスや成果物も特定しましょう。 ・XDDP の真の目的（効率化によって余裕ができた時間を自身のスキル向上に使う）を組織内で共有しましょう。（不当応援の排除） 	

予 防 の 処 方 せ ん

生活習慣	分類	環境変化への対応不良 使用環境の変化・経済環境への変化など
	説明	当初の条件が時間の経過とともに変わっても、変化への対応を実施しようとし ない。
原因となる 行動パターン	プロセステーラリングという考え方を持っていない、テーラリングを実施 していない。	
発症者	開発者	
予防可能 な病名	・#11 プロセス化石化症	
予防法	<p><#11 プロセス化石化症></p> <ul style="list-style-type: none"> ・XDDP の導入は「自社のプロセス改善」と心得ましょう ・実施するプロジェクト毎に、XDDP のプロセスを基本として自社で特異なプロ セスを加味したうえでテーラリングを行えるようにしましょう。 ・テーラリングを行う際は、XDDP のプロセス・成果物（特に変更のプロセス・ 変更3点セット）は変更せず、追加のプロセス・成果物で自社のものが適用可 能であれば適用しましょう。 	

付録 1.1 図『「生活習慣」が引き起こす病気と病気同士の関係』

『「生活習慣」が引き起こす病気と病気同士の関係』を図 7.1 に示す。

それぞれの病気の合併症を矢印で示しており、ある病気が発症した際に続けて発症する危険がある病気を確認することができる。

また、L1 生活習慣、L2 病原でまとめ、管理者/開発者に分けたものを図 7.2,図 7.3 に示している。

すべての病気に対して言えることであるが、生活習慣(L1)、病原(L2)を原因として各病気が単独で発症する可能性は常に存在する。

ある生活習慣(L1)、病原(L2)を原因とした病気が発症した場合、同一の原因を持つ他の病気も同様に発症する可能性がある。

また、その病気が矢印に関連している場合、矢印の接続先の病気は合併症としてこれから発症する可能性があり、矢印の接続元の病気は既に発症している可能性がある。

つまり、矢印が入ってきている病気は生活習慣(L1)、合併症という 2 つの要因で起こるため、より発症の確率が高いと考えることができる。

例を示す。

最初に「#14 本質理解欠乏症候群：我流症」が発症したとする。

「#14 本質理解欠乏症候群：我流症」が発症したということは『生活習慣(L1)：企画不良』が原因である「#2 レビュー不全症候群：有識者依存症」「#5 トレーニング省略症」「#24 ブルーカラー症」「#25 先陣恐怖症」すべてが発症する可能性がある。あるいは、既に発症しているのかもしれない。

その中でも特に、「#24 ブルーカラー症」は「#25 先陣恐怖症」「#7 標準プロセス呪縛症」の合併症でも起きる可能性を持つ、発症危険性の高い病気である。

あるいは、以下のようなストーリーも想像できるだろう。

自分のやり方で対応していた管理者がいた（#14 我流症）

それでうまく行っていると感じていたが、いつしか挑戦を恐れるようになった（#25 先陣恐怖症）

しかし、自分のやり方は自信があるため、同じことを実施することが良いと思い込んでいった（#24 ブルーカラー症）

ある時、別のプロジェクトを実施することになったが、今までのプロセスから逸脱したことは行わず、新しい取り組みへ挑戦しないようになっていった（#7 標準プロセス呪縛症）

