Chapter-0 はじめに

本書について

本書は「リモートがデフォルトであるべき」の前提のもと、リモートワークを実現するための考え方とやり方を整理し、提示したものである。無料のオンラインドキュメントであり、誰でも自由に読める。

想定する読者層

オフィスワーカー全般とする。現時点でフルリモートができていない、あるいはできていても会議ばかりしている人すべてに向ける。新人、中堅、ベテランは選ばないし、平社員やマネージャーや経営者も選ばない。すべての人に向ける。本書は未だにフルリモートすらろくにできていない現実を危惧して書いたものである。IT 従事者でなくても理解できるよう平易な解説を心がけたが、オフィスワークとリモートワークの基礎は前提とさせていただきたい。たとえば右記のような用語は解説しない――ハイブリッドワーク、フレックス、Microsoft Teams、SaaS、ドキュメント、1on1。

本書の読み方

もちろん本の読み方など自由にすればいいが、本書としての意図を述べておく。

まず、新しい概念を扱う関係上、どうしても新しい用語を定義することがそれなりにある。既存の曖昧な用語を「こういう意味として使います」などと定めたりもする。うんざりするかもしれないが、扱う内容自体は難しくないし、新しいものを扱う際の宿命でもあるので、ご容赦いただきたい。定義は必ず書くようにしているので、適宜前をさかのぼって参照してもらいたい。

本書の構成は、章が集まるだけのシンプルなものだが、前から順に読んでもらうことを意図している。「まえがき」章、「全体像」章を経て内容に入るようになっており、その内容も基礎となるものから順に取り上げている。なので、後の章では、前の章で使った概念が登場することがある。それも新しい用語だ。というわけで、軽い読み物として読むと苦戦するかもしれない。勉強や調査をするときのような、気合を入れた読み方が少しは要るかもしれない。

本書は読み物として仕上げてはいるものの、読書体験として楽しませる気はない。豊富なストーリーやユーモアは眼中になく、真のリモートワークを実現するためのあれこれを伝えることだ

けを考えて書き上げた。悪く言えば、教科書的なつまらない読み物かもしれない。しかし、リモートワークという、割と誰にでもイメージできる題材を使って、こうするべきとの大胆な理屈を刺激的に展開しているので、興味はそそられるし、興奮もすると思う。楽しんでいただければ幸いだ。

補足資料など

- 不明用語を解説してくれるチャットボット（GPTsでつくってます）
  - https://chatgpt.com/g/g-67dfb57e1ef88191b92f1bbf2f437aee-remotismyong-yu-jie-shuo
- 本書の原稿（Markdown と PDF）
  - https://github.com/stakiran/remotsim

注意事項

- 本書は筆者個人の見解であり、所属組織とは一切関係がありません。
- 本書の内容に基づく運用結果について、筆者は一切の責任を負いかねますので、ご了承ください。
- 本書に記載されている手法名、会社名および製品名は、各社の商標または登録商標です。なお、本書中では ™、®、© マーク等は表示していません。

筆者について

吉良野すた(sta, stakiran)と申します。

- ホームページ
  - 吉良野すたホームページ

お問い合わせ

筆者ホームページのお問い合わせ先からお願いいたします。コメント欄もあります。

更新履歴

- 2025/03/24 v1.0
  - 初版

Chapter-1 まえがき

リモートをデフォルトに。

出社回帰の動きが目立っている

新型コロナの影響で急速にリモートワークが推進されたが、落ち着きを見せた今、出社回帰の動きが目立つ。GAFAM 含めた海外テック企業による Return to Office (RTO) の流れはよく見聞きするし、国内でも2024年12月、LINEヤフーが原則週一 or 月一出社を導入したことで話題になった。

そもそもハイブリッドワークと呼ばれるように、出社とリモートのバランスを取ることが重要、との合意がすでに浸透しているように思う。特に週一、週二など基準を設けて全員に一律に課すケースが多い。ハイブリッドワークとは所定の出社頻度を課すことである、と定義しても良いくらいだ。私は企業の採用ページや OpenWork をよく読むが、フルリモートを公言している企業はテック企業に絞っても一割にも満たないと感じる。

フルリモートでも会議尽くし

またフルリモートであっても日中会議ばかりしているケースは多い。特にマネジメント層にもなると、午前も午後も数件以上埋まっていることが珍しくないだろう。ましてミュート・デイ（一日一度も誰とも会議しない日）の実施、それこそハイブリッドのノリで週に n 回実施するなど不可能であろう。

これは構造的な問題であり、権限を多く持つ者の仕事術が一つしかないためである。具体的には、時間を細切れにして、各細切れの中で用件を進めるしかない。また細切れは一日に何本もあるため、保たせるためには脳の消費を節約せねばならない。人間の認知能力はたかが知れていて、一日中使えるほど潤沢ではない。ではどうするかというと、非言語情報の比重が大きい「一緒に過ごして話す」やり方になる。つまりは会議だ。非言語情報は言語情報と比べて疲れにくいし、感情や雰囲気が使えるので融通を利かせやすく、曖昧や妥協もしやすい。また情報量も豊富なので退屈しないし、社会的動物としての欲求も満たしてくれるため満足感もある。使わない方がおかしいくらいだし、実際、至るところで当たり前に発生する自然現象なのだが……。

リモートなのに会議ばかりしている現状を見ると、何かが違うと思ってしまうのは私だけだろうか。

リモートであるべきだ

出社回帰の動きに伴い、出社 vs リモートの議論も目にする機会が増えてきた。どちらの言い分も真っ当であり、最終的にはケースバイケースですね、臨機応変ですねで終始してしまいがちだ。ここに異を唱えたい。

リモートであるべきだ。今でこそ「出社」の概念が普及しているが、かつては「住み込み」が幅を利かせていた。あるいは住み込みではないにせよ、昭和の時代は長時間残業で家に帰れなかったり、転勤や単身赴任などという非人道的な所業がまかりとおっていたし、今も続いている。

さて、「住み込み」と「出社」とでは、どちらが仕事のパフォーマンスが上がるか、生産性が出るかと言われたら、当然住み込みだろう。物理的に集めて集中してもらった方が上手くいくのは当然だ。しかし我々は人間であって、機械ではない。生活があり、人生がある。だからこそ出社の概念にシフトしてきた。「出社」とは自宅から会社に通う概念を指す。会社に合わせた住居に住むのではなく、自分の生活に寄せた住居に住み、そこから会社に通うというパラダイムシフト（支配的価値観の変化）なのだ。

段階	概念	関係	関係に名前をつける
1	住み込み	ワーク > ライフ	ワークファースト
2	出社	ワーク ≒ ライフ	ワークライフバランス

「出社」の概念は今や当たり前となっている。仕事が捗るからといって住み込みを強要されたら、どう思うだろう。ありえないはずだ。無論、仕事によっては住み込みが必要なこともあるし、自ら希望して住み込みにしてでも働きたい場合は選べば良い。しかし万人に原則的に適用するデフォルトとして住み込みを採用するのはありえまい。仮にオフィスワークの求人で「住み込み必須」「社員寮への入寮が必須」とあれば敬遠するはずである。

話を戻そう。これと同じ構図が今まさに起きている。

あえて言い切るが、「リモート」とは「出社」の次のパラダイムであり、現代においてデフォルトであるべき水準である。

段階	概念	関係	関係に名前をつける
1	住み込み	ワーク > ライフ	ワークファースト
2	出社	ワーク ≒ ライフ	ワークライフバランス
3	リモート	ワーク < ライフ	ライフファースト（あるいはワークインライフ）

出社 vs リモートの議論は不毛だ。リモートに決まっている。リモートであるべきだ。出社に戻そ

うとするのは、住み込みに戻すようなものである。一度上がった生活水準を戻すのは、ハラスメントでさえあると思う。昔はセクハラやパワハラが当たり前だったからあなたも耐えろ、と言うのか。言うまい。このたとえは乱暴かつ卑怯だろうが、リモートの軽視はそれくらいにありえないことであることを声を大にして伝えたいのである。

無論、だからといって今すぐにリモートに移るのも難しい。それでも技術と方法は整っている。もう投資と、工夫と整備の次元でしかない。やるか、やらないか、ではなく、問答無用でやるべきなのである。

現代だからこそリモートが役に立つ。

VUCARD の時代

VUCA（ブーカ）の時代といわれて久しい。変動、不確実、複雑、曖昧の頭文字を取った言葉で、要は正解が無く先も見えず変化も激しい時代だといっている。また多様性の概念は現在 DEIB（デイブ）にまで進んでいる。

段階	略称	英語	日本語
1	-	Diversity	多様性
2	D&I	Diversity + Inclusion	多様性と包括性
3	DEI	DI + Equity	多様性と包括性と公平性
4	DEIB	DEI + Belonging	多様性と包括性と公平性と帰属意識

しかし DEIB（というより DEI）は 2025 年 1 月、トランプ大統領が廃止したことで話題を呼んだ。文脈としては「過激で無駄な」ものの廃止、特にマイノリティの優遇の廃止を決めたように読め、DEI 自体が下火になるわけではないと思いたいが、これ幸いと多様性観点のアピールを取り下げる動きが加速して、結果的に下火になるかもしれない。まさに VUCA だ。

よりビジネスに絡むもので言えば、少子高齢化による労働力の減少は以前から盛んに叫ばれているし、生成AIも台頭し始めている。個人的には翻訳が便利だと痛感しており、言語の壁はさらに低く薄くなると感じずにはいられない。グローバル化も加速するだろう。そうなったときに、出社前提の価値観では人材を活用できない。文化も地理もタイムゾーンも異なる海の向こう側の外国人と出社前提で働くこと自体に無理がある。かといって、言語の壁越えが当たり前になってもなお変わらないのであれば、グローバルな競争についていくことすらできなくなる。少なくとも、できる側とできない側の格差が広がるし、失われた30年も終わらないだろう――

と、安直な妄想を書いたが、現代はそういう時代であろう。私はこれを **VUCARD（ブーカード）** と呼んでいる。

> VUCARD = VUCA + Remote + Diversity

多様性の盛り上がりは、言い換えると生活水準の向上である。また、リモートはすでに述べたとおり第三パラダイムであり、やはり私たちの水準を一段引き上げている。よって、現代は VUCA でありながら生活水準も高い時代と言える。

リモートで対抗する

では VUCA にも高い生活水準にも応えるには、どうすればいいのか。

リモートである。

後者の水準に応えるのにリモートが必要であることは、住み込みからの変遷を見ても明らかだ。しつこいようだが、**リモートは前提である**。その上で、VUCA にも対抗できるやり方と考え方を重ねれば良い。というより、重ねるしかない。私たちは、この難題に挑まなくてならない。一度上がった水準は下げるべきではないからだ。上がってしまった以上は、もう挑むしかないのである。

十分可能であるということ

さて、本書では、そのやり方と考え方とやらを示していく。もちろんかんたんではないし、むしろ難解の域であるが、一言で言えば**とりあえず変革は要する**。変革をとりあえずと言わねばならないくらいの、劇的な転換が必要となる。変革として比較的身近なのはデジタル・トランスフォーメーションこと DX だろう。これはデジタルのやり方と考え方に根本からシフトすることを指す。たとえば紙と電話をゼロにして、アプリだけでも成立できるようにする――と、この DX も序の口にすぎない。他にもオンラインで完結させるとの転換を指すオンライン・トランスフォーメーション（OX）や、各自のペースでコミュニケーションを行う非同期を前提にするアシンク・トランスフォーメーション（AX）も必要となる。DX、OX、AX と、変革を三つも要するわけだ。OX と AX は造語であるが、言葉遊びがしたいわけではない。それほどの難題であると言いたかった。

といっても身構えることはない。別のたとえをしよう。「住み込み」で暮らしていた人が「出社」にシフトする。よりシンプルにたとえたいなら、実家暮らしから一人暮らしをする場合を考えてほしい。このような人、つまりは出社型の生活や一人暮らしを知らない人に、これらを一通り教えるとしたら、どうだろうか。本一冊では足りないだろう。しかし、すでに出社や一人暮らしをしている人からすると、そんなに難しいものではない。面倒ではあるし、適性次第では難易度は大きくブレるが、それでも**現実的に十分可能である**。リモートも同じだ。

本書では「住み込み」の人に「出社」を教えるように、「出社」の人に「リモート」を教える。言葉で教えるため、どうしてもそれなりに用語が登場するし、分量も多いのだが、そういうものである。内容そのものはさほど難しくない。煩雑だし、新しい概念も多いため難しく感じられるが、対象読者にあたる方であれば理解はできるはずである。

本当に難しいと言えば、以下の二点だ。

- 実際に実現していく際は、まずは経営層のレベルで投資しコミットする必要があること（変革の必要条件）
- 今現在当たり前に使っているやり方、考え方、また文化や価値観といったものを自覚して変える必要があること

人は変わらないというが、まさにこの二点は変えることそのものであり、それゆえ難しい。現に現代でもリモートの普及はまだまだだし、私がこのような本を書く羽目にもなっている。

さあ、リモートをはじめよう。

改めて強調しておきたいのは **デフォルト** ということだ。デフォルトとは「基本的に」とか「原則的に」、あるいは「特殊な事情がないならとりあえずは」といった意味である。私はリモートをデフォルトにするべきだ、時代としても自然だ、そのためのやり方と考え方をこれから見ていこう、と言っているのであって、万人が常にリモートを強要する過激派ではない。必要に応じて出社や住み込みはすれば良いし、やりたい人はやれば良い。

それでも、デフォルトというからには、それなりに浸透せねばならない。リモートで問題なくまわるのが前提にあって、そこに出社や住み込みをオプションで重ねるのである。ハイブリッドワークは逆であり、出社が前提で、リモートをオプションにしていたが、そうではない。出社という第二パラダイムは終わったのだ。現代は第三パラダイム、リモートの時代に入ったのである。

以上、デフォルト・リモートの話はこれくらいにしておいて、以降ではこれを支えるやり方と考え方に入っていく。

Chapter-2 全体像

本書の全体像を示す。まずはいくつかの概念を定義し、全体像が三層から成ることを示した後、本書の構成を述べる。

デフォルト・リモート

まず前章で主張した「リモートをデフォルトにしたあり方」を **デフォルト・リモート（Default Remote）** と呼ぶことにしたい。

定義

デフォルト・リモートとは、少なくともリモートワークでは完結するあり方を指す。ここでリモートワークとは単にリモートであれば良いわけではなく、拘束（Bind）を伴わないことを要求する。拘束には場所と時間があり、出社しての打ち合わせは場所と時間の両方を、リモートでの打ち合わせは時間を拘束していると言える。つまり**オンライン会議、ボイスチャット、電話等に頼ったあり方はリモートワークとは言えない。**

拘束の補完

一方で、私たちは人間であり、拘束はある程度は必要である。拘束するからこそ捗ることもあるし、単純な話、よほど特殊な特性でもない限りは病んでしまう。人間は社会的な生き物である。拘束の完全排除は難しい。そこでデフォルト・リモートでは「必要に応じて差し込む」形でカバーする。これをコミュニケーションの注入（Communication Injection, CI）と呼び、後の章で詳しく取り上げるが、出社や会議といった拘束的な営みを強要するのではなく、必要に応じて実施するのである。

もちろん、必要に応じるからといって全員に強要していては意味がない。差し込みに応じる人と応じない人が出てくるだろうし、普段応じない人がたまに応じたりもするだろう。従来では、このような「n通りのあり方」の共存は許容せず、生産性や方向性（ビジョンという言葉はよく使われる）の名目で一つのあり方のみに統一していたが、デフォルト・リモートでは違う。前者の共存を選ぶ。筆者は働き方の多様性（Workstyle Diversity）と呼んでいるが、従来は働き方の多様性が無いとも表現できる。たとえばチームで「毎日朝 9 時から朝会をします」と決めた場合、これは事実上「朝9時の会議に参加するような働き方」の強要に等しい。「自分は夜型で 13 時から仕事したいので参加できないです」「テキストで共有しますね」は通じないだろう。デフォルト・リモートでは、朝9時に参加する人達と、参加しない人達の両方の共存を目指す。後者の 13 時の人は、育児や介護や体質といった事情がなくてもいい。有給休暇と同じで理由など要らず、当たり前に共存できるべきである――と、そう考える。

拘束の無さがデフォルト

したがって、デフォルト・リモートとは、拘束の伴わないリモートワークをデフォルトにしたあり方とも言える。あくまでデフォルトにすぎず、ゼロ拘束を常に目指すものではない。必要に応じて拘束は差し込むし、そもそも拘束されても支障のない者はそれで良い。それこそ従来どおりでも良いし、従来以上に出社や住み込みを増やしたって良い。それでも、少なくともデフォルト・リモートで仕事が成立すること、そして働き方の多様性があってn通りの働き方が共存できることが前提である。

三層からなる全体像

デフォルト・リモートに必要なものは三層で整理できる。

層	単語
3	Technology（技術）
2	Technique and Thought（やり方と考え方）
1	Tenet（テネット）

テネット

テネット（**Tenet**）とは、直訳すると教義や信条であるが、ここでは文化、信念や価値観、メンタルモデルといったものを指す。

デフォルト・リモートは、まずテネットによって阻まれる。ここで阻まれてしまっては、いくらやり方や考え方や技術が揃っていても意味がない。デフォルト・リモートの実現はテネットから始めねばならない。具体的には現状のテネットを自覚することと、デフォルト・リモートに合ったテネットを追加すること。その上で使い分けられるようになることが求められる。

テネットについて、少し立ち寄っておこう。メンタルモデルから行く。メンタルモデルというと小難しいが、**こういうパターンがあればこうするという刺激と反応が定着したもの** と考えればよい。たとえば不特定多数に読まれる場所にアウトプットを出せない人は多いが、これはコミュニケーションを「特定の具体的な誰か」を想定して「その人のために」届けるものと捉えているからであり、メンタルモデルの一種と言える。このメンタルモデルにとらわれている限りは、このようなアウトプットはできない。また、「特定の具体的な誰か」の形成には関係構築が必要であり、手間暇かけることを惜しまない、あるいはかかるものだと無自覚にとらえている。よって関係ができていない赤の他人に対してアウトプットをしたり、率直な発言をしたりすることもできない。一方で、感情が高ぶっていれば可能だし、礼儀作法に基づいて形式的にやりとりすることもできるように、そこまで支配的ではない（より強いテネットが優先されている）。また、先生や上司や顧客といった役割の仮面を被ることでも抗えるだろうし、インターネットで姿が見えない相手に対して遠慮がなくなることもよく知られている（他のメンタルモデルが発揮されている）。

次に信念や価値観は、個人的なものと考えてほしい。たとえば私は何でも自分のやり方でこなしたいという思いが強く、資質診断体系の一つストレングスファインダー（1）では状況の主導権を握りたがる「指令性」が強く出るし、キャリアにおいて譲れない軸をはかるキャリア・アンカー（2）だと自分のやり方で進めたがる「自律性」が強く出る。私にとって「自分の思い通りにやる」ことは最優先事項であるし、仕事でもプライベートでも平然と主張する。しかし、自動的に主張できるほど強いものではなく、上手い落とし所に落とせず妥協となった結果、自分のやり方でできなくて苦しむことも多い――と、個人的な例を挙げたが、信念とは一言で言えば **頻出する**

思考パターン だ。価値観は信念の一種で、価値の良し悪しを決める信念である（とここでは定義しておくが、細かい定義は枝葉でありどうでもよい）。思考でしかないので、無自覚に常に行動に移せるほど強くはなく、パワーや調子次第では平然と鳴りを潜める。しかし自分の軸そのものではあるので、満たされないと不満が募るし、疲弊にもつながる。

最後に文化については、組織や地域から国に至るまで複数の層がある。思考から行動まで、地に張る根のように自らに絡みついており、**基本的に常に発動する。自覚もしづらいし、できたところで他人事になりがちである、つまりは行動に移しづらい**。エリン・メイヤーの異文化理解力（3）をベースに、「日本の文化」の例をいくつか挙げると「一緒に過ごすことで信頼関係をつくっていく」「ネガティブなフィードバックは遠回しに言うし、1対1の状況にして慎重に言う」「議論では対立を回避する」「階層主義的かつ合意形成」「原理や応用よりも周囲が何を言っているかを重視」などがある。一見すると当たり前に聞こえるかもしれないが、どれも文化の一つでしかない。文化が違えば、平社員がいきなり社長にコンタクトを取りに行ったり、上司やチームメンバーに合意を取らず勝手に行動して結果を見せて納得させたりといったことこそが当たり前になったりもする。

改めてまとめると、テネットには文化と信念とメンタルモデルがあり、各々の特徴は次のとおり。

- 文化は常に発動するもの。自覚しづらいし、したところで抗う行動には移せない
- 信念は個人的な最頻出思考パターン。行動に繋げられるとは限らないが、満たせないと溜まる
- メンタルモデルは刺激に対する反応が染み付いたもの。モデルが想定していない行動は取れない、が他の要因による取れることも多い

デフォルト・リモートでは、これらすべてを捉える。既存のものも捉えるし、リモート向けの新しいものも導入する。その上で使い分ける。つまり、なるべくリモート向けのテネットを使うわけだが、既存のテネットを無視し続けても支障が出る。リモート向けをなるべく使いつつ、既存のも引き続き使うという、使い分けのバランスを探らねばならない。もちろん万人が、あるいは同じ人であっても常に同じバランスであるはずもなく、最適なバランスは状況によって異なる。バランスは**探り続けなければならない**。もっと言えば、探り続けられるだけの持続性を持たなくてはならない。

やり方と考え方

やり方（Technique） とは、人間自らが行動することを前提とした手段である。手順、プロセス、メソッド、戦略や戦術など言い方は色々ある。ルール、ガイドライン、プラクティスやマインドといった「現実的に守らなければならないもの」もやり方に含める。守るための何らかの行動が事実上発生するからだ。IT の世界では手作業と呼ぶこともある、つまり自動化されておらず、人間自らが手を動かさないといけない。

次に **考え方（Thought）** とは、捉え方や向き合い方を指す。やり方と違って行動を伴うとは限らない。スタンスやスタイルは考え方だ。パターン、原理原則、構造や標準といったものも捉

え方の一つであり考え方と言える。もう一つ、守っても守らなくてもいい雑なマイルールは考え方になる。逆に、マイルールであっても、権力があって配下に強制できるのなら、配下にとってはやり方になる。強制はされないが、確度の高い著名人や偉人のマイルールは考え方だ。これを権力者が安易に真似して強要すると、強要先にとってはやり方となる。

ややこしいので整理しよう。

まず指定と干渉、の二つの概念がある。

- 手順のように、どう行動するかを直接指定したものがある（指定事項）
- ルールのように、直接は指定していないが結局行動に干渉するものがある（干渉事項）

その上で「守らなければならない感じ（ニュアンスとしては Should 以上）」と併せてマトリックスにできる。

	指定事項	干渉事項
守らなければならない	やり方	やり方
守らなくてもいい	やり方	考え方

つまり基本的にやり方ばかりだが、**守らなくてもいいもので干渉的なものが「考え方」になる**。

考え方は通常、ビジネスでは扱われないか、雑談あるいは助言として扱われることもあるという程度だが、**デフォルト・リモートでは積極的に扱う**。むしろ、各自が各自の考え方をいかに言語化し、出して、議論していけるかが鍵となる。すでに述べたとおり、デフォルト・リモートではリモート向けと従来向け、二つのテネットのバランスを探らねばならないわけだが、ここで考え方を使う。やり方、特に「守らなければならない」ものだけでは情報が足りないからだ。バランスを探るためには、妥協してはいけない。少なくとも情報は出し切らねばならない（出し切るのが辛くて、かつ妥協してもいいのならそれはそれで良い）。情報として見える形で出すからこそ、ではそれを踏まえてどうするかを考えることができる。見える化しない限りは、何も変えられない。

しかし、考え方だけでは先に進めない。実際に行動に移すためにはやり方が必要である。この点も同様、従来の「守らなければならない」やり方だけでは足りない。デフォルト・リモートのためのやり方というものがあり、すでに知られているものから、私が言語化して整理したものまで多数存在する。上手く使いこなしてほしいし、既存を使うだけではなく**必要に応じてやり方をつくる** こともぜひ身につけてほしい。本書ではやり方をつくる部分は取り上げないが、別に建物やプログラムや芸術作品をつくるわけではなく、（やりたいこと次第だが）難しくはない。ひとり暮らしをしていて、不便なことがあれば自分なりに工夫するだろう――そう、つくるというよりも工夫の範疇で済むことが多い。

最大のハードルは、やり方や考え方を自分でつくる、つくってもいいとの発想を持っていない（テネット的に言えば **概念を自製するメンタルモデルが無い** ）ことであり、本書の主目的はこれを崩すことである。本書では私がつくったやり方や考え方も多数登場するが、それでいいのであ

る。きちんと言語化すれば伝えられるし、思考も試行もできるのだ、むしろこれを皆でやるのだ、という点をお伝えしたい。

技術

技術（Technology）とは、やり方や考え方を道具化するために使うものである。

まず道具から論じよう。道具は重要である。たとえば予定を管理する場合、以下のように、道具に頼れば頼るほど便利だろう。

- 1: 頭で覚える
- 2: 紙に書く
- 3: アナログなカレンダーに書く
- 4: 個人用のカレンダーアプリに書く（場所を取らない、修正が楽である等の利便性が強い）
- 5: 共有可能なカレンダーアプリに書く（他の人と共有したり、空きに勝手に突っ込んでもらったりできる）
- 6: グループウェアを使う（チャットやタスク管理など周辺ツールと連携して、日々の仕事を滑らかに管理できる）
- 7: 理想のカレンダーアプリを自分でつくる

下に行くほど高度な道具を使っており、より複雑なことが行えるようになる。代わりに、学ばねばならないものも増える。2: と 3: はアナログの話であり、基本的に誰もが使えるだろうが、それでも紙やペンを使えること、またカレンダーという概念への理解が要求される。4: と 5: はデジタルであり、PC やスマホが使えなくてはならない。6: はビジネスレベルの話で、さらに多くの概念を要する。当然ながらペンで字を書くスキルや、キーボードやタッチパネルで文字入力するスキルも必要だ。7: に至ってはプログラミングを覚えなくてならない。

大別すると、道具には知識とスキルが要求されるわけだが、これらを要求するのは技術である。技術とは道具をつくるための構成要素であり、もっと言えば制約だ。つまり**「道具」には「それ自身を構成する技術」から来る制約が事実上存在しており、これら制約を踏まねばならない**。知識とスキルを使って踏まえるのである。ルールという制約を守らねばゲームが成立しないように、技術から来る制約を踏まえねば道具も使えない。もっともゲームルールほど厳密ではなく、要は制約を守れれば良いだけであるから工夫の余地は大きい。同じ技術であっても、（その技術が持つ制約を踏まえられる）知識やスキルは様々存在するし、日々新しいものが生み出されたり、古いものが忘れ去られたりしている

と、抽象的な物言いになってしまったが、重要なのは以下の二点である。

- 道具を使った方がはるかに便利である
- 道具はただ使うだけでは済まず、技術から来る制約を踏まえねばならない → 知識とスキルが必要である

ようやくデフォルト・リモートの話に入るが、リモートをデフォルトにするために様々なやり方と考え方を整理する。道具無しでの運用もできなくはないが、カレンダーの例だけ見てもわかるように容易ではない。もちろんカレンダーほど単純なものでもなく、すでに述べたとおり、現代でも未だに実現されていない程度には高度でもある。頼らねばならない道具の数も、道具自体の複雑性も **かなりハードとなる**。仮に 30 個のアプリを使うことになったとしよう。30 個すべてについて、いちいち真面目に勉強・練習するだろうか。また、平均して月に1-2回、新しいアプリを使わなければならなくなったとしよう。毎回勉強し直したり、既存アプリから乗り換えたりするのは骨が折れる。この数字は、決して誇張ではない。やや誇張ではあるが、デフォルト・リモートを実現するために、これくらいが要求されたとしても「そんなものかな」と思える程度ではある。

果たして、そんなハードなことができるのだろうか。結論を言えば、できる。基礎素養を身につけた上で、その応用で対処すればいい。人生において、私たちが当たり前にやっていることにすぎない。それを働き方に関してもやるだけだ。

特に重要となるのが **IT リテラシー** である。リテラシーとは「読み書き能力」の意であり、転じて「基礎的な素養」の意味として使われている。IT リテラシーというと、PC やスマホを活用するための基礎的な素養と解釈され、スマホが使えない高齢者やリモートをせず出社と対面をしたがる中高年を「IT リテラシーが低い」と言ったりする――のだが、本書ではこの解釈は使わない。**IT リテラシーを、文字通り「IT（情報技術）を用いた読み書き能力」と定義したい**。読み書きであることに注意したい。いくらフルリモートであっても、オンライン会議ばかりしている人は IT リテラシーが低い。ただ、IT リテラシーと呼ぶと従来の意味と混乱してしまうから、以降では **文字どおりの IT リテラシー（Literal IT Literacy）** と呼ぶことにする。

これでようやく言いたいことが言える。

デフォルト・リモートには、文字どおりの IT リテラシーが必要である。そして、文字通りの IT リテラシーを鍛えるためには、技術から来る制約を踏まえねばならない――つまりは色んな知識とスキルが要求される。手書きのようにペンを使えます、書類の概念がわかります、書類の形式に従って書き込んでいけます程度では済まない。道具には物理的なハードウェアと、電子的なプログラムを制御するソフトウェアがあるが、デフォルト・リモートが使うのはソフトウェアである。国内では IT パスポート試験やプログラミング教育が運用されているが、まさにソフトウェア技術を万人が学ぶことの重要性を示している。技術を知っているからこそ、ソフトウェアを利活用しやすくなる。というより、ある程度知らねばまともに使えない。ソフトウェアはそういう世界なのである。

そういうわけで、本書ではソフトウェア技術という観点からの解説も適宜行う。極力噛み砕いて解説するつもりだが、マニアックだとか縁が無いなどとは思わず、追いかけてもらえると幸いである。

といっても、身構えるほどではない。ゲームや物語を楽しんできた人は多いと思うが、300 作品遊んだからといっても、300 すべてのそれぞれと律儀に向き合っているわけではないはずだ。ゲームにせよ、物語にせよ、基本的にはそれなりの型があり、型を踏まえたバリエーションとして様々な作品が存在する。型が身についておれば、現実的な手間で難なく楽しめる。デフォルト・リ

モートも似たようなもので、馴染みがないから難しく感じられるかもしれないが、慣れの問題にすぎない。

本書の構成

本書の構成を述べる。

本書は第一部と第二部から成る。

第一部は「まえがき」と「全体像」から成る。リモートは出社の次の第三パラダイムであり、デフォルトでリモートであるべきとの強い前提を提示した。また全体像として 3T ――テネット（Tenet）、やり方と考え方（Technique and Thought）、技術（Technology）の三層を示し、最後には文字通りの IT リテラシーが必要であるとも述べた。

これ以降は第二部であり、主要なテーマごとに、デフォルト・リモートの目線で現状と提案を述べていく。適宜 3T も取り上げながら、テーマごとに「大体どうやればデフォルト・リモートができそうかが見えてきた」を感じてもらうことを目指す。本書の解説は唯一解ではないが、一から学んだり考えたりするよりは、はるかに近道となるはずだ。また根っこの意識としても「出社回帰でも仕方ないよね」から「デフォルト・リモートは現時点でも十分現実的にできそうである」「やるべきである」に変わるものと期待する。

Chapter-3 コミュニケーションと情報共有

仕事における最頻出ワードの一つが「コミュニケーション」だろう。近年は「情報共有」の重要性も強調される。しかし、どちらも非常に抽象的かつ多義的な言葉で、議論自体が行われないか、行われても形だけだ。たとえば全員が意見を並べて終わる。あるいは一見すると建設的な議論に見えても、その実、多数派と現行踏襲の圧力をじわじわとかけているだけだったりする。

どちらもテネットのレベルで染み付いているため、パンデミックのような外部のインパクトか、トップダウンによる荒療治でもしない限りは変わりづらい。前者を期待するわけにはいかないし、後者についても、トップに立つ経営者達が原始的なテネットを持っているため期待できない。むしろ出社回帰の流れを見てもわかるとおり、逆行させてしまう。

この流れを食い止めねばならない。コミュニケーションとは何か。情報共有とは何か。今現在私たちが行っているのがどういうもので、ではデフォルト・リモートのためには何が必要なのか。本章にて詳しく掘り下げていく。

前提として、本書では乱暴を承知の上で **仕事とは情報のやりとりである、と定義する**。コミュニケーションも情報共有も、情報のやりとりの仕方の一種と言える。この前提で掘り下げ

る。

コミュニケーションと情報共有の関係

残るかどうか x リアルタイムかどうか

コミュニケーションと情報共有の関係は、以下の二軸を使ってマトリックスにすることで見えてくる。

- 情報が残る、残らない
- 情報をリアルタイムでやりとりする（同期的）、しない（非同期的）

マトリックスの前に、二軸を理解するための例に立ち寄ろう。

□さんと□さんが口頭で会話する場面を考える。

まず会話は、情報としては残らない。「頭の中に記憶があります」は当てはまらないからだ。記憶ではなく記録でなくてはならない。

次に同期性だが、同期的とはリアルタイムであるという意味だ。今回の場面も含め、口頭の会話はリアルタイムであり同期的と言える。お互いの時間と場所を拘束していて、発話された言語的な内容はもちろん、声の雰囲気や身振り手振りといった間接的な情報（非言語情報）も絶えず受け止めながら、瞬発的なやり取りを繰り返す。発言のラリーが激しく続くとは限らないが、意識としては絶えず拘束されている。たとえ無視や軽視を選んだとしても、それはリアルタイムな拘束のもとでそうしているだけだ。

もう一つ、LINE でも Teams でもいいのでチャットを考える。□さんが□さんに向けて質問を出したら、□さんはこれを見て回答を書く。まず情報としては残る。口頭とは違って、やりとりしたメッセージが記録として残るからだ。もっともツール側の制約で指定期間後に消えるか、設定で意図的に消すこともできるが、それでも消すまでは残っているため「残る」と言える。では同期性はどうか。□さんが通知を受け取って、すぐに回答を書き込んだ場合は、リアルタイムなやりとりと言えるだろう。一方で、空いたタイミングで適当に返せばいいノリが形成されているなら、おそらくリアルタイムではなく時間差でやり取りされていくことになる。もちろん、現実的にはアンマッチ――□さんはリアルタイムにやりたいのに、□さんにその気がなくて□さんが不満を溜める、なんてこともよくある。既読スルーの概念はよく知られている。

さて、二軸を理解したところで、マトリックスを示そう。コミュニケーションと情報共有の定義も併せて示す。

	情報が残らない	情報が残る
リアルタイムである（同期的）	1 同期的なコミュニケーション	2 同期的な情報共有
リアルタイムでない（非同期的）	3 非同期的なコミュニケーション	4 非同期的な情報共有

ここで **コミュニケーションとは、情報が残らないやり取りを指す。また 情報共有とは、情報が残るやり取り** を指す。その上で、どちらも「同期的（リアルタイム）にやるか・やらないか」に分かれる。つまり「コミュニケーションと情報共有」なるものは、計 4 つのパターンに分かれている。

左上から右下へ。「情報の養育」へのシフト

左上、1 の同期的なコミュニケーションは現在でも主流だし、何なら自然だ。技術と方法無しに、非言語情報も交えて瞬発的にやりとりするためには、同じ時間と場所に同座した上で同期的に交わすしかない。自然は弱肉強食の世界であり、瞬発性がものをいう。動物である以上、この摂理には抗えない。人間も例外ではなく、適応を促す仕様が組み込まれている。欲求だ。大胆に言い切ってしまうが、**「非言語情報の瞬発的なやりとり」は、欲求として組み込まれていると言っていい。コミュニケーションとは、この欲求を満たすための行為である。** 意思疎通や情報交換などという定義は表面的なものであり、問題を捉えられない。欲求なのだ。食事や睡眠や排泄と同じで、欠乏するから自然と満たすものである。

それもただの行為ではない。コミュニケーションとは相手ありきの欲求充足行為でもある。要求する負荷も相当なもので、自分ひとりで好きな場所とタイミングで食べたり寝たりすればいいものではない。対象者全員を拘束――時間と場所を強制してしまう。

さて、私たちは、こんな行為で情報をやりとりするわけだ。非効率極まりない。人間のスペックなどたかが知れているからだ。同時に一つのことしか喋れないし、聞き取れない。場においても同時に存在できる話題は一つだけだ。加えて時間、体力、認知能力といったリソースの限界もあり、食事や睡眠による補充も必要とする。つまりコミュニケーションに費やせるリソースは限られている上に、根は欲求の充足なのである。必然的に相手を選ぶことになる。認知上の限界はダンバー数（1）として知られているし、現実的に階層組織で搾取的に制御するしか解がないのもご存知のとおり。

こんな原始的な行為に頼りきっていること自体がそもそもおかしいのである。人間の仕様だから仕方ないのではない。人間だからこそ、安易なコミュニケーションの乱用を堪えて、より賢い方法を使うべきだ。私たちには理性があり、技術と方法がある。できる。やる必要が無いと感じるならやらなくてもいいが、本書では許容しない。リモートこそ第三のパラダイムだ。必要性はある。やるべきだ。

左上への偏重を減らすにはどうすればいいか。右下 4 の非同期的な情報共有を目指せばいい。欲求の充足から切り離し、人間のスペックにも頼らないようにすることで、より融通も利くしパフォーマンスも出せるやり方を開拓できる。そのためには情報を残すことと、リアルタイムをやめ

て各自のペースでやりとりできること（非同期性）の両方が要る。お互いが直接やりとりして欲求を満たし合うのではなく、外に出した情報を育てるのだ。

子どもやペットを養育するイメージを浮かべればいい。いわば、外に出した情報を養育する。生物はかまうペースを見誤ると最悪死んでしまうが、情報も同じである。また養育は利己的ではなく（養育対象に対して）利他的に過ごすものだが、同様に情報も同じである。**非同期的な情報共有とは、情報の養育とも言える。**

デフォルト・リモートでは、コミュニケーションよりも情報共有を重視する。**情報の養育をもって仕事を進めるという『仕事のパラダイムシフト』でもあるのだ。**とはいえ、私たちは人間でしかなく、コミュニケーションによる充足も引き続き必要なので、適宜補充しなければならない。この補充を、仕事として行うかどうかには議論の余地がある。会社を例にすると、性欲があるからといって仕事で満たすかというと、ノーだろう。食欲については、昼休憩の形で時間だけが確保されている。ではコミュニケーションは？　性欲のように会社側では放棄するか。食欲のように時間だけ設けて各自に任せるか。それとも積極的に介入するなり、何ならある程度強要してしまうか――やりようは色々ある。このあたりは後で議論する。

情報のやりとりとして各パターンを見る

冒頭でも述べたとおり、仕事とは情報のやりとりである。情報を上手くやりとりできれば上手くいく。しかし、コミュニケーションなどという「欲求を満たすための」「負担の高い」行為が未だに使われている。もっと上手く情報をやりとりするには、この左下のやり方から脱さねばならない。右下への移行が現実的だ。一方で、人間としてコミュニケーションは必要であるため補充もしたい。そういうわけで、マトリックスの計 4 パターンは結局すべて使うことになる。

4 パターンのそれぞれについて、どんなあり方や手段があるかを見ていきたい。

1 同期的なコミュニケーション

	情報が残らない	情報が残る
リアルタイムである（同期的）	1 同期的なコミュニケーション	~~2 同期的な情報共有~~
リアルタイムでない（非同期的）	~~3 非同期的なコミュニケーション~~	~~4 非同期的な情報共有~~

原始的にはこれ一択である。すでに述べたとおり、私たちの欲求を充足する手段としては優れているが、それだけだ。一応「瞬発的に非言語情報を伝える手段としては最優秀」と褒めることもできるが、有用なシチュエーションは限られている。たとえば娯楽と本番くらいだろう。顧客に向けての提供であれば致し方ないが、困ったことに、内部でチームで仕事をする際も採用されがちである。

最も典型的なのが出社と会議だろう。たとえるなら、どちらも練習試合である。試合時間も長

い。その分、だらだらできる余地は大きい（小さいかほぼ無いこともある）が、それでも貴重な時間の大半が費やされてしまう。1 日 8 時間とすると、定時で通勤時間がなかったとしても 1 日の 1/3。残業か、通勤があれば、実質 1/2 を占めることも珍しくない。この野蛮な搾取を正当化するために、時給の概念がよく使われる。比例した見返りを出してるからそれでいいでしょというわけだ。

時間は貴重だし、人間は怠惰だ。ゆえにこそ、私たちは「何かをするための時間」を設けて、そこに参加して、その間だけその何かをするという**時間枠の発想**を採用してきた。カジュアルには予定と呼ばれる。管理ツールとしてはカレンダーが知られている（ちなみに秘書は自律的なカレンダーと言える）。素人からプロまで、多くの人が当たり前に使うものだ。怠惰な人間が貴重な時間を上手く使うための、現実的な方法がこれくらいしかないのである。

この時間枠の発想は、格差と搾取を生む。強い者が立てた枠に参加できた者が強くなるため、いかにして枠を立てるか、また参加するかのゲームとなる。すでに述べたとおり、人間の性能にも限界があるため、枠と出入りの数は限られる。選ばなくてはならない。選ぶとは捨てることだ。毎回常に選ぶのは大変だから、身内をさっさと選んだ上で身内以外は基本的に捨てるとの戦略を取ったり、選び方を定めた上でシステマチックに運用したりするが、選ぶ側の論理で制御できることに変わりはなく、選ぶ側次第と言える。つまり構造的に政治になり、強者と弱者が生まれ、強者の枠を享受できる者達が勝ち続ける形で、あるいは多くの枠に入れない者達が負け続ける形で格差になる。さらに強者は「享受したければこの枠に入って来い、嫌なら出ていけ」と負荷の高い時間枠を強要できる。選択肢の無い弱者は耐えるしかない。あるいは、逃げたり壊れたりしても、（すでに多数の弱者と少数の強者との構図が出来上がっているのであれば）替えはいくらでもいて補充されるから心配ない。むしろ、このようなムーブをどれだけ回せるかがそのまま成果に繋がる。構造的に搾取のインセンティブが高い。

時間枠の発想にとらわれている限りは、この構造上の限界を越えることはない。現代でも同じだ。せいぜい**枠への参加コストが減る程度である**。現状のリモートワークを思い浮かべてほしい。出社や移動のコストが減っているだけで、会議自体はオンライン会議の形で引き続き行っているはずだ。残念なことに、ただのトレードオフでしかない。参加コストを減らせる代わりに、会議中に得られる情報量や瞬発的な立ち回りの融通が落ちてしまう。そしてご承知のとおり、このリモートの塩梅は受け入れられなかった。パンデミックが落ち着いて「やむを得ない事情」がなくなった途端、出社回帰に倒され始めている。経営者も強者として運用しやすいため、これ幸いと適当な理屈をつけて出社を導入する。

理屈に騙されてはいけない。時間枠の発想、その根っこはコミュニケーションという欲求充足行為だ。当然ながら、より満たせる相手を選ぼうとする。あけすけに言えば好き嫌いで選ぶ。**本質的には好みの異性を漁るのと大差ない**。それを現代風に横文字で正当化すると、たとえばマッチングとなる。カルチャーなる概念を持ち出してカルチャーマッチと言ったりする。カルチャーマッチ自体は有用な概念だが、原始的な好き嫌いの世界への安住を正当化する隠れ蓑として使われてしまっている。

トレードオフの理そのものから脱却せねばならない。ビジネスチャット、Web 会議、バーチャルオフィスといった手段とその工夫は、トレードオフのスライダーを動かす道具でしかない。それではダ

メなのだ。ゆえに、これ以上、スライダーを動かす道具について議論するつもりはないし、する必要もない。

2 同期的な情報共有

	情報が残らない	情報が残る
リアルタイムである（同期的）	~~1 同期的なコミュニケーション~~	2 同期的な情報共有
リアルタイムでない（非同期的）	~~3 非同期的なコミュニケーション~~	~~4 非同期的な情報共有~~

時間軸の発想を打ち破るためには、情報が残ることと非同期性（リアルタイムでないこと）を導入すれば良い。といっても、いきなり導入できるものではないし、常に両方とも発揮するだけで済むものでもない。私たちは人間であり、人間ゆえに欲求の充足自体は必要だ。すでに述べたとおり、4 つのパターンをバランスよく使って、でも左上ではなく右下をメインに据えたい。

というわけで、次は情報が残るようにしたパターン 2 を見ていこう。同期的なコミュニケーションにおいて、情報が残るようにするとどうなるか。同期的な情報共有とは何なのか。

最も身近な例はワークだろう。ワークショップや演習やレビューなどである。学生や研究者だと実験も加わるし、本番の概念がある仕事の人は練習やトレーニングも行うはずだ。プログラミングではペアプログラミング、モブプログラミングといった協働的な作業手法も存在する。いずれにせよ、**何らかの中間成果物を残しながら** やりとりをする営みと言える。中間的な成果物として外に出し、それを仕上げようとする。時間内に最終的な、あるいは暫定的な成果物にできればいいが、できなかった場合は持ち越しだ。できるだけ粘ることもあるし、状況次第ではできるまで帰れないこともある――とここまで書いて申し訳ないが、これらは同期的な情報共有とは言えない（部分的に行えていることはある）。情報共有ではなく作業そのものである。意味を理解してもらうための準備として、取り上げただけだ。

同期的な情報共有とは、情報を残しながらリアルタイムにやりとりすることだ。ワークでは（作業に伴って出来上がる）中間成果物を残していたが、そうではなく、やりとりした情報そのものを残す。議事録をその場その場で動的に残すと言ってもよく、IT エンジニアを中心に **議事メモ** と呼ぶことがある。平易な用語だが、この概念を理解するのはかなり難しい。

ここまで聞くと、速記や録音・録画など丸ごと残す手法を思い浮かべる人が多いと思う。これらは同期的な情報共有 **ではない**。**単に同期的なコミュニケーションを記録しただけでは、同期的な情報共有とは言えない**。同期的なコミュニケーションで済むのなら、それで良い。ただ、何もしないと記録が残らず、参加者以外に情報が行き渡らない。これを記録して、かつ公開して見れるようにすれば非参加者にも見えて便利だし、見えていることが抑止力となって不正や政治を減らせる。この性質を透明性と呼び、重要な議論でもあるが、後で述べる。話を戻して、ここで言いたいのは「コミュニケーションを残せ」ではなく **「残せる手段を使ってコミュニケーションをせよ」** である。

もっと言えば、聞く・話すでコミュニケーションするのではなく読む・書くでやる。もちろん同期的であるから、リアルタイムでなくてはならない。100% 読み書きだけに頼って一切口と耳を使いません、も可能であるが、少なくとも読み書きが主でなくてはならない。いわゆる議事録は聞く・話すが主で、記録は従だった。議事メモはそうではなく、読む・書くが主で、聞く・話すが従となる。主従が入れ替わるレベルのパラダイムシフトとなる。

当然ながらやりとりの仕方も違ってくる。いくつか Before/After で挙げてみよう。Before がパターン 1 の同期的なコミュニケーション、After が今扱っているパターン 2 の同期的な情報共有である。

- 発言
  - Before: 喋る
  - After: 書く or 書いておく
- 会議のセッティング
  - Before: 場所を押さえる
  - After: 書ける場所（たとえばページ）を押さえる
- 発言権
  - Before: マイクでたとえる。同時に一人しか持てない → 喋れるのは一人だけ
  - After: フラッグ（旗）でたとえる。今現在の書き込み先を提示する → 複数人が同時に書き込める

具体的なケースでも見ていこう。ある 6 人チームがいて、うち□さんが遅刻の常習犯であり、何とかしたいとしよう。チーム全員で会議することになった。□さんはナルコレプシーであり、夜型の生活リズム（たとえば 13:00 〜 21:00）であれば問題なく勤務できるが、□さん含めてメンバー全員がまだ知らない。常に眠そうにしている様は全員知っていて、最初はキャラクターの範疇であったが許容範囲を超えておりこうなっている。

Before の場合、おそらくマネージャーが場を仕切って、□さんにヒアリングしたり、他のチームメンバーからも意見を聞いたりするだろう。会議時間は有限であり、マネージャーはおそらく忙しくて、この場で結論を出したがる。具体的な対応策を何かしら一つはひねりだそうとするだろう。そのためにマイクのコントロールもする。極端な話、□さんにマイクを渡す時間と回数を減らせば、マネージャーや他のメンバーの発言を有利にできる。逆に、□さんに渡す分が多すぎて□さん劇場になってしまうかもしれない。もちろんコントロールが上手くいかず、メンバーが感情的になって言い争いになったり、□さんが萎縮してフリーズしてしまってそれを皆でねちねち責める構図になるかもしれない。

After の場合はどうなるか。まずは書き込める場所、たとえばノートアプリやドキュメントツールのページを設けて、議題を書き込まなければならない。怠惰でなければ事前に書き込んでおくはずだし、複数の議論を立てることもありえる。会議中は、やはりマネージャーが仕切ることになるだろう。マイクは無い。代わりにフラッグがあって、まずは一つ目の議題に立てる。メンバー全員は、この議題について意見を書き込んでいく。それらを見たり、必要ならやりとりもして議論を深めていく。とはいえ、書いただけで終わりだと何も起きないので、結論を出さねば（書き込まね

ば）ならない。あるいは結論にしたい部分を選んで「これにします」と書いてもいい。いずれにせよ、意思決定者たるマネージャーが行うはずだ。**意思決定とは、意思決定した情報を書き込むことである**。これで一つ目の議題が終了する――と、このようなイメージになる。議題を一つずつ皆で見ていくわけだ。他にも、あえてフラッグを設定せずに「会議中は誰でもいつでも好きな議題に書き込んでいいよ」「何なら議題を足してもいいよ」とすることもできる。このあたりの方法論は後で扱うことにして、ここでは Before とは異なった、読み書きベースのやり方にシフトしている様を感じ取ってほしい。

Before にせよ、After にせよ、意思決定者とファシリテーター（この場合はおそらくマネージャー）の存在は欠かせない。ただし手段がだいぶ違う。Before では、喋る人にマイクを渡して、マイクを手にした人だけが喋る世界観だ。一方、After では、皆が書き込めるエリアがまずあって、そこに議題と議論を書き込む。比較してみよう。

- 意思決定
  - Before: 意思決定者が「こうします」と言う
  - After: 意思決定者が「こうします」、あるいは「これにします」を書く
- ファシリテーション
  - Before: ファシリテーターが発言してほしい人にマイクを渡す
  - After: ファシリテーターが議論したい議題にフラッグを立てる

また、従の手段をどう使うかも比較しておく。

- 主従
  - Before: 聞く・話すが主、読む・書くが従
  - After: 読む・書くが主、聞く・話すが従
- 従の使い方の例
  - Before: ホワイトボードに描く、録音や録画、議事録係を据えて議事録を取る etc
  - After: フラッグの配置を知らせる、意思決定に足る情報が出ていないので書き込みを催促する etc

いかがだろうか。Before（同期的なコミュニケーション）が唯一の方法ではない、とおわかりいただけるかと思う。仕事とは情報のやりとりだが、何も聞く・話すだけではない。読む・書くも使えるし、何なら主従を逆転させて読み書きをメインにすることだってできる。もちろん、第一部で述べたとおり、読み書きするための力は必要だ。これを文字通りの IT リテラシーと呼んだ。

最後に、ツールについても言及しておこう。読み書きというとチャットを浮かべる人が多いかもしれないが、力不足である。**チャットで非同期的な情報共有を行うのは難しい**。この点も後で詳しく述べるが、軽く扱うと、チャットはメッセージを時系列に表示するだけの単純な世界観だ。タイムラインともいう。このタイムラインには「今現在投稿されている話題」という**空気**が存在する。空気を破るのは難しいし、破ったとしても、こんな一次元的な世界では複数の議論なんてとても扱えない。直感的に理解したければ、X(Twitter) で日々交わされるレスバを浮かべるといい。双方がよほど賢くなければ、まともな議論なんてできやしない。できたとしても、あとで追いかけるのも難しい。チャットも時間枠の発想でしかない。対面で集まって喋るコストを減

らすかわりに、伝えられる情報量を（非言語+言語から言語のみへと）減らしただけだ。よって、**非同期的な情報共有がしたいなら、チャット以外のツールを使わねばならない。**パンデミックにより、Teams や Slack などのビジネスチャットを覚えた人が多いだろうが、その程度では正直言って話にならないのである。ノートやドキュメントといった、より読み書きのしやすいツール（とこれを使うITリテラシー）が求められる。

ここで、After のやり方が常に優れていると言っているわけではないことに注意したい。本書のスタンスはデフォルト・リモートであり、Before（同期的なコミュニケーション）よりは After（同期的な情報共有）であるべきと主張したいが、Before も After もやり方の違いでしかない。After を使えば、この 6 チームが□さんのナルコレプシーを疑って、診断してみてはどうかとの結論に至れるかというと、わからない。文字通りの ITリテラシーがないか、あったとしても読み書きが苦手であれば、まともな議論にまで届かない可能性が高い（特にマネージャーと□さん）。ただし、これは Before も同じことだし、そもそも構造上の限界があることはすでに述べたとおりだ。

3 非同期的なコミュニケーション

	情報が残らない	情報が残る
リアルタイムである（同期的）	~~1 同期的なコミュニケーション~~	~~2 同期的な情報共有~~
リアルタイムでない（非同期的）	3 非同期的なコミュニケーション	~~4 非同期的な情報共有~~

パターン 2 は左上に「情報が残る」を追加したものであったが、別の進行方向を考える。非同期性を導入した場合、パターン 3 の非同期的なコミュニケーションとなる。

この言葉は、IT エンジニアや一部のビジネスパーソンであれば聞いたことがあると思う。本章で述べている情報共有も含んでいたりするが、いったん脇に置いてもらいたい。直感に反するかもしれないが、本章では別の定義を使いたい。

非同期的なコミュニケーションとは、本人以外の誰かから**間接的に、同期的なコミュニケーションによって** やりとりすることを指す。たとえば□さんが□さんとやりとりしたいとしよう。普通に□さんと会話したのなら、パターン 1 の同期的なコミュニケーションと言える。では、□さんとは別の□さんから、□さんがこのように言っていたと告げられるのはどうか。□さんとは同期的だが、□さんとは同期的ではない。しかし、□さんの情報は伝わっている。伝え方は（□さんとの）同期的であるため、情報としては残らない。

sync_and_async

説明にすると小難しいが、要は□さんとの同期的なやりとりによって、□さんと非同期的にやりとりできている。これが非同期的なコミュニケーションである。本人以外と同期的にコミュニケーションすることで、本人の情報を非同期的に手に入れるわけだ。もちろん、□さんが言いたいことを、□さん経由で□に伝えることもできるだろう。

このパターンは、言うまでもなく身近に存在する。又聞きや仲介の概念はおおよそ誰もが知っているだろうし、現代でも多くの組織では階層が敷かれていて伝言ゲーム的に情報が行き来する。そうでなくとも、SNS や、人間関係そのものがネットワーク的であり、ノード（人）からノードへと情報が複雑に行き来していく。ご承知のとおり、元の情報が正確に伝わる保証はないし、むしろ歪んでいく。一次情報を確認しろ、はビジネスでも自己啓発でも鉄板の格言だ。

ここで名前を変えたい。この第三のパターンを、ここでは非同期的なコミュニケーションと呼んでいるが、既存のニュアンスとは異なるため紛らわしい。そこで **間接的で同期的なコミュニケーション** と呼ぶことにしたい。少し長ったらしいが、ここで用いている意味として捉えやすいはずだ。マトリックスは次のようになる。

	情報が残らない	情報が残る
リアルタイムである（同期的）	~~1 同期的なコミュニケーション~~	~~2 同期的な情報共有~~
リアルタイムでない（非同期的）	3 間接的で同期的なコミュニケーション	~~4 非同期的な情報共有~~

この間接的で同期的なコミュニケーションは、手段としてはパターン 1 の同期的なコミュニケーションと同等であるため割愛する。もちろん性質は変わってくる。基本的に本人と同期的なコミュニケーションを取れるのが理想であり、これができないから間接的にするわけだが、その分、やりとりする情報の精度は落ちていく。

戦略としては、本人（一次情報）との同期的なコミュニケーションを諦めずにちゃんと取りに行ったり、それができなければ複数人とやりとりすることで間接情報（二次以上の情報）を拡充させて確度を高めることになる。ビジネスのテクニックに落とし込むと、ステークホルダーの全体像を洗い出して、本人と誰が繋がっているかを可視化し、個別にどのようにコミュニケーションしにいくかを計画し、その進捗を管理する、などとなるだろう。自己啓発的に言えば、日頃から交友関係をつくっていけ、そのための時間とつくって投じていけとなる。「根回し」の重要性もよく説かれる。文化的に言えば、日本では関係ベース――一緒に過ごすことで信頼関係をつくっていくことが大前提なので、ある程度の時間は差し出さねばならない。普段の仕事では忙しいか、役割の仮面があって機会がないから、通常は業務外のイベントや懇親会を使う。近年では 1on1 を始めとするエンゲージメント活動で、業務時間の一環で行えることもある。

いずれにせよ、単に機会をつくれればいいというものではない。間接的に情報を出してくれる程度の関係性がないと、そもそも始まらない。加えて、本人以外の人物がその人について喋るというのは、一般的に好ましくはない。少なくとも表立っては行いづらいし、実際、会社のエンゲージメント活動程度で引き出せることはほぼない。非公式の無礼講という建前があってようやく引き出せるものだ。いわゆる飲みニケーションは、唯一日本にインストールされている無礼講の場である。ちなみに「会食」も似たようなものであり、日本に限らずどの文化でも見られると思う（飲みニケーションは言わばカジュアルで人数多めな会食）。この会食志向に加えて、現代は SNS の影響もあって燃えやすいため、リスク回避志向も強化されている。クローズドな Discord コミュニティの台頭は象徴と言えるだろうし、X(Twitter) もインフラと評されるほど人が

多く、オープンで誰もが群がれる場所でもあったが、あえて距離を置く人も少なくない。読者の身近にもいるのではなかろうか。そういうわけで、クローズドな無礼講を望む流れは今後も続くし、強化すらされると思われる。本人と同期的コミュニケーションができないから間接的にしたいのに、その間接的な部分でも、ゴッリゴリに同期的コミュニケーションが要求される構図となっている。私たちはそのために社交への参加に講じなければならなくなるが、コミュニケーションは欲求充足行為であり、気持ちがいいため満足してしまう。このあり方は続く。

結局のところ、**間接的にしただけでは、同期的コミュニケーションからは逃れられない。**

4 非同期的な情報共有

	情報が残らない	情報が残る
リアルタイムである（同期的）	~~1 同期的なコミュニケーション~~	~~2 同期的な情報共有~~
リアルタイムでない（非同期的）	~~3 間接的で同期的なコミュニケーション~~	4 非同期的な情報共有

左上から「情報が残る」を足して右に行くと、読み書きを主とする同期的な情報共有になる。非同期性を足して下に行くと、間接的で同期的なコミュニケーションになる。では両方足して、右下に行くとどうなるか。

経路としては、左上のパターン２から非同期性を足す時計回りがわかりやすい。非同期的な情報共有とは、同期的な情報共有を非同期的に行うものであり、言い換えると情報の養育を行う。非同期的なので時間は拘束しない。つまり集まることさえしないので、**会議という概念がない。**

一体そのようなあり方をどうやって実現するのか。追って見ていこう。

一番浮かべやすいのはチャットだろう。メールや手紙でもいい。情報を相手に渡して、いったんおしまいとする。そのうち相手が見て、アクションを起こしてくれるだろう。必要なら自分に返事を出してくるはずだが、これは自分の受信箱をチェックしておけばいい。あるいはメールならポップアップを出してくれたり、チャットでも通知を出してくれて気づくこともできる。手紙の場合は、ポストをチェックした同居人が「届いてるよ」と教えてくれたりもする。もちろん意識高く、習慣や日課のように定期的にチェックしても良い。なければ何もしないし、あれば対処を考える。ツールが変わろうと、リアルタイムでないやり取りはこの程度だ。送っておしまいにすることと、受信箱をチェックすることの二点、これだけである。

