SEのための

開発現場で役立つ箴言集

佐野 洋（さの ひろし）
現在、フリーコンサルタントとしてPMファクトリーを主宰。数十年にわたりPOSレジのファームウェア開発やプロマネ業務に従事。好きな言葉は「Boys be ambitious」。信条は「弱き者も強き者も共に手を携えてその生涯を生き抜くこと」。
Web http://www.pmfactory.ambitious/

#01 見積もり回答は契約なのだ

見積回答書の基本要件は、わかっている内容のみを見積もり、わかっていない内容は見積もりに含まれていないことを明記しておくことだ。

#02 目には目を、歯には歯を

「目には目を、歯には歯を」は、物ごとに対する対応の原則のひとつである。短時間の雑な見積もり要求に対しては、山ほどの条件付き〝超々〟概算見積もりしか出すことはできない。

#03 結果をあせるな、利をあせるな

ものごとを成し遂げる場合には必ず守るべき手順がある。特にその基本となる部分に手抜きや検討不足があると、物事は無事には終わらない。あせりは我々の正常な判断を狂わせる。

#04 今日やるべきこと、明日やればよいこと

今日やるべきことは今日中に済ませること。〝明日やる〟は、ほとんどやられない可能性が高い。

#05 当たり前は難しい

当たり前なことは簡単なことと同じではない。簡単なことが複数集まれば難しいことになる。そのことを理解していなければ簡単なことで簡単に失敗をすることになる。

#06 とりあえず高いという人種

賢い開発マネージャは見積もり担当者に対して最初から決して高いとは言わない。これ安すぎないのか？何か抜けや見落としがないの？金額の高低よりも内容の妥当性を検証する。

#07 許容範囲 ～どこまでも許されるわけもない

量、質、時間、いずれにも許容範囲というものがある。的の中心に当たらなくても的のどこかに当たれば合格とされる。そこで注意しなければならないことは、自分勝手に的の大きさを変えないことだ。

#08 他人の評判に従って自分の行動を決めてはいけない

他人の評価で自分の行動を決定することは、自己の自律性の放棄につながり、妥当性のある合理的な思考や行動を妨げることになる。

#09 投げることよりも受けること

一方的に話をしても伝えられることは少ない。相手が何を必要としているのかを話し合いの中で探り、相手の必要としているものを先に与えてこそ自分の伝えたいことが伝わる。

#10 フェイス・トゥ・フェイス

重要な問題についてのコミュニケーションの鉄則は、当事者同士による対面直接コミュニケーションに拠るべきであり、決して電話やメールなどで済ませるべきではない。

#11 直接話をするということ

IT技術は情報通信を飛躍的に進化させたと言われているが、その進化の実態は間接コミュニケーションを急速に拡大した反面、人々の間における直接コミュニケーション能力を著しく落としている。

#12 物事は、事前の準備があれば失敗はしない

万一失敗があった場合の対処方法とは、現代用語でいうところの「コンティンジェンシープラン」に相当し、どのような異変にあっても失敗しないだけの工夫とは現代用語でいうところの「リスク回避策」にあたる。

#13 教養の幅は人間性の幅を広げる

情報の理解度の深さや広さのことを教養という。今まで見聞きしたこと、経験したこと、学習したことの量と質が自分の理解力や教養の深さや広さを決定する。

#14 わからない質問には無理に答えるな

相手の質問の意味がわからないときがある。自分の理解力の低さが原因だと思ったならば、何がわからないのかを相手に伝え、教えを乞う。もしそうでないと判断したら沈黙し話題を変えるしかないだろう。

#15 仕様調査法のいろは

まずは仕様の全体像の把握から始める。疑問点・不明点を発掘し、その解消に向け要求元に対して積極的にアプローチする。仕様検討の段階で要求者と徹底的に仕様を検討する。

#16 ぶれない軸 ～コンセプトの効用

コンセプトはものごとの全体像を一言で表したもので、そのものごとの本質を表している。基本コンセプトを最初に明確にしておくことは不要な間違いやリスクを避ける効用がある。

