

インプレスR&D [Next Publishing]



Cloud シリーズ

E-Book / Print Book

Next Publishing プレビュー



NextPublishing Sample

目次

はじめに	6
表記関係について	6
第1章 RPACommunity とは	7
第2章 RPAを入れたり自動化を検討する前に考えたいこと	9
2.1 はじめに	9
2.2 世の「情報システム」にも色々ある	9
2.3 SIerによるスクラッチ開発の自社基幹システム	10
2.4 パッケージソフトを使った自社基幹システム	12
2.5 自前システム(SQLServer+AccessVBAとかでしっかり目の構成)	13
2.6 自前システム(と呼べるかどうかわからないけどExcel+VBAぐらいのもの)や、システムっぽいものがないかも？	14
2.7 色々と書いてきましたが	15
第3章 RPAから始めるDX推進、成功の秘訣はこれだ！	16
3.1 はじめに	16
3.2 トヨタ生産方式とRPA	17
3.3 7つのムダとは	18
3.4 RPA推進のベストプラクティス	20
3.5 魂の入った業務選定	21
3.6 終わりに	23
第4章 業務断捨離のすすめ	24
4.1 今日から私は庶務担当	24
4.2 はじまり	24
4.3 何をする(した)か	24
4.4 なぜ庶務業務は属人化しやすいか	25
4.5 WHYから始めよ	29
4.6 断捨離(できたケース)	30
4.7 断捨離(できなかったケース)	35
4.8 悩み(悩んだこと)	36
4.9 気づき	36
4.10 さいごに	37

第5章 対象とする業務が決まつたら	38
5.1 はじめに	38
5.2 「仕様書」	38
5.3 業務情報	39
5.4 入力情報	40
5.5 出力情報	43
5.6 処理情報	45
5.7 補助情報	46
5.8 結び	47
第6章 スマートマットをハックしてさらに便利にする	49
6.1 スマートマットとは	49
6.2 スマートマットライト	51
6.3 スマートマットライト1と2の違い	55
6.4 スマートマットライトをハックする	59
6.5 パケットキャプチャで通信の内容を確認	59
6.6 ブラウザーの開発者ツールで通信の内容を確認	60
6.7 APIの仕様を推察	61
6.8 GASの実装	64
6.9 結果表示	72
6.10 さいごに	73
第7章 RPAによるリアルタイム処理	74
7.1 RPAのメリット	74
7.2 RPAのリアルタイム処理とは？	74
7.3 リアルタイム処理の実装	76
7.4 リアルタイム処理の適用事例	77
7.5 リアルタイム処理の課題と解決策	79
7.6 リアルタイム処理の今後の展望	81
7.7 最後に	81
第8章 Raspberry PiとGCPを使って、SMSのE2E監視を実装してみた！	83
8.1 はじめに	83
8.2 E2E監視の重要性と構築した経緯	83
8.3 監視の構成	84
8.4 SMS送受信シミュレータの実装	86
8.5 SMS送受信スクリプトの処理概要	86
8.6 インターネット接続処理の概要	87

8.7 ドングルに設定必要な AT コマンド	87
8.8 Web API の実装	87
8.9 自社内製の監視ダッシュボード側の実装	89
8.10 初期費用とランニング費用	89
8.11 初期費用について	90
8.12 ランニング費用について	90
8.13 運用について	91
8.14 まとめ	92
第9章 現場で使える Python 自動化入門	93
9.1 はじめに	93
9.2 頭をすっきりテスト駆動開発	93
9.3 みんなで読もう Sphinx	95
9.4 最後の救い log 取得	101
9.5 さいごに	103
第10章 GAS と AWS を使って Web 操作をハックしてみた!	104
10.1 はじめに	104
10.2 情報収集の自動化(GAS)	104
10.3 Web 操作の自動化(AWS)	106
10.4 GAS から Lambda の呼び出し(GAS)	109
10.5 さいごに	109
第11章 フローで思い通りの結果を得るために大切なこと	110
11.1 はじめに	110
11.2 テストは、2段階に分けられる	110
11.3 Case アップロードしたはずのファイルが開かない	111
11.4 バグのない正しいフローを作るために	140
11.5 終わりに	142
第12章 普通の OL なら業務に役立つロボットを必ず作れます。もちろんこれは体験談です。 143	143
12.1 自動化の手順も手腕も、三者三様にござりまする。	144
12.2 ロボット作成時の注意事項を申し上げますね。	148
12.3 突然のツール変更も視野に入れた準備が必要です。	148
12.4 “最初から 100% 自動化を目指さない” が成功の鍵！	150
12.5 安易に下請けを使うと、危険です。保守ができずに、すぐ錆びてあっけなく壊れます。 151	151
12.6 最初は、半分だけ手伝ってもらうつもりで進めましょう	155
12.7 現状の状態と、どの程度効率化できるかを可視化する	158

12.8	およそ大体のRPAアプリで準備されている部品の簡単なまとめ	162
12.9	黄金パターンをご紹介します (Excel間のデータやり取り、ログイン等に使用)	164
12.10	作業を見直します。	166
12.11	そのまま乗せ換えることが正解とは限りませんよ？	169
12.12	分解して再構成する。	169
12.13	分担を決める。	171
12.14	ツールを使い自動化/ロボット組み立てを行います。	172
12.15	普及型ローコードRPAツールに有効な共通の構成	173
12.16	Excelはお好きでしょう？ もう少し話しましょう。	175
12.17	終わりに これからはロボットと二人三脚で	178
12.18	次に御目文字できる日を楽しみに励みますわ	182

はじめに

お読みいただきありがとうございます。

某通信会社でインフラ設備の運用保守業務を担当し、日夜、現場に根ざした自動化・効率化に取り組み、日常の課題を解決しています。

SRE(Site Reliability Engineering)を目指して活動していますが、運用保守業務はいわゆる「コストセンター」と呼ばれ、サービスやシステムの信頼性を高める活動や付加価値を創造する活動にもあまりコストを掛けられません。

本書は、既刊「現場で使える!Google Apps Script レシピ集」、「図解と実践で現場で使える Grafana」の執筆メンバーに加えて、RPACommunity の有志が自動化のノウハウをまとめたものです。

普段、現場で自動化に取り組んでいる経験が誰かのお役に立てれば幸いです。

2023年 7月

北崎 恵凡

表記関係について

本書に記載されている会社名、製品名などは、一般に各社の登録商標または商標、商品名です。

会社名、製品名については、本文中では©、®、™マークなどは表示していません。

第1章 RPACommunityとは

RPACommunity 代表 Mitz (松岡 光隆)

ITを活用した業務の自動化やIT推進全般に興味を持つ全国の有志が集うコミュニティで、2018年3月に立ち上げました。

名称には「RPA」(ロボティック・プロセス・オートメーション)が付いていますが、RPA専用のツールやサービスだけに限らず、様々なツールやプログラムも含めIT技術全般を対象にした学びの場を提供しています。

発足から5年以上経ちメンバー数も増え続け8,000名を超えていながら、今現在でもコミュニティ内では様々な手法での業務自動化トークが飛び交っています。

そんなRPACommunity内の「自動化ネタ」を、まだRPACommunityを知らない方々にも知っていただきたいという想いを持つ有志が集って、今回の書籍作成に至りました。

RPACommunity¹では2023年5月時点で300回以上のイベントを実施しており、その資料はこちらに掲載されております。

図1.1: connpassのQRコード



また、2020年以降のイベントは全てYouTube²に公開しております。

1.<https://rpacomunity.connpass.com/presentation/>

2.<https://www.youtube.com/@RPACommunity>

図1.2: YouTube の QR コード



ご興味ある方はぜひご覧ください。

本書はこれらRPACommunityでの情報や登壇をベースに、RPACommunity有志メンバーが書き上げております。

第2章 RPAを入れたり自動化を検討する前に 考えたいこと

山田 雄一

2.1 はじめに

RPAを導入したり、自動化を検討したり、DXを始めたい、またそれ以前の話として現状の情報システムを改善したい……と思ったこと、ありませんか？また、自分は乗り気でなくとも、上から社命として言われてしまったり¹したこと、ありませんか？

筆者としては、ここ数年(2015年、16年以降)は、RPAやノーコード、ローコードツールが一般的になったことにより、単純なシステムリプレース以外に、取れる選択肢がずいぶんと増えたな……という印象があります。また既存システムにちょい足ししてみたり、既存システムのデータエントリー部分をノーコード、ローコードツールを使うようにすることで、こういった業務上の課題を解決できるかも……？という機運が高まってきており、実際に活用例も多くなってきてているように思います。

これらのRPA系のツールは、「RPA系のツールを使えば業務改善ができる」っぽく見えることが多いのですが、実際は打出の小槌のようにあらゆる業務改善や課題解決に効いてくれ、私達の業務を楽な方へ導いてくれるものなのでしょうか？

この章では、こういったときに何を判断基準にしたり、どういった懸念点があるのかをまとめてみました。RPA導入以前の検討段階として、「RPAを導入したり、自動化を検討する」までに必要な検討の助けになり、ある程度²の道筋を示せればいいなと思っています。

2.2 世の「情報システム」にも色々ある

世の中には色々な会社があり、色々な業務プロセスがあり、そして色々な情報システムがあります。情報システムについては色々なものがありますが、ターゲットを絞らないとこの手の話はついつい発散してしまうものなので、参考となりそうなケースを次のように分けてみましょう。

- ・スクラッチ開発の自社基幹システム
- ・パッケージソフトを使った自社基幹システム
- ・自前システム(SQLServer+AccessVBAとかでしっかり目の構成のもの)
- ・自前システム(と呼べるかどうかわからないけどExcel+VBAぐらいのもの)
- ・システムなし

1. こっちの方がもっと深刻である

2.もちろん筆者の個人的な考え方と独断と偏見によるものです

これらについて、それぞれ次のような対応策を今回は考えてみましょう。

- ・今のシステムを改修して隙間を埋める
- ・RPAで隙間を埋める
- ・RPAじゃないツールで隙間を埋める
- ・今のシステムをそのまま諦めて使う
- ・人を増やす

上で挙げたそれぞれのシステムx対応策について、それぞれの対応策の場合の検討ポイントをそれぞれ考えていきたいと思います。

2.3 SIerによるスクラッチ開発の自社基幹システム

SIerによる受諾開発(スクラッチ開発)で納品され、運用されている自社基幹システムの場合、受諾開発という契約の特性を考える必要があるケースがあります。

2.3.1 今のシステムを改修して隙間を埋める

この対応策は、開発してもらったSIerに再度追加改修をお願いすることになることが多いですが、改修時のコストがかさむこともあります。

また、SIerの開発人員も暇ではないので、継続的に追加改修をお願いしていない場合、改めて追加改修をお願いしたとして、実際に当時作っていた人が継続して対応してくれるか、品質が担保³されるか、みたいなところが悩ましくなることもあります。だいたい裏側はわかんないので……

また筆者の実体験として、継続的に改修を行っているシステムであれば、継続改修されているのでコストもそこまでかかるないこともあります。ですが、しばらく触っておらず、時間が経ってから改修を行うようなケースでは、過去のことを思い出す/既存システムの解析にもコストがかかるので、結構地味に大変だったりします(大変なときは総じて見積もりに跳ね返ります)。

2.3.2 RPAで隙間を埋める

現行システムの改修と比べ、RPAで隙間を埋めた時の最大の利点は、「システムの内部を修正していないのに、業務オペレーションは簡単にできる(かもしれない)」点です。

RPAはよくも悪くも「画面操作をするツール」で、システムの内部を修正しませんが、これはケースによっては大きなメリットになります。

○うれしいケース①

たとえば、社内でISO認証などの外部監査がある認証を取得しており、この監査情報の取得や集約がシステムに組み込まれている場合です。こういう場合に社内システムの改修をすると、改修箇所(時によっては監査箇所全般)の処理内容について、再度監査が必要になってしまうケースがあり

3. ここで言っている品質とは単に「バグがない」というだけではなくて、ソースコードの保守性であったり、応答速度であったり、データ量であったりといった非機能要件も含みます。

ます。

ところが、RPAでの業務改善を行うとあら不思議！社内システムから見ると、「画面操作をしているだけ」ですね！

これならどんな改善をしても安心！

○うれしいケース2

たとえば、最初のスクラッチ開発はSIerに頼んだが、その後の改修については他社に頼んだり、自分たちで行うケースもあるかもしれません(プロジェクト炎上したりするとありがち)。