相手が一人だけなら単純だが、現実はそうでもない。相手は一人ではなく複数人いるし、グループやチームといった複数人を束ねた単位がつくられていてその単位ベースでやりとりすることもある。メーリングリスト、チャンネルやチャネル、グループと呼び名は様々だが、複数人を示す単位は今や当たり前だろう。一方で、そのような場でやりとりしづらいものは、個人同士でやりとりしたりもする。ダイレクトメッセージや個別チャットなどと呼ばれる。また、場であっても、広く見え

ているものもあれば、参加者にしか見えないプライベートなものもあり、後者が好まれることも多い。これだけでもかなり複雑なのに、状況が忙しかったり、相手がせっかちだったりすると返事を急かされることもありえる。というわけで、以下の複雑性が存在する。

- 物量。相手の数と、やりとりする情報の量
- 空気。「場」を相手にする必要性と、どの場に出すか・出さないかといった機微
- 優先度。重要性や緊急性による返信要否や頻度の変化

果たして、これらをコントロールしながら仕事していけるだろうか。ノーである。

まず個人レベルでは、いつ何をどれだけやるかを自身でコントロールできねばならない。これを（個人レベルの）タスク管理と呼ぶが、タスク管理を身につけている人は非常に少ない。おそらく読者の 99% は身につけていない。一見すると多忙な人でも、その実、会議や会食の予定を入れてばかりの「予定の奴隷」だったりする。カレンダーに沿って、ただただ予定に参加するのみ。大変かもしれないが、簡単である。小学生でもできる。と、別に責めているのではなく、タスク管理がそれだけ難しいことを意味する。そもそも体系化すらされていないので学ぶ機会もない。できなくて当然だし、カレンダーで上手くいくから学ぶインセンティブもない。しかし、非同期的な情報共有にあたっては **必須である**。この複雑性を自ら扱えねばならない。無論、個人の努力に帰結させるのではなく、やり方と考え方と技術で支援するべきであるが、**非同期的な情報共有では個人のタスク管理が必要、とのテネットは揺らがない**。ここから逃げてはならない。なおタスク管理の話は前作でたっぷりと扱ったため、本書では割愛する。

容易に想像できるように、個人レベルでコントロールできただけでは完結しない。この複雑性を全員が現実的に扱えるように、そもそも組織が立ち回る必要がある。テネット、やり方と考え方、技術がすべて絡む。現行のものでは到底足りないし、とても一言で表現できるものでもないため、本書全体で示していく、とさせていただきたい。

最重要なのがやはりテネットで、**非同期的な情報共有でも仕事が成立するような前提にシフトしなければならない**。たとえば非同期的な情報共有では、各自が各自のペースで読み書きを行うことになるため、それ相応の余裕が要る。業務時間中に 1 日 1 時間、2 時間、できれば 3 時間の余裕がほしい。Google で有名な 20% ルールという取り組みがあるが、たとえるならこの 20% を恒常的に確保するようなものである。しかも「余裕」であるため、合間に家事をしたり、のんきに散歩したりしても良い。同僚と雑談したり、ゲームをして過ごすのもアリだし、誰がどう過ごしているかは関知しない。する必要がない。管理などもってのほかだ。余裕（余白と呼ぶこともある）の重要性は聞き飽きているとは思うが、まさにこれを **前提として** 確保するのである。この余裕の概念を **スラック（Slack）** と呼び、後で詳しく議論する。

やり方と考え方については、一言で言えばドキュメンテーションである。日本語では「文書」であるが、従来のニュアンスでは語弊を招く。コミュニケーションが情報を残さないやりとりなのに対し、**ドキュメンテーションとは情報を構造的に残すやりとり** と言える。単に残すのではなく、構造的に、がポイントだ。構造というと、文書の目次や見出しはわかりやすい。多くのビジネスパーソンはパワーポイントなど「スライド」にも馴染みがあるだろう。ブレストをしたことがある人なら、付箋で散らかしたものをグルーピングしてまとめていくといった構造のつくりかたも知っているだろう

し、個別の詳細文書やページのリストアップしたりカード状に並べたりと見せ方も馴染み深い。構造は、内部と外部にある。本という情報でたとえるなら、本の内部で目次で示される構造になっているし、複数の本をいかに探すか・位置づけるかという観点でも、書店や図書館などは各々整理している。日本十進分類法のような共通の分類体系もある。

こう聞くと、構造なるものは、私たちには縁がなさそうだが、そうではない。**万人が構造を扱えるようにならねばならない**。非同期的な情報共有は構造無しには成立しない。特に、日本十進分類法のような権威的な構造だけで済むものではなく、**自分たちで必要に応じて構造をつくったり変えたりといった「構造自体のやりとり」**が必要になる。たとえば、議論したいことが 6 個ある場合、1-議題 1-ページ（あるいはエリアでもいい）をつくって、言いたいことは各議論のページに書いてね、とする。こうすれば各議論は各ページ内で完結するからスッキリするし、共有や言及もしやすい。たとえ議題が 200 個あろうともコントロール可能になる。200 個すべてを必ず短納期で扱えるわけではないし、そんなことは不可能だが、捉えることはできる。これはドキュメンテーションという、構造的に残す営みを採用してはじめて行えることだ。もちろん全員がこの営みに適応できねば成立しない。コミュニケーションの営みだけでは不可能である。そのために、色々なやり方と考え方も使うし、本書を通じて示していきたいが、一言で言うと「構造を自ら扱うようにもします」なのである。

とはいえ、ドキュメンテーションのために構造の視座を持って行動してください、と各自に委ねるのも酷だ。教育や啓蒙面でカバーすることもできるし、するべきだが、ここでは割愛する。技術でカバーするべきであろう。パターン 2 でも少し述べたが、現代的なノートアプリや共同ドキュメントツールを使えば、比較的やりやすくなる。認知負荷のかからない構造をつくったり、意識せずに従えるように見せたりするのはもちろん、可視化やリマインドにより（いつどこをどれだけ読み書きするかの）タスク管理も支援できる。現状、キラーアプリはまだないので、新しくつくるか、既存ツールを上手く連携してやらねばならない。生成 AI に絡めて未来を想像するなら、各個人に精度の高い秘書 AI がついて、AI に従うだけで済むかもしれない。何なら読み書きも AI が行ってくれて、私たちは指示と確認が主な仕事になるかもしれない。誰もが AI 部下を抱えて万人マネージャーの世の中になることすらありえる、と妄想はここまでにしておくが、なるべく任せられた方が良さそうなのは言うまでもないはずだ。この、いわば **認知負荷の委譲** を行うためには道具が必要で、道具を扱うために技術が必要という点は、前の章でも述べたとおり。

また、別の観点として、ドキュメンテーションは読み書きをするため **なるべく素早く表示できねばならない**。ページの表示に 10 秒かかりますでは話にならない。10秒、5秒、3秒、1秒、0.5秒、0.2秒、0.1 秒は全部体験が異なる。パターン 1 の同期的なコミュニケーションが好まれるのは、単純にフィードバックが高速だからでもある。ここパターン 4、非同期な情報共有の場合、その指標にあたるものは表示速度だ。アプリやゲームやウェブサイトもサクサク動けないとイライラするだろう。このイライラは天敵であり、0.1 秒でも減らすべきである。青信号なのに前の車が発進しないとイライラするが、**私たちは意図どおりに進まないとイライラする。情報のやりとりも同じだ**。0.1 秒でも、と書いたのは大げさではない。当然ながら、少しでも早く表示するには多くの技術を要するし、容易な作業でもない。お金もかかるわけだが、投資しなければならない。組織の設計も多分に関わってくる部分であり、後の章で改めて扱いたい。

コミュニケーションと情報共有を分離する

「情報の養育」と「養育者のメンテナンス」を分ける

ここまでの議論を整理すると、デフォルト・リモートではパターン４の非同期的な情報共有を重視する。情報を養育するのである。しかし、私たちは人間でもあり、従来の同期的なコミュニケーションも引き続き必要である。この考え方を上手く表現するために、養育のたとえを拡張した上で、以下三点にまとめる。

- やりとりの理。仕事とは情報のやりとりである
- 情報の養育。情報のやりとりを上手くやるためには、情報を養育すればよい
- 養育者のメンテナンス。養育するのは私たち養育者であるが、養育者も人間であり「欲求の充足」が必要である

養育というと、養育者のメンテナンスは後回しにされがちである。育児では子どもが第一、教職では生徒が第一であって、養育者たる親や教師は二の次。育児の負担が甚大なのは当然だし、教職員がやりがい搾取的なのもむしろ本望だとの暗黙もあるようだ。昨今ではようやく見直されてきていて、働き方改革や夫婦間の家事分業は当たり前となりつつあるが、それでもなお利他的な――より率直に言えば自己犠牲の本質は変わらない。これを利他の原則と呼ぶことにする。利己的な者が養育に携わることはそもそも許されない。よほど有能であれば上手くやれるだろうが、そうでなければ人でなしの烙印さえ押されるだろう。

情報の養育においては、利他の原則は通用しない。私たちが利他的になれるのは、相手が生物だからにすぎない（よって人間に限らずペット相手でも可能だ）。情報は違う。比喩的に情報も生き物だと表現したが、実際に私達が利他的になれる、文字通りの生き物かというと、もちろん違う。というわけで、利他的になれないのに養育をしなければならない苦悩を背負うことになる。情報自体が、あるいは養育が面白いという興味関心も強いエネルギーではあるが、利他の原則に比べると心もとない。**苦悩は必然だと考えて差し支えない**。

この苦悩に耐えるために、私たち養育者自身のメンテナンスをしっかりと行う。コミュニケーションという欲求充足行為をそれなりに行わねばならない。もちろん、デフォルト・リモートでは非同期的な情報共有がメインであり、コミュニケーションはなるべく行わない方がいいので、従来どおり無闇に集まって喋るわけにもいかない。まず非同期的な情報共有をしっかりと確保した上で、それからコミュニケーションを別途確保することになる。別の言い方をすると、従来は「同期的なコミュニケーション」の中で議論（情報のやりとり）も行っていた。充足も、議論も、どちらもいっしょくたにやっていたのである。これを分けねばならない。充足するための時間と、議論するための時間を分けて、前者では充足に専念し、後者では議論に集中する。シングルタスク化と言ってもいい。**コミュニケーションしながら議論するという「ながら」をやめて、コミュニケーションと議論を別々に行うのだ。**

コミュニケーションの注入

非同期的な情報共有がメインのあり方において、必要に応じてコミュニケーションを確保することを**コミュニケーションの注入（Communication Injection, CI）**と呼ぶ。

食事は一日三回くらい摂るだろう。間食する人もいるかもしれないし、いいかげんで頑丈な人や忙しい人はもっと乱れるだろう。それでも一定か、支障がない程度に不定期的に摂りはする。コミュニケーションも同じだと考える。コミュニケーションをしていないと何かが欠乏して、おかしくなってしまうから、そうなる前に補充をする――そう考える。

たとえば毎週月、水、金の11:00 〜 12:00を、コミュニケーションの時間に充てる。この間は仕事さえもせず、メンバーとのコミュニケーションに専念する。といっても、集めるだけ集めてはいどうぞ、と言われても困るし、日本は会食文化でもある、そのままだとコミュニケーションとして盛り上がりづらい。会議においてファシリテーションが推奨されるように、このコミュニケーションの時間においても何らかの仕掛けや誘導が要る。複数人だと話しづらいから常にペアをつくるようにするとか、必ず何らかのテーマを持参してまずはそこから広げるようにするとか、そもそもコミュニケーションは会話とは限らないわけで、それこそ一緒にゲームをして過ごしても良い。スプラトゥーンやマリオカートでもいいし、ボードゲームやカードゲームでもいい、マイクラもアリだろう。FPS や格闘ゲームは学習量も多いし、競争心を煽りすぎるから良くないと思うが。

いずれにせよ、注入するための「コミュニケーションの時間」はそれなりにデザインせねばならないということだ。これを**コミュニケーションタイム**と呼ぶ。コミュニケーションの注入とは、コミュニケーションタイムを上手くつくった上で、必要に応じて差し込むことなのだ。目的をはっきりするために、より限定的な名前をつけても良い。いくつか例を挙げる。

- 雑談タイム
- ゲームタイム
- 体操タイム

コミュニケーションタイムのあり方は様々だ。少しわかりづらいものもいくつか挙げよう。

- 鑑賞タイム
- シング（歌）タイム、あるいはカラオケタイム
- 呆活タイム

鑑賞は映画を想定しているが、マンガタイムでも良い。オフィスにマンガ喫茶のようなスペースをつくって、コミュニケーションタイムとしてマンガタイムを設ける、といったことも至って真面目にアリである。シングタイムについては、社歌を想起してブラック企業を彷彿とさせるかもしれないが、コミュニケーションの一種として上手く使えば光るとは思う（筆者は嫌いなので遠慮したいが）。呆活については、能動的にぼーっとする活動を指す。起源はわからないが、筆者の造語ではなく実在する言葉だ。より賢くやりたいなら瞑想タイムの方がいいかもしれないし、こちらはビジネスでの事例も海外では珍しくない。

注意点として、作業や重荷になってしまわないようにしたい。それでは意味がない。仕事は非同期的な情報共有でやればいい。そうではなく、養育者のメンテナンスとしてコミュニケーションにフルコミット、フルダイブするための時間をつくろう、と言っているのだ。ゲームはおそらく相当好みが分かれるだろう。ボードゲームやカードゲームのような頭脳ゲームで充足できる人もいれば、そんなに頭なんて使いたくない冗談じゃないという人もいるはずだ。一番わかりやすいのはシングタイムで、歌はおそらくやりたくない人の方が多いと思う。呆活についても同様で、じっとすることが特性上苦手な人もいる。そのような人にとっては苦痛でしかない。**コミュニケーションタイムは多様でなくてはならない** のだ。

主な課題を二つほど挙げよう。一つは日本の「個人主義的ではあるが集団には迎合する」価値観である。たとえば「うちではコミュニケーションタイムは単一のあり方で済んでます」「全員これでいいと納得いただいてます」とマネージャーが主張するチーム、あるいは部門長が言う部門があるとしよう。鵜呑みにしてはいけない。無難に過ごすために、本当は嫌だけど隠しているケースが往々してあるからだ。これでは充足にならないか、あるいは足らない。情報の養育に耐えきれずに、どこかで潰れてしまう。この建前というハンデを乗り換えるには、どうしたらいいか。ケースバイケースだ。極端な話、慢性的に孤独で自傷的なメンバーであれば多少強引に巻き込んでも充足させた方が健全になる事が多いし、プライベートの交友関係が充実している人にはコミュニケーションタイム自体要らない。多様性の議論で公平性（Equity）があるが、まさに公平性が求められる――必要な人に、必要なだけの注入をするべきだし、足りている人に無理に注入する必要もない。**公平な注入が望ましい**。

もう一つの課題は、コミュニケーションの欠乏を測定する手段が無いことである。腹が減った、のどが渇いた、眠たい、漏れそう、ムラムラする、キレそうあたりはわかりやすい。欠如を感じるか、感じない程度に補充すればいいし、欠如してて暴走しそうになっても私たちは人間であり理性がある、通常は耐えられる。しかし、コミュニケーションはそうではない。じわじわと蝕んでいくし、どれだけ蝕まれたらどんな悪影響が出るかもわからない。敏感な人は「寂しい」「誰かと喋りたい」と行動に起こせるが、誰もがそうとも限らない。食事や睡眠といった原始的な欲求でも同様で、飲まず食わず寝ずに仕事をしすぎておかしくなる人も珍しくない。コミュニケーションの欠乏を知るための、キラーなやり方は現状無いと思われる。だからこそ、予防的にコミュニケーションタイムを定期的に設けて、そこから過不足に応じて調整していくしかない。

コミュニケーションの注入は、かんたんな営みではない。少なくとも人ごと、組織ごとにデザインしなければならないものだし、調整も要するものだ。今後デフォルト・リモートが進んでくると、ここの需要は確実に増えると考える。コミュニケーションタイム・デザイナーのような役割も生まれるだろう。

関係の構築は二の次

コミュニケーションの主目的として関係構築――一緒に働く人との信頼関係をつくることが挙げられる。先に結論を言うと、**コミュニケーションの注入を採用したあり方では、関係構築は捨てる**。あるいは少なくとも軽視する。これでも表現がキツイなら「重視はしない」でもいい。

コミュニケーションの注入が目指すのは、己の欲求充足までである。関係構築は想定していない。では代わりの手段はというと、無い。関係構築はしない。なぜなら、デフォルト・リモートでは、情報の養育を行うからである。情報の養育に関係性、よりダイレクトに言えば仲の良さや親しさは無用だ。適切な人が適切な情報を出すだけでいい。読み書きするだけでいい。それだけでしかない。むしろ、親しくないからといって出し惜しみするのは怠惰を通り越して罪ですらある。**情報の養育において、情報の出し惜しみは罪である。**

とはいえ、さすがに信頼の概念自体を捨てるわけにもいかない。信頼のつくりかたを変えるだけだ。具体的にはたった一つで、**信頼の構成要素として「一緒に過ごしてきた時間」をなくす**。代わりに「どれだけ情報を出してきたか」や「一緒に情報を養育してきたか」が重視される。文化的に受け入れづらいかもしれないが、そもそもデフォルト・リモート自体がパラダイムシフトのオンパレードであった。今さらである。遠慮せず主張させていただく。もう一度言う。信頼の基準を変えます。

幸いにも、これを支援する概念が一つある。**マスクド・アイデンティティ**と呼ぶ。アイデンティティとは、ここでは本名・容姿・性別を指す。従来はアイデンティティを開示するのが当たり前であった。ようやく最近になって、接客業スタッフのリスクを減らすために名札にニックネームを採用する事例が見られているし、インターネットなど一部の業界ではハンドルネームやアバターは主流だが、それでも素のアイデンティティは全体的には求められていた。マスクとあるとおり、これを隠すのがマスクド・アイデンティティだ。□さんがいたとしよう。

- □さんのアイデンティティ
  - 名前：佐藤剛（さとうたかし）
  - 性別：男性
  - 容姿：描写は割愛するが、履歴書やデジタルツールで画像やビデオを使うし、仕事中でも当然晒している

マスクド・アイデンティティの場合、□さんはこの素のアイデンティティを隠すことができる。何ならダミーのアイデンティティをつくることもできる。

- □さんがつくったダミーその1
  - 名前：ストロングシュガー
  - 性別：非公開
  - 容姿：下記のアバターを使っている、ビデオは無し、声はボイスチェンジャーを使用

    

    strong_suger

社内ではストロングシュガー氏が存在していて、日々情報をやりとりするし、仕事もしているのである――と、これは妄想に近い例であり、実運用上の課題も技術含めて色々あるが、ここまでにしておく。要は **信頼の基準を変えるには、アイデンティティを薄めればいい**。別の言い方をすると、情報のやりとりさえ上手くできるのなら、相手は何者でも構わない。「一緒に過ごしてきた時間」という因子に密接に絡みつくものの一つが「素のアイデンティティ」であるため、これ

をなくしてしまうわけだ。非現実的に聞こえるかもしれないが、たとえば社員として採用するレベルでは素のアイデンティティを使うが、普段の仕事ではマスクドで良い、との運用にすれば秩序は保てる。コミュニティ内で素のアイデンティティを使わなければならない、など固定観念でしかない。デフォルト・リモートの道のりは遠い。必要な概念は何だってつくるし、変更もしていく。ともあれ、このような工夫を駆使していくことで、マスクド・アイデンティティを実現していけるはずだ。

無論、目的は信頼の基準のシフトであって、マスクド・アイデンティティは一手段でしかない。他にも良い手段があれば、そちらを使ってもいい。

健全性は透明性をもってでしか示せない

見えていることの功罪

同期的なコミュニケーションはリアルタイムかつ情報も残らないため、閉鎖的であるとも言える。その場に参加した者にしか情報が行き渡らないし、誰を参加させるか・させないかでコントロールもできる。構造的に政治と格差に生み出してしまう。そこで、右下に寄せて非同期的な情報共有にすると、情報は残るため（制限を課さなければ）誰でも見れるし、リアルタイムですらないため参加のハードルも低く、閉鎖的にならずに済む。

このような閉鎖性の排除は、透明性を生み出す。透明性とはその名のとおり、透き通っていて中が見える様を指す。情報が残っていること、そしてそのアクセスが開かれていることの両方を求める。残っているだけでは意味がない。いつでも誰でも自由に見れなくてはならない。非同期的な情報共有だからといって、情報にアクセスできる者が限られている場合、それは透明性がないと同じだ。透き通った水は誰が見ても透明であるように、情報も、誰に対しても透明であるべきだ。

透明性がもたらすメリットは二つある。一つは、いつでも誰でもアクセスできるがゆえに化学反応が起きやすい、あるいは IT を活用してデータとして処理しやすいことであるが、本節では割愛する。最大のメリットは **抑止力になる** ことだ。透明であるとは、見えているということであり、いつどこで誰が見ているかわからないと言える。やましいことがしづらいのである。やましさの少なさを、起きた場合に速やかに対処できる度合いも含めて健全性と呼ぶことにすると、**健全性を高めて維持するためのベストプラクティスは、透明性の確保である**。通常は何らかの審査や評価を行う機関を設置するが、閉鎖的だと構造上、腐敗する。特定のクラスター（通常は権力者）を都合よく守る代わりに、それ以外を搾取する。この理不尽を隠すのにブラックボックスが役に立つ。これは組織力学、あるいは自然現象と呼べるほどの摂理だろう。ここに抗うには、見えるようにするしかないのだ。もちろん、透明になったからといって常に 100% 健全であれるはずもないが、それでも不透明ゆえの構造的な欠陥は潰せる。本書ではデフォルト・リモートという、出社に続く第三のパラダイムを是としているが、透明性も同様だ。健全性という観点で

見れば、透明性のある組織の方が水準が高い。透明な組織を目指すべきである。そして、これを可能とするのが、非同期的な情報共有である。

と、ここまで透明性を褒めちぎってきたが、良いことばかりではない。すぐに想像がつくように、透明であるということは、いつ誰に見られるかわからないことと同義である。精神的な負担が高い。透明性と精神的な負荷はトレードオフの関係にあり、これを **透明性のトレードオフ** という。透明性を高めることは、皆の負荷を上げることにも等しい。かといって、安易に不透明なやりとりを許容していては、すぐに形骸化してしまう。Slack や Teams といったビジネスチャットを導入して、せっかく広くコラボレーションができるというのに、プライベートなチャンネルや DM が乱立して閉鎖性が高まった、との例はあるあるだろう。これを防ぐためにプライベートチャンネルの作成を制限する取り組みさえある。

要するに、何もしなければ不透明まっしぐらとなる。なぜなら私たちは不透明を志向するからだ。人間であるゆえに怠け者で、いつ誰に見られているかわからない状況を想定し続けるのが純粋に辛い、また認知資源などリソースとしても限界があるという点が一つ。もう一つは、私たちには指向的なメンタルモデルを持っている――コミュニケーションにおいて、基本的に常に特定の誰かを想定して、その人のための情報を伝えようとする点が挙げられる。筆者はこれを **コミュニケーション1.0** と呼んでいる。この 1.0 のメンタルモデルは透明性と相性が悪い。1.0 は、想定した相手にのみ届くことを前提とするからだ。ゆえに、そもそも想定しないか、あるいは曖昧にするという指向性の緩和が必要であり、これを **コミュニケーション2.0** と呼んでいる。透明性を高めるためには、2.0 を身につけてもらわねばならない。経営者や組織長など従業員全体に伝える立場の人やインターネットで広く発信している人などは慣れているが、大多数はそうではない。では、どうやって身につけてもらうか、の解を筆者は持っていないが、透明な情報共有や不特定多数型のコミュニケーションの機会を増やして、やらせて、慣れさせていくしかあるまい。ソフトスキル、あるいは社会人の素養の一つとして鍛えるべきものだ。

透明の代行と、その先

とはいえ、2.0 を万人に要求するのは現実的ではない。多様性も無視している。不透明なあり方も許容できるべきだ。しかし、不透明な者が多数を占めると、組織としての健全性を保てない。この議論を進めるためには **透明の代行** に頼るといい。情報を全部見えるようにはしないが、必要な情報は適宜出すと考える。自律的に出せるのならそれでも良いが、タスク管理と同様、自律性を求めるのも酷である。情報を出す部分を技術あるいは役割によって代行させるのが現実的だ。

大きなスケールで見ると、経営者は会社の情報を外に出すための代行者と言える。メディアに出演して積極的に出す者もいる。技術で言えば、生成 AI を使うのも良いだろう。生成 AI は大量の情報から要約をつくるのが得意だ。外に出す情報をつくらせて、人間によるチェックも入れて承認したら公開する、とすれば共有もしやすくなる。このような発想を当たり前のように取り入れるのだ。たとえば 6 人からなる小さなチームがあったとして、マネージャーが透明の代行をする。他にできる人がいるならその人でもいいし、むしろその方がいい。前述、コミュニケーションの

注入の項で、コミュニケーションタイム・デザイナーが生まれるかも、との議論をしたが、同様に、透明の代行を設計・運用する役割も生まれると思う。

しかしながら、透明の代行は本質的な問題解決にはならない。代行と名付けたが、より正しく言えば説明責任の実施でしかない。透明とは、理想を言えば、ありのままを見えるようにすることである。ここまで議論した「透明の代行」は、情報を必要に応じて出すということでしかなく、説明責任を推進していると表現した方がしっくりくる――のだが、透明性とはその程度ではない。聞かれたら答える、必要になったから出すといったリアクティブ（反応的）なスタンスではなく、聞かれてもいないし必要かもわからないがとりあえず出しておくというプロアクティブ（予防的）なスタンスであるべきだ。**透明とはプロアクティブ（Transparency is proactive）なのである。**

したがって、**まずは透明の代行によって情報を出しつつも、ここであぐらをかかず、プロアクティブなスタンスを目指していきたい。**そのためには、上述したコミュニケーション2.0も含めて、テネットのレベルでのアップデートが必要だ。無論、新しいテネットを教えて知ってもらっただけでは行動にまで繋がらない。組織運用やソフトウェアといった道具を使わせて、アップデートしてもらうまで導きたい。

路線としては二つあるだろう。一つはティール組織（2）で、これは資本主義的・競争的な階層組織でもなく、ボランティアでよく見られるフラットな組織でもない、第五の組織パラダイムである。単体で業務が完結する小集団が最小単位であり、これ単体で採用――つまりは社員を増やせるほど自律的な権限を持っている。加えて、小集団と個人は自由に繋がって連携することが許されており、集団への参加離脱も、集団の作成と解体も動的に行われる。それでありながら、会社全体の方向性もコントロールしており、動的に更新される憲法とその運用によって、また憲法ベースのあり方を徹底的に教育することによって実現する。

細かい運用方法は複数存在するだろうが、ラルーが示す事例ではコミュニケーションを強調している。情報共有ではない。コミュニケーションだ。従来の組織パラダイムだと、制約とコストがキツすぎてまともにやりとりできないでいたが、ティール組織ほどの最適化されると可能である。必要なときに、必要な人（達）から、必要な情報を引き出すことができてしまう。筆者はこれを**生きた文脈（Live Context）**と呼んでいる。情報として残されてはいないが、情報を持つ人（達）からローコストで引き出せる状態なので、事実上、情報共有に等しい恩恵を受けることができる。プロアクティブではなく、リアクティブであるが、その敷居が徹底的に低いのである。たとえば、社員はいつでも誰にでも何でも聞くことができる。新人が社長に聞くこともできるし、社長が突然聞いてくることもある。顔も名前も知らない、同じ社員でしかない赤の他人が聞いてくることもあるし、自分から聞きに行くこともある。リアクティブゆえに透明では断じてないが、透明に近しいレベルでのやり取りが可能になっていると言えるだろう。この程度であれば、新しいテネットにアップデートせずとも、比較的馴染みやすいのではと思う。

もう一つが、透明性を促進するための新しいコミュニケーションツールをつくる路線だ。筆者は**キャスト**を提唱している。キャストとはチャットに代わる概念で、ブロードキャストのような広域な発信を思い浮かべてもらいたい。チャットでは特定の誰かを指定して、その人にだけ送信するが、**キャストでは常に社員全員に届く。**特定の誰かに届けることはできない。もちろん、自分

自身にも他の社員全員からのキャストメッセージがすべて届く。当然ながら全部読むことはできないため、読み方には工夫が必要で、以下に頼ることになる。

- 1: プロンプト・エンジニアリング
  - ほしい情報を取り出すために、生成AIに指示を出す
- 2: ブロードリスニング
  - 大量で多様な情報から、全体の傾向を把握する
    - たとえばクラスタリングする（データの集まりをグループ分けして視覚的に見せる）
  - 最近では東京都知事選で安野たかひろが使っている
- 3: プリミティブ・フィルタリング
  - キーワード検索、時期を指定した絞り込みなど、従来のフィルタリング

キャストを導入すると、「常に全員に届く」しか選択肢がないため、根本の発想が切り替わる。閉鎖性に行きたがる私たち人間を荒療治できるだろう。もちろん、ビジネスとして使うためには他にも色々な機能が必要だ。チャットでも、LINE だけではビジネスには耐えられないが、Slack や Teams ではワークスペース（チーム）やチャンネル（チャネル）やスレッドを導入することで実現した。ビジネスチャットと呼ばれる。同様に、ビジネスキャストと呼ぶべきツールをつくることができれば、透明性を促進できよう。生成 AI のおかげでプロンプト・エンジニアリングとブロードリスニングが可能になったからこそ実現しうる。

と、これは一例である。キャストでなくてもよいが、何らかの新しいコミュニケーションツールが必要だと思う。いや、もはやコミュニケーションではない。むしろ **コミュニケーションの域を超えた、新しいやりとり** を開発しなければプロアクティブには届きまい。本章冒頭にて、コミュニケーションは情報が残らないやりとりだと定義したが、加えて人間という対象を想定する営みでもある。ならばこれを裏切ればいい。直接的な相手が人間であるという単純な想定をしないやりとりを考えてみるのである。キャストもまさにそうして生まれたものだし、後の章では「AI」を想定するメンタルモデル（コミュニケーション3.0）も取り上げる。

透明性の先――到達性と偶発性

非同期的な情報共有により、情報の透明性が確保されやすくなり、組織の健全性にもつながる。透明性のトレードオフを飼いならし、できればキャストツールのような革新的な道具を新たに生み出すことで、透明性への耐性を底上げしていけるだろう。では、透明性を十分に維持できたら、それで良いのだろうか。

無論そんなことはない。仕事とは情報のやりとりであり、やりとりというからには、必要な人に必要なタイミングで届かなければ意味がない。これを到達性という。実用的には、透明性よりも到達性の方がはるかに重要であり、同期的なコミュニケーションにこだわる理由もまさに到達性のためである。だからといって、やっぱり同期的なコミュニケーションがいいですね、と逆行させるとデフォルト・リモートは遠のく。かといって、透明性だけが高い状態に甘んじているわけにもいかない。

到達性を高めるとは、本質的には一言でしかない。**モチベーションの問題** だ。モチベーションを高めることと、下げないことの両面をいかに多角的に取り組むかでしかない。

仮に□さんにＡという情報を届けたいとしよう。どんな手段が考えられるか。上司が肩を叩いたり電話を入れたりして「読んでおけ」あるいは「今すぐ読め、読むまで離さん」などと命令するのは乱暴だが単純なやり方だ。普段の会話で言及して意識させることもできるし、普段読み書きしている場所に書いておくことでも気づいてもらえる。チャットでメンションをつければ、すぐに目を通してはもらえる可能性は高いだろう。同僚からの又聞きや、何らかの資料でふと目にするといった偶発的な経路も考えられる（狙って届かせるのは難しいが）――と、方法は無数にあるが、大別すると能動か受動のどちらかである。□さんが自らＡを取りに行くか、あるいは外からＡを投げてもらうかだ。そして、どちらも本質的ではない。情報Ａが□さんのもとに届いたとして、結局□さんがＡとちゃんと向き合わなければ意味がない。**届いているが「スルーしてます」では意味がないし、表面的には「読んでます」と言ってても実際読んでいないのなら、やはり意味がない**。ちゃんと向き合うかどうかは、本人のモチベーション次第である。

モチベーションの因子は様々だ。信念や空気といった揺るぎないものもあれば、気分や不快感といったいいかげんなものもある。明確な理由が存在するとも限らず、なんとなくで自棄を起こすことだってあるし、逆に大したメリットがないのにアスリートのごとく面倒な日課に固執し続けることもある。わかりやすい例で言うと、仕事がどんなに忙しくても、給与明細は見るだろう。給与の改定に関する全社通知も見るはずだ。

私たち「モチベーションの奴隷」に情報を到達させる、比較的優れた方法が実はコミュニケーションだ。オフィスでの何気ない会話がアイデアを生む、とはよく言われることだが、これはコミュニケーションによる欲求充足を行えながらも、意図しなかった情報と出会えることがある状態だと言える。筆者はこれを **心地よい偶発性（Comfort Happenstance）** と呼んでおり、デフォルト・リモートにおいても、引き続き頼れる選択肢である。ただし、すでに述べたように、コミュニケーションの注入として行うべきだし、行ったとしても、そこでやりとりされた情報は透明にするべきだ。しかし、透明にするとの前提だとコミュニケーションはしづらい。録音と録画が行われている飲み会など喋りづらい。

折衷案として「業務時間中のコミュニケーションであるから透明化は受け入れよ」が使える。録音・録画でも、あとでサマリーを書いて公開するでも、聞かれたら答える＆そのタイミングで書いて残すようにもするでも何でもいい。ただ私たちは怠け者なので、透明化の作業をやってもらう運用では十中八九形骸化する。透明化もデフォルトで自動的に行われるほどに、道具を拡充した方が現実的だ。いずれにせよ、プライバシーも含めて、難しいトレードオフとなってしまう。

そういうわけで、仕事に関する話をそもそもコミュニケーションでやりとりしないのが良い。**コミュニケーションは、仕事とは切り離して、仕事に関する情報が交わされないような形で注入する**。そして、仕事に関する情報は、透明化を前提とした情報共有でやる。コミュニケーションはあくまでも充足行為であって、情報のやりとりの手段として頼るものではないのだ。

しかし、これでは偶発性がない。心地よい偶然性においては、仕事とは関係のない雑談から、仕事に関係する話に繋がっていくものだったが、これは使えない。なら偶発性との出会い方を

変えればよい。非同期的な情報共有による偶発性に頼ればいい。雑談とは雑多に話すことであるが、デフォルト・リモート用に最適化すると、こうなる――雑多に書く。これを **雑残（ざつざん）** と呼ぶ。雑多に残すと言ってもいい。雑談は同期的なコミュニケーションを用いた、コミュニケーション 1.0 的な偶発性の演出と言える。1.0 的、つまり特定の具体的な相手がいて、その人（達）だけを相手にする。雑残はそうではない。雑残は、非同期的な情報共有を用いた、コミュニケーション 2.0 的な演出である。2.0 的、つまりは不特定多数だ。誰といつやりとりするかなんてわからない。するかもしれないし、しないかもしれない。しかし、皆が同じ前提で、雑多に残しているため、色んな情報と出会うことができる。もちろん、情報として残っているため、IT で支援することも可能だ。それこそ生成 AI 経由で取り出せば良い。前述のキャストでも、この考え方は組み込んでいる。

味気ないと思うか。非人間的だと考えるか。そうではない。その部分はすでに述べているとおり、コミュニケーションの注入で満たせばよい。それ以外の、仕事に関する部分を、非同期的な情報共有で行うというだけだ。偶発性の演出も含めて行うというだけだ。

雑残でモチベーションなんか出るわけがないと思うか。そうではない。私たちには好奇心がある。興味関心がある。読書、インターネット、SNS に触れてきた人であればわかるはずだ。**情報だけで満たせる**。満たせるだけの情報を出せばいいし、やりとりができればいいだけだ。

というより、できるようにしなければならないと言った方が正しい。でなければ、心地よい偶発性が持つ「難しいトレードオフ」から脱却できない。トレードオフの理を壊すには、新しい概念が必要なのだ。雑談をアップデートする。といってもシンプルな話で、

> 雑談 = 雑に話すコミュニケーション + 偶発性の演出

から、

> 雑談 = 雑に話すコミュニケーション
>
> 雑残 = 雑に残すコミュニケーション + 偶発性の演出

に変えるのみ。雑談さんの肩の荷を一つ下ろして、雑残さんに任せるだけである。

まとめ

- 「情報の養育」へのシフトを目指す
  - 「同期的なコミュニケーション」から「非同期的な情報共有」へ
- とはいえコミュニケーションも必要なので、必要に応じて設ける（注入）
- 情報の透明性こそがあるべき姿であり、これで成立するあり方を考えていくべき
  - ティール組織の「生きた文脈」
  - 必ず全員に届くキャスト
  - 雑談よりも雑残（ざつざん）

Chapter-4 マネジメント(1/2)

マネージャーなる役割が当たり前に存在するように、仕事において「マネジメント」は避けては通れない。マネージャーというと、エンタメでは演者を世話する者を指し、ビジネスでは部下を管理監督する者を指すが、本章では後者を扱うものとする。前者は、あえて呼ぶならケアラー（世話係）またはエージェント（代理）であり、本書では扱わない。

さて、リモートワークとマネジメントの相性の悪さを体感する者は多いと思うが、そのとおり限度がある。デフォルト・リモートにおけるマネジメントもやはり再考しなければならない。

管理しなければしないほど良い

リモートワークとマネジメントレスは同義である

同義であるとは言い過ぎだが、おおむね的を射ている。そもそも **リモートとマネジメントは相反するもの** であり、リモートを手に入れたいならマネジメントは減らさねばならない。デフォルト・リモートならなおのことだ。

マネジメントとは何だろうか。「管理」である。管理を横文字で言った言葉にすぎない。では管理とは何か――と追求したところで、言葉遊びに溺れるだけだ。正解は無いし、辞書的意味を持ち出したところで役には立たない。管理とは何かを問うことは、人生とは何かを問うことにも等しい。不毛だ。必要なのは、管理なるものを自分なりに定義して、その前提で思考と行動に落とし込んでいくことである。本章では **管理とは制約に従わせること** と定義する。また、管理ではなくマネジメントを使うことにする。これは「マネジメント」という言葉の持つ、ある種の緩さを尊重したいためだ。逆に「管理」という言葉の堅苦しさではやりづらい。

ここまでを整理すると、デフォルト・リモートを目指したいならマネジメントを減らすべし、もっと言うと制約に従わせることを減らすべしと言える。制約という言葉も堅苦しいが、基準と捉えて差し支えない。**マネジメントとは基準に従わせること** である。基準といっても色々ある。100% にどれだけ近づいているかという進捗、守るべき文化やルールや手順、使いすぎても余らせすぎてもいけない時間やお金といったリソース、いつ何を誰に伝えればいいかというプロトコル（やりとりの仕方）など様々なものに基準が設定される。ちょうどピッタリになるよう守らされたり、上回らないように、あるいは逆に下回らないよう言われたりする。基準には幅があり、設定された範囲内に入らなければならない（あるいは範囲外に出なければならない）。

baserange

見てわかるとおり、基準は狭くして厳しくもできるし、広げて緩くもできる。もちろんずらせば範囲

そのものが変わって、基準内だったものが出ていったり、逆に新しいものが入ってきたりする。頻繁にずらしていては収拾がつかないから、通常は一度設定したものを使い続ける。また、一つの基準だけでは足りないことも多く、ご存知のとおりネスト（入れ子）が多発する。基準の中に基準があって、さらにその中に、とマトリョーシカのごとく基準が入り組んでいるわけだ。収拾がつかなくなって、形骸化する。まだある。そもそも基準が設定されているとは限らず目に見えないことも多いし、基準を満たしているかどうかの判定も人間による曖昧なものだったりもする。めちゃくちゃだ――と、範囲のたとえはこの辺にしておく。このたとえも便宜にすぎない。

言いたいことはただ一つで、**マネジメントは本質的に無理ゲーなのである**。よほど状況を限定するか、人間扱いをやめて駒として搾取するかしない限り、こんなものが成立するわけがない。

この無理ゲーに対抗するために、人類が取ってきたのが拘束だ。前章でも述べたが、時間と場所を拘束して一緒に過ごせば、同期的なコミュニケーションが使える。そうすると基準に使わずに、あるいは使うふりをして、リアルタイムにあーしろこーしろと言って制御できるようになる。**マネジメント≒リアルタイムなテコ入れ** というわけだ。本質的に無理ゲーだからこそ、リアルタイムにテコ入れをし続けて何とか凌ぐくらいしかできないのである。当然ながらリモートとは相性が悪い。リモートは少なくともリアルよりは拘束が緩いし、デフォルト・リモートに至ってはそもそも拘束をしない。マネジメントはできない。

だからといって、常に 100% マネジメントレスであるべきかというと、そうではない。重要なのはバランスだ。デフォルト・リモートでもコミュニケーションは排除せず、適宜注入する形でバランスを取る。マネジメントも同様で、基本的にはマネジメントレスでありたいが、必要に応じてやるべきだし、ゴッリゴリにマネジメントしなくては務まらない状況ももちろんある。誰かがやらねばならない。そこは役割の分担と連携を工夫することで吸収していく。詳しくは後の章で扱う。少なくとも全員がマネジメントしなければ・されなければならないわけではないし、極力なくすことが前提である。**デフォルト・リモートは、同時にデフォルト・マネジメントレスでもあるのだ。**

マネジメントのパラダイムシフト

マネジメントレスといっても、常に 100% マネジメントをなくせるわけではない。目指すべきはどこまでマネジメントするかというコントロールだ。できるだけマネジメントを減らすよう動的に調整していくと言ってもいい。そのためには、いくつか押さえておくものがある。マネジメントパラダイムとして整理していこう。

マネジメントは 1.0、2.0、3.0 の 3 つに整理できる。

パラダイム	解説
マネジメント 1.0	やり方を管理する
マネジメント 2.0	結果をレビューする
マネジメント 3.0	状態をモニタリングする

順に詳しく見ていく。

マネジメント 1.0

1.0 はやり方を管理するもので、仕事の仕方や働き方に強い制約が働く。所定のオフィスに出社する、所定の時間に勤務する、所定の時間に打ち合わせをする、使えるデジタルツールが限られている、プロセスが定義されていてその通りに従うしか道がない、アトランダムにマネージャーや顧客から割り込まれて対応を余儀なくされる――こういったことはすべてやり方が管理されていることと同義である。無論、仕事である以上、完全に自由に過ごすことは難しいが、それでも 1.0 は脱せるものである。たとえば出社よりもリモートを、コアタイムよりもフレックスを、利用可能なアプリや SaaS の緩和を、プロセスよりもガイドラインを、また割り込みはその場で処理するのではなく溜めておいて後で処理するバッチ処理的発想をすればよい。北米や西欧など残業を無能と評する文化があるが、同様に、やり方の管理も無能の表れにすぎない。

唯一の例外は、接客や生産など品質を厳格に揃えなければならない場合だが、この場合は拡大と安定のために人材が駒化する。アルバイトや非正規雇用の形で使い潰される構図はご承知のとおり。また、人間では限界があるので自動化を推進して機械化もする。機械は、あらかじめ組み込んだとおりにしか動けないものであるため、人間が従う以上にやり方を厳密に定義する必要がある。

要するに**マネジメント 1.0 は機械化**と言ってもいい。効率的ではあるが、非人道的だ。当然ながら普段のチームや組織運営で 1.0 をしても、上手くいくはずなどない。大企業や公務員など安定した組織であればステータスや報酬があるため、我慢してもらえる余地が多少大きくなるという程度だ。その安定神話も VUCA な現代では崩れつつあるし、余地が多少あっても健全であるとは言えそうにない、とは容易に想像できるだろう。**機械化により主体性は殺されるので、ボトムアップを期待できなくなる**。トップダウンしかできない。もちろん、この VUCA な現代に、トップダウンだけで何とかなるなんてことはない。搾取すればある程度の結果は出せるが、そんなことは当たり前だ。王政もそうである。カリスマであれば優れた結果を出せるだろうが、カリスマだろうと何だろうと王のあり方自体が間違っている。古代や中世ではないのだ。デフォルト・リモートが住み込み、出社に続く第三の水準であることはすでに述べた。仕事だからといって、王政的な暴政を許していいはずがない。

マネジメント 2.0

次のパラダイムがマネジメント 2.0 で、これは結果（成果物）をレビューする。クリエイターをイメージしてもらうといい。といっても、いついつまでに所定の成果物を出すことを要求されるし、そのための打ち合わせは 1.0 的ではあるものの、基本的にどう過ごしてどうつくるは自由である。やり方ではなく結果に焦点を当てている。

管理とレビューは違う。管理は、管理する者が制約を振りかざすことにも等しいが、レビューは主観的なものだ。レビューを行う者をレビュアーと呼び、管理と同様、外せない制約に従っているかどうかもチェックはするが、それ以上にレビュアー自身がどうしたいのか、どうするべきかも重

視する。つまり**2.0 は対話が発生する**。結果をつくる側と、それをレビューする側（レビュアー）がそれぞれ主観を持っており、これをぶつけて議論して、最終的な落とし所をつくる。上手く化学反応が起きて、一人ではいけない高みに行くことも多い。

2.0 は、単に機械化する 1.0 よりもはるかに大変なことだ。というより、人一人がレビューに使える時間などたかが知れている。従来の階層組織では――特に権限を握る上位者では到底務まらない。階層的な組織では 2.0 は難しい。より小回りの利く、小さなグループでなくてはならない。現時点での解の一つは、前章でも挙げた **ティール組織** だ。といっても、常にレビュアーと同じチームに居る必要はない。チームの違うレビュアーをスポットで頼ることもできる。この際に便利なのが、ティール組織において **助言プロセス** と呼ばれる考え方で、意思決定の権限はないが助言は行う。通常、レビュアーの方が権限が高く、レビュアーの指摘が事実上命令となってしまうことがある。たとえばマネージャーや経営者がレビュアーとなった場合は、それがそのまま鶴の一声となってしまったりする。助言プロセスではそうはせず、あくまでも助言だけ行う。意思決定権は、あくまでも成果物をつくる側にあり、レビュアーはたとえ上位者であっても助言者しかない。助言を踏まえて、それをどこまで取り入れるかはつくる側が決める。

先ほど 1.0 では主体性は殺されると書いたが、2.0 では逆で、主体性を尊重する。ただし本人だけに任せてしまうと偏ってしまうから、レビューの形でサポートする。できれば意思決定とは切り離して、助言プロセスのように助言だけ行えるのが良い。クリエイターがまさにそうであるように、主体性を持って取り組むことは大変だが、人間的だ。私たちは主体性に取り組めた方が快適だし、生産性も上がる。それは私生活を鑑みれば明らかだろう。家族や友人の言いなりでいい、という人は少数のはず。基本的に自分のやりたいように生活しているはずだし、それを尊重したくて一人暮らしや別居、そうでなくとも自分の部屋を設けることは自然な行動だし、仕事場でも自宅でもひとりになれない人はサードプレイスを求めたりもする。**主体性は重要なのだ。マネジメントだからといって潰していいものではない**。2.0 であれば主体性を潰さず、かつ個の偏りも修正できる。

2.0 ができるかどうかは権限委譲にかかっている。仕事の意思決定権をつくる側に委ねられるか。この一点に尽きる。委ねられるほどの実力がないなら、実力をつけてもらうための投資をしなくてはならない。もちろんマネジメント 1.0 の形で妨害してもならないし、そもそも 1.0 で済む部分は機械か、1.0 用の役割に任せればいい――と、これは結局組織のデザインに等しい。マネージャーであれば自身が抱えるであろう n 人のチームを、経営者であれば数十、数百、それ以上の組織をどうつくっていくか、また継続的に修正していくかの話になる。一から試行錯誤してもいいが、修羅の道だろう。まずは組織パラダイムとして知られているティール組織を学ぶのがてっとり早い。階層的な組織（ラルーの言葉で言えば達成型組織）では構造的に 2.0 が難しい。組織パラダイムそのものを変えた方がいい。

あるいは IT エンジニア、特にソフトウェアエンジニアに頼ってもいい。ソフトウェアエンジニアというと、私たちが普段使っているアプリをつくる人達であり、プログラミングを想起するかもしれないが、本質は違う。本質は概念をつくることだ。問題を解くために、どういう概念をつくってどう連携させればいいかを創造している。この概念の落とし込み方が違うだけだ。

engineering_is_concept

筆者はソフトウェアエンジニアこそが最強だと考えている。これほど概念の創造に慣れた人達もいまい。かつ、つくった概念をソフトウェアなどの実用的な世界に落とし込むのが上手い。また、落とし込むことを前提として考えるから、概念をつくる段階でも実用性が宿っている。いわば実用的な概念をつくるのが上手い。ソフトウェア開発手法であったアジャイルは今やビジネス一般にまで下りてきたし、リーン・スタートアップ(1)もソフトウェア・エンジニアリングを新規事業に持ち込んだものだし、IT インフラをプログラムコードで自動化する営み IaC(Infrastructure As Code)も同様で、手作業的だった IT インフラにソフトウェア・エンジニアリングの発想を持ち込んだことで自動化を実現し、現代の IT を支えている。ソフトウェアエンジニアが本気になれば、マネジメント 2.0 を実現する組織のデザインなど造作もないはずだと筆者は信じている。

マネジメント 2.0 の限界

2.0 が現代の最適解に思えるが、そうとも限らない。1.0 ほどではないが、2.0 でもそれなりのマネジメントが発生する。特に成果物に焦点を当てているので、要件が厳しくなるか、逆に曖昧だったりするとマネジメントも複雑になりがちだ。またレビュアーの実力、調子、相性にも大きく左右されるため属人性が高い。レビュアーが未熟だと役に立たないか、下手すれば足を引っ張ってしまうし、優秀であっても相性が悪ければ上手くいかない。問題 vs 私たちであるべきなのに、クリエイター vs レビュアーとなりかねない。これでは勝ち負けであり、政治と大差ない。実際にレビュアーを通すための工作に頭を悩ませた経験のある人や、それが日常となっている人は読者にも少なくないのではと思う。

さらにデフォルト・リモートでは、レビュアーがリモートに弱い点が致命打を与える。クリエイターの読者は共感すると思うが、レビューは多かれ少なかれ対面で行われるはずだ。出社できるよう会社の近くに住まねばならなかったり、フルリモートでも毎日のように打ち合わせが入ってしまうことも珍しくない。そしてそれを「レビューを良いものにするために、日頃からコミュニケーションや情報共有をして関係性を深めている」などと正当化する。間違ってはいないが、原始的かつ前時代的だ。単に現代的なやり方を知らないだけだ。本書はそのためにあるし、追々紹介していく。コミュニケーションと情報共有については前章で述べた。

話を戻して、2.0 もリモートと相性が悪い。2.0 は対話だからだ。**対話とリモートは相性が悪い**。対面で聞く・話すで行うリアルタイムなやり方には敵うはずもない。デフォルト・リモートでは対話という概念への偏重そのものを薄めなくてはならない。対話はコミュニケーションベース、つまりは欲求充足行為ベースの営みでしかなく、情報のやり取りという観点ではさほど最適ではない。そしてコミュニケーションは、必要に応じて注入すればいいとは前章で述べた。対話的なレビューも同様で、注入するのは良いが、これをデフォルトにしてはならない。しかし、2.0 自体が対話であるから、2.0 でいる間はどうしようもない。次のパラダイムが必要だとわかる。

もう少し寄り道させてほしい。2.0 は結果をレビューするものだが、ここにはしばしば 1.0 的なマネジメントが持ち込まれる。QCDS はご存知だろうか。Quality(品質)、Cost(コスト)、

Delivery（納期）、Scope（スコープ）の略で、いずれも 1.0 のように厳しめの基準をつくって、これに従うことを厳しく要求する。1.0 はやり方の管理だと書いたが、結果も併せて管理するパターンが実は多い。この場合は 1.0 的と言える。2.0 ではない。2.0 というからには、レビュアーとの対話がなくてはならない。QCDS の観点も参考であって、最終的にどうするかはレビュアーやクリエイターが決めるべきである――が、QCDS を使っている限りは、中々そうはならないし、QCDS の名のもとにやり方のみならず結果までも管理されている現状は、読者の多くも身に沁みていると思う。

代案として、造語だが SSS を使うのはどうか。SSS とは Skill（スキル）、System（仕組み）、Standard（標準）の略だ。QCDS が外に基準をつくっていたのに対し、SSS では中（働く私たち自身）の底上げとサポートを重視する。別の言い方をすると、スキルに頼った主体性を重視する。もちろんスキルだけで仕事が上手くいくほど甘くはなく、決め事や最適化は必要なので、そこを仕組みと標準で補う。本質的な仕事をスキルをもって主体的に取り組み、それ以外の些細は仕組みで楽をする、あるいは標準をつくってそれに従えば済むようにする。ここで標準とは、プロセス（手順）というよりはガイドライン（方針）のニュアンスである。プロセスを整備するくらいなら、仕組みを整備した方が良い。先でも述べたが自動化だ。しかし自動化はコストも技術も必要で難しいことが多いため、プロセスに従う役割をつくって、その人達に任せればいい。バックオフィスは仕事としては非本質的だが、必要なことを役割分担した例である。この考え方をもっと推し進めて、プロセスベースの仕事を委譲する。一例として雑務を挙げると、マネージャーという役割はどのチームにもいるだろうが、同様に、雑務全般を専任する役割がいてもいい。雑務にそこまで人件費をかけられるか、とはもっともな言い分だが、チーム全体に曖昧に押し付けるよりはマシである――と、役割分担の話に逸れてきたが、役割分担はデフォルト・リモートでも非常に重要であり、後で詳しく扱う。

マネジメント 3.0

マネジメント 2.0 の限界と、QCDS がしばしば 1.0 に逆行する点を述べた。次のパラダイムが必要である。それがマネジメント 3.0 で、状態をモニタリングする。

個人にせよ、チームにせよ、高いパフォーマンスが安定的に出る状態があるはずだ。これを **理想状態** という。3.0 では、理想状態をモニタリングする。理想状態から遠のいている場合は、速やかに対処する。つまり **理想状態に長く居続けることを目指す**。これは 2.0 以前とは明らかに異なる潮流であり、やり方や結果はマネジメントしない。理想状態が最高の状態なんだから、理想状態であれば問題ないよね、高い成果も出てるはずだよね、とそう考える。

3.0 に則った例は二つある。一つがセルフモニタリングだ。元はスナイダーによる自己呈示――つまりは自分を印象良く見せることの重要性を述べたもので（2）、自分をそのまま開示するのではなく、周囲の反応や期待に応じて表現を変えた方が良いとしており、後者のあり方をセルフモニタリングと定義している。現代では「自分の調子を客観的に把握するために記録を取る手法」との意味合いが強く、ここでもこちらを使いたいが、どういうときに不調になるか、あるいは好調になるのかを記録ベースで知ることで、不調を回避したり好調に再現性をもたせたりできる。これは何らかの成果というよりも自分の調子に注目しており、まさに 3.0 的と言える。もう一つ

が Thoughtworks 社による Engineering Effectiveness で、ソフトウェア開発の話になるのだが、出力ではなく入力を計測すると銘打っている。たとえば書いたコードの量や働いた時間を計測するのではなく、レビューにかかった時間、割り込み・中断の量、計画外の作業の量を計測する。これは理想状態を乱す因子を計測しているとも言えるだろう。

ここでマネジメントパラダイム各々がどこを対象としているかを整理する。

management_paradigm

1.0 では仕事の部分、やり方を管理していた。これは機械のやり方であり、統一的な効率化はできるが非人間的である。2.0 では成果物の部分、結果を管理していた。しばしば 1.0 的な基準を課されがちだが、上手くやればレビュアーとの対話に持ち込める。それでも本質的には成果が第一であり、成果を出すまでは終われない。クリエイターにも激務や搾取が多いのはご承知のとおりである。3.0 は違う。仕事をして結果を出す「私たち自身」を対象としている。私たちワーカーがファーストなのだ。別の言い方をすると、**3.0 はワーカーファーストに基づくベストエフォート** と言ってもいい。正しい、正しくないではなく、この第三のパラダイムを前提として仕事が成立するにはどうすればいいかを考えるべきである。

ベストエフォートで仕事が成立するのだろうか。する。ただし、継続的な議論と方向の修正もつけねばならない。この潮流を表現した言葉が、いわゆるアジャイルである。アジャイル（Ajile）は直訳すると「迅速な」「素早い」で、元は 2000 年頃から整備されてきたソフトウェア開発の手法や哲学を指す。

それ以前はウォーターフォール（Waterfall）が主流で、滝が流れ落ちていくように事前に計画した工程を一つ一つ消化していた。これはわかりやすく、統一しやすく、かつ管理されできればいいので、多くの無能を救うという意味でも必要悪であったと思うし、今なお疑うこと機能する場面は開発に限らず多い。しかし VUCA な昨今では相性が悪い。アジャイルの細かい定義や歴史は割愛するが、一言で言えば **直近何するかだけを決めて動いてみる** ことである。一週間や二週間といった単位（スプリントという）を定めて、次のスプリントで何するかを決めて、そのとおりに動く。動いてみた後は振り返りを行って、次も同じように何するかを決めて動く――この繰り返しで、行動という名の成果物を積み上げていく。このような世界観では管理の代わりに、対話が必要となる。終わりはない。その時その時で、皆でベストエフォートしていくだけだ。無論、会議や組織がそうであるように、適切なファシリテーションや意思決定ができないと地獄と化すし、方向性を上手く御せないと中長期的な価値にも繋がらない。あるいは撤退を見誤ってサンクコストを費やし続けてしまう。このあたりは組織の摂理でしかなく、アジャイルに限らず何であっても発生する。やり方と考え方を工夫して何とかすればいいだけだ。今回の場合、ベストエフォートでも仕事が成立するようなものを編み出せばいい。

ここでデフォルト・リモートが効いてくる。アジャイルとは「直近のスプリントにフォーカスしたベストエフォート」とも言え、そのためには継続的な議論と方向の修正を要する。問題は、これをコミュニケーションで担おうとするとコストが過大となってしまう点だ。散々述べているとおり、コミュニケーションは欲求を充足するための負荷の高い行為にすぎず、情報のやりとりという仕事の面では

最適ではない。実際にソフトウェア開発の現場でも、昨今はアジャイルが主流だが、毎日のように打ち合わせが走っている。ペアプログラミング（2人）やモブプログラミング（3人以上）といった共同作業的な手法が使われることすらある。全員が納得しており、魅力的な報酬もあるのであればそれでも（全員に高負荷を課しても）いいだろうが、本書としては第三の生活水準を志向しており、賛同しない。「いやいやアジャイルで残業がなくなったよ」と述べる者もよく見るが、残業がないから良いというものではない。毎日定時であっても高負荷なことはあるし、逆に消化不良で納得していない人もいるだろう。もちろんスプリントで決めたことを守るために忙しくなりすぎるのも論外だ。アジャイルはコミュニケーションの機会が増える分、拘束を増やす免罪符になりやすい。別にアジャイルだからといって、ビジーである必要などないのに。それでもコミュニケーションに頼っている以上は、構造的にビジーになりやすい。だからこそ、デフォルト・リモートなのである。コミュニケーションではなく、非同期的な情報共有によって仕事をなす。**それで成立する程度のベストエフォートを維持する**。もちろん、コミュニケーション自体を完全になくすべきではないから、適宜注入すればいい。

アジャイルにデフォルト・リモートを適用すると、物理的な余裕が生まれる。たとえば半日くらいは誰とも会議せずひとりで過ごせるし、一日中そうすることも難しくない。これをミュート・デイと呼び、デフォルト・リモートを測る目安の一つである。さて、ミュート・デイが当たり前に行える状態だと、ただのチームワークが**チームワークとソロワークになる**。チームワークは本番試合的であり、短期的に前に進むのには向いているが、木こりのジレンマに陥りやすい。ボロい斧を研いだり新しい斧にすればいいのに、木を切るのに忙しいからできないなどと言ってボロい斧で切り続ける。本番ばかりしていても成長や上達はない、あるいはできても向いている者だけだ。QCDS以前の潮流にKKD（勘・経験・度胸）があるが、こんなものは精神論でしかない。仕事だからこそ、本番以外の練習試合や個人練習にあたるものが必要である。それがソロワークであり、その名のとおり一人で仕事をする。内省、創造、あるいはカル・ニューポートがいうディープワーク(3)など、ひとりで深く集中して仕事を行う営みには様々な呼び名があるが、まさにそれらを指す。私たちは誰しも自分の強みを持っており、様々な情報を生み出せる。また自分が持つ文脈を最も尊重できるのも自分だ。だからこそ文脈を考慮しない正論は嫌いだし、文脈にはプライベートなものや言語化できないモヤモヤもあるため他人に共有するのも難しい。自分で扱うしかない。そうするからこそ、**真に自分の文脈と強みに基づいた情報を出すことができる。この情報をチームに持ち込んで議論するのだ**。これにより、チームワーク一辺倒だったあり方よりもはるかに高みにいける。すぐに結果は出ないし、瞬発的な対応能力も落ちるが、別に良い。すでに述べたように、それでも成立するようなベストエフォートをするのだと決めて、実践して、維持すればいいだけだ。