このように、今後起こりうる病気を想定することでリスク管理への適用も可能であるかもしれない。

これらのリスクに対し、可能であれば先回りして対策をしておきたい。

同一の生活習慣に対する対処であれば予防の処方せんを、合併症への対処であれば治療を参考にさせていただきたい。

「生活習慣」が引き起こす病気と病気同士の関係（共通）

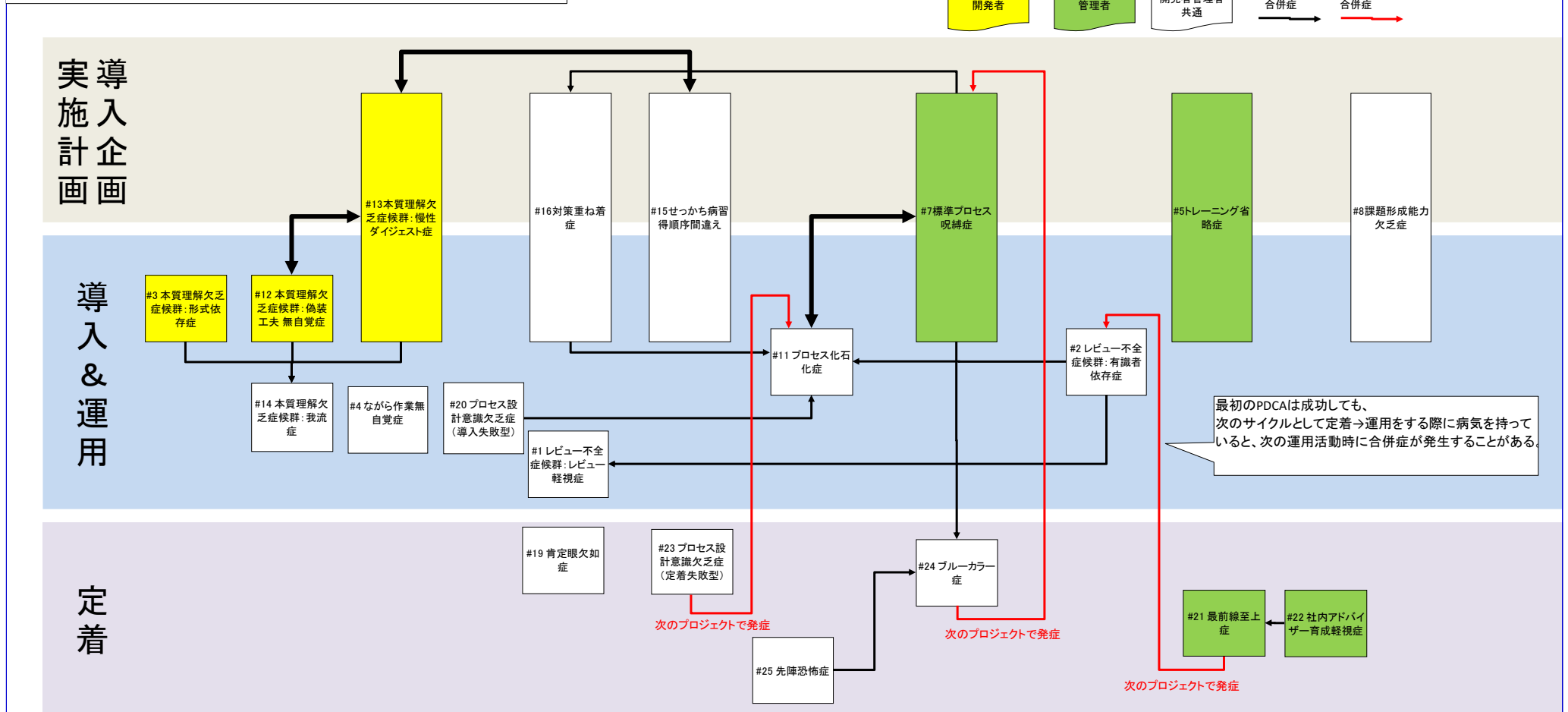


図 7.1 「生活習慣」が引き起こす病気と病気同士の関係（共通）

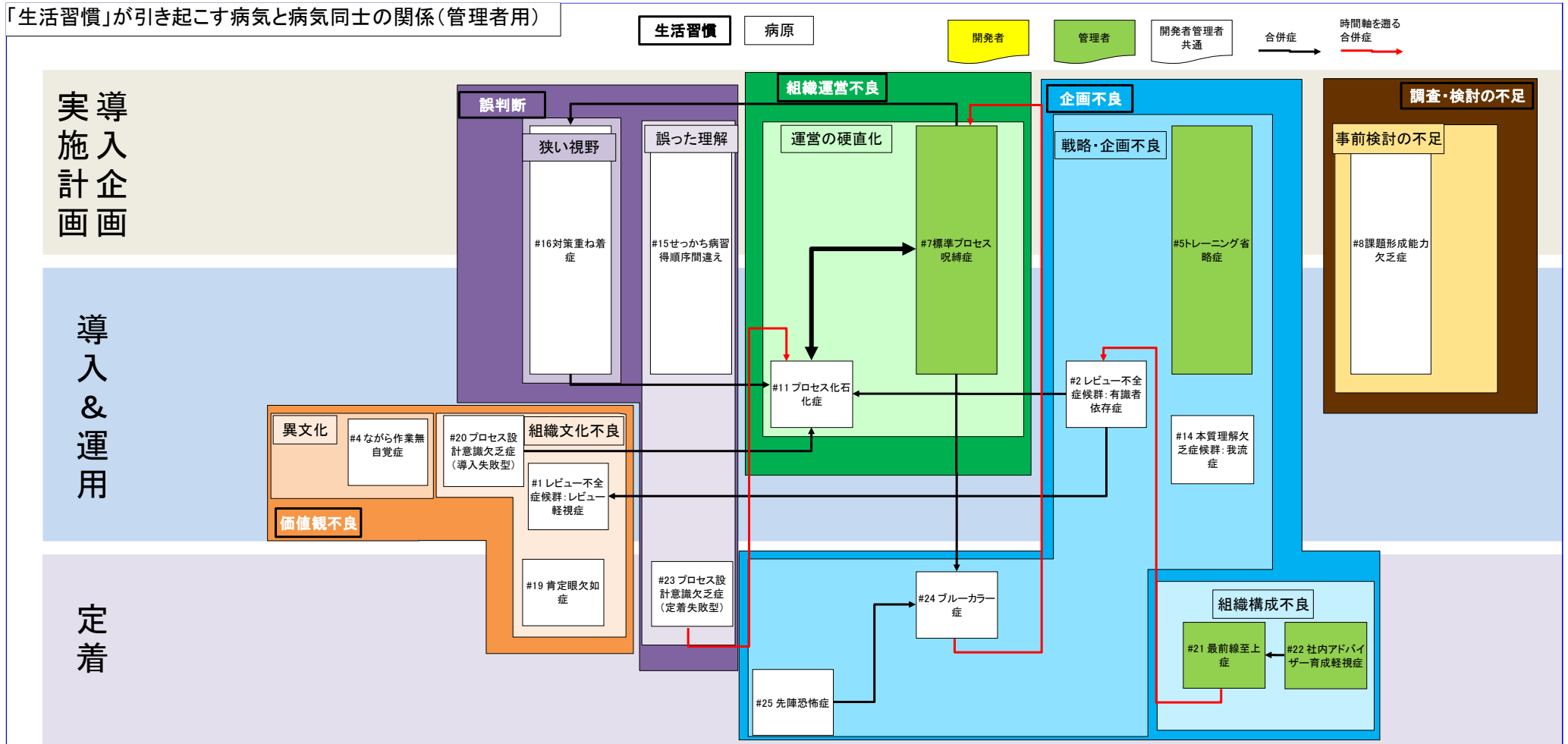


図 7.2 「生活習慣」が引き起こす病気と病気同士の関係 (管理者用)