よく受諾開発の契約書にはあるあるなのですが、システムを開発納入した会社以外が改修した場合は、納入会社はシステムの瑕疵担保責任をその後一切追わない、とする契約となっているケースがあつたりします(責任を負えなくなるのでそれはまあ当たり前といえば当たり前)。

こういった場合に、システムを直接改修するのではなく、RPAを使って改善を行った場合、やはり社内システムから見ると、「画面操作をしているだけ」になるのでシステム利用時の瑕疵担保責任が継続され、そういう点でも安心だつたりします。

2.3.3 RPAじゃないツールで隙間を埋める

RPAじゃないツールで隙間を埋める、というのもケースによっては非常に効きます。

基幹システムを作ったはいいが、そんなに今後長いことこれを使わないので、システム自体に改修は入れたくない、でも現状の業務は厳しいので改善をしたいとか、RPAではやりにくい大量データの捌きを行った上で、処理を実現する必要がある……といった場合に、たとえばETLツールがハマったりすることがあります。

ただ、RPAじゃないツールの場合において、データのI/OはAPI経由だったり、DBへの直接アクセスが手段になります。ですが、パフォーマンスが劣化したり、本来の基幹システムからのデータI/OではないI/O(特に書き込み処理)が発生するので、既存システムとのそういうの統制を取ったりするのが思った以上に大変です。当然ながら前述したような監査があるシステムの場合、悩ましいことになります。

2.3.4 今のシステムをそのまま諦めて使う

改修コストとそれにともなって得られるメリットが見合わない場合は、こうなりがちです。やむを得ないので……

2.3.5 人を増やしてオペレーションで解決

人を増やして対応するのはある意味最強の一手段です。ハネムーン係数なども改善するかもしれません。やつたね！業務が強くなるね！

※ただしボトルネックに人が増えない場合は、何も変わらず教育コストだけがかさむケースも……

2.4 パッケージソフトを使った自社基幹システム

パッケージソフトの場合には、パッケージソフトならではの検討ポイントがあるのでは？と筆者は考えています。

2.4.1 今のシステムを改修して隙間を埋める

そもそも、大規模な基幹パッケージ(たとえばSAPとか)を入れていて、既に運用が回っているフェーズならば、システムを改修したいと考える会社であれば、ほぼほぼアドオンによるカスタマイズが入った状態になっているはずです。その上で現在のシステムと業務にギャップが発生しているので、そこに更にアドオンによるカスタマイズを付け加える形(付け加えたいと考えている状態)になっているのでないかなと考えられます。

ただ、パッケージソフトのカスタマイズというのは、一般論でいうと2割の処理にカスタマイズを入れるのであればフルスクラッチで作るのとコストが変わらないと言われているぐらい、非常に開発コストがかかるものです。

また、一般的な大規模パッケージのベストプラクティスは「極力カスタマイズなしで使い、業務をパッケージに合わせる」となっていることが多く、そういった意味でも改修は悪手となりがちです。ですが、現場要望もあって、情シスとしても押し負けてしまうことが多く、なかなかそういった理想的な形にはならない実情がありますね……。

パッケージソフトにおいて、カスタマイズがないと何が嬉しいのか？

カスタマイズなしで使うと何が嬉しいかというと、パッケージのバージョンアップの際にカスタマイズはすべて見直しが要ることが多く、カスタマイズがなければパッケージのバージョンアップがサクサクで進むので、維持管理コストがずいぶん軽くなるのです……(カスタマイズがあればあるほど、再検証や再開発のリスクが上がる)。

なお、大規模パッケージを使うときにも、会計系の箇所については原則として一切カスタマイズを入れないケースが多く、これは前述の「監査がある場合」に近い話が待っていることが多いです。

具体的には、こういったパッケージを入れるときの大きなメリットのひとつとして、会計監査の省力化があげられます。

フルスクラッチで会計システムを構築した場合、大きな企業さん⁴では担当の会計事務所にシステムの監査をしてもらう必要があり、これが経理担当の悩みのタネだったりします。ですが、有名な会計パッケージであれば、それをそのまま素直に使ってさえいれば、ある程度の会計システムとしての正しさをパッケージが保証してくれます。そのため、会計的にチェックすべき点が相当シンプルになり、会計事務所が嬉しいので、経理担当も非常に嬉しいんだそうで……。

2.4.2 RPAで隙間を埋める

こういったケースでもやっぱりRPAはオイシイです。だって、画面しか触らないですからね。

4. 上場してたりだとか……

また、前述の「2割の処理にカスタマイズを入れるのであればフルスクラッチで作るのとコストが変わらない」みたいな話を、パッケージソフト側ではなく、その外で持てるようになるので、そういう意味でもいいです。

加えて、コスト面でも有利なことがあります。フルスクラッチ開発と比べても、大規模パッケージのカスタマイズができる人というのは、そのパッケージに精通したコンサルレベルのSEであることが多いです。そういう人にお願いするとどうしても単価そのものが高くて、結果これが開発コストとして高くなりがちです。RPAであればそういった高単価の人をアサインせずとも、業務改善が実現できるかもしれません。これは嬉しいポイントですね。

2.4.3 RPAじゃないツールで隙間を埋める

RPAじゃないツールも前項の「RPAで隙間を埋める」と同じ観点でオイシイです。ただ、内部データを直接APIで触ることになるので、投入したデータがパッケージ上でうまく処理されることを検証しておく必要があります(筆者は実際にそういうテストをやったこともあります……笑)。

2.4.4 今のシステムをそのまま諦めて使う

人生諦めも肝心です。投入コストより回収リターンのほうが多い場合は「何もしない」が最適解だったりしますよね……(特にパッケージは改修コストがかかりがちなので、ノーカスタマイズで使って人が合わせる方がよかったりすることもままあります)。

2.4.5 人を増やしてオペレーションで解決

人こそパワー！(※前の「スクラッチ」の話と全く論点は変わらないです)

2.5 自前システム(SQLServer+AccessVBAとかでしっかり目の構成)

さて、コストをがつりかけてSIerに基幹システムを作ってもらったりはしておらず、パッケージソフトを入れていない会社というのも意外に多いものです(私も前職は社内SEだったのですが、このパターンでした)。

2.5.1 今のシステムを改修して隙間を埋める

自前システムだと改修しやすい面と、改修が難しくなる面があります。

改修しやすいポイントは、自前で作っているのでコスト面や対応時間がコンパクトですみます。改修が難しいポイントとしては、そもそもそのために人数を多く割いていなかったりするので、マニピュレーションが難しいことがあります。また、少人数で開発をすることになるので、そんなチームにエキスパートがごろごろいるわけでもなく、結果として技術的負債が蓄積しやすい点でしょうか(そのため、改修コストがどんどん上がっていく傾向にあります)。

2.5.2 RPAで隙間を埋める

RPAについては、自前システムだと悪手になるケースが多いです(自分たちでシステムの表も裏も全部ハンドリングできているはずなので、RPAを使うより直接改修したほうが普通は望ましいはず)。さて、RPAを使ったほうがいいケースはあるのでしょうか?

たとえば、自前システムを構築している基盤にない機能をRPAで簡単に付け足せる場合などは、RPAを導入するメリットがありそうです(正直、前職ではこのシナリオで少し導入を検討したことあります。結局入れなかったのですが)。

2.5.3 RPAじゃないツールで隙間を埋める

外部データ連携などをする場合や、新しい機能を実現したい場合など(最近だとAIとか?)、RPAじゃない他のツールが刺さるケースもあるかもしれません。

2.5.4 今のシステムをそのまま諦めて使う

諦めも肝心です。でもどこかで重い腰をあげないといけないタイミングというのも……

2.5.5 人を増やしてオペレーションで解決

もうここまでくると、既存のシステムを守るという観点よりは、新しいシステムや業務プロセス改善を実現するために人を増やす必要があるかもしれません。

2.6 自前システム(と呼べるかどうかわからないけどExcel+VBAぐらいのもの)や、システムっぽいものがないかも?

小さい会社で簡単なシステムを作つて運用しているケースも多いでしょう。ExcelVBAなどがよく活躍するケースです。あとは、「情報システム」と言われて思い浮かぶものがないケースもあるでしょう。

でも、ITが基幹システムのようにしっかりと連動して動くようになっていなくても、ITを使った業務プロセスが回っているケースなども、広義のシステムと言えるかもしれません。

2.6.1 今のシステムを改修して隙間を埋める

簡単なシステムや小さなマクロに分割されているほど、改修というよりはひとつのシステムにまとめていくほうがいいかもしれません。

2.6.2 RPAで隙間を埋める、RPAじゃないツールで隙間を埋める

ここはもうツール導入の可否みたいな話で、論点がまとまります。

そもそもその話として、RPAやその他のツールで隙間を埋めるにせよ、ある程度の粒度のシステムになってからな気がします。その際に、ハンドオペレーションが入っている箇所をシステム化し、一気に自動化できるかもしれません。

なお、それは一般的には「DX」と言います。

改善の伸びしろがいっぱいある状態は、ある意味ではいいことなのかもしれません。

2.6.3 今のシステムをそのまま諦めて使う

まあそんな判断になることもあるかもしれません。投入できるコストは有限ですし。

2.6.4 人を増やしてオペレーションで解決

これぐらいの規模感だと、「増やせるものなら増やしたい」というのが本音になりそうな。

2.7 色々と書いてきましたが

色々なケースを挙げ、RPAが効きやすそうなケース、効きにくそうなケースを様々に検討してみましたが、いかがだったでしょうか？

ちなみに情報システムの再構築における様々な論点については、「SEC BOOKS：システム再構築を成功に導くユーザガイド 第2版～ユーザとベンダで共有する再構築のリスクと対策～」という書籍があったのですが、絶版になってしましました。ただ、PDFは無料公開されていて、誰でも読めるようになっており、ここに様々な参考となる情報が掲載されています。

<https://www.ipa.go.jp/archive/publish/secbooks20180223.html>

RPAでの改善を検討する際にも、本来的なアプローチとしては

- ・システムリプレースや改修するほどではないね
- ・RPAで改善をしようね

という検討結果を経て、その後からRPAでの改善が始まるのでは？と思いますので、こういったソースにも当たってみて、一読した上で論点整理をしておくと、少し導入がスムーズになるかもしれません。⁵

この記事が最適なITシステム構築、ツール選定戦略の一助となれば幸いです。

5. 上長に説明するときとか、資料があると説明力が段違いだったりします

第3章 RPAから始めるDX推進、成功の秘訣はこれだ！

瀧谷 匠

3.1 はじめに

皆さんこんにちは。株式会社ASAHI Accounting Robot研究所の瀧谷です。ここからは私が前職の製造業で16年間研鑽し、現在も日々勉強をしているトヨタ生産方式とRPAのお話から始まります。そして、RPAの導入が決まった後の社内推進体制をどうするか、RPA推進で気を付けるべきポイント、自働化したい業務をどのように優先順位をつけていくか、RPA推進がどのようにDXにつながっていくかをお話ししていきます。

図3.1: 自己紹介

ASAHI Accounting Robot Research Institute

自己紹介

詳細な自己紹介

QRコード

LinkedIn

しぶや たくみ
瀧谷 匠 / μ

株式会社 ASAHI Accounting Robot 研究所
テクニカルエバンジェリスト / DXアドバイザー

全ての人の仕事を楽にするをモットー
製造業16年、うち4年間DX・RPA推進を経験

認定DXアドバイザースペシャリスト
デジタル庁デジタル推進委員

DXアドバイザー
デジタル推進委員

会社紹介

あさひ会計グループ内の業務効率化を目的に結成された我々株式会社ASAHI Accounting Robot研究所（通称：ロボ研）は、現在、全国、世界の企業に向けて”ヒトとロボット協働時代を推進する”というミッションを掲げ、遊びごころを持って全力で取り組んでおります。RPAやAIを中小企業が身近に活用する時代になることを目指しています。

私のパートでは、通常とは異なる漢字を意図的に使用しています。言葉に意味を持たせることで、皆さまによりわかりやすくイメージをもっていただきたいからです。以下にその解説をいたします。