テネットをアップデートする

マネジメントパラダイムを三段階で整理した。最重要なのはやはりテネットで、3.0の部分でもアジャイル、ベストエフォート、ソロワークといったテネットをまさに示した。しつこいだろうが、デフォルト・リモートを実現するためには、まずはテネットのアップデートが必要なのである。その上で、それらのテネットでも成立するやり方や考え方をつくればいい。もちろん、できればソフトウェアなどの形で道具化をした方が敷居は下がる。

マネジメントに関するテネットはここまでで示せた。以降ではやり方や考え方の話に入っていく。

トピック指向

管理とは基準に従わせることであるが、マネジメント 3.0 だからといって基準自体が完全になくなるわけではない。基準はある。ただし緩く、かつ動的に扱うだけだ。

別の言い方をすると羅針盤やコンパスといった言葉が似合う。大体こんな感じでやりましょう、とラフな目標を決めて、とりあえず目指す。目標というと原則的に目指さねばならないニュアンスがあるが、そうではない。最初の目印として参考にはするが、鵜呑みにはしないと捉えるのだ。仮説検証という言葉があるが、この概念を目標にも適用する。いわば **仮説的な目標** だ。あくまでも仮説であって、目標そのものが正しいとは限らない。マネジメント 3.0 では、目標すらも仮説検証に組み込む。しかし、これでもまだ理解しづらい。目標という言葉は、どうしても大きな粒度を想起させる。よって、目標の概念そのものを放棄した方がいい。アジャイルだ。直近何するかだけ考えるのだ。これは何するかを小さく設定して、試してみることにも等しい。

そういう意味では **仮説的なタスク** と呼ぶ方が正しい。タスクにも原則的にやらねばならないニュアンスが染み付いているが、そうではない。仮説でしかないから、タスクのとおりにこなすとは限らないし、途中でやめてもいい。もちろん、タスクとして設定した分以上に高品質にこなしてしまうこともありえる。検証した結果どうなるかなんてわからない。その代わり、どうなったか、どうしたかは記録していくし、次に何をすればいいかも設定し続ける。そうして「仮説的なタスクとその検証結果」が積み上がっていく。数十どころではない、数百、あるいは中長期的になると数千ものタスクが積み上がることも珍しくない。小さく仮説検証を、意思決定を積み重ねていくのである。また、どのタスクがいつ役に立つかもわからない。タスクは部品でもあり、使い方次第で進み方を変えることもできるし、何なら並行させたっていい。

ここまでくると、タスクという言葉もふさわしくないとわかる。そこで **トピック（Topic）** という言葉を使う。直訳すると「話題」だ。一つの話題を一つのトピックで表現し、議論や作業など何らかの行動をして、結果を反映する。トピックにはタイトルと結果がある。タイトルが仮説で、結果が検証結果だ。仮説を合っていたかもしれないし、間違っていたかもしれないし、まだ何もしてないかもしれない。もちろん条件付きで合っていると言えなくもない、のような結果もありえる。いずれにせよ、トピックは何らかの情報をもたらす基本単位となる。またトピックは部品でもあり、**使うことができる**。あるトピックから別のトピックをつくったり、とりあえずコピペして複製して別パターンの行動をしてみたり、部品のようにトピックとトピックを組み合わせて別のトピックを見い出してもいい。ちゃんとした意思決定をする際は、根拠となるトピックを多数ピックアップして、だからこうなるといった言い方をすることもできる――と、このようにトピックを用いてやりとりすることを **トピック指向** と呼ぶ。

前章にてデフォルト・リモートでは「情報の養育」をすると述べたが、マネジメントでも同じことをやる。管理対象をトピックの形で表現し、これらを育てていくと考える。ただし管理対象というほど狭義ではない。トピックとは話題だ。話題として典型的な管理が必要なものも含まれるが、そ

うでないものも多い。極端な話、「朝食何食べてます？」トピックをつくって、皆が自分の朝食を紹介しあったりそこで雑談したりしてもいい。雑談的なトピックはトピックの一種である。逆に、オフィスの引っ越しに伴って、当月中に旧オフィスの私物の全撤去しなければならない場合は、当月中の完了が必要なため厳しめのフォローが要るだろう。こういうときも「旧オフィスの私物撤去」トピックをつくって完了をフォローすればいい。おそらく未完了の人には、追加のリマインドを送ることになる。あるいは単に期日を設定した上で残っているものは問答無用で廃棄する、としてもいい。廃棄までは定期的にリマインドするとのバランスもいいだろう。こういった活動を、どのようなツールと運用でやればいいかという話はあるが、いったん置いておく。ここで言いたいのは、トピックが極めて汎用的かつ局所的――特定の話題にのみフォーカスできる概念であることだ。

トピック指向とは、トピックという基本単位を使って日々やりとりをするものである。マネジメント1.0、2.0、3.0のすべてが起こり得るが、一方的なマネジメントにはなりにくい。なぜなら誰もがトピックを扱えるからだ。必要に応じてトピックをつくれるし、トピックには必要な人が必要なときに出入りすればいい。いつ誰が何を書き込み、意思決定者がなぜその決定を下したかも記録として残るため透明性も確保される。トピックを育てていけばいいため、無闇に拘束して原始的なコミュニケーションをする必要もない。あるいは必要であっても、そこで行われたことをトピック化すれば、続きを育てていける。

つまりトピック指向を使うと非同期的な情報共有を行いやすい。というより、トピックの概念を用いなければ不可能だろう。プログラミングと同じだ。プログラミングも、原始的なコミュニケーションのように長々とコードを書いただけでは限度がある。よほどの天才でもなければ、短期的で場当たり的なものしかつくれない。そうするよりも関数、構造体、モジュール、クラス、ライブラリといった形で処理単位を部品化して、組み合わせて使った方がはるかに良いものをつくれるし、品質もスピードも上がる。また天才であっても、書くコードが汚いだけであって、頭の中ではそういった部品的な整理はしている。この部品的で柔軟な理があるからこそ、ソフトウェアはこれほどまでに発展し、GAFAMなる巨人が生まれるほどの力を示してきた。人間の認知能力には限界があるし、コンピュータは論理しか扱えない。だからこそ、**認知しやすい粒度で情報を部品化して、これを組み合わせてより大きなことを成すというモジュラーのアプローチが重要となる。**

トピック指向について長々と紹介してきたが、本質は至ってシンプルで、このプログラミングの考え方を、情報のやりとりそのものにも適用するだけだ。といっても、プログラミングに馴染みがない人はピンと来ないかもしれない。これ以上の解説には立ち入らないが、可能であれば（小中学生向けの教材でもいいので）触ってみることをおすすめする。モジュラーの発想は、英語や数学など新たな学問を身につけるほどのパラダイムシフトをもたらす。たとえば「部品は必要に応じてつくれるよね」「構成されている部品の一部を差し替えることもできるよね」といった視座に立てる。自分が陥っているテネットを疑うこともなく、盲目的で受動的なあり方から脱せない人は多いが、ここから脱せるようになる。IT、ICT、DXなど現代はデジタルを強調する時代でもあるが、デジタルはモジュラーだ。この発想を理解できれば、ついていける。部品の把握や組み合わせ方のスキルは大変なこともあるが、それだけである。

トピック指向を実現するためのやり方、考え方、道具

続いて、トピック指向を実現するためのやり方、考え方、そして道具について見ていく。先に結論を述べておくと、2025 年現在であっても道半ばであり、キラーと呼べる道具はまだない。それでも真にトピック指向を、マネジメント 3.0 を、その先のデフォルト・リモートに手を届かせたいならば、模索していかねばならない。

直接的なツールはない

まず始めに道具（ツール）――特にソフトウェアに頼りたくなるが、トピック指向を上手く実現できるものはない。

手段自体は概ね揃っている。筆者は QWINCS（クインクス）と呼んでいるが、情報をやりとりするツールは以下の 6 種類に大別できる。

- Q&A
- Wiki
- Issues（チケット管理）
- Note（共同編集可能なノートやドキュメント）
- Chat
- Sticky board（Miroのような二次元的なキャンバス）

QWINCS の詳細は後の章で扱うこととする。どれを使ってもトピック指向は行えるが、ツール側の機能と設計が足りず、人による運用負担が大きい。情報のやりとりは日常的に行うものであり、手作業で運用するには無理がある。昔はソフトウェアなど存在せず、すべて手作業で担っていたが、情報化社会とは生活水準もスピード感も違う。現代において担わせるのは（意図的に搾取的な構造を採用しない限りは）非現実的だ。資本主義としては正しいかもしれないが、本書としては論外とさせていただく。現実的には、せいぜい十分な資金力、堅固な文化、優秀で替えの利く駒を多数持つ大企業であれば行えるくらいだろう。JTC――Japanese Traditional Company という揶揄的なネット用語があるが、日本の伝統的な大企業はまさに好例だ。読者の中にも該当していそうな会社の社員がいるだろう、そして力技ゴリ押しの運用がまかり通っている現実や、そんな現実を変えていくことの構造的な難しさを体感しているかと思う。こんな力技は伝統的な巨人だからこそ成せる伝統芸能であって、現代ではもう汎用的な仕事術ではない。もっと踏み込んだツールが要る。

では、なぜトピック指向を行えるソフトウェアが生まれていないのだろう。すでに技術と方法は揃っている。リソースをかけさえすれば実現できる段階にあるというのに――。何度か強調しているとおり、テネットである。ツールの前に、テネットをほぐさねばならない。唐突だが、5 人チームがいて、あるテーマを議論するとしよう。仮に無限に費やせた場合に、論点を 30 個出せるとす

る。これを **議論のキャパシティ** という。キャパシティに近づければ近づけるほど良いと考える。無論、現実はそうもいかず、たとえば 1 回の打ち合わせで 3 点しか出なかった、といったことがざらにある。同期的なコミュニケーションの限界だ。打ち合わせの場では同時に一人ずつしか喋れないし、喋った内容も覚えないといけないし、非言語情報も常時踏まえなくてはならない。ハンデを背負っているようなものだ。仮に 30 の論点のすべてを議論しようとなると、一体どれだけの時間がかかるのか想像もつかない。議論のキャパシティは持て余されている。

この持て余しをトピック指向で解決する。単に論点としてトピックをつくればいいだけだ。いきなり 30 個をつくることはできず、まずは仮説的なトピックや、題名すらない雑多なトピックをつくってみる。読み書きを通じて議論していく。次第に他の論点が見えてくるから、それもまたトピックとしてつくる――。つまりトピックを読み書きできる場に皆で集まって、各自が自由に読み書きする形となる。聞くも話すも必要ない。場に集まって、適切にトピックをいじるだけだ。情報を養育すると述べたが、情報の養育とはトピックの養育に他ならない。**トピックの養育というテネットを持つことで、議論のキャパシティを最大限まで生かせるようになる**。ここがスタートラインだ。

ということは、このテネットで情報のやりとりを完結させるには、を考えなくてはならない。そのようなツールをつくらなければならないのである。QWINCS のどれを使っても、おそらく捌ききれないことは想像に難くない。養育のテネットに基づいてつくってないのだから当たり前だ。一例として **1000 の壁** を挙げる。従来のツールでは 1000 個の情報――ファイルでもドキュメントでもページでも何でもいいが、1000 個以上を抱えると個人では把握しきれなくなる。整理の仕方としてリスト（一覧）、テーブル（表）、フィルタリング（絞り込み）やソート（並び替え）、カテゴライズ（箱に入れる）やタグ付けなどが知られているが、これらを用いても同様に、1000 の壁は越えられない。越えられないから取捨選択が必要で、アーカイブやミニマリズムといった「諦める」営みが顔を出す。要するに **結局は人間の認知能力に限界があるため、多少ソフトウェア側を工夫したところでたかが知れているわけである**。スポーツ選手がそうであるように、知的な労働者として訓練を積めばもう少し限界は引き上げられるが、これこそ万人向けではない。そもそも人にはモチベーションというものがあり、かつ、このような「仕事の仕方」でしかないことに熱意を注げる酔狂な者は少なくない。

1000 の壁を越えるためには、どのような発想をツールに盛り込めばいいのだろうか。現時点で使えそうなヒントは二つある。

一つが Cosense に代表されるようなネットワーク的アプローチだ。情報と情報を繋いで、全体としてネットワーク構造にする。脳のつくりや人間関係からウェブサイトまで、ネットワーク構造は身近なものだ。つなぎ方が意味的にしっくり来るものであれば、少ない経由数で目的の情報に辿り着ける。記憶がまさにそうである。

つまりトピックもネットワーク状に扱えばいい。日頃読み書きしているトピックとトピックを繋げばいい。たとえばトピックがページとして表現されているなら、URL があるはずだから、URL を書くことで接続になる。ページ A にて、ページ B の URL を書いたら、これは A → B への接続に等しい。A と B できちんと読み書きしているのであれば、記憶として定着するだろう。B を思い出せなくても、A を思い出せたら、A から辿って B に辿り着ける。また、より親切にしたいなら、ページ

B でも「A からリンクされている」旨を表示するといい。これをバックリンクと呼び、Obsidian など一部のノートツールがサポートしている。Cosense はまさにこのあたりの思想を体現しており、万のページをつくっても破綻しない情報空間をつくりだせる。筆者の個人ワークスペースも 2 万ページを超えているが、問題なく続いており、日常生活から本書向けの研究まで、あらゆる情報を扱ってくれているし、必要に応じて引き出せる。もう Cosense なしには生きていけない体になってしまった。

そして、もう一つは生成 AI である。活用されている方はご存知のとおり、生成 AI は大量の情報からの要約や変換を、自然言語の指示ベースで行うことができる。内部的にはただの確率計算で、膨大なインプットから統計的に「次に最も来る可能性が高いもの」を出しているだけだが、そのインプット量とチューニングが凄まじいおかげで、人間にとって比較的自然な、あるいは人間顔負けのアウトプットを出せてしまっている。内部が厳密にどう動いているかは人間には理解できないが、大体上手く動いているならそれでいいのである。そういうものと受け入れて、活用すればいいのである。量子みたいなものだし、自然科学だってそうだ。よくわからんけどそうなってる。そういうものだと受け入れて、それを使ってさらに生み出していく。その積み重ねだ。すでに生成 AI でも、こういう命令を出せばこんな感じで結果が返ってくる、のような体系が整理されており、プロンプト・エンジニアリングと呼ばれる。プロンプト（命令）をエンジニアリング（工学）的に捉えている。

さて、このような世界観では、人間がちまちまとトピックをつくって、トピック同士を繋げて、などと作り込む必要がない。適切な指示さえ書ければ、ほしいデータは何でも引き出せてしまう。実際に筆者も論文や長い記事を食わせて一行、三行、十行で要約してもらったりといったことはよくやるし、セルフモニタリング目的で日々つぶやいた（自分でも読み返したくない長い）ログを与えた上で、調子が良いときや悪いときの傾向を教えてもらったりもできる。最近では選挙――東京都知事選での活用事例もあり、ブロードリスニングなる概念も使われ始めていて、何千人もの意見の傾向をすぐに可視化することさえ手に届いている。

当然ながらトピック指向にも応用は可能だ。何十何百ものトピックを与えて傾向を可視化してもらったり、意思決定の案を出してもらったりできるだろうし、読み忘れたトピックを教えてもらったり、書き込みはしないが追いかけておきたいトピックを定期的に知らせてもらったりもできる。人間自身が貧弱な認知能力を丁寧に使いこなす泥臭さは圧倒的に軽減される。

ここまでをまとめよう。

- 道具（ツール）として QWINCS があるが、どれを使ってもトピック指向には届かない
- テネットのアップデートが必要である。議論のキャパシティを最大限まで引き出したい
- 従来のツールでは 1000 の壁を越えられない。人間の認知能力に限界があるため
- 越えるために使えそうなのがネットワーク構造と生成 AI である

つまり、**ネットワーク構造と生成 AI サポートを取り入れたトピック指向ツールがあれば良さそう** だとわかる。

それでも人間の役割はなくならない。最終的な判断は人間が行うものだし、生成AIに渡す情

報自体も人間がつくる必要がある。これすらも生成AIにつくらせることも不可能ではないが、私たちは人間であって機械ではなく、文脈のない正論に従える生き物ではない。ディストピアでもない限り、「己の意思を言語化した情報」を使うとの前段が必ずある。ただ、それをトピック指向として行う必要は薄いだろう。トピック指向は、人間の認知的限界をカバーするために部品化とネットワーク化に頼るものである。生成AIに認知的限界はない。なので、同期的なコミュニケーションの録音を全部与えたり、それこそ普段の様子を全部録画して与えても扱い切れる（現時点ではまだ実用的ではないか、少なくともバックグラウンドで十分な処理が必要な段階だが）。ただ、そうはいっても、人間の認知的限界の部分でボトルネックになる点に変わりはなく、ここを緩和するのに、やはりトピック指向は使える。生成 AI 時代であっても、トピック指向には意味がある。

マネジメントよりも意思決定とファシリテーション

トピック指向は情報の構造にすぎず、どの情報を採用して何を成すかまでは支援しない。また、いつ誰がどのトピックをどのように更新するかの支援も無い。構造だけでは行動にはつなげられない、あるいはできる人とできない人、やる人とやらない人に分かれてしまう。一部の人だけが使う「マイノリティ」の域を出なくなる。それではデフォルト・リモートにならない。万人がメインの手段として使うレベルで整備しなくてはならない。

具体的には意思決定とファシリテーションである。意思決定とは、何をやるかを決めることである。これは同時にやらないことを一時的に、あるいは恒久的に捨てる決断でもあり、やることや捨てることに対して責任を抱えるとも言える。次にファシリテーションとは、自律的に動けない人に対して、こう動いてくださいと援助することである。背中を押す程度の緩いものもあれば、無理やり腕を引っ張るような強いものもある。

トピック指向のツールには、事実上意思決定とファシリテーションの仕組み（役割含む）も盛り込まねばならない。直接盛り込むか、別にツールをつくってカバーするかは本質ではないが、仕組み自体は必須だ。従来は同期的なコミュニケーションによって、リアルタイムかつ非言語情報も踏まえた濃密な過ごし方をもって行っていたが、デフォルト・リモートではやらない。かといって 1.0（やり方の管理）や 2.0（結果のレビュー）といったマネジメントにも頼りたくない。どちらも完全にゼロにはできず、部分的に頼ることにはなるだろう。あるいは、そういう役割を設けて専任させる。それでもオプションでしかなく、メインの手段としてはならない。正しい、正しくないではなく、デフォルト・リモートを実現するためには、オプションの域に留めねばならないのだ。すでに述べたが、そもそもマネジメントをせず、ベストエフォートで済むようにしたいのである。そういうわけで **マネジメントの代わりに、意思決定とファシリテーションを行う。**

意思決定

意思決定を行うには、トピックに意思決定を書き込めばいい。通常、意思決定には意思決定者がいるため、これは事実上 **意思決定者がトピックを扱えるかどうか** の議論になる。対面で喋らないといけないなどというテネットを持っているうちは不可能だ。関係者を集めて、会議を

開催して、面と向かってこうしましょうと言うかわりに、その議論を表現したトピックをつくって、意思決定者がこうしましょうと書き込むのである。書き込むだけで完了だ。いちいち関係者を拘束して、非言語情報も交えて伝える必要などない。意思決定の情報が伝われば十分なのである。もちろん意思決定を行う（書き込む）前の議論やキャッチアップや相談もトピック上で行うし、決定したことを関係者に知ってもらう必要もある（こちらはファシリテーションで扱う）。つまりトピックをもって意思決定とその前後のやり取りをできるようにしたい。

文字どおりの IT リテラシーも重要だが、最大の障壁はテネットである。目安としてミュート・デイを紹介したが、**意思決定者もミュート・デイが行えるほどに非同期的な情報共有に習熟せねばならない**。意思決定者の好例はマネージャーや経営者だが、この人達は同期的なコミュニケーションが染み付いている。一日一回も打ち合わせをしない、など耐えられることではないだろう。そんなテネットを壊さなくてはならない。まずコミュニケーションという欲求充足は注入により各自満たしてもらうようにする。自分が満たしたいがために、部下やメンバーを集めるなどあってはならない。職権乱用とも言えよう。次に、非言語で説得するのではなく、言語で納得させるよう発想を切り替えてもらう。意思決定に正解に無いし、対面も使わないので、対面で行えていた非言語ゴリ押しによる感情的な（瞬発的な）、あるいは情緒的な（持続的な）説得も使えない。かわりに、十分な文脈と論理を残すことで、言語情報として納得してもらわねばならない。当然ながら言語化とトピックへの書き込みが必要で、喋るかわりに、書き込む時間を相当取ることになる。一日一回も打ち合わせをしないかわりに、一日何十ものトピックに書き込むことになるだろう。そういう過ごし方に耐えられるようになるべきだ。できるかどうかで言えば、できるはずだ。よほど適性がないか、特殊な事情や特性があれば別だが、稀だと思う。なぜできないかというと、テネットが邪魔をしていて行動しないから、鍛錬しないからに他ならない。必要性については、すでにデフォルト・リモートとして述べてきた。あとは実現のための行動だけである。従来のテネットが邪魔をしているから、こうして言語化していることで自覚を促している。

ファシリテーション

次にファシリテーションだが、こちらはテネットよりも手段が重要となる。トピック指向は、一言で言えばタスク管理ができるなら問題なくこなせる。いつ、どのトピックを読み書きするか、を全部自己管理できればいいだけである。仮に直近アクティブに扱うべきトピックが 30 個あるとしても、バランスよく読み書きすればいいし、パンクするなら調整をすればいい。

このように単独でコントロールしていける能力を **自律性** と呼ぶ。社会人なら自律性くらいあるだろう、ゆえにファシリテーションなど不要だ――と言いたいところだが、そんなことはない。**自律性を持ってない方が圧倒的マジョリティである**。そもそも、自律性を持っていないからこそ、安易にコミュニケーションにすがって、人や場といった「強く働きかけてくれる存在」に頼っているわけだ。学校の教室、あるいはオフィスのフロアや自席の島が特にわかりやすいが、その場にいるだけで場が働きかけてくれる。筆者はこれを **雰囲気リマインド（atmosphere remind）** と呼んでおり、多くの人が無自覚に頼っているものだ。

よほど独立的な人間か、ある種の特性でも持っていない限りは、これに頼らず過ごすのは不可能ではなかろうか。と、そう思えるくらいには強力な手段である。当然ながらコミュニケーションを

廃するデフォルト・リモートでは使えないので、代わりの手段が要る。それがファシリテーションだ。**ファシリテーションとは、自律的に動けない人達を動かすことである**と定義したい。

ファシリテーションにはギビング（Giving）とリマインド（Remind）がある。それぞれ詳しく見ていく。

ギビング

ギビングとは、お題がないと動けない人――「探索ができない人」を助けるために、お題を与える（ギブする）ことを指す。メンターやコーチは適切な問いを与えることで相手を支援するが、これもギビングの一種と言える。いわば特定の相手向けの密なギビングであり、「問い」の形で与えている。別の例として、匿名掲示板やコメント欄などに雑多に主張や提案が書き込まれることがあるが、これも不特定多数へのギビングだと言える。さらに、ブレストなど発想法を用いたワークは、参加者全員がギビングを行っているにも等しい。参加者の□さんが書いた付箋は、他の全員に向けてのギビングとみなすことができる。だから□さんも□さんも、その付箋を使って自由に発想を膨らませればいい。許可や遠慮なんて要らないし、する方が舐めている。

ギブアンドテイクという言葉があるが、ギブにはテイク――受け手の存在も欠かせない。与えたところでスルーされては意味がないし、表面上受けているように見えても内心でスルーしているならば、やはり意味がない。もちろん、与えたその時に今すぐ受けて使えるとは限らないし、これを隠すために同期的なコミュニケーションでは取り繕いや心理戦、たとえば「丁寧で柔らかい言い回し」の作文とその注意深い発言などに労力を費やすこととなる。不毛だ。ワークなど意図的に同期的なコミュニケーションを行う場合以外は、**ギブとテイクは分離した方がいい**。つまり非同期的な情報共有だ。トピックの形で、あるいはトピックの中にギブしておき、それを受け手が自分のタイミングで使えばいい。お題はその場でリアルタイムに与えられるものではない。トピックとして与えられているものであり、どのお題をどう使うかは受け手次第なのだと考える。

というわけで、探索ができない人たちのためにお題を与えないといけないのに、そのお題はどこかに散らばっているので自分で探せ、との構図になってしまっている。いわゆる「服を買いに行くための服がない」だ。この**最初のハードル（エントリーハードル）**を壊すには、教育と整備しかない。買いに行ける程度の服や、それが手に入る場所を近場に（何なら自宅に）つくるなり、普段自宅で来ているだらしがない服で外出してもいいのだとのテネットにアップデートすればいい。アップデートしたところで、最初は恥ずかしさでまともに歩けないだろうから、少しずつ練習して鍛えればいい。泥臭いし、乱暴だが、エントリーハードルはそうしないと越えられない。服のたとえから話を戻すと、たとえば「お題がなくて動けないときにとりあえず見る場所」を決めておいて、動けないときに見に行けばいい。そういう場所を探して、手間なく見れるよう整えて、必要なら操作も練習して、とやればいい。場所を思い出せないのなら、暗記学習のように定期的に復唱でもして刻み込むなり、練習の機会をつくって動いてもらえばいい。こういった初動に同期的なコミュニケーションが必要であれば、そういうイベントをつくって注入すればいい。**デフォルト・リモートにおける、同期的なコミュニケーションの典型的使用例の一つは、エントリーハードルを越えるときだ**。小難しく言っているが、リモートワークが盛んな組織でもオンボーディング時では集合の機会を増やすだろう。それと同じだ。

もちろん、大前提として、そのとりあえず見に行ける場所なるものが存在し、すでに十分な情報が残されていなくてはならない。事前に全員分を用意することなど不可能であり、**事実上自分なりの情報基地が要る**。ノートアプリ等を重用することで実現する PKM（Personal Knowledge Management）がまさにそうだが、日頃から自分なりに「仕事と自分の文脈」と「得た知識や知見」と「どういうときにどこが使えそうかの道しるべ」をまとめておくのである。そこまでしておくからこそ、こういうときはこの辺を見ればヒントになる、といったパターンを蓄積できる。一部の才人は脳内だけで完結できるようだが、通常は難しい。だからこそ PKM のような概念があるのだし、似た考え方や事例は、本書の読者であれば一度くらいは見聞きしたことがあるのではないか。要はそれなりに手垢のついたジャンルではあるのだが、一言で言えば、トピック指向を個人で行えばいい――が、これを万人に要求するのも非現実的だ。趣味として楽しんでいた時期のある筆者でさえ、正直勘弁してほしいと思う。情報の基地をメンテナンスすること自体は手段でしかない。やらずに済むに越したことはない。

やはり生成 AI がほしい。あなたがお題がなくて動けないとして、その状況を見て「ああ、なるほど、であればこういうお題はどうです？」とか「この辺を見たらたぶん見つかると思いますよ」といったことを提案してほしいのである。人間では担えまい。よほどの役職者やお金持ちであれば秘書を雇えるだろうが、これも現実味がない。生成 AI に担ってもらえばいい。自分の状況を生成 AI に見せるだけで、ギビングをしてくれる――そんな機能なりサービスがあればいい。現状でも実現できなくはないが、情報の受け渡しがまだ煩雑であり、人間が自分で言語化したりコピペしたりといったことが求められる。万人向けではないし、慣れた者であっても形骸化しがちだ。人間は怠惰なので、少しでも、それこそ 0.1 秒でも手間は減らさねばならない。これはシームレス（継ぎ目が無いこと）の追求とも言え、非常に重要なテーマであるため後の章にて詳しく扱う。

リマインド

ファシリテーションとしてギビングを示した。「お題がないと動けない人達」に「お題を与える」ことで動いてもらおう、というアプローチである。特にデフォルト・リモートではギブとテイクを分離し、その場でギブしてもらうのではなく、事前に残されているヒントを各自が読むことで使うとの考え方を述べた。エントリーハードルの問題があり、生成 AI の活用で軽減できないかという形で締めている（正直言えば半ば逃げている）。

もう一つのアプローチとして、リマインドを示す。

リマインドとは、本人が事前に設定した用件を、あとで本人に伝えることを指す。本人で完結していることに注目してほしい。他者からの命令や、場の同調圧力、その他搾取的な強制などは含まれていない。それら原始的なマネジメントを使えば、ファシリテーションなどどうとでもなる。そうじゃなくて、ここでの議論はマネジメントレスでベストエフォートなあり方のはずだ。その前提で、自律性の欠如を補完するために、リマインドが必要だという話をしている。具体的には、自分が主体的に用件（タスクと言い換えてもいい）を設定し、でも設定しただけだと忘れてしまうから、忘れないように、というより忘れても思い出せるように「後で伝えてもらう仕掛け」を仕込むのである。

リマインドとして最もわかりやすいのは目覚まし時計だろう。起きるという用件を、音を鳴らす形で起こす。それも指定した時間に起こす。スケジューラーなどアプリでは予定の n 分前にその旨を表示してくれるパターンが多く、これは事前に設定した予定を、直前になって教えてくれている構図になる。他にもロケーションリマインドと呼ばれる、GPS と連携して、特定の地理的範囲に入ったら・出たらお知らせしてくれるものもあるし、同居人に「明日は 7 時に起こして」と頼むことで仕込む人間リマインドや、外出時にゴミ捨てを忘れないために玄関にゴミ袋を置いておくといった動線リマインドも知られている。名前をつけるまでもない、自然な営みであり、のやり方も様々だ。

さて、このリマインドをトピック指向で使いたいわけだが、どうするか――**トピックの追加や読み書きをリマインドさせる**。

例として、直近 30 個のトピックと付き合わないといけないとしよう。今すぐに 30 個すべてを更新するのは難しい。気合でひとまず一周することはできるだろうが、各トピックの状況は変わる。今日明日くらいで詰めたいトピックもあれば、一年以上かけて育てていくものもあるわけで、これらを踏まえながら更新し続けるとなるとお手上げのはずだ。タスク管理ができる者なら「できる」のだが、できる人は少ないし、できることを前提としてはならない点はすでに述べた。できるようにするためにリマインドでカバーする。では、どのようにリマインドを仕込めばいいだろうか。

結論を言うと **自分なりにリマインダーを養育する**。

単純に考えるなら、トピックごとに「これは明日また教えて」「これはあとでやりたいので 30 分後に教えて」「これは当面はいいけど、忘れたくないから、そうだなぁ、10日後にまた教えて」などと設定していけばいいが、これは一回限りである。トピック指向は恒常的に続く。かといって、こういう細かい設定を今後ずっとやり続けたいかというとノーだし、できるかというとノーだろう。何度も言うように、自律的に設定を継続できるなら苦労はしない。では定期的な設定にするのはどうか。「明日また教えて」は「毎日教えて」とみなし、「30分後に教えて」は「毎日業務中は30分ごとに教えて」とみなせばいい。悪くはなさそうだが、今度は逆にうっとうしい。うっとうしいものは次第にスルーし始め、形骸化してしまう。形骸化を防ぐためには、自然なタイミングでなくてはならないし、できれば刺激が促されるような予期せぬタイミングも演出してほしい。ならば完全にランダムにしてしまうのはどうか。10秒後に知らせてくれるかもしれないし、5時間後に知らせてくれるかもしれない。いつ知らされるかは読めないが、知らせていい時間帯と密度を設定できれば、ある程度は使い物になるのではないか……というと、そんなことはない。特に深く集中したい場合は割り込み自体が害悪であり、この間はリマインドされてはならない。あるいはしてもいいが、あとでチェックできるようにしたい。

と、こうして考えていくと、どうも自分なりの設定を模索せねばならなそうだとわかる。当然ながら手作業で行うのは大変だ。現状では定期タスク系のツールが使えるだろう。タスクシュート、ルータム、Routinery を使って、トピックごとにタスクを（定期タスク）としてつくる。出現頻度は適当に設定しておく。すると毎日、今日消化するタスク（トピック）が表示されるから、やるかやらないかを決めればいい。私たちはタスクの登録と頻度の設定をすればいいだけで、あとはツール側がその頻度に基づいて毎日忘れずに表示してくれる――と、これで済むならいいのだが、お

そらくは済まない。これでもまだまだ手間が大きい。特にタスクシュートや協会や認定トレーナーが存在するほど煩雑な体系となっている。筆者も自分なりにこの手のツールを模索していた時期があるが、本質的に煩雑なのはたしかだと思う。そもそもこれで上手く人は、タスク管理ができる人だろう。最初から苦労はしないか、多少苦労するだけでものにできる。もっと楽な手段でなくてはならない。

現状そのような手段はないが、提案はある。**カジュアルリマインド** だ。カジュアルリマインドとは、その名のとおりカジュアルにリマインドを設定することで、リマインドしたときに「次はいつリマインドするか」も指定する。

たとえばトピック A のリマインドを設定したい場合、まずは「2時間後」に設定する。そうすると 2 時間後にリマインドされるが、そのときに A をやるかどうかを判断する。ここからが重要だが、**次の** リマインド時期も指定する。選択肢としては 2-20-200-2000-20000 を使うとしよう。つまり 2分後、20分後、200分後（3.3時間）、2000分（≒33時間）、20000分（≒2週間）の中から選ぶ。あとですぐにやりたいなら 2 や 20 がいいだろうし、今日はもういいなら 2000 だろうし、当面要らないなら 20000 を選ぶだろう。いずれにせよ、直感的に選べばいいだけなのですぐにできる。このようなリマインドは、半永久的に続くことから **エンドレス・リマインド** と呼ぶ。もちろん完了や中断などやめたい場合は、単に次の指定をなくせばいい。選択肢として「もうリマインドしない」も置くとわかりやすいだろう。

エンドレス・リマインドであれば、私達は何十ものリマインドを抱えることができる。これを使って 1-トピック 1-エンドレスリマインドを仕込めばいい。もちろん、2-3 個のトピックを一度に更新したいならそれらで 1-リマインドにしてもいいし、「この辺のトピックを更新する」みたいな曖昧な用件でリマインドしてもいい。ともかく、エンドレス・リマインドであれば、何十と抱えることができる。最初のうちは大量にリマインドされてうっとうしいし、「7つくらいリマインドが積まれてて消化が大変なんだが……」なんてことにもなるだろうが、次の時期は自分で設定するため、次第に適切なペースに収束していく。**養育** と書いたのはそのためだ。また、設定は見返すこともできるので、「この辺は明後日にしてたけど明日にするか」とか「今必要になったのでリマインド待たずに今やるわ」といったこともできる。

当然ながら、エンドレス・リマインドを自分で設定するのは面倒である。トピック指向を行うツール側でできるといいだろう。たとえば今見ているトピック（を表現したページ）にて、選択肢のボタンがもう表示されていて、押すだけで設定できるといい。この設定すら面倒、あるいは不向きな受動的な人は、マネージャーや他メンバに設定をお願いしてもいいだろう。何なら生成 AI に設定してもらうことだってできそうである。

このようにエンドレス・リマインドの概念を導入することで、一日何十以上ものリマインドを抱えられるようになる。たとえ自律的にタスク管理ができない人でも、エンドレス・リマインドであればリマインドされたことに反応するだけで済む。あくまでもポテンシャルの話だ。「万人が何十件抱えるべし」と言いたいのではなく、必要なら抱えることができるのだと言いたい。トピック指向では多数のトピックを扱うことになるし、従来の働き方のように原始的な拘束やマネジメントで急かすものではない分、各自が主体的に多くのトピックに手をつけることで生産性をカバーする。多く抱えられる能力があるに越したことはない。このリマインドのあり方なら可能である。

カジュアルリマインドが当たり前に使われる世界線では、私たちはリマインドに従って動く働き方になるだろう。従来はメールの受信箱、チャットの通知欄、予定表、プロジェクトのスケジュール管理表、あるいはオフィスなど雰囲気リマインドで動いていたが、ここに「各自が持つリマインドに従って動く」という新しい駆動が追加される。従来ほど強要的でなく、しかし自由すぎて自己管理できないと忘れてしまうものでもない、第三の塩梅と言える。

発想法的な営み ～スプレディケーション～

デフォルト・リモートの鍵は情報共有にある。もっと言えば、皆が情報を出し惜しみせず出せるかどうかが肝心である。そのために欲求充足行為であるコミュニケーションをサブの手段に据えて、かわりに非同期的な情報共有をメインにする。また、プロトコル（共通的な取り決め）としてトピック指向を使えば便利であるとも述べてきた。これは古典的なマネジメントからの脱却にも等しく、むしろマネジメントレスを推進するものである。それでも無秩序で済むほど単純ではなく、意思決定とファシリテーションが肝ということで深堀りしてきた。

ここまででおおよそ出揃ったが、まだ足りないテネットがある。ブレストなどアイデア出しに代表されるような発想法的な営みである。便宜上 **スプレディケーション（Spreadication）** と名付けたい。スプレッド的（発散的）なやり取りを指し、ブレストのようなやり方を一般化したものだと思ってもらって構わない。従来のテネットとしてコミュニケーションとドキュメンテーションがあるが、どちらもトピック指向にはあまり向いていないため、新しいテネットが要るのだ。

まずスプレディケーションの解説をしばらく行い、ひととおりできたところでトピック指向との関係につなげていきたい。

コミュニケーションとドキュメンテーションの間

違いを比較することでスプレディケーションに迫っていこう。

コミュニケーションとは、本書では拘束を伴う欲求充足行為であると述べたが、別の本質も含んでいる。二者間でのやりとりであるという点だ。具体的な相手が常に存在しており、その人のための情報を出す。三人以上とか、チャットツールを使うとかいったバリエーションこそ豊富なものの、本質は二者間でしかない。言い換えると相手がいなければ、そしていたとしても相手のことを知っていなければ、情報を出せない。どういう情報を出せばいいかがわからないからだ。この原始的なメンタルモデルを前章では「コミュニケーション1.0」と呼んだ。

次にドキュメンテーションとは、ドキュメントつまりは文書をつくることであり、本質としては「文書」なる成果物とその読み書きが発生する点にある。本章でもマネジメント 2.0（成果物のレビュー）を挙げたが、文書は典型的な成果物である。文書も同様に社外向けの正式なものだったり、内部メンバー向けのラフなものだったりと幅広いが、本質は変わらない。それでも甚大なコミュニケーションコストを減らせることが大きく、文書という形で情報を育てる視座を持てるようになる。読むことも、書くことも自分のペースで行えるため拘束も要らなくなる。したがって生活

水準も上がり、第二パラダイム「出社」から第三パラダイム「リモート」にシフトできるポテンシャルさえ持つ。デフォルト・リモートの要が情報共有であり、情報の養育であることは前章で述べた。ドキュメンテーションであれば可能である。一方で、文書をつくることは難しいし、非同期的に仕事が成立する塩梅で読み書きするのも難しい。だからこそマネジメントを行うのだが、強いマネジメントだとデフォルト・リモートは遠のく。それでマネジメント 3.0 との形でマネジメントレスを勧めてきたのが、ここまでの流れである。

つまりコミュニケーションは原始的であり誰でもできるが無駄が多く、ドキュメンテーションは技術的であり上手くやるためにはスキルとリテラシーを要する。もう少し、ちょうどいい感じの塩梅はないのだろうか――と考えて、第三の「やりとりのあり方」を模索することは自然だろう。幸いにもヒントは古くから存在する。それが発想法的な営みであり、これをあり方として捉えるためスプレディケーションと名付けた。

スプレディケーションとは

スプレディケーションとは発散、収束、蒸留から成る活動を指す。まず発散では情報を自由に出す。次収束では、発散は継続しつつも、発散した情報を整理する。具体的にはグルーピングしたり、関連をつなげたり、あまり価値がなさそうなゴミを隔離したりする。これで木と森を行き来しながら、死角を潰していく形となり、全体も細部も充実していく。次第に本質が見えてくるので、それを拾って言語化する。これを蒸留という。きりがないので、どこかで止めねばならない。蒸留まで済むのが理想だが、何らかの意思決定をして終了にする――と、小難しく書いているが、ブレストやひとりブレストで行っていることと同じだ。

要するに、ここでスプレディケーションと名付けた**ブレスト的な営みには、コミュニケーションやドキュメンテーションに並ぶ「あり方としてのポテンシャル」がある**。重用している組織も多数存在するのではないか。たとえば面白法人カヤックでは[ブレストに始まりブレストに終わる]と[]としている。雑談や会議やチャットでコミュニケーションしたり、Word や共同編集ドキュメントやウィキなどで文書をつくったりするのと同じように、日頃からスプレディケーションを行うのだ。

スプレディケーションであれば、コミュニケーションほどコストが高くつかない。各自が言いたいことや書きたいことを好き勝手に出せばいいし、相手のことを知る必要すらなく、すでに出された情報を見て、さらに思いつくことや足したいことを出せばいい。先ほどギビングを解説したが、いわば全員がギビングを行うことになる。誰が出した情報を誰が使ってもいいし、もちろんあなたが出した情報もいつ誰にどれだけ使われるかわからない。まさにブレストのように、各自が自由に発散すればいい。慣れるまではやりづらいだろうが、慣れたらこれほど楽なやりとりもない。

ただし、発散するだけでは収拾がつかないため、収束と蒸留も行う。この段階でファシリテーションが必要になる。慣れたらマネジメント 3.0 として示したギビングとリマインドで事足りるが、それまでは従来の強いファシリテーションがまずは必須だろう。たとえば従来の会議と同様、60分で開催して、進め方も定義する。最初の20分で発散、次の20分で収束、最後の10分で蒸留をして、10分ほどバッファ（Buffer, 予備時間のこと）を残そうといった具合である。特に重要なのが役割分担であるが、これは後の章で詳しく扱うこととする。軽く取り上げておくと、最低限、

意思決定を担う意思決定者は不可欠である。意思決定者は、発散した内容を根拠に意思決定を組み立てる。もちろんそのままでは理解しづらいので収束をするべきだし、できれば蒸留までしたい。もしそこまで関与できないのなら、意思決定者への提案までをスプレディケーションで行い、提案は別のやり方を使うのが良い。他の役割としては、いくつか例を挙げよう。

- 意思決定を行う意思決定者（Decision-Maker）
- 発散が得意なブレスター（Braster, Brain-storming + er）
- 収束が得意なサマライザー（Summarizer）
- 重箱の隅をつつくような細かい指摘が得意なスティッカー（Sticker）
- 出された情報の見栄えをきれいに整えるタイディスト（Tidist）
- 興味、実力、時間などがなくて傍観するだけだが、たまに出すかもしれないオブザーバー（Observer）
- 時間配分を知らせたり、重視したい話題を盛り上げるために皆を誘導したりといった旗振りを担うファシリテーター（Facilitator）
- ファシリテーターのうち、時間配分の制御だけ独立させたタイムキーパー（Time keeper）
- etc

役割分担に正解はない。また、固定的なものではなく状況に応じて使い分ければいいし、二役以上を兼ねてもいい。最も重要な原則として、スプレディケーションには **思っている以上に適性と制約が出る**。発散としてアイデアを色々出すのが得意な人もいれば、頭こそかたいものの、出ている情報を咀嚼して整理するのが得意な人もいるだろう。息するようにかんたんにできることもあれば、何をどう頑張ってもできないしやりたくもないこともある。状況によっては、疲れているから頭を働かせるのはできないがツッコミくらいはできるとか、逆に持て余しててじゃんじゃんひらめくのでたくさん出したいといったこともある。この手の営みをすでにこなしている人は薄々感じているだろうが、できる・できないは思っている以上に存在する。だからこそ尊重するべきである。向いていないことを無理にやらせるべきではないし、向いていることに専念してもらった方がいいはずだ。役割分担を工夫することで実現する。

役割分担にはもう一つ、大きなメリットがある。役割という形で言語化しておけば、生成 AI の利活用にもつなげやすい。特定の役割を自律的にこなす AI を AI エージェントと呼ぶが、AI エージェントをつくるためには、人間が言語的な指示を与える（与えておく）必要がある。プログラミングもそうだが、エスパーのように悟ってもらうことはできず、何をしてほしいかは人間が定義しなければならない。役割分担を活用できていれば、役割の言語化もできるはずで、したがってエージェント化にも繋げやすい。AI エージェントは機械だから、人間よりも酷使できる。あるいはエージェントとは言わずとも自動化もできる。たとえば単調なタイムキーパーを任せるくらいはできよう。

ついでに、もう一つだけ応用例を述べておこう。同期的なスプレディケーションのファシリテーションには、最適な型が多数存在すると考えられる。先ほどの例を抜粋するが、

> 最初の20分で発散、次の20分で収束、最後の10分で蒸留をして、10分ほどバッファ（Buffer, 予備時間のこと）を残そうといった具合である。

もちろんこの型は唯一ではない。20-20-10-B10 ではなく、20-20-20 もありえるし、40-10-5-B5 もありえる。個人やチームの数だけ最適解が存在するだろう。ということは、このようなファシリテーションの型もまた、必要に応じてつくって使えるようにできるといい。そのためには「ファシリテーションの設定」なる概念を扱えるようにすればいい。それも、人間がファシリテーションするのは無駄だから、AI エージェント（あるいは AI ではなくプログラムでも良い）に任せたい。高度な話になるため軽くに留めるが、このあたりを実現するには **Facilitation As Code（FACIaC）** に頼る。IT インフラを手作業ではなくプログラムコードでつくる営みを Infrastructure As Code、略して IaC と呼ぶが、これのファシリテーション版である。コードとしてファシリテーションの設定を書いておき、ファシリテーションプログラムが、そのコードのとおりにファシリテーションを行う。そうすると、私たちはコードとしてファシリテーションの設定を書き、貯めることで使い分けられるようになる。たとえば、スプレディケーションが不慣れな人向けの「ベーシック・ファシリテーション（4人以下、60分）」をつくって共有することもできる。

非同期的なスプレディケーション

コミュニケーションや情報共有に同期と非同期がある点は前章で述べた。同様に **スプレディケーションも非同期的に行うことが（も）できる。**

デフォルト・リモートでは同期（リアルタイムなやりとり）を極力排除すること、あるいは必要に応じた注入に留めることが肝心である。同期のかわりに非同期（各自のタイミングで情報を出す）への比重を増やさねばならないし、何なら非同期をメインの手段に据えねばならない。しかし、ここまで見てきたように、非同期は難しい。おさらいしておくと、まず非同期的なコミュニケーションとは、本人以外の人物から、本人が持つ情報を同期的に聞くことである。□さんと話すかわりに、□さんを知る□さんと話すわけだ。これは間接的なだけであって、非同期ですらない。次に非同期的な情報共有については、前章でも「情報の養育」として解説してきたが、概念的に新しいし確立された手段もまだない。そこでスプレディケーションである。

スプレディケーションは非同期的にも取り組みやすい。一番わかりやすいのは発散だろう。何らかのテーマでスプレディケーションが行われているとして、発散がしたいのなら、単にそこにアクセスして何かを書き込むだけである。すでに書かれている情報を無視して、テーマから思いつくことを殴り書くだけでもいいし、もちろんある程度読んだ上で思いついたことを書いたり、ある書き込みに対するツッコミやそこから連想することを書き足してもいい。いつ、誰が、どこに、どれだけ書こうが自由だ。コミュニケーションのように特定の誰かを相手にするわけではないので遠慮は要らないし、文書のように整理してつくる必要もないから気軽だ。まるでひとりで X(Twitter) にポストするかのような、あるいは生成 AI に話しかけるかのような気持ちで、雑に向き合えばいい。

別の言い方をすると、スプレディケーションは（特に発散は）雑にやりやすい。**私たちが非同期を苦手とする理由の一つは、雑に情報を出すことに慣れていないからだ。**コミュニケーションでは人を相手にするからそんなことできないし、文書では雑につくったところで誰にも理解できないし見てももらえなかったが、スプレディケーションは違う。そもそも発想法的な営みであり、雑であろうと情報を出すことが歓迎される。雑でもいいので、とりあえず出して、それらを見ながら、さらに膨らませたり揉んだりするのだ。創造的であり、遊戯的でもある。

日本人的な勤勉で慎重な価値観からすると何事かと思うかもしれないが、違う。私たちが情報を出し切れば、たいていの仕事はどうとでもなる。長らく続く課題は「情報の出し惜しみ」であり、コミュニケーションと文書では限界があるのだ。スプレディケーションなら出せる。出しやすいがゆえに、非同期的にも出しやすい。もちろん、すでに出されている情報も踏まえたい場合は、その場で読んで理解する労力が必要だが、こんなものは技術でカバーできる。それこそ生成 AI に要約してもらえばいい、その要約の出し方をシームレスにすればいい（シームレスについては後の章で取り上げる）。そもそも、スプレディケーションにより皆が情報をたくさん出す世界線になると、**純粋に面白くなる**。別に真面目な議論だけ扱わねばならないルールもない。前章で雑談ならぬ雑残を取り上げ、残すことによってでも「心地よい偶発性」を確保できると述べたが、まさにスプレディケーションで満たせる。それこそ「今日の昼ご飯は？」とのテーマで発散するだけでもいい。いろんな人の、いろんなメニューや食べ方が集まって純粋に面白いはずだ。面白いので見に来る。見に来るので書きたくなる。雑残でスプレディケーションの成功体験を覚えたら、次は真面目な話題でも使いたくなる。

先ほどから「雑」という言葉を使っているが、このテネットは重要なのでもう少し寄り道しよう。種類としては二つある。ラフとデサルトリーだ。**ラフ（Rough）とは、何らかのお題がある状態で、そのお題と雑に向き合うことを指す。一方、デサルトリー（Desultory）とは、お題がない状態で雑に向き合うことを指す**。お題が無いと動けない人のためにギビング（お題を与える）をすると先述したが、ギビングされなくてもとりあえず動いてみる、何かしら出してみるのである。私たちは思っている以上に、お題がないと動けない。ビジネスで言えば、課題や司令がないと動けない。筆者はこれをお題駆動（Issue Driven）と呼んでいるが、要は言われたことしかできない受動的な生き方が染み付いてしまっている。そうではなく、お題がなくても、相手がいなくても、文脈がわからなくても、とりあえず適当に決めて何かしら出してみるのである。

たとえば「当部門の既存事業が芳しくないので新規事業を考えたい」とする。部門には 20 人のメンバーがいる。これをスプレディケーションで始めようとすると、「新規事業を考える」のようなテーマで場を一つつくることになる。あとはメンバー各々が自由に発散していくだけ――なのだが、これだけではおそらく何も書けないと思う。一応、テーマはあるが、抽象的すぎて使い物にならない。ラフとして動くことはまだできない。それでも、何かしら出してみるのがデサルトリーだ。たとえば□さんが「最近 ChatGPT の月3万するプランを使い始めたんですよ」的な雑談を始めた（実際は書いて残すので雑残）とする。一方で□さんは「芳しくないってどういうこと？」との疑問を書いたとする。デサルトリーに出された情報が二つも出てきた。メンバー全員は、これらをお題とみなして、思いつくことを出せるかもしれない。既存事情と生成 AI をかけ合わせるとどうなるかなとか、芳しくなさの具体的情報を与えるくらいはできるはずだ。もちろん、これ以外の話題を提供してもいい。こういった雑な営みを、同期的に行うのは苦しいが、非同期ならできる。各自が空いたときにアクセスして読み書きすればいいからだ。一日だけでは大した成果は出ないが、これが積み重なると次第に情報が増えてくる。増えてきたら、次は収束をして全体像や傾向がわかるようになる。最終的には本質を蒸留できるだろう（もちろんできないこともあるが、こうして情報を貯めてきたことや整理してきたことは無駄にはならず私たちの血肉になる）。

デサルトリーに情報に出し、ラフに情報を出すことで広がりを持たせられる。スプレディケーション

ならまさにうってつけだし、非同期的に進めていくこともできるわけだ。そうして情報が充実すれば、**情報を** ベースとした議論や意思決定が可能となる。政治でも雰囲気でもなく、情報を拠り所にできる。無論、発散はともかく、収束や蒸留には高度なファシリテーションや意思決定の力が必要となるが、それはスプレディケーションに限った話ではない。それでも、その前段となる発散――情報を出す部分の敷居を圧倒的に下げられる点が強い。

スプレディケーションとトピック指向

長らくスプレディケーションの解説が続いたが、本節の主題はトピック指向であるため接続していこう。

おさらいすると、トピック指向とは話題（トピック）単位で情報をつくり、ある話題はその話題用の場所（ページや領域）に書け、とするものであった。トピックは部品でもあり、部品と部品を組み合わせて装置をつくるように、トピックとトピックを組み合わせて新たなトピックをつくっていくこともできる。また部品ごとに独立しているので、古典的なマネジメントに頼らずに進めやすい。そのかわり、各自がトピックを自発的につくる能力を要する。これをカバーするのがファシリテーションで、お題を与えるギビングと、必要に応じて用件を思い出させるリマインドを紹介した。

さて、スプレディケーションとの関係だが、トピックをつくりやすくなる。早い話、トピックの名前としてお題を設定して、そのお題に関する情報を発散していけばいい。何ならお題すら要らない（デサルトリー）。デサルトリーができずとも、すでにいろんなトピックが存在しているはずで、その中から必要なものを選んで書き込んでいけばいい。普段の議論や共有も、それ用のトピックをつくってここを見てくれ、とすればいいだけだ。

コミュニケーションでは人を探して、その人に出す。ドキュメンテーションでは文書をつくって、そこに書く。スプレディケーションでも同様に、対象を定めなくてはならない。私たちは人間なので、対象無しに動くことはできない。**対象が人である必要はないし、本書も「対象＝人」との固定観念を打ち破るつもりで書かれてはいるが、対象自体をなくすことはできないのである。**スプレディケーションであっても対象は要る。それがトピックだ。

すでに述べたとおり、トピック指向におけるトピックは自由度が高い。トピックは要するに「名前のついた箱」でしかなく、どんな名前をつけることもできるし、何を入れることもできる。小さな箱もつくれるし、大きな箱もつくれる。別に箱自体はどうつくっても良いのである。箱（トピック）という基本単位があることが大事なのだ。この単位が存在することで、私たちは箱に対して情報を入れる、という単純な動き方ができる。

別の言い方をすると、このトピックの概念を前にして、**ではトピックを育てていきましょうとなったときに、既存のメンタルモデルでは上手くいかない。**コミュニケーションのメンタルモデル（対象が人）でも、ドキュメンテーションのメンタルモデル（対象が文書）でもだ。どちらでもトピックなんてものを想定していないので、当然ながら上手くいくはずがない。メンタルモデルがまだできていないから動けない。だからスプレディケーションが要る。**人でも文書でもなく、変幻自在な箱（トピック）という概念を扱う** ためには、発想法的なアプローチが必要なのである。

難しいことではない。経験したことのある人も多いだろう。何なら日頃から使っている人もそれなりにいるのではなかろうか。一番有名なのはブレスト――ブレインストーミングだろうが、そうでなくとも、アイデアを出すために、思いつきを書いて、並べて、それらを眺めてうんうん唸るくらいは、おそらく一度くらいはしたことがあると思う。あるいは優秀な人であれば、脳内で無自覚にやっているかもしれない。日本で言えば人との会話、特に飲み会や対話会など、クローズドな少人数の場で、肩肘張らずにあれこれ考えながら喋る機会もあるはず。あるいはひとりで散歩しながら思索に耽ける人もいよう。いずれにせよ、そんなに突拍子なものではないと思う。筆者はただ、そのような営みに意図的に目をつけて、コミュニケーションでもドキュメンテーションでもない、第三のあり方として引っ張り上げただけだ。名前の収まりをよくするために、発散や広がりを意味するスプレッド（Spread）から取ってきて、スプレディケーションとの造語をつくっただけだ。何も難しくはない。自然な営みを、もっと重用してみようという、それだけの話である。

スプレディケーションツール

最後にスプレディケーションを行うためのツールを議論しよう。

まず始めに思い浮かべるのは、Miro を始めとしたホワイトボードだろう。Microsoft Whiteboard や Box Canvas など他のプラットフォーマーもサポートし始めていることからも、ホワイトボードの有用性は明らかだろう。付箋を貼り付ける形が多いが、付箋に限らずテキスト（単語や文章そのもの）、図表、動画やファイルなどを添付することもできる。またアプリに限らず、アナログに行うこともできるし、やったことある人も多いと思うが、ここではデジタルツールを扱いたい。

現状ホワイトボードをスプレディケーション用途で使うには、ややカ不足である。単一のテーマを短時間・短期間で集中的にやる程度だ。30 の議論を全部扱う、といったことはできまい。この手のアプリは「ボード」という単位を扱うことが多いが、1つのボードで使えるのは1つのテーマだけだ。仮に 30 の議論を扱いたいなら、これは 30 のテーマということであるから、30 のボードが要る。ボードという視座で行き来や連携を考えねばならないが、ボードでは粒度が大きすぎる。ましてテーマよりも小さな粒度も扱うトピックとなると、お手上げだ。たとえば 1000 のトピックを扱えるか、つまりは 1000 のボードを扱えるだろうか。不可能だ。ボードの視座を持つアプリは現存しない。筆者の知る限り、Postalkがスタートラインに立っているかも、という程度だ。Postalk ではボードとボードをリンクで繋いでボードのネットワークをつくれる。

現実的には一つのボードを広く使って、様々な議論を行うケースが多い。これは本質的には土地問題となる。XY座標な二次元空間の、どこにどんなテーマを置くか（建物を立てるか）にも等しい。現実世界でも都会と田舎があり、日本でも東京の 23 区があるように、二次元ボード上でも特に情報が集まる地域とそうでない地域が生じてしまう。土地というリソースを巡る戦い、つまりは政治の発生が不可避である。もっとも電子的な世界なので、SF のような工夫はできる。三次元になるが、マインクラフトというゲームでは、ネザーゲートと呼ばれるワープが使える。地上世界とネザーの二種類の世界があり、ネザーは地上よりも距離感が 1/8 になる。地上で 1000m 進むのは大変だが、ネザーゲート経由でネザーで移動すれば、1/8 の 125m で済む――とはいえ、焼け石に水であり土地問題からは逃げられない。

つまりホワイトボードを使ってスプレディケーションをやるなら、ボード単位での行き来と向き合うか、土地問題と向き合わねばならない。どちらもキラーとなるアプリや機能はまだない。

次に浮かべるのがノート（共同編集ドキュメント）だろうか。Microsoft Word Online、Googleドキュメント、Box Notes、あるいはベンチャー企業や個人向けならNotionが一番有名かもしれない。いずれにせよ、ページやドキュメントといった単位で「情報を出す場」をつくることができ、かつアクセスできる者なら誰でもいつでも読み書きできる。何なら同時にアクセスして同時編集もできる――と、**スプレディケーションを行うポテンシャルは十分にあるが、現状ではまだ実用には今ひとつだ。**というのも、利用者に要求されるリテラシーが厳しくて、スプレディケーションに集中しづらいためだ。