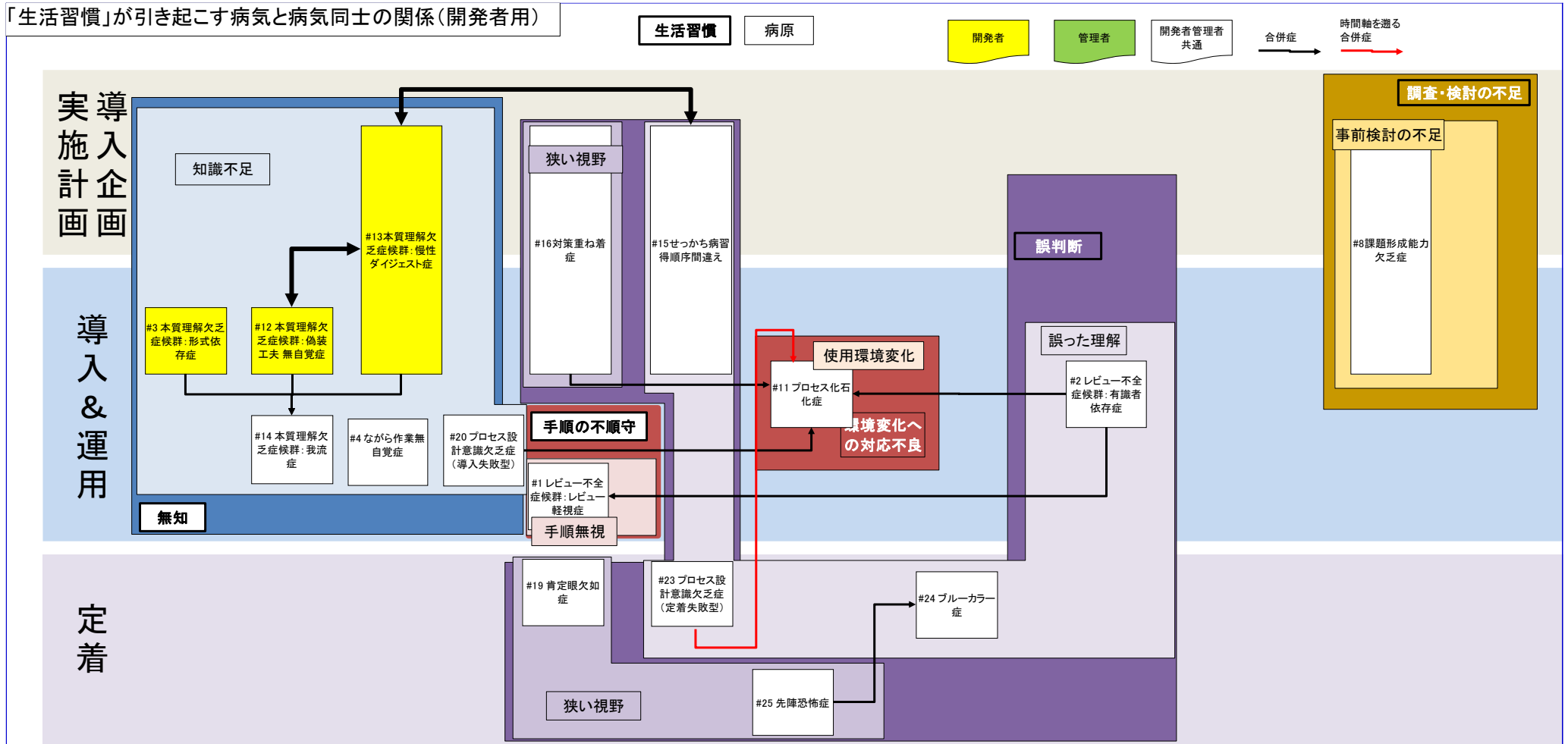


図 7.3 「生活習慣」が引き起こす病気と病気同士の関係 (開発者用)

付録 1.2 プロジェクトリスクマネジメント計画への適用

プロジェクト開始時の開発計画やプロジェクトマネジメント計画を策定する時点において、リスクマネジメント計画はきわめて重要な位置を占める。プロジェクトにおけるリスクとは、プロジェクトで強く約束したこと（QCD など）にインパクトを与えるある事象（リスク事象）が確率的に発生することであり、そのリスク事象はプロジェクトに事実として存在する懸念事項などが要因となって顕在化する。プロジェクトリスクマネジメント計画では、それら発生する可能性のあるリスクの発生を回避したり、インパクトを軽減させるための対応策を立案し、加えて、もし計画した対応策を実施したにもかかわらずリスクが発生してしまった場合に講じる対応策をあらかじめ計画しておく（コンティンジェンシープラン） ことを行う。

上記リスクマネジメントの実施内容を本冊子における「生活習慣」「予防の処方せん」「治療の処方せん」に当てはめてみると、次の関係が成立する。

「生活習慣」： リスク要因

「予防の処方せん」： リスク対応策

「治療の処方せん」： （病気の発症が明らかになった場合の）コンティンジェンシー計画実施策

以上より、「『生活習慣』が引き起こす病気と病気同士の関係」の3つの図を用いて、プロジェクト中に発症する可能性のある病気の予防や発症してしまった病気に対する治療の処方せん、およびその病気が併発する可能性のある病気の予防の処方せんを、プロジェクト開発計画時のリスクマネジメント計画にそのまま適用することができる。（図 7.4）