- ・自働化 →RPAによる業務改善は自動（自ら動く）ではなく自働（自ら働く）という意味。RPAがただ動くのではなく人に代わって働くということを表している
- ・観る →目で見るだけではなく、しっかりと観察して分析するという意味

3.2 トヨタ生産方式とRPA

ここからは、トヨタ生産方式とRPAというテーマでお話をします。

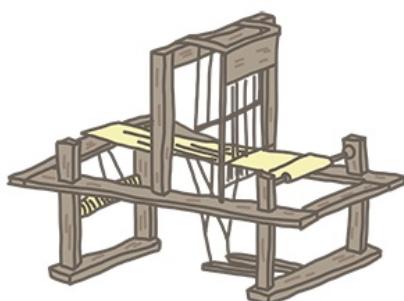
このテーマを聞いて、きっと皆さんこう思ったんじゃないでしょうか。

- ・生産方式って製造業の話で自分たちには関係ないよね
- ・生産の仕方の話とRPAって何の関係があるの？

トヨタ生産方式は、決して製造業のみに関係する話ではありません。また、RPA推進に大いに役立つヒントがあることを私は身をもって実体感、実体験しております。

まずはトヨタ生産方式について、トヨタ生産方式の原点は自動織機（自動的に糸を織物にする機械のこと）の開発で有名な豊田佐吉氏が研究してきた織機に“縦糸が切れると自動的に機械が止まる”といった、異常が起きると機械が自動的に判断して止まる機構を機械に組み込んだことに原点があります。

図3.2: 機織り機イメージ(イラストは人力のもの)



その後、不良品を造らない、人を機械の番人にしないといった「ニンベンのついた自働化」の考え方と、佐吉氏の息子でトヨタ自動車の創業者である豊田喜一郎氏が1937年ごろに提唱した必要なときには必要なだけつくるといった「ジャストインタイム」の考え方方が「トヨタ生産方式の二本柱」と言われています。これを戦後、具現化したのは、トヨタ生産方式の生みの親である大野耐一氏です。大野耐一氏は資金や資材の乏しい中、日本と比べて非常に高い生産性の欧米の企業に打ち勝つには「ニンベンのついた自働化」と「ジャストインタイム」の実現しかないと考え、現場にてその具現化にむけた研究を重ねました。そして、自働化によるムダ徹底的排除とアメリカのスーパーマーケットにヒントを得た「後工程引取り」の実施により、ジャストインタイムを実現しました。

異常（エラー）があると自動的に止まる、不良品（不良書類・不良データ）を造らない、人を機

械（パソコン）の番人にしない、必要なときに必要なものを必要なだけ（ロボットを）つくる、これらキーワードを聞いて私は真っ先に「これこそRPAだ！」と思いました。

- ・1. 異常があると自動的に止まる

—エラーが出たら例外処理でエラー内容をメールで報告したり、ウインドウを閉じて終了する

- ・2. 不良品を造らない

—不良書類・データを造らないよう、事前検証や照合をしっかりする。ミスも起こらない

- ・3. 人を機械の番人にしない

—ロボットのスケジュール実行で無人化

- ・4. 必要な時に必要なものを必要なだけ

—深夜や休日であっても、24時間365日ロボット実行できる

どうでしょう。これらのトヨタ生産方式の重要なポイントは、RPAの機能や全社展開に相通じるものがあるという気がしませんか。言い方を変えると、RPAを導入し業務自働化を進めていく際、このトヨタ生産方式の概念が必ず“推進の羅針盤”になると考えます。

事実、トヨタ生産方式は製造業のみではなく業界問わず、たとえば医療関係や物流業、小売業、はたまた片づけや貯金などにも応用されています。

皆さんは毎日の仕事の中で「付加価値を生む仕事」ができていますでしょうか。俯瞰して見てみると、意外とできていないものです。私自身、普段の仕事の中にムダはまだまだ潜んでいると感じます。

ムダ・ムリ・ムラ（3Mと言います）の改善手段はいろいろありますが、その中のひとつとして付加価値のない業務はRPAという名のデジタルレイバーに任せて、人は人にしかできない創造性のある仕事に注力できるようにしていきましょう。

3.3 7つのムダとは

さて、前章で3M（ムダ・ムリ・ムラ）の話が出たところで、ここからはもう少し掘り下げて、7つのムダというテーマでお話をします。

トヨタ生産方式には、7つのムダという考え方があります。これらのムダが自分のまわりにないか常に目を光らせて、見つけたら徹底的に排除しなければなりません。

トヨタ生産方式「7つのムダ」

- ・【か】加工のムダ

- ・【ざ】在庫のムダ

- ・【つ】造りすぎのムダ

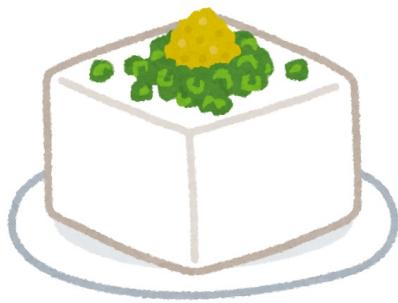
- ・【て】手待ちのムダ

- ・【と】動作のムダ

- ・【う】運搬のムダ

- ・【ふ】不良・手直しのムダ

図3.3: 飾って豆腐(かざってとうふ)



それぞれの頭文字を取って「飾って豆腐(かざってとうふ)」と覚えます。一見するとどれも製造業の現場ならではのムダのように觀えますが、事務・バックオフィス業務に置き換えると、このように考えることができます。

- ・加工のムダ →ファイル、資料作成に凝りすぎる（レイアウトやアニメーション）
- ・在庫のムダ →不要な書類・データを大量に保管している
- ・造りすぎのムダ →必要以上に資料を作っている（似たような資料を複数の担当者で作っている）
- ・手待ちのムダ →他部署や他の担当者からの情報を待っている
- ・動作のムダ →ファイルや情報を頻繁に探している
- ・運搬のムダ →仕事の導線が悪い
- ・不良・手直しのムダ →資料の作り直しが発生している（指示が悪い）

会社勤めの方であれば、誰もが思い当たるところがあると思います。このような付加価値を生まない作業を徹底的に排除する、そのためにはムダを見つける感性と、見つけたムダを即座にカイゼンする実行力、さらに諦めない気持ちが求められます。また、これらはRPAを社内で推進していく上でも気を付けなければならないポイントでもあります。

今やっている業務をそのままRPAで自動化するのではなく、初めに業務の棚卸をしてその業務が本当に必要なのか、業務手順にムダがないか、複数の担当者で同じことをやっていないか、やり方を変えると一本化・標準化できないかという目線で見て、どうしてもやらなければならない業務のみを自動化する（私が講演でいつもお話をされる KAIZEN の三原則、ヤメル・ヘラス・カエルのことです）。このように7つのムダの観点で業務を見直すことで、その後の自動化を最短で安定したものにすることができますし、その活動自体が部署や担当者のレベルアップにつながり考え方が変わってくると実感しています。

カイゼンにも、RPAによる自動化推進にも終わりはありません。トヨタ生産方式には「カイゼンした今が一番悪いと思え」という言葉があります。RPAの社内展開がうまくいってよかったですと安心するのではなく、今が一番悪い・もっと上を目指すんだという意気込みで今日も仕事に取り組んでいきましょう。

業務効率化、それはまさに“終わりなきカイゼン”です。

3.4 RPA推進のベストプラクティス

さて、ここからは「RPAの進め方、現場主導か推進部門主導か」というテーマでお話をします。なぜ今さらこんな話をするのかと申しますと、私はエバンジェリストとして、また北海道のRPA推進者として、各種様々なイベントや勉強会等で登壇する機会が多いのですが、その際毎回といつていい程議題にのぼる内容があります。

それが

「RPAを推進するベストプラクティス（最善の方法、最良の事例）は何か」
ということです。

結論から申しますと、どのような進め方がベストかは各会社の風土やITリテラシー、推進者の立場やスキルなどなど、パターンによって変わります。その会社や企業ごとに進め方は違ってよいですし、マッチする進め方が必ずあります。

企業内でのRPA推進には、大きく分けて「現場主導」と「推進部門主導」のふたつがあると言われています。私が考えるそれぞれのメリット・デメリットは以下です。

3.4.1 現場主導

よい点

- ・RPAを適用する業務フローを一番わかっている人がロボットを作成できる
- ・自分（所属担当）の業務改革に直結できる

難しい点

- ・専任者を置きにくい（兼任にするとロボット作成時間の確保が難しくなりがち）
- ・ロボット作成者が多く必要なので、育成に時間及び費用がかかる
- ・ロボットの管理が大変（野良ロボットができやすい）

3.4.2 推進部門主導

よい点

- ・専任者が集中してロボット作成できる
- ・ロボット作成者の育成は必要最低限の人数で済む
- ・ロボットの集中管理が可能

難しい点

- ・業務の詳細を把握してロボット作成しなければならないため、現場の協力が不可欠となる（業務フローやマニュアルの作成が必須で、RPA導入目線での業務変更や、詳細なヒアリングが必要）
- ・現場の担当者はRPAを「自分ごと」としてとらえにくく

いかがでしょうか。どちらの進め方も一長一短があります。それぞれのよい点・難しい点を自社に置き換えたときにどうかしっかりと吟味して、どちらの方法でRPAを進めていくことが自社のベストプラクティスであるかをジャッジすることが肝要です。

私は、前職の製造業でRPA推進をしていたときは「推進部門主導」を選択して進めました。現場

主導を選択しなかったのは、大きく以下3つの理由があります。

- ・理由①
 - 各現場がとにかく忙しく、現場主導で進めた場合RPA専任者を置くことはできず兼任となり、そうなった場合ロボット作成時間の確保が厳しいのは明らか。
 - ・理由②
 - 過去に別のITツール展開を行った際に現場主導を選択し推進を試みたが、遅々として進まずうまくいかなかった事例があったため。
 - ・理由③
 - 業務効率化の目標を立て、達成するための実績管理がしやすい。また、ロボット作成工数のタスクマネージメントが容易にできる。
もしこれらに心当たりがあれば、推進部門主導で進めるのも一考です。
- RPA推進で悩みはつきものです。トヨタ生産方式では「困らんやつほど、困ったやつはおらん」と言います。悩むということは、問題を見つけているということ。問題を見つけなければ解決策を考えることはできません。一人で抱え込みず、悩みは皆で共有し皆で解決していきましょう。

3.5 魂の入った業務選定

さて、ここからは「導入推進の切り札！らむだ式！RPA適用業務選定のすべて」というテーマでお話をします。

イベント・セミナー等に参加して、皆さんと情報交換・交流していると、このような悩みをとても多く聞きます。「RPAは費用対効果が出せない（出すのが難しい）」「自働化する業務がなかなか見つからない」

私は今まで、イベント登壇など機会があるごとに幾度となくお話ししてきましたが、あらためて私が前職でRPA推進時に実際に使用していたRPA適用業務選定基準について詳しくご紹介します。

RPAをスタートする際に「どの担当のどの業務からどういう順番でロボットを作成していくか」、これはRPA推進者であれば誰もが必ず最初に悩むところです。たとえば、ロボットの作成は簡単だが効果（費用対効果）が少ない業務といった場合は、ロボット作成工数（時間）のほうが目立ってしまい、RPA推進プロジェクト自体が効果がないものと見られてしまうことがあります。逆に、費用対効果はとても大きいかもしれないがロボット作成に〇か月要します、というような業務だと、実際に費用対効果が出るまでの間、活動していないように見えてしまい、RPA推進プロジェクト自体の存続に関わってしまうことにもつながりかねません。つまり、①どの業務をRPAで自働化するか、②その業務をどの順番で進めてゆくか（自働化する優先順位）、この2点はRPA推進の際、非常に重要なポイントとなります。と、わかってはいるものの、研修・セミナーを聞いたり、関連書籍を読んでみたり、そういう中に記載のあるような開発標準例ではなかなか自社に合った方法が見当たらないということも多いのではないでしょうか。

そんなときは、これからお話しする「らむだ式！RPA適用業務選定」を使っていただくことを強くおスメいたします。なぜこんなにも自信をもっておスメできるかというと、机上で考えたものではなく、「RPA推進者とRPAを適用する現場が一緒に悩み、考え、何度も改良を繰り返して作