これらノートは、ホワイトボード的にたとえるなら「横幅が固定され、かつ情報の配置も行単位のみに縛られた不自由なホワイトボード」でしかなく、自由に書き込みづらい。実質的に言語情報を上手く構造化して整理しながら立ち回らないと、すぐに縦長のコンテンツとなって読むに耐えなくなる。そもそも無闇に縦長にすること自体が望ましくなく、こまめに別ページに分けるべきである。無論、分けただけでは後でアクセスしづらくなるから、ここまで述べたようにページ間の関係をつないでネットワーク構造をつくっていかないといけない。まさにトピック指向が収束や蒸留はまだしも、初期段階では気軽に発散できることが最大のウリなのに、トピック指向を意識しなければならなくなる。そんなことを意識しなくてもスプレディケーションできるよう、ツール側でお膳立てしてくれたらよいのだが、この概念自体が新しいものだし、ノートアプリらが備えているはずもない。唯一の例外は（ノートというよりWikiのカテゴリだが）Cosenseで、すでに述べたとおり、トピック指向用のツールとしては頭一つ飛び抜けている。スプレディケーションもやりやすく、十分にCosenseに慣れた者達を十分にオンボーディングした後で、少人数で取り組めば、現時点でも実現できるかもしれないとの感触がある。それでも一般向けにはまだまだ遠い。そもそもそれを行えるチームであるなら、スプレディケーションに頼らずともどうとでもできるだろう。再現性がない。万人が使えるようにするためには、お膳立てが――ツールからのファシリテーションがもっと要る。

ではチャットはどうか。Slack、Teams、Discord、あるいは国内ならChatworkを使っている人もいるだろう。LINEやX（Twitter）は対象外とする。なぜならビジネスに耐えうるポテンシャルを持たないからだ。耐えるためにはワークスペースやチャンネル、スレッドといった概念が要る。そのようなチャットツールをビジネスチャットという。ここではビジネスチャットを想定する。ビジネスチャットでスプレディケーションはできるか。

答えはノーだ。**チャットは本質的にコミュニケーションツールでしかなく、**コミュニケーションの下位互換または一時的な代替でしかない。ドキュメンテーションはもちろん、スプレディケーションもできやしない。それでも現代ではツールとして主流であり、さほどリテラシーを持たない人でも使いこなせることが多い。近年賑わせている生成AIも、ChatGPTを始めチャット的な見せ方で提供されることが多い。それだけ現代人にとって馴染み深いインタフェースなのだ。からくりは単純で、人を相手にするというコミュニケーション1.0のメンタルモデルでも親しみやすいことと、あとは単一の投稿口から投稿し、単一のタイムラインを見ればいいだけというわかりやすさがあるからだ。ホワイトボードやノートのように、どのボードやノートにアクセスすればいいかとか、アク

セスした後にどこに書き込めばいいかといったことを考える必要がない。それでもビジネスチャットなので、チャンネルやスレッドのレベルでは意識が必要だが、最悪しなくてもいい。読者の中にも、スレッドやチャンネルを守らずに投稿している人の例は一度は見たことあるだろうし、何なら経験もあるだろう。コミュニケーションなので、投稿さえすれば何とかなる。形式はさほど重要ではない。だからこそ、構造化できずに大量の情報や深い議論ができない。表面的で瞬発的なやりとりに終始するだけだ。こんなものでスプレディケーションをするなど、自殺行為にも等しい。

別の言い方をすると、チャットは時系列指向とでも言える。投稿したメッセージが時系列で表示され、今現在どんな話題が扱われているかの空気が常に存在する。コミュニケーションにおいて空気は重要だ。空気とは全く関係の無い話をいきなり出すことはできないし、出したところで混乱するだけだ。トピック指向的に言えば、**同時に扱えるトピックを一つのみに限定している**ようなものである。当然ながら、その単一の話題を誰がどう制御するかとの問題が常に発生する。強い統制が要る。権力が要る。権力があるから、権力者の言い分になる。それを巡る政治が生じる。チャットではコミュニケーションの構造的な限界さえも突破できやしない。

それでもチャットの、かんたんに使えるメリットは見逃せない。スプレディケーションツールに上手く組み込めれば、リテラシーの低い人達でもかんたんに使えるようにできるかもしれない。

と、ここまでホワイトボード、ノート、ビジネスチャットと見てきたが、これぞと言えるツールはまだない。

Chapter-5 マネジメント(2/2)

自律と他律、そしてスラック

本章はマネジメントの章だが、そもそもマネジメントとデフォルト・リモートは相性が悪い。そこでマネジメントパラダイムを比較した上で、脱管理――マネジメントレスを目指そう、そのためのやり方や考え方としてトピック指向を使おう、そしてファシリテーションとスプレディケーションをもって支援しようと述べてきた。

さて、マネジメントレスにするということは、各自が自律的に動かねばならないことを意味する。しかし誰もが自律できるとは限らないし、普段できる人でもできないことがある。自律のフォローも考えなくてはならない。

自律と他律

自律という言葉を使っていくため、本節での定義から始めたい。**自律とはベストエフォートが許された状況下で、自分ひとりでも「問題の無いペース」で行動することを指す**。ここで行

動とは、情報を出したり、タスクをこなしたりといったように他者が絡むものである。つまり問題が生じない程度に他者にパスをする、し続けるということをひとりで行えるのが自律だ。

当たり前の能力に聞こえるが、そうでもない。むしろ自律できる人の方が珍しい。社会人を含めても、自律できている人は 5% もいないだろう。もっと少ないかもしれない。個人的な話になるが、筆者は同僚からよく几帳面だとかレスが早いといわれる。象徴的だったのは「すべてのメッセージに答えてくれるので助かっている」との声だった。社会人なら当たり前ではないか、何を言っているんだとさえ思ったが、そうではなかった。それができるのは、単に私にある種の才能があって、かつ長年培ってきたタスク管理を始めとするスキルもあったからだ。当然ながら誰もができることではない。少し前にタスク管理の本を書いたが、思っていたより大作となってしまったし、これを書きながら、この内容を誰もが当たり前に知っていて実践できているかといわれると、そんなわけないよなぁとようやく腑に落ちたのである。この「世間とのズレ」は本書でも続いており、本書は「リモートワークくらい当たり前にできるでしょう」「むしろなんでできないのかが不思議で仕方がない」などと心底首を傾げる筆者が、腰を上げて言語化したものである。これくらいにしておくが、自慢がしたいのではない。むしろ辟易している。自律やリモートワークができなくても、泥臭く頑張って結果を出せる人達の方がよほど素敵ではないか。私はどちらも容易くできる力を持つかわりに、泥臭く行動して結果を出していく力がない。だからデフォルト・リモートを実現したくても、こうして読み物を書いて届けることしかできないである。そういうものだ。**両取りできない力というものがある**。

話を戻そう。自律できない人はどうすればいいか。他律するのである。**他律とは外部の干渉に頼ることで自律性の無さをカバーすることを指す**。人々が出社したがる理由の一つも他律にある。会社の自席につけば、同僚達がいて「場」ができる。自律する力がなくても、場に注目して、従っていけば何とかなる。会社に限らず学校もそうだし、集中するためにカフェや図書館に出かけるのもそうだし、何なら家族やシェアハウスなどで同居人がいることや、ペットがいることもそうである。

別の言い方をすると、他律とは自律性の無さを環境の力でカバーする（してもらう）ことである。環境には人やペットなど生物も含む。朝起きれない人が目覚まし時計をセットするのも他律だし、同居人に頼んで起こしてしまうのも、あるいはペットがうるさくて起きないといけなくなるのも他律だ。この例で自律的と言いたいなら、体内時計に従って何にも頼らず安定的に起きるくらいが必要だろう。あるいは、機械は非常に再現性が高いので、目覚まし時計だけで破綻無く続くなら自律的と言っていい。先述したリマインドも、まさに自律を促すものである。

自律の多様性

デフォルト・リモートにおいて重要なのは自律である。しかし他律的な人もいるため、カバーしなくてはならない。どうやるか。

結論を言うと、自律者（自律的な人）も、他律者（他律的な人）も、どちらも共存できねばならない。別の言い方をすると、自律者向けのマネジメントと、他律者向けのマネジメントを双方共存させる。これを **自律の多様性（Autonomy Diversity）** と呼ぶ。

全体像の章でも述べたが、多様性とは n 通りのあり方が共存できることだと思う。単にこのようなマイノリティがあります、と知ってもらうだけでは意味がない。腫れ物になって終わりだ。実際に共存するためには、たとえば問題なくチームで働くためには、マイノリティの事情も踏まえたやり方と考え方の議論にまで踏み込まねばならない。今の文脈で言えば、自律者と他律者が共存できている状態こそが多様である。自律者優先で他律者が苦しむ、あるいは他律者優先で自律者が苦しむのは多様ではない。やり方や考え方を知らないと、どちらか片方に寄せるしかなくて、寄せられなかった側のすべてが割りを食う。日本では平等が好まれるが、平等とは単一の強要にすぎず、多様ではない。秩序や効率のために単一に寄せたがる気持ちもわかるし、状況次第では必要でもあるが、常時使えるほどの汎用性はもはやない。どちらに寄せる、ではなくて、どちらも共存できることを目指すべきである。

そのために、以下の三点を踏まえていく。

- 1: 自律者と他律者それぞれの過ごし方を存分に確保する
- 2: 自律者をデフォルトのあり方にする
- 3: 自律者と他律者間の協調と連携を整備する

この三点は自律と他律に限らず、共存による多様性の実現全般に適用できる。多様性の構成要素をアイデンティティ（この場合は自律者と他律者の二つ）と呼ぶとして、1 はアイデンティティ各々の尊重、2 は総体として何を第一にするかの定義であり、3 は各アイデンティティを損ねないために協調と連携の二つのアプローチから調整することを示している。本質的には役割分担の話であり、後の章で詳しく扱う。

では自律と他律の場合を詳しく見ていこう。

1: 自律者と他律者の尊重

まずは自律者と他律者、双方のあり方をどちらも尊重する。

自律者は非同期的な情報共有で良い。すでに自律的に動けるはずだし、そのためのスキルやリテラシーや習慣がなかったとしても勉強・鍛錬をしていける。むしろ、自律的に動けるのに他律的なあり方を強要される方がストレスとなる。そういう意味で、**自律者のあり方とは、他律を課されずに済むあり方である** とも言える。たとえばチームで進捗確認のために毎日 10:00 から朝会をやるとして、これの強要も他律的だ。自律者にとって、毎日 10:00 に会議するなど不便な制約でしかない。そんなことをしなくても、主体的に問題のないペースで報連相くらいはできる。

しかし他律者はそうもいかない。自律ができないからこそ、朝会という儀式をつくって、そこに参加しさえすれば済むようにしなくてはならない。会議の場であればメンバーが集まっているし、有限時間内に決められたアジェンダをこなさねばとの雰囲気もあるから、自然と熱も入る。何なら会議中にメンバーが問いを出す（ギビング）ことで、その問いに応える形で情報を出せばいい。場があるとギビングしやすい。マネージャーが n 人のメンバーの進捗を順に聞いていく会議はあ

りがちだが、これはマネージャーが他律的なメンバーの世話を焼いているとも言えるし、他律的なマネージャーが自身の自律性の無さをカバーするための場をつくっているとも言える。そういうわけで、**他律者には己の自律性の無さをカバーするための場と役割が必要である。**

典型的には、自律者はフルリモート・フルフレックス（4）で済む。あるいは必要に応じてコミュニケーションも注入するが、いつ、どれだけ注入するかはその都度調整する（その結果として定例会議的に行うこともある）。一日一度も会議してません、一ヶ月一度も出社していません等も当たり前にように起きる。逆に他律者は従来の働き方と同様、それなりの拘束を伴う。自律性の度合いにもよるが、弱い場合はフル出社もありえる。ある程度強い場合は、週に一度くらい出社してコミュニケーションを重視するといったハイブリッドなスタイルで済むかもしれない。またハイブリッドであっても、週一で良い人もいれば週二、週三が欲しい人もいるし、普段は週一でもプロジェクトが忙しくて週四になることもある。だからといって週四を強要してはならない。自律性が比較的高い他律者であれば、週三以上は強いストレスになるかもしれない。

整理していこう。原則としてまとめると、次のようになる。

- 自律と他律のバランスは動的（Dynamic）である
- 同上、グラデーション（Gradation）でもある
- 加えて、それ以上は許容できず支障が生じるライン（Line または Limit）がある

バランスは動的なので変えていいし、グラデーションでもあるので同じ自律者でも程度は異なる。しかし守るべきラインを犯してはならない。この三原則を守れるように立ち回れたらいい。

自律と他律のバランスを論じるために、定義を追加したい。**時間または場所の拘束が伴う状態を他律的という。**出社、会議、イベント、その他通勤を含む移動はすべて他律的である。他律的でないことを自律的という。出社も会議もイベントも移動もなく、各自が自由に過ごせる状態と言い換えても良い。

これでバランスの指標を定義できる。**拘束率（Bind Rate）**とは、「仕事を行う可能性がある時間帯」における他律的状態の割合を指す。仮に 1 日 8 時間労働で、他律的な状態が 6 時間あるとしたら、拘束率は 75% だ。自律者は拘束率が低い方が望ましく、他律者は高い方が望ましい。また同じ自律者であっても、6.6%（30分/450分）と 23%（90分/390分）とでは全然違うことがわかるだろう。6% の人にとって、20% の拘束率は許容できないか、ストレスが有意に増えるラインかもしれない。逆に 20% の人にとって、6% の拘束率は少なすぎて、やはり支障が生じるかもしれない。

最適な拘束率は個人次第かつ状況次第であり、一律で決めるのは難しい。尊重にあたっては **自らソフトリミットとハードリミットを設定する** のが良い。ソフトリミットとは超えてもいいが警戒すべきライン、ハードリミットとは超えてはならないラインを意味する。たとえば自律者としてソフトリミットを 20%、ハードリミットを 50% に置いたとすると、基本的に 20% は超えないよう過ごすべきだがやむを得ない場合は超えてもいい（おそらくパフォーマンスや QoL は落ちる）。それでも 50% は基本的に超えてはならない、とする。イメージとしては、ソフトリミットは「普段は残業しないけど業務上必要になったので仕方なくやる」、ハードリミットは「休日出勤する」くらいで捉

えて構わない。いずれにせよ、最適なリミットは本人にしかわからないので、本人が決めるしかない。リミットが緩くてもいい、あるいは設定の必要性を感じない人は、おそらく他律的な人だと思う。適当に設定しておけばよい。極端にはソフトリミットを 100% としておけばいい――これは完全に他律的な状態であり、仕事時間≒拘束時間となる。現代の働き方はこの状態を求めがちだが、本書では見ての通り抵抗している。

話を戻そう。拘束率について議論したが、これだけでは平均的であり扱いづらいため、もう一つ、モードなるものを導入する。**バインドモード** とは拘束を積極的に使う時間帯や日を指す。**ミュートモード** とは拘束を使わない時間帯や日を指す。すでにミュートデイを述べたが、これは日中ミュートモードを適用することを意味する。同様の適用をバインドモードで行ったものが **バインドデイ** だ。モードとしては、一日（デイ）と半日（ハーフ）を使うといい。英語だとわかりづらいので、言い直して終日（デイ）と半日（ハーフ）と呼ぶことにしよう。つまり終日バインド、半日バインド、終日ミュート、半日ミュートが存在する。

このモードをどう使うかだが、**自律者はなるべくミュートモードを増やし、他律者はなるべくバインドモードを増やす** といい。というのも、拘束率だけでは以下の問題が生じるためだ。

- 拘束率は低いが、拘束のタイミングが細かく散らばっているせいで、自律的な状態でも集中しづらい
- 拘束率は高いが、拘束の時間が短い（自律的な状態になるタイミングが細かく散らばっている）せいで、いいところで中断になる

自律的な状態にせよ、他律的な状態にせよ、まとまった時間を取ることが重要である。 別の言い方をすると「30分以上のまとまった時間を取ることができません」制約を課されたとしたらどうか。集中できないし、切り替え（コンテキストスイッチング）にも疲弊する。そうではなく、3時間ひとりで集中できますとか、今日は終日出社してて積極的に話しかけていい・会議していい日です、とした方がやりやすいはずだ。その単位として、半日と終日が使いやすいのではと言っている。別に1時間でも30分でも良いが、切り替えを舐めてはいけない。個人的には最低でも 90 分は欲しいと思う。

最後に、もう一つだけ定義したい。自律者にとってのミュートモード以外の過ごし方、他律者にとってのバインドモード以外の過ごし方を **ウェルカム状態** と呼ぶ。ウェルカム状態のときは、基本的に自分の望む過ごし方をするが、他方の過ごし方を要求されたらその都度判断して調整する。自律者の場合、自律的に過ごしてはいるが、他律的な過ごし方も受け入れる余地をつくると言える。たとえば自律的に過ごしているときに、同僚からちょっと会話したいと言われたらその都度判断し、必要なら応じる（他律的な状態に移る）――と、当たり前に聞こえるかもしれないが、逆だとわかりづらいので逆の例も見よう。

他律者の場合は、他律的に過ごしてはいるが、自律的な過ごし方を受け入れる余地を持つ。たとえば、基本的に常に誰かとの会議をしている他律的なマネージャーがいるとして、ある会議でメンバーから「自律的な個人ワークが 30 分ほしい」といわれたとする。これは「自律的に過ごす時間を 30分 ほしい」という意味であり、もっと言えば「この 30 分は自由に過ごします」「そのかわりワークは自律的にやります」だ。マネージャーがこの提案を受け入れたとすると、マ

ネージャーも **会議中でありながら 30 分だけ自律的な状態に移る** ことになる。といっても、別の仕事をするのではない（してもいい）。マネージャー自身も、非同期的な情報共有をするのである。この会議に関するトピックがあるはずで、そこで追加の検討を行うなり、メンバーの検討の様子を読んだ上で助言や議論を考えておいたり、もちろんトピック上で非同期的にやりとりをしてもいい。ただし、自律的な時間であるため拘束してはならない。あるいは拘束しようとしても、メンバーはそれを無視できる（無理に無視する必要はない）。たとえばマネージャーがそのメンバーに声をかけても、メンバーは無視してもいい。なぜなら会話は拘束であるが、今現在はミュートモードであるため応える必要がないからだ（もちろん応えてもいい）。

いかがだろうか。拘束の考え方が染み付いている人には、ずいぶんと奇怪に映るかもしれないが、傲慢であると認識してほしい。自律者に歩み寄れていない。自律者に歩み寄るとは、ミュートモードを取り入れるということなのだ。逆の立場で考えてほしい。自律者に対して、ちょっと会話したくて声をかけたときに、「いや非同期的にトピックに書き込んでやりとりできますよね？」「会話する必要ないですよね？」と一蹴されたらどうか。これは自律者が他律者に歩み寄っていない構図であり、他律者のあなたが自律者に歩み寄っていないのと同じである。つまりウェルカム状態のときは、対抗するモードを受け入れる余地を持つ。その名のとおり、受け入れようとする姿勢を持つのである。

定義が続いてしまったが、これで揃った。まとめよう。

自律者と他律者の各々を尊重するために、以下の概念を導入・運用すればよい。

- 拘束率
  - 自律者は低く、他律者は高い
  - 同じ自律者（他律者）であっても、最適な率は違う
  - 自身でソフトリミットとハードリミットを設定し、各々のリミットを尊重する
- モード
  - 自律者はミュートモードを、他律者はバインドモードをなるべく使う
  - モードとしては終日と半日がある
  - まとまった時間を取ることが重要である。拘束率だけではまとまった時間を取れない可能性がある
- ウェルカム状態
  - モードでない時間帯も、基本的には自分に合った過ごし方（自律 or 他律）をすればいいが、他方からの過ごし方を提示されれば受け入れる余地を持つ
  - 受け入れるとは、自律者にとってはバインドモードに移ること、他律者にとってはミュートモードに移ることである
  - 前者はわかりやすいが、後者はわかりづらいので要注意

2: 自律者をデフォルトのあり方に

共存するための第一歩、各々の尊重について見てきた。続いて二点目を見ていく。

組織として機能するためには、ベースとなるテネットを一つ決めねばならない。私たちは人間で

あり、認知能力に限界があるので、組織として統制と効率を手に入れるためにはテネットを揃える必要がある。二つも三つも存在していては、思うように制御できない。だからこそ、歴史的にも、宗教だろうと会社だろうと国だろうと、組織は必ず単一のテネットをベースに置いてきたし、その座を巡る戦いが常に起きる。戦争もそうである。

デフォルト・リモートとしても、この理から逃れることはできない。そしてリモートを推すあり方なので、ベースもリモートの方に寄せることになる。自律と他律で言えば、**自律の方に寄せる**。デフォルト・リモートは、デフォルトで（基本的には）リモートであるべきとするものだが、そのためにはデフォルトで自律的でなくてはならない。すでに述べてきたように、リモートとマネジメントは相反するため、マネジメントレスに寄せなくてはならないが、マネジメントしないということは、各々で自律するということである。自律性は避けては通れない。

といっても、他律者をないがしろにするわけではなくて、前節で述べたように各々尊重するバランスは確保している。前章でもコミュニケーションを廃するのではなく、必要に応じて注入するように修正してきた。それでもベースのテネットをないがしろにしていい理由にはならない。他律者も、なるべく自律性を身につけるべきである。

そのためには **鍛錬（Training）を取り入れる**。鍛錬とは、スキルや素養を鍛えるために、地道で地味な練習をすることである。反復的な動作で筋肉を壊す筋トレ、タイピングゲームなどによるタイピングの練習などはわかりやすいが、同様の営みを、自律に関してもやる。自然に身につくものではないので、鍛錬の形で身につけてもらうしかないわけである。理想を言えば、鍛錬を意識させず楽しく身につけていければいいが、それは面白いゲームをつくることと同義であり、本書の範疇を超えるため割愛する。

鍛錬の話に戻そう。横暴で時代に逆行するイメージを持つかもしれないが、量と質次第だ。鍛錬という営み自体は、辛くはあるが有益である。組織パラダイムで言えば、ティール組織はオンボーディングの形で積極的に取り入れることが多い。ティール組織では自律的な小集団が自由に連携することでネットワーク構造を形成するが、ネットワーク全体の秩序を保つために、社内憲法のようなガイドラインを周知する。これを周知するために、オンボーディング時にしっかりと叩き込む。横文字的で目新しく映るが、要は研修である。研修というと新人に課すニュアンスが強いと思うが、そうではなく、社員全員に課すのである。あるいは一度課して終わりではなく定期的に課すし、何なら守れていないと思われる人にも課したりする。学校や軍隊での過ごし方に近い。もちろん安易に拘束と負荷を増やして詰め込めば良いというものでもない。

鍛錬を取り入れるためには、自律者が当たり前に行えていることを言語化・体系化した上で、研修や個別フォローとして提供する体制を整えねばならない。テネット、やり方と考え方、ソフトウェアなど道具の使い方まで、鍛錬内容は多岐に渡るはずだ。一週間くらい丸々費やしても不思議ではない。一方で、大企業でもなければ、長期的な集中鍛錬は難しいし、そもそも集中的な鍛錬はそれ自体が他律的なものであり、ともすると本末転倒となる。

つまり **デフォルト・リモートをなるべく損ねない形で自律性を手に入れるための鍛錬の体系** が求められる。本書として提案したくもあるが、長くなるため割愛する。いくつかヒントを挙げておきたい。

まずは自律的な鍛錬方法がほしい。鍛錬というと集合的に行うことが多いが、これ自体が他律的であり自律者にそぐわない。たとえば教材を動画で提供して、各自好きなタイミングで鍛錬できるようにするといい。自律者であれば、それだけでも必要な鍛錬を行える。他律者には厳しいだろう。自律的な鍛錬方法は、自律者に他律的な過ごし方を強要しないためのものである。

次に他律的な鍛錬方法は恒常的に確保したい。入社時に一度だけ、ではなく任意のタイミングで使えるようにしたい。もちろん鍛錬を受けるために申請だの費用だのといったハードルがあってはならない。全社員誰でもいつでも自由にいくらでも受けることができる。デフォルト自律は重要なのだから、デフォルト・リモートをやるのであれば、会社として投資しない理由はない。問題は鍛錬の頻度で、わかりやすいのは定期開催だが、これだけだとカバーしきれない。

そこで **鍛錬イベント** の開催を開放するといい。社員はいつでも誰でも「今から〜〜の鍛錬を 20 分くらいします」と鍛錬のための催しを開催できる。開催情報は全社員が見ることができ、誰でも参加離脱は自由に行える。たとえば Cosense や Notion などの同時編集を鍛えたいとして、「Cosense もくもく練習会」や「Notion で会話してみる会」を開催するといい。イベントは開かれているので、いつ誰が入ってくるかはわからないが、それでいい。最悪ひとりで鍛錬することになるが、開かれてはいるので単にひとりでやるよりも気が引き締まる。似た取り組みとして「もくもく会」があり、参考になるかもしれない(7)。

注意点としては、**あくまでも鍛錬のためのイベントであって、交流ではない** ことだ。自己紹介や雑談は不要どころか、省くべきである。基本的には「ひとりで鍛錬する人」が単に集まるだけの、まさにもくもく会的なスタイルが良い。そうではない協調的なイベントの場合、ファシリテーションを相当きつくしなければならない。おそらく前述した Facilitation As Code が必要になる。人間によるファシリテーターでは難しいだろう。

ここで「別にそこまでストイックでなくてもいいのでは」「それなりに楽しく鍛錬できたらそれでいいのでは」と思われるかもしれないが、良くない。スポーツやゲームなどで鍛錬する人はわかると思うが、鍛錬は短時間で負荷をかけるものだ。そうでなくては身につかないし、身につくにしても時間がかかりすぎる。研修のような強制集合型以外のやり方で、現実的にスキルや素養を上げるには、数十分以内の短時間で高付加なイベントを各自のペースで使う形にしなければならない。鍛錬イベントは娯楽的な部活動ではなく、社会人として仕事に従事するプロフェッショナルのトレーニングなのである。

いったんまとめよう。自律性を鍛錬するためのアプローチとして、以下二点を述べた。

- 1: 自律的な鍛錬方法
- 2: 他律的な鍛錬方法
  - オンボーディングや研修等、従来から知られる本格的なもの
  - 鍛錬イベント

これらを駆使すれば、自律者にも他律者にも尊重した鍛錬が可能となる。万人が自律性を鍛えていけるはずだ。

しかし適性の壁は無視できない。自律性にはある種の特性や才能が絡む。極端な話、ADHDの多動性など自律と相性の悪い特性があったり、逆にASDのこだわりの強さが自身のライフスタイルに向いた場合は、アスリート顔負けの自律的な生き方を難なくこなせてしまったりする（才能の域と言えよう）。どちらも発達障害のカテゴリーだが、白黒で論じれるほど単純ではないし、程度の差はあれど割と誰にでも当てはまる。当然ながら定量的に測るのも難しい。難しいが、難しいだけであって、**適性というものはたしかにある。できる人は容易くこなせるし、できない人はどれだけ頑張ってもできない。**

したがって、体育会的な「やればできる」「できるようになるまで頑張れ」は望ましくない。ベストエフォートでいい。特にデフォルトで自律的にしたいので、他律者たちがどれだけ自律性を身につけられるかが争点となるのだが、これも可能な範囲でいい。そもそもやる気が全く起きなくて鍛錬したくないならそれでもいい。そのような人に鍛錬を強要してはならない。ならないのだが、それでもデフォルトで自律ではあるので、なるべくやってもらいたくはある。

一般化すると、人材は以下の4つのタイプに分かれる。

- 1: 自律者
- 2: 他律もできないことはない自律者
- 3: 自律もできないことはない他律者
- 4: 他律者

1が最も自律的で、4が最も他律的である。1と4は直接関わるのは難しい。というのも、他律者は自律者が使う非同期的な情報共有を行えないからだ。使うスキルや言語能力があったとしても自律的に動けないので、仕事として成立するレベルでこなせない。逆に自律者は他律者が使う同期的なコミュニケーションを行えない。あるいは行いたくないのでやり方で揉める。そこで、どちらもできる2や3のタイプが仲介に入ることになる――と、要は**役割分担の話に帰着させて、上手く連携させればいいのである。**4の他律者に、1の自律者のようになれと強要したり、逆に自律者に他律者を強要したりするのではない。特にデフォルトで自律的だからといって、1を強要していいものではない（なるべく3や2になってほしくはある）。できないものはできないと認めて、許容して、その上で連携を考えるのである。

3: 自律者-他律者間の協調と連携

ここまでで自律者・他律者をそれぞれ尊重すること、しかし自律をデフォルトにすることを述べてきた。デフォルトつきの尊重を実現するためには、協調と連携が必要である。

協調とは、相手のやり方に歩み寄ることを指す。また連携とは、役割と仕組みで工夫することを指す。どちらも先述済だ。協調についてはウェルカム状態を取り上げたし、連携については、さきほどかんたんに4つのタイプを取り上げた。自律と他律、と単純に二分割するのではなく、グラデーションだと考えてもう少し幅を持たせるとやりやすい。たとえば4つにするだけでもだいぶ違う。両極端の1と4だけでは上手くいかなかったものが、間に2と3を設けることで連携しやすくなる。自律者と他律者の連携方法に正解はないが、参考までに一つのやり方を述べてお

く。

非同期的な情報共有を前提とした上で、以下のように分担する。

autohetero_coop

1の自律者は、各自で自律的に動いて情報共有をする。場には参加しない。

4の他律者は、従来の働き方と同様、出社や会議といった「場」に参加する形で、その場において情報を出す。場で完結しており、情報共有はしない。

これだけでは両者が歩み寄れないので残りでカバーする。

2の「他律もする自律者」は、基本的に自律者のように振る舞うが、必要に応じて場に参加することもできる。特に、場に参加しながら情報共有をする。たとえば会議に参加しながら、あるいは出社して雑談している最中でもリアルタイムに情報共有をする。

3の「自律もする他律者」は、基本的に他律者のように振る舞うが、必要に応じて自律的に情報共有もできる。ただし 2 と違って、他律がベースなので、場に参加している間はその場に集中する。情報共有は参加後、空いたときなどに行う。

このように分担すると、どのタイプも自分に合った過ごし方ができる。典型的には次のようになるだろう。

- 4の他律者は、作業やタスクといった「典型的な仕事」に集中する
- 1の自律者は、典型的な仕事よりも、改善・新規・整備といった「中長期的な取り組み」に集中する
- 2の他律もする自律者は、自律者から他律者への橋渡しを行う（他律者のために、自律者の情報を届ける）
- 3の自律もする他律者は、他律者から自律者への橋渡しを行う（自律者のために、他律者の情報を届ける）

現状では典型的な仕事をすることがまず絶対条件であり、加えて中長期的な取り組みも片手間でやらされるケースが多い。そうではなく、役割として分けてしまうわけだ。

典型的な仕事は、場に集まりたがる他律者の方が強い。ならば任せればよい。そのかわり、中長期的な取り組みという面倒なことはしなくていい。その分、典型的な仕事を頑張ってもらう。従来どおり会議に、コミュニケーションに、締切に、と疲れるだろうが、仕方がない。逆に自律者は、典型的な仕事が免除される分、中長期的な取り組みにフルコミットせねばならない。典型的な仕事の大変さはないが、かわりに正解がない中、調査や検討、言語化、それらを上手く見せたり整理したりといったことを続ける辛さがある。また出した情報へのフィードバックや議論も多いので、反論や批判に耐えるメンタルも求められる。そして、両者を繋ぐ存在として 2 と 3 を使う。もちろん 1 の自律者が他律的になってもいい（常態化すると 2 になる）し、4 の他律者が

自律的になってもいい（常態化すると3になる）。また他律者が中長期的な取り組みをしてもいいし、逆に自律者が典型的な仕事をしてもいい。

ただし、他律者は自律的には動けないので、「検討事項が30個あって、締切もないけど、自己管理して全部できるだけ検討しつくしてね」なんてことはしなくていいし、逆に自律者も他律的には動けないため、「毎週金曜日は出社日なのでよろしくね、また毎日朝会と夕会があるので参加してね」等には従わなくていい。それでも、非同期的な情報共有が前提であれば、必要な情報は出ているので問題はない。自律的に動けないからダメ、出社しないから・会議に参加しないからダメということではないのだ。そんなものはやり方にすぎず、自分に合うものを使えばいい（というより**合わないものは使わなくていい**）。そのための役割分担なのである。

恒常的な余裕

マネジメントするということは、マネジメントするための時間を捻出するとも言える。ここまでマネジメントレスのあり方を整備してきたが、捻出するべき時間がなくなるわけではない。というのも、マネジメントレスとはマネジメントの主体を自分自身にシフトさせただけにすぎないからだ。より正確に言えば**セルフマネジメント**にすぎない。マネジメント自体は消えていない。マネジメントするための時間の捻出も引き続き必要となる。

スラック

ここでスラックの概念を使う。スラックというと、いくつかの書籍が扱っているテーマでもある（5）が、ここでは別途定義したい。

スラック（Slack）とは、自由に過ごせる余裕時間を指す。できれば拘束が無いのが望ましいが、あっても構わない。たとえば出社している（オフィスという場所に拘束されている）状態であっても、1時間のスラックを取ることはできる。スラックの間は、いかなる割り込みも発生しないか、しても無視できる。もっと言えば仕事をしなくてもいい。リモートの合間にやるような家事やネットサーフィンや散歩も堂々とできるし、上司やメンバーへの作文も要らない。文字通りの余裕時間だ。資本主義的なテネットでは、空いた時間は少しでも回そうとしがちだが、スラックの考え方では許容しない。**従業員の権利として一定のスラックを保証し、かつ、差し込めるなら積極的に差し込んでもいいと考える**。先述のモードでいうと、バインドモードとミュートモードの他にもう一つ、スラックモードがあると考えればいい。昼休憩はよく知られたスラックモードだ。

スラックの量だが、昼休憩だけでは到底足りない。**昼休憩とは別に**、1日1〜2時間のスラックが欲しい。昼休憩の1時間も含めれば、1日2〜3時間のスラックが**業務時間の中に**存在することになる。これだけあれば、自律と他律のバランスは調整しやすくなる。昼休憩のように仕事をしない休憩時間として使ってもいいし、他律者が自律的な鍛錬に充ててもいいし、もちろんバインドモードやミュートモードをして仕事に励んでもいい。特に他律者はビジー――目の前の仕事で手一杯となってしまい、デフォルト・リモートの足を引っ張ってしまうため、スラックを入れてビジーから引き離してやらねばならない。もちろん仕事上、ビジーがやむを得ない場合や、自

ら望んでそうありたい場合はそうすればいいが、それは単に自分のスラックを使えばいいだけである。昼時が忙しい場合は、14 時くらいまでバリバリと仕事をして、その後で 1.5 時間のスラックを使って遅めのご飯を取りつつ、ゆっくり休むことも堂々とできるのだ。

ビジーは悪

スラックと聞くと、単に時間の一部を開放したように聞こえるが、少し違う。デフォルト・リモートに必要な諸活動のために、各自に使ってもらうことを期待している。デフォルト・リモートの肝はセルフマネジメント――自律的に情報共有を行うことであり、文字通りの IT リテラシーを用いた、読み書きの営みになる。それなりに疲れるし、切り替えや集中も要る。まとまった時間（モード）でなくてはならず、細切れの時間など論外だ。

また、モードであっても、ビジーであっては意味がない。ビジーな状態では非同期的な情報共有などできやしない。**ビジーは敵だ。切り離さなくてはならない**。だからスラックを使う。最悪仕事をしなくてもいい余裕時間、という概念を意図的に導入している。こうして半ば無理やりにしてもスラックを確保することで、ようやくスタートラインに立てる。

もう一度言う。非同期的な情報共有という、デフォルト・リモートにおける普段の文脈では**ビジーであること自体が間違っている**。ビジーに溺れてしまわないように、スラックを確保せねばならない。与えられたスラックを使ってビジーになるのは自由だが、ビジーだからセルフマネジメントできませんという言い訳は（前述の他律者タイプ以外には）通用しない。仮にデフォルト・リモート的なティール組織をつくったとすると、セルフマネジメントができない者は解雇し、セルフマネジメントの余地なくビジーなあり方を強要する顧客やパートナーとは距離を置かねばならない。それほどに重要である。現時点でできないならば、できるよう従業員を鍛錬したり、顧客やパートナーを説得、何なら対峙もする。

「それで仕事が成立するものか」と思う人も多いだろう。する。そもそも仕事に正解などなく、常に決めの問題でしかない。デフォルト・リモートが成立するように仕事をする、と決めればいいだけだ。これができないのは、単に前時代的なテネットにとらわれているからにすぎない。もちろん使い慣れてて成果も出ているテネットに依存するのはわかるし、短期的にはベターなのもわかるが、本書ではそれをよしとせず、第三のパラダイム「リモート」にシフトするべきと述べている。成立する・しないの問題ではなくて、**するようにしろ** と言っている。そのためのテネット、やり方や考え方、道具を本書にて解説している。

まだ足りないならもう一つ、筆者が **タートル・パラドックス** と呼ぶ現象を取り上げよう。ウサギとカメは、一見するとウサギが早いし、短期的にはその通りだが、中長期的にはカメの方が早く（長く）なる。カメは持続的だからだ。これを一般化すると、持続するための準備をしっかり行った上で進んでいった方が、すぐに飛び出す短絡的なやり方よりも進めると言える。プログラマーや作家は心当たりがあるだろう。馴染みがないなら旅行や引っ越しでもいい。設計、プロット、計画といったものをちゃんとつくった方が、最初は遅いが後々持続的に進み続けられるはずだ。もちろん当初のとおり進行することはなく、変更は入るが、それもキャッチアップできる。かつ、しっかり準備しておけば破綻せずに変化し続けられる。ビジネスでは品質とスピードはトレードオフとい

うが、違う。単に未熟なだけである。準備して、整備しながら進めば、中長期的に見れば品質もスピードも両立できる。

従来の働き方ではこれができない。本質的に搾取的だからだ。歴史を見てもわかるように格差が生じて緩やかに衰退するか、あるいは格差はあるが、上位側もがんじがらめとなり（恒常的にビジーで）身動きが取れず組織や社会の奴隷になる。少なくともボトルネックになる。デフォルト・リモートは、この前時代的なやり方に終止符を打つ一手でもある。ここまで見てきたとおり、実践していくのは並大抵ではないが、破綻なく持続していけるポテンシャルがある。もちろん、本書の内容が唯一の正解とは思わないし、そんなはずもないが、たたき台にはなろう。何ならハリセンである。前時代的な読者をぶっ叩いて、目を覚ましてもらおうとしている。いかがだろう、覚めただろうか。

もちろん、この主張は今すぐビジーを完全に捨てろと言っているわけではないし、ビジー自体が完全になくなるわけではない。どこかで誰かがやらねばならないときはあるし、やりたい人もいる。それは他律者として尊重すればいい。ただ、ビジー・デフォルトにしてはいけないし、今はなっているので解体するべきだと言っているだけである。

スラックの運用方法

業務時間中に、昼休憩も含めて 1 日 2〜3 時間のスラックが欲しいと述べたが、これは一例である。細かい運用方法は各自で調整すればいい。本項でもいくつか述べよう。

たとえば **スラックタイム** が使える。前章にてコミュニケーションタイムを取り上げた。「何らかのコミュニケーションを行うための時間」を設計して、必要に応じて注入するものだ。これと同じことをスラックでもやる。現状でも、昼休憩はすでにそうなっているだろう。昼休憩はおそらく 12:00〜13:00 の 1 時間だと思うが、これはスラックタイムとして 12:00 〜 13:00 を確保していることに等しい。

となると、スラックタイムの分は給料を出さなくてもいいか、と思いがちだが、そうではない。業務時間の一部ではあるのだから、出すべきだ。そういう意味では、**現状の昼休憩は（給料が出ていないという観点では）スラックではない**。もちろん、昼休憩も給料を出すからといって、今後はセルフマネジメントを頑張ってね、というのでは意味がない。単に従業員の負荷が上がっただけだ。そういうけちくさい搾取はやめて、やはり 2〜3 時間をドンと確保してほしい。とはいえ、労働基準法では所定の休憩時間を一斉に取ることが定められており、スラックタイムを休憩とみなすのは難しいだろう。仮にできたとしても、他律者は自分で休憩を差し込めず過労になりがちだ。スラックの他にレスト（Rest）を設けるといい。スラックは仕事も可能な余裕時間だが、**レストは仕事を禁止する余裕時間** である。他律者に休んでもらうための業務命令的な（他律的な）休憩と言える。オフィスでやるなら、たとえば 12:00 〜 13:00 をレストタイムとみなして、フロアへの立ち入りそのものを禁止するといったレベルの施策になる。効果は高いが、自分で休める自律者にとっては不便な制約となる。

スラックタイムとスラックモードの違いも取り上げておこう。スラックタイムは、スラックするための時間（時間帯）であり、会社やグループや個人が定義した上で、他者に参加してもらう形態を取

る。会社として 11:00 〜 14:00 をスラックタイムにしますと一律に決めることもできる（他律的であるためデフォルト・リモートとしては望ましくない）し、マネージャーがメンバー全員にこのリズムで取りましょうとしてもいい（これも同様に望ましくはない）し、個人がマイルールを運用してもいいし、そのマイルールをスケジューラーで公開して他の人も使えるようにしてもいい。いずれにせよ、スラックするための時間として言語化されており、他者も使えるようになっている。一方、スラックモードは、単に自分がスラックを過ごすだけだ。いつ、どれだけスラックを使うかは自分が適当に決めればいいし、それを公開する必要もない。メリデメを比較しよう。

- スラックタイム
  - ☐ 自分の状態を他者に伝えられる、自分の過ごし方をノウハウとして共有できる（Spend Knowledge）
  - ☐ 言語化や共有の手間がある、ともするとマネージャーや経営者が一律に課して他律的になりがち
- スラックモード
  - ☐ 言語化や共有の手間がない、他律に陥りにくい
  - ☐ 相手の状況がわかりづらく拘束しづらい、他律的な人は自分でモードをコントロールできない（ので働きすぎてしまう or サボりすぎてしまう）

もう一つ、テネットとして **ワーク・ライフ・スラック（Work-Life-Slack）** も取り上げておく。

ワークライフ系の用語としてワークライフバランス（ワークだけでなくライフも重視しよう）、ワークアズライフ（6 ワークとライフのバランスではなく報酬とストレスのバランスを主体的に設計する）、ワークインライフ（ワークはライフの一部でしかない）などがあるが、どれも余裕が欠けている。たとえばプライベートも忙しい人は、人生そのものが忙しい状態になる。これではデフォルト・リモートのために時間を使うこともできないし、そもそもテネットややり方や考え方と向き合うことすらできない。余裕を、物理的にある程度確保することがそもそも重要である。ワークとライフの二要素ではなく、ここにスラックという第三の要素も並べるのだ。仕事と私生活、このレイヤーにスラックが並ぶ。

スラックとはそれほどに重要な概念だし、この視座でスラックを採用することによって、人生の捉え方そのものが変わる。スラックを前提とした立ち回りになる（ならざるをえない）。まさにテネットが変わるわけだ。そういう意味では、組織でスラックを導入するのが難しい場合は、まずは個人的な人生に導入して練習してみてほしい。

まとめ

- そもそもリモートとマネジメントは相性が悪い、マネジメントレス（脱管理）を目指すべき
- マネジメント 1.0 から 3.0 へ
  - 1.0、やり方の管理
  - 2.0、成果物のレビュー

    - 3.0、状態のモニタリング（理想状態の維持）
- トピック指向により話題を部品化して扱っていく、積み上げていく
    - マネジメントの代わりに意思決定とファシリテーションを使う
    - コミュニケーション、ドキュメンテーションの他に、スプレディケーション（発想法的な営み）も使う
    - 現状キラーアプリがないのが課題
- 自律の多様性
    - マネジメントレスで自律的に動ける人ばかりではない人達も存在する（他律者）
    - 自律者も他律者も双方尊重した上で、上手くいく連携を考えねばならない
    - 自他のタイプを理解することと、連携するための役割分担を設けること等
- スラックの必要性
    - マネジメントレスとはセルフマネジメント（自己管理）であり、自己管理のために使うスラック（時間その他の余裕）が必要
    - ワークとライフの二軸で語られがちだが、ワーク・ライフ・スラックの三軸にしてもいいくらいに重要

Chapter-6 シームレス

シーム（Seam）とは「継ぎ目」を意味する。それがレスなのだから、シームレスとは継ぎ目が無いことを意味する。特に機能や体験が継ぎ目を感じさせず、滑らかなさまを指す。シームレスとは滑らかさだといってもいい。

デフォルト・リモートはシームレスとの戦いでもある。従来の、対面で非言語情報を使ったコミュニケーションは非常にシームレスであり、私たちはこれに慣れきっている。一方、リモートでは、かなりのシームがあってストレスとなる。特にオンライン会議やチャットがわかりやすいはずで、直接会って話した方がやりやすいのに、あーもうめんどくせえ、とコロナ禍の最中に歯がゆい思いをした人は少なくあるまい。シームレスという観点では、対面リアルタイムに敵う手段は存在しない。オンラインでのやりとりはどれも下位互換でしかない。

シームレスはコミュニケーション以外にも顔を出す。突然だが、あなたは普段仕事でどれだけのツールを使っているだろうか。メール、チャット、スケジュール管理、タスク管理の四点に絞っただけでも、複数のツールを使っているのではなかろうか。また扱うタスクや情報や人材の量次第では、一つのツールであっても文脈がその分増える。ブラウザのタブ、デスクトップアイコン、画面上のウィンドウが汚い机のように散らばっている人も少なくないと思う。そして、それらをあくせく切り替えている。切り替えが必要ということは、シームがある（シームレスでない）ということだ。当然ながら疲れるし、余裕もなくなる。仮にそんなシームフルな人々が多数を占める組織において、連携していくとなると、ご承知のとおり拘束に頼るしかない。打ち合わせを設定して、この時間内で何とか前に進めるぞ、と忙しなく課すのである。これではデフォルト・リモートなどできやしない。

そういうわけで、デフォルト・リモートがしたいなら、シームレスを確保していくことも重要となる。

シームレスの 3F

まずはじめに、シームレスを確保するためのアプローチを 3 つ紹介する。

タスクでも文脈でも頭の使い方でも道具でも何でもいいが、A → B → C と切り替えていくとする。シームは AB 間と BC 間に存在しており、ここが大きいほどシームフルと言える。

image_of_seam

このシームをできるだけ小さくしたいし、できればゼロにしたい。そのためのアプローチが 3 つある。

- Focus
  - 集中を最大化するために、シームをなくす or まったく別のあり方につくりなおす
- Flow
  - 一連の流れを滑らかにするために、シームを小さくする
- Feedback
  - 切り替えに必要な反応をできるだけ早くする
  - Flow の一種だが、重要なので切り離して解説する

3F という整理の仕方が気に入らないなら、単に **シームをなくすか小さくするかの二択である**と思えばいい。

seamless_approach

個別に詳しく見ていこう。デフォルト・リモートの実現にあたっては、これらをフル活用して、とにかくシームを少しでも潰していってほしい。

Focus ～シームをなくす～

Focus はシーム自体をなくすアプローチを取る。具体的には誰かに任せてやりとりのシーム自体をなくす「委譲」、人間ではなく機械にやらせることでシーム超えを機械に代行させる「自動化」、そして A → B → C を A → D に、あるいは A だけにしてしまうような抜本的な再構成を行う「変革」がある。

委譲

委譲とは、別の人に任せることだ。自分がBやCに切り替えるかわりに、誰かにAの状態を渡して任せる。その人がBやCに切り替えるなり同等のことをするなりして担えばいい。

たとえばA（仕事）からB（掃除）とC（料理）に移るとしよう。ひとりでやるとなると、仕事から掃除、掃除から料理、あるいは仕事 → 料理 → 掃除だったり、料理と掃除は並行して何度も切り替えるかもしれないが、ともかく自分が切り替えの負荷を負う。しかし家事代行に任せれば負わずに済む。かわりに、自分の文脈は伝えないといけない。この部屋で仕事をするから話しかけてこないでください、でも緊急時はノックをすれば可能ですとか、掃除はここからここまでをこんな感じでしてくださいとか、料理はこういうメニューを何日分つくって冷蔵庫に入れてくださいなど。

委譲のパラドックス

委譲とは権限委譲であると言っても良い。権限と役割を言語化し、役割に権限と人材を割り当てる。同じたとえを使うなら、次のようになる。

まず言語化された概念が各々ある。

- 役割: 掃除代行、家事代行
- 権限: 自宅への出入り、リビングへの出入り、リビング内の諸設備の使用
- 人材: □さん、□さん

次に、これを組み合わせて委譲をつくる。

- 役割: 掃除代行
  - 権限: リビングへの出入り
  - 人材: □さん

上記は掃除代行という役割に□さんを据え、リビングの出入りを与えている。委譲のつくりかたは無数にあるので、たとえば自宅への出入りも与えた役割をつくって、信頼できる□さんに与えることもできる。

言葉遊びに見えるかもしれないが、まさにこの言葉遊びをやるのである。**委譲とは言語化した部品とその構築に他ならない**。「自宅に他人を入れるのなぁ……」などと思考停止していては何もできやしない。一方で、厳密に定義するのは非現実的だし、できたとしても管理コストが肥大化してしまう。委譲したはずなのに、報連相という名の承認が発生してしまっている。この現象を **委譲のパラドックス** と呼びたい。まるでそんなものは存在しないかのように、正しく実現することがまるでできないし、通常はやろうともしない。代わりに信頼が使われる。長い時間をともに過ごすことで文脈を共有する。当然ながらデフォルト・リモートとして頼るものではない。

もし正しく委譲されているなら、少なくとも密な報連相は不要になるはずだ。あるいは行うにして

も非同期的な情報共有で良い。

パラドックスを打ち破る

パラドックスを打ち破るにはどうすればいいか。言語化と構築、その活用と議論を続けるしかない。前章のトピック指向はまさに役に立つし、いきなりまとまった情報を出せないのなら、まずはスプレディケーションを行うといい。当然ながらスラックが要る。毎日 30 分のスラックすら取れないほど忙しい状況に溺れるのではなく、30分、いや 1 時間でも 2 時間でも決して多くはないが、スラックを確保して、その時間を委譲の検討に使うのだ。

仮に仕事のマネージャー職が忙しいとするなら、なぜ忙しいか、具体的にどんな役割や権限から構成されているのかといったことを言語化して、じゃあそれはどうすれば任せられるようになるかを模索する。上手くいくかどうかはわからない委譲の案をトピックとしてつくって、これを任せてみたいんだけどどうだろうか、のような提案をする。トピックを通して議論を重ねていき、トピックの形で委譲の精度がつくりこまれていく。もちろん「そういうことをする時間がそもそもないんだよ」との問題（最初のハードル問題）も阻むが、そこは一部の仕事を諦めるなり、とりあえず誰かに雑に任せてみるなり、プライベートの時間を使うなり何かしら腹をくくるしかない。自律的に動けないなら、まずは他律的な場を設けてもいい。

いずれにしても、そうした活動を、情報共有として残すことが肝要である。残すからこそ、あとで読み返して続きにつなげられるし、他の人も読み書きできるので広がる。組織が、チームが、各個人が、自分の立場やステレオタイプにとらわれず委譲のための言語化と構築を楽しむようになるし、権力が偏っている部分も可視化されるので是正しやすくなる。もちろん抑止力にもなる。ここまでしないと委譲の考え方が文化的に広がらず、権威性と属人性の高い営みから脱せない。委譲というと、権力者が一方的に任せるか、一部の有能な者が器用に色んな人に任せるかを思い浮かべるかもしれないが、そうではない。万人が、もっとカジュアルに、誰かに任せること。それも同期的なコミュニケーションではなく、非同期的な情報共有によって行うこと。また、行い続けること。いわば **委譲の検証** を続けることで、委譲という営みそのものに慣れる——これがデフォルト・リモートにおける委譲の形だ。

センシティブの分離

委譲時によく出る問題が **センシティブ安全性** だ。プライバシーや機密情報が混ざっている（かもしれないから）から任せられません、だと「しない」に倒れてしまう。これでは何も任せられない。この問題をクリアするために、現代ではまだまだ古典的なやり方をしている。招き入れる人材を時間をかけて選ぶ——つまりセンシティブな情報を渡すに値するほど信頼できるのかを、時間をかけて探っていくわけである。これにはコストもかかるし、信頼できる人物を相性や好き嫌いの問題で落としてしまうといった機会損失も生じる。マッチングと言えば聞こえはいいが、単に他のやり方を知らないだけだ。

センシティブ安全性を高めるには、日頃からセンシティブな部分とそれ以外を分けるといい。これを **センシティブの分離** という。

単純な例として、自己紹介資料を考えてみよう。普段社内で使っている自己紹介資料を、そのまま社外の人に見せられるかどうかを考えてみてほしい。おそらく「できない」と思う。なぜなら社外秘が含まれている **かもしれない** からだ。一方で、分離できていると「できる」ことがある。

分離の発想で捉えると、自己紹介の内容は「社内社外問わず出せるもの」と「社外秘（社内にしか出せないもの）」の二種類に分けられる。これを普段から意識して盛り込む。パワポ資料であるなら、2枚構成にして、1枚目を前者のみ、2枚目を後者も含めて構成する。これなら社外向けには1枚目だけ使えばいいし、社内では2枚目もつければいい。手元のテンプレートには 2 パターン分持っておくか、2枚の方だけ持っておいて、社外向けのときは 2 枚目だけを抜けばいい。こういう発想を日頃からしておけば「2枚目を抜けばいいだけだな」と一瞬で判断できる。

通常はここまで考えない。センシティブかどうかを意識せず、とにかく情報を書き込んでいるはずだ。当然センシティブとそれ以外とが混ざり合っており、可能性としては「含むかもしれない」となるため、安全に倒して「出さない」になる。これでは委譲の文脈でも「渡せない」、つまりは任せられないとなってしまう。

いちいちそんなことを考えるだなんて不可能だ、と思われるかもしれないが、まったくそんなことはない。ソフトウェア開発では、すでにこの手のリテラシーは当たり前に要求される。詳しい解説は省くが、開発者なら .gitignore、envファイル、ハードコードや直接配置ではなく環境変数から読み込ませる、といったプラクティスはご存知のはずだ。そうでなくとも、私たちは相手次第で何の情報を出すかを変えている。仕事の悩みを会社の同僚に話すときと、部外者の友人やパートナーに話すときとでは違う。前者相手なら社外秘も喋れるが、後者には喋れないから、よりぼかした言い方をするだろう。さらに言えば仕事に限らず、子どもや学生であっても、□さんが話してくれた「□さん自身」のことを勝手に他者に喋ってはいけないことくらいはわかる（わからない人は信用を失う）――と、使い分ける能力と習慣は備わっている。

どちらかと言えば、情報共有など **残す情報 の方でも分離をはかるべし**、が本項の主張だ。「うちのチーム内で読み書きしてる情報なのでチームの外には出せません」ではなく、出せるものと出せないものを日頃から意識して、書き分けて、出せる方は日頃から開放するくらいのつもりで出せるようにしなさい、というわけだ。センシティブな情報はウイルスと考えてもいい。少しでも混ざっていたら、その病理は全体に広がってしまう。「ここらへんはクリーンですよ」と自信を持って示すには、日頃からウイルスが入らないように、あるいは扱う必要があるならそれ用の部屋を作ってそこから出さないようにしなければならない。日頃から取り組まねばならないのである。

そういう意味で、**委譲のしやすさを推し量りたければ、単に情報の透明性を見ればいい**。透明であるとは、それだけセンシティブの分離が進んでいることの証左でもある。分離しているからこそ公開できるのだし、分離を継続できているからこそ公開を運用し続けることができる。

最後に、もう一つだけ委譲の例を見ておきたい。

新規事業でつくったアイデアやプロトタイプを検証したい場合、社内に出して見てもらうのもいいが、社外にも出せた方が多様で有益であろう。しかし、安易にインターネットや SNS などで無闇に公開したところで相手にはしてもらえないし、センシティブ安全性も低い。まっとうな企業な

ら考えもしまい。だから検証コストが高くなる。社内で人材を集めるか、社外の場合でもちゃんとした業者や伝手を辿ることになる。もう少し楽にアウトソースできないものか。

アルバイトの一環として検証要員を集めるのはどうか。オンボーディングも含めて、1日4時間、週に3日ほど検証してもらうとしよう。時給は仮に 2500 円とする。検証ネタから機密情報を引けば、機密保持契約を結ばせるといった手間も減らせる。そして検証自体も、情報のやりとりだと考えれば、リモートで完結できる。ただし人材が他律的な場合は、パフォーマンスが出ないか、サボる可能性が高いため、オンライン会議の形で拘束するのが良い――と、細かいことはもっと詰めねばならないが、やろうと思えば可能だとわかるだろう。

強いて言えば、このような新しい試みを、会社の名前を背負って行いづらいという事情はある。子会社がまさにそうだが、本体とは切り離した独立的な組織をつくって、そこでやるのが鉄板だ。特に親会社側の制約がネックになっている場合に、その影響を受けない出島的な組織をつくることはイノベーション戦略として知られており、出島戦略と呼ばれる。

一番肝心なのは、一言で流したが以下の部分。

> 検証ネタから機密情報を引けば、機密保持契約を結ばせるといった手間も減らせる

センシティブの分離ができているならば、検証ネタから機密情報を取り除くことくらい造作もない。日頃から分離を意識して行動しているはずだからだ。たとえば事業アイデア自体が「概念的な部分」と「機密情報を含む文脈」に分けて構成されているはずで、アルバイトに渡す検証ネタは、単に前者をベースにして、ダミーの文脈をつければいい。そのネタはダミーを含むだけあって本物ではないが、本物でなくとも検証はできる。

自動化

委譲は人間にまかせていたが、自動化は機械に任せる。自分自身が A から B に切り替える（AB間のシームを越える）手間を負うかわりに、機械に B を任せる。もちろん機械が B を行うためのお膳立てと、機械そのもののメンテナンスなどは併せて必要だ。

自動化の例

たとえばあなたがマネージャーで、部下として□さん、□さん、□さんを抱えているとする。三人分の 1on1 を今日まとめてやるとしよう。1on1 では相手ごとの文脈を踏まえねばならないので、□さん用 → □さん用 → □さん用、のように切り替えが必要だ。何も準備しないと、おそらく、

- 13:00 - 13:30 □さんと1on1
- 13:30 - 14:00 □さんと1on1
- 14:00 - 14:40 □さんと1on1

のようなスケジュールになる。切り替えも大変で、13:30 からの□さんとの 1on1 の最中に、

□さんの頭から□さんの頭に頑張って切り替えることになるだろう。すぐに上手くいけばいいが、神経はすり減らすだろうし、手を抜けば切り替えが十分でないまま進むかもしれない（□さんにとっては良くない体験になるだろう）。

これの自動化を考えてみよう。やりようは無数にあるが、たとえば 1on1 の設定を行うツールをつくって、かつ部下がそこから予約する形にする。このツールでは、1on1 の時間は 1 時間で固定されており、内訳は 15 分の「休憩・準備」と 45 分の「1on1」となっている。スケジューラーへの登録時もこれを反映する――と、これなら自動的に休憩が 15 分入るので、切り替えが多少楽になる。

もっと極端な例だと、自分の情報を食わせた AI を用意して、部下にはこれと 1on1 をしてもらう。結果はすべて記録されるのであとで読めばいい。これなら自分が 1on1 の打ち合わせを行う必要すらなくなる（切り替えそのものがなくなる）。かわりに AI の準備や、AI と行った 1on1 結果の確認は必要だが、自分が打ち合わせをするよりはスケールしやすいだろう。たとえば部下が 10 人、20 人いても成立できる。

ずいぶんと冷たく聞こえるかもしれないが、実はデフォルト・リモートにおける 1on1 としてアリだ。すでに現実的に実現もできる。チャットボットをつくればいい。たとえば内蔵プロンプト（指示）として、以下を組み込んだボットをつくればいい。ChatGPT はまさにそのためのサービス GPTs を出している。

あなたはマネージャーであり、部下の□さんと 1on1 をします。
□さんからの質問や会話に応えなさい。
あなた自身の文脈と見解は、以下「Context」に記載されています。これに基づいて答えなさい。

```
# Context

## 想定Q&A
- Q: ～～ですか？
    - Ans: ～～です。
- Q: ～～には？
    - Ans: ～～が良いと思います。
- ……

## 仕事に関する助言や心配
- ……
- ……

## □さんに言いたいこと
- ……
- ……

## 自分の直近の過ごし方
私の今週の過ごし方を、日記として振り返っていきたい。今週と言えば、まずは……

# 以下は□さんからの質問
```

そうすると、チャットボットは、この内蔵プロンプトの内容に従って回答を出してくれる。擬似的に上司と 1on1 しているに等しい。テキストでのやりとりが面倒なら音声でやりとりすればいいし、内蔵プロンプトをつくるのが手間なら、日頃読み書きを行っているツールを監視して情報を取得させてもいい（センシティブ安全性は下がる）。あるいは、他律的に 1on1 チャットボット用プロンプトをつくるイベントを開催するのもアリだろう。具体的な実装方法は割愛するが、この程度であれば現状でもつくれるはずだ。