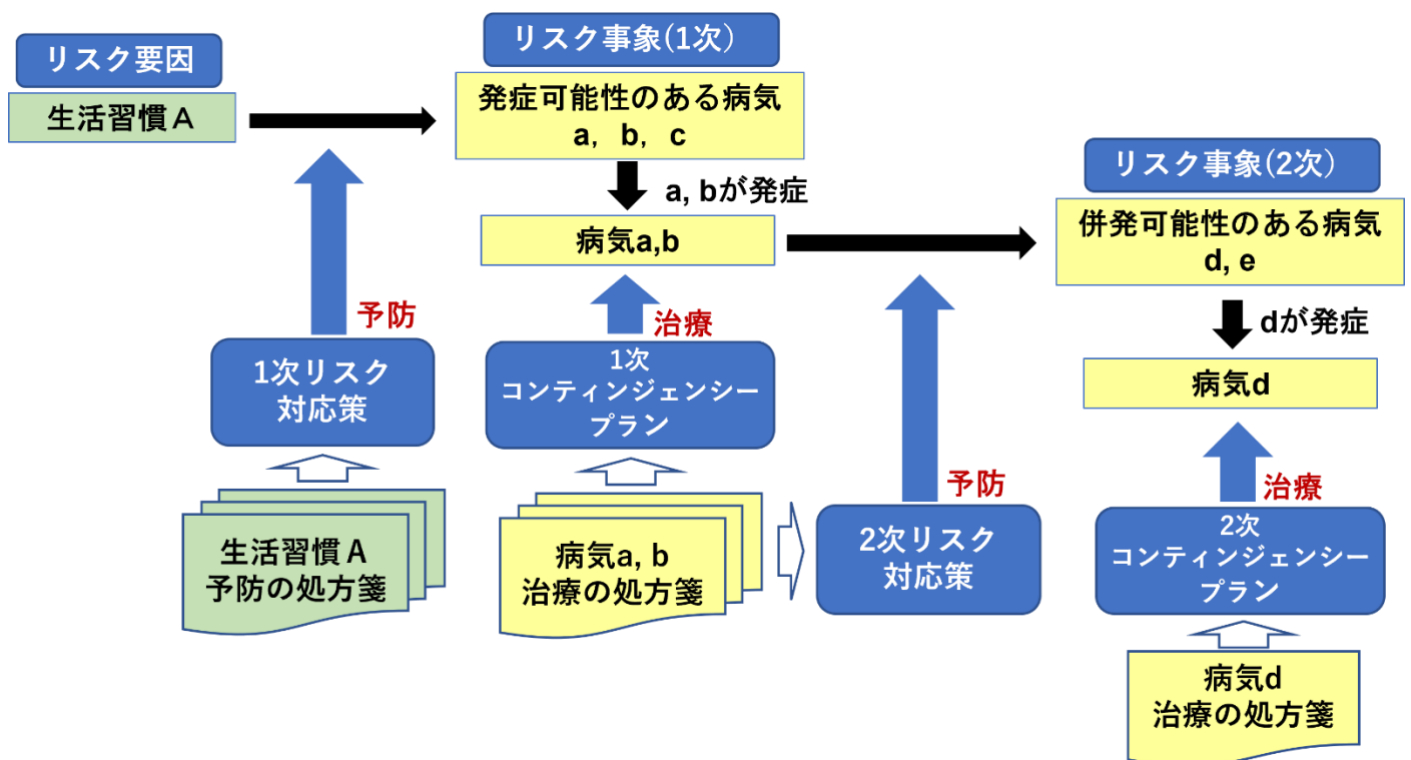


図 7.4 リスクマネジメントへの処方箋の適用

生活習慣 A の予防の処方せんの内容は、生活習慣 A を要因として発症する可能性のある病気 a,b,c の予防策となる。しかし、もしこの予防策実施にもかかわらず病気 a, と b が発症してしまった場合には、そのコンティンジェンシー策として病気 a と b の処方箋がそのまま適用できる。これは同時に、病気 a,b の発症によって合併症として併発する可能性のある病気 d,e の予防策ともなる。

以下の図 xx は、

予防の処方せん「誤判断（共通）」において、生活習慣の説明および原因となる行動パターンの存在が、そのプロジェクトや実施する組織において事実として認められ、かつ「狭い視野」によって引き起こされていると考えられる場合、プロジェクト中に発症する可能性のある病気をリスクマネジメントによって予防・治療を計画する場合の適用例を表したものである。