#17 傑出することよりも高度の平凡さを

何かをやろうと思えば、たとえ他の人より出遅れていたとしても遅すぎることはない。肝心なのは、それを継続することにある。

#18 大事なことを聞き逃している ～空気読みの空気知らず

相手の言葉が理解できない場合、それに先立つ自分の発言に独りよがりがなかったかどうか点検してみることだ。相手のメッセージがわからないこと自体が重要なメッセージである場合もある。

#19 アジャイル開発の真髄

Agile（俊敏な）開発は、コミュニケーション重視の人間中心のプロセスである。ウォーターフォール開発の原点も最初はそのようであった。

#20 どこまでも譲れるわけもない（時間編）

限界を超えた要求をする者は破綻の責任を負う必要があり、破綻することを承知のうえで実行した者も共に責任を免れ得ない。

#21 日本語文章のあいまいさを避ける

主語がない文章には主語を書き加える。可能なかぎり短い文にする。並列的な表現は箇条書きにする。時間軸にて整理したチャートで現在・過去・未来の時制を明確にする。

#22 不条理な要求への対抗策

自分の弱みにばかり気を取られていると敵が見えなくなる。敵の弱みにも注目するところからタフな交渉が可能となる。

#23 納期優先とは見切り発車のこと？

〝納期優先〟は実に嫌な言葉だ。時間がなくなった挙句に必要な手順を抜くことを暗に示唆している。もっと最初の段階で腹をくくってしっかり取り組んでおけばよいものを。

#24 溺れるものは必ず藁をつかむ

焦っている場合、本来の段取りを無視して枝葉末節なことがらに先に手をつけがちになる。切羽詰ったときにつかんだものはやはり藁が多い。

#25 できないのなら、できるようにする

転ばぬ先の杖。忙しいから焦って急ぐのではなく、早い段階で準備をしておくことだ。多くの時間不足の問題は生産性が低い問題ではなく、準備のタイミングが遅すぎるところにある。

#26 「俺たちに明日はない」のか？

楽が手抜きとなり、手抜きが失敗を生み、それが自分の評価を下げ、しまいには仕事を失い大きな苦難に陥るということが理解できないような残念な人や組織にはなりたくない。

#27 過酷な状況を乗り越える

危機的なプロジェクトにおいて、笑うに笑えない状況の中でも笑えば、また新たな知恵が浮かぶであろう。眉間に寄ったしわを伸ばそう。

#28 笑えない状況下で笑えるか

あせりは失敗を生み、笑顔はたとえ演技でも気持ちに余裕を生み出す。

#29 仕事の最初にやるべきこと

プロジェクト開始時に、最終的に自分が望む結果を描いておき、時間を逆に遡れば、何をいつまでに実行すべきかはっきりする。これができる人は成功の確率が高いが、できない人は失敗の確率が高い。

#30 データドリブン ～データに基づく仕事を

データに基づいて開発を遂行するということは、過去の自分ないしは自チームのQCDなどのデータおよびそれらから導き出された次の目標データに基づいて仕事をするということである。

#31 「なぜ？」の問いかけは

コミュニケーションの質を深める。その結果、個人とその相互作用を活発にし、チームの自己組織化を促し、顧客との協調を深め、変化への俊敏な対応を可能にする。

#32 喉元過ぎれば熱さを忘れる ～失敗に学べない

皆でやりとりした情報もすでに散逸してしまっている。障害報告には、××機能のプログラムミスについての簡単な説明および修正・評価済みの記述があるだけ。

#33 自分に対する評価を捨てるべし

他人からの評価で動くのではなく、自分の頭で考え、自分が仕事でなすべきことを自覚し、自分で目標を設定し、自ら行動すること。

#34 どこに向かって歩いているの？

達成すべきあるいは達成したい目標が明確でなければ目標は達成できない。明確な目標はチームの力を結集させ、無駄な活動を防止し、プロジェクトの効率的な活動の原点となる。

#35 塵も積もれば山となる

ひとが見向きもしないような小さなものでも有益なものを積み上げていけば社会全体を動かすような巨大な益を生み出す。塵も積もれば山となると言うが、積み上げるものは塵ではなく人の知恵だ。

