

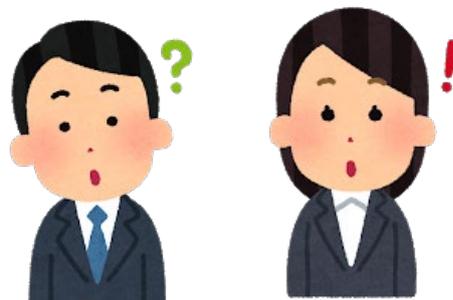


DX 時代の
品質を考える
金言コラム



DX 時代の新しい考えに基づく品質とは？

IT 業界・専門家による 9 編の品質金言コラム集



DXIT フォーラム

DX 時代の品質課題を考えるワーキンググループ (品質 WG)

内容

はじめに	3
対象読者	3
本書の構成.....	3
金言 1. 品質は儲かるのか？（五味 弘）	4
金言 2. ソフトウェア品質を気にする人が目を向けるべき本来の守備範囲は？（溝口 則行）	5
金言 3. IT システム品質の陰の立て役者（河野 太基）	6
金言 4. DX 品質こそユーザー目線で（齋藤 幸久）	6
金言 5. 価値と品質のはざま（五味 弘）	8
金言 6. 非機能要求と例外（山本 修一郎）	9
金言 7. DX 時代の品質を見直す時期に（三部 良太）	10
金言 8. DX 時代の成功のメトリクス「真剣度+参画率」（金子 博）	11
金言 9. 目玉の数さえ十分あればどんなバグも深刻ではない ～ リーヌスの法則（溝口 則行）	12
執筆.....	13

画像提供：いらすとや

本書の内容に関して

- ・本書の著作権は、DXIT フォーラムが保有しています。
- ・本書の一部あるいは全部について、著者、発行人の許諾を得ずに無断で改変、公衆送信、販売、出版、翻訳/翻案することは営利目的、非営利目的に関わらず禁じられています。詳しくは「書籍・刊行物等に関するよくあるご質問と回答」<https://www.ipa.go.jp/publish/faq.html> をご参照ください。
- ・本書を発行するにあたって、内容に誤りのないようできる限りの注意を払いましたが、本書の内容を適用した結果生じたこと、また、適用できなかった結果について、著者、発行人は一切の責任を負いませんので、ご了承ください。
- ・本書に記載した情報に関する正誤や追加情報がある場合は、DXIT フォーラムのウェブサイト <https://www.ipa.go.jp/digital/dx/dxit-forum.html> に掲載します。

商標

- ※本書に記載する会社名、製品名などは、各社の商標または登録商標です。
- ※本書の文中においては、これらの表記において商標登録表示、その他の商標表示を省略しています。あらかじめご了承ください。

はじめに

DX時代になり品質に対する考えも従来の考えからは大きく変わってきている。従来のバグや性能面などで障害を出さない守りの品質から、使う側や作る側、そして社会全体の価値を基準とする攻めの品質に変革している。

そこで本書ではこの攻めの品質にどう立ち向かうのか。そのための悩み・苦しみを一緒に考えていく。

そして品質を定義し確保し発展させるためにどうすればいいのか。これを読者ととともに一緒に実現するために本書を執筆している。

どうか本書のコラムを読んでいただき、一緒に戦っていこうではありませんか！

対象読者

DX時代になり、それに対応する新しい品質を求めている方に向けて、本書を執筆しています。品質に対して悩んでいる方から、品質実現に邁進されている方に向けて、本書を贈ります。

本書の構成

本書はコラムを記載しています。

またコラムの執筆陣が参加しているDXITフォーラム 品質WGが別に執筆している「DX時代の品質」読本も公開していますので、本書とともにそちらもお読みください。

金言 1. 品質は儲かるのか？（五味 弘）

ソフトウェアは新しい時代を迎えている。そのソフトウェア品質も新しい時代に対応する必要がある。従来のバグ密度で計測する機能適合性だけの品質でなく、非機能要件と言われた側にある性能効率性はもちろん、ユーザ利用時の品質である満足性も含めた価値そのものが品質である。なお SQuaRE^(*)にはこの手の品質特性が多数記載されているので見てほしい。