り上げた適用業務選定」だからです。まさしく「魂の入った業務選定」です。ポイントは以下です。

1. RPAで自動化したいという業務を7つの項目で点数付けし業務を数値化
2. 点数は1点～5点の5段階評価（最高得点は7項目×5点＝35点）
3. 点数が偏る場合は採点条件を常に見直す（現在7代目）

図3.4: らむだ式RPA適用業務選定

らむだ式RPA適用業務選定

	項目	定義	1点	2点	3点	4点	5点
1)	開発工期	新規開発or既存ロボット改造度合を考慮した開発工期(トライ＆エラー期間含む)	3か月以上	1か月以上～3か月未満	2週間以上～1か月未満	1週間以上～2週間未満	1週間未満
2)	作業頻度	<月当たり> 作業回数×作業人数	1以下 <small>例)月1回×1人</small>	2～3 <small>例)都度(月2～3回×1人)</small>	4 <small>例)週1回×1人</small>	5～20 <small>例)毎日×1人</small>	21～ <small>例)毎日×2人以上</small>
3)	業務工数	<月当たり> 2)×1回あたり作業時間(分)	～200 <small>例)年間10h</small>	201～420 <small>例)年間50h</small>	421～2,400 <small>例)年間100h</small>	2,401～4,800 <small>例)年間500h</small>	4,801～ <small>例)年間1,000h</small>
4)	業務内容	全業務フローに対する人の判断割合 (100%は開発不可)	80%～100%	60%～79%	40%～59%	20%～39%	0%～19%
5)	業務手順書	作成依頼してから業務手順書を提出できるまでの期間(※RPA開発部分の業務手順書)	3か月以上	1か月以上～3か月未満	2週間以上～1か月未満	1週間以上～2週間未満	すでにある or 1週間未満
6)	システム横断	業務手順がExcelのみで完結しない	Excelのみで完結 (マクロで対応可)				Excelのみで完結しない
7)	標準化	業務手順パターン数、分歧条件数をもとに仕様確定までに要する準備期間 (例：作業者別に手順やフォームが異なる。 設備別に手順が異なる) ※業務ヒアリングによって頭在化する標準化されない業務手順や複雑な分歧条件(潜在的リスク=推進部門の過去の導入実績/経験をもとに算定)	3か月以上	1か月以上～3か月未満	2週間以上～1か月未満	1週間以上～2週間未満	1週間未満

具体的な内容を観ていきましょう。

- ・1. 開発工期 予想されるロボット開発にかかる日数
 - 既存ロボットを改造することで対応できると工期が短くなり高得点となる
- ・2. 作業頻度 月当たりの作業回数×作業人数
 - これは定義そのままですね
- ・3. 業務工数 月当たりの作業時間
 - 2に1回あたりの作業時間をかけます
- ・4. 業務内容 業務全体に対する人の判断割合
 - 人の判断に入る業務でも、その前後を自動化するという方法があります
- ・5. 業務手順書 業務手順書があるか、なければ提出までどれくらいかかるか
 - 業務が担当者の頭の中だけにあるようでは困るので業務手順書は必須です
- ・6. システム横断 業務が複数システムを横断しているか
 - Excelだけの業務だとRPA以外でも効率化できます
- ・7. 標準化 業務を標準化するまでにかかる日数
 - 業務パターンが多岐に渡るとロボットが長大になるので事前に業務標準化します
 - 非常にシンプルかつ明瞭でわかりやすい、7つの項目のみにあえて絞ってあります。

この業務選定を使用した場合、大きな効果がふたつあります。ひとつはその業務がどういう基準

で選定されたかが数値という形で一目でわかりますので、点数が低くてロボット作成対象から外れてしまった場合でも、なぜそのようなジャッジになったのか納得のいく回答・説明ができるということ。もうひとつはRPA推進の費用対効果などを上長や役員へ報告する際に明確な根拠を提示できるということです。事前に基準に合わせて選定をしておけば、効果の算出は非常に容易です。

トヨタ生産方式では「百聞は一見にしかず、百見は一行（行動）にしかず」と言います。100回聞くよりも1回見る方がよくわかる、というのは皆さまご存じの通りですが、さらに一歩踏み込んで100のことを見たり、100回見るよりも、自分で1回行動してみることが大事という意味です。

今回私が紹介した「らむだ式！RPA適用業務選定」は私自身の経験からまとめたものであり、あくまで一例です。もし冒頭で記載したように適用業務がなかなか見つからずに悩まれているようであれば、まずはこちらをたたき台にして自社のオリジナル選定基準を作成することで、RPAの社内展開をもう一步前に進める機会になると思います。

3.6 終わりに

ここまでトヨタ生産方式とRPAについて、社内のRPA推進体制を決めて、集まった自働化対象業務を適用業務選定を使って明確な根拠を持って自働化していくことの重要性についてお話しきました。

私の4年に及ぶRPA/AI-OCRの全社展開成功経験、またたくさんのRPA推進者の方との情報交換でわかったこと、それは企業のDX推進には現地現物で「デジタル化について広く正しく周知する」とことと「デジタル化のハードルを下げる」ことが欠かせない、ということです。つまり、RPAを“きっかけ”として、DXの一歩を踏み出すことにより、RPAがDX推進の基礎土壌となってデジタル化のハードルが下がり、休むことなく次々にデジタル化に取り組むことで従業員の意識が変わります。この過程を進めていくことで、デジタルを使用した企業変革（DX）は必ず実現できます。まさしくこれは「人に優しいデジタル化」と言えます。RPAもDXも進めるのは「人」です。人を中心に、人を大切に進めていくことがとても重要です。

私のモットー、活動の原点は「全ての人の仕事を楽にする」です。この気持ちはRPAと初めて出会った2018年から一貫しています。初めてロボットを作ってそれが完成して動いたとき、私は「これまで周りのみんなの仕事を楽にできる！みんなを助けることができる！」と感動したことを今でも鮮明に覚えています。全社員が自分だけではなく、隣の・同じチームの・同じ部のメンバーの仕事を楽にしていく、これこそがRPA推進の原動力であり、そのように進める中で社員全員の気持ちが前向きになって、もっと仕事を楽にしよう、みんなで一緒に頑張ろうとなり、そして会社の雰囲気がガラッと変わって、ふと気づいたらいつのまにかDXが進んでいた、そのようになるのが理想ではないでしょうか。

ぜひ我々と一緒に笑顔で楽しくRPA・DXを進めていきましょう。

第4章 業務断捨離のすすめ

北崎 恵凡

4.1 今日から私は庶務担当

これから話す内容は架空の会社の技術部門の一組織で、本業であるエンジニアリングの傍ら、庶務担当を命ぜられ、業務断捨離を進めてきた体験談(フィクション)です。どのように考えて行動し、苦楽を経験し、業務改善してきたかの一節を共有できれば幸いです。

4.2 はじまり

少子高齢化、働き手世代の人数減少と団塊世代の引退(2025年の崖問題¹)。じわじわと差し迫ってくる不安がある日、現実のものとなった。担当者が3人→1人に減り(2人は円満引退・シニア再雇用で別の道へ)、技術系管理職の宿命とはいえ、すべての庶務業務が自分に回ってきました。組織変更や人事異動も重なり、業務整理をせざるを得ない状況に陥りました。何から始める(た)か。最初に頭の中に浮かんだのは、業務のデューデリジェンス(本来は投資を行うにあたって、投資対象となる企業や投資先の価値やリスクなどを調査すること)でした。表現として「業務整理」「業務棚卸し」でもよかったです、判断軸を設定し、価値がないものに工数(コスト)をかけてもムダだと思ったからです。

営利企業の目的はふたつしかない。

- ・売上(収益)増加(P)
- ・コスト削減(L)

本業はインフラ設備の運用保守業務を担当し、SRE(Site Reliability Engineering)²を目指して活動していますが、運用保守業務はいわゆる「コストセンター」と呼ばれ、サービスやシステムの信頼性を高める活動や付加価値を創造する活動にもあまりコストを掛けられず、日々昼夜、自動化・効率化に勤しんでいます。庶務業務もエンジニアリングの一種と捉えて、同じように取り組めないか、と考えはじめました。

4.3 何をする(した)か

- ・庶務業務一覧の作成

1. 経済産業省が「DX レポート」にて提示した日本の近い将来に対する警鐘で、日本企業が DX の取り組みを十分に行わなかった場合、2025 年以降に年間で最大 12 兆円の経済損失が発生し、国際競争力を失うという課題。合わせて老朽化した IT 設備が残ると、技術的負債によりサービス競争力も低下することが懸念される。

2. Google 社が提唱したサービス・システムの信頼性、安定性、拡張性、効率性を高めるためのエンジニアリングの一種で、システムのエラーを防止するために、自動化、モニタリング、テスト、リカバリなどのプロセスを導入することにより、人為的なミスを減らし、高いシステム信頼性を実現することを目指した方法論。システムの性能改善を目的としたメトリクスやデータ分析に基づく意思決定も重要な要素となっている

- ・当番表の作成 + ロギング・数値化
- ・報告フォーマット(1枚)の作成
- ・Before / After シートの作成
- ・セルフサービス化
- ・コミュニケーションコストを下げる(情報の公開、チャットボットの活用)
- ・機械化・自動化
- ・事務局の立ち上げ

4.4 なぜ庶務業務は属人化しやすいか

本業でないことは誰も興味がないし、担当者に丸投げ・任せきりになり、属人化しやすいです。担当者が突然いなくなると困る潜在的な問題を抱えています。業務知識を公開(属人化→形式化)し、業務理解に努める仕組みが重要です。

4.4.1 庶務業務一覧の作成

庶務業務一覧を作成しました。

図 4.1: 庶務業務一覧

No.	業務名称	業務内容	責任者	担当者	業務確認書	更新日	工数(h/月)
1	業務A	- 項目A	責任者A	担当者A 担当者a	資料A	2023年4月30日	12
2	業務B	- 項目B - bはbbをXXまでに行う	責任者B	担当者b	-	2022年1月15日	2
3	業務C						
4							
5							
6							

・採番

一業務に詳しい担当者間であれば「XXの件」で通じるかもしれません、業務内容を知らなければ説明や指示にコミュニケーションコストが発生するので、庶務業務の「No.3」の件、というようにミニマムコストで認識の齟齬が発生しないコミュニケーションが大事です。

・業務名称

一誰にでも(業務を知らない人でも)内容をある程度把握できるわかりやすい名称にします。

・業務内容

一軽微な内容であればここに記載します。量が多く内容が煩雑な場合は業務確認書を作成します。

・責任者

一責任者不在は担当者が困ります。相談相手も報告先もなく、業務が属人化・形骸化しやすい原因を作ります。

・担当者

・業務確認書

- ・工数(h/月)

一定量的に数値で判断できるのは重要です。工数＝コストです。業務内容に対して適切なコストかどうか(サンクコストも含めて)判断します。

カテゴリー分け、優先順位づけ(重要度と緊急性の2軸マトリクスで行うことが多いですが、困り度も考慮に含めて)行います。

4.4.2 当番表

リマインダー、チェックリスト、ロギングの機能を持ち、自動的に数値化まで行います。SREではトイル(Toil)³と呼ばれる種類の業務で、1人あたりのトイルに割かれる時間を50%未満にすることが推奨されています。