ソフトウェア開発工程で起こる問題や課題への対処法を、格言を交えて紹介しています。好評公開中！
URL http://gihyo.jp/dev/serial/01/engineer-proverbs

SE稼業は忘己利他（もうこりた）
――現場に転がる箴言集

SEのための

開発現場で役立つ箴言集

佐野 洋（さのひろし）
現在、フリーコンサルタントとしてPMファクトリーを主宰。数十年にわたりPOSレジのファームウェア開発やプロマネ業務に従事。好きな言葉は「Boys, be ambitious」。信条は「弱き者も強き者も共に手を携えてその生涯を生き抜くこと」。
URL http://www.pmfactory.jp/pmfactory_ambitious/

#36 腐ったリンゴをどうするか

雑とは手抜きのこと。手抜きの原因は主に時間不足によるあせりや本人のモチベーションの低下によるところが多い。

#37 死出の旅路

困難な登山の行程を行き抜くためには、第一に食糧や備品などの装備が必要だ。開発においては適切な開発費・期間・体制の確保などの事前準備にあたる。

型というものは、ものごとの試行錯誤の結果、無駄なものをそぎ落とした普遍的かつ合理的な一連の動作のことだと言える。我流は人の成長をある低いレベルで止めてしまう。

#38 型より入れ

何かに行き詰ったとしても人は今までやってきたことをなかなか変えることができない。例え、その行動が非効率だったとしても。

#39 現状維持の法則

設計書がなければソースコードの解析による開発となり、まちがいなくその設計品質は劣化する。設計品質が劣化すれば製造品質も劣化する。

#40 設計書がありません

日本語の読み書きができれば文章を正しく理解できるわけでもないのと同じように、プログラム言語の読み書きができれば一連のソースコードの文脈を正しく把握できるわけでもない。

#41 ソースコードの文脈を読む

仕事の依頼者と請負者の間には相互に果たすべき義務がある。依頼者は自分の依頼するものを説明する義務があり、請負者はそれを実行する義務がある。

#42 「相互義務」があることを知る

プロジェクトの失敗はリーダーのせいである。ソフトウェア開発では、プロジェクトリーダーが七割、メンバーが三割と言っても過言ではない。

#43 いつもうまくいきません

自分の失敗責任は最小化し、他人の失敗責任を最大化するような人は自己の成長もリスクの回避も無理だろう。失敗の現実を直視できない人のリスクは高くなる。

#44 私たちの責任 ～責任の背負い方

期限のない約束は白紙手形。期限が明らかでない仕事を請けるということは、期限について依頼者に全権を委任したということだ。10分後に完成してくれと言われても反論はできない。

#45 精神主義ではどうにもならない

手順抜きは手抜きと言うこと。たまの息抜きは生産性を向上させるが、手抜きは品質を低下させ余計に手間がかかる。

#46 カラスの勝手な人たち

リスクは人にあり、モノやカネ自体にはない。実はあなた自身が最大のリスクなのだ。

#47 人は必ずミスをする ～リスクは人にあり

仕事の業務品質を保つための正常領域における複数の条件を一つずつ外していくと業務品質はだんだんと低下して異常領域に近づく。そしてある最後の条件を外したときに異常領域に陥ることになる。

#48 どこまでも譲れるものではない

イマジネーションは芸術の世界。ソフトウェア開発はロジックの世界であり、自分の勝手な想像で作り上げるものではない。確定仕様に基づかない想定開発は今すぐやめよう。

#49 それでもやめられない悪習慣

例えば、単体テスト項目が作れないという問題。何をテストすべきかの元ネタである詳細設計書をしっかりと作ること以外にない。目の前の炎を消したければ火元を消す以外に有効な方法はない。

#50 目先の問題に惑わされるな

単体テスト時に結合テストをやると言ってもやれるわけもない。単体テストに合格していないレベルのソフトウェアが結合できるわけもない。

#51 やるべきことをやるべきときに

レビューは構造物の基本的な構造や仕様の再確認を目的とする。細かなバグを見つけることが目的ではない。バグを見つけるのはデバッグであり、単体テスト、結合テスト、総合テストの役割である。