(*) SQuaRE (Systems and software Quality Requirements and Evaluation) システム及びソフトウェア製品の品質要求及び評価に関する国際規格 ISO/IEC 25000、別冊の DXIT 品質 WG 読本を参照

この新しい時代の品質には問題がある。ひとつはこの品質をどのように計測するかである。従来のバグ密度は確かに計測でき、第三者を納得させることもできた・・・と言いたいが今ではアジャイル開発が多くなり、バグは次の繰り返し開発のときに直せばいいということもあり、これでさえ納得させるのは難しくなっている。

さらに価値をベースにした品質は計測が難しい。いや中の人はいろいろと工夫して計測しているのだが、それを第三者に納得させるのが難しい。この問題を解決するためにここで議論することが始まった。しかし難しい。まさに「俺たちの戦いはこれからだ！」。

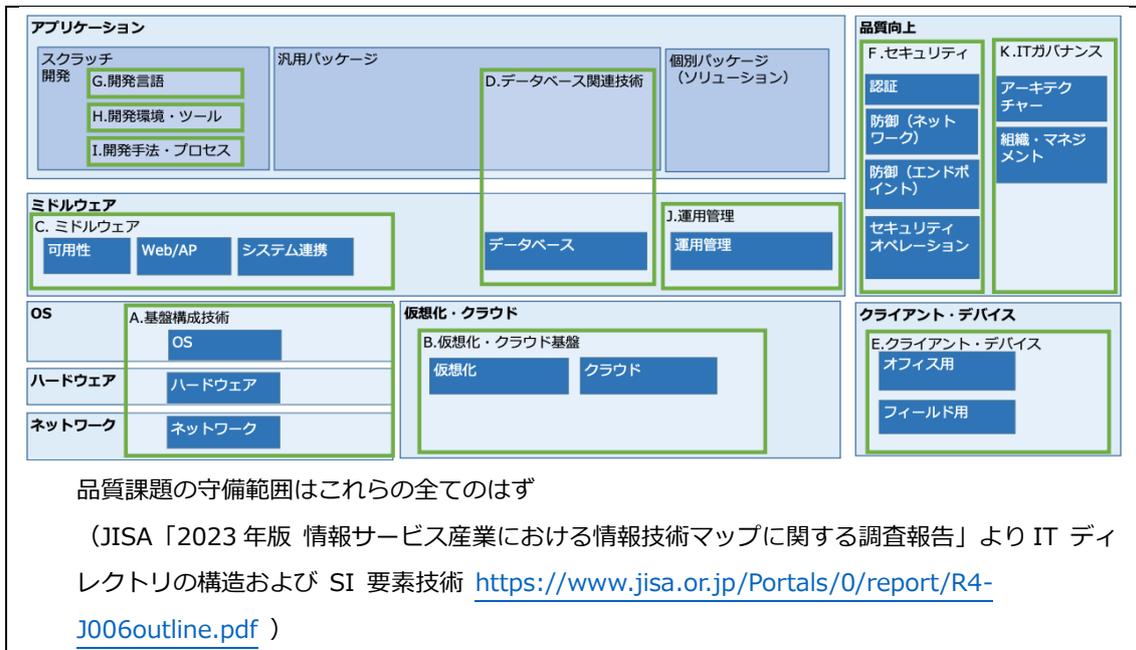
そして大問題がある。それはつまり、品質は儲かるのか？である。上記にもあるように品質で第三者を納得させることができなければ、儲けることも難しい。言葉を着飾って並べても納得させるのは難しい。ずばり数字で攻めなければ納得させられない。そしてそうでなければ企業活動でもなく工学でもない。

この新しい時代を私たちと一緒に苦しんで、そして楽しんで、向かっていきませんか？その仲間を私たちは待っています。いつでもいつまでも。



金言 2. ソフトウェア品質を気にする人が目を向けるべき本来の守備範囲は？ (溝口 則行)