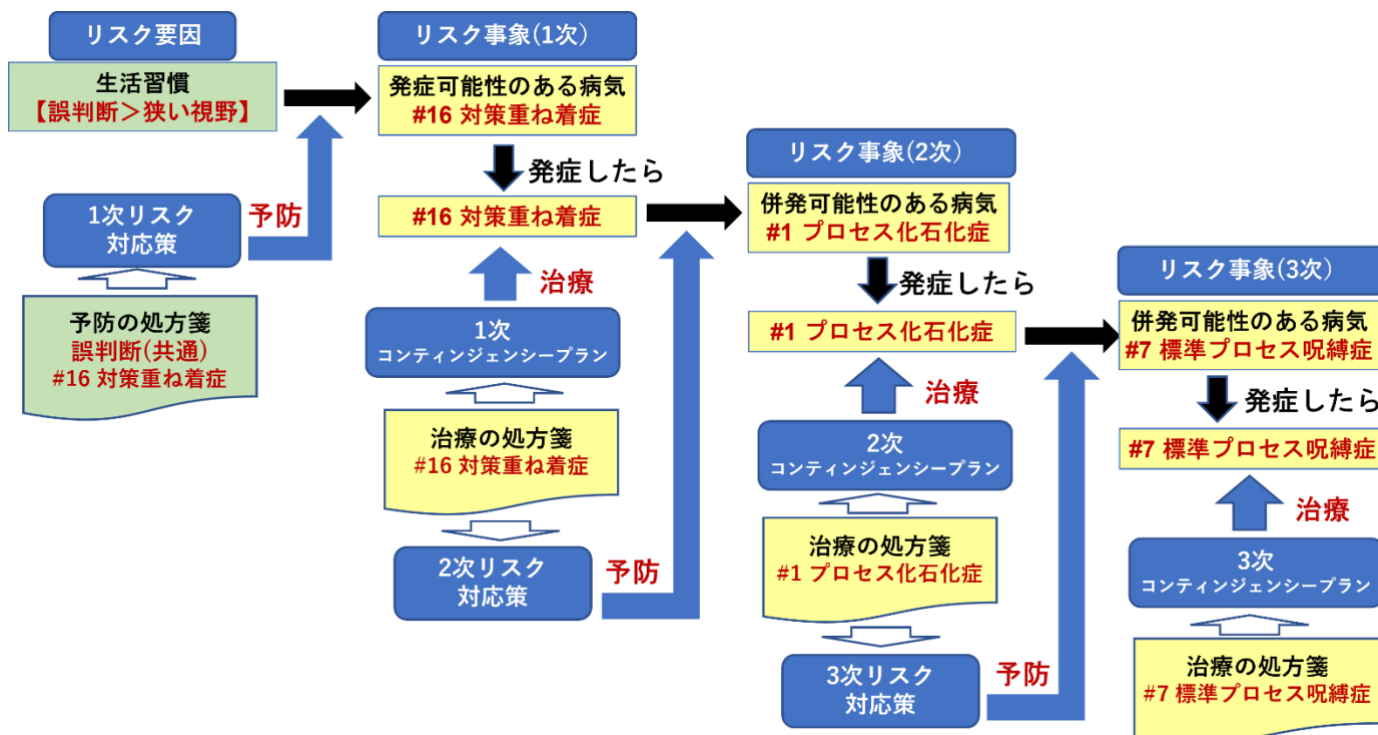


図 7.5 リスクマネジメントへの処方箋の適用例

参考文献

- [1] 「派生開発」を成功させるプロセス改善の技術と極意
清水吉男 著 (技術評論社、2007 年、ISBN978-4774132495)

- [2] [改訂第 2 版] [入門+実践]要求を仕様化する技術・表現する技術 -仕様が書けていますか?
清水吉男 著 (技術評論社、2010 年、ISBN978-4774142579)

- [3] 失敗知識データベース
<http://www.shippai.org/fkd/index.html>
(特定非営利活動法人 失敗学会)

- [4] ソフトウェア病理学 - システム開発・保守の手引
Capers Jones 著 / 島崎恭一・富野壽 監訳 (構造計画研究所、1995 年、ISBN978-4320027497)



XDDP 導入を成功に導く処方せん小冊子

発行日	2018年5月18日	ver1.0
	2021年8月21日	ver2.0
	2023年5月26日	ver3.0

著者 派生開発推進協議会 T22 研究会
斎藤 芳明
梶本 和博
葛西 孝弘
工藤 寛
野沢 友希
高嶋 厚志