図4.2: 当番表

2023年5月8日	当番	担当A		
作業種別	業務	頻度	内容	実施記録
アテンド				未
確認・収集				済

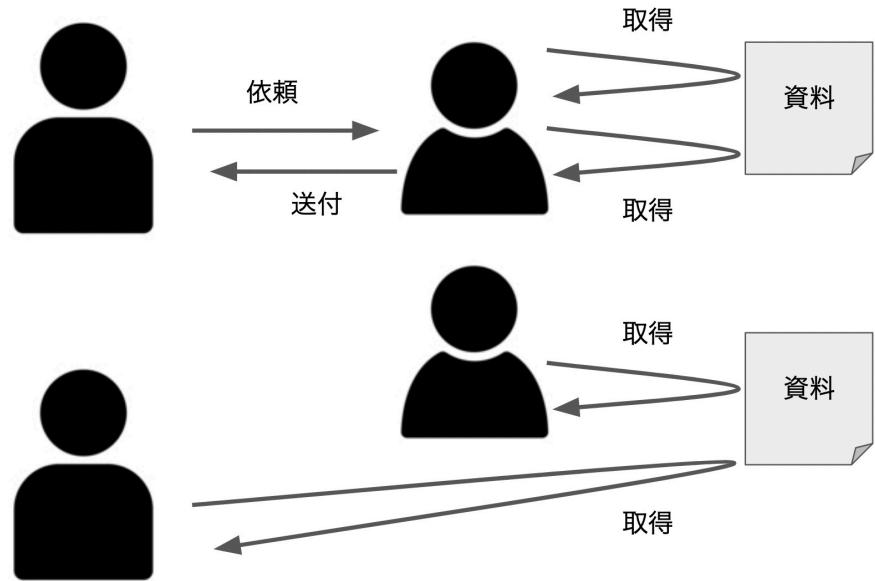
4.4.3 情報の公開

密室の議論は内容が不透明で属人化を生みやすいので、基本的に情報は公開するようにします(ただし、セキュリティ上の情報機密区分に応じて内容と公開範囲は適切に設定します)。情報が公開されれば、情報の伝達がPUSH型からPULL型になることでレイテンシー(待ち時間)が短くなり、コミュニケーションコストが下がります。

3. 時間(Time)、オペレーション(Operation)、インシデント(Incident)、負荷(Load)の頭字語。同じことを繰り返し手動で実行する、スケールしない作業を指します。

図 4.3: 情報の方向

情報の伝達(PUSH型からPULL型へ)



4.4.4 業務確認書

異なる業務を同じフォーマットで見られるようにして、情報を横断的にアクセスできるようにしておくことが重要です。

図 4.4: 業務確認書

情報のフォーマット化(形式化・体系化)



4.4.5 報告フォーマット

庶務業務は煩雑で細々とした内容が多く、情報量が多くなりがちです。整理前と整理後を1枚で見えるようにすることが重要です。

図4.5: 報告フォーマット

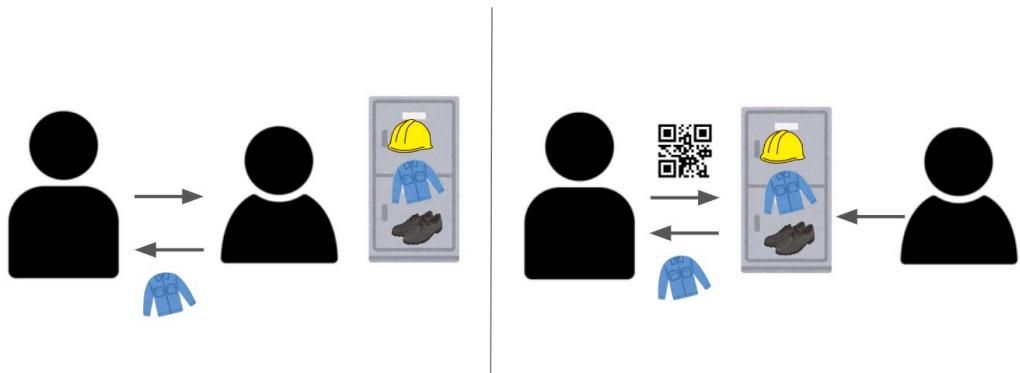
2023年4月30日時点	整理後	整理前
変更あり (対応必要)	チームA No.AA, BB, CC	No.MM, NN, OO, PP
変更なし (対応不要)	チームB No.DD, EE, FF	
変更なし	チームC No.XX, YY, ZZ	

4.4.6 Before/Afterシート

これまでの習慣や慣れた行動を変える、変えさせるのは大変です。なかなか理解は得られなくても、シンクコストを下げるために、わかりやすく伝える工夫をします。基本は「人を仲介・介在させない=セルフサービス化」、「コミュニケーションコストは最小限に」です。

図 4.6: Before / After

Before / After



4.4.7 事務局の立ち上げ

業務を前へ進めるためには、利害関係が対立した場合に備えて、とりあえず、落とし所、最後の砦、困ったときの駆け込み寺を用意することが重要です。

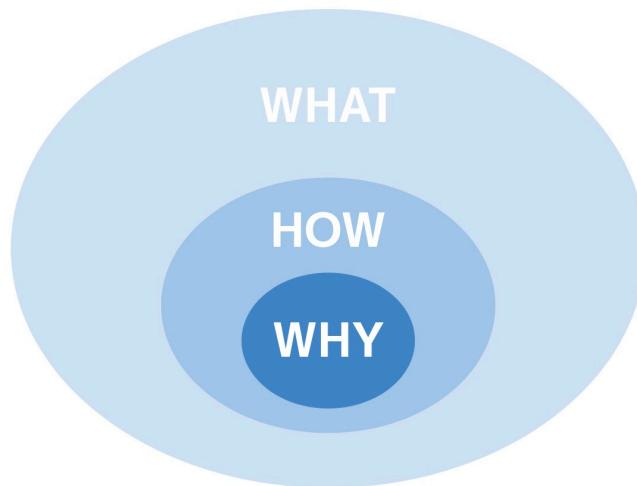
4.5 WHYから始めよ

ゴールデンサークル理論⁴を利用して業務を査定していきました。「何の業務をやっているのか」「なぜ、その業務をやっているのか」「その業務のやり方はベストなのか」業務を担当している方への敬意と感謝を忘れずに、問い合わせ続けることが重要です。

4. 経営者やマーケターによって使用される、より深い目的や価値観を明確にするためのフレームワーク。

図4.7: ゴールデンサークル

なぜ、その業務が必要なのか



4.6 断捨離(できたケース)

4.6.1 根本原因が既に解決していた

時間経過とともにルーティン化・マンネリ化・形骸化していた。

図4.8: はじまり

横展開・横串(はじまり)

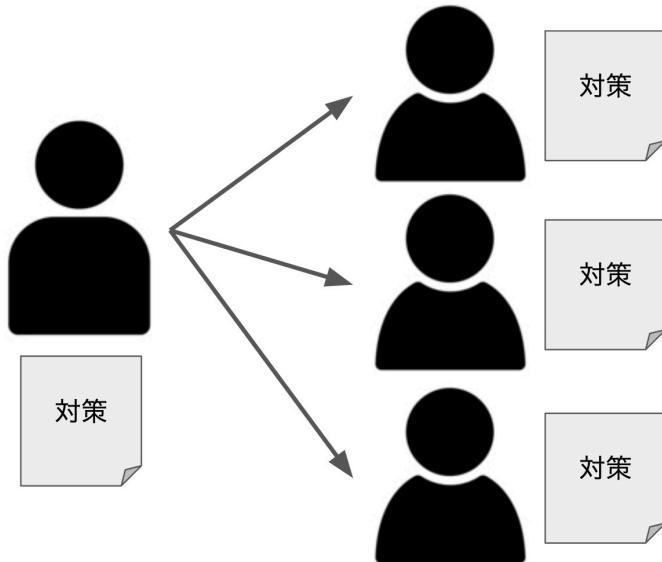
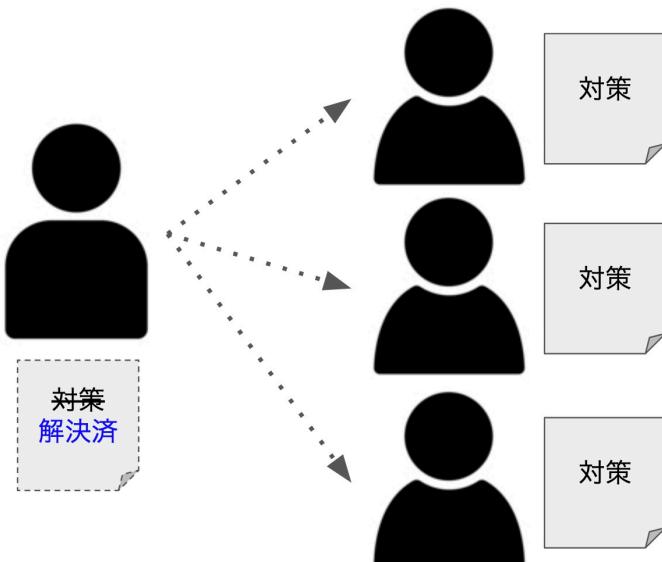


図4.9: その後

横展開・横串(その後)



4.6.2 他では簡易に実施、または、実施していなかった

図4.10: 井の中の蛙



4.6.3 機械化・自動化

巡回業務をなくすことができました。

- ・在庫確認(スマートマットの活用)

図4.11: スマートマットライト1

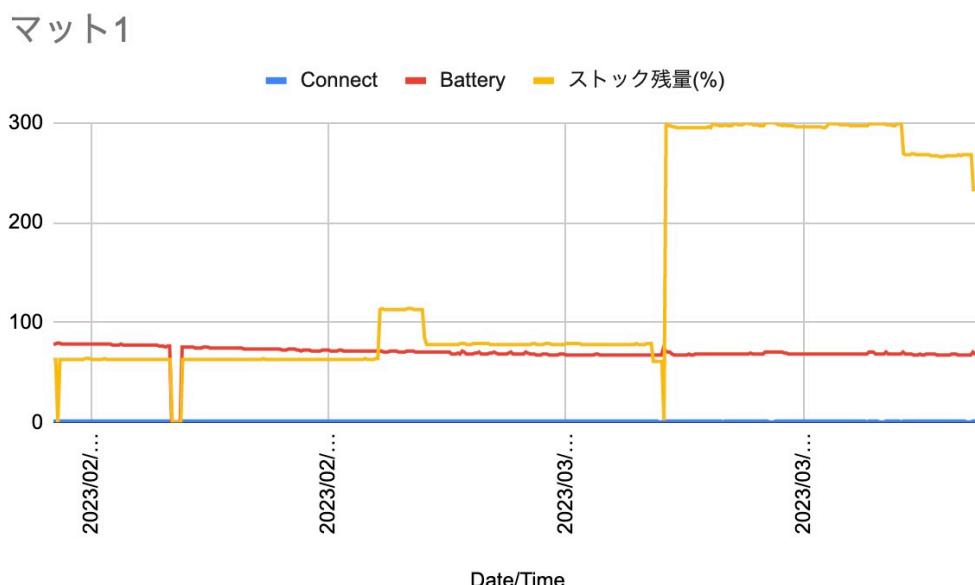
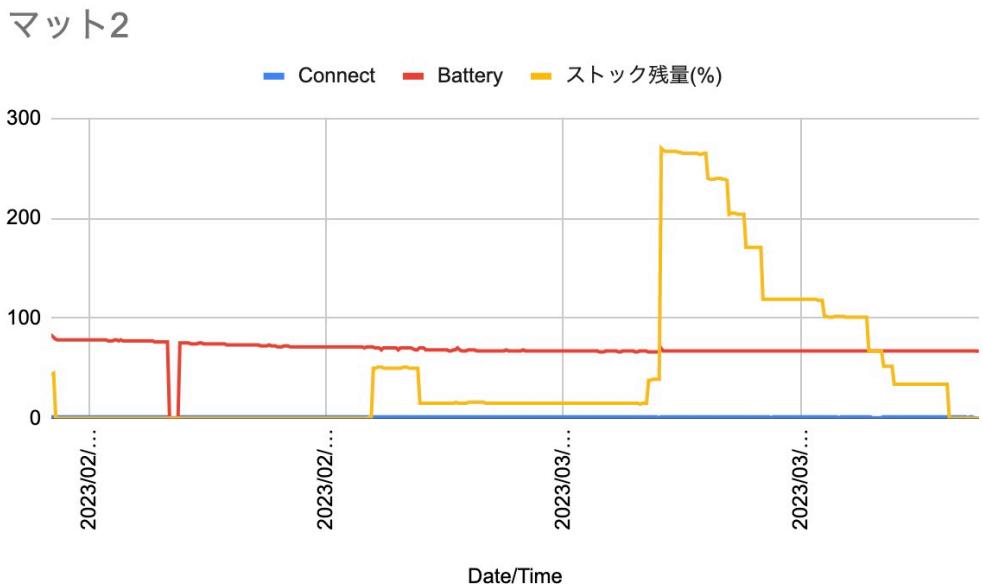


図 4.12: スマートマットライト2



- ・滞在人数の可視化(スマートAIカメラの活用)