昔々、メインフレームまたは汎用機と呼ばれる共通の稼働基盤の上にすべてのアプリケーションサービス(ソフトウェア)を稼働させていた時代、各サービスの品質の差は開発したソフトウェアの品質の差だったのではないかと。しかし、提供する IT サービスごとにハードウェア環境、ネットワーク環境、ミドルウェア、運用組織、さらにはユーザの価値観まで違う時代に、事前に決めた開発ソフトウェアの仕様とのズレを一生懸命洗い出すアプローチは、品質のほんの一部しか見ていないのではないかと。これはつい最近にそうなったわけではなくて、平成になった頃にはすでに変わり始めていて、21 世紀になった頃にはすっかり変容していたはずである。令和になったこの時期に、いまだに「品質 = ソフトウェアの守備範囲」だと扱われていることに憤慨しているのは私だけなのか。



「非機能要求グレード」(<https://www.ipa.go.jp/archive/digital/iot-english/jyouryuu/hikinou/ent03-b.html>)は明確に品質課題のテーマである。Google の SRE (<https://cloud.google.com/sre?hl=ja>)も提供する IT サービスの品質への取り組みである。インターネットでの大規模 DDos 攻撃(Distributed Denial of Service attack)も通信事業者の通信障害による影響も品質問題である。アプリのバグと比べて「どうでもいい問題」というわけではない。IT 屋の品質保証の建付けを基礎から作り直した方が良さそう。

金言 3. IT システム品質の陰の立て役者 (河野 太基)

われわれが IT システムを企画・設計・構築しようと考えたときに、最初に整理するのが機能要求だろうと思う。期間やコストなど様々な制限もあると思うが、その中でどんな目的でどんな処理をするシステムが欲しいのかをできるかぎり明確にするわけだ。機能要求はビジネス上の要求や目的そのものであり、これさえ明らかに出来ればシステムは出来そうだと思うのだが、そうでもない。

実はシステムを実際に具現化するためには、こうした機能を支える一定の環境や条件が必要になってくる。実はこうした機能要求を支える非機能要求がとても重要なのである。機能さえ満たせば処理に何時間・何日かかってもいい、セキュリティは度外視しても良い、将来にわたって拡張性は持たなくても良いとはなかなかならないだろう。

前ページのコラムでも紹介があった「非機能要求グレード」

(<https://www.ipa.go.jp/archive/digital/iot-en-ci/jyouryuu/hikinou/ent03-b.html>)

はまさにこの領域をカバーするために IPA から公開されている標準で、現在では英語や中国語にも翻訳されグローバルで使われている。これを利用するためのさまざまな解説やガイド群も充実している。本当に欲しいシステムを手に入れるために、安定したシステムを実現するために、できあがってシステムを見て思っていたのと違ったと落胆しないために、一度目を通しておかれてはいかがだろうか。



金言 4. DX 品質こそユーザ目線で (齋藤 幸久)

DX の名のもとに日々新たな IT サービスが生まれ出されている。アジャイル全盛、とにかく早く作って早く失敗しろ (fail fast) 等様々なことが言われているが、往々にして「作り手の論理」が優先されていないか。その (生煮えの) サービスを使う人の評価こそサービスの成否を決するということを今一度肝に銘じたい。

昔話であるが、とある企業が鳴り物入りで社内システムを刷新したことがあった。初期画面には責任者の顔写真付きの挨拶文。ところが...ログインしてもメニューが現れるまで早くて 5 分、運が悪ければ永遠に先に進めず、ひたすら責任者の御尊影を拝み続ける羽目に陥った。システムの容量設計が大外れしたが故の悲劇であり、クラウド時代ではこんなことは起こらないと信じたいが、かくして、そのシステムは新機能がてんこ盛りではあったものの、未永く社員から蔑まれ続けることになる。

ユーザが淡い期待を抱いてそのサービスに触れた際に、期待を裏切るような事象は徹底して排除されるべきである。情報漏洩は論外、ストレスなく利用開始でき、ユーザの期待通りに動くかどうかは初めの一歩となる。当たり前のように聞こえるが、悲しいかな、これができていないサービスは世の中に溢れている。想定ユーザ層 (例えば高齢者や主婦) がスムーズに初期設定を完了できるか? 等、ユーザに寄り添った姿勢を忘れないようにしたい。



金言 5. 価値と品質のはざま (五味 弘)