もちろん、これは情報のやりとりとして 1on1 を捉えたものであり、コミュニケーションとしては満たされない。コミュニケーションを満たしたいなら別の仕組みを考えねばならない。といっても、コミュニケーション自体は当人同士で集まらないとできないだろうから、会議の設定や当日何やるかなどをどう自動化するかの話が主軸となるだろう。

自動化の肝は DX

委譲は人に任せるものであるため、人の扱い方に慣れている方が有利だ。同様に自動化は

機械、特にITに任せるものであるため、ITの扱いに慣れた方が有利である。

ITの力はもはや疑うまでもない。本書でも何度も言及した生成AIがまさにそうだし、GAFAMの天下も言わずと知れている。この恩恵にあやかるべしと説くのが、いわゆるDX――デジタル・トランスフォーメーションだ。直訳するとデジタル変革であり、ビジネスモデルのレベルで「ITありき」のあり方にシフトすることを指す（1）。従来であれば、社員ひとりひとりにそれなりのPCと月額1000円以上するSaaSを複数本契約することはしづらいだろうが、DXだと当たり前にやる。必要なことだからだ。かわりに、オフィスへの投資は控えめかもしれない。あるいは、東京に大きな本社を置くのではなく、全国各地に小さな集合場所をつくるスタイルになるかもしれない。シェアハウス、シェアオフィスとシェアの流れも来ているし、シェアリング・エコノミーなんて言葉もある。オフィスのあり方そのものも過渡期であるように思う。と、このように、ITありきで考えると、色んな前提が変わりうるし、実際変わっている。この流れについていくためには、ITの思考回路にならないといけない。

DX、自動化、そしてデフォルト・リモートは、すべてつながっている。どれもITありきだからだ。ITを重用するからこそできることを定義しているだけである。とはいえ、具体的に何が要求されるかの細部は微妙に異なる。DXで求められるのは「ビジネスと会社に詳しい人」によるリスキリングだ。リスキリングとはRe + Skill + ingであり、新たなスキルを身につけることを指す。この場合はもちろんITだ。すでにITに詳しい人が、ビジネスのことや会社自体のことを知るのは難しい。それよりも、すでにそれらを知っている人がITを学んだ方が近道なのである。次にデフォルト・リモートで求められるのは、すでに述べたとおり文字通りのITリテラシーだ。

最後に、本題の自動化に求められるのは、ITという手段に関する知識と経験である。先ほど1on1チャットボットの話をしたが、生成AIの扱いに慣れた者なら、これくらいはすぐに思いつくし、つくれるだろう。もっと上手いやり方も検討できるはずだ。なぜかというと、生成AIと呼ばれる技術の、実際の知識や経験があるからだ。大体どういう概念があって、どう組み合わせるとこういうことができて、その概念を実現するためのツールやサービスは大体こんな感じで、使い方はこんな感じで……のようなことが知識や経験として蓄積されており、これらをベースにした、現実的な検討を行うことができる。

ただし、手段を知っているのと実際に使うのとには大きな差がある。私たちが従来のテネットによって思考も行動も縛られていることは、すでに散々見てきたとおりだ。本書はデフォルト・リモートという形で、従来のテネットに気づいてもらい、何なら抗ってもらうことも期待している。この作者は一体何を馬鹿げたことを言っているのだ、と思わず、自分を疑うつもりで向き合ってみてほしい。

すでに蓄積している人であれば、まるで霧が晴れたかのように、自動化のアイデアが思い浮かぶようになると思う。思い浮かばない人は、単に動機が薄いのでなければ、手段の理解が浅い。ITをもっと使ってみるか、勉強してみよう。スマホアプリやオンラインゲームも含めれば一生使いきれないほどのボリュームがあるし、理論面でも同様に人の一生では追いきれないほどの厚みがある。退屈することはない（数学と同じで合わない人にはとことん合わないが）。自分なりに楽しむだけでも、長い目で見れば変わってくる。

変革

変革とは、今現在採用しているテネット、やり方や考え方、道具などを抜本的に変えるアプローチである。変革により、従来 A → B → C だったものが A → C になったり、A → D になったり、あるいは A になったり D になったりする。いずれにせよ、従来存在していた二つのシーム（AB 間と BC 間）がなくなるわけだ。

先述の1on1 チャットボット は、変革的でもある。マネージャー自らが打ち合わせの形で 1on1 するという常識を崩し、1on1 とはマネージャーがつくった情報源と壁打ちすることである、としたわけだ。これにより打ち合わせが不要となり、部下は自分のペースで壁打ちできるし、マネージャー側も忙殺されなくなる。コミュニケーションは充足できないだろうが、情報のやりとりはできる。

やってみる

変革において、よく言われることは「常識を疑え」だが、このアドバイスには情報量が無く、実質何も言っていないに等しい。実際に行動を始めるためには、もう少し踏み込まねばならない。

変革には失敗がつきものである。1on1 チャットボットも、実際に有益かどうかはやってみないとわからない。やってみればいいのである。やってみるためには、ある程度の余裕つまりはスラックが必要だ。スラックがないとやってみるどころか、考えてみることすらできない。具体的には、変革のためにたっぷり費やせる役割と、その役割がつくった変革の案をみんなで試したり議論したりする余裕、の二つが要る。両者とも確保しないのは論外だし、前者があっても後者がないと相手にされなくてやはり意味がないし、後者だけでは変革と言えるほどのアイデアが出てこない（改善くらいなら可能）。

そして、やってみるためには、本章で述べてきたシームレスがまさに役に立つ。変革は創造的かつ破壊的なものであり、私たちが普段とらわれている煩雑な制約のもとではまともに動けやしない。シームをできるだけ取っ払って、真にゼロベースで本質を考え抜かねばならない。変革者が抱える従来の仕事は他の人に任せるべきだし、自動化によりシームレスの底上げも図るべきだ。たとえば、一切の仕事を免除して変革に専任できる者をひとり捻出すること、また、その者が出した変革案を揉む時間を全員に週に 2 時間以上（たとえばレビューに 1 時間、事前の勉強やキャッチアップに 1 時間）確保してもらうこと――これくらいはやらねばならない。できる、できないではなく、できればスタートラインに立てるという、ただそれだけの話だ。この程度ができないと変革は一生成せないし、実際成せていない組織も多いだろう。当たり前である。舐め過ぎだ。片手間で傷無しに行えるものではない。

変革には投資が不可欠

変革には経営層レベルでの投資が不可欠である。

よく言われるのがボトムアップ――現場からの草の根的な取り組みが拡大して全社に影響を及

ぼすようになったというストーリーだが、そんなものは結果論であり、投資に対する怠惰を正当化しているにすぎない。

変革を起こすためには、トップがこうすると腹をくくって無理やりにでも推進するか、あるいはリソースや権限を然るべき組織に渡して任せるかしかない。そうしてスタートラインに立った上で、それでも成功するかどうかがわからないものなのである。何事も動き出してみることで色々わかってくるが、変革も同じだ。変革とは、Before/After が激しくて規模も大きな改善にすぎない。改善の本質は同じで、必要な前提を揃えて挑戦する、し続ける、食らいつく。それ以外に無い。

投資できない経営者は（変革という意味では）無能である。投資しなければならない。そして、そのためには投資に踏み切れるほどの情報が必要であり、つまりは情報を集めたり考えたり議論したりするための仕組みとスラックが必要である。経営者だからこそ、半日くらいはスラックが欲しい。

より正確に言えば、変革は経営者が行うことではない。単純な話、**経営者には向いていない**のである。経営者の役割は（プレイヤーとしての役割を手放す規模を想定するが）組織を持続させることである。そのためにベター（無難）な選択肢を取り続けることである。必要な仕事は、いくら忙しくてもすべて（意思決定して捨てる・スルーするのも含めて）捌かねばならないし、不正解を踏まないよう守らねばならない制約も多い。典型的には組織の顔として、代表者や番人としてあり続ける。

たとえばビッグテックの例だが、行き過ぎた人材を解雇するために、リモートから出社に戻します、なぜなら出社の方が生産性が出るからです、のようなことを綺麗に並べ立てていたりする。この行いは、単にリストラしますだと心象が悪いから見せ方を良くしているのであって、本質的にはリストラだ。制約も多い大きな会社の経営者は、会社を守り続けるために、こういう政治的な戦いに浸かっている。規模は違えど経営とは、組織の運営とは、そういうものであろう。経営でピンと来ないのであれば、フィクションに出てくる為政者の描写を見てもいい。過激な言い方をすると、経営者もまた奴隷だ。常人に務まるものではないのでニンジンが要る。通常は高い報酬、権力、ステータスといった外発的なものに頼る。

一方、変革とは、制約にとらわれず、より良いあり方をゼロベースで考えることである。一部の才人や狂人は除くが、経営とは世界観が違いすぎて向いているわけがないし、実際できないからやろうともしない。文化的にトップが独裁しづらく、内部でも大量の政治を要する日本ではなおのこと難しい。そもそも忙しすぎてそんなリソースがない。できるわけがない。

CXO

変革は、それを専任する役割を設けた方がいい。

現代ではこのような概念はまだないが、変革担当部門の責任者ということで Chief Transformation Officer、CXO とでも名付けよう。小文字だと「Chief なんとか Officer の総称」になるから大文字だ。CXO は意思決定権は持たせないが、どう変革するかは全面的に

担う（もちろん経営者自身も議論に参加してよい）。検討に必要な権限もすべて与える。おそらく個人では厳しくて、少人数か、もう少し大きな集団になるだろう。そういう意味ではXPO（Transformation Program Office）と呼んだ方がいい。社内外ではCEOが会社の代表として立ち回るが、同様に社内ではCXOが立ち回ることになる。ともかく、できるかどうかもわからない変革のためだけにフルコミットする、創造的な探求者集団である。ちなみに同等の議論は多数存在する（2）が、従来の組織構造ベースで肩書だけ与えても再現性は薄い。詳細は次の役割分担の章で述べるが、変革のための構造を別途つくらねばならない。

経営者の仕事は、CXOの意見を踏まえて、実行に移すかどうかを下すだけだ。もし下せるほどキャッチアップできないなら、実行の権限さえもCXO/XPOに渡さねばならない。それが嫌なら、キャッチアップしなければならない。キャッチアップ or デリゲーション。本当に変革がしたいなら、どちらかを選ばねばならない。

実行にあたっても、別に一気に全社に適用することだけが唯一ではない。特定の人や部門にだけ適用してみて様子を見る（パイロット）、変革後の理が動いている場所をつくって誰でも遊びに来れるようにする（プレイグラウンド）、あるいは過激だが手持ち無沙汰な社員をパイロットに引き込む号令を出して、引き込まれたくなければ従来の仕事をもっと頑張れよとプレッシャーをかけてもいい（プレシャライズ）。やり方は無数にあるが、本質は役割分担だ。CXO/XPOは変革専任者または集団だが、つくった変革の案を試す役割というのも要る。この役割の確保の仕方としてパイロット、プレイグラウンド、プレシャライズがあるわけだ。もちろん他のやり方を考えてもいいし、既存の変革まわりの事例や理論も参考にできるだろう。

一つ注意したいのが、**言い出しっぺの法則に陥ってはならない**という点である。従来では提案者が責任を持ってある程度の推進まで担うべし、とされている。起業家はまさにそうだし、社内で新規事業を立ち上げる担当者も企画からその実行までトータルに立ち回らねばならないし、それができない者は能無しとみなされる。また熱いビジョンを持ちつつも、リソースをいかに上手く使うかというコントロールしてプロジェクトを回す「プロデューサー」も昔からよく知られたポジションだ。あるいは特許が要求する「発明」の要件も、技術的なつくりかたまで要求しているし、組織で特許を出すとなればある程度の政治も発生するわけで、事実上クリアする要領も要求される。

いずれにせよ、発案者が推進まで行う「言い出しっぺの法則」が前提となっているが、**変革はそうじゃない**。むしろ推進は別の人に任せて、CXO/XPOはその前段の検討、調査、議論に集中するべきだ。もちろん、言い出しっぺの法則でできるならそれに越したことはないが、そんなものは一握りのスーパーマンにしかできない。もっと成功率を高めるには、アイデアと推進を分離した方がいい。無論、そうなるとアイデア役と推進役の間にシームがあるから、これらは取り除く。まさにシームレスの出番だ。

イノベーション（変革と同義だと考えていい）は遊びから生まれるというが、なぜかというと、単に推進から切り離して追求できているからではないか――筆者はそう考える。ならば、いかに推進から切り離して独立させるか、でも推進につなげるための連携もカバーするかを考えればいい。本書では、このような分離の思想を何度か強調したきたが、まさにそのとおりだ。そういうわけで、変革とデフォルト・リモートは相性がいい。というより、デフォルト・リモートは変革の一種とも

言える。

(余談) 変革とイノベーション

変革とイノベーションの違いに確たる正解は無いだろうが、筆者の理解を述べておく。目的の有無だ。

変革は「リモートをデフォルトにしたい」など目的があって、そのために何ができるかを詰める。本書でもデフォルト・リモートとして様々なやり方や考え方を導入してきた。一方、イノベーションとは「与える影響の大きい」「何か新しいこと」であり、それ自体が何らかの直接的な目的で駆動しているわけではない。影響と新しさは市場は決める。仮に本書のデフォルト・リモートの理論が AI エージェントの発展に多大な貢献できるとして、それで実際貢献してエージェント開発が進み、私たちの生活が便利になったとすれば、デフォルト・リモートはイノベーションと言えるだろう。別の言い方をすれば、イノベーションとはブレイクスルーを起こした「新しいもの」を指す。

イノベーションは市場次第であるため再現性がない。一方、変革はゴールがあるので、勝つまで続ければ多少は成功率が上がる。ただ無限に推進することはできないし、リソースが持ったとしても現場が疲弊してしまうから現実的じゃない。アイデアと推進を分けて、アイデアの部分でとことん詰めるのが良い。それを言い出しっぺの法則から脱せよという意味である。このような取り組みは前例がなさそうなので、新たに整備せねばなるまい。筆者はセパレータブル・イノベーション（Separatable Innovation）と呼んでみている。アイデアと推進を分けることからそう名付けたが、次の研究テーマになるかもしれない。デフォルト・リモートで整備してきた諸々も役に立つと見ている。いや、その前にデフォルト・リモートの検証が先か。

(余談) 分離エンジニアリング

コミュニケーションと情報共有、情報の養育と養育者のメンテナンス、センシティブ情報とそれ以外など、従来混ざっていた AB を A と B に分ける、との考え方を本書では何度か強調した。分ければ分かる、とは誰の言葉だったか。そのとおりである。デフォルト・リモートでも、この考え方は本質と言えるほどよく登場する（この後も登場します）。

最近思うのは、このような営み自体を工学（エンジニアリング）にできるのではないか、ということだ。今のところ分離エンジニアリング（Separation Engineering）と名付けている。AB を A と B に分けることを扱った理論であり、その名のとおり工学なので実践的なアプローチと知見を重視する。内容としては分けることのメリットと分け方、そして分けることを前提とした思考回路をつくるための前提の整理、などが含まれるか。できればソフトウェアなど道具もつくりたい――。いかがだろうか。これも次の研究テーマになるかもしれない。

Flow ～シームを小さくする～

Flow はシームを小さくするアプローチだ。自分が強くなることでシームの影響を減らす「鍛錬」、仕組みの方を改善することでシームを小さくする「効率化」、あとは仕組みよりも道具がわかりやすいが、使い分けという意味での切り替えに要するシームを日頃から小さくする――特に大きくなるのを防ぐ「保全」がある。

鍛錬

鍛錬（Training）とは、その名のとおり鍛えて自分が強くなることで、シームの影響を減らすものである。

本書で強調する「文字通りの IT リテラシー」は、まさに鍛錬の対象となる。書き言葉への言語化、タイピングの早さ、コミュニケーションツールや情報共有ツールの操作、それらツールの大体の基礎理論と技術的限界に対する理解、箇条書きや Markdown といったフォーマットの使いこなし etc――習熟すればするほど有利になる。一つの目安として、デフォルト・リモートをここまで見聞きしてきて、納得はともかく「大体わかった」とすんなり理解できた人は、おそらく十分鍛錬を積まれていると思う。そうではない人は、本書の長文と向き合うモチベーションと筆者の国語力を棚に上げれば、おそらく鍛錬が足りない。足りないから、至るところに存在するシームにもいちいちつまづいてしまうし、それを肌で感じてしまってやる気さえも失せてしまう。逆を言えば、鍛錬して強くなればシームの影響も減って、デフォルト・リモートもより現実的かつ有用なものとして感じられるようになるだろう。

鍛錬に関する詳細は割愛させていただくが、前章ではマネジメントの観点で自律的（手段は用意するけど勝手に各自で鍛えてね）・他律的（一緒に鍛えるための場をつくるので参加してね or 参加させるね）の二つを取り上げた。本当はスポーツやゲームのように、具体的なメニューと理論にまで踏み込んでいきたいが、本書の範囲を超えるし、筆者としても準備が足りないので今回は割愛したい。必要なことではあるので、課題なのは間違いない。

効率化

己を鍛えることでシームを越えやすくするのが鍛錬であったが、逆にシーム自体を小さくすることで、己を変えることなく越えやすくするのが効率化である。通常、効率化というと鍛錬の側面も含むが、ここではあえて分けている。効率化とはシーム自体を小さくすることであり、これはもっと言えば仕組みに働きかけることを意味する。

仕組みに手を入れる

たとえば普段 10 秒かかっていたことを 5 秒にする。鍛錬した人が 5 秒でできるようになりました、ではなく、仕組みの方を改善して誰がやっても 5 秒で済むようにしたいのである。効率化の文脈では個人の努力に注目が集まりがちだが、それは鍛錬のカテゴリーにするべきだ。効率化は違う。むしろ個人の努力なしに、いかにシームを小さくするかが肝心なのである。個人の努

力に帰着させてしまうのはむしろ怠惰であり、放棄であり、害悪ですらある。問題を履き違えてはならない。効率化とは、仕組みを改善することなのだ。**仕組みに手を入れねばならない**。先述した委譲、自動化、変革ももろに要求されるし、本質的には延長の関係でしかない。シームが小さくなるのか、それとも丸ごとなくなるかの違いだけである。

効率化において、意外と見落とされがちなのが安定感だ。結果が読めないのはストレスを生む。1分後に必ず応答が来るのと、1〜55秒の範囲でランダムに応答がくるのとでは、同期的な場面であれば前者の方がマシだろう。後者の方が期待値が高いから良いと考えがちだが、私たちは人間である。タイミングが読めた方が行動しやすいし、精神衛生も良い。タイミングが不定な方が良いケースは、ゲームなど体験そのものを楽しんでいるときだけだ。そういうわけで、シームの観点で効率化を行うのであれば、期待値よりも安定感を重視した方が良い。とはいえ、期待値が高い方が有効な場面もあるだろう。特に多数の「決まった作業」から構成される、マニュアル的な仕事であれば、全体最適で見れば期待値を追求したくなる。しかし、それでもなるべく安定感を取ってほしい。デフォルト・リモートでは私たち人間の過ごしやすさが第一だからだ。

例を一つ見ていこう。マネージャー□さんの承認を必要とする手続きがあるが、□さんが多忙ゆえに、依頼を出しても結果が不安定だとする。しかし一番仕事ができる□さんはマネージャーから信頼されており、比較的すぐに相手にしてもらえる。□さんの承認を効率化したいとしよう。まずは□さんのように優秀になるとか、□さんに根回ししてもらうとか、□さんとの関係を深めようといった案が浮かぶかもしれないが、いけない。マネージャーの承認の仕方という仕組みを改善しなければならない。たとえば「朝、昼、夕方で最低1日3回はチェックする」とすれば、承認結果が来るまでのリードタイムは（日をまたがなければ）半日以内となる。これはつまり、1日3回チェックできるような仕事の仕方をマネージャーが身に付けねばならない。あるいは、それができないのなら、権限委譲（Focusのアプローチ）をして、承認の機会を手放さねばならない。どちらもしないのは、ただの傲慢である。

マネージャーは忙しく、拘束も多くなりがちなので、どうしても「あとで処理する」機会が出てくる。**あとで処理する能力または仕組みは必要不可欠だ**。ありがちなのがバインドモード（拘束的な過ごし方）を増やすことである。つまり、

- 「毎週金曜日、みんなで出社しようか」
- 「みんなの席は固まっているから、いつでも誰でも声をかけられる」
- 「もちろん私（□）もいるから、必要なら自由に声かけてもらえばいいよ。もちろん承認もやるよ」

のようなことをしがちだが、アンチパターンだ。というのも、他律的に集まって実現できるのは、承認**のやりとり**だけであって、承認作業そのものではないからだ。承認作業自体はマネージャーの□さんが握っており、□さんの能力と仕方にかかっている。集まったところで改善などできやしない。□さんのような、□さん向けのテクニックが表面化そして正当化されるだけだ。そうではなく、□さんが頼っているもの自体を効率化しなければならない。

性能の効率化

もう一つだけ、別の観点から論じておこう。性能の効率化である。デフォルト・リモートでは IT によるシステムやツールを多用するが、これらの動作が重たい――表示や処理が遅いことは、それだけで強大なシームとなる。ゲームや配信をしている人は、遅延は敵であるはずだ。体感的には 0.x 秒の違いしかなかったとしても、人間は意外と敏感なのでストレスになる。そういうのが嫌だからこそ、まさに人々は対面を求める。

この IT まわりの「重さ」の問題は、非常に奥深い世界だが、一つだけ端的な解を挙げておくと **性能を上げろ** である。パソコンのスペック、ネットワークの速度と安定感、打ちやすいキーボード、握りやすく滑りやすいマウス、十分に広いデスクと座り心地のある椅子、ディスプレイの大きさや枚数、普段使っている SaaS の制限解除（≒有償プラン契約）etc。このあたりの話は後の章で取り上げるが、ハードウェア的にもソフトウェア的にも性能は高いに越したことはないし、**少なくとも違和感やストレスを感じない程度には引き上げなくてはならない**。田舎暮らしや営業職には自動車が必須だろう。ケチくさがってママチャリや交通機関で我慢しなさい、などとはしないはずだ。ゲームや配信がしたいなら、それなりのスペックを揃えるだろう。ここをケチると自分のゲーム体感はもちろん、視聴者見えの映像品質もお粗末となって見向きもされない。

この概念を筆者は **最低限の性能（Minimum Spec）** と呼んでいる。説明書にありがちな「動作するための必要最小限の性能」ではなく、デフォルト・リモートを問題なく行うために必要な **最低限の** 性能である。デフォルト・リモートをするなら、最低限の性能はひととおり言語化しておき、全員が達成できるようにしなければならない。具体例は割愛するが、一つ挙げると、2025 年 2 月現在で、Windows のパソコンを使うなら、メモリは 16GB は必要だと思う。32GB でも問題ないくらいだ。8GB 以下は人権がないと言っていい。もちろん、仕事や役割によっても必要量は異なるが、それでもベースラインとして 16GB は必須であり、8GB 以下は論外だ。デフォルト・リモートはただでさえツールに頼るのだからなおさらである。この感覚を理解しなければならないし、**理解できない人が社員（が使う道具や環境）の性能、その整備や議論に携わってはならない**。

保全

保全（Maintenance）とはシームの肥大化を防ぐ取り組みを指す。

汚部屋のたとえ

振り返りを考えよう。すべての仕事と自分の人生について振り返りたいとする。一ヶ月に一度だけ行うとしたら、どうだろう。その一ヶ月間、何をしたかを思い出すのに苦労するはずだ。詳細を思い出すのに、頭を切り替える必要があるかもしれないし、思い出すための資料や記録をあちこち行ったり来たりして読み返すかもしれない。いずれにしても、たくさんのシームを越えることになる。では一週間に一度ならどうか。一ヶ月よりはマシだし、きりも良さそうである。では毎日は

どうか。振り返りはしやすいだろうが、面倒くさいかもしれない。しかも一週間や一ヶ月といった俯瞰的な振り返りも別途行う必要がある。

あるいは、もっとかんたんな例を出すと、部屋の掃除がわかりやすい。部屋が汚いとは、シームが多いと言い換えてもよい。頭の使い方ややることを切り替えるのはもちろん、単に探し物をするだけでも苦労するだろう。逆に部屋が綺麗だとスッキリと行える。しかし、綺麗も度が過ぎて、それこそ神経質にすべてのものを厳密にカテゴリー分けして引き出しに仕舞っているとしたら、それはそれで運用の手間がかかるに違いない。筆者がよく使うたとえとして「汚部屋のたとえ」がある。汚部屋とは足の踏み場もないほど汚い部屋を指すが、汚部屋であっても本人が生活できるのなら問題はない。逆に、それでは支障が出る場合は、面倒でも掃除や整理整頓をするしかない。お部屋自体が悪というわけではなく、あくまでも本人次第なのである。

シームも同様だ。本人にとって支障がないなら放置すればいい。ただし、放置できるということは、本人の処理能力が高いということである。実際、汚部屋でも支障なく暮らせる人は頭の性能が良く、ツールに頼らずとも原始的なコミュニケーションでどうにかなってしまう。リモートには頼らないし、頼ってもスマホで仕事できる。一方、支障がある人は、スマホでは足りないためパソコンを使うし、部屋も綺麗にする。と、このように、人によってまったく違ってくる。どれくらい保全するかは自分自身で模索するべきだ。**他人の感覚はあてにならない**。

保全 or 連携

とはいえ、デフォルト・リモートに話を戻すと、保全は必要である。まずリモートに寄せる都合上、ツールを重用する――つまりシームの影響が大きいので、汚部屋が平気な人でも保全を組み込まねばならない。ただでさえ汚部屋を綺麗にするモチベがないのである、放っておいたらあっという間にシームに押し潰されるか、そもそもデフォルト・リモートに必要な環境の整備さえもできずに「面倒くさいな出社でいいじゃん」「打ち合わせすればいいじゃん」となってしまう。足を引っ張りかねない。組織やチームのデフォルト・リモートを邪魔させないためには、最低限、保全の営みを容認してほしい。たとえば毎週数時間くらいを各種ツール（上で扱っている情報）の整理に充てても全く不思議ではない。汚部屋で済む人は「整理なんて要らないよ」と感じてしまうが、それは汚部屋で済むあなただけの話だ。無論、自分が汚部屋のまま過ごすのは自由である。個人のデスクトップアイコンやブラウザのタブがいくら散らかろうが、どうでもよい。

欲を言えば、**汚部屋に耐えられるような他律者は、汚部屋を保全する活動のかわりに「自律者に情報を連携するための活動」をしてほしい**。このあたりの連携は前章で詳しく議論したが、デフォルト・リモートでは、非同期的な情報共有をしない他律者の人達がそのとおりに生きる余地もある（尊重）。しかし、デフォルト・リモート自体を成立させるためには、他律者が出した情報も共有してもらわねばならない。そのための活動をせよ、と言っている。要は他律者同士で集まって喋って終わりではなく、その情報を全員が読み書きできるよう残せ、あるいは残せる人と連携して残してもらえ、ということだ。保全は汚部屋に耐えられない人達に任せればいい、代わりに空白となっている他律者ベースの情報の連携をカバーしてほしいのである。

コミュニケーション 3.0

最後に、保全における鉄板テクニックを取り上げておく。高頻度の小さな保全を行い、その結果を踏まえた上で、低頻度で大きな保全を行うといい。振り返りでいうと、毎日の振り返りを軽く行い、週一の振り返りは、その日次の振り返り結果（7個あるはず）を使って行う。月一の振り返りは、週一の振り返り結果（4〜5個あるはず）を使って行えばいい。こうすると、いきなり一週間分・一ヶ月分の振り返りを行うよりも楽にできる。最初の毎日の振り返りはとてつもなく面倒くさいが、これこそ典型的なシームであり、効率化や鍛錬で減らすなり Focus で挙げたようにシームそのものをなくすなりすればいい。

ここで「生成 AI を使えば、小さな保全は必要ない」と思われる人がいるかもしれない。そのとおりだが、現時点ではまだ難しいと思う。生成 AI に食わせるための情報を日頃から記録しておく必要があるわけだが、その記録の習慣や仕組みがそもそも無い。あるいは、やろうとしても、センシティブ安全性が低くてまともにできないのではないか。

結局、生成 AI に渡す情報を日頃から出せる素養が要る。あるいは「生成 AI が情報を吸い上げる場」で活動するという発想が要る。現時点で持っている人は少ないだろう。そもそも生成 AI のために情報を出すなど従来には無かったメンタルモデルであり、筆者は **コミュニケーション 3.0** と呼んでいる。特定の人を想定する 1.0、想定しないか曖昧にする 2.0 は前の章で出したが、これに続いて、AI を想定する 3.0 というわけだ。3.0 を身につけるには、情報を出す行動のフットワークを上げねばならない。シームも取っ払っていくことになる。高頻度の小さな保全はその第一歩となる。

たとえば分報をベースに、日、週、月単位の振り返りを雑に行ってみるのは良い取り組みだ。古くからは GTD のレビュー（3）、特に日次・週次・月次レビューも知られているが、実践の難しさでも有名らしい。コミュニケーション 3.0 に備えるなら、この程度は当たり前にできなくてはならない。一方で、わざわざ人間が人力で改めて情報を残すのにも限度がある。適性の有無もあろう。そういうわけで、実際は「普段活動している場所」にある情報を、AI に自動で食べてもらうのがいい。あるいは新しい場所で情報をやりとりするなら、AI に食べてもらうことも前提にする。ここでセンシティブ安全性が立ちはだかるため、センシティブの分離も要求される――と、このような素養と準備が求められるようになるだろう。

Feedback ～フィードバックを早くする～

Feedback は、その名のとおりフィードバックに絡むアプローチだ。**フィードバックが必要という状況は、それ自体がシームである**。この場合、シームを減らすとは、フィードバックを早くすることと同義となる。本節ではゲーミフィケーション、並列（パラレル）、ループとサイクルを取り上げる。

ゲーミフィケーション

ゲームが持つ楽しさを取り入れることをゲーミフィケーションという。進捗の解放やポイント報酬、

爽快感のあるグラフィック、他の利用者との競争、自分のパフォーマンスを客観視するためのログやリプレイ――ゲームで当たり前に使われている諸概念を、真面目な仕事の文脈にも取り入れるわけだ。

目的はただ一つで、**モチベーションの増加と持続** である。単純な話、楽しい方が続きやすい。ゲームは楽しい。だからゲームは続きやすい。そうでなくとも、ゲームの魅力は今さら解説するまでもない。家庭用、スマホ用、PC 用ゲームに限った話ではなく、スポーツもゲームの一種だし、ボードゲームや将棋や麻雀といった知的な遊びもゲームだ。ゲームとは極めて日常的な営みである。

ここで考えないといけないのは、ゲーミフィケーションを行うとして、具体的に何ができるかだ。大会やグラフィックやリプレイといったものをいきなりあらゆる場面でつくることなどできやしない。かといって、いちいちデザイナーに頼んで一からつくってもらうわけにもいかない。仕事は人やチームの数だけ存在する。きりがない。さらに、デフォルト・リモートであるから、現地で対面で集まるのではなくリモートでも、非同期的であっても成立させねばならない。

この難題に挑むために、まずはゲームの本質を定めよう。

フィードバックの物量を桁違いに増やす

ゲームの本質とは何か。正解は無いだろうが、WNGF という言葉を聞いたことがある。出所は不明だが Winnable（勝てる）、Novel（斬新である）、Goal（目標がある）、Feedback（フィードバックがすぐに得られる）の略らしい。なるほど、たしかに本質を捉えていそうだ。では、このうち難題を打ち崩せそうなものはどれだろうか。この問いにも正解はあるまい。問答したいわけじゃないのでここらにしておくが、本節では **ゲームの本質はフィードバックの物量が桁違いに多いこと** としておく。

今はデフォルト・リモートの文脈で、シームレスに近づくために、Feedback のアプローチとしてゲーミフィケーションに頼ろう、というものだが、これで方針が定まった。**フィードバックの物量が桁違いに増えるようにすればいい**。具体的には以下のアプローチを取れる。

- 1: 数の力に頼る
- 2: 高密度に過ごす
- 3: ガバナンスを利かせる

それぞれ詳しく見ていこう。

1: 数の力に頼る

Q&A サイトに質問を出すと、場所と話題にもよるが多数の物知りや暇人から回答が届く。有名人が SNS で何かを募集すると、色んな属性を持ったファンからの多様な回答が集まる。また生成 AI はインターネット上の情報も幅広く食べているし、精度を上げるために学習量に物を言わせてもいる。私たち人間は原始的な生物にすぎないが、技術により、桁違いの人間や

情報を集めることができる。

この、いわば数の力に頼るアプローチには、二つの効果がある。まずは純粋に多くの情報が得られるので便利だ。スパムなどゴミが集まりすぎると破綻するとか、議論のスタートラインに立てる知識のない素人の意見が集まったところで役に立たない等、万能ではないが、それでも強力だ。仮に「収集」と呼びたい。その名のとおり、集めやすくなる（というより集まりやすくなる）わけだ。次に、数がある程度以上集まると、個々の集合を超えた、総体としての能力が出てくることである。これは創発と呼ばれたり、知性の文脈では集合知または集団的知性と呼ばれたりするが、ともかく、どこかで「全体」が「単なる部分の総和」を超えてくる。

数の力に頼ることで、収集と創発の恩恵が得られる。収集は問題解決に適する。創発は変革につながる。いずれにせよ、情報がたくさん集まり、それのフィードバックもたくさん行われるような場があることで、社員全員が影響を受ける。ゲーミフィケーションというと、フォロワーやポイントを設けたり、議論が過激化しやすい話題を強調して誘導したりするだろうが、無用だ。むしろ注目を集めることが目的となって本末転倒となる。

余計な仕掛けは何も要らない。情報が集まり、フィードバックも集まり、その流量がとてつもない場所が出来ている――これだけでいい。これだけで楽しく、面白くて、自然と足を運ぶようになる。人が集まり、誰もが自分なりに主体的に関わるようになって、まず収集の効果が得られるようになる。そうして盛り上がり続けると、どこかで誰かが面白い情報を出して、それに他の人も乗って、と革新的なアイデアや議論が出てくるようになるだろう。あるいは、革新性がなくとも、バタフライ・エフェクトのように少しずつ色んな場所で相互に影響しあって、良い方向に変化していくだろう（逆もある）。従来では決して生まれなかった創発が起こるのだ。

デフォルト・リモートの現場にまで落とし込むと、社員全員が情報とフィードバックを遠慮なく出しまくれる場をつくればいい、となる。本当はそれこそインターネットのように社外も含めて集めた方がいいのだが、センシティブ安全性を保ちながら行うのは非常に難しく、本書では割愛する。社内にのみ焦点を当てる。数の力とは、社員全員の力に等しい。

シンプルなやり方として **Rapid Q&A** を紹介する。

これはビジネスチャットを使うもので、社員全員を一つのチャンネルに入れた上で、そのチャンネルをいつでも誰でもなんでも質問・回答できるように解禁する。仮に従業員 3000 人の会社があるなら、3000 人全員を入れる。3000 人全員が、いつでも誰でも自由に質問と回答を行う。初動のハードルさえ越えたら、流速は到底追えないほど早くなるだろうが、それでいい。社員たちは、タイムラインをスクロールしながら眺めるように、Rapid Q&A のチャンネルを眺めるようになる。そして回答できるものに気軽に回答していく。質問すると、（おそらく知っている人が誰かして見ていてすぐに書くので）すぐに回答が届く。

Rapid（迅速）の由来もここにある。質問もすぐ書けるし、回答もすぐ返ってくるわけだ。その際、絶対に周知させねばならないコンセプトが一つあって、**探すな、訊け** だ。Q&A の文脈では「まずは過去の質問を探せ」「FAQ やドキュメントを読め」といわれがちだが、そんなことはしなくていい。質問があれば、何でもいいのでとにかく尋ねればいいだけだ。過去の重複とか、ドキュ

メントに書いてあるとか、社員なら誰でも知ってそうだからなどは関係がない。Q&A は質問者が答えを知れたらそれでいい。「過去の質問や既存のナレッジを読むこと」は手段であって、目的ではない。よく「わかる人に聞いた方が早いよ」などと言ってデジタルツールに頼ろうとしない人がいるが、発想としては同じだ。これを環境として無理やり実現するために、全社員を一つのチャンネルに押し込めているだけである。

注意点としては、通知の排除と匿名性の確保が両方要ることだ。まず、この流速で通知が来るのは鬱陶しすぎるため絶対に禁止しなければならない。メンションも使用禁止だし、それでも使用する者は追放しなければならない。次に、実名アバターだと、おそらく初動のハードルは越えられないだろうから、隠した方がいい。しかし Slack にせよ Teams にせよ、匿名で投稿する機能はないため机上の空論だ。API を使って外部アプリから投稿する形を取れば不可能ではないが、それはシームフルであり定着しないだろう。素のビジネスチャットの使用感でそのまま行えることはマストである。実質的に実名ベースで少しずつ盛り上げていく他はないのだが、それができるだけのテネットを現代人はまだ持っていない。Rapid Q&A チャンネルを遠慮なく使ってもいい号令を、組織として正式に出さなければならないだろう――と、運用上の課題は多数存在するが、この程度ならどうとでもクリアできよう。

Rapid Q&A はシンプルでありながら、数の力に頼るための本質と課題を洗い出してくれる。

2: 高密度に過ごす

まさに出社のスタイルで期待することでもあるが、みんなが同じ時間、同じ場所に集まっておれば、話したいときに話したいだけ話すことができる。筆者が生きた文脈（Live Context）と名付けたティール組織のあり方もまさにそうだった。コミュニケーションを「相手に何か言う」「そのフィードバックが来る」と捉えると、一緒に過ごす密度を高めれば高めるほどフィードバック効率が上がることになる。ただし、そのためにはティール組織のような全く新しい組織パラダイムで再構築しなければならない。従来の階層的な組織や、従業員のライフを重視してワークの密度を上げづらい組織では不可能だ。本質的な仕事にだけ集中できる、コンパクトで快適で最低限のチームを組織全体で実現した上で、かつ物理的に集めることで、ようやく可能となる。

これは思っているよりもずっと難しい。筆者がよく使う例が「チーム一つ一つが採用の権限も持つ」であり、これは人事部ではなくチーム各々が自分達でチームメンバーの採用を完結できることを意味する。もちろん人事面は素人だろうし、組織としての統一も必要だから人事的な機能も存在するが、あくまでサポートするだけだ。人事的な機能はサポーターであって、意思決定権は持たない。サポーターを使った上で、実際どうするかはあくまでも各チームが握っている――本質への専念とは、このレベルである。スタートラインと言ってもいい。このスタートラインに立つことさえできずに、この高密度な過ごし方をしたところで持続などできやしない。あるいは、せいぜいカリスマひとりが多数の信者をコントロールする構図だが、そのカリスマがボトルネックになる。おそらくカリスマに気に入られた人しか桁違いのフィードバックを享受できまい。そして、その座を奪い合う政治も勃発する。あるいは競争から下りる「静かな退職」も増える。本書でも何度か述べてきたが、構造的な限界がどうしてもある。

ティール的な高密度が難しければ、もっとライトなやり方もある。IT エンジニアであればペアプログラミング、あるいはモブプログラミング（4）をご存知かもしれない。プログラミングという従来個人で行うべき創造的な作業を、あえて複数人で行うものである。ドライバーという「つくる人」を一人だけ設定し、残りの全員はナビゲーターという「ツッコミを入れる人」となって、ドライバーがつくるものを見ながら議論をする。

言ってしまえば **全員が同じ時間、同じ課題にフルコミットする** わけだ。本書ではここまで拘束という言葉を使ってきた。特にコミュニケーションは時間と場所を拘束する、だから負担が高くて私たちに優しくない。この悪しき状況を緩和していくのは当然であり、住み込みから出社のパラダイムとなっているが、本書ではその先、第三のパラダイム「リモート」を提唱し、デフォルト・リモートと呼んできた。拘束は避けるべきものだが、本項の文脈では逆で、拘束に身を委ねる。それも時間と場所だけじゃない。**課題も拘束している**。時間は必須だが、場所はなくてもいい。その場合はリモート経由で時間と課題を拘束することになるわけで、リモートワークの知見は引き続き要求される。

プログラミングというと限定的なので、ペアワークあるいはモブワークと呼びたい。ペアの場合は二人、モブの場合は三人以上だ。いずれにしても、手を動かすドライバーは一人で、残り全員はナビゲーターとなって、時間と課題を拘束する。内職は許されないし、ちょっと席外しますとか割り込み入ったので少し抜けますさえも許さない。仮に 30 分の間、タスク A に集中すると決めたら、全員は例外なく 30 分間、タスク A に専念しなければならない。高密度とはそういうことだ。そこまで徹底するからこそ、フィードバックも桁違いに増やせる。ただし負荷も高く、一日中どころか半日さえも保たないので、使いすぎないように注意してほしい。細かいテクニックはペアプログラミングやモブプログラミングの方で知見が貯まっていると思うので、必要なら調べてほしい。

ペアワークとモブワークは、ティール組織を実現するよりはかんたんだ。かつ、（高密度を重視する）ティール組織の本質を捉えることもできる。たとえばワークをどのように進めるかはきつくファシリテーションするし、プロセスも作り込むことになるだろうが、そういったガバナンスはまさにティール組織の特徴でもある。たとえばティールの一種として挙げられる「ホラクラシー」では、戦術会議（Tactical Meeting）の形で会議プロセスを憲法的に定めていたりする。もちろん、そんなきつい働き方になんとなく従うのも難しく、ビジョンへの理解や納得感の醸成といったエンゲージメント活動にもかなり費やすことになる。Flow の節で述べたように鍛錬も要るだろう。

3: ガバナンスを利かせる

フィードバックを桁違いに増やすための、もう一つのアプローチは共通言語をつくることだ。属人性を回避するために、社内全員に通用する共通の言語と、それに基づいた体系をつくる。かつ、体系を正しく使っていることを、透明性をもって保障する。あるいは不正があってもすぐに是正できるようにする。

例として業績評価を挙げよう。業績評価は避けられない営みの一つだが、しばしば属人的となる。等級、給与体系、評価観点こそ定められているものの、それらをどう解釈し、誰に対してどこを当てはめるかは人間が決めるため、結局属人的となってしまう。なぜアイツが評価されてる

んだ、との不満を抱いた人は少なくあるまい。あるいは、あなたが高い評価を得る側の人間であるなら、評価を得るための作文や立ち回りには腐心してきたはずだ。もしくはインポスター症候群にかかっていて「なんで私なんかが評価されているのだろう」と罪悪感を抱いているかもしれないが。いずれにせよ、極めて属人的な営みなのだ。

さて、属人性を潰すために取る手段の一つがガバナンスだが、ガバナンスというと権力者が属人的に統治するイメージを思い浮かべるかもしれない。違う。フィードバックを増やすという文脈では、属人的である限り、たかが知れている。物量も小さいし、安定もせず、政治に強い者だけが勝てる世界を脱せない。そうではなく、万人が享受できる、再現性があって、安定したガバナンスを利かせたいのである。

どうやるか。共通の言語と体系、そしてその運用の透明性が必要だ。言及しづらいので**トランスパレント・ガバナンス（Transparent Governance）**と呼びたい。透明な政治と呼んでもいいが、政治という言葉のイメージに引っ張られたくないので、あえて横文字を使わせてもらう。

業績評価において、トランスパレント・ガバナンスを実現するにはどうしたらいいだろう。筆者はバッジの概念を提唱している。バッジとは実績を言語化したもので、社員はバッジを使ってお互いに評価を行う。この仕事はバッジAとバッジBから構成されているよね、という事前の合意を取り、仕事を終わらせたあとに、事後評価を行って、AとBを入手する。このあたりのやりとりはすべてシステム上に記録され、社員なら誰でも見ることができる（透明性）し、関係するメンバーもやりとりや評価に参加できる。また各社員がどんなバッジを持っているかとか、いつ入手したのか、誰が評価したのかといったことも全部見える。

このようなバッジシステムがあれば、トランスパレント・ガバナンスを存分に利かせることができる。等級については、バッジのセットで定義すればいい。マネージャー職に必要なのはこれこれのバッジ、高次の専門職に必要なのはこれこれのバッジ、といったように定義していけばいい。

特に難しいのはバッジ自体の定義とメンテナンスであるが、本書では割愛する。軽く挙げておくと、BMPO（Badge Management Program Office）のような専門組織をつくったり、バッジメンテナーのような役割をつくってリードしたりする。従来の人事部や現場の人間だけでまかなうのは難しい。特にバッジは動的かつ多様であるため、人事という静的なテネットとは相性が悪い。また、バッジと聞くとゲーム、あるいはライトノベルの一ジャンルとして盛り上がっている「なろう系」、特に異世界モノ（のギルドのシステムなど）を思い浮かべるかもしれないが、まさにそのとおりだ。そういったコンテンツをつくれるクリエイターを登用した方が、おそらく上手くいく。外部のクリエイターに社内の知識を学んでもらうのが厳しいのであれば、社員の中からクリエイターの資質を持つ者を登用するといい（リスキリング）。

パラレル

パラレルとは並列という意味であり、複数のものを同時に実行するさまを指す。

生成 AI の使用を例にしよう。ChatGPT でも Claude でも何でもいい。重用している人であれば、聞きたいことはそれなりに多いはずだ。仮に 5 個ほど浮かんでいるとする。あなたならどうやるか。

最も単純なのは、ChatGPT の対話画面を一つ出して、順に一つずつ聞いていくことだろう。これは単純だが、いちいち待たないといけないので時間がかかる。また、待っている間に、聞きたかったことを忘れるかもしれない。あるいは忘れてはいないが、やる気がなくなって聞く腰が上がらなくなるかもしれない。これでは不便だ、というわけで、おそらくブラウザで複数のタブを立ち上げると思う。それこそタブを5つ開いて、個別に聞いてしまえばいい。あとで確認する手間は生じるし、確認を忘れてしまうこともよくあるが、それでも一つずつ実行するよりははるかに効率的だ。フィードバック効率も段違いであると言える。単純計算で 5 倍だ。

この例は比較的わかりやすいが、私たちは並列の概念をイメージしにくい。人間がシングルタスク脳だからだ。そもそもマルチタスクというのは幻想で、シングルタスクを高度に切り替えているにすぎない。コンピュータの仕組みもそうなっている。しかし、処理はすぐに終わるとは限らず、終わるまでに時間がかかることが多い。処理を走らせておいて放置しておくことになる。

並列にやりたければ、ガンガンと処理を走らせることだ。どんな状況においても、まず並列して走らせることはできないかを考えてみるといい。それだけで並列の数を増やせる、ひいてはフィードバック効率を挙げられる。**並列の概念は非直感的でイメージしづらいので、意識的に考えてみなくてはならない。**

次に、すでに述べたように、単に走らせただけでは意味がない。あとで結果を確認しなければならない。ここで出てくるのがマネジメントの概念である。セルフマネジメントをして自分自身が定期的にもれなく確認できるようにするのか、それとも誰かに任せてその人たちに報告させる（マネージャーなど従来の役職者はまさにこれだ）のか、それともリマインダーを上手くつくって自動的に思い出させてもらうようにするか――やり方は色々とあるが、いずれにせよ結果の確認を阻むシームがあると言えるわけで、本章でここまで述べてきたシームレスの取り組みがまさに使える。

最後に、並列を増やしたければ、リソースをケチらないことである。また生成 AI を例にしよう。本当にたくさんの質問を並列で動かしたい場合、アプリやブラウザから使う ChatGPT ではなく、プログラムから使う API（特定の機能を実行させるための機械用窓口）を使うべきだろう。API であれば、呼び出した分だけ使えるので、それこそ 10、100 の問い合わせを並行で走らせることさえできる。従量課金なので、その分、お金はかかってしまうが、ケチらずに走らせまくることで見えてくる世界というものがある。

IT エンジニアの界隈では Cline や Devin といった AI 支援が盛り上がっている。これらは「単に ChatGPT からコードをコピペする」というシームを解消したものであり、コードを書くのに使うエディター上に直接コードを挿入したり、何ならエンジニアが人力で行うサーバー上の作業を勝手に行ったりする。裏では多数の API リクエストが動いており、したがって料金も安くなくて、数十ドル、数百ドルくらいすぐに行ってしまう。逆を言えば、遠慮なく並列に走らせているからこそ、そういったシームレスを実現できていると言える。

というわけで、フィードバックを増やしたい――特に回数を増やしたいのであれば、並列に着目してみるのもアリなのだ。

ループとサイクル

ゲーミフィケーションでは、高密度な場や仕組みをつくることでフィードバックの物量を増やそうとした。並列では、順に一つずつではなく同時並行に複数を走らせることで、フィードバックの回数を増やそうとした。もう一つ、ループとサイクルを紹介するが、これらは**安定したフィードバック**を狙うものである。本章でも何度か述べたが、フィードバックはランダムに来るよりも、固定のタイミングで来てくれた方が精神衛生に良い。もう少し一般化すると、一定のリズムや来てくれた方が何かとやりやすいと言える。ループやサイクルは、その名のとおり循環的なプロセスを定義することで、リズミカルにフィードバックが来る流れをつくるものだ。

ループとサイクルの違いを見ておこう。厳密な違いは状況によるか、そもそも詰められていないこともあり一概には言えないが、ループは主体を変えずに行為を一巡させ、サイクルは主体（たとえば状態）を変えながらその変化を（たとえば状態を）一巡させる。ここでは単純に**状態の一巡に注目したものをサイクル**と呼びたい。

ループとサイクルの例

たとえば PDCA サイクルは、Plan → Do → Check → Act であるが、今現在は Plan のフェーズにいますよという「状態」を強調する。各状態ごとの動き方と、前後の状態との連携方法を設計することで、プロセス全体に統一性をもたせる。実際、PDCA サイクルは元々は製造の品質管理の文脈で開発された。「状態」がわかりづらいなら「モード」でもいい。

もう一つ、似た概念として OODA ループがあり、Observe → Orient → Decide → Action を回すが、戦闘機のパイロット向けの、軍事的な意思決定プロセスである。状態よりも行動に注目しており、パイロットは素早く意思決定を回せる。PDCA サイクルが管理プロセスだとしたら、OODA ループは行動フレームワークとでも言えるだろう。

フィードバックのためのループとサイクル

さて、フィードバックを増やすために、どんなループやサイクルが使えるかとの話に移っていきたい。PDCA や OODA のように、既存の理論をあれこれ示してもいいが、本書としてあえて行う価値はあるまい。かわりに、フィードバックのためのループやサイクルのつくりかたを議論したい。

まずはループをつくるといい。つまり、行動をしてフィードバックをもらうという一連の流れを**自分なりにつくる**。観察から把握できることもあれば、「自分はこうしている」のようにやり方を言語化することもあるが、どちらでもいい。いずれにせよ、複数の行動パターンが一巡する形での言語化と構造化ができるはずだ。最後に、可能なら名前をつけたい。そうして自分なりの「何とかループ」をまがいなりにもつくれたら、あとはそれを意識して取り組み続けることでフィードバックが

安定する。改良する場合も、何とかループのどこをどのように変えるか、という形で堅実に変えていける。

一つ例を示すと、筆者は習慣化の文脈で GSAAD ループなるものをつくった。これは Get → Set → Act → ADjust から成るループであり、仕掛け先を決めて（仕掛け先を Get して）、そこに仕掛けを施して（仕掛けを Set して）、仕掛けに従う形で日々行動（Act）して、その行動の結果を踏まえて諸々を微調整（Adjust）して――と続く。

gsaad

たとえば新しく散歩の習慣をつくりたい場合、普段見ているカレンダーに、定期的な予定として散歩を入れる。あるいは、リモートワーク終了時に会社のパソコンを仕舞う棚があるとして、その前にランニングシューズを置くというタスクをつくってもいいだろう。この場合、仕掛け先と仕掛けのペアは二つあって、カレンダーとシューズ配置タスク、そして棚とシューズである。どちらのやり方が定着するかは人次第だし、他のやり方が必要かもしれないが、ともかく筆者は、仕掛け先に仕掛けるという、このやり方を GSAAD ループと呼んで捉えたわけだ。実際、これのおかげで、筆者は息するように習慣を追加したり修正したりできる。習慣が定着しない＝仕掛け先 or 仕掛け方が悪い、との形でフィードバックもわかるため、次の対策も立てやすい。

ループだけでどうにかなれば単純なのだが、そうもいかないことも多い。おそらく通常は「フィードバックがもらえる状態」と「それ以外の状態」があって、両状態を行き来するはずだ。それこそ仕事では、成果物を提出して見てもらう構図がよく生じると思うが、これは事実上成果物をつくるフェーズ（状態だとわかりづらいが同義だ）、渡すフェーズ、相手に見てもらうフェーズ、見てもらったあとにその結果を教えてもらうフェーズなどに分かれている。この場合、フィードバックと呼ばれる部分は「教えてもらうフェーズ」だけであり、このフェーズにいかに安定的に繰り返し至れるかが肝心となる。これを **フィードバック状態** と呼びたい。個人で好き勝手に動ける場合は、自分なりのループを確立して回せばよかったが、それができない場合は、フィードバック状態を見極めた上で、この状態と出会える機会を最大化せねばならない。サイクルの出番だ。

最もわかりやすいのは定例会議だろう。プロジェクトでは毎月、毎週、あるいは厳しい場合は毎日進捗会議が開催される。この会議はフィードバックがもらえる場であり、フィードバック状態と言えるだろう。仮に毎週金曜日の 13:00 〜 14:00 に進捗会議があるとしたら、サイクルとしては、たとえば次のようになる。

- 1: 月火水木の「活動状態」
- 2: 金曜日の「フィードバック状態」

厳密には金曜日の午前と、14 時以降も「活動状態」かもしれないが、おそらく会議を踏まえた準備や振り返りが必要なはずだ。この二つも独立させてみると、適当な例だが、次のようになるだろう。

- 1: 月火水木の「活動状態」

- 2: 金曜日午前の「準備状態」
- 3: 同上 13:00 〜 14:00 の「フィードバック状態」
- 4: 同上、14:00 以降の「振り返り・付き合い状態」

これは週一頻度のフィードバックに対応するサイクルだと言える。そのために、フィードバック当日は準備や振り返り（あとはおそらく飲み会などの付き合いがあるのだろう）に充てている。汎用的に使えるかは怪しいが、これをたとえば次のようにつくってみよう。

- 1: Action（活動）
- 2: Prepare（フィードバック機会直前の準備）
- 3: Feedback（フィードバック）
- 4: After（アフター）

頭文字を取って、APFA サイクルとでも名付けよう。APFA サイクルは、週一頻度のフィードバックに対応できるサイクルの一つであり、直前の準備に力を入れることと、フィードバック後を大胆にアフターと捉えてしまうことが特徴である。適用シチュエーションは限られるが、活動・準備・振り返りや気分転換をバランスよく回せるサイクルである。

と、勢いで一つつくってみたが、いかがだろうか。APFA サイクルの議論は割愛するが、こうしてつくってみることで行動に統一感が生まれる。リズミカルにこなせて、メリハリもつけられるのでモチベーションも殺されにくい。言語化しているので他者にも通じるし、チームで取り組んでみることもできるし、もちろん改良もできる。

さて、ここまでループとサイクルをつくる例を、かんたんながら見てきた。基本はこれだけであり、ループとサイクルをつくれるだけでもフィードバック（の安定化）を引き寄せていける。

ループやサイクルの入れ子

もう一つだけ、応用を述べておくと、ネスト（入れ子）することもできる。ループやサイクルの中に、ループやサイクルを入れるのである。

ループのネストの例を挙げよう。ダブルループ学習という理論がある。これは行動した結果を踏まえて、改善を施して、また行動するというシンプルな改善ループ（シングルループ）の他に、目標やルールや価値観といった前提を決めてシングルループに行くというループ（ダブルループ）も定義している。つまりループがネストしている。

double_loop

普段はシングルループを回していくが、時にはダブルループを回すことでシングルループ自体の改善もできる。

典型的には、自由にバリバリ動ける部分はなるべくループをつくり、組織としての統一や安定感が欲しいところはサイクルをつくるといい。例を挙げよう。仮に 6 人のチームがいるとして、マネー

ジャーが 1 人、メンバーが 5 人とする。メンバーの 4 人は従順だが、1 人だけ、□さんだけ自律性が強いとしよう。チームとしてはハイブリッドワークをしていて、毎週水・金で出社にしているが、□さんは難色を示してリモートを主張している。□さんは優秀であり、自律的に人並以上の成果を出す力を持つが、会議や報連相のタイミングが合わないせいで、今のところ戦力外に近い扱いを受けている――こういうときもループとサイクルが役に立つ。

要は□さんからのフィードバック、□さんへのフィードバックの部分でシームが大きいのだから、ループとサイクルをつくって解消すればいい。おそらくチーム全体としては、サイクルを回しているだろう。自律的な□さんはそれに従わない。しかしフィードバックは必要だから、「□さんの成果を受け付ける時間帯」をつくって、サイクルに組み込む。□さんだけ特別扱いしているが仕方ない。かわりに、□さんもその機会には参加する。おそらくリモートで 1 時間なのか、数時間なのかはわからないが、会議をするだろう。□さんが気難しい人の場合、□さん向けの行動理論として「□さん用ループ」も必要かもしれない。

一方で、□さんの目線に立つと、普段は何らかのループで行動しているはずで、言語化・構造化しておけば再現性を出せる（□さんのような人材を増やしたり、□さんの立ち回りを誰かが真似たりできてチームの多様性が増す）。チームからの会議要請に応えるためには、全体として「会議モード」と「自律モード」のシンプルなサイクルをつくって、会議モード側で吸収することになるだろう――と、このように過ごし方をサイクルとループに分解しながら、お互いに連携していけば、上手く歩み寄っていくことができる。私たちはこう動いてます、私はこうしてますのような行動パターンをサイクルとループで表現し、それを見ながら、ここをこのように変えましょうと建設的に調整していける。あるいは歩み寄りが難しくても、自分の側で「相手に対応するための時間帯」をループやサイクルとして組み込めば、割り切って対応できるようになる。

まとめ

- リモートで過ごすからこそ、あらゆるシーム（継ぎ目）はできるだけ減らすべき
- Focus、シーム自体をなくすアプローチ
  - 誰かに任せてやりとりのシーム自体をなくす「委譲」
  - 人間ではなく機械にやらせることでシーム超えを機械に代行させる「自動化」
  - A → B → C を A → C にしたり A にしたりなど、抜本的な再構成を行う「変革」
- Flow、シームを小さくするアプローチ
  - 自分が強くなることでシームの影響を減らす「鍛錬」
  - 仕組みの方を改善することでシームを小さくする「効率化」
  - 切り替えに要するシームが大きくなるのを防ぐ「保全」
- Feedback、フィードバックを早くするアプローチ
  - フィードバックの物量を桁違いに増やす「ゲーミフィケーション」
  - フィードバックが必要な行動を並行して走らせる「並列（パラレル）」
  - フィードバックを安定化させるための「ループとサイクル」

Chapter-7 役割分担

現代では「何でも屋」が要求される。採用面でこそポジションなるものが明確に定義されてはいるものの、共通言語でしかなく、その通りに取り組むだけで済むほど単純ではない。泥臭いとの言葉がよく使われている。仕事は、現場は、実状は綺麗事ではない。だからポジションを鵜呑みにせず、必要なことは何でもせよ、場にも適応せよ、との要請が事実上存在している。この要請に応えるには、何でもできる or できるようになるための要領の良さと、それを支えるための高密度な連携つまりは拘束が必須だが、これではデフォルト・リモートなどできたものではない。

発想を変えねばならない。何でも屋を営む器用貧乏への偏重をやめて、必要に応じて役割を動的につくって、変えて、柔軟に分担していくことを考える。何でも屋は無数に存在する役割の一つでしかない。本書でもすでに、デフォルト・リモートを実現するために新しい役割を何度かつくってきたはずだ。そういうことを誰もが行えるようにしたい。**役割そのものを扱う目線に立つ**のだ。