図 4.13: スマートAIカメラ1

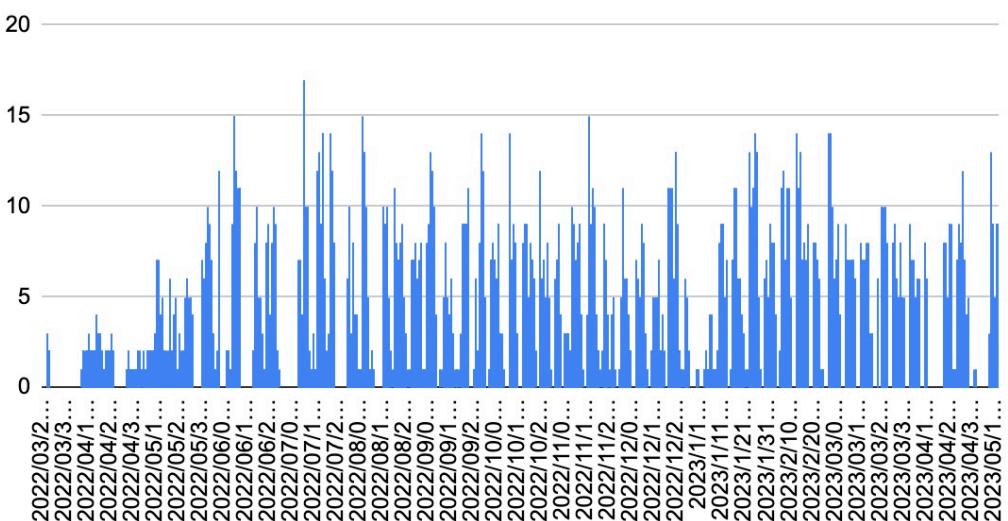
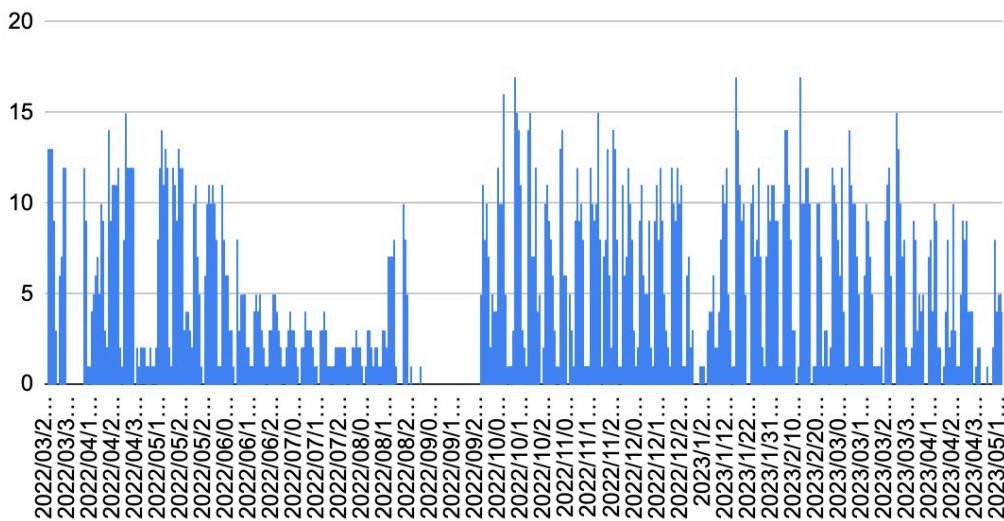


図 4.14: スマート AI カメラ 2



- ・職場環境の快適化 (IoT デバイスの活用)

図 4.15: CO2 濃度

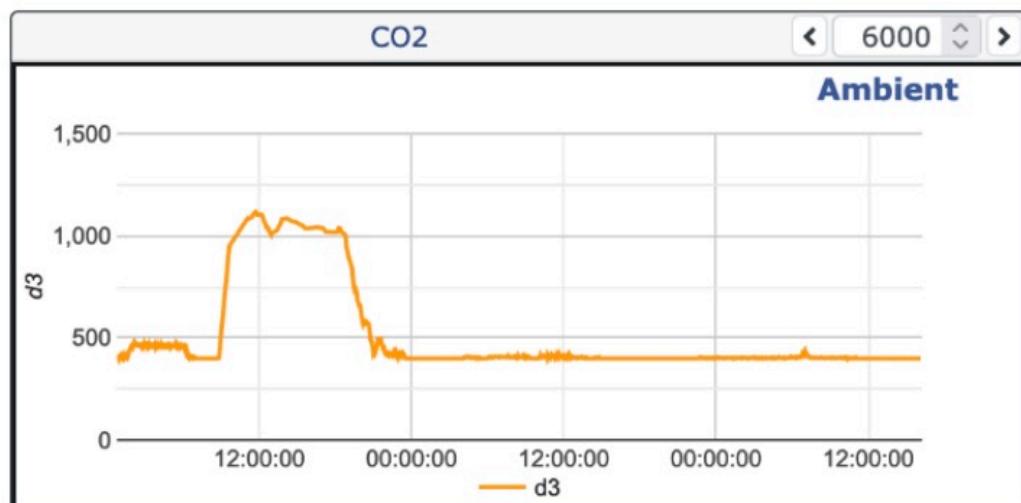


図 4.16: 気温

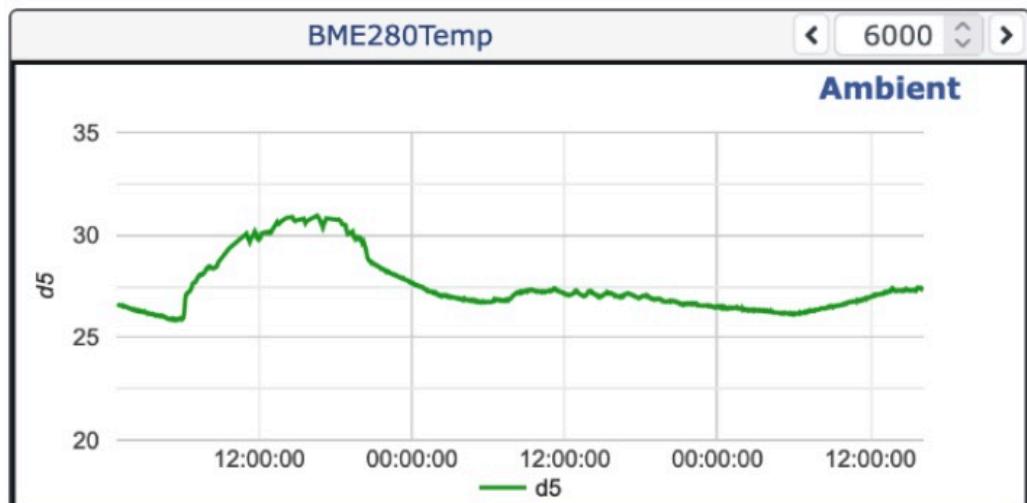
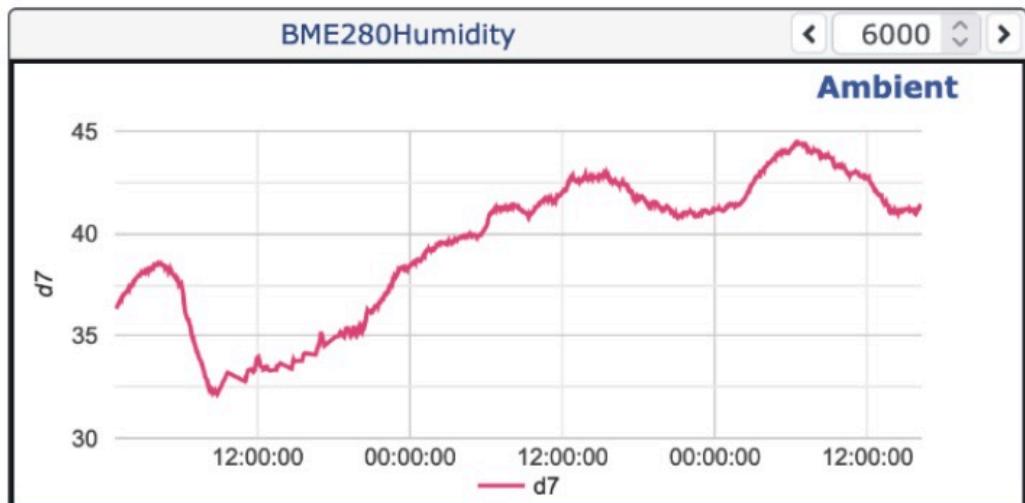


図 4.17: 湿度



4.7 断捨離(できなかつたケース)

- ・ 現場業務
 - 物理的に人を動かす必要がある場合には効率化に限界がある
 - ・ (郵便物、宅配便、請求書、訪問者対応、など)
 - 頻度を減らす、まとめて対応する、などの対応方法になる
- ・ 法的規制
 - XX法、施行規則、関連細則、安全衛生管理、消防法など、ルールを守る必要がある

- ・業務は勝手に止められない。頻度・回数など
- 用語の解釈にも注意が必要
 - ・「巡視」……人が見て回ること。ロボット化できない
- 政府によるアナログ規制撤廃⁵には期待しているが、対象・候補があまりなかった
- ・事務局の廃止(解散)

4.8 悩み(悩んだこと)

- ・Google Apps Scriptの引き継ぎ(技術スキルが必要)
 - 背伸びをせず、自分たちの身の丈にあったスキルレベルで実装が求められます
 - ・高いスキルを求めるに、後任者がメンテナンスできなくなり、技術的負債を生みやすくなります
- ・心構え
 - そもそも、その業務をなくせないか
 - 面倒なこと、苦痛なこと(つらみ)をなくす、緩和できないか
 - 自分の仕事がなくなる不安がある
 - ・(別の仕事の割り当てを先に示す必要がある)

4.9 気づき

4.9.1 よかったこと

- ・一緒に活動する仲間ができた(いろいろな人と知り合えた)
- ・チームワークがよくなった(団結力が向上した)
- ・相談相手ができた(メンタル面の支えができた)
- ・業務の中身を知ることができた
- ・問題・課題を具体的に把握できた
- ・集合知を生かすことができた(集合知の力を發揮できた)
- ・上司/部下それぞれの視点・視野・観点を理解するようになった
 - (客観的・中立的に物事を見られるようになった)
- ・事務/技術スキルが身についた
 - スプレッドシート、フォーム、GAS、など
- ・技術部門なのでアナログ業務(巡視)よりデジタル(コードベース)の方が理解しやすい、引き継ぎしやすい
- ・技術で解決する欲求が高まった(活動のモチベーションを維持できた)
 - 滞在人数可視化、職場環境のIoT化、計測(スマートマットもそのひとつ)

5. デジタル原則を踏まえたアナログ規制の見直し (https://www.digital.go.jp/assets/contents/node/basic_page/field_ref_resources/c43e8643-e807-41f3-b929-94fb7054377e/1420dca1/20221221_meeting_administrative_research_outline_08.pdf)

4.9.2 反省点

- ・ジョブディスクリプション(職務定義書)までは踏み込めなかった
- ・メンバーシップ型→ジョブ型雇用の夜明けが見られなかった
- ・(差し迫った状況ではなく)職場メンバーと(ふりかえりとして)座談会をやりたかった

4.9.3 副次的効果

- ・ESサーベイ⁶の向上（責任者・担当者の明確化、ひとりIT用務員の相談相手ができる）
- ・心理的安全性の確保(庶務業務の一覧化、見える化(誰でも見られる場所へ公開)、数値による定量化・客觀化、負担の平準化(得手・不得手、適材適所の場合を除く))
- ・担当者のローテーション(サイクル、任期、計画性が生まれた(我慢しよう、先が見えない不安から開放))

4.10 さいごに

2年間に渡る庶務業務整理がひと段落し、成功・失敗いろいろとありましたが、本書の執筆を機会にふりかえることができてよかったです。私が経験したベストプラクティスが、少しでもみなさまの「見える景色を変える」⁷お役に立てれば幸いです。