DX はデジタル技術を使って対象を変革し、そこに参加するエコシステム全体の価値を向上する。DX 時代の品質を見るときも、単純な機能品質だけでなく、まさにワインバーグの言うように品質とは価値であり、品質は価値を基準に考えるべきである。また価値の対象も IT システムだけでなくエコシステム全体を俯瞰し考え、さらに価値の有効な時間も 1 回だけ一時的でなく、持続し永続的な価値向上する品質を考える。・・・価値は金儲けに繋がるのだから。

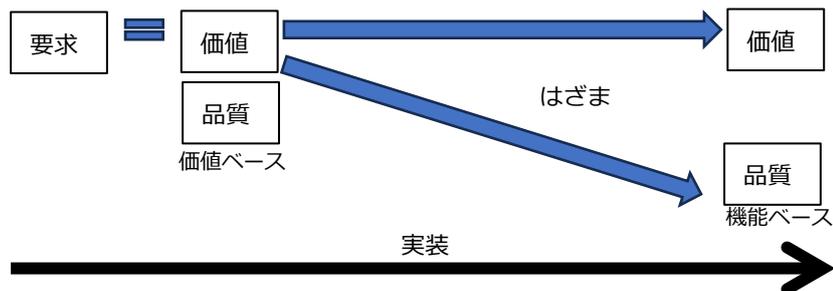
このように品質をエコシステム全体で永続的な価値向上できる能力と考えると、価値と従来の機能中心の品質には大きな隔たり、はざまがある。こうなるとはざまを埋めるにはコストが掛かり、価値と品質を結ぶ赤い糸は見つからない。両者を結ぶにはもう工学ではなく哲学になり、多くの金が必要になるだろう。・・・価値を生むには金が掛かるのはどこも同じだから。

これをエンジニアとして、また営利を求める企業として、はざまを埋めていく。これが我々のミッションであり、品質の神様から与えられた使命である。・・・使命を果たせば金が儲かるのだから。

原初の価値はユーザの要求であり、そこが価値と品質の開始地点であり、これが IT システムの実装に近づくほど、品質から価値が薄れ、価値と機能面が高まった品質のすきまは大きくなり、やがてはざまになる。これを逆に考えれば、ユーザ要求の段階であれば、要求から発する原初の価値と、まだ空想上の実装予定のシステムの価値ベースの品質は一体となっている。・・・迷ったらスタート地点に戻るのは迷子の鉄則だから。

しかしながらユーザ要求というものは油断できないものである。短期から長期的視点、一部から全体、小さいものから大きいもの、実装平易なものから夢物語など要求はてんでばらばらであり、そこから発する価値も千差万別である。またその価値すらも虚ろなものである。・・・要求の丸飲みは熱湯を飲むのと同じだから。

要求を解きほぐし、整理整頓清潔清掃(5S)し、価値を見出すのがはざまを埋める唯一のものである。もう一度、ユーザ要求を 5S して価値と品質のはざまを埋めていこう。



金言6. 非機能要求と例外 (山本 修一郎)

「仕事量を見誤ったか」生産能力を越えた仕事を受注した結果、品質検査工程で上司から手抜きを強要された社員による外部通報が発生して社会問題化した、ある自動車企業の経営者の言葉だ。この会社のグループ企業では、同じことを繰り返している。

ところで、企業の生産能力を越えた過大な注文を受ければ、所要量の製品やサービスを生産できないか、生産できたとしても製品やサービスの品質が低下するのは明らかである。



この経営者は生産能力という最も重要な組織能力の正確な値を認識していなかったことになる。正しい生産能力を知らなければ、注文数が生産能力を越えたかどうか分からない。逆に、生産能力の上限が分かっていたら、過剰な注文数を例外として検知することにより、それ以上の受注を制限できる。

同じことが情報システム障害の原因にもなることがある。たとえば、サービスを受付サーバと処理サーバによる分散システムとして構成することがある。処理サーバの処理能力には上限がある。処理能力を越えたサービス依頼を受付サーバで受理した結果、処理サーバが過負荷となってシステム全体が停止することになる。