本章では役割そのものを扱えるようにするためのテネット、やり方や考え方を見ていく。

テネットを壊す

既存のテネットが邪魔をしていて、まともな議論ができない。まずは壊すことから始めよう。三つほど壊す。

平等指向をやめる

私たちは単一のあり方に全員を従わせようとしがちだ。その方が効率的に管理できるし、調和は日本の文化でもあるし、現代はチームの時代だが、チーム一丸となるためには、単一の指針をつくって全員が足並みを揃えるのが良いともされる。抗う方が難しい。**平等指向**と呼ぶことにしよう。平等とは全員に一律同じ水準を課すことであり、多様性の観点で言えば、ただ一種類の要素だけ認めることに等しく、多様性が無い状態だ。

チーム全員が残業しているとしよう。ここに中途社員として新人□さんが入ってきた。□さんは残業しないポリシーを持っている。さて、□さんのポリシーは通るだろうか。おそらく通らない。「うちは残業も必要だから」「みんな残業しているから」といった言い分で、□さんは潰されてしまうだろう。形式的には就業規則を持ち出して、必要に応じて上司命令で残業もできるようになってるんだよなどと説得されるかもしれない。あるいは賢い人なら、負荷の分散や均一化という観点で攻めてくるだろう。いずれにせよ、残業をしないというあり方は認めてもらえず、マジョリティの残業前提なあり方へと平準化されてしまう。これが平等指向だ。

平等指向は壊さねばならない。平等指向がある限り、デフォルト・リモートなんてできたものじゃ

ない。出社派 vs リモート派の構図を脱せないからだ。出社派もリモート派もどちらも共存できるし、必要に応じて切り替えられるのが理想である。V.S. の構図では、どちらかが、あるいは両方が負けてしまうし、負けないための戦いつまりは政治が勃発する。Win-Win を目指さなくてはならない。もちろん、理想を掲げるだけでは意味がない。ここまで様々解説してきたように、理想を実現するためのテネット、やり方と考え方、道具を揃えて、使いこなさないといけない。平等指向は最初に立ちはだかるハードルであり、必ず壊さなければならない。

平等は思考停止にも等しい。個々を理解して最適に配分することを諦めており、とりあえず全員に平均を課せばいいか、とする投げやりでおざなりな所業である。労働時間一つを取ってみてもそうだ。1 日 8 時間とか、月所定 150 時間とか、冷静に考えてみればおかしいことこの上ない。もっと働きたい人もいれば働きたくない人もいるし、普段働ける人でも働けなくなったり、その逆もある。同じ人でも日によって違うだろう。なのに、バカの一つ覚えみたいに 8 時間だの 150 時間だのと課して、律儀に守っている。こう言うとよく「法律だから」「ルールだから」などと返されるが、そういう議論はしていない。余談だが、労働基準法では上限こそ定めているものの下限は定めていない。

せっかくなので、思考停止に抗ってみよう。たとえばマネジメントの章で挙げたスラックタイムを使えば、「業務時間中の余裕時間」をつくれる。1 日 8 時間が必須であっても、スラックタイムを 3 時間設定すれば、実質的な労働時間は 5 時間だ。もちろん、スラックの使い方は自由なので、もっと働きたい人はその 3 時間も労働すればいい。スラックタイムという概念一つで、かなり融通が利くようになる。労働時間の管理やワークとライフの切り替えが難しいなら、できない人向けにできるようにする仕組みや役割をつくって使えばいい――と、他にやりようはいくらでもあるだろう。難しいことではない。つくろうと思えばつくれると思う。なぜつくれないか、いや、**なぜつくらないかというと、単に平等指向のテネットに阻まれているだけ** だ。

とはいえ、平等指向を知って、じゃあ今日から行動できるかというと、おそらく難しい。平等は事なかれ主義にとって居心地のいいものとなっている。平等として課された事項は「理」であり、それに従っておれば要らぬリスク――衝突であるとか、出た杭として打たれたり等を防げる。そもそも法律や規則などペナルティが設定されていることもあるし、破ったとの評もついてまわる。リスクがでかすぎる。代わりに、理を我慢するコストは支払わねばならないが、支払いさえすれば無難に生き続けることができる――。

たしかにそのとおりだが、それでは働き方などちっとも良くならない。理をつくるのは現在の権力者であり、当然ながら権力者は利益を得るために搾取を行う。あるいは、そのつもりがなくても、権力者も人間であり限度があって、ボトルネックが生じるから、そこの奪い合いが生じて勝敗が出来て、敗者が搾取される。**この構造は、平等である限りは脱せない**。平等ではなく、必要な要素を n 通り全部つくって、それらの使い分けと連携が行えるようなあり方を目指すべきだ。

デフォルト・リモートは、リモートワークの観点から脱平等を目指していると言ってもいい。

兼務やマルチタスクをやめる

私たちはすぐに兼務やマルチタスクをしたがる。たとえば新規事業や組織改善、それこそデフォルト・リモートを推進していくなら専用の組織や部隊をつくるのは当然だが、おそらくそうはならず、すでに忙しい誰かを捕まえて兼務にするだろう。その意図は三つある。

- 1: 専任の体制にかかるリソースを抑えたいから
- 2: 新しい存在をつくると、政治的な均衡が崩れかねないから
- 3: 忙しい人（≒経験豊富な人や優秀な人）でなければ務まらないと思っているから

一理あるし、それはそれで一つの正解だが、デフォルト・リモートとは相容れない。

まず前章で述べたとおり、変革には投資が必要であり、デフォルト・リモートも変革だ。専任を惜しむこと自体が論外である。また政治については、そもそもなくすことを目指している。最後に、忙しい人は手を動かして前に進むのは得意だろうが、それ以外の仕事が向いているとは限らないし、おそらく向いていない。よほどの才人でもなければ、あれもこれもできるなんてことはない。そもそも忙しくてまともにコミットできやしない。

似た話としてマルチタスクもある。マルチタスクとは、同時に多くのタスクを抱えて、それらを並行しながらちびちび進めるさまを指す。といっても、人間もコンピュータも本当に並行で走らせているわけではない。原理的には同時に一つのことにしか着手できず、この着手の対象を高速に切り替えることで、あたかも並行しているように見せているだけである。コンピュータであれば、機械なので問題なく切り替えられるが、人間だとそうもいかない。一つずつ確実に終わらせるか、あるいは最低でもきりのいいところまで終わらせてから切り替えるのが良い。しかし、一方で、私たちはわかりやすい刺激に反応することで欲求を満たせるようにもなっている。SNS やゲームや動画はまさにそこを突いて、陥らせることで搾取するものだ。

そういうわけで、あれもこれもと手を付けるマルチタスクのスタイルが自然と起きてしまう。なまじ優秀だと、それでも仕事ができてしまうし、現代はそういう人達こそが勝者として君臨する世界でもある。悪いことではないし、そのおかげで今があるわけだが、マルチタスカーの独壇場となってしまっている。実際に仕事でも泥臭く何でも屋が要求されがちなのは、冒頭で述べたとおり。

一言で表現するなら「ビジー」であろう。兼務にせよマルチタスクにせよ、抱え込みすぎである。そして欲張りすぎである。個人のスタンスとしては尊重するべきだし、今までと同様、存分に頼ればいいが、決して万能なやり方ではなく、**欲求を満たしながら前に進んでいくスタイルでしかない。スタイルの一つでしかないのだ。**ビジーへの一辺倒を食い止めるためには、ビジーも一要素でしかないことを認識しなければならない。

役割と働き方を分離する

役割に半ば自動的にひっついてくるものがある。働き方だ。役割の果たし方など本人のやりやすいようにすればいいはずなのに、なぜか果たし方まで指定されることが多い。その筆頭が働き方であり、マネージャーの役割を求めているのになぜか「出社が必須」などと指定されたりする。

一般化すると、労働 = 役割 + 働き方と言えるだろう。労働という概念は構造として契約的であり、したがって管理が付き物なので、労働の目線で働き方の指定を廃するのは難しい。しかし役割は別である。役割のやりとりにおいて、働き方は要らないはずだ。働き方の整合自体が要らないと言っているのではない。**役割の** やりとりにおいては要らないのだと言いたい。役割そのものを扱って、本質的な議論をするためには不純物はできるだけ取り除かねばならない。デフォルト・リモートもそうで、情報のやりとりとコミュニケーションを分けた上で、前者を突き詰めている。別の例を出すと、アイデアを考える際は、実現性や採算性はまず置いといて、純粋にアイデアを出し合ったり、そのために批判や否定を禁止したり個人的な意見も積極的に出したりするだろう。余談だが、こういう姿勢をアサーティブ・コミュニケーションと呼んだりする。

まずは役割そのものを扱う。必要そうな役割を言語化して、提示して、議論して、修正していく。実行を考えるのはその後でいい。働き方もそうだ。働き方は手段の話であり、実行のフェーズで考えるべきものである。その前段階に持ち込んでいいものではない。あるいは、持ち込んでもいいが、役割そのものの議論という本質を妨害してはいけない。

この分離の考え方を使えば、働き方自体も役割でカバーできるようになる。すでにマネジメントの章にて、自律者と他律者を連携する方法を紹介した。具体的には、仲介として自律寄りの他律者と他律寄りの自律者という二つの役割を設定した。こうすれば自律者も他律者も共存できる。もし分離を知らないと、どちらか片方に寄せられてしまう。リモートと出社で言えば「毎日出社せよ」などとなりかねないし、現になっている。ハイブリッドワークも同じで、どちらに寄せるかというモードを切り替えているにすぎない。その意味では「毎日出社せよ」と大差無い。ハイブリッドワークだからといって高みの見物を決めている場合ではなく、むしろ同じ穴の狢なのである。

役割分担を促すために

従来の働き方は他律的かつ拘束的であり、コミュニケーションを重視する形で臨機応変に対応するしかなかった。あるいはその機会すら与えず、トップダウンでタスクとルールをプロセスを決めて、有無を言わさず隷属させられていた。いずれにせよ、何でも屋として泥臭く立ち回る要領が不可欠であった。

デフォルト・リモートは違う。構造的な限界が生じる従来のやり方そのものから脱するために、リモートこそがあるべき姿だと据えて、必要なものを導入してきた。特に本章では役割そのものを扱う目線に立てと書いた。なぜかというと、役割は動的な共通言語として使いやすいからである。

デフォルト・リモートでは拘束を極力無くし、自律的に働くことで仕事と組織を成立させようとするが、何も無いと無秩序である。皆が合わせるべき基準は絶対に要る。しかし、安易につくりすぎたり、変更不可能な教典をつくってしまうと、すぐに管理過多となってしまう。そこで動的にする。絶対解ではなく仮説として設定して、ひとまずそのとおりに動いてみる。あとでその結果を

フィードバックしあって、次どうするかを考える。これを当たり前のように使えるようにして、必要に応じて素早く回す。フィードバックの話やアジャイルの話はすでに述べた。同じことを役割にも適用するだけだ。別の言い方をすると、自分たちが使う「役割」なるものを、受動的にただただ消費するのはやめて、自らつくったり変えたりする。

というわけで、本節では役割とは何かを見ていく。その後、役割そのもののやりとりを上手くやるためのテクニックとして、表明とロールプレイを取り上げる。

役割とは

デフォルト・リモートにおける **役割（Role）** とは、「何かをする人」を指す。**「何か」は自由に定義していい**。抽象的にもできるし、具体的にもできる。重要なのは第三者が見てもわかるように言語化することと、具体的に誰かを充てることである。しかしタスクではないので指揮命令系統や管理は存在しないし、仕事に直接関係しない役割でもいい。たとえば「雑談チャンネルに愛猫の写真を投稿する人」もありえる。役割は要らなくなれば消せばいいし、過不足があれば変えてもいい。大きすぎる役割を細分化したり、逆に小さすぎて扱いづらい役割を束ねることもできる。

役割をこのように捉えることで、役割を使ったやりとりを行えるようになる。誰でも自由につくったり消したり、割り当てたり外したりできるので融通も利く。無論、組織やチームとして必要な役割はタスクも含めてしっかりと定義し、メンテナンスするべき（静的な役割）だが、それだけではない。そうすることしかできないのと、そうすることもできるのとでは全然違う。デフォルト・リモートでは後者である。ひとりひとりが役割そのものを扱えるようになる。特に必要に応じて新しくつくったり、少し修正したり、あるいは要らないものを消したりできる。他の人がつくった役割に乗っかってもいいし、気に入らない部分があるならそこだけ直した修正版をつくってそれを被ればいい。役割は元来静的なものとみなされてきたが、そうではなく、もっと臨機応変に扱ってもいい（動的な役割）。

動的な役割を導入することで、役割が共通言語になる。一方的なトップダウンでもなく、泥臭い何でも屋（とそれを支えるための大量のコミュニケーション）でもない、第三のあり方が可能となる。必要な議論は役割を使って行う。ここは要らないからカットできるよねとか、これとこれが足りないんじゃないとか、□さんの役割が多くてこれ以上は厳しそうなど、現実的な議論を、言語化・可視化させた役割をベースにして行えるようになる、というより行いやすくなる。

もう一つのメリットは、ツールで扱いやすいことだ。マネジメントの章にて、デフォルト・リモートの実現にはマネジメントレスが必要で、そのためにトピック指向が使えると説いた。役割もトピックとして扱えばいい。よほど機密性やプライバシーの強い役割以外は、全部公開する。これで全社員に見てもらえるようになり、非同期的なコラボレーションが生まれやすくなる。各役割に関する情報や議論はそのトピックの中に書き込めるし、タスク管理がしたいなら、そういう機能や工夫を別途使えばいい。すべての役割がトピックとして保存されて誰でもアクセスできることが重要だ。せっかくの役割も、情報ととして残せなければ意味がない。デフォルト・リモートでは非同期的な情報共有を行うのだった。いくら役割が共通言語化していても、実際に顔を合わせて会

話しないとわかりません、では意味がない。

ここでティール組織の一種「ホラクラシー」を思い浮かべるかもしれない。本質的には同じだが、デフォルト・リモートの方がもっと雑である。あるいは、仕事に直接関係しない役割を勝手につくってもいいし、現時点で誰も共感していないが仕事上重要だと思うテーマを掲げて、それを検討する役割をつくってしまってもいい。好き勝手にやればいい。

役割という名の仮説検証 と言ってもいいだろう。とりあえず役割をつくって、全員に見てもらえるようにしておきつつ、必要なら個別に強調したり、日々のコミュニケーションやその他議論で取り上げたりする。必要な役割なら残るし、不要な役割なら外れる。でも誰かが被ってくれるかもしれないし、逆に誰も被らず放置されるかもしれない。情報はいつどこで誰の役に立つかわからないが、それは役割も同じである。もちろん、現実的にはチームやプロジェクトの一員として働くことになるわけで、自分の役割≒チームやプロジェクトが求める役割、となってしまうことも多いが、それでもそれしかできないのと、他の役割も被れるポテンシャルがあるのとでは全然違う。デフォルト・リモートにおいては、役割は常に開かれている。

表明

役割のやりとりで難しいのは、今抱えている役割を外すことである。仕事術や生活術の文脈でも「捨てる」ことは主題にして難題だが、役割においても同様だ。私たち現代人は豊かだし、一部の経営者のように命を懸けて意思決定を続けるシチュエーション下でもあるまい。捨てることに慣れていない。特に日本は八百万の神や付喪神など宗教的に物を大切にする価値観が染み付いているし、文化的にも関係性を重視するため、なおさらだ。この捨てる素養は本書では割愛するとして、代わりに、要らない旨をはっきり主張する取り組みを紹介する。

表明 とは、ある役割に対するスタンス（特に直近取り組めない旨）をはっきりと主張することをいう。たとえば 6 人チームで、あるテーマで議論をしているとき、通常は 6 人全員で取り組むことが暗黙の了解となりがちだ。一方で、実際には一部の人だけが参加していたりする。ならばそうすればいい。一部の人だけが参加できる状態にするために、参加しない人は参加しない旨を表明すればいい。

表明時に主張したいのは、以下の二点である。

- 当事者性
- 取り組まない理由

当事者性については、意思決定者（意思決定権≒責任を持つ者）、当事者（積極的に取り組む者）、助言者（意思決定権はなく当事者でもないが情報は提供する者）、傍観者（見に来たり情報提供したりするかもしれないが基本何もしない人）のうち、どれであるかを示す。この四分類は目安であり、使いづらいならもう少し細分化してもいい。たとえば部外者（何もしない人）を追加すれば、傍観者よりも強く無関心を示せる。特に重要なのは **傍観者未満の、関与しない側の立場を示すこと** である。私たちは人間であり、無限には抱えられないので取

捨選択は絶対に要る。また組織でもあるので連携も要る。何でも屋から脱して、上手く分担するためには、何をやらないかを議論できねばならない。そのために表明を使う。

ちなみに当事者性を表明していない状態は「未定」であり、当事者にできるか打診してもいいルールにする。つまり、打診されたくなければ表明しておかなければならない。もちろん一度表明したら未来永劫関わらなくて済むかというと、そんなことはなくて、傍観者だと言っているのに当事者や助言者になってくれないかと言われる or 自分が誰かに言うこともあるだろう。それでも、普段から表明を尊重することで、取捨選択が文化的に進むようになるし、そういうイレギュラー自体も減らしていける。

次に「取り組まない理由」については、積極的に出す、必要に応じて出す、基本的に隠す、の三つの運用がある。まずは隠すのが良い。有給休暇を取るときにいちいち理由を言わないし、言う必要がないのと一緒である。理由は人それぞれだし、下手に公開してもそりが合わずにぎすぎすするだけなので、隠してしまえばいい。ある役割で傍観者を表明して断ったのなら、別の役割で貢献すればいい。持ちつ持たれつだ。

しかし、隠すだけだと表明に味をしめたサボりが生じる。ここでサボりとは表明が明らかに多く、かつ別の役割で貢献している形跡もない状態を指す。細かい判定ラインはここでは取り上げないが、サボりは組織を腐らせるため放置してはならない。取り組まない理由を出してもらうべきである。作文されると議論や歩み寄りが難しくなるので、正直に出してもらうのが良い。たとえば「だるい」「興味がない」「向いてない」「□さんと絡まないといけないのが嫌だ」などでもいい。端的に使えるのは、文字数を少なくすることだ。100 文字以内とか、50 文字以内、30 文字以内など大胆に絞ることで、本質的な理由が出やすくなる。理由は正直に出してもらった方が、じゃあどうすればいいかの議論もしやすい。社員全員向けの公開はハードルが高いだろうから、公開範囲は絞った方がいいだろう（特に人との相性の話）。この運用はサボりの抑止力にもなる。また、サボりに対処する役割をつくるのもいいだろう。

ちなみに、サボりでなくとも、取り組まない理由を出してもいい。特に自分の向き不向きや状況を知ってもらった方が、人材としても扱いやすいので、むしろ推奨したい。従来はコミュニケーションの形でお互いを知れていたので何となく知れていたが、デフォルト・リモートではコミュニケーションはオプションだ、知りづらい。だからこそ、自ら積極的に表明をして知ってもらった方が上手くいく。とはいえ、いちいち理由を出すのは純粋に大変なので、基本的には隠す（出さない）でいい。動的な役割を運用すると、何十何百何千という役割が出てきて、日々つけたり外したりするのが当たり前なので、いちいち理由なんて書いていられまい。

ロールプレイ

私たちは多かれ少なかれ演技をしている。演技というと俳優を浮かべるかもしれないが、これに限らない。たとえば会社の上司は、組織の一員としてルールや文化に則った行動をするだろう。公人や軍人はさらにわかりやすい。逆に友人や家族にしか見せない顔もあるはずだ。あえて被らない人もゼロではないが、たいてい私たちは状況に応じた仮面を被っている。役割分担を促すには、このような「仮面」を意図的に設計して使いこなせば良い。これを **ロールプレイ** と呼

ぶ。

ロールプレイのメリットは、素の自分を使わずに済むことである。仮面が規定したとおりに動けばいい（演じればいい）のでやりやすいし、言動に伴う責任も仮面に押し付けることができる。もちろん演者は存在するわけで、演者を完全に隠蔽することはできないが、それでも素の自分よりは行動しやすい。また、ツールを使えば演者の隠蔽も可能である。

同じ議論はすでにしている。コミュニケーションと情報共有の章にて扱った「アイデンティティ」だ。本名、性別、容姿をアイデンティティとした上で、そもそも素のアイデンティティだけを使う必要があるのか、仮想のアイデンティティ（マスク）を被ってそれで仕事してもいいのではないかと提起し、これをマスクド・アイデンティティと呼んだ。ハンドルネームやプロフィール写真は馴染み深いが、その程度ではない。必要に応じて仮面をつくって、被ってから行動するのである。

たとえばシャルロッテというお嬢様言葉を使うキャラをつくったとする。無邪気な子どものように容赦ないコメントを出し、ですわ口調を使う。

charlotte

このキャラなら素のアイデンティティでは言えないことでも言いやすいし、角も立ちにくい。そうでなくとも、YouTube を見る人ならずんだもんやゆっくり霊夢はご存知かと思う。何ならザ・ファブルの「佐藤明」や葬送のフリーレンの「フリーレン」を演じてもいい。もっとも演者は存在するわけで、ある種茶番めいてはいるものの、茶番の力を舐めてはいけない。ロールプレイをしてもいいとの合意があれば、素のアイデンティティよりもはるかに立ち回りやすくなる。ロールプレイを自然に多用できるようになった状態を **浸透** と呼ぶ。演劇の世界でもなければ、仕事や会社において浸透まで導くのは難しそうだが、日常生活ではエンタメやフィクションとして定着しているし、部分的には冗談やユーモアの形で使われている。

ひとまずのハードルは気恥ずかしさであろう。このハードルを落とす方法は二つある。

一つは、演者は隠すことだ。ツールや仕組みの力を使えばいい。わかりやすいのは、アカウントとしてシャルロッテを発行することである。この仮想人格シャルロッテは、誰かが専任して操作してもいいし、空いているときに誰でも使えるようにしてもいい。ビジネスチャットのアカウントとしてつくる程度もあれば、文字通り社員ごとに一つだけ付与される「社員アカウント」としてつくることもできなくはない。

もう一つは、チーム内など狭い範囲で使ってみて慣れていくことだ。会議室を思い浮かべてほしい。会議室は誰でも使えるが、ロールも同じである。シャルロッテの着ぐるみが用意されていて、余っているなら誰でも使えるイメージ。これを **リソースとしてのロール（Role As A Resource）** と呼ぶ。再三強調するが、仮面は必要に応じて使ってもいい。使えるようにしておくためには、事前の用意と、使ってもいいことの合意が必要である。仮面は概念的なものなので、個数に上限はないが、適当に個数を定めておいた方が都合が良い。

たとえば 6 人チームにおいて、シャルロッテを 2 つまで、としておくと、最大 2 人分のですわ口調

が出現する可能性がある。最大 2 人なので、安易にノリとして皆がですわ口調を真似してふざける展開を防ぐことができる。いや、言い方を変えよう。**ふざけてお茶を濁したり誰かを茶化したりする展開は潰さなければならない**。演劇を嗜む人ならわかると思うが、演技は真面目になりきるものである。気恥ずかしさからの逃避行動は、容易に茶化しや濁しにつながるため潰さねばならない。特に茶化すのは論外である(1)。演劇は演技したい人だけが集まる場だから御しやすいが、ここで述べているロールプレイは違う。潰すための、何らかの仕掛けが要る。それが個数の設定である。個数を決めると、いつ誰がそれを使うかというやりとりが生じて、ロールプレイが形式的になる。形式的であれば、ビジネスライクに真面目に運用しやすいだろう。実際、実運用としては、たとえば「率直にコメントしたいので、今からシャルロッテを被りますね」のような光景になる。

まとめると、気恥ずかしさのハードルを緩和するために、非同期的な情報共有ベースでロールプレイする場合は、演者を隠せばいい。同期的なコミュニケーションベースでやるなら、隠しようがないので、仮面をいくつか用意して、それを形式的に使う形で慣れていくのが良い。

ここまでロールプレイを仕事として使う前提で書いたが、コミュニケーションの注入用途でも使える。素のアイデンティティで雑談せよ、といわれても関係性次第ではやりづらいが、何らかのキャラクターを被って演じよ、であればやりやすい。といっても即興だと素人では難しいので、軽いシナリオと仮面を用意して、そのとおりに演じて遊ぶ(見世物ではないので雑でいいしアドリブでもいい)のがいいだろう。その用意もおそらく素人には難しいので、できれば演劇に詳しい者――俳優やその卵、あるいは鑑賞者として詳しい者などに「演劇タイム」の設計と啓蒙を担ってほしいところではある。つまり社員たちは、コミュニケーションの充足として演劇を使うようになる。もちろん万人向けではないし、慣れるまでは気恥ずかしいし、(社歌を歌う等と同様だが)下手に推奨するとそれだけでブラック企業臭がしてしまうが、雑談の偶発性に漠然とすがるよりは建設的だと思う。別に演者は隠してもいい。たとえばボイスチェンジャーと仮面を多数用意して、好きなものを使って会話するプラットフォームをつくれば、素のアイデンティティを隠しながらも生き生きとした掛け合いができるだろう。

ワーカーを補う

何でも屋のあり方はすでに述べたとおりだが、もう一つ無視できないものがある――ワーカー(Worker)の重視だ。直接作業・間接作業でいう直接、フロントオフィス・バックオフィスでいうフロントなど、私たちは通常あらかじめ決められたタスクをこなす存在として働かされる。新規事業や改善といった投資的な活動は、タスクとして正式に決まらない限りは眼中にないし、勝手に行うことすら許されないことも多い。早い話、現場にいるメンバーは、その現場の仕事をしなければならない。このような典型的な仕事を行う役割を **ワーカー** と呼びたい。

ワーカーへの偏執は、デフォルト・リモートでは悪手だ。ワーカーを過剰に重視するとワーカー以外のあり方が許容されなくなり、デフォルト・リモートも目指せない。ワーカーは数ある役割の一つにすぎないし、ワーカーだけやってればいいように思えるのは、単に管理側・経営側がそう仕

向けているからにすぎない。仕事に正解はない。ワーカーへの一辺倒をやめたあり方にシフトすればいい。デフォルト・リモートならまさに可能である。

さて、本章冒頭でもテネットを解体してきたが、ここでも同じことをする。ワーカーを重視するテネットを解体し、代わりにどのようにカバーしていくかを見ていく。

兵農分離

ワーカーは、兵士と農民でいうと兵士のようなものである。従来、兵士は平時には農作業も行い、有事の際には兵隊として働いていたようだ。これに異を唱えたのが織田信長で、兵士と農民を専業させることで兵士の質を底上げしたとされる。兵農分離と呼ばれている。兵農分離というと、豊臣秀吉の刀狩のように武士階級の支配を強化する側面もあったようだが、ここでは統治ではなく専業化の意図としてとらえる。

逆に新規事業や改善、その他直接的な仕事とは直結しないような会話、対話、検討などは農作業のようなものである。誰かがやらねばならない。短期的にはやらなくても生きていけるが、中長期的には首が締まってしまう。だから兵農分離の以前は、兵士自らが兼務して行うしかなかった。同じことが現代でも起きている。こういった活動もワーカー自身にやらされているのはご承知のとおりである。現場から上がった管理職が、現場の仕事もこなしつつ管理もするという「プレイングマネージャー」はよく知られている。あるいはマネジメントの領域はプロジェクト、プロダクト、チーム、ピープルなど多数存在し(2)、それぞれ専門性を有するものだが、ひとりのマネージャーが全部負ってしまっているケースも多いと思う。

現代版の兵農分離が必要であろう。ワーカーとそれ以外を分けて、ワーカーには典型的な仕事に集中してもらうのである。かわりにそれ以外の活動は免除して、ワーカー以外の役割が担ってもらえばよい。ただ、これだけだとお互いがコラボレーションできないから、間を繋ぐ連携役も要る。

以降では、この現代版・兵農分離を進めるための考え方やその構成要素を紹介していく。

メタワーカーと連携

兵農分離を実現するにあたって、兵士側の概念(ワーカー)が定義できたが、農民側の概念(メタワーカー)がまだできていないため準備しよう。また兵士と農民を繋ぐ役(連携役)も必要なので、併せて論じる。

ワーカーは兵士的であり、典型的な仕事をこなすが、そうじゃない間接的な仕事や投資的な取り組みもありうる。これをメタワークと呼びたい。仕事のための仕事とでも言えよう。メタワークを行う者を **メタワーカー** と呼ぶ。従来メタワークは、バックオフィスなど正式に整備されたもの以外は、ワーカーが行うものだった。現代版・兵農分離では、これを分ける。ワーカーはワークに集中し、メタワークの部分は、別途役割をつくって担わせる――これがメタワーカーである。

つまり、従来以下のようになっていたものは、

> ワーカー = ワーク + メタワーク

役割を分けて、以下のようになる。

> ワーカー = ワーク
>
> メタワーカー = メタワーク

両者を繋ぐ連携役も必要だから追加しよう。結果として、以下のようになる。

> ワーカー = ワーク
>
> 連携役 = ワーカー-メタワーカー間を連携
>
> メタワーカー = メタワーク

メタワークをワーカーが兼ねていたのと同様、連携役もワーカーが兼ねがちであるため分けるのである。何度もいうように、ワーカーはワークに専念してもらう。連携は連携の役割が担えばいい。ということは、連携役はワークもメタワークも抱えないことを意味するわけで、サボりではないかと思われるかもしれないが、そんなことはない。連携するためにはワーク、メタワークの双方の理解が必要であり、そのための活動に腐心することになるからだ。しかし、ワークやメタワークの担当者として責務を負っているわけではないから、連携に専念できる。

連携役

連携役が行うことは、ワーカーの文脈をメタワーカーに伝えることと、メタワーカーの成果をワーカーに伝えることの二つである。メタワーカーの成果物をワーカーが使うことで、より仕事が上手くいくようになるわけだが、そのためには使ってもらわねばならない。ワーカーは仕事に専念しているのでその気はない。というわけで、連携役がカバーしなければならない。そのためワーカーは、仕事に専念と言いつつも連携役との連携は必要になる。

逆にメタワーカーも、ワーカーの文脈なしにメタワークをしたところで、見当違いの成果物にしかならない可能性が高い。加えて、つくった成果物を使ってみてどうだったかのフィードバックも欲しい。だがワーカーは仕事に専念しているので、そんなことには費やしたくない。ここも連携役がカバーすることになる。ワーカーとやり取りして得た「ワーカーの文脈」を、メタワーカーに伝えるのである。

伝え方は様々だが、大まかには、ワーカーと一時的に一緒になって働くことで体感的に知る（スタッフ）、一緒に働きはしないが日々のコミュニケーションや情報共有を上手く差し込んでワーカー側の情報を吸い上げる（スパイ）、あるいはワーカー自身に情報を出してもらうための仕組みや場をつくってワーカーに参加してもらうよう巻き込んでいく（システム）がある。どれが最適かは状況次第だし、正解もないので、連携役本人がやりやすいようにすればいい。

気をつけるべきはスタッフである。従来のテネットとして「一緒になって汗水流してようやく認めてもらう」価値観があるが、こんなものはくそくらえである。これで上手くいくのは当たり前だ。もっと賢く、早く上手くいくためにはどうすればいいか、が本章の主題であり、本書のテーマでもある。しつこいようだが、デフォルト・リモートは我々の水準を第一に考える。前時代的な拘束ありきの過ごし方に頼りすぎてはいけない、が本書の立場である。したがってスタッフの乱用も控えた方がいい。むしろ、スタッフを使わずに連携を行えるようにしていきたいし、それこそが連携役の責務とさえ言ってもいい。とはいえ、現実的にスタッフに頼る比重は重くなりがちなので、やはり連携役はスタッフが使える人にこそ向いている。

メタワーカー

ワーカーと連携役について見てきたが、ではメタワーカーとは何だろうか。仕事のための仕事とは何なのか。

それは本書冒頭から論じてきたテネット、やり方や考え方、あるいは道具を何とかすることである。合わない部分を変えたり、機能していない部分を直したり、思い切って捨てたり、もちろん新しいものを導入したり、考え直すところから始めることもある。コミュニケーションと情報共有の章にて「木こりのジレンマ」を取り上げた。斧で木を切ることだけが仕事なのではない。木を切るための斧をメンテナンスしたり、買い替えたり、あるいは複数人で分業しているならどうやれば最適化できるのかとか、単に木こり各々の調子を整えたりすることもすべて含まれる。

抽象的に言えば、私たちには決め事はあれど正解など何ひとつないし、私たちは人間であり生物的な限界を踏まえて立ち回らねば壊れてしまう or 壊れないまでも不調が続いてポンコツになる。一方で、私たちは学んで成長できる生き物もであるので、勉強や鍛錬といった営みも有効だし、そもそも現代はデフォルト・リモートなる第三パラダイムであって、私たちはもっと自分の人生を尊重してもいい。ワークだけがすべてではないのである。ならば、ワークを支えるための諸活動にもフォーカスがあたることは極めて自然だ。無論、それでワークがおろそかになっては本末転倒であるが、だからといってワークへの一辺倒では現代の限界を越えられないのである。メタワーカーも必要なのだ。

もう一度、木こりを例にしよう。5 人の木こりと 1 人のマネージャーから成るチームだとする。従来の働き方だと、おそらくマネージャーを除いた 5 人は全員がワーカーになるだろう。日々マネージャーに管理監督されながら、ひたすら木を切るはずだ。ここにメタワークの考えを入れると、たとえば木こりは 4 人、メタワーカーが 1 人となる。このメタワーカーは、4 人が持続的にエンゲージメントを高めるためのメタワークに専念する。エンゲージメントとは「企業の利益」と「個人の満足」の均衡だと思えばいい。前者が過ぎると従業員が死ぬし、後者が過ぎると事業が成立しないため両者の均衡を取らねばならない。

現代はまだまだ前者に寄りがちだ（そしてそれをビジョンだのコミュニケーションだのと麻痺させて誤魔化している）。メタワーカーは、後者に寄せていく存在と言い換えてもいい。具体的に何を行うかは文脈次第だが、斧マニアとなって斧の知識・技術・運用を教える役割、雑務をすべて

引き受ける役割、あるいは集中しすぎなメンバー達を休ませるタイムキーパーやファシリテーター的役割かもしれないし、これらを二つ以上兼ねるかもしれない。いずれにせよ、従来だと「いいからお前も木を切れ」一択となってしまうし、斧の扱いや雑務や休憩も「各自で上手くやれ」になりがち（そしてワークで忙しいのでおろそかになる）だが、そうではなく、メタワーカーに専任させるのである。

メタワークをワークとして行うな

人事や総務などバックオフィスの形で体系化されたジャンルがある。これらはメタワークではなくワークの範疇と言える。有用なメタワークは（直接的な活動でなくとも）ワークと同等の扱いになることがある。人事や総務といった鉄板は、言うなれば昔の時点でワークと化した元メタワークだ。

このようにメタワークが正式に、あるいは本格的に採用されてワーク化すること **昇格** と呼ぶ。非常に重要なことだが、**昇格したメタワークはメタワークではない**。うちはバックオフィスがあるからメタワークもできてますよ、は正しくない。バックオフィスもワークである。

ではワークとメタワークの違いとは何か。同じ活動であっても、ワークになったりメタワークになったりするのはなぜか。何の有無や程度で決まるのか。それはタスクの程度である。ここでタスクとは「やらないといけないこと」を指す。もちろん絶対的な正解などない。やらないといけないとは、誰かがそう決めたということである。会社のトップかもしれないし、顧客かもしれないし、上司かもしれないし自己判断かもしれない。チームで決めたことかもしれない。要は意思決定して生まれた「やると決めたこと」だ。タスクはその性質上、管理の色が強くなる。進捗や評価といった概念が登場する。ベストエフォートでは済まないことが多く、クリアの基準が（暗黙的な場合も含めて）設定されており、クリアするまで終われないか、何なら拘束が続いたりもする。確実に遂行するために、厳しい献身が要求されがちだ。**ワークとはタスクと付き合うこと** を指す。逆に、メタワークではタスクを扱わないか、扱っても一時的に少しだけである。

つまり**メタワーカーはタスクを負わない**。もしタスクを負っているメタワーカーがいるのだとしたら、それはワーカーであってメタワーカーではない。メタワークをワークとして確実に遂行すると決めること自体は構わないし、推進して遂行する段階ではそうした方がいいが、まずは分けて考えてほしい。

- 以下が望ましい
  - 1: ワークを、ワークとして（タスクを負う形で）行う
  - 2: メタワークを、メタワークとして（タスクを負わない形で）行う
- **以下は望ましくない**
  - 3: ワークを、メタワークとして行う
  - 4: **メタワークを、ワークとして行う**

とにかく 4 を回避したいのである。メタワーカーはタスクを負わず、その身軽さと自由さをもって創造的に取り組むものなのだ。タスクを課して邪魔するなどあってはならない。そう、メタワークは創

造的で探究的なものだ。管理的で実践的なワークとは世界観が違う。

もう一度言う。メタワーカーはタスクを負わない。だからといって、いたずらにサボっていいわけではない。自分なりに自分のペースで頑張ればいいが、ベストエフォートは義務である。ワーカー達がお金や時間を費やしてもいいと思える程度の期待と、できれば成果を出していかなくてはならない。そのためには本書で散々強調している自律的な行動と、非同期的な情報共有が必須となる。つまりデフォルト・リモートが必要である。これも繰り返しになるが、ワーカーは構造上、他律的で拘束の多いスタイルになる。ここにメタワークの営みを差し込める余地はない。だからこそワーカーとは独立して行動して、成果を出して、それを持ち込む形となっている。

様々な兵農分離モデル

兵農分離の考え方に正解はない。他にも雑多に紹介しておく。

ワーカー・トライアングル

ここまでワーカー、連携役、メタワーカーの三役のモデルを述べてきたが、これを **ワーカー・トライアングル** と呼ぶ。

worker_triangle

典型的な仕事を行うワーカー（Worker）、メタワークを行うアドバイザー（Advisor）、両者の連携やフォローを担うヘルパー（Helper）に分かれている。名前のニュアンスに注目してほしい。

メタワーカーの立ち位置は「アドバイザー（助言者）」となっており、助言のニュアンスであることがわかる。本章では自らテネットややり方や考え方を **つくる** ことも想定していたが、既存の情報――論文や書籍に頼って、こういうものがありますよと勧めるスタイルでもいい。個人的には、そういった権威的な情報も補強としては大事だが、メタワーク自体は借り物よりも自分なりに考えて言語化したものの方が役に立つと考える。そもそも仕事のやり方や考え方といったものは本質的に個人的なものであり、大半は「誰もが薄々知っている」か「知ればすぐに理解できる」かんたんなものであるため、権威性はさておいて、情報量を出すことが大事だ。自分の情報として出すことを恥ずかしがって権威にすがるのは、心理としてはわかるが、メタワーカーとしては望ましくない。助言に責任を負うコンサルやカウンセラーなら権威性も大事（ちなみに責任を負う≒タスクも負っているのでワーカーである）だが、メタワーカーは違う。負わなくていいし、負ってはならない。だからこそ権威性にとらわれず、自分の主観と創造性をフルに発揮して全部出せばよい。**本質的に個人的なものであり、何が刺さるかなんて誰にも（本人にさえも）わからないのだから、とにかく物量を出して、たくさん議論をすることを目指したい。そうすれば「刺さるものと出会う確率」を最大化できるし、そういう場は「面白い」ので人も集まって、より盛り上がる。**

と、少し話が逸れたが、要は何を助言するかの「何」の部分は自由だ。つくってもいいし、借り

物でもいい。ただ筆者としては、前者の比重を重くした方が良いのでは、とのお節介を述べただけだ。

もう一つ、連携役の立ち位置は「ヘルパー（支援者）」となっており、支援のニュアンスだとわかる。本章ではここまでワーカー → メタワーカーへの文脈の伝達と、メタワーカー → ワーカーへの成果の伝達の両方をやるのだと書いてきたが、これは支援のごく限定的なあり方にすぎない。つまり、メタワークによりつくった成果をワーカーに使ってもらうことこそが支援、との前提に立っている。もちろんこれ以外の支援もある。ピープルマネジメント的な「継続的な対話」は典型例だろう。従来はワーカーのマネージャーが担っていたが、ヘルパーがやればいい。第三者として行うのにもいいし、ワーカー側の文脈も勉強した上で濃密な議論をしてもいいし、一部のコンサル企業が取り入れているような「キャリアアドバイザー」的な立ち位置でキャリアに関する壁打ち役となってもいい。やりようはいくらもであるが、ワーカーやヘルパーが望む（潜在的に望むことも含む）支援を行うのが良い。そういう意味で、ワーカーとメタワーカーを顧客としたビジネスと言えなくもない。

ミドルオフィス

現状の会社組織はバックとフロント、二種類のオフィスに分類している。バックオフィスとしては人事、経理、法務や総務や監査、その他情シスなどがあり、歴史的に昇格したジャンルだけが厳選されている。これ以外はすべてフロントオフィス（という言い方はしないが本項では使わせてもらう）の扱いとなる。フロントというと、顧客と直接関わる営業や問い合わせ担当を浮かべがちだが、それだけではなく、オフィスワーカー全般を含むし、研究や経営さえも含む。本質的にはどちらもワークであり、以下の構図になっていると思う。

- (ワーク)
  - (フロントオフィス)
    - 経営
    - 顧客と直接やりとりする系
    - 顧客と直接やりとりしない系
  - (バックオフィス)
    - リソースを管理する系（人事や経理など）
    - ガバナンスを運用する系（法務や監査など）
    - システムやツールを管理する系（情シスなど）

構造的にワークの域を脱せていない。いくらワークの中で役割分担を押し進めたところで、（もちろんワークの最適化はできるものの）意味がないのである。メタワークの目線も必要であり、取り入れるためには、ワークの域を出たオフィスを設置せねばならない。これを**ミドルオフィス**と名付けたい。金融業界で用いられる用語のようだが、ここでは無視してほしい。位置づけとしては、次のようになる。

- (ワーク)
  - (フロントオフィス)

        - 経営
        - 顧客と直接やりとりする系
        - 顧客と直接やりとりしない系
    - （バックオフィス）
        - リソースを管理する系（人事や経理など）
        - ガバナンスを運用する系（法務や監査など）
        - システムやツールを管理する系（情シスなど）
- （メタワーク・連携役）
    - （ミドルオフィス）
        - …

メタワークや連携役を担うためのオフィスである。ただし、ワーカー・トライアングルと違うのは、何と何を繋ぐかという話だ。ワーカー・トライアングルではワーカーとメタワーカーを繋いでいたが、ミドルオフィスではフロントオフィスとバックオフィスを繋ぐ――というと語弊があるので、言い方を変える。**ミドルオフィスとは、フロントオフィスとバックオフィスを含むワーカー社員全員を短期的に助ける、あるいは中長期的に導くための機能である。**従来はバックの一部がフロントを助ける構図だったが、ミドルオフィスは違う。ワーカー同士の助け合いに加えて、メタワーカーや連携役がワーカーを助ける部分も追加する。

たとえば従業員 500 人の会社がいるとして、デフォルト・リモートを押し進めたいとしよう。おそらく従来であれば、経営者がトップダウンでやれと下して丸投げするか、バックオフィスとして推進部隊をつくるか、フロントオフィスのメンツで委員会的な集まりを立ち上げて進めていくだろう。**いずれにせよワーカーの理で動いているので、本質的な検討はできない。**一方、ミドルオフィスがこれに絡むとしたら、メタワーカーや連携役の人材があたることになる。ワークは負っていないので、時間や余裕はたっぷりある。濃密な検討が可能だ。その上で、その成果をワーカーに届ければいい。そこで得たフィードバックを持ち帰って、また検討する。もちろん非同期的な情報共有を行って、ワーカーに対しても透明性を保てばなお良い。文字どおりの IT リテラシーを持たない社員がいるなら、教育の場をつくればいい。片手間ではなく、腰を据えて検討していける。

ここで「検討」と表現していることに注意してほしい。推進するのはあくまでワーカー達であり、ワークとしてやらねば通用しないことが多い。しかし、最初からワークとして取り組むと、従来のテネットややり方考え方から逸脱できないし、成果も急ぎがちでしょうもないことしかできない。だからミドルオフィスなのだ。忙しいワーカー達に変わって、しっかりと検討して、成果を残す。この成果は、ワーカーだけでは辿り着けないものだ。この成果を踏まえてどうするかはワーカー達が（特に意思決定者たちが）判断すればいい。ミドルオフィスがあると、判断に使える「質の高い材料」をつくれるようになる。

二層仲介

立場Aと立場Bがあり、この二つだけでは相容れないため、間に仲介役をかます――との考え方はすでに述べてきた。

- 非同期的な情報共有と、同期的なコミュニケーション
- 自律者と、他律者
- ワーカーと、メタワーカー

仲介役というと、AとBを繋ぐCを考えがちだが、実は「A寄りの仲介者」と「B寄りの仲介者」の二つを考えた方が上手くいきやすい。実際にマネジメントの章では、自律者と他律者の間として「他律もできないことはない自律者」と「自律もできないことはない他律者」を設定した。この仲介のあり方を **二層仲介** と呼びたい。

二層仲介では、「A寄りの仲介者」がBにAを伝える。A寄りということはAを知っているが、仲介者なのでBも知っており仲介ができるわけだ。逆に「B寄りの仲介者」はAにBを伝える。B寄りなのでBを知っているが、仲介者なのでAも知っており仲介ができる。二層仲介のメリットは、各立場から仲介者を出しやすいことだ。「A寄りの仲介者」は立場Aから出しやすく、「B寄りの仲介者」は立場Bから出しやすい。立場AとBがあるからといって、**片方の立場に強く偏っているということは実はあまりない**。出社とリモートの議論でも、年に一度も出社したくないフルリモート勢はそうはいないだろう。たいていはどちらかに寄っているだけだ。この傾向を踏まえると、二層仲介は自然に使いやすい。もちろん、普段の業務があるのに仲介も行う、となると大変だし、業務のせいで仲介活動がおろそかになることは目に見えているので、仲介者の業務はある程度免除しなければならない。ちなみに、普段の業務（ワーク）を完全免除して、仲介に専任させたのがワーカー・トライアングルの連携役（より正確に言えばヘルパー）である。

もう一つ、二層という名前から「A寄りの仲介者」と「B寄りの仲介者」間でやりとりする構図を浮かべるかもしれないが、違う。やってもいいが、コミュニケーションの経路が無駄に増えてコストがかかるので、「A寄りの仲介者」が直にB派とやりとりするのが良い（あるいはB寄りの仲介者がA派と直にやりとり）。

二層仲介の例を一つ紹介しておこう。

これは情報のやりとりを「読み書き」で行う立場と、「聞き話し」で行う立場があるとして、これだけでは相容れないという前提で、仲介を入れたものである。デフォルト・リモートでまさに遭遇する場面だ。非同期的な情報共有は読み書きで行うため、読み書き派が幅を利かせる。一方で、従来からの同期的なコミュニケーションを重視する人達は聞き話し派だ。両者は相容れない。だからこそ、本書でも役割分担の章をこうして設けて、連携役だの二層仲介だのといった概念を導入することに割いている。

さて、二層仲介としては、「読み話し」派と「聞き書き」派を導入する。このあり方を **表現多様性** と呼びたい。つまり立場は以下の4つとなる。

linguistic1

これを使って、以下のような連携を組む。

linguistic2

まず、「1:聞き話し派」と「4:読み書き派」は各自でやりとりする。お互いは直接はかかわらない（もちろんかかわってもいい）。ここを二層仲介で仲介する。聞き話し派のやりとりを「3:聞き書き」派が聞いた上で、読み書き派に書いて伝える。逆に読み書き派のやりとりは「2:読み話し」派が読んだ上で、聞き話し派には話して伝える。また、仲介役の聞き書き派と読み話し派は、お互いにやりとりすることもできる。

いかがだろうか。このように仲介役を二つ設定することで、連携を工夫しやすくなる。

※スプリンターとサステイナーについては軽く解説しておく。スプリンター（Sprinter）は短距離走的に高密度に働く者つまりはワーカーを指し、サステイナー（Sustainer）は中長期的な備えや改善その他フォローに勤しむ者つまりはメタワーカーを指す。聞き話しはスプリンターがやりやすく、読み書きはサステイナーがやりやすいと言っている。もちろん聞き話し派がサステイナーになってもいいし、読み書き派がスプリンターになってもいい。

特区

特区 とは、ワーカーの理が通用しない特殊な組織を指す。シームレスの章で述べた出島戦略（3）は、イノベーションを狙うものであり特区の典型例と言える。

ここまで見てきたように、メタワークをワーカーの理屈で行うのは難しい。大胆に切り離さなくてはならない。たとえば意思決定権やガバナンスをワーカー側の人が握っていてはいけない――となると、実質的にかなり特殊な組織を別途立ち上げることになる。特に大企業のようにルールが厚くて細かい組織だと、**その組織内にいる限りは（ワーカー前提かつ昔に組まれたであろう）組織の理屈に抗えない**。どうするかというと、ルールの適用が緩和される役職者への登用を行ったり、子会社など別に会社を立ち上げたりしている。いずれにせよ、現在ではほとんど確認できないし、できても非常に狭き門である。

逆を言えば、いくら組織内に新しい組織を立ち上げても意味がない――とまでは言わないが、遠回りが過ぎる。どうせ親組織のワーカー的な理に阻まれる。そういうものと受け入れて泥臭く立ち回ることを要求しがちだが、違う。そもそも泥臭さを根本から排除して、メタワークに全振りできる組織をつくらねばならない。そこまでして特区と言える。無論、これは変革の一種であり、すでに述べたとおり投資を要する。特区をつくるだけの投資ができるかどうかに尽きる。

特区の目的は何でもいい。出島戦略はイノベーションを目指すものだが、イノベーションに限らなくてもいい。たとえばデフォルト・リモートを前提するなら、コミュニケーションの注入は間違いなく鍵になるわけで、これをサポートする特区をつくれる。1on1 特区として社員全員はいつでも誰でも特区のメンバーと会話ができるなど。会話が苦手なら、上述したロールプレイを導入してゲームなり掛け合いなりをしてもいい。それこそスプラトゥーンやマインクラフトを遊べる環境を整えて、特区メンバーと遊べるようにしたっていい。これら全部を含めると、

> コミュニケーション特区として 1on1、スプラ、マイクラを提供しており、社員はいつでも誰でも特区メンバーと一緒に遊ぶことができます。

とでもなるだろう。他にも、社内でどこに行っても馴染めない者を集めた「モンスター特区」もありえる。目的はモンスターの利活用を探ることであり、ワーカーの理から切り離して過ごしてもらうことで真価を見いだせるかもしれない。たとえば ASD や ADHD といった発達障害者を集めると、優秀なフィードバック集団として使えるかもしれない。社員らは ASD らしい率直で容赦のないコメントと、ADHD らしい豊富なアイデアと連想を享受できる。どちらも健常者では（心理的安全性が十分された上で濃密に議論でもしない限りは）まず縁のない高密度なフィードバックである――

と、これは大胆な例だし、本書の範疇を越えるためこれ以上は割愛するが、特区の目的は何でもいいわけである。ただし、目的自体は何かしら定めるべきだ。

定めずに自由にやる余地がほしいなら、メタワーカーが生きるミドルオフィスやワーカー・トライアングルの方が良い。ミドルオフィスにせよ、（ワーカー・トライアングルでいう）ヘルパーにせよ、いろんなメタワーカーがいて、イノベーション担当もいれば注入担当もいる。あるいは同じ人でも、あるときはイノベーションのための研究開発を、またあるときは注入担当をしたりもする。誰が何を担当するかが課されることはなく（メタワーカーはタスクを負わない）、常に主体的に取り組まれる。課すのはその後、メタワークの成果物がある程度見えてきて、検討から推進の段階に移るときだ。

一方、特区であれば、イノベーション特区であるならメンバー達はイノベーションのために 100% を注ぐし、注入特区の場合も同様、注入に捧げる。**メタワークとは違って、必要に応じてタスクも課す**。目的が明確ゆえに、推進の段階に入りやすいためだ。そういう意味では、イノベーション特区であるなら「イノベーション特化のワーカー集団」、注入特区であるなら「コミュニケーションの注入に特化したワーカー集団」と言った方が正しい。ただ、既存の組織だと構造的に特化して動きづらいから、動きやすい組織（特区）を別途つくるというだけだ。

このように特区には何らかの目的があるため、社員から見るとわかりやすいし、会社としても投資がしやすく成果も測定しやすい。だからといって管理を無闇に敷くのは論外である。あるいは管理が必要にしても、ワーカーの理ではなく特区のメンバーで主体的かつ段階的に適用していくべきだ。特区を侵食させてはならない。役割の理屈で捉えると、会社という役割 A と、特区という役割 B があるとの構図になる。A と B は双方尊重し、侵食させない前提で両者の連携を考えねばならない。たとえば二層仲介は使える。

(余談) NERG

モンスター特区という表現は過激であったが、この文脈でマイルドな表現をすると、たとえば Neuro ERG となるだろう。

まずERG（従業員リソースグループ）とは、特定のマイノリティな属性が集まった社内グループを指す。個人や草の根的なグループとは違って、会社や全社員への影響を与えるものである。言い方を変えると、正式に存在を認められ、会社の名前で活動や啓蒙が行えるグループとも言える。社外に発信するとは限らない（できれば最低でも会社としてどんなERGを持つかの公開くらいはしたい）が、社内全域への発信はマストである。イメージで言えば、労働組合は一般社員のERGと言えなくもない。現状ではLGBTQ+など典型的な多様性のERGが運用されている。

本題に入ろう。NERGとはNeuro ERGの略であり、ERGのニューロ版である。ニューロとはニューロ・ダイバーシティ(4)を指し、狭義には発達障害の多様性とでも言えるが、広義には神経の多様性となる。発達障害というと大げさだが、私たちは単に知らないだけで、先天的に脳の特性に偏りがある。たとえば筆者は、本書を見てもわかるように、アスリート的なストイックな生活スタイルと粘り強い言語化を持ち味としているし、これくらい造作もないことだが、だからといって誰でもできるかというと違うだろう。スキルや努力の差もあるが、それ以上に特性の違いがある。他にも朝型と夜型があるし、外交的な人もいれば内向的な人もいるし、脳内でイメージを扱える能力にも人によって開きがあるし、脳内で描くことさえできない「アファンタジア」なる性質もあったりする。私は多様性として、こういった特定面も考慮していきたいと思う。発達障害（だと字面が強いので「ニューロ」を使う）は、第一歩として取り組みやすい。ニーズもあるはずだ。

NERGに限らず、ERGを突き詰めたあり方の一つが特区になる。業務の片手間でありきたりな活動をするのではなく、当事者として専任して、存分に調査検討や議論や発信、対話や検証を行っていく。それを可能とする特殊な区域をつくるわけだ。

(余談) ソフトウェアとオルグウェア

役割分担はソフトウェアエンジニアが得意ではないか、という話をする。さらにエンジニアリングの本質が「概念を扱うこと」にあるとした上で、ソフトウェアと同列の営みであるオルグウェアを提唱する。

まず役割分担とは、突き詰めると役割の定義とその連携の設計に等しい。そして連携に持続性や再現性を持たせると、集団としての強度が上がって組織となる。つまり役割分担とは組織の設計にほかならない。

本章の解説は筆者がつくった仮説にすぎない。汎用的につくりこんではいるものの、あらゆる人や組織に適用できるほど万能ではないだろうし、そんなつもりもない。理想を言えば読者自身に担ってもらいたい。自分で、自分の組織に合った役割をつくっていくのである。では、たとえば本章で解説したようなものを誰もがつくれるかというと、そんなことはない。しかし、思っているほど難しいことでもない。

幸いにも、このような営みを得意とする人達がいる。ソフトウェアエンジニアだ。その名のとおり、

ソフトウェアを（つくることも含めて）扱う人達であり、ソフトウェア・エンジニアリング（ソフトウェア工学）の知見をフル活用して、現実の要件をソフトウェアに落とし込む。あるいはすでに動いているソフトウェアの改良やメンテナンスを担う人達である。

さて、ソフトウェアは電子的な世界だ。数学とは別の、デジタルの論理に基づいて構築せねばならない。人間相手のコミュニケーションとは訳が違うし、非言語情報はもちろん、自然言語すら通じない。このギャップを埋めるにはどうすればいいか。最も重要なのは **概念を扱う能力である**。デジタルの論理を踏まえた上で、「こんな感じの概念をつくって」「これとこれをこのように連携させて」「ここがちょっと弱いから補強するために、こういう概念をつくって、こう接続しておいて」のように概念的な操作を行っていく。プログラミングでたとえると、関数やクラスやモジュールをつくるのがまさにそうだし、もっと小さな単位として変数やスコープもある。生成 AI でいうと、情報はトークンやチャンクといった単位で捉えようとする。エンジニアでなくとも、IT に触れる者であれば、日々よくわからない概念と「なんか難しい横文字の用語」の形で接しているはずだ。

ソフトウェアエンジニアは、まさに息するように概念を扱っている。いわば「人間の論理」と「デジタルの論理」を中継するために、上手くいきそうな概念を適宜つくるわけである。仮に **概念的中継** と呼びたい。

さて、概念的中継を要するジャンルがもう一つある。役割分担だ。あるいは言い方を変えて、組織の設計でもいい。組織についても（権力者が恣意的に振り回すのでなければ）論理を構築せねばならない。私たちは日本的な組織構造に浸っていると思うが、ここ数十年で大して変化はしていない。なぜかというと（組織自体の力学や政治的なあれこれを除けば）単に概念的中継の能力が足りないからである。自らつくる力がないから、外の成功事例を見て真似ることしかできない。

構図を整理すると、次のようになるだろう。

engineering

私たちは人間なので、人間の論理――たとえば本書でも絶対視をやめようとしているコミュニケーションしか使えない。一方で、デジタルや組織にはそれ自体の固有の論理があり、何とかして扱わねばならない。そこで概念をつくって、これを使って人間側でのやりとりや向こう側での実装を行う。概念を扱う能力が最も重要なのだ。そして、現時点で最もここに長けているのがソフトウェアエンジニアだと考える。よって、**組織の設計を、役割分担を上手くやりたいのなら、ソフトウェアエンジニアあるいはその工学に頼るのが近道** だと考える。ちなみに筆者にもソフトウェア・エンジニアリングの心得があり、本章のような言語化と整理が出来ているのもまさにその知見が生きている。筆者は堂々とソフトウェアエンジニアを名乗るには心苦しい半端者だが、もっと強い人もたくさんいよう。その気になれば、本章や本書が霞むような概念くらいつくれるだろう。

このような見解は決して目新しいものではない。古くからソフトウェアエンジニア達は、エンジニアリングの過程でよく組織の設計にも携わってきた。現代でもソフトウェア・エンジニアリングの本

の、三冊に一冊は組織の話だと思う。エンジニアの人は、技術がしたいのに組織論の本ばかりでなんだかなぁと感じたことがあるかもしれないが、そんなものだ。本質的にはどちらも同じで、概念を扱っているにすぎない。ただ、概念をどう実装するかが違うだけだ。ソフトウェア・エンジニアリングを使ってソフトウェアにするか、それとも組織論を使って（ルールや体制などを整備して）組織にするかが違うだけ。

というわけで、本項の主張を一言でまとめると、役割分担（組織の設計）にもソフトウェア・エンジニアリングが使えるのではないか、である。

だとした場合、用語がないと不便であろう。デジタルの論理に対応するためにつくるものがソフトウェアであり、そのための理論がソフトウェア工学（ソフトウェア・エンジニアリング）であった。同様の定義ができるのではないか。

組織の論理に対応するためにつくるものを**オルグウェア（Orgware）**と呼び、そのための理論をオルグウェア・エンジニアリングと呼ぶのはどうだろうか。ちなみに「組織工学」は生物的な組織の工学としてすでにあるため使えない。新しい着眼であることを目立たせるためにも、オルグウェアという見慣れない横文字を使えばいいだろう。

たとえば本章で述べてきた兵農分離のモデルは、まさにオルグウェア・エンジニアリングの一コンテンツとして扱える。他には、本章で何度か扱ったティール組織も、まだ確立には至っていないため扱っていきたいところだ。この調子で、もっと積極的にオルグウェアをつくって、議論して、試して知見を貯めて、理論として反映していけばいいのではないか。すでに概念とお友達なソフトウェアエンジニアの皆さんならできると思う。無論、オルグウェアはソフトウェアとは違って、すぐにつくって試せるわけではないが、そんなものは試せる仕組みをつくれば済む話だ。個人的には AI エージェントに期待している。AI たちに、人間がそうするように組織的に過ごしてもらってシミュレーションさせるわけだ。手段はともかく、厄介なのがテネットだが、本書にてしつこいくらい壊してきたつもりだ。