#52 レビューは銀の弾丸ではない

何百何千とあるテスト項目を一、二時間で全件レビューしようとすることなど、最初からできるわけもないとわかっているのに無理矢理やろうとする。レビューの本意は基本骨子の検証だけである。

#53 分別をわきまえよ

#54 顧客が望んでいるように

顧客からの要望といっても、顧客の言うことが本当にその顧客のためになるのか自分でも考えてみる必要がある。何でも言うことを聞くことが必ずしも顧客満足になるわけでもない。

#55 少しでもましな負け方を

大混乱状態に必ず陥る。このような結果になることが確実視されている場合に取るべき態度は、あきらめではなく被害を最小限にとどめようようとする意思と行動だ。

#56 共有分業と分離分業

各担当間で情報共有を行いきちんと一つの仕事にまとめあげることを共有分業と言う。一方、情報共有もなく、各担当がバラバラに行動し、まとまった一つの仕事を完成できないやり方を分離分業と言う。

#57 困っている者同士はよく譲り合う

富や知恵は持てる者から持たざる者へと譲りを行わなければ、その社会や共同体は確実に衰弱し最後には滅びることを知っておくべきだろう。

#58 カビ同士でも助け合う

人が集団や組織を作って仕事をするのは皆が得するからだ。組織の中にいて助けも助けられもせずに孤立状態にいることは愚の骨頂である。

#59 捨てる神あれば拾う神あり

余裕がないと他人を助けられないか。他人を助けるか否かは、余裕よりもその気があるかどうかで決定する。

#60 行動できないのは性格のせいではない

自分の感情はどうあれ、人はやるべきことはやれるのである。やるべきか、やらざるべきか、その選択権は自分自身にある。

#61 この世の渡り方

「お願いします」と「ありがとう」が言えればこの世は渡っていける。

#62 自律した人・組織

自律とは、何でも勝手にやれることを意味しない。他者、特に弱い立場の人々や組織とともに生きる姿勢が多くの人々の勇気とやる気を喚起する。

#63 連携プレー／コンビネーション・プレー

システムや組織がまともに機能するには、その構成要素や構成メンバー間の密接な連絡提携、結合、団結が必要である。雑な連携は雑なシステムを生み出す。

#64 重症患者の取り扱い方 ～トリアージ

黒・赤と判断された助かる見込みのない低レベルの成果物は直ちに提供者に返却しやり直しを要求する必要がある。いつまでもだめな提供物につきあって時間を浪費してはいけない。

#65 とりあえずの結論に飛びつく ヒューリスティクス

単純化ヒューリスティックとは近道の解決法とも言われ、目前の問題に対して誤るリスクはあっても、とりあえずの解答を出してみるという思考である。

未知のものをできるだけ正確に把握するためには事前調査に時間をかけ、チェック作業中においてはそこで得られた情報を小まめに記録し蓄積しておくことが必要となる。

#66 最初から完全を望まないこと

バラバラに散らばったものはシステムとして機能しない。分散したもの同士を一定のルールに従ってまとめ、ある目的として機能するようにしたものがシステムである。人も物も、同じ原理で動いている。

類似不具合を解消するには類似不具合とその解決法をまとめた「類似不具合集」を作成し、全員でそれを共有すればよいだけのことである。学ぶことで問題は確実に減る。

一定の割合で販売台数が増えていくビジネスプランは販売現場のコミットメントを取っていない創作であり、美しい不具合習熟曲線は後付けで作成されたものであり、いずれも虚偽のデータである。

#67 統合しないと機能しない

次の5つの特性を備えたものが良い商品の条件である。「信頼性、可用性（継続的稼動能力）、保守性、使用性（使いやすさ）、設置容易性」

ソフトウェア開発工程で起こる問題や課題への対処法を、格言を交えて紹介しています。好評公開中！

URL http://gihyo.jp/dev/serial/01/engineer-proverbs

#68 類似不具合の解消法

#69 逸脱の法則

#70 RASUI ～商品の命は信頼性だけではない

gihyo.jp

SE稼業は忘己利他（もうこりた）
—現場に転がる箴言集