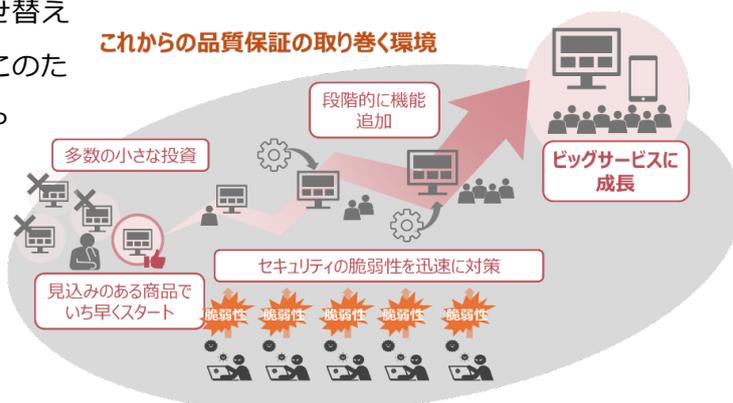
DXのめざす姿の一つにデータ駆動経営がある。企業の生産データをリアルタイムに収集することで、生産能力に応じて適切な受注判断ができる。分散型情報システムの例でも、後続サーバの処理能力を越える可能性を先行サーバ側で例外として検知できれば、システム全体の性能や信頼性などの非機能要求の低下を抑止できる。このように、システムの構成要素間の依存関係に基づく機能例外を考慮することによって、システム全体の非機能要求を達成できる。

金言 7. DX 時代の品質を見直す時期に（三部 良太）

IT システムが人力で行っていた作業を機械化していた時代には、「品質」とは、人がやっていた作業を間違いなく、高速に行うことであった。これは一つの品質の定義として意味があった。しかし、DX 時代になり、ネットワーク上に散らばっているリソースを利用して、今まで人ではできなかったことが実現できるようになってきた。その際に、上記の定義の品質で品質保証することに意味があるのだろうか？

例えば、ソフトウェア構成。丹念に品質保証された OS やミドルウェアの上にアプリケーションを開発していた時には、開発したアプリケーションに対してカバレッジやテスト密度で管理をしながらバグをつぶしていけばよかった。しかし、クラウド上のプラットフォームに OSS やネットワーク越しのサービスを組み合わせる構成になると、自分が作った部分だけがバグがないことを保証しても意味がなく、構成全体としてうまく動くことを考える必要がある。さらに、上記構成は開発時に作成したものが半永久的に固定されるわけではなく、脆弱性対応や EOL(End Of Life, 製品のライフサイクル終了)などによって開発後も継続的にバージョンアップや載せ替え

を検討し続けなければならない。このためにかかる多大な変更コストやテストコストは避けられないのか？当初の開発時にこのことを想定した。準備を品質保証として組み込む必要があるのではないか。



例えば、迅速性と品質の関係。リーンスタートアップにおける最小限の製品（MVP: Minimum Viable Product）を作る際に、従来のやり方で品質保証すべきだろうか。脆弱性対策や個人情報保護などの守るべき部分はしっかり守ったうえで、目的とするサービスをできるだけ早く市場にリリースし、反応を見てフィードバックをかけるというサイクルを、迅速性をもってまわすためにはどのようなやり方をすべきだろうか。

ニュースでも、ソフトウェアの問題が原因でサービスが停止したり、生産がとまったり、損害を生じさせたりする事例が報道されている。従来の品質保証プロセスが意図通りに遂行されなかったために生じた事故もあるだろうが、根本的に品質の考え方を見直す時期が来ているのではないだろうか。

金言 8. DX 時代の成功のメトリクス「真剣度+参画率」(金子 博)

DX を実現することは、未知のブルーオーシャンを探求することに等しいと言える。開発の目的は設定しやすい一方で、ゴールの具体性や効果などを正確に設定することは難しい課題である。なぜなら、開発プロジェクトのゴールはしばしば曖昧になりがちであり、真の共創開発が求められる一方で、その実現が難しいことが多いからだ。プロジェクトマネジメントの視点から見ると、プロジェクトの開始時にステークホルダー分析を行うことが求められている。この分析結果と対策が、プロジェクトの成功を大きく影響することは間違いない。

特に厄介なのは、ユーザ側の立場がベンダ側に比べて強い場合だ。過去の開発プロジェクトで築かれた関係性が、新たな時代の開発においては逆効果になることがある。関係性は、長い付き合いの中で組織の DNA として定着し、受け継がれていることが多い。

DX 時代における開発の目指すべき姿は、アジャイル開発の一環として捉えられる。アジャイル開発では、ユーザも開発の一員として参加し、ゴールの定義と評価に責任を持つ。顧客価値を最大化するためには、ユーザが開発にどの程度真剣に、そして具体的な参画をしたのかの割合が DX 時代の成功メトリクスとなるのではないかと。たとえば真剣度+参画率のようなメトリクスになるかもしれない。



金言 9. 目玉の数さえ十分あればどんなバグも深刻ではない ~ リーヌスの法則 (溝口 則行)

Linux 開発者である Linus Torvalds が言ったとされる法則がある。Linus の読み方が「リーヌス」か「リーナス」か「ライナス」かが決着したのかどうかは定かでは無いがここでは [「伽藍とバザール」の山形浩生さんの日本語訳](#) に合わせた。

「目玉の数さえ十分あれば、どんなバグも深刻ではない」

このくだけた表現は OSS 関係者の間では有名だが、別の言い方もある。

「ベータテスタと共同開発者の基盤さえ十分大きければ、ほとんどすべての問題はすぐに見つげだされて、その直し方もだれかにはすぐわかるはず」

しかし、後者の言い換え版の方がむしろマジ顔で攻撃されやすい様な気もするので、前者のカジュアル表現の方が無難なのかもしれない。さらに、別の表現もあって伝えたいことはほぼ同じ。

「はやめのリリース、しょっちゅうリリース」

これら法則が示すことは「品質保証部門を大増員しよう」ということではなくて、むしろその逆。「バグがあっても誰かがすぐに見つけてくれるし、その方が手っ取り早いから、とつとつリリースしちゃおうぜ」だ。(このあたりから筆者流の解釈が色濃く始めている)。そして、OSS 界隈ではこれがうまくいっている。…と、OSS 関係者は思っている。このやり方(バザール方式)は、OSS が脆弱な開発体制なのでやむを得ずやっている逃げの策というわけではない。この方が優れたやり方だと信じられている。なぜか？



本書で他のメンバが書いていたが「開発後も継続的にバージョンアップや載せ替えを検討し続けないといけない」のが普通で、「開発時に作成したものが半永久的に固定されるわけではない」。リーヌスの法則は、これに対応するベストプラクティスのひとつだと思う。

ソフトウェアがどんなに複雑で大規模なものだったとしても、結局は「何か」を実現するための手段であり部品であることに変わりはない。ならば、ソフトウェアの品質が「合格かどうか」は、使ってくれる人たちに一部委ねてもいいんじゃないのか。

けれど、みんな「客をデバッグに付き合わせる気か！」って怒鳴られた経験があるだろうから、「リリースしちゃおうぜ」って言い出すのはけっこう勇気がいることではある。

執筆 (順不同)

五味 弘	独立行政法人情報処理推進機構
齋藤 幸久	株式会社電算システム
河野 太基	富士通株式会社
溝口 則行	独立行政法人情報処理推進機構
山本 修一郎	名古屋国際工科専門職大学
三部 良太	株式会社日立製作所
金子 博	株式会社東芝

DXIT フォーラム DX時代の品質課題を考えるワーキンググループ (順不同)

主査	五味 弘	独立行政法人情報処理推進機構
	小西 広朗	株式会社電算システム
	齋藤 幸久	株式会社電算システム
	金子 博	株式会社東芝
	三部 良太	株式会社日立製作所
	野尻 周平	株式会社日立製作所
	片岡 祥啓	株式会社日立製作所
	岡田 侑里	株式会社日立製作所
	河野 太基	富士通株式会社
	佐々木 崇晃	みずほリサーチ&テクノロジーズ株式会社
	田中 秀人	独立行政法人情報処理推進機構
	溝口 則行	独立行政法人情報処理推進機構
	滝田 典子	独立行政法人情報処理推進機構
	山崎 昭司	独立行政法人情報処理推進機構
	西本 靖	独立行政法人情報処理推進機構
アドバイザー	山本 修一郎	名古屋国際工科専門職大学

DX時代の品質を考える金言コラム

2024年4月10日 発行

Copyright DXIT フォーラム 2024