まとめ

- 役割そのものを扱う目線に立てねば、デフォルト・リモートは成せない
- まずはテネットが邪魔しているので解体する
  - 単一のあり方を強要する平等指向
  - 兼務やマルチタスク
  - 役割と働き方の混在
- 役割でのやりとりを促すために、表明やロールプレイが使える
- 特に典型的な仕事を行う「ワーカー（兵士）」への一辺倒がボトルネックであるため、現代的な兵農分離を使って緩和する
  - 特にワークを行わないメタワーカーと、ワーカー - メタワーカー間を繋ぐ連携役が必要
- 現代版・兵農分離の例

- ワーカー・トライアングル（ワーカー、アドバイザー、両者を連携するヘルパー）
- ミドルオフィス（バックオフィスとフロントオフィスを支える）
- 二層仲介（AとBの間を「A寄りのB」と「B寄りのA」が仲介する）
- 特区（ワーカーの理に左右されない特殊組織を独立させてつくる）

Chapter-8 文字通りのITリテラシー(1/2)

デフォルト・リモートではITツールを用いたやりとりを多用する。リテラシーというと読み書き能力を指すが、同様にITリテラシーなるものが不可欠となる。しかしITリテラシーというとIT知識全般を指す言葉として使われているため、本書では違った定義を用いた。具体的には、文字通りにITを用いた読み書き能力を指すこととし、**文字どおりのITリテラシー（Literal IT Literacy）**と呼んだ。

本章では、文字どおりのITリテラシーを強くすることを目指して、具体的な話を盛り込んでいく。すでに知っている人には退屈だし、逆に縁が無い人にはそこまで鍛えねばならないのかと気が遠くなるだろう。律儀に全部をマスターする必要はないが、押さえれば押さえるほど不便がなくなり、慣れやすくなってデフォルト・リモートに近づける。

環境

リテラシーというとツール（道具）を使う能力を思い浮かべがちだが、これに限らない。ツールを使うのは人であり、人は何らかの環境で過ごす。よって環境がツールの運用に与える影響は決して小さくはない。ツール以前に環境に目を向けねばならない。

作業空間

作業空間とは仕事をする場所を指す。英語ではワークスペース（Workspace）だ。ガレージや書斎、あるいはオフィスの自席などが典型例だろう。またITツールの中では、チームと情報が集まる単位にこの言葉がつくことが多い。本項では前者、物理的な空間を指すことにする。

四つの要件

まずは要件から考えてみよう。作業空間の要件としては、以下の四点を意識したい。

- 切り替え。生活空間との切り替えが上手くできる、しかし手間がかかりすぎないこと
- 切り離し。生活空間と切り離されていること
- 強制力。集中できる程度の圧があること、またありすぎないこと

- メンテナンス。心身の調子が悪くならないための各種メンテナンス。換気、温度と湿度、清潔感 etc

切り替えとは仕事から私生活に、あるいは私生活から仕事に切り替えることを示す。結論から言うと、切り替えるときは **環境も** 切り替えた方がいい。人間は意識だけで切り替えられるほど器用ではない。ワークアズライフのような、いっしょくたにしてしまう考え方が持ち出されることもあるが、単に切り替えが下手だから仕事三昧でいいかと開き直っているだけだ。仕事中毒かつタフな人には向いているが、万人向けではないし本章でも想定しない。

というわけで、作業空間と生活空間の二つがあるのが望ましい。古典的にはオフィスと自宅であり、切り替えはできていたが、切り替えに要する時間がかかりすぎていた。デフォルト・リモートでは自宅がメインとなるため、切り替えに要する時間の問題は割愛する。自宅の中で部屋単位で切り分けることになるだろう。あるいは、部屋が少ないならパーティションやカーテンなどで仕切ってでも区切りたい。物理的に区切るのである。区切るということは、シームレスの章で述べたシーム（継ぎ目）を設定するということであり、シームがあることで切り替えを意識させることができる。つまり **切り替えるための負荷を意図的に入れる** 必要がある。この手間を惜しんではいけない。

加えて、もし家族と過ごしていて、意図せず作業空間に割り込まれるのであれば、時間帯を指定してその間は原則入ってくるなと徹底してほしい。切り替えは自分の意図どおりに、あるいは「ちょっと休みたいな」と思ったときに休めるよう自然に行えるべきである。割り込みによって無理やり切り替えさせられるのは負荷でしかないし、割り込みの可能性があるだけでもまともに集中できなくなる。家族がいる人でリモートが下手な人は、たいていはシームの設定が上手くいっていない。

次に切り離しとは、作業空間が生活空間（仕事をせず私生活を営む空間）から物理的に切り離されていることを指す。具体的には、生活空間から作業空間が直接見えないことと、作業空間で過ごしているときの生活音や作業音が生活空間に届かないことを指す。ニュアンスとしては **隔離** だと考えてほしい。ちょっと離れるからいいやではなくて、まるでウイルスを病棟ごと隔離するかのように厳しく切り離すイメージ。あるいは出社でもいい。ともかく、それくらい切り離さないと、自分としても心理的そして生理的に切り替わりにくいし、同居人や隣人にも迷惑がかかってしまう。そういった問題をカバーするために、だらだらとした過ごし方や浪費、ひどいと暴力的にストレスをぶつけてしまう。切り離されていないのだから当然だ。まずやるべきことは、切り離すことである。デフォルト・リモートの場合、作業空間≒自宅になると思うが、自宅では空間的に限度があったりする。自宅以外の場所も検討せねばならないかもしれない。この点は次項で述べるとして、いったん続きを見ていこう。

三つ目の強制力とは、場がもたらす圧である。カフェで作業する人は多いと思うが、なぜ捗るかというと、カフェという場がそれなりに圧を出してくれているからだ。人目があるから怠けられないし、覗き見や盗難といったセキュリティ的な警戒も要る。自宅とは違って何でも揃ってはいないから脱線もしづらい。同様に、オフィスも強制力は強い。仕事ではないが学校もそうである。仮に学校で習ってきた教育を、全部自宅で自分でやる（教材としてリモート動画は完備している

とする）としても、おそらく大半の人にはこなせまい。だからこそ塾や家庭教師といった「場を設定する」ビジネスも成立する。強制力の恩恵は、普段意識していないだけで相当にある。

さて、デフォルト・リモートはリモートだし、マネジメントさえもなるべく省いて自律的に過ごすことを狙っている。強制力を排除したあり方と言っても過言ではない。となると当然、強制力の恩恵があってこそできていた人達は、まともに仕事ができなくなる。どうするか。

注入すれば良い。**コミュニケーションは必要に応じて注入すれば良いと強調してきたが、強制力の圧についても同様だ。**これは圧をもたらす場を必要に応じて使うことと同義であり、事実上 **そのような場が選択肢として存在していることを要請する。**存在しないなら、新たにつくらねばならない。オフィスは場の一つではあるが、万能ではない。特に通勤の負担が大きいし、遠方で住んでいる場合は非現実的だ（出張ベースや単身赴任も不可能ではないが人間的ではない）。

というわけで、おそらくオフィス以外の場をつくることに投資しなければならないはずだ。あるいは、投資までできないのであれば、テクニックを取り揃えるしかない。カフェ、公民館や図書館、コワーキングスペースなどの情報を共有したり、自宅で強制力を高めるための整備をしたり（ビデオ会議を常設して相互監視できる仕組みをつくるなど）すればよい。なお、公共性の高い施設を使う場合は、コンプライアンス――特に会社が規定する情報セキュリティの問題もたちはだかる。機密性を扱う仕事をカフェでやっていいかという話であり、厳しい企業であれば禁止しているはずだ。だとしたら緩和しなくてはならない。細かいテクニックや道具や認定制など色々と導入せねばならないだろう。

最後にメンテナンスについては、人間としての調子を落とさないための作業をこまめにやるという話である。例を挙げる。

- 二酸化炭素が溜まりすぎないようにする
  - 換気する
- 暑すぎず、寒すぎないようにする
  - 温度と湿度を調整する
- 不衛生で気分を落としたり、体調を崩したりしないようにする
  - こまめに掃除する（特にゴミの除去）
  - 手洗いやうがいをする

私たちが使っていた（or 使っている）オフィスは、実はこれらのメンテナンスがかなり行われている。一方、自宅はそうでもなかったりする。気づかず受けていたメンテナンスの恩恵が得られず、自宅でリモートをするとなぜか居心地が悪いし、気分が悪くなる――なんてことになる。おそらくメンテナンスが足りていない。一つの目安として、自宅に他人を呼ぶ習慣があるのであれば、おそらくメンテナンスは足りている。麻痺した自分や家族、親しい友人などを除いて、他人が自宅に居座るためには、それなりにメンテナンスされてなければならないからだ。

厄介なのは、メンテナンスにもそれなりの手間とお金がかかることだろう。暑さと寒さを一つ取ってみても、時期にもよるが冷暖房と加湿器を常時稼働しなければならなかったりする。部屋にも

のがなければ購入と設置からだし、電気代もかかる。若い場合はあまり気にならないかもしれないが、加齢に伴ってそんなことは言ってられなくなる。若い読者の人は、30 代以上の人が妙に服装やグッズにこだわって自衛しているのを見て滑稽に思ったことがあるかもしれないが、あれもメンテナンスのようなものだ。環境側を変えられないので、道具を使ってカバーしているのである。

四つのプレイス

要件の他に、「場所の種類」という観点でも見ておこう。サードプレイスという言葉を聞いたことがあるだろうか。このような概念を四つに整理する。

自宅のように私生活を過ごす場を **ファーストプレイス** という。オフィスのように仕事として過ごす場、つまりは職場を **セカンドプレイス** という。デフォルト・リモートでは、基本的に自宅で過ごすことになるが、これはファーストプレイスで仕事をするとも言える。

ファーストプレイスは本来仕事として使うことを想定していない。すでに前述したように、作業空間をどう確保し、切り分けるかという設計をやらないといけない。

一方、セカンドプレイスは、デフォルト・リモートでは不要になるか、たまに強制力の圧を受けるために使う程度の場所となる。出番自体がなくなるか、少なくとも減るはずだ。コストがかさむなら解約するか、最低限組織としての拠点機能をもたせる程度に縮小してもよい――と、ここまでが鉄板だが、逆の発想もできる。つまりファーストプレイス的に（私生活的に）過ごす場として使うのはどうか。たとえばホテルのように宿泊機能を整備して、かつプライバシーも保てるようにする。オフィスにカフェや遊技場をつくる試みは（主にリッチな外資系によって）当たり前に見られるが、さらに発展させて寝泊まりも可能にするのである。そうすると他律派――出社が多い人達にとってインセンティブとなる。デフォルト・リモートではリモートが基本であり、出社は罰ゲームの様相を呈することが多いが、このような機能があれば、むしろご褒美になる。

近年では **サードプレイス** という考え方も見られるようになった。自宅でも職場でもない、第三の場所だ。原義はオルデンバーグによるもの（1）だが、いったんおいておくとして、プレイスの種類を整理してみよう。

place_matrix

ファーストプレイスとセカンドプレイスの共通点は、義務があることだ。ファーストでは生活や家族の養育の義務があり、セカンドではもちろん仕事の義務があった。いずれにせよ **義務から抗うことができなかった**。これを可能にしたのがサードプレイスの概念であり、要は義務を負わずに済む私生活的な場である。典型的にはパートナーと子どもとの付き合いに疲弊する人が、そういうことを意識しない、でも私生活的にくつろげる場ということになる。あるいは独身であっても、普段は健康と仕事のために普段はストイックに暮らしているが、それをする必要がない場でも良い。形態としては、何らかのコミュニティや施設に通うか、別荘や「誰もいない自然環境」など孤独なマイ環境を確保するかになる。

デフォルト・リモートにおけるサードプレイスの活用は単純で、ファーストやセカンドだけで私生活の成分（特に気分転換や休息）が足りない場合に適宜補充する。選択肢は無数にあり、何らかの店舗や施設に通うのが好きな人もいれば、公園や広場などお金のかからない開放的な空間も使えるし、その中にある「何らかのコミュニティで使われてるたまり場」でたむろってもいい。何ならサイクリングやツーリングなど場所にとらわれず動き続ける活動もある。自分の感性に従って、無理のない範囲で楽しめばいい。

とはいえ、ひとりでの開拓には限度があるから、社員全員でサードプレイスの考え方やノウハウを共有できるとなおよい。また、原義の概念では交流を重視しているが、交流はなくてもいい。むしろ孤独な場所であってもいい。重要なのは仕事の義務からも、私生活の義務から、どちらからも逃れることにある。別の言い方をすると**義務から逃れるために交流が良い人もいれば、孤独が良い人もいる**。適切な方を選べばいい。もし交流の方が合っていそうなら、オルデンバーグの理論の方が参考になるかもしれない。ただし個人の便宜ではなく地域社会の QoL 向上や再生の目線に立っているし、1999 年の話であってリモートワークの視座も反映されていない。

残る一つは **フォースプレイス** であり、義務から逃れる点ではサードと同義だが、どちらに主軸を置くかが違う。サードプレイスは、どちらの義務からも逃れているが、過ごし方としては私生活である（つまり仕事はしない）。逆に **フォースプレイスは、過ごし方としては仕事になる（仕事をする）**。義務はないが仕事はするという、一見するとおかしな状態であるが、不思議なことではない。仕事というとわかりづらいが、マトリックスのとおり「キャリア」と言えば、わかりやすいだろう。つまり目先の仕事ではなく、未来の仕事のために――特に **自分の** ために過ごす。普段の仕事を行うプレイスでは忙しくてやりづらいだろうから、場所ごと変えて、何なら新たに開拓するわけだ。もちろん目先の仕事であっても、腰を据えて考えたいことや個人的に改善したいことなどがあれば取り組んでもいい。発想自体は何も目新しくはないが、そうするための場所（フォースプレイス）を確保する点が違う。

フォースプレイスの選択肢としては、数時間以内などスポットで集中できる施設がいいだろう。カフェやコワーキングスペースも選択肢に入るし、カラオケも使えるかもしれない。トリッキーな選択肢としてはラブホテルもあり、個人で使うのは高額かつ敷居も高いが、事実上数時間単位で使えるホテルのようなものである――このように選択肢は無数にある。最も重要なのは **自分が使いたいときに使いたいだけ使えるかどうか** である。たいていの施設ではおそらく満たさない。時間帯によって席が埋まっていたり、締まっていたりするからだ。このレベルの利便性を手に入れるためには、高額を支払って専有的な形態にするか、いい感じの店を見つけて常連として定着するかを狙うことになる。また、環境からは離れるが、自分の生活リズムを固定することでも実現しやすい（フォースプレイスを利用する時間帯が固定的になるため合わせやすい）。

デスクまわり

文字通りの IT リテラシーとは IT ツールを用いた読み書きであり、単に相手がいて会話できればいいほど単純ではない。読み書きを行うための道具のみならず、それを支える土台も必要と

なる。具体的には PC、ディスプレイ、キーボードやマウスといったハードウェアを配置、操作するための環境――つまりは **デスクまわり** だ。

デスク

何もよりもまずはデスクだ。机ではなく、あえてデスクと言っているのには理由がある。というのも、**机というと多機能で雑に使うイメージを持つだろうが、それではダメなのである**。デスクとは、文字通りの IT リテラシーを発揮するために最適化された机を指すことにしたい。

クリアデスクという言葉があるが、まさに正しい。ただしセキュリティの文脈は関係無いため無視して論じる。書くためのキーボードやマウス、読むためのディスプレイを問題なく配置するためには事実上、机上がスッキリしりしていなくてはならない。机上が汚いと、ベストな位置ではないところに置いたキーボードを、体をひねらせながら打つなんてことになる。それでも仕事はできるが、デフォルト・リモートでは読み書きこそがメインの営みになるのだし、何なら仕事≒読み書きですらある。汚い机でこなせるだろうか。否、不可能だ。やりづらさが面倒くささに負けてしまうし、机には着いてても結局オンライン会議ばかりしてしまう。デフォルト・リモートは比較的長時間、机に座って読み書きを行うことを求める。それができる程度に机は洗練されていてほしい。だからこそデスクだし、クリアデスクなのだ。

またクリアデスクには脱線を防ぐ効果もある。スマホ、本、ゲーム機、嗜好品などを置いている人は多いと思うが、ついつい脱線してしまってはいないだろうか。それで休憩や切り替えができるなら問題ないのだが、筆者としては正直疑っている。偏見かもしれないが、そういう人達は会議を好み、コミュニケーションは大事などとぬかしながら、自分の頭を切り替える作業に相手も巻き込む。別に仕事の仕方に正解はないのでそれでもいいのだが、デフォルト・リモートとしては悪手だ。拘束を前提にしてはいけない。切り替えは自分でやるか、望んだ相手を巻き込んでやるものだ。そして、切り替えとは、前述したように環境次第であるから、デスクから離れた方がいい。離れるのを促すためにもクリアであることが望ましいのである。

デスクの広さについては、狭すぎなければ問題はない。逆に広すぎる方が持て余してしまって、おそらくクリアから遠のく。これも偏見かもしれないが、L 字デスクレベルの広い机で（SNS に晒す際の一時的な誤魔化しは除く）きれいなものは見たことがない。かといって、飲食店のひとり用テーブルや学校の机のような狭いのも論外だ。目安を語るのも難しいし、どこまで参考になるかも怪しいのだが、縦はキーボード 3 枚分、奥行きはキーボード 2 枚分くらいが良いのではないか。キーボードが大きい人（テンキーなどを使っている人も含む）は広い方がやりやすいだろうし、逆にコンパクトなものを使っている人は多少狭くても済む気がする。経験則に基づいた目安だが、どうだろうか。ちなみに筆者もこだわりはなく、2025 年 2 月現在はバウヒュッテの BHD-1200M-WD（幅 120 cm x 奥行き 55 cm）を使っている。

無論、必要な道具が多くて、物理的に広いスペースが無いと話にならないのであれば、大きなデスクを使えばいい。極端な話、デスクだけでは足りず、ガレージのような部屋レベルの空間が必要な人もいるだろう。

椅子

もう一つ、デスクとは切っても切り離せない **椅子** も取り上げたいところだが、言えることはほとんどない。個人差が大きすぎるからだ。

それでもあえて言うのなら、可能な限り、高価なものが良い。数千円よりは 1 万円、1 万よりは数万、数万よりは 5 万、5 万よりは 10 万。それで椅子の高水準な世界観を知ってから、次を考えればいい。どちらかと言えば椅子よりも、椅子の座り方や配置の方が重要なのだが、筆者に知見がないため解説は割愛する。調べればガイドラインの類もすぐに出てくるが、個人差を吸収できるほどではないと思う。デスクワーク、特にリモートワークにおける椅子の話は、ぜひ詳しい人にまとめていただきたいものだ。

ちなみに筆者は深く考えず 17 万円のアーロンチェアに投資したが、10 年以上長持ちするし、座面がメッシュでお尻が蒸れないので気に入っている。だらしのない座り方ができないのもデスクから離れるメリハリになって良いのだが、のんびり動画視たりゲームしたりするときは少しつらい。理想を言えば仕事用がアーロンチェア、私生活用には別のくつろげる椅子を設けたかったが、色々あって厳しいのでアーロンチェア 1 脚で生活している。もう少しくつろげる別の椅子にしたいとは何度か思っているが、行動には移せていない。

もう一つ、お尻が痛くならないものも必須だ――と言いたいところだが、これもそうではないらしい。たとえばアーユル・チェアーという長時間座れない椅子すらある。そもそも椅子は多目的なもので、ゲームのために長時間の着座に耐えうるゲーミングチェアもあれば、アーロンチェアのようにくつろげはしない真面目モードの椅子もあるし、アーユル・チェアーのような短時間タイプもある。特に短時間のものは矯正など健康の側面も出てくることが多い。そういう意味では、スタンディングデスクも同じようなものだろう。椅子を廃することで切り替えを促すものであり、アーロンとアーユルの中間くらいのブツではなかろうか。

アタッチとデコレーション

机の周辺に便利な収納をつけるなど、機能を追加することをアタッチと呼びたい。またフィギュアを置くなど実用性を抜きにして配置することをデコレーションと呼びたい。アタッチやデコレーションはどうすればいいのか。

好きにすればよい。

個人的にはクリアデスクが望ましいと考えるが、ちょっとした休憩や切り替えをデスクから離れず済ませたい人も多いと感じる。また、クリアすぎて味気がないと、かえってモチベーションが削がれてしまう（あるいは上がりにくい）人もいるだろう。むしろ筆者ほどのクリア派こそがマイノリティかもしれない。どの程度アタッチやデコレーションをするかは、各個人が調整すればいい。

一つ言えるとすれば、まずは何もないクリアな状態から始めて、足りないものを一つずつ追加していく **加算式** を推奨する。本章でここまで述べてきたとおり、文字どおりの IT リテラシーを発揮する「読み書き」の営みは、思っている以上に大変なものである。

想像してみてほしい。あなたはこれから毎日フルリモートで働くが、毎日半日くらいは、デスクに座ったまま読み書きをしている。半日というと、1日8時間としたら4時間くらいか。実際はそんなに保たないし、リモートなので休憩などは好きにすればいいが、それでも仕事を成立させるにはそれなりのやりとりが必要で、1日3時間はかけることになる。**1日3時間、デスクの前で、それなりに集中して読み書きを行えるだろうか**。誰かと喋らず、キーボードで（あるいは音声入力を使ってもいい）で書いたり、ディスプレイ越しにひたすら文章を読んだりする。デフォルト・リモートであれば、これくらいは当たり前だ。毎日続くものだ。コミュニケーションの注入は適宜行うが、それでも1日3時間は続くとしたら――。

いかがだろうか。たぶん、大半の読者には難しいのではないか。単に慣れてないとか、読み書きという営み自体が好きじゃなくて耐えられない等もあるが、それ以上に**気が散って脱線してしまうせいでできない**ことが多いのではないか。あるいは自覚がないかもしれない。脱線は手強いのである。特に読み書きは、対面で非言語情報を豊富にやりとりするものではないから、頭の処理をずいぶんと持て余す。持て余した分は何かで埋めたがるわけで、タッチやデコレーションが過剰だったり単に散らかったりしているとそれらに行ってしまう。

読み書きは、集中を要する営みである。対面やオンラインでのコミュニケーションとは全く異なるもので、**脱線しやすい営みでもある**と言える。だからこそ、脱線しない程度にデスクを整備しなければならない。しかし、人間なのでクリアすぎても病んでしまう。どこまで付け足してもいいか、散らかしてもいいのか――そのバランスは、自分自身にしかわからない。どう探るかも別に正解はないが、クリア状態から少しずつ増やしていく加算式がわかりやすいと思う。

ワークスペース・エンジニアリング

正直言うと、クリアデスクで自律的に読み書きを営めるのは、現時点では才能といっていいものだと思う。筆者はすでに本書で扱ってきたような自律的な働き方をゆるく啓蒙してきたが、ついてこれる者はほとんどいない。自律性を持つ者はITリテラシーが足らず、ITリテラシーを持つものは自律性が足りない。前者にはIT介護が必要だし、後者には他律のためのフォローが要る。いずれにせよ拘束してコミュニケーションを取らねば成立せず、したがって、いつまで経ってもリモートワークが本格的に普及しない（あるいはできているところでもオンライン会議ばかり）のである。

シームレスの章にて鍛錬の必要性を述べたが、才能で片付けてしまわないための理論と仕組みの整備が急務だと感じている。仮に、本節で扱ってきた環境の話を体系化するならば、ワークスペース・エンジニアリングとでもなるだろうか。あるいは範囲を狭めて、デスクの話にしたいならデスク・エンジニアリングか。あるいはクリアを前提とするならクリアデスク・エンジニアリングでもいい。いずれにせよ、個人差が大きいことなので実践的に取り組まねばならず、エンジニアリング（工学）として整備することになるだろう。無論、理論をつくっただけで使ってもらえるはずもなく、鍛錬として、仕事や私生活のライフサイクルに組み込んでいかねばならない。なんとも壮大な話である。

ハードウェアとスペック

空間、デスクと続いて、次に目に入るのは PC だ。あるいはディスプレイ、キーボードやマウス、イヤホンヘッドホンにマイクといった周辺機器も含まれる。さらに言えば、ネットワーク回線や PC の性能も見逃せない。これらハードウェアとスペックの話をしていこう。

といっても、心がけるべきは一つだけで、

どのハードウェアも、支障が出ない程度のスペックにせよ。

この一点に尽きる。

自動車でたとえるなら、何百万以上の車を選ぶ必要はないが、自動車でなくてはならない。原付や自動車ではいけない。自動車を前提とする暮らしに、原付や自転車で対抗するには無理がある。あるいは二輪ならスピード面では足りるだろうが、運搬面では不便だろう。同じことである。文字通りの IT リテラシーを発揮するための最低水準というものがある。ここを下回るなど論外も甚だしく、自動車生活において自転車で済ますような無知で無謀な行為だ。

スペック

まずスペックまわりは、高い分には全く問題無いとして、最低でも以下は欲しいと思う。なおグラフィックボードは割愛する。

- メモリは 16GB 以上
- CPU は Core i7 以上、2.5 GHz 以上
- ディスクは SSD で 500GB 以上
- ネットワークは 20Mbps 以上、できれば 30Mbps 以上

これ以下だと現代の IT ツールがまともに動作しないか、しても動作が重くてストレスフルになる可能性が高い。クラウド利用がメインなので性能は貧弱でもいいのでは、と思われるかもしれないが、そんなことはない。ブラウザにせよ、専用アプリにせよ、それなりにパワーを要するし、メモリやディスクへの読み書きだって行っている。むしろ時代が進むだけスペックの単価は安くなるから、遠慮なく処理を走らせるようになる（たとえばビジュアルをより綺麗にする等）わけで、最低ラインの水準も上がっていく。無理やりたとえるなら、車道の制限速度が上がるようなものだ。20 年前は 40km/h だった道が、現代では 80km/h だったりするわけである。40km/h で走れればいいと誤解していると置いてけぼりになるし、80km/h 前提から見ると迷惑ですらある。

ネットワーク

ネットワーク部分をもう少し補足する。速度には上り（アップロード）と下り（ダウンロード）があるが、可能なら両方とも最低ラインを満たしたい。普段コンテンツを消費する用途では下りが重要だが、デフォルト・リモートでは読み書きを行う。書くこともするわけで、書くとは情報のアップ

ロードにほかならない。ここで「文章などテキストをメインに扱うだろうからそんなに速度は要らないのでは？」と思いがちだが、そうでもない。

デフォルト・リモートだからといって、テキストだけでやりとりするわけでない。画像――特に画面のキャプチャもアップするし、動画だって撮る。なくてもできないことはないが、動画像をさっさと撮ってアップロードした方がわかりやすいことは多い。そもそも仕事ではサイズの大きなファイルのやりとりくらいあるだろう。

もう一つ、通信のリアルタイム性を示す ping 値（レイテンシ）があり、これは 20ms 以内、できれば 10ms 以内を目指したい。昨今ではリアルタイムに複数人が同じ場所に集まるスタイルのアプリが増えていたり、コンプライアンスが厳しい企業だとシンクライアントを使わされることもある。リアルタイムな反映の良し悪しが仕事のパフォーマンスに影響する時代だと思う。

ネットワークの計測方法は、Google 検索で「speedtest」などと検索すると出てくる「インターネット速度テスト」が手軽だ。

speedtest

ネットワークとして何を使うかにも踏み込もう。有線と無線があるが、できれば有線をおすすめする。無線はレイテンシが悪かったり、通信品質にむらがあったり、通信制限が厳しかったりしてデフォルト・リモートとは少々相性が悪い。できないことはないが、おそらくストレスフルな機会が増える。気にならないなら構わないし、気になるかどうかは人によると思う。無線の性能では許容できない人は、投資をしてでも有線を敷いた方がいいだろう。

スペックについてはそんなところか。他にも語れることはたくさんありそうだが、マニアックな領域であり、筆者も無知なため割愛しておく。詳しい人にお任せしたい。（お金に余裕があれば高性能で揃えればいいが）おそらく普通は各個人の予算とデスク環境に応じて、最適な構成を考えることになるだろう。

ディスプレイ

モニタと呼ぶこともあるが、本章ではディスプレイで統一する。

スペックとしては最低限、解像度が1920 x 1080（フル HD）、リフレッシュレートは 60Hz は欲しい。これ以上でも問題はないが、これ以下だと画面が狭すぎて不便だったり、描写が滑らかではないせいでストレスを溜めたりする可能性が高い。仕事では 30Hz あれば十分といわれたりもするが、昨今では動画コンテンツも多いし、ゲームライクな体験も使われるので足りないし、論外だと思う。60Hz でも少ないくらいだ。もしノート PC を使っていてディスプレイを選べない場合は、外付けも検討したい。

気になる論点と言えば、ディスプレイの枚数だろう。ここは個人差が大きいので何枚でもいい。

筆者はシングルタスクにこなしたい人間であり、シングルディスプレイにしているが、画面自体は

仮想デスクトップ機能で切り替えている。仮想デスクトップとは画面を論理的に n 枚分用意することで、Windows だと Windows キー + Tab キーで呼び出せる。作業用画面とブラウザ用画面を分けるとか、プロジェクトごとに画面を分けるといった使い方ができる。課題は二つ以上のウィンドウを見比べたいときにやりづらいことであり、このときだけは物理的に二枚以上にして並べた方がいい。二枚欲しいと思ったことは何十回とあるが、普段から二枚だと持て余すし、デスクのスペースなども考慮して手を出せないでいる。これは筆者の例であり、たいていの人はそこまでシングルタスク脳ではないと思う。自分がどこまでマルチタスクするかを踏まえた上で、枚数を増やしていくといい。

例外として、仕事として複数のウィンドウを見比べなければならない場合は、必要なだけ用意するべきだ。デイトレーダーや配信者がわかりやすい。あるいは長大なエクセルファイルを扱う一部の SE や事務の人も複数枚が必須だろう。と、このように絶対に複数枚必要なシチュは案外限定的だと思う。複数枚こそが捗るという人は、その人の指向がそうであるというだけだ。あくまでも自分に合った枚数を探るのが大事である。仮に同じ仕事をするチームメンバーであっても、一枚の人がいれば四枚の人がいたっておかしくはない。どちらが正しい、正しくないではない。自分に合った枚数にするべきだ。ちなみにディスプレイのサイズと解像度を大きくすることでも対応できるが、値段で言えば廉価なディスプレイ数枚よりもはるかに高くなる。

他には、リモートだと「私生活用のPC」と「仕事用のPC」が分かれていて切り替えが面倒くさかったりするが、この対処のために両方とも常設する（ことでディスプレイが二枚になる）パターンがありえる。デスク的に苦しいのであれば、ディスプレイは一枚にした上で、HDMI 交換機などで出力を切り替える手もある。もちろん、両方とも常設して常時起動していた場合、仕事と私生活の切り替えがしづらくなる点には注意したい。前述したとおり、切り替えには環境レベルでの切り離しが重要であり、そういう意味では面倒でも部屋を分けるなり片方を片付けるなりした方がいい。

周辺機器

続いて周辺機器を見ていきたいが、ここも語りだすときりがないし、スペックよりも個人差の大きい領域となっている。そこで値段の観点から論じることとする。

といっても言えることは一つで、**可能な限り、万単位の買い物をしたい。**

マウス、キーボード、イヤホンやヘッドホン、マイクなど基本的な機器は、使えればそれでいいかと安物を選びがちだが、甘い可能性がある。文字通りの IT リテラシーが要求される世界では、聞いたり話したりすることよりも読んだり書いたりすることが増えるし、前者についてもリモート越しだ。これら操作を司る機器が安物だと、思うようにいかないことがある。車でたとえるなら、座席の座り心地、ハンドルやレバーの握り心地や柔らかさといった部分の差として出るようなものだ。なくてもいいが、あった方が良いし、長らく過ごすのだから投資するだけの価値がある。こればっかりは、実際に使ってみないとわからない。しかし無理して手を出すほどではないし、安物で十分という人もいるが、筆者の経験上、リモートワークが苦手な人は安物の域を出ていない気がする。安物ゆえにさして快適ではあい、どころか不快であるせいで遠のいているのだとした

ら、これほどもったいないことはない。

無論、安くない買い物となるので、財布との相談となるし、そもそも会社として投資したい部分だ。たとえば社員ひとりあたり、年間 n 万円の予算を与えて、自由に買えるようにする。おすすめの情報や事例を自由に社内で共有してもらえば、社員たちは自律的に情報を漁るようになるだろう。非同期的な情報共有の、良い練習になる。お金というわかりやすいリソースをどう使うか、という状況は、主体的に情報を読み書きしてもらうインセンティブとして使いやすい。

話を戻そう。その他の周辺機器を支える道具が多数存在するが、各自で取り入ればいい。筆者は以下を使っている。

- マウスパッド
  - 本来はゲーマーが使うものだが、マウス操作の滑らかさが安定するため、もう手放せない
  - 数千円で手に入る
  - 筆者が使うのは SteelSeries QCK heavy で大型サイズ。マウスパッドの位置を調整しなくていいので大型にしている
- モニターアーム
  - デスクを広く保つために、ディスプレイを浮かせるモニターアームを使っている
  - 値段は数千円から数万円以上と幅が広い
  - 筆者はハーマンミラーの MM_DYN013 を使っている、4.5 万と高価な投資だったが特に不満なく使えている

道具を試す際は、これらハードウェアの操作に直結するものか、デスクの改善（≒広くする）につながるものに絞るといい。USB 給電の扇風機など「グッズ」は無数に存在するが、文字どおりの IT リテラシーに直結しないため効果は薄いし、ただの趣味の凝り性となって手段と目的が逆転しがちだ。趣味としてなら、あるいはアタッチ（デスクの機能拡張）としては悪くないが、そんなことよりも、まずは基本的なハードウェアの方こそを改善するべきである。

(余談) 費用負担の考え方

デフォルト・リモートを推奨するのであれば、社員にできるだけ投資するべきだ。ひとりあたり初期投資 30 万円とか、月 1 万円支給などがあっても決しておかしくはないし、オフィスの維持にかかる必要を充てればお釣りがくるくらいだ。逆を言うと、オフィスを潰すか、少なくとも縮小する覚悟を持て、腹をくくれということになる。結局、経営者が投資するかどうか次第だ。

パンデミックがまさにそうであったが、やむを得ない状況をつくりだすことで普及と啓蒙を促進させることもできる。オフィスを縮小してしまえばいい。縮小してしまえば、従来ベースの出社のやり方は物理的に通じなくなる。リモートでこなすには、を考えなくてはならず、従業員たちも主体的になる。

(余談) ノートかデスクトップか

PCとしてノートとデスクトップが存在するが、正直どちらでもいい。

ここまで述べてきたとおり、デフォルト・リモートでは作業空間は固定する。ノートだろうとデスクトップだろうと、デスクかその周辺に置いて固定するはずだ。そんなことよりもスペックの方が大事である。現代ならノートPC単体でも問題ないスペックのものが手に入る。強いて言えば、細かいカスタマイズや自力でのメンテナンスはデスクトップの方がしやすいが、自作パソコンの領域になるし、筆者もてんで無知なので割愛する。

(余談) スマホを排除する

スマホはただの道具だが、環境を論じる上では無視できない。また現代では仕事道具でもあり、本章ではここまでデスクとPCを前提として論じてきたが、ピンと来ない人もそれなりにいると思う。はっきりと主張しておきたい。

デフォルト・リモートでは、スマホではなくPCを使うべきだ。

スマホは外出先で使ったり、寝転びながらなど「ながら」で雑に使ったりする分には優れているが、「やむを得ない」代替手段でしかない。PCの方が生産性が高いし、デフォルト・リモートもPCの活用を前提としている。スマホでも部分的な作業やコミュニケーションは行えるものの、メインにはなりえない。

たとえるならゲームである。FPSでもマイクラでもいいが、本格的にやるのにスマホで済ませようとは思わないだろう。それなりの性能と操作が要求されるので、PCでなくてはならない。デフォルト・リモートも同じだ。ここまで見てきたように、文字通りのITリテラシーを発揮する読み書きとは、それなりの操作を伴う。人間相手のコミュニケーションよりも、ツールやゲーム相手の操作の方に近い営みである。だからこそ、しっかりと操作できる道具と環境を整備せねばならない。

そういう意味では、スマホは良い試金石になる。**デフォルト・リモートが成立していると、スマホは基本的に要らなくなる。**そもそもスマホが手放せない事情の大半は、単に出社のパラダイムが蔓延っていて移動と待機が多いからであり、デフォルト・リモートになればそれらは減る。現時点で「スマホ無しなんてありえない」という場合は、おそらくデフォルト・リモートがまだまだ身についていない。

もちろん、プライベートで使うのは自由だが、すでにデスクの項で述べたとおり、無闇に一緒に過ごすと脱線が増えてしまう。必要に応じて隔離できる程度にはスマホ依存を断てた方がいい。たとえば、デスクのそばに私生活用のスマホを置いている人は、おそらくデフォルト・リモートは難しいのではないか。高頻度にスマホをいじるせいで認知資源を食い潰したり、せっかく読み書きに集中し始めているのにスマホに脱線して集中が切れたりするだろう。代わりにスマホを玄関に置いて、主に外出時にのみ触れるようにするだけでも違う。シームレスの章でも述べたとおり、たとえ小さなシームであっても、あるのとないのとでは全然違う。すぐアクセスしてしまうのが問

題なら、軽微でもシームを設定すれば抑止できるのだ。なお依存症などひどい場合や、PC を知らずスマホしか知らない一部のデジタルネイティブは除く。

といっても、この立場は少々極端なものであり、万人にやってもらうのは非現実的だろう。実際はデフォルト・リモートがある程度始まってきたら、スマホで「ながら」に過ごしても成立するような方法論やソフトウェアが生まれるはずだ。ながらのニーズ（あるいは患者が多すぎて根絶しづらいとの社会問題と捉えることもできる）はある。ならば、ながらの排除よりも共存の方向に動く方が自然だろう。

そもそも今後音声認識と生成 AI はさらに発展するはずで、文字通りの IT リテラシーの要求水準は下がるだろう。あるいはリテラシーそのものが変化するかもしれない。読み書きではなく、スマホ経由で AI を相手にした聞き話しになるのではないか。ひょっとするとスマホさえも使わずに、もっと軽い手段になる（あるいは設置型のデバイス）かもしれない――と、妄想ではどうとでも言えるが、生成 AI の勢いを見ると、これくらいもそう遠くない未来に思える。

筆者としては、今のところ文字どおりの IT リテラシーは必要であり、鍛えていくべきだと考える。いくら AI であろうと、原始的なコミュニケーションでできることなど限りがあるからだ。単純な会話とノリで通じるほど仕事は甘くないし、それでは現代と同様、拘束のキツイ働き方から脱せない。人間との拘束が、AI との拘束にすり替わるだけだ。この構造的限界を越えるには、私たち自身が読み書きの力を身につけ、拘束なしで自律的に動き、非同期的に連携しあえるようにならねばならない。かんたんではない。だからこそ、今からでも進めていかねばならない。無論、これは万人に強制するものではなく、あくまでも選択肢として確立したいという意図だ。

冒頭でも述べたとおり、現代はデフォルトがリモートであるという第三のパラダイムなのである。そしてその本質は、単にリモートでコミュニケーションを取ることではなく、構造的に拘束を生み出すコミュニケーションそのものを減らしていくことにある。聞くと話すでは難しい。読み書きが必要だ。そして、そのためにはリテラシー（読み書きの素養）の強化が要となるのである。

読み書きのスキル

文字どおりの IT リテラシーを発揮するには、様々な能力が要求される。

才能や適性を論じても仕方がないので、ここでは能力を発揮するための「スキル」という観点で、どんなスキルがあるか、どうやって鍛えたりカバーしたりするのかを論じていく。具体的には言語化、構造化、処理、出力、整形の五つのスキルを取り上げる。

言語化スキル

読み書き、特に書くというからには、とにもかくにも言語化ができねばならない。

まずは書き言葉を扱うスキルだ。話し言葉と書き言葉は全く違う。たとえるなら、話し言葉を使ったやりとりはサッカーであり、常に状況を見ながら臨機応変に対応するものだ。一方、書き言葉を使ったやりとりは野球のようなもので、リアルタイム性はそこまでなく、ベンチで控えるなどじっくりと観察したり休んだりする余地も大きいが、サボることはできず、動くときに備えて戦略やウォームアップを準備していかなくてはならない。同じなのはボールや言葉だけであって、あり方からして全くの別物である。

何が言いたいかというと、**いくらサッカーをしても野球が上手くならないように、いくら話していても書くことは上達しない**。そういうわけで、月並みな結論になるが、書いてやりとりする機会をまずは増やさねばならない。書くのは苦手？言語化は苦手？当たり前だ。サッカーしかしてないのに野球は苦手と言っているようなものである。まずは経験を積め。話はそれからなのだ。

言語化ができる人は、それなりの経験を積んでいる。たとえば、1 日 2 時間以上、デスクと向き合って何かしら書くという過ごし方を何年も、何十年も過ごしている。当然ながら、書き言葉の経験がない人を鍛えるのはかんたんではない。少なくとも継続的に取り組んでもらうことになる。

一つの目安として、**毎日 2000 文字くらいの情報を難なく書けるようになったら及第点** だと思う。出来は問わなくてもいいが、最低限第三者に通じる必要がある。チームメンバー相手でもいいし、家族や友人でもいいし、ブログのようにインターネット上の不特定多数でもいい。内容も日記でもいいし、日報でもいいし、エッセイでもフィクションでも何でもいい。この時点では、まとまった文章でなくてもいい。X(Twitter) のような 50 文字のつぶやきを 40 回投稿する、でもいい。ただしコピペや書き写しやリポストはダメだ。あくまでも自分の言葉で書かねばならない。

目安を示したところで、言語化スキルの構成要素を見ていこう。結論を言うと **語彙力** であり、以下の三つがあるように思う。

- 量。どれだけたくさんの単語を知っているか
- 質。同上、どれだけ正確に知っているか
- リカバリ。語彙力の欠如を自覚できるか、またカバーするための行動を取れるか

量はその名のとおりで、極端な話、国語辞典内のすべての単語を知っている人は強い。難しいのは国語辞典がすべてではない点で、日本語以外の自然言語があるのはもちろん、分野ごとに専門用語もあるし、会社やコミュニティでは独自用語もある。これら語彙は知っていれば知っているほど有利である。残念なことに、話し言葉を使うやりとりでは、専門的な議論でもなければ（日常的な語彙以外は）ほとんど鍛えられない。

次に質については、各単語の意味（特に定義）と使い方を正確に理解して、使いこなすことを指す。たとえば「エンゲージメント」という言葉の意味は、人によって違う。お互いに認識が違うままでやりとりをしても、雑談や親交の強化くらいはできるだろうが、まともな議論にはならない。しかし辞書の定義、経済産業省の定義、会社の定義などをお互いが知っておればスムーズになる。難しいのは正解が一つとは限らないことだ。何なら存在しないこともあるし、存在はしているが辞書的定義ほど単純ではなく文化的なニュアンスされているケースもある。「コミュニケー

ション」「議論」「情報」は、どれも手強い言葉だ。

最後に、これが最も重要だが、リカバリである。**自分の語彙力を相手も持っているとは限らないし、逆に相手の語彙力に自分が追いついていないときもある。そういったズレがあったなら、直していかなくてはならない。**

たとえば本書では、コミュニケーションという言葉を「欲求を充足する行為」として使ってきた。かわりに、やりとりするという行為は「情報」を「やりとりする」と表現してきた。これはリカバリの一種である。筆者と読者とでは十中八九語彙が合わない（特に筆者側が新しい理論をつくろうとしていて独自の定義が多数存在するため）はずなので、明示的に定義や使い方を差し込んでいるわけだ。こうすれば、読者は筆者の語彙に合わせていくことができる――と、これは作家と読者の構図だが、もちろんカジュアルには「コミュニケーションと書いてますが、どういう意味として使っていますか？」のような確認もまさにリカバリだし、エンゲージメントという言葉が使われているが、意味を知らないので自分で調べてみたというのもリカバリだ。

そういうわけで、いくら量と質が優れていても、相手に寄り添えない人は言語化スキルが低い。逆に、量や質が乏しくても、相手に寄り添おうとリカバリを試みる人は、言語化スキルはそこまで低くはない。リカバリこそが重要なのだ。もちろん、リカバリもそれなりに頭や神経を使うものであり、日頃から使っていないと中々こなせない。あるいはこなせていた人でも、権力を得て機会を失っていくと衰えていく。

ちなみに、言語化スキルを鍛えるのに他者が必要に見えるが、実は自分だけでもできる。**未来の自分が見て（背景や意図はさておき意味は）理解できる文章を書けるようになればいい。**未来の自分は他人だ。

構造化スキル

まとまった情報を正しく伝えるためには、言語化して文章に落とすだけではなく、その構造化も求められる。

構造化も非常に奥深いが、いくつか挙げてみよう。

- 段落
- 目次や章節項
- 起承転結、ストーリー
- メリットとデメリット
- 要約、背景、課題、提案、詳細

本書もまさに構造化している。今見ているページのこの辺は、構造で言えば以下のとおり。

- 書籍『Remotism』
  - 第一章『はじめに』
  - ……

- 第九章『文字通りの IT リテラシー』
  - 『読み書きのスキル』
    - 『構造化スキル』□今はここにいる

『文字通りの IT リテラシー』章の、『読み書きのスキル』節の、『構造化スキル』項ということになる。改まった説明になるが、本はこのような構造化がなされており、構造単位ごとのタイトルを取ってきて目次をつくることで、全体像を眺められるようになっている。

他にも論文には論文の構造があるし、パワポのような資料にもやはり構造がある。ブログのような記事であっても段落が分かれていて、段落ごとにだいたいこんな感じの意図が込められている。あるいは少なくとも空行で区切って、段落の塊ひとつひとつがリズムを持って読めるようになっているはずだ。あまりに当たり前なので意識していない人も多い。その塊についても、様々な意味をもたせることができる。たとえば「メリット・デメリット」とか「メリデメは次のとおり」などと書いていると、おそらくその塊はメリットとデメリットを挙げて比較する意図があるだろう。

より詳しく言うと、配置の構造と意味の構造がある。章節項といった分類や、箇条書きで並べるといったことは配置の構造にあたる。目的は読み手の認知コストを減らすことであり、たいていセオリーが存在する。たとえば本には目次があって、章（大）、節（中）、項（小）の 3 階層程度で分類されていることはたいてい通じるし、むしろ用意してなければ読み手は不便に感じるだろう。対して、9 章として「文字通りのリテラシー」を扱おうとか、「読み書きのスキル」の項をつくろうとか、構造を説明するための箇条書きを書こうとかいったことは意味の構造にあたる。配置の構造として配置する各枠に、どういうものを置くかを設計している。起承転結や序破急、はじめにやあとがきやコラムといったものは意味の構造（として使えるテンプレート）にあたる。正解もなければ鉄板もなく、あるいはあったとしてもそれを愚直に適用するだけではどうにもならなくて、人の数だけ解がある。たまに何とかメソッドのような形で独自の理論がつくられることもある。

このように、言語化して文章に落とした情報は、何らかの構造をもって並べるのが鉄板だ。構造化せず一息に書き殴ることもできるし、アイデア出しの文脈ではあえてそうする（フリーライティングと呼ぶ）こともあるが、読みづらすぎて読めたものではない。プロの作家やライターなら没入させることもできなくはないが、才能の域であり、万人に求めるのは酷である（筆者も全然できてない）。

そんな構造化だが、もちろんそれなりの手間がかかるし、知らないと「使える構造」のレパートリーが乏しくてスタートラインにすら立てないかもしれない。何かを読んでいる以上、構造にはたくさん触れているが、単に読むのと、実際に自分でつくるのとは全然違う。思っている以上にできないし、**できるにしても思っている以上に労力がかかる**。前項のリカバリにて歩み寄ることの重要性を述べたが、構造化もまさにそうである。読み手に負担なく、しかし正確に理解してもらうために、書き手側で手間暇をかける。

さて、構造化の鍛錬であるが、長い文章を書けばいい。2000 文字、5000 文字、1 万文字、数万文字、あるいはそれ以上も可能である。段階的に実績を積んでいくといい。先ほどの「未来の自分は他人」も踏まえると、**未来の自分が読んでも理解できる長文をつくれるようになるといい** と言える。

ちなみに本書は 15 万文字を超える書籍であり、本格的な構造を必要とする。ここまでくると、慣れた筆者であっても片手間というわけにはいかず、数ヶ月以上の作業を要している。そして構造化に正解はないので、本書がどれだけ読者に刺さるかもわからない中、無限の可能性を進み続けなくてはならない。いわゆる創造（クリエイト）の世界に入る。構造化とは、創造的な営みなのである。

処理スキル

言語化と構造化のパフォーマンスをどれだけ上げられるかは、三つのスキルに依存する。わかりやすいのはタイピングやフリック入力の速さだが、これは出力スキルとして次項で述べる。ここでは出力の前に行う、脳内での処理の話をする。処理スキルと呼びたい。

脳科学や哲学に立ち入る気はないが、筆者なりの解説をしたい。脳内では観念 → 概念の順で処理していると思う。観念（Notion）とは考えのことであり、言語で表現する以前の何かである。一方、概念（Concept）とは、第三者でもわかるよう言語的に表現された考えである。概念は、観念を伝えるための主な手段だが、唯一ではない。絵や図表などイメージで伝えることもできるし、音や食感や匂いを使うこともできる。ただしそれら抽象的な表現は、受け手によって解釈がわかれるので、確実に伝えたいなら概念つまりは言語を使うのが良い。一方で、そもそも言語では表現しきれない感情や雰囲気といったもの、いわゆる非言語情報は、言語以外の手段の方が通じる。

と、誰もが知っていることを小難しく書いたが、**処理スキルとは観念の処理を指す**。言語化以前の、頭の中にある観念を感覚的にあーだこーだと操作したり、走査したり、あるいは捜査のように調べたりする。所詮、言語で表現できる情報など、観念のごく一部でしかない。身も蓋もないが、処理スキルこそが重要である。何かを書くとは、観念を言語化して表現することなのだ。仮に観念が 1000 しかない人と、100 万もある人とでは、後者の方が書けることが多い。100 と 100 万だ。違いは歴然である。100 しかない人が、言語を使って頑張って膨らませることはできるが、脳内で膨らませられる 100 万の人には遠く及ばない。

わかりやすいのが作家だろう。作家の才能とは、脳内でどれだけつくれるかという処理スキルの才能である。想像力とか創造力と呼ばれることが多いが、要は脳内での処理に秀でているわけだ。先の例で言えば、作家は 100 万の側の人間であり、もちろん 100 万全部は表現しきれないから、何をどの順番で表現するかを絞っていく。以前 X(Twitter) かどこかで「100 → 1（100 から 1 をつくる）」と表現したポストを見たことがあって、上手い表現だと思った。ちなみに処理スキルがないか、鍛えても身につかない「才能なし」は、脳内にはあまり観念がない中、言語化したものをこねながら膨らませていく必要があり、これを筆者は「0 → 1（0 から 1 をつくる）」と表現している。当然ながら 100 から 1 に絞る人達に敵うはずもない。あるいは敵うにしても、脳内ではなく、脳内の外で言語を使って営むのだから大変だ。100 → 1 の人は 1 日 10 時間以上出力できるが、0 → 1 は数時間しか保たない。

読者の一部は薄々感じているかもしれないが、筆者は 100 万ではなく 100 の側の人間であ

り、100 から 1 ではなく 0 から 1 でつくる人間だ。本書もそうしてできている。なので、仮に本書に関して対談を行ったとしても、私はすらすらは語れまい。「内容なんていちいち覚えてねえよ」「覚えてるわけねえだろ」と言いたくなる。もちろん、書いたものを読めば思い出せはするし、プレゼンのように事前に準備をすれば喋れもする。ただ、100 しかない人間なので、デフォルト・リモートの観念あれこれが常に脳内にあるわけではない（あるいは脳科学的には有りはするが引き出すのが下手なのかもしれない）。

さて、処理スキルだが、鍛えるのは難しい。先天的なものだと思っている。できることと言えば、自分や相手の処理スキルを推し量ることくらいだろう。

処理スキルの程度を見たければ、出力するためにどれだけ道具や環境や生活にこだわるかを見ればいい。処理スキルがある人は、出せる道具を使って出せばいいだけなのでこだわりがない。それこそ文章ならテキトーなパソコンのテキトーなキーボードで十分だろう。普段の仕事でも IT ツールや整理法など頼らず、原始的なコミュニケーション＋αくらいで済ませているだろう。また私生活もだらしなくていいし、シームレスの章で述べたように汚部屋であってもいい。汚部屋であろうと、それこそ不健康で不養生だろうと、何とかなるだけの性能があるので、いちいち改善する必要がない。一方、処理スキルが少ない人は、脳内だけではあまり何もできないので、色んなツールやり方考え方を駆使して、情報として外に出さなければならない。出した情報を見ながら作り込むのだ。読み書きも多用する。なんたって口頭だけでは情報として残らないし、脳内で残して処理するのが苦手ゆえに不利なのである。疲れるから（バテないための）セルフマネジメントも要るし、下手をすればアスリート的な生活になる。筆者もこちらのタイプだ。

出力スキル

脳内で処理したものを言語化・構造化して外に出すのが出力スキルである。また、書き言葉は、外に出したものをあれこれいじくることで品質を上げていくものであり、いじくるための道具のスキルも欠かせないが次項の整形スキルで扱う。

文字入力スキルとは、主にはタイピングを指す。スマホのフリック入力、マイク越しの音声認識、手書きの OCR（光学文字認識）でも良いが、2025 年時点ではまだまだタイピングが無難だと思う。タイピングなら秒間 10 打くらいをこなせる人は少なくないし、競技の世界だが熟練したタイパーなら秒間 20 打を超える。入力方式としてはローマ字が主流だが、親指シフトやステノワードなど同時押しに頼って高速入力するものもある。

タイピングスピードはあまり重視されない印象があるが、そんなことはない。速ければ速いほどいい。100 文字の文章を 3 分で打つのと 2 分で打つのでは後者がいいし、2 分よりも 1 分半、1 分半より 1 分、1 分よりも 50 秒で打てた方がいい。

速ければ速いほどフットワークが軽くなり、書くためのハードルが下がる。階段でたとえると、タイピングが遅い人は階段を見るたびにげんなりするか、下手をすればわざわざ混んでいるエスカレーターを使う羽目になるが、速い人はデフォルトで階段を使えるようなものだ。さらに速いと

デフォルトで二段飛ばしをする。階段一本だけでは大した差ではないが、このような移動が何時間も、何日も半永久的に続くとしたら、両者の差は莫大なものになる。デフォルト・リモートは、このたとえでいうなら、階段を当たり前に昇降することを要求するようなものだ。目安としてミュート・デイが使える。仕事か、私生活か、あるいは両方でもいいが、一日中誰とも一言も喋らず読み書きだけで過ごせるだろうか。タイピングが遅いと、やりとりがいちいちストレスフルなので耐えられない。逆にある程度以上速ければ、ある程度はスムーズなのでストレスはないか、許容範囲であり耐えられる。別の言い方をすると、**ミュート・デイをクリアできる程度の速さがあるならそれでいい**。

そしてもう一つ、見過ごされがちなのが、**読み書きにおいては、タイピングスピードは思考力そのものである**という点だ。本項冒頭で述べたとおり、書き言葉は外に出した上でそれをいじくることで洗練させていく。話し言葉が相手の雰囲気と言葉を受けて考えるが、書き言葉ではそれがない。じゃあ何を受けるかというと、自分が読んでいるものであり、特に何かを書いている場面では「自分が書いたもの」自体である。そういう意味でも、できるだけ速い方がいいし、空き時間に鍛錬したっていいくらいだ。とにかく、タイピングスピードは軽視されがちだが、もっと重視するべきものである。とはいえ、そろそろタイピング以外の手段も盛り上がってくるとは思うし、現時点でも仕事でフリック入力や音声認識に頼っている人は少なくないと思う。

ちなみに文字入力を補完するスキルもありえる。IME の辞書登録はわかりやすい。もし「コミュニケーション」という単語を頻繁に使うなら、「こみゅ」で登録しておくと、「こみゅ」から変換するだけで一息に打てるようになる。辞書登録と言えば、普通に変換しても出てこない難しい漢字やカタカナを登録するものと思うかもしれないが、実は頻出する単語を早く打つためにも使える。この考え方の実現方法は他にもあり、定型文をあらかじめ登録しておいてコピペできるようにした「定型文コピー」ツールや、`d[[` と打つと 2025/03/15 のような現在日時に展開されるなど、おまじないとその展開内容の組を登録しておいて省力化を図る「Text Expansion」などが知られている。そうでなくとも IME の予測変換はお馴染みだろうし、プログラミングでは生成 AI の力で続きを出力してくれるものもある。

最後にタイピングの鍛え方を議論しよう。といっても近道はなく、練習を重ねるしかない。どちらかと言えば練習のモチベーションをどうやって維持するかを考えた方がいい。20 年前は今ほど SNS や動画がなく、そもそもスマホもなくて、PC によるテキストコミュニケーションが旺盛だったため、娯楽として当たり前にタイピングを行っていた。やらないと和に入れないし、楽しさを享受できない、というわけで自然と腰が上がるし、楽しいので勝手に上達していった。現代はそうではなく、テキストコミュニケーションはオプションでしかない。正直なところ、どうやってモチベーションを上げるのか、維持するかの解を筆者は持たない。せいぜいタイピングゲームで楽しむのと、スマホや SNS から距離を置いて余裕をつくることくらいか。業務時間の一環として、皆で練習できたら楽しいし「まあ仕事の一環なら」と腰も上がりやすいと思う。おそらくタイピングを練習するための、新たな理論――は大げさにしても、教育やメニューを開発せねばならない。長い道のりにも聞こえる。そうしている間に、生成 AI ベースの、脱タイピングな文字入力が確立される方が早いかもしれない。

鍛え方としてもう一つだけ言っておきたいのは、タイピングゲームだけでは足りないという点だ。実

際に何かを書く際は変換もするし、構造化もするし、次で述べる整形も行う。むしろこれらの方が比重としては大きく、いくらタイピングだけが速くても、これら編集が遅ければ、総合的にはそんなに速くない。編集部分も含めて、実践的な速さが欲しいのであれば、やはり実際に書く経験を増やすしかない。日記でも、ブログでも、小説でも何でもいいので、文章を書くことだ。無論、これも万人が気軽に楽しめるものではないので、やはり何かしら教育をつくらねばなるまい。

整形スキル

書き言葉は、外に出したものをあれこれいじくることで品質を上げていくものだ。処理スキルとは違って、脳内で自由にいじくれるものではない。いじくるための道具が必要だ。ということは、そのような道具のスキルが欠かせない。これを整形スキルと呼びたい。

整形スキルはさらに二種類に大別できる。キャレット（文字入力のカーソル）を思い通りに動かして範囲選択やら消去やらコピペを行うスキルと、テキストエディタやその他文章を加工するツールを使いこなすスキルの二つだ。

前者、キャレットのコントロールについては、慣れるしかない。ひとつ例題を出そう。以下に「りんご」「ばなな」と書かれた行がある。この下に「りんごばななりんごりんごりんごapple」と書きたいとする。なおキャレットは「I」の位置にあるとする。なお、これは test.txt というファイルとして開かれており、下記以外の内容はない。つまり「I」の行は先頭行だし、新たに書き足す部分は末尾行となる。つまり test.txt は以下のようになっている。

```
I

りんご
ばなな
（ここに書きたい）
```

あなたならどうやるか。愚直に全部手打ちしてもいいが、おそらくコピペに頼るはずだ。りんごと書かれた行を範囲選択するか、自分でりんごと打った上でそれを選択してコピーして、まず貼り付ける。以下のようになると思う。

```
りんご
ばなな
りんごりんごりんごりんごI
```