6. ES(Employee Satisfaction) 従業員満足度調査。従業員エクスペリエンス、従業員エンゲージメントと表現されることもある

7. 敬愛する沢渡あまね氏の著書「バリューサイクル・マネジメント」から表現を借用

第5章 対象とする業務が決まったら

小崎 肇

5.1 はじめに

対象とする業務が決まった段階では、業務の量、掛かった時間、難易度などの情報は得られても、業務の概要は見えていません。もし。この状況のままで作成を開始したら、開発しては疑問が発生し、疑問発生の都度質問をし、回答を待つ、その繰り返しに陥ります。これでは質問する方もされる方も時間が取られてしまいますので、まずは対象とする「業務の概要」をまとめていきましょう。「業務の概要」をあらかじめまとめておくことで、RPAに対してどのような機能を、どのように組み込むか設計することができます。設計せずに、RPAなどを作成する方もいらっしゃると思いますが、得てしてRPAとしてのゴールを見失いがちです。ゴールを決めるためにも、「業務の概要」として、「仕様書」を作成しておきましょう。

5.2 「仕様書」

仕様書は、業務を遂行する人と、RPAなどを開発する人との共通資料となるものです。まずは関係する人々が理解できる、統一した言葉を使ってまとめていきます。時として、部署外の人々にわからない言葉があるかもしれません。その場合は、脚注などに説明を加えていくといいでしょう。略称、略号に関しては、資料の中で統一されてあれば使用しても構いません。

仕様書を書くには、少なくとも、4つの項目(業務情報、入力情報、出力情報、処理情報)が必要です。

1. 業務情報：業務名の他、主となる担当者、いつ、どのように業務を行っていたかの情報をまとめます。
2. 入力情報：業務を行うには、基となる情報（データ）があります。その情報がどのようなものかをまとめます
3. 出力情報：業務を行った結果、新たな情報（データ）が発生します。その発生した情報が、どのようなものなのかをまとめます。
4. 処理情報：入力情報上のデータを、どのように抽出、加工するかをまとめます。細かくまとめる必要性はなく、概要がわかる粒度に留めます。
5. 補助情報：業務を行ったときに発生したことのある不具合、原因、その対応方法などをまとめます。

5.3 業務情報

図 5.1: 業務情報

業務名	○○部 ○○業務
業務概要	□□の情報を、●●し、△△を作成する
処理サイクル	□隨時 □日次 ■週次 □月次 □四半期 □半期 □その他
処理開始条件	月曜日の10:00～
先行業務（担当部署）	○○部からの○○ファイル生成
後続業務（担当部署）	不明（△△部 △△業務担当者）
ロボット名	

入力情報	処理情報	出力情報
<ul style="list-style-type: none">○○ファイルマスタファイル	<ul style="list-style-type: none">○○ファイルから条件合致するデータを対象とするマスタからメールアドレスを参照する。メアド単位のファイルを生成する	<ul style="list-style-type: none">△△ファイル△△ファイル電子メール

5.3.1 必要な情報

仕様書を作成する際に、必要な情報（1. 業務情報）は以下のような項目です。

・業務名

—「売上報告」「売掛情報」など、部署で認識されている名称を記入します。

・業務概要

—その業務が何をしているのかを、数行で記入します。

・処理サイクル

—その業務を毎日行うのか、週なのか、月なのか。それとも任意のタイミングで起動するのかなどを記入します。

・処理開始条件

—その業務を開始するキッカケ、条件を記入します。たとえば時刻、ファイルの有無、メールの受信などです。

・先行業務（担当部署）

—入力とする情報には、必ずその情報を出力した業務が存在します。開発に際して、直接影響する訳ではありませんが、内容を確認する窓口情報などがあると安心です。

・後続業務（担当部署）

—出力する情報には、普通であればかならず後続業務に入力情報を扱う業務が存在します。開発に際して、直接影響する訳ではありませんが、たとえば処理した出力ファイルの変更を打診するなど、連絡する窓口情報などがあると安心です。

- ・ロボット名

一仕様書を書く段階では不要ですが、開発の際のプロセス名、ファイル名などに使用します。

開発資産に応じた名称を、記入します。

5.3.2 任意な情報

仕様書を作成する際に、あった方がよいと思われる情報は以下の通りです。

- ・業務手順書の有無

一現在、業務を行っている人のメモが書き込まれたものではなく、部署として公式に作成されたものです。古い状態のものであっても、あるかないかを明確にしておきましょう。古い状態の場合には、この機に最新化してみましょう。

- ・業務での異常発生時の対策

一入力とするファイルのデータ品質が悪く、業務遂行できない、有事の際の対策を記入します。
たとえば、今は実行しなくても大丈夫なのか、先の業務手順書をもとに、手作業で遂行するのかなどです(※ これはRPAを改修、データを綺麗にするための猶予時間の確認のためであり、この対策があるからRPAの品質を下げてもよいというものではありません)。

5.4 入力情報

図5.2: 入力情報1

正式名称（呼称）	○○ファイル
現在参照するメディア	<input checked="" type="checkbox"/> Windows Local <input type="checkbox"/> Windows Server <input type="checkbox"/> Google Drive <input type="checkbox"/> SharePoint <input type="checkbox"/> その他
現在参照するフォルダ 固定部／可変部 可変部命名ルール 生成方法	G:¥マイドライブ¥@個人事業主¥仕訳ツール 固定 既存 / 手動 / 自動
現在参照するファイル 固定部／可変部 可変部命名ルール 1処理辺りの数 17ファイル辺りのデータ数 第三者制限	202201_○○名簿.xlsx “202201” 処理翌月の年月 1 / n 平均 件 / 最大 件 無 / 条件付き / 提示不可

図5.3: 入力情報2

正式名称（呼称）	〇〇ファイル
ファイル形式 EXCEL SHEET	シート名： 固定 ／ 最左端 ／ 不定（条件） 固定シート名「」 開始セル： A2 列情報：別添
	文字コード： SJIS ／ UTF-8 改行コード： LF ／ CRLF ／ 不定 セパレータ： カンマ ／ タブ ／ パイプ 列情報：別添
	MAIL
	アカウント：
WEB（起点URL）	別添遷移図参照
その他	

5.4.1 必要な情報

仕様書を作成する際に、必要な情報（2. 入力情報）は以下のような項目です。

- ・正式名称（呼称）
 - 正式名称の他に、部署で呼称する名称、略称も併記します。
- ・現在参照するメディア
 - Windows上のローカルフォルダー、ローカルネットワーク上の共有サーバー、Google Driveなどのプラットフォームがあります。
- ・現在参照するフォルダー
 - 入力情報が格納されている位置情報を記入します。Windowsでしたらファイル名を除くフルパス、Google Driveならフォルダー(FOLDERID)です。
 - 1. 固定部/可変部
 - ・フォルダーの情報の中で、固定部分、可変部分を見極めます。
 - 2. 可変部命名ルール
 - ・たとえば実行日を基準として、前月の年、月がフォルダーネ名になっている等のルールを記入します。
 - 3. 生成方法
 - ・そのフォルダーが、どのように作成されるのかを記入します。
 - ・たとえば、先行業務の担当者が手で作成している等です。

フォルダーの命名

とあるお客様の開発現場で、現在使用しているフォルダー情報をサンプルとして共有していただき、そのフォルダー情報の可変部分をサンプルの通りにしたRPAを開発したことがあります。受入試験をしていただき、合格をいただいて、本運用も順調に動いていました。しばらくして、そのRPAが異常終了したと連絡を受けました。原因調査を進めいくと、本来あるべきフォルダーの命名は"yyyy△MM"(△は半角空白)だったのですが、そのときのフォルダーは、"yyyy△△MM"だったのです。これは、先行業務の担当の方が手作業で作成した際のミスが原因でした。また先行業務の担当の方と相談して、RPA側で翌月のファイルを作成してあげ、極力自動化してしまうのも、異常終了リスクを減らす方法です。

・現在参照するファイル

一入力情報のファイル情報を記入します。Windowsでしたらファイル名、Google Driveならファイル(FILEID)です。

—1. 固定部／可変部

・ファイルの情報の中で、固定部分、可変部分を見極めます。

—2. 可変部命名ルール

・たとえばファイル名の一部に社員番号が含まれる等のルールを記入します。

—3. 1処理辺りの数

・入力情報のフォルダーに、そのファイルが何個格納されるかを記入します。

・フォルダーにひとつしかない場合、フルパスを指定すると狙い撃ちで読み込む処理を行えます。

・フォルダーにファイルが複数ある場合、全てのファイルを対象にするのか、いずれかのひとつを対象にするのか、確認が必要です。

—4. 1ファイル辺りのデータ量

・対象とする入力情報のデータ量を記入します。

・たとえば、1ファイル辺りのデータ量が数千だとしても、処理対象が複数の場合はそのファイル数倍のデータ量になります。

・1ファイル辺りのデータ量が巨大であったら、その入力情報のデータの持たせ方を変えるなど、検討が必要な場合も出てきます。

—5. 第三者制限

・部署で内製するのであれば、入力情報を第三者に開示する必要はないですが、外製とする場合は、個人情報などの扱いに気をつけなくてはならない情報が含まれている入力情報を開示しなければならない場合があります。そういった場合は、「条件付き」として、以下のような処置を施すことを検討する必要が出てきます。

・情報にマスクを掛ける

・情報量を減らし、正しいデータの桁数などに合わせたダミーデータを準備する

・レイアウトはそのままで、処理で使う情報以外を空白にしてしまう

・選択肢として「提示不可」も設定しましたが、このデータですと内製化も難しくなると思われますので、「条件付き」として扱える入力情報にする事を考えましょう。

—6. ファイル形式

- ・ファイル形式によって、読み込むべき方法が変わります。
- ・たとえばExcelには複数のシートが内包できるので、どのシートを対象とするのか確認することが必要です。
- ・CSVですと、文字コード（SJIS/UTF-8）、区切り文字（カンマ、タブなど）、改行コード（Lf/CrLf）を確認することが必要です。

0で始まる数字のデータ

とあるお客様の開発現場で、現在使用しているファイルをサンプルとして共有していただき、Excelファイルを入力にしたRPAを開発したことがあります。Excelファイル中に従業員番号という項目があり、その桁がまちまちなのです。業務のご担当様に確認したところ、CSVファイルをExcelで開いて、列を入れ替えたりしていることが判明。0で始まる数字のデータを含んだCSVファイルをExcelで開くと、先頭の0はなくなることは認識していたようですが、これまでは人が手で対応されていたそうです。もちろん、CSVファイルを入力にしたRPAに改良し、人手を介することはなくなりました。

5.5 出力情報

図5.4: 出力情報1

正式名称（呼称）	△△ファイル
現在出力しているメディア	<input checked="" type="checkbox"/> Windows Local <input type="checkbox"/> Windows Server <input type="checkbox"/> Google Drive <input type="checkbox"/> SharePoint <input type="checkbox"/> その他
現在出力しているフォルダ	G:¥マイドライブ¥@個人事業主¥仕訳ツール¥202202¥
固定部／可変部	“202202”
可変部命名ルール	処理年月
生成方法	既存 / 手動 / 自動
現在出力しているファイル	△△_202202.csv
固定部／可変部	“202202”
可変部命名ルール	処理年月
新規／更新	新規 / 更新
1処理辺りの数	1 / n
第三者制限	無 / 条件付き / 提示不可

図5.5: 出力情報2

正式名称（呼称）	△△ファイル
ファイル形式 EXCEL	シート名： 固定 ／ 最左端 ／ 不定（条件） 固定シート名「」
SHEET	開始セル： A2 列情報：別添
CSV	文字コード： SJIS ／ UTF-8 改行コード： LF ／ CRLF ／ 不定 セパレータ： カンマ ／ タブ ／ パイプ 列情報：別添
MAIL	アカウント：
WEB（起点URL）	別添遷移図参照
その他	

5.5.1 必要な情報

仕様書を作成する際に、必要な情報（3. 出力情報）は以下のようない項目です。

- ・正式名称（呼称）
 - 正式名称の他に、部署での呼称、略称も併記します。後続処理を扱う部署での呼称、略称もあるといいでしょう。
- ・現在参照するメディア
 - Windows上のローカルフォルダー、共有サーバー、Googleなどのプラットフォームがあります。
- ・現在参照するフォルダー
 - 出力情報として保存する位置情報を記入します。Windowsでしたらファイル名を除くフルパス、Googleならフォルダー(FOLDERID)です。
 - 1. 固定部／可変部
 - ・フォルダーの情報の中で、固定部分、可変部分を見極めます。
 - 2. 可変部命名ルール
 - ・たとえば実行日を基準として、前月の年、月がフォルダーネ名になっている等のルールを記入します。
 - 3. 生成方法
 - ・そのフォルダーが、どのように作成されるのかを記入します。
 - ・多くは、出力情報を格納する際に作成することが多いとは思います。部署間の場合には、後続の業務の担当者が、ご自身の手で作成する場合もあるかもしれません。
 - ・現在出力しているファイル
 - 出力情報のファイル情報を記入します。Windowsでしたらファイル名、Googleならファイル(FILEID)です。

—1. 固定部／可変部

- ・出力ファイルの情報の中で、固定部分、可変部分を見極めます。

—2. 可変部命名ルール

- ・たとえば出力ファイル名の一部を社員番号とする等のルールを記入します。

—3. 新規／更新

- ・出力情報のフォルダーに、出力ファイルを新規作成するのか、あるいは既存のファイルに追記するのかを記入します。
- ・新規作成の場合は、出力ファイルに対して上書きする方法、あるいは一度出力ファイルを削除してから改めて新規に作成する方法があります。
- ・追記の場合は、出力ファイルを事前に読み込む必要が発生します。

—4. 1処理辺りの数

- ・出力情報のフォルダーに、そのファイルが何個格納するかを記入します。
- ・特に処理には影響しませんが、後続処理には大いに関係しますので、しっかりとまとめ、確認しておきましょう。

—5. ファイル形式

- ・ファイル形式によって、書き込むべき方法が変わります。
- ・たとえばEXCELには複数のシートが内包できるので、シート名を固定にするのか、可変でいいのか、
- ・新規作成の場合、必ず存在する"Sheet1"にするかなど、後続処理の担当者と確認することが必要です。
- ・CSVですと、文字コード（SJIS/UTF-8）、区切り文字（カンマ、タブなど）を確認することが必要です。

—6. 第三者制限

- ・出力情報には、第三者制限を受けた入力情報が転記されている場合があります。他にも、業務ロジックで新たに生成された第三者に開示できない情報が含まれている場合もあるかもしれません。
- ・たとえばメールに添付する場合の誤送信対策を確実に行う必要性などの確認項目になります。

5.6 処理情報

仕様書を作成する際に、必要な情報（4. 処理情報）は以下のようにまとめます。

- ・入力情報の準備をする、使用する、出力ファイルを書き出す（追記する）など、情報の状態の変化を明示する。

一例)

- ・○ 所定フォルダーから、Tファイルを開く。
- ・× エクスプローラーで所定フォルダーを開き、中のTフォルダーをダブルクリックして開く。
- ・何の目的にしているのかを書き出す。その際に、手段・方法（どのように）は記述しない。

一例)

- ・○ TファイルのX項目から、MファイルのY項目を得る。
- ・× TファイルのX項目から、VLOOKUP関数を使用して、MファイルのY項目を得る。
- ・○ SシートのX項目から、値が100以上の情報を得る。
- ・× Sシートにオートフィルターを設定し、X項目から、値が100以上の情報を得る。
- ・業務上の条件分岐を記述する。その際、「処理対象とする条件」と対を成す「処理対象外とする条件」を併記すると理解しやすい。

一例)

- ・○ メールアドレスが設定されている情報から、電子メールを作成する。
- ・○ メールアドレスが設定されていない情報は、その情報件数を計測し、処理の最後に、その情報件数を載せた電子メールを作成する。
- ・業務上の繰り返し作業の範囲を明示する

一例)

- ・○ 明細ファイルがなくなるまで、以下の処理を繰返す。明細ファイルのデータをサマリファイルに転記する。
- ・× 明細ファイルのデータをサマリファイルに転記する(前述の入力情報で「1処理ファイルの数」でわかりますが...)。

5.7 補助情報

仕様書を作成する際に、必要な情報（5. 補助情報）は以下の通りです。

- ・想定される不具合、原因と対策

—先行する業務の出力情報が、本業務の入力情報になります。その時の出力情報の品質が悪いと、本業務での処理の妨げになります。ここでは過去に遭遇した不具合と原因、対策を記入します。

一例)

- ・不具合：ゼロで始まる数字の箇が、先行するゼロがなくなっていて、それ故マスタ上の値と合致せず、対象外となる。
- ・原因：先行業務では、CSVファイルを扱うのであったが、値の修正するシーンがあり、その際の修正にExcelを使用していたため。
- ・対策：正しい情報を入手する。あるいは、所定の桁になるまでゼロを前置する。

一例)

- ・不具合：A列から有効データが格納されているはずが、先行業務のメモ用に列が挿入され、有効データがB列からになっている。
- ・原因：先行業務の担当が後で調査するための情報を残したまま、本処理が実行されてしまった。
- ・対策：有効データの開始列を判断する文言を探し、その文字がある列から処理するようにする。

・「設定ファイル」に関する設定方法、参照方法

一たとえば、メールの件名のような固定的な情報は、RPA プログラム自身に持たせるのではなく、Excel のようなファイルに持たせ、そのファイルから情報を得る方法がいいでしょう。そうしておけば、開発が完了し運用フェーズに入ってから、もし業務の仕様が変わって、メールの件名を変更する場合にも、RPA プログラムを修正するのではなく、そのファイルを修正するだけで動きを変えることができます。そのような役割のファイルを「設定ファイル」と言います。「コンフィグファイル」とか、「CONFIG ファイル」と呼ばれることもあります。「設定ファイル」を準備したら、その管理方法を決めます。

—1. RPA プログラム内部の一部として管理する

- ・同じロボットを複数動かす時に、ロボットそれぞれに違う情報を与えることができる。
- ・逆も言え、同じロボットを複数動かすときに、同じ情報かどうかは保証できない。

—2. RPA プログラム内部とは、別区画に管理する。別区画の情報は、RPA プログラム内部の一部として管理するファイルにて管理する。

- ・同じロボットを複数動かすときに、ロボットそれぞれに同じ情報を与えることができる。
- ・別区画が使えなくなると、そこを参照する全てのロボットが実行できない。

5.8 結び

「業務の概要」をまとめるために必要な情報を列挙してきました。無論、これが足りないとか、これは要らないとか、色々なご意見があると思いますが、その都度決めていただければと思います。

本書で紹介した様式は、システム設計における図式化手法のひとつである、HIPO (Hierarchy plus Input Process Output) の様式を模したものでです。

HIPO は、「図式目次」と「IPO ダイアグラム」のふたつから構成されます。機能の階層(Hierarchy)を「図式目次」に記述し、図式目次の機能ごとに入力(Input)、処理(Process)、出力(Output)の関係を「IPO ダイアグラム」に記述するのですが、機能の階層より、入力、処理、出力の情報をまとめための「IPO ダイアグラム」を参考にしました。

この表はパワーポイントの「表」を挿入して作成しましたが、実際には、Excel ファイルなどにまとめた方が便利です。

- ・ファイル中の文字列を検索しやすい
- ・ハイパーリンクを設定して、展開されている情報をジャンプさせる
- ・設定した印刷範囲外に、メモ情報を残せる
- ・シートを保護し、情報を保護する

「業務の概要」として作成した「仕様書」は、レビューなどを経て、正式な書類として残すようにしていきましょう。無論、業務の内容が変わったら改修も必要ですし、業務が不要になったら文書の抹消も必要です。

無意味な作業

とあるお客様の開発現場で、A口ボットで使用しているプラットフォームを別のプラットフォームへと変更する改修を依頼されました。改修対象となるA口ボットを共有され、中身を解読し始め、改修概要をまとめた資料を作成し、現場担当者様との打ち合わせに臨んだのですが、開口一番、「A口ボットは現在運用していない」と。その場でB口ボットの改修に当たることになったのですが、共有されたB口ボットは、プラットフォーム変更以前に業務ルール改編による改修が進行中で、2週間ほど無意味な作業をしていたのでした。

最後に、入力とする情報は「ナマモノ」です。常に変化しています。そのような情報を扱う業務も「ナマモノ」と言えるでしょう。それを詳らかにしても、過去のものになっていきます。新しいパソコンを買った途端、古いパソコンと言われてしまうように。しかし、過去にものになるからといって、「業務の概要」をまとめない手はありません。改めて業務に向き合って、「業務の概要」にまとめてみてください。まとめてみることで、本当に必要な業務なのか？他で同様なことをしている業務ではないか？を知ることができます。「キッカケ」になります。これも「自動化」への一歩になります。

第6章 スマートマットをハックしてさらに便利にする

北崎 恵凡

6.1 スマートマットとは

あらかじめ登録した商品の重さを定期的に計測し、指定した条件になると自動で発注するためのスマートデバイスです。製品としてスマートマットライト¹とスマートマットクラウド²の2種類があります。スマートマットライトはAmazonの日用品(水やペーパータオルなど、約3,000の対象商品)の自動注文に特化しており、管理画面が用意されています。マットの価格³は安価(セール時～通常: 1,980～2,980円)で、マット10枚まで利用可能です。スマートマットクラウドは法人向けサービスで、棚卸自動化、入出庫管理、自動注文など、フルスペック・フルサービスの在庫管理・発注プラットフォームです。違いの詳細は製品のヘルプセンター⁴に掲載されています。

1.<https://service.lite.smartmat.io/>

2.<https://www.smartmat.io/>

3.<https://www.amazon.co.jp/dp/B08G8B2928>

4.<https://smartshopping.my.site.com/help/s/article/000001605>

図6.1: 戸棚のスマートマットライト



図6.2: ペーパータオルの在庫



6.2 スマートマットライト

スマートマットライトは、マット本体を購入すれば月額利用料は発生しません(買い切り)。マットは単三電池4本で動作し、Wi-Fiへ接続します。定期的に(デフォルトでは1日4回)重さを測定し、AWSへデータが送信されます(通信をキャプチャーして確認)。データは専用の管理画面⁵から、対象のマットを選択→詳細情報→消費レポートへ移動して確認することができます。

5.<https://lite.smartmat.io/>

図6.3: スマートマットライトの管理画面

The screenshot shows the SmartMat Lite management interface. At the top left is the logo "SmartMat Lite". At the top right are user profile and settings icons. The main area has a green header bar with the text "商品一覧" (Product List). Below it is a sidebar with links: "対応商品ラインナップ" (Supported Product Lineup), "ヘルプセンター" (Help Center), and "お問い合わせ" (Contact Us). A button "スマートマットを購入する" (Buy Smart Mat) is also in the sidebar. The main content area is titled "商品一覧" (Product List) and shows "マット登録台数: 10台中2台" (Number of mats registered: 2 out of 10). It lists two products: "W4118100" and "W4118100". Each product card shows a small image of a pink mat, signal strength bars, and a progress bar indicating connectivity status: 216% for the first and 0% for the second. A red callout box above the second product says "マットがWi-Fiに接続していない商品があります。? Wi-Fiの再接続方法" (There is a product that is not connected to Wi-Fi. ? Reconnection method for Wi-Fi). Below the product cards is a dashed box containing the text "新規マットのセットアップをはじめる" (Start setting up a new mat) and a plus sign icon.

図 6.4: 詳細情報

詳細情報

商品情報	
	Ar ハンドタオル 200枚入 5個パック >
100%時の商品重量	2.07kg >
注文設定	
注文方法	買いどき通知
注文タイミング	20%以下 >
重複注文防止機能	オフ
? 重複注文防止機能とは？	
残量情報	
先週は 67.1%	
昨日は 33.3% 商品を消費しました	
商品残量	216% >
消費ベース/日	8.1% >
最終計測日時 ?	2023/03/29 12:49
計測頻度	1日8回 >
↗ 消費レポートを詳しく見る >	

図6.5: 消費レポート

消費レポート

先週は**67.1%**、昨日は**33.3%**商品を使いました

注文回数

2回

消費ベース/日

8.1 %

使い切るまでの
平均日数

12.3 日

現在の商品残量

?

計測誤差がある場合

216 %

残量グラフ

1週間

1ヶ月

1年



6.3 スマートマットライト1と2の違い

結論から書くと、機能上の違いはほとんどありません。スマートマットライトの裏側のシール(図6.6の矢印)を剥がすとネジが1本