ここから、おそらくまずappleと入力した後、キャレットを少し戻して、ばななと入力して完了するだろう。

差が出るとしたら、キャレットを動かす部分だろう。「行末に移動」「行単位の選択」といった操作は一発で行うことができ、これを使うとカーソルキーを連打したり押しっぱなしにする手間がなくなる。

あるいは、慣れた人なら「りんご」「ばなな」の二行をコピーすると思う。これを貼り付けて、ばなな の前にある改行を消せば「りんごばなな」ができる。あとはりんご部分をコピーして、ばななの後ろに行って四回貼り付けてからappleを打てばいい。あるいはもう一度貼り付けてから Back Space を三回押してもいい（りんごばなな → りんごにする）。いずれにせよ、慣れた人なら、初挑戦でも 5 秒以内にこなせると思う。

と、小さな差かもしれないが、**書くということはキャレットを動かすということでもある**。キャレットのコントロールは、タイピングに匹敵するほど重要なスキルではないか。通じなかったら申し訳ないが、ゲーム用語で言えばキャラコン（キャラクターコントロール）のようなものだろう。キャレットなのでキャレコンとでも呼ぼうか。キャレコンの鍛え方はもっと整備した方がいいと思う。

次に後者、テキストエディタやその他加工ツールのスキルである。現代では Web サービスを使うことが多く、ブラウザでテキストボックスに文字入力する機会が多いと思うが、実は賢くない。チャットなど短文を扱う程度ならそれでいいが、まとまった文章をつくるのには向いていない。テキストエディタという専用のツールが欲しい。Web サービスとしてテキストエディタが提供されていればそれでもいいが、オンラインゆえの遅延があるので、できれば PC 上で動かすものが良い。昨今では、IDE（統合開発環境）という高機能なエディタもあって、Visual Studio Code は有名だが、これも動作が重かったり、高機能で脱線しやすくなったり、日本語の扱いに弱かったりする。自分でカスタマイズできる玄人なら問題なく使いこなせるが、そうでなければテキストエディタが良いと思う。この話題は、これだけで本数冊以上になってしまうので割愛する。筆者は普段秀丸エディタを使っていて、解説本を書いたこともあるので、興味ある方は参照してほしい。秀丸エディタに限らず、エディタ全般の話を噛み砕いた。

最後に、加工ツールというのは置換や変換といった処理を行うツールを指す。たとえば「キャレット」と書いてるけど正直わかりづらいので「カーソル」に直したいとき、人力でひとつひとつ直すのは馬鹿らしい。置換機能を使えば一括で行える。この程度ならテキストエディタで行えるが、もっと込み入った加工ツールも色々とある。本書の範囲を越えるため割愛する。また変換については、わかりやすいのは翻訳だろう。こちらについては、正直言って生成 AI で十分だ。すでに ChatGPT などを使って、文章を貼り付けて、言い回しを丁寧にさせたり、要約させたり、もちろん翻訳させたりしている人は多いと思う。それで十分だ。一応、ChatGPT とのコピペは面倒くさい（シームレスじゃない）から何とかすることもできるが、現状は IT エンジニアレベルの工夫を要する程度には大変、というか自製が必要であるため割愛する。近いうちに専門知識がなくても使える機能や製品が出てくるだろうし、主力製品はすでに（使い物になるかは怪しいが）サポートし始めている。

Chapter-9 文字通りの IT リテラシー (2/2)

コラボレーションスキル

仕事は複数人と行うものであり、情報のやりとりが発生する。長らく原始的なコミュニケーションが使われているが、第二パラダイム「出社」から第三パラダイム「リモート」に移るには、コミュニケーション自体を減らさねばならない。少なくとも絶対視を取り下げなくてはならない。これをデフォルト・リモートと呼んだ。同期的なコミュニケーションの代わりに、非同期的な情報共有を行うべしと主張した上で、そのために必要なものを整備してきた。

最終的には IT ツールを使って、この中で情報をやりとりすることになる。欲求を満たすためのコミュニケーションというよりも、純粋に情報を読み書きする営みになる。そういう意味で、コミュニケーションという言葉はふさわしくない。コラボレーションと呼ぶことにしよう。となれば、デフォルト・リモートではコラボレーションのスキルが必要と言える。もっと言えば、IT ツールを使って情報をやりとりするスキルだ。

そこで本節では、コラボレーションに使えるツールを大別した上で、かんたんに紹介する。

QWINCS

六種類に大別しており、これを **QWINCS（クインクス）** と呼ぶ。以下の略である。

- Q&A
- Wiki
- Issues
- Note
- Chat
- Sticky Board

私たちが思い浮かべるのは Teams や Slack などのビジネスチャットだろうが、これは Chat でしかないし、デフォルト・リモートの実現には全然足りない。他にまだ五つもある。最適なコラボレーションの仕方はケースバイケースだが、最適を考えるためにも、まず手札を知っておかなくてはならない。

以降からひとつずつ解説する。ただし理解のしやすさを考えて、QWINCS とは異なる順番にする。内容としては、実例も挙げていくが、各ツールの考え方を知ってもらうことを重視したい。コミュニケーションであれば、集まって喋ればどうとでもできていたが、（本節で述べているツール利用前提の）コラボレーションはそうもいかない。ツールごとの世界観を理解した上で立ち回らなくてはならないし、全く新しい考え方もするので、そもそも知っておかねばならない。知っておけば、応用もしやすいはずだ。

Chat（チャット）

- Q&A
- Wiki
- Issues
- Note
- Chat □ここを解説
- Sticky Board

概要

チャットは古典的に知られるものであり、大衆への普及は LINE が担ったと思う。しかし LINE はビジネス用途には不十分だ。Slack や Teams といったものはビジネスチャットと呼ばれ、ワークスペースやチャンネルやスレッドといった分類の機能がついている。細かいコンセプトは製品ごとに異なっていて、Slack では他製品との連携と過去メッセージの検索、Teams は同社製品との連携、Discord は音声通話の拡充だ。

チャットは比較的にかんたんに使えるし、コミュニケーションの延長でわかりやすくもあるからか、パンデミック時に重宝した。それ以前も主に IT エンジニアやクリエイターは Slack を、ゲーマーは Discord を使っていたが、特定分野向けのニッチなツールだった。現在ではビジネスの前提となる汎用ツールのポジションにいると思うし、特に Teams の単語はニュースでも見かけるほど。つまり現時点で知名度も高い。チャットは QWINCS の中では唯一、一般層にも下りてきているものだ。

さて、チャットは **時系列指向** である。すべてのメッセージにはタイムスタンプ（投稿日時）がついていて、最新の投稿が新しく表示され、時系列で並ぶようになっている。これを表示する見せ方はタイムラインと呼ばれる。構造としては、**今現在やりとりされている内容があって、事実上空気となっている**。現実でも空気は読まないといけないが、チャットでも同様で、全く関係のないやりとりを差し込んだり、論点が 6 個あるからといって全部書いたりするのは望ましくない。一応、過去のメッセージはさかのぼって読めるが、まともに読まれることはほとんどないし、結局空気があるので蒸し返しづらい。時系列的な世界では空気が支配する。もっと言えば、空気を制御できる者が支配する場となり、支配者がボトルネックになるし、支配者に取り入るための政治も必要となる。あるいは明示的な支配者がおらず、牽制しあって形骸化するケースも多い。

加えてチャットは閉鎖的にもなりやすい。極端なのは DM など 1 対 1 の個別チャットであり、そこでやりとりされた情報は二者以外には見えない。そうでなくとも、私たちは心理的に公開範囲は狭める傾向があり、主に必要最小限の人数だけ集まったプライベートなグループで密にやりとりをする。村社会を誘発すると言ってもいい。自分の村だけ気にしていれば済むという点では楽だが、情報の透明性は皆無に等しい。先進的な会社であれば「社員全員が集まるチャンネル」のようなオープンなチャンネルもあるが、時系列である限り空気は存在する。オープンであるだけマシだが、焼け石に水だ。デフォルト・リモートには全然足りない。

そんなチャットにも、わかりやすさ以外に優れている点が一つある。通知だ。チャットは私たちの

仕事の動線となっており、メンションをつけるなどして通知を飛ばすこともできる。各自の自律性に委ねるのではなく、意図的に気付かせることができる。

立ち位置

デフォルト・リモートでは、チャットはオプションの手段でしかない。全く使わなくても不思議ではないくらいだし、少なくともメインで使うものではない。より厳しく言えば、**チャットがメインの手段であるうちは、デフォルト・リモートなどできやしない。**

有効な場面は以下の 3 点だ。

- 1: 声を掛ける
- 2: センシティブなやりとりをする
- 3: 雑に情報を残す

1: の声掛けは、現状「通知機能を持つ動線」となっているので、気付かせたい場合にチャットでメンションを飛ばせばいいという話である。ただし乱用は通知地獄につながるし、現代でもすでにそうなっている。本当に重要な呼びかけと、どうでもいい宣伝の通知が混ざっていたりする。デフォルト・リモートでは相当注意深く使わねばならないが、チャット自体にその機能はない。

2: のセンシティブなやりとりとは、機密情報やプライバシーなど社内への共有がまずい情報があるということだ。そういうものは対象者だけに宛てて送るしかない。DM や関係者だけを集めたグループチャットは引き続き有効である。一方で、気軽に使えるので、それ以外のやりとりでも使われてしまいがちなので、上手く制御せねばならない。この点もチャットの能力だけではカバーできない。

3: については、アイデアやメモなど雑に情報を残したい（脳内ではなく書き言葉として外に出したい）場合に、チャットの世界観がやりやすいということである。情報を書く手段としては、後述するようにもっと優れたツールがあるわけで、これらを使わないのは正直意味がわからないほどなのだが、無論知らねば使えないし、知っててもスキルが低ければ腰が上がらない。一方、チャットであれば、すでに慣れているから使いやすいし、実際構造としても QWINCS の中では最もシンプルだ。自分の場所にアクセスして、ひとつだけ存在するテキストボックスに書き込んで、投稿ボタンを押すだけ。内容も同じ画面で、時系列で表示されるだけ、と一画面で完結している。これを筆者は **シングルエントランス** と呼んでいる。出入り口がひとつだけ、とわかりやすいのである。

筆者の体感だが、（オフィスワーカーやエンジニア、ゲーマーやクリエイターも含めて）一般人のリテラシーはシングルエントランスが限度だと感じる。これ以上複雑になると、勉強や訓練を積まないと使いこなせないし、わざわざそんなことはしない。何なら仕事の文脈でも渋るほどだ。シングルエントランス以上を扱えるのは「マニアック」の域である。そういう意味で、一般人に使ってもらいたいのなら、シングルエントランスレベルのシンプルな UI は必須だと思う。チャットはオワコンだが、シングルエントランスとしてはわかりやすいため根絶は難しい。結局、デフォルト・リモートでもそれなりに付き合わねばならない。現実的には QWINS（QWINCS の C 以外）にシングル

エントランスを組み込むか、あるいはチャットの内容を、生成 AI を使って QWINS に流し込むかになると思う。

(余談)チャットと通知の分離、そもそも分離について

個人的にはチャットに代わる新しいツールが必要と考える。

声掛けを行うためだけの、小さなツールだ。たとえばベルウェア(Bellware)はどうか。呼び鈴などのベルから来ており、ベルを鳴らして「ここを見て」ができるだけのツールだ。チャットのようにメッセージを打つことはできず、「ここを見て」の「ここ」は別ツール(たとえば QWINS)上でつくらねばならない。あくまでもベルを鳴らす(鳴らしておく)だけなので、すぐに反応する必要はない。

もし緊急性が高い場合は、ベルウェアとは別のアラートウェア(Alertware)を使う。アラートウェアは、「来たらすぐに応答しろ!」レベルの緊急的なものだけを扱うツールだ。常時目に付くところに置いておくなり、携帯なりしておくべきだが、普段は何も起きないはずだ。

いかがだろう。従来ベルもアラートもチャットでいっしょくたになっていたものを、ツールとして分けてしまうのである。これなら通知地獄からは回避できる。ベルは自分のペースで確認し、アラートは仕方ないがすぐに反応する、と棲み分けできる。ベルやアラートの使い方が下手くそなら意味はないが、それは使い方の問題なのでどうとでもできる。デフォルト・リモートでは各自が各自のペースで動くので、アラートのような強い割り込みは極力排除しなければならない。いっしょくたになったチャットでは不可能だが、分けてしまえば可能となる。

このように用途ごとにツールを分ける考え方は重要である。現実の家事や料理でも、たくさんの小さなツールを使い分けると思うが、同じことだ。ツールが明快で、かつ慣れてしまえば、10 個だろうと 100 個だろうと使い分けられる。従来は原始的なコミュニケーションにて何でも力業で何とかしていたが、デフォルト・リモートはそうじゃない。特定の場面に対応するための、小さなツールを使い分けた方が上手くいく。

ツールというと現在は IT ツール、要はアプリを思い浮かべるだろうが、アプリは電子的であり、家事や料理で使う物理的なツールよりは馴染みづらい。そういうわけで、理想を言えば、ベルウェアやアラートウェアといったツールもハードウェア化して、物理的に使えればいいのではと思っている。たとえばデスクにベルとアラートがひとつずつ置いてあって、一目見て状況がわかるとか、ボタンを押して該当メッセージを開くとかできればいい――が、ここまで来ると妄想の域なので、これくらいにしておく。

Wiki(ウィキ)

- Q&A
- Wiki □ここを解説
- Issues
- Note

- ~~Chat~~
- Sticky Board

概要

ウィキと言えば Wikipedia が有名だが、原義はハワイ語のwikiwiki（急いで）から来ており、ブラウザで複数人でページを編集できる素早さを謳っていた。一般人には縁がないか、ゲーム攻略 Wiki などマニアックな情報共有の文脈で知っているくらいだろうが、逆に IT エンジニアにとっては身近な情報源だと思う。仕事の一環でウィキに書く人も少なくないのではないか。

世代としては以下のように三世代まであると思う。

- 第一世代
  - 原始的なウィキ
  - オープンソースで開発されている
  - 例: Pukiwiki、Mediawiki（Wikipediaのベースとなっている Wiki エンジン）
- 第二世代
  - エンジニア向けに最適化されたウィキ
  - 企業向け製品として提供されている
  - 例: esa.io、GROWI、Qiita Team
- 第三世代
  - 今のところ、Cosense のこと
  - 個人 or 企業向け製品として提供されている
  - 例: Cosense

他にも Teams や GitHub、その他プロジェクト管理系の製品がウィキ機能を持つことがあり、製品によって第一か第二かは異なる。基本的には第一的で、GitHub など IT エンジニア向けの製品なら第二なこともある。第一と第二の違いは IT エンジニアレベルでの使い勝手であり、チャットでたとえるなら第一が Teams、第二が Slack のようなものだ。

最大のメリットは、静的なページを素早く編集できるところにある。特に現代は豊富な処理や見栄えを重視した体験が多いが、ウィキはむしろ逆行していて、（特に第一世代は）質素だ。その分、読むのも書くのも素早くて、チャットとは比較にならない。しかし書き手への犠牲も大きく、マークアップ記法という専用の記法を使いこなすことになる。現代では Markdown が使われるが、第一世代時点ではまだなく、独自記法であることが多い。

チャットの項にてシングルエントランスを述べたが、ウィキはそうじゃない。記法の習得、書くモードと読むモードの切り替え、ページが多数存在していて、リンクで構造化もされていて、どこに何があるかを把握して、適切な情報を適切なページに書き込むという整理の思想 etc――要求されるリテラシーは高い。

これらウィキの限界を突破したのが第三世代であり、Cosense である。Cosense では、慣れたら学生やずぼらな人でも育てていけるほど簡潔に使える。またリンクを積極的に書いてネット

ワークを育てられる（育てやすい）点も、ここまで述べてきたとおり。細かな操作感やカスタマイズ性にも優れていて、現在でも頭一つ飛び抜けている。ウィキは、実用的には後述のノートに取って代わられているが、Cosense はノートにも負けない使い勝手を実現できている。

さて、ウィキは **ナレッジ指向** である。ナレッジとは、ここでは「全員に役立つような情報」を指す。話題 A があったとき、ウィキではページ A をつくって書くわけだが、ではページ A には何を書くか。共同編集だからといって皆が好き勝手に書いていいかというと、ノーだ。ナレッジを書かねばならない。

実際、ウィキの運用では、誰がどこを更新するかといった管理が敷かれがちだ。管理があるということは管理者が存在することも意味する。そこがボトルネックになるし、そこを取り合うための政治が生まれる。それでもチャットしか使えないよりははるかにマシだが、デフォルト・リモートを支えるほどの情報共有手段にはなりえない。唯一、Cosense は抗える余地があるが、やはりウィキであり、公式もナレッジ指向（は筆者の言い方だが）を推奨している。

立ち位置

デフォルト・リモートにおけるウィキの使いどころは、意外と難しい。

情報共有の面では後述するノートに取って代わっているし、短文であればチャットが強い。ウィキはポテンシャルこそ最強であるものの、**自由に使えすぎるせいで、組織で上手く運用させるガバナンスを利かせるのは難しい**。下手に利かせると、それこそ管理者が幅を利かせてしまって形骸化するし、そもそもシングルエントランスが限度の大多数にマスターしてもらうこと自体が難しい。仮にすべてクリアできるとすれば、十分に優秀で、ノリも似た少人数による運用であるが、それで上手くいくのはウィキに限ったことではない。デフォルト・リモートの実現を目指すなら、少人数だろうと大人数だろうと、それこそ千や万の従業員であっても成立させねばならない。

ウィキには何かが足りない。情報の構造化と素早い読み書きはもうある。あと何が足りないか――筆者が思うに、リードであろう。情報を持つすべての人が、すべてを出せるようにする。そのための仕組みと役割をつくって、無理なく自然に出してもらう。このあたりの話はマネジメントの章で取り上げた。具体的にはトピック指向（1-話題 1-部品）の前提に立った上で、ファシリテーションとしてギビング（お題を与える）とリマインドを、またコミュニケーションとドキュメンテーションに代わる第三の概念としてスプレディケーション（発想法的な情報の出し方）を論じた。そのとおり、ただの書き方や構造にとどまらず、マネジメントをどうやるかにも切り込まなくてはならなかった。

現時点で使えるとすれば、何らかの小さな単位ごとにウィキをつくって、その文脈ではそのウィキを見てもらうという使い方である。これを **Wiki As A Workspace（ワークスペースとしてのウィキ）** と呼びたい。つまり、社員全員が恒常的に読み書きするような「全体的な場」としてのウィキではなく、チームや各用途ごとにつくるわけだ。ビジネスチャットはすでにそうなっている。Teams にせよ、Slack にせよ、全社員が一つのチャンネルに入ってそこでやりとりするなんてことはなく、適当な単位に分ける。ウィキも同じようにすれば、比較的上手くいく。もちろん、それら

小さなウィキ同士をまたいだ情報共有をどうやるかの課題は残っており、ここに踏み込むとなると、上述のとおり、何らかのリードが必要となる。

まだ答えはないし、色々考えられよう。たとえば前述の役割分担を踏まえるなら、チームのウィキ内の情報をチーム外に共有する「共有役」をつくればいい。仮に従業員 2000 人の会社で、1 チーム平均 10 人だとしたら、200 のチームつまりは 200 個のウィキがあるはずで、共有役も 200 人となる。共有役 200 人のみが参加するウィキを一つつくるのはどうか。もう一つを挙げると、生成 AI には llms.txt という潮流がある。人間ではなく生成 AI が情報を読むために、ウェブサイトのコンテンツ全部をひとまとめにした llms-full.txt、またはそれだと長過ぎるならリンク先を網羅したリンク集 llms.txt をつくっておくというものだ。各ウィキは、外に見せられない情報は省いた上で llms-full.txt をつくって、公開すればいい。生成 AI がそれらを読めば活用できる。ちなみに、このやり方はチャットでも有効だ。要はコンテンツすべてを一ファイルにまとめて公開するだけだからだ（厳密には少々ルールがあるが割愛する）――と、このようにやり方は色々ある。

いずれにせよ、Wiki As A Workspace 的なウィキ間を連携する視座が要る。できれば連携は皆に任せるのではなく、仕組みとして提供したい。それこそツールとして提供できればベストだ。これをチャットで行ったものをビジネスチャットと呼んだ。であれば、ウィキについても **ビジネスウィキ** なるものがあるはずだ。

Note（ノート）

- Q&A
- ~~Wiki~~
- Issues
- Note □ここを解説
- ~~Chat~~
- Sticky Board

概要

ノートは近年よく見られるコラボレーションツールであり、QWINCS では最も後発となる。特徴はリアルタイムな共同編集であり、一つのページやノートに複数人が集まって、同時に書き込める。ウィキとは違って、記法を書かせるのではなくグラフィカルに書かせる。たとえば箇条書きを書きたい場合、ウィキだと - や * のような記法を書かねばならないが、ノートだとツールバーとして箇条書きアイコンが提供されていて、範囲選択した上でアイコンを選ぶことで適用できる。記法で書くモードと、記法に従って表示する「読むモード」を切り替える必要もなく、常に読むモードで編集する。これを WYSIWYG という。ウィジウィグと読み、What You See Is What You Get（見たまま）という意味だが、小難しいので無視していい。

WYSIWYG まわりの動きを振り返っておきたい。Microsoft Word も WYSIWYG であるよう

に、一般的には WYSIWYG は当たり前であり「小難しい名前をつける意味がわからない」と思うかもしれない。

しかしコンピュータはそうではなく、プログラミングがまさにそうであるように、コンピュータ向けの記法で書いて、それをコンピュータに解釈させる世界観となっている。人間向けの読みやすい見せ方は、必要に応じてつくればいい。となると「書くモード」と「読むモード」は最初から分かれていることになり、前者、書くモードではコンピュータ向けの記法を使うことになる。が、コンピュータ向けだと人間には書きづらいので、できるだけ人間でも書きやすいように進化してきた。その叡智として、現在普及しているのがいわゆる Markdown である――のだが、正直まだまだ人間には扱いづらい。このような記法をマークアップ言語と呼ぶが、マークアップ言語自体が人間には優しくなく、IT エンジニアやライターなどプロが使うものの域を出ないし、何ならプロであっても使えない人は多い。ウィキの限界もこの点にある。

幸いにも、技術の発展に伴って、このモードを分けるやり方をせずに済むようになってきた。手元で、個人でファイルとしてつくる程度なら Word も含め昔からあったが、この域を超えて、コラボレーションとして使えるようになってきた。それがノートである。整理すると、次のようになる。

- 1: WYSIWYG、だが個人でファイルとしてつくるだけであり、コラボレーションできない
- 2: Wiki、コラボレーションは可能だが、マークアップ記法が難しくて万人向けではない
- 3: Note、コラボレーションが可能な WYSIWYG

つまりノートとは、**コラボレーション可能な WYSIWYG なのだ**。たとえば Microsoft Word もオンライン版が存在しており、クラウド上に置いたワード文書を複数人で同時編集できる。本質的には、WYSIWYG でつくっていた文書にコラボレーション機能がついただけだ。そういうわけで、ウィキが持っていた素早さはない。むしろ素早さを犠牲にしてでも、コラボレーションの利便性を優先しがちだ。優れたツールなら動作も軽快だが、それでも多機能になりがちである。しかも、クラウド上にデータを配置するから（共同編集まわりの制御も含めて）複雑な通信が発生する。本章序盤で述べたネットワークの性能がなお要求される。

ノートの例をいくつか挙げる。おそらく何十と存在しており、これらはごく一部にすぎない。

- Microsoft Word Online
- Google ドキュメント
- Box Notes
- Notion
- Dropbox Paper

Microsoft Word が馴染み深い人は Word のように「ファイルが」存在するモデルが身近だろうが、ノート自体はファイルを意識させないことが多い。たとえば Notion ではページをつくるが、これはあくまでクラウド側に保存される単位であって、手元にファイルとして存在するわけではない。一方で、Box Notes のように手元でファイルとして同期するが、中身はただのショートカットであり実質クラウド側のページにアクセスさせるものもある。というわけで、**手元でファイルとして保持するというメンタルモデルを捨てねばならない**。ウィキはファイルとは無縁のモデルなので

捨てやすいが、ノートとなると捨てづらい人が多いと見受けられる。ファイルの考え方を捨てると、たぶん理解しやすいはずだ。

さて、ノートの本質だが、実は一箇所に集まることである。話題Aを表現したいとき、チャットであればAについて書いて誰かに送るし、ウィキならページAをつくってナレッジを書き込むだろう。また文書としてやりたければ、A.docxのようなファイルをつくることになる。ノートはどれも違う。ノート上に、Aというページをつくる。ここまではウィキと同じだが、このAというページはただのページではない。皆が集まってリアルタイムに書き込める **場** なのだ。

ゆえに筆者は **ギャザー指向** と呼んでいる。ノートAをつくるとは、話題A目的で集まるための場をつくることに等しく、ただのノートというよりはワークスペースとでも言えるだろう。軽量ワークスペースでもいい。実際、何十何百というノートをあちこちにつくって、話題Aはこっちでやろう、話題Bはこっちでやろう、話題Cは混んできたのでページDをつくってこっちでやろう、といった立ち回りになる。また、不自由なく立ち回れるようにするために様々な装飾や連携、埋め込みや貼り付けもサポートされる。本質的に多機能になるわけだ。

立ち位置

ギャザー指向のノートは、デフォルト・リモートにはうってつけだ。チャットでは時系列的であり、過去の情報や大量の情報を共存させるのが不可能であったが、ノートならできる。たとえ1000ノートであろうとも、ノート自体は気軽につくったり消したりできるし、リンクをつければ行き来もできるのだからどうとでもなる。整理の問題にすぎない。

問題は、その整理が難しいことだ。やりようはいくらでもあるし、別に雑でもいいのだが、チーム全員（あるいは社内全体の情報共有であれば社員全員）に適用するのが難しい。ある人の整理の仕方は、当然だが他に人に刺さるとは限らないし、従うためにはその整理の仕方を学ばねばならない。誰が好き好んでそんなことをするのか。かといって、トップダウンでこうしろと強要していては、ウィキの項でも述べたように管理的な世界となる。この問題を何とかするのは、これもウィキでも述べたが、連携の視座が必要であろう。ノートとノートを連携する役割や仕組みをつくる。

加えて、ある一つのノートを取ってみても、その中で誰がどこにどのように何を書くかという問題もある。**ノート内部の** 整理の話だ。ウィキではナレッジ指向であるから、ナレッジだけを書けばよかったが、ノートはそうもいかない。というより何を書こうが自由なのである。無論、だからといって適当にやれば済むものではない。ノート間の連携という目線での整理を **ノート外部の整理** と呼ぶことにして、外部の整理であれば、実はそんなに難しくはない。（管理者の価値観を敷いた支配になるが）階層的な分類は昔から知られているし、ウィキの項でも「連携役が集まったウィキ」はどうかとのアイデアを論じた。やりようを考えやすい。

しかし、内部の整理では、ノート一つ分の領域のみが存在するという制約の中で整理を考えなくてはならない。これはかなり難しいし、正直言って知らないと無理だ。そもそもそんな理論やツールは現状ない。ないからつくらねばならない。本書の域を越えるため割愛するが、以前の著書で扱ったことがあるので、興味があれば見てほしい（ブロックライティング）。あるいは、場のよう

な「コラボレーションのメンタルモデル」を廃して、純粋に情報をやりとりできるようにした何かに頼った方が早い。マネジメント章で取り上げたスプレディケーションは、このアプローチだ。いずれにせよ、まだ現代にはない仕組みと体系をつくらねばならない。

というわけで、結論としては、ノートにもデフォルト・リモートをまかなえるポテンシャルはあるものの、整理の問題でつまづくと思われる。ウィキの項ではファシリテーションと書いたが、ノートでは整理だ。このニュアンスの違いは重要である。整理とは、構造に注目した言葉であり、所定の構造に従うことを要請する。読み書きのスキルの節で挙げた構造化スキルと、特にノートに落とし込む営みができねばならない。何度も言うがこれは難しいし、無理だと思うので、事実上ノートだけでは無理だ。

Sticky Board（無限キャンバス）

- Q&A
- ~~Wiki~~
- Issues
- ~~Note~~
- ~~Chat~~
- Sticky Board □ここを解説

概要

この手のツールとして Miro が有名だろう。古くからアナログでもよく使われているし、ブレインストーミング、通称ブレストは本書の読者層であれば誰でも知っていよう。特にデジタルツールとして実現したものを無限キャンバス、またはデジタルホワイトボードと呼んだりする。QWINCS としては Sticky Board としているが、正直言えば語感を良くするためのこじつけにすぎない。付箋そのものを Sticky note と呼ぶし、Windows の公式アプリも Sticky Notes と名付けられているものの、無限キャンバス＝付箋ではない。付箋に限らず、画像やリンクなど様々なコンテンツを載せられる点はノートと同じである。また、その名前のとおり、二次元空間上に無限に伸ばせるし、ボード自体も複数枚つくれる上、コラボレーションツールなのでリアルタイムな同時編集もできる。

ツールの例を挙げると、Miro の他には Microsoft Whiteboard、postalk、Box Canvas などがある。個人が無料で使えるものは案外少なく、企業向け製品が多い。また Zoom が有名だと思うが、会議中に使える機能として搭載されていたり、前述のノートや、本書では取り上げないが作画ツール上でも同じようなことができたりもする。概念自体は決して目新しくはないが、無限キャンバスはあくまでのコラボレーションツールである。チャットやウィキといった、普段のやりとりをフルカバーできうるポテンシャルを持つほどのツールを指す。現状、このレベルに至っているのは、Miro と postalk くらいだろう。筆者の知らないツールが他にもあると思うが、まだまだ未成熟のジャンルと言っていい。

さて、無限キャンバスの本質だが、**空間指向** とでも言える。無限キャンバスは二次元だが、実は三次元も同じだ。VRChat やマインクラフト、その他バーチャルオフィスやメタバースやその他オープンワールドなゲームでも何でもいい。空間がででんと一つだけあって、どの座標に何を配置するかという立ち回りをする。現実世界がまさにそうであるように、中心（ゼロ座標地点とは限らない）に機能が集まり、中心に近いほど地価が上がる現象が起きる。格差も生じやすい。ただ、デジタルゆえに融通を利かせられる余地は大きく、世界（無限キャンバスで言えばボード）自体を複製したり、テレポートしたり、あるいはマイクラのネザーにように距離を圧縮した世界をつくったりできるが、それでも中心優勢の世界観からは逃れられない。

立ち位置

デフォルト・リモートとしても、無限キャンバスをメインのコラボレーションツールに据えるには厳しいものがある。空間指向から生じる中心優勢の格差がどうしても合わないし、そもそも必要な情報はいつでもどこでも誰にでも届くようにするべきであって、あえて空間的制約を入れる意味がない。

しかしスポットで使う分には優れていることもある。すでにそうであるように、何らかの作業（個人もあれば協働もある）――特に発想法を伴うような創造的なことをやる分には、直感的で使いやすい。ただ現状では拘束と司会を伴った状態で、有限時間で高密度に仕上げるとの使い方しか想定されておらず、日常的にボードをつくって、いついつまでに各自書き込んでおいてね的な使い方はおそらく通じまい。ノートの項で挙げたように、自由すぎてとっかかりが得られないか、管理がきつすぎて形骸化するという **自由と管理のジレンマ** に陥ってしまう。無論、拘束前提で創造的にコラボレーションできる点は優れているが、デフォルト・リモートはそうじゃない。せいぜいスポットであるべきで、それをメインにしてはいけない。あくまでも非同期的な情報共有としてのやりとりをメインにするべきだ。無限キャンバスでは、ノートと同様、制約が足りないのである。

もう一つ、無限キャンバスが特に優れているのは **楽しさと定着のしやすさ** である。

マイクラがまさにそうだが、空いた空間に誰でも自由に建築ができたり、それを好きなときに眺められたりする。仮にマイクラをバーチャルオフィス扱いして、日常的に使うとすれば、私たちは楽しみながら日々を過ごせるだろう。同様に VRChat も、筆者は未経験なのだが、おそらくアバター越しのロールプレイを楽しみやすいと思う。Miro のような二次元でも、事例は聞いたことがないが、楽しめる文化を何かしらつくれるはずだ。いずれにせよ、楽しさは原始的なモチベーションであり、ツールを使わないと話にならないという問題をクリアしやすい。もっとも現実的には、楽しいものにはそれだけの習熟コストと機材コストが必要なので厳しい。

次に定着のしやすさについては、土地勘や空間認識といったものを使って空間的に覚えやすいということだ。ウィキやノートでは、テキストで書いたタイトルとその接続が論理的に存在しているだけだった。苦手な人にはとことん苦手だし、マジョリティ自体も慣れてなくてシングルエントランスくらいが限度である。一方で、空間的な世界であれば、苦手な人はどのみちダメだが、現実で当たり前に使いこなしている人は多いし、多少不器用でもそのうち慣れる。無限キャンバスが、

仕事上の「地元」になる。

というわけで、無限キャンバスは空間指向としての強みこそあるものの、中心優先の格差からは逃れられないし、キラーアプリもまだないし、とデフォルト・リモートとしてはイマイチである。あるいは現状と同様、スポットで密にやりとりする際のオプションの域に留まるだろう。筆者としても妙案は思いつかない。一方で、リモートでバーチャルにコミュニケーションを注入する用途としては使えると思う。特に役割分担の章で述べたロールプレイは、仮想世界の方がおそらくやりやすい。

Q&A

- Q&A □ここを解説
- ~~Wiki~~
- Issues
- ~~Note~~
- ~~Chat~~
- ~~Sticky Board~~

概要

Q&A は Q&A サイトとして知られている。有名なのは Yahoo!知恵袋だろう。Stack Overflow のような技術者向けのサービスもあれば、Quora のように知識（カッコよく言えば教養）を重視したものもある。ただの電子掲示板やフォーラムとは違って、質問を書き、それに対する回答を書くという設計が意図的に存在する。また BA（ベストアンサー）の概念があり、質問者は回答の中から最も良かった回答を選べる。全体としてはゲーミフィケーションが敷かれており、ベストアンサーをたくさん取る人はポイントも高くて、ランキング上位になれる――といったものだが、残念なことに生成 AI により下火になっているし、そもそも **企業向けのコラボレーションツールとしては開拓されてすらいない**。

Q&A を成立させるのは難しい。前提として FAQ とは異なる点に注意したい。FAQ はよくある質問とその回答を事前に準備したものだが、Q&A は利用者がいつどんな質問をするか、また回答者がいつどこにどんな回答をするかがわからない。そんな Q&A を成立させるには、質問も回答もそれなりに盛り上げねばならないわけだが、ここがまず難しい。**数の力が要る**。to C 向けのサービスが成立するのは、インターネット全体にリーチできるからだ。企業内でやったところで、物理的に数が足らないのは目に見えている。1000 人でも足らないし、1 万人ですら足りまい。ただでさえ忙しくて回答を書く暇なんてないし、見るモチベーションも薄いわけで、盛り上がってなければわざわざ見に来る理由もない。

次に、こちらの方が深刻だが **エンタメ化と管理のジレンマ** が生じる。すでにQ&A サイトがそうであるように、何もしなければエンタメ性の高いネタ――たとえば X(Twitter) やはてなブックマークでバズっているようなネタが盛り上がることになる。私たち現代人の大半がそうした刺激の強い

ネタに慣れきって麻痺していることは周知の事実だと思うが、同じことが Q&A でも起きる。では、そんな状態で、果たして仕事に関する真面目な Q&A をやりとりできるだろうか。かといって、これを禁止しようと管理やガイドラインを強化すると、今度は使われなくなって形骸化する。

最後に、**ナレッジの再利用と結びつけてしまう点も課題** だろう。一番わかりやすいのは「質問する前に過去ログを確認しろ」というもので、これは過去の質問と回答がすでにあるんだから、まずはそっちを調べて自力で解決しろと言っている。当然ながら大半の人はそこまで面倒を負わない。日本の、転職サイトで「真面目な社員が多い」と書かれるような会社の社員であれば、かろうじて通じるかもしれない。そうでなくとも、質問者は過去に同じ質問があったらどうしよう、回答者は同じ回答を誰かがしていたらどうしようとの意識が頭をよぎる。つまり過去の情報を気にしてしまって、使うことに集中しづらい。

と、成立の難しさを三点ほど挙げてみたが、コラボレーションツールとして成立させるにはこれらをクリアしなければならない。たとえばシームレスの章で取り上げた Rapid Q&A は、三点目のナレッジ再利用を排除した提案となる。全社員全員を一つのチャットチャンネルに集めて、誰でも何でも質問回答できるようにしたものだ。ナレッジ再利用は要らない、とのルールを敷けば質問や回答はしやすくなる。また、エンタメ化についても、雑談を禁止するか、質問できる内容を仕事や会社ネタだけにするなど限定的にすれば、ある程度は防げる（それでも社内ニュースネタでエンタメ化すると思うが）。数の力については、単一のチャンネルに全員を押し込めることで確保している。流速の速いタイムラインがたった一つしかないので、社員全員はそこをウォッチするようになる。シンプルだが、変にカテゴリーなどでページを分けるよりははるかに効果があるはずだ。

さて、Q&A は **アンサー指向** だ。質問を書いて、それに対する回答を書くことに特化している。質問者が満たされたかが重要であり、ベストアンサーによって実現する。要は質疑応答なので、他のコラボレーションツールほど汎用性は無いように思えるが、質問の仕方次第だろう。たとえば Quora では（知識を残すのが目的なので）自問自答も推奨されているし、Q&A からは外れるが、サイボウズ株式会社は企業理念の一環で「質問責任と説明責任」が出てくる。汎用性は十分だと考える。

ちなみに、この構造を一般化したものが、次で述べる Issues である。Q&A は質問に対して回答をぶら下げるものだが、Issues では話題に対して情報をぶら下げる。

立ち位置

現状コラボレーションツールとして開拓されていないため、可能性を論じることとする。Q&A は、デフォルト・リモートを推進する起爆剤になりえる。質問と回答という概念がわかりやすく、情報のやりとりとしては良い制約になるからだ。本書では情報のやりとりを加速させるために様々なテネット、やり方や考え方、道具を見てきたが、質問と回答という考え方は端的だ。わからないことは何でも質問し、わかることは何でも回答できるようにすればいい。仕事とは情報のやりとりであり、やりとりは質問と回答であると捉えてしまうと、仕事とは Q&A であるとすら言える。コミュニケーションは別途（Q&A に縛られない形で）注入すればいい。Q&A によるやりとりは常時適用してもいいし、それがつらいなら、ロールプレイとして Q&A モード（質問と回答しかできないモー

ド）をつくって、道具として使えるようにしておくといいだろう。

とはいえ、生成 AI でカバーできるならそうしたいところだ。機械なら休む必要がないし、質問に回答が追いつかないなんてこともない。そのためには会社やプロジェクトといった固有の文脈も AI に教えねばならず、文脈をどれだけ情報として記述し AI に放り込めるかが重要となる。シームレス章で述べたコミュニケーション 3.0 はまさにこのことを言っているし、本章でもウィキの項で llms.txt を紹介した。

おそらく今後、私たちはわからないことを何でも生成 AI に聞いておくような世界観になるだろう。聞いておくと、裏で AI が調べてくれてまとめてくれる。私たちはあとで読むか、リマインドしてもらえばいい。そしてこれは Deep Research としてすでに現実になりつつある。筆者も Q&A サイトと生成 AI の関係（仮説：下火だよね？）を知るために、ChatGPT の Deep Research で依頼している。回答が来る間も本文を書き続け、合間に読んで「お、結果来てる――うん、だよな、下火だよな」などと確認ができた。そういう意味では、人間の方こそがボトルネックなのだから、リマインド（あるいは一般化するとタスク管理）こそが重要になるのかもしれない。余談だが、タスク管理については、前作タスク管理を噛み砕くにて、これでもかとカバーしたのでよろしければぜひ。

Issues

- ~~Q&A~~
- ~~Wiki~~
- Issues □ここを解説
- ~~Note~~
- ~~Chat~~
- ~~Sticky Board~~

概要

比較的古くから使われてきたコラボレーションツールとして Issues がある。Issues という言い方は筆者の命名だが、チケットのことだと思えばいい。主に問い合わせ対応やタスク管理の文脈で、対応事項を「チケット」として発行し、その件についてはそのチケットのページ内でやりとりしてきた。このようなあり方のツールを Issues と名付けた。

IT エンジニアの中では GitHub Issues が知られてている。あるいは Redmine などプロジェクト管理ツールも古くから存在する。Zendesk のようなチケット型の問い合わせ形態は、開発者でない人も見たことがあるかもしれない。問い合わせというと昔はメールであり、それがチャットや SNS など対話的なあり方になってきているが、ここにチケット型も加わっている。といっても、主に B to B の文脈であり、仕事でなければ目にする機会はまだないかもしれない。

特徴はチケットが **タスクである** という点であり、したがってオープン（未完了）とクローズ（完了）の概念が存在する。クローズしたらそのチケットは用済みだ。そういうわけで本質的にはタスク管

理であり、どのチケットに誰がアサインされているかとか、チケットを並べて状況を俯瞰するとかいった機能とよくセットになる。現代のプロジェクト管理はまず備えているだろうし、GitHub もまさに俯瞰するための機能（Projects）をリリースして久しい。

私があえて Issues という名前をつけたのには理由がある。GitHub Issues の果たした役割があまりに大きかったからだ。GitHub Issues は、軽快な動作とシンプルな UI により、ノートやメモとして使えるポテンシャルすらあった。実際、エンジニアの中には、備忘録やコミュニケーション用途で使う人もいるし、筆者もそうしていた時期がある。つまりタスク以外にも、汎用的に情報全般を扱えるポテンシャルがある。実際に GitHub も、Issues をベースとした GitHub Discussions なる機能を追加している。

Issues の本質は **トピック指向** だ。トピック指向自体は本書マネジメント章でも述べたが、一言で言えば 1-話題 1ページ で扱うあり方を指す。Q&A を一般化したものと言ってもいい。Q&A は、話題を質問に固定したもの（ついでに言えば回答だけを書き込めるようにしたもの）にすぎない。チャットにせよ、ウィキやノートにせよ、一つのエリア（チャンネルやページ）には様々なコンテンツを書き込めていたが、Issues は違う。いや、Issues でも別に何でも書き込めるが、元がチケット的で、あるタスク一件のみを扱う空気感が強い。その延長で、ある話題のみを扱うとの力が働きやすい。加えて、タスク的に完了の概念もあるので、用が済んだ話題は完了にしてしまえばいい。そうして、必要に応じてトピックをガンガンつくって、終わった議論やタスクは完了にしていく――本書が述べてきたトピック指向も可能となる。

立ち位置

Issues の概念、もっと言えばトピック指向はデフォルト・リモートに不可欠である。あるいは他のやり方もあるかもしれないが、本書ではマネジメントを廃した上で、トピックを単位にしたやりとりを自律的に行うことで実現するとの立場を取っている。トピック指向は重要であり、現状この思想に最も近い Issues は、QWINCS の中でも最重要と言っていい。仮に筆者がデフォルト・リモートを啓蒙するとしたら、GitHub の導入と Issues の啓蒙をまず先にやる。エンジニアではない人も向けて、社員全員に啓蒙するし、そのためのコストも厭わない。

とはいえ、GitHub Issues だけでまかなうのは難しい。まず、ものがタスク管理であるから、カジュアルなやりとりや大量のやりとりがしづらい。たとえば、ひとりで 300 件の話題を抱えるのは難しいだろう。タスク管理ができるなら可能だが、万人に求めるのは酷である。次に、Issues ではトピック同士のつながり（ネットワークの構築）ができない。この点は、ウィキの項で述べた Cosense に分がある。あるいは、コラボレーションではなく個人で取るノートの話になるが Obsidian も有力だろう。つまり、読み書きの営みの延長としてリンクをつくれねばならない。これらのツールは、テキストを `[ ]` や `[[ ]]` で囲むだけでリンクをつくれる。特に Cosense は既存のトピック名（ページ名）を自動で補完してくれるので、なおつくりやすい。記憶もそうであるように、本格的に情報を扱うとなるとネットワークは現状無難な解だと思う。ネットワークで捉えるためには日頃からつくりこむ必要もあるわけで、その作業がリンク（トピック A から B に接続を張る）である。リンクのしやすさが重要なのだ――が、Issues は現状この視点を備えていない。

もっとも、マネジメント章でも述べたように、ここも生成 AI が置き換えるかもしれない。人間がちまちまネットワーク構造なぞつくらなくても、全部生成 AI に食わせてしまえばいい。だが現状、AI に雑に情報を食わせただけでは、満足のいく情報は取り出せないだろう。結局、AI から情報を取り出す（ための指示を出す）のは人間であり、情報の構造は人間自身もある程度は押さえておかねばならない。ネットワークをつくる営みは、その役に立つ。

ただ Issues はそもそもネットワークを扱えないため、使うとすれば、トピック指向の最初の一歩（1-話題 1ページでのやりとり）に慣れる用途だろう。それだけも価値がある。現状はそもそもトピック指向すらまともに普及していないのである。トピック指向自体を試して慣れる用途として、Issues は役に立つ。

端的な取り組みとして、イシュー・デー（**Issues Day**）を使うといい。これはコラボレーションツールとして Issues のみを使うというもので、特に **チャットを使ってはならない** とする。仕事全体で難しければ、特定のプロジェクトだけとか私生活のある作業だけなど限定してもいい。とにかく、チャットを廃して、Issues だけで成立するように頑張る。つまり現状チャットがメインの手段となっているところを、Issues に置き換えてみるわけだ。

こうすると必然的に、1-話題 1-ページの思考回路でやりとりをせねばならず、どうやればいいかもわかってくる。課題も見えてくるだろう。デフォルト・リモートはその延長上にあるわけで、あとはそれら課題を解決していけば自然と到達していける。ツールは GitHub Issues をおすすめするが、チケット型のツールなら何でもいいし、ノートやウィキで行ってもいい。ただ、GitHub Issues ほど軽快なツールもそうはないから、下手に迷うくらいなら GitHub Issues でいい。必要なら勉強や練習の時間も取ってほしい。

(余談) コラボレーションツールへの投資

費用負担は会社にしてほしい旨を先述したが、これはコラボレーションツールでも同様である。コラボレーションツール≒SaaS となるだろうが、社員全員に支給するべきだし、それができずとも必要なときに自由に調達できるようにしたい。

よくセキュリティや決済を理由に承認制を敷くことが多いが、これらツールも文字どおりの IT リテラシーの要であるため省くべきだ。省いても成立するには、を考えなくてはならない。

まずは既存のフロントオフィス的な情シスでは権限的に難しいため、前章で述べたような特殊な体制が要る。個人的には情シスを細分化して、全社員向けのコラボレーションツールのガバナンスのみを行う組織をつくるのがいいと思う。オープンソースを統括する OSPO（2 Open Source Program Office）は知られているし、シームレスの章でも BMPO（Badge Management Program Office）を取り上げたが、そんなイメージ。CTPO（Collaboration Tool Program Office）だろうか。無論、ワーカーの理では上手くいかないだろうから、メタワーカー的なポジションの方が良い。

その上で、使用や決済の履歴を透明にして誰でも見れるようにする。透明性があるということ

は、いつ誰に見られているかわからない、また悪いことをしてもその証拠が残るということであり、抑止力になる。不安ならそれら記録を監視する役をつくってもいいし、現代ならAIに解析とレポートをお願いすることもできよう。

センシティブの分離も押し進めたい。たとえば雑談的に情報を残す「雑残」であれば、機密情報を混ぜずにやりとりしやすい。初期の取り組みとしては有効だろう。そうして慣れてくると、仕事のやりとりさえも行えるようになる。たとえば他者や顧客も巻き込んだコラボレーションもやりやすくなる。

またツールを自由に試せる余地も欲しい。私有のクレジットカードで有償プランを契約し、業務で使うことを容認する。費用は社員自らが立て替え、会社に申請することで取り戻せるようにする。この考え方は交通費や出張費ではすでに身近だと思う。この方が融通が利くのである——とはいえ、筆者は法的には素人であり、現時点で現実的に取り組めるかは正直わからない。詳しい人の議論もうかがいたいところだ。

私有で試せるようになったとしよう。だからといって機密情報をガバガバ入れてしまうわけにはいかないため、センシティブの分離はやはり重要である。やりとりする情報すべてを監視するわけにはいかないし、そんなディストピアなどあってはならないが、それで成立させるためには結局従業員を信頼するしかない。従業員各々のリテラシーが必要となる。キツイ管理よりもゆるやかな監視（透明性）、承認よりも記録、センシティブの分離といったことを底上げせねばならない。

一方で、私有のツールを上手くガバナンスする方法も、つくれないことはないと思う。以下にいくつか述べておく。

- ランダム・モデレーター
  - 私有でつくったワークスペースに、ランダムに選んだ他の社員も招待する
  - モデレーターがいつ見ているかわからないので抑止力になる、また偶発的な交流や学びも生まれる
- 強制同期
  - ツール上でやりとりしたデータを必ず会社側のデータベースに同期（アップロード）させる
    - ツール側の連携機能を使って強制同期先と連携することを必須にする
  - 同期先を監視できるようになるし、同期されること前提で使うことになるので透明性やセンシティブ分離の意識も働きやすい
  - ただ現状はツールごとに個別実装が必要となり大変なので、この考え方は業界レベルで標準化したいところである

現実的には、全社員にマスターしてもらうよりも、マスターしている社員（Master）としていない社員（NYM, Not Yet Mastered）に分かれるだろう。Masterは自由にコラボレーションツールを使えるが、NYMは使えないし使わない。たとえばNYMは他律的で、出社や会議も多くて、コミュニケーションを主に行うような人材となるだろう。デフォルト・リモートでも多様性は大事だ。それでいい。ただ、情報のやりとりと役割分担の仕方を工夫して、どちらも尊重しつつ仕事を成立させればいいだけだ。NYMの理に寄せてMasterを潰してはいけない（現状はこれだ）し、逆にMasterの理に寄せてNYMを苦しめてもいけない。

まとめ

- デフォルト・リモートでは IT ツールを用いた読み書きが必須
  - IT ツールを用いたリテラシー（読み書き能力）が文字通り求められる
  - これを「文字どおりの IT リテラシー」と呼んだ
- 環境面
  - 作業空間。特に生活空間との分離と、必要に応じて四つのプレイスを使い分けること
  - デスクまわり。クリアデスクが望ましいが、個人差がある。アタッチ（機能）とデコレーション（装飾）は各自で最適化する
  - ハードウェアとスペック。PC、ネットワーク、周辺機器も軽視してはならない
- 読み書きのスキル
  - 言語化スキル。語彙の量、質、そしてお互いの認識をすり合わせていくリカバリ
  - 構造化スキル。章節項や起承転結は有名だが、ある程度の文章は構造化しないと読めない。書き手が負うコストでもある
  - 処理スキル。脳内で処理するスキル。≒才能
  - 出力スキル。≒タイピングスピード。読み書きにおいては思考力そのものなので速ければ速いほど良い
  - 整形スキル。エディタを使いこなすスキルはわかやすいが、キャレット（カーソル）を操作するスキルは盲点
- コラボレーションツールを扱うスキル
  - 主なツールとして QWINCS の 6 種類があり、それぞれの世界観や癖を理解しておきたい。チャットだけでは話にならない
  - QWINCS = Q&A、Qiki、Issues、Note、Chat、Sticky Board
- 環境とツールは重要どころか前提なので、会社として投資したい

Chapter-10 あとがき

出社 vs リモートの議論は不毛です。リモートが良いに決まっています。それを住み込み → 出社 → リモートとのパラダイムシフトとして主張し、デフォルト・リモートと名付けたうえで、じゃあどうやって実現していくかを議論してきました。新しい概念ばかりつくってしまいましたが、内容自体は決して難しくありませんし、おそらく似たことは世界中で色んな人が断片的に考えているはずです。本書の意義は二つあると思います。一つは、断片的なそれらを整理した上で、わかりやすく提示したことです。もう一つは多様性を組み込んでいる点であり、既存の出社派やコミュニケーション派も尊重した上で、上手く共存するためのあり方をつくっています（それゆえ煩雑となってしまいましたが）。

散々述べてきたとおり、私たちは「コミュニケーションありきのあり方」に凝り固まっています。本書では丁寧に、かつ容赦なく解体してきたつもりです。解体を達成できたのなら、筆者としては言

うことはありません。また筆者としては、情報をいかにして伝えるかのみを考えておりますので、読書の読み方にとらわれず読むのも歓迎しますし推奨します。たとえば ChatGPT など生成 AI の力を借りて要約したり、質問をぶつけたりできるでしょう。本書は無料のオンライン書籍であり、コピペもしやすいはずです。

個人的な話ですが、本書はポートフォリオの側面もあります。私は仕事術や働き方の開発・啓蒙を仕事にしたく、また世の中としても誰かがやらねばならないと考えていますが、未開拓なジャンルを、無名の個人が、会社員の傍らで切り開くには限度があります。しかし、このような視座を持ち投資してくれる酔狂な人などそうはいません。私の立ち回りが下手くそなのもあります。そのとおりです。私に並の要領があったならとうに起業でも何でもしているか、それができる仲間をすでに得ているはずですし、リモートであるべきだなどと働き方に執着することもなかったでしょう。逆を言えば、そんな私だからこそ本書は生まれました。私にできることは、自分のすべてを込めて、自分なりに言語化したものをぶつけ続け、そのフィードバックや議論も受け止めて、またこねて、ぶつけて、を繰り返すことだけなのです。その先の行動と結実は、それができる人に任せます。私はその人たちを動かすためのゼロイチをつくります。幸いにも売り込める先は皆無ではないですし、読者の中にもいらっしゃるかもしれません。本書は、その人たちに向けたポートフォリオでもあります――というわけで、もし雇っていただける方がいましたら、ご連絡をお待ちしております。もちろん単に話を聞いてみたいとか、本書らしく非同期的にやりとりしてみたい等でも何でもお待ちしております。

2025/03/23 吉良野すた

Chapter-11 参考文献

参考文献は章ごとに記す。b や c がついているのは、本文からジャンプするための便宜であり意味はない。なお URL は本文から直接リンクする。

まえがき

なし

全体像

b1

さあ、才能(じぶん)に目覚めよう 新版 ストレングス・ファインダー2.0 / 2017 / トム・ラス (著), 古屋博子 (翻訳)

b2

Schein, E.H. (1978) Career Dynamics: Matching Individual and Organizational Needs.

- ただし、この時点では、現在使われている分類までは導かれておらず、8 つの分類で知られる現在のあり方は後年整理されたものである

b3

異文化理解力――相手と自分の真意がわかる ビジネスパーソン必須の教養 / 2015 / エリン・メイヤー (著), 田岡恵 (監修), 樋口武志 (翻訳)

コミュニケーションと情報共有

c1

R.I.M. Dunbar. (1992) Neocortex size as a constraint on group size in primates.

c2

ティール組織 ― マネジメントの常識を覆す次世代型組織の出現 / 2018 / フレデリック・ラルー (著), 嘉村賢州 (著), 鈴木立哉 (翻訳)

マネジメント

d1

リーン・スタートアップ　ムダのない起業プロセスでイノベーションを生みだす / 2012 / エリック

リース (著), 井口 耕二 (翻訳), 伊藤 穣一(MITメディアラボ所長) (解説) (その他)

d2

Snyder, M. (1974) Self-monitoring of expressive behavior.

d3

大事なことに集中する――気が散るものだらけの世界で生産性を最大化する科学的方法 / 2016/12/9 / カル・ニューポート (著), 門田 美鈴 (翻訳)

d4

フレックスタイム制のわかりやすい解説＆導入の手引き / 2019 / 厚□労働省・都道府県労働局・労働基準監督署

- フレックス制を解説した情報として提示した。フルフレックスとは「コアタイムのないフレックス制」の通称であり、政府として正式に定義した概念ではない。スーパーフレックスと呼ばれることもある

d5

ゆとりの法則 ― 誰も書かなかったプロジェクト管理の誤解 / 2001 / トム・デマルコ (著), 伊豆原 弓 (翻訳)

いつも「時間がない」あなたに:欠乏の行動経済学 / 2015 / センディル・ムッライナタン (著), エルダー・シャフィール (著), 大田 直子 (翻訳)

d6

超AI時代の生存戦略 ―― シンギュラリティ<2040年代>に備える34のリスト / 2017 / 落合 陽一 (著)

d7

パーティーが終わって、中年が始まる / 2024 / pha (著)

- もくもく会は同書著者の pha が始めたものだという。説明自体は 1 ページもないが、もくも

くのニュアンスは同書全体に漂っていると思う
- もくもく会の概念は、わかりやすいようで意外と難しい。IT 界隈ではすでによく使われていると思うが、ただのカジュアルな勉強会や発表会に成り下がっている。違う。pha ではなく私の持論だが、むしろ図書館や自習室のような私語厳禁が近しい。本質は「交流はしないが存在はしている」ことであり、「ひとり」と「交流が可能な場」の中間に位置するものだ
    - 余談だが、このように存在（Presence）を扱う潮流は近年見られる。私はプレゼンシングと呼んでおり、サービスの例として Remotty や Teracy がある。本書、後の章で 6 ジャンルのツールを取り上げるが、プレゼンシングツールも今後ジャンル化するのではないかとさえ思う。もくもく会は、このプレゼンスの概念を理解するためのとっかかりとして重宝する

シームレス

e1

マッキンゼー緊急提言 デジタル革命の本質: 日本のリーダーへのメッセージ / 2020 / マッキンゼー・デジタル・日本

e2

DX実行戦略 デジタルで稼ぐ組織をつくる 2019 / マイケル・ウェイド (著), ジェイムズ・マコーレー (著), アンディ・ノロニャ (著)

- 記事レベルでは検索すればすぐに見つかる。参考までに ChatGPT o1 pro の Deep Research に調べてもらった結果 も共有しておく

e3

ストレスフリーの仕事術―仕事と人生をコントロールする52の法則 2006 / デビッド アレン (著), David Allen (原名), 田口 元 (翻訳)

e4

Software Teaming: A Mob Programming, Whole-Team Approach / 2022 / by Woody Zuill (Author), Kevin Meadows (Author)

- 2022 年と新しいが、上記書籍は 2nd Edition のようだ。原典時点では物理的に一台

のコンピュータに集まるスタイルであり、リモートな現代では足りない部分も多いが、コーディングスキル鍛錬の Dojo や、椅子やレイアウトといった作業環境の最適化など本書が重視する視点（鍛錬面と環境面）も含まれる
    - 余談だが、デフォルト・リモートでモブワークをする場合、リモートベースで鍛錬と環境をどう整備するかが重要となるだろう。本書本文では議論していないが、おそらく整備してからモブワークを始めるよりも、モブワークをしながら共有や議論を重ねていく方が良い。最初は思うように行かずストレスフルだろうが、そこも含めて全員でフルコミットするわけである。需要とポテンシャルもありそうだし、すでに知見を貯めているチームもいると思う。次の研究テーマにしてみてもいいかもしれない
- ちなみにAgile Allianceが提供する11ページのPDFもあり、こちらはアップデートされていて、たとえば消毒や在宅勤務など感染予防の観点、参加強制ではなく個人の選択に委ねる旨なども盛り込んである

役割分担

f1

演劇入門　生きることは演じること / 2021 / 鴻上尚史 (著)

- 同書曰く、俳優の仕事は傷つくことであり、心を無防備にして演技をする。また『練習が終わった後、恋人役をやった俳優に「お前、いつもあんな風に恋人といちゃついてるの？」とニヤニヤしながら近づいた人間がいたら、僕はすぐにその人に向かって「もう稽古場に来なくていい」と告げます』とも書いている。俳優でさえそうなのである

f2

エンジニアのためのマネジメント入門 / 2023 / 佐藤 大典 (著)

f3

ソフトウェア・ファースト / 2019 / 及川 卓也 (著)

f4

令和3年度産業経済研究委託費 イノベーション創出加速のためのデジタル分野における「ニューロダイバーシティ」の取組可能性に関する調査 調査結果レポート

文字通りの IT リテラシー

g1

Ray Oldenburg. (1999) The great good place : cafés, coffee shops, bookstores, bars, hair salons, and other hangouts at the heart of a community.

g2

深層考察：『オープンソース プログラム オフィス』 組織構成、役割、責務、および課題 / 2022 / Ibrahim Haddad, Chris Aniszczyk, **翻訳：工内隆・伊達政広**（The Linux Foundation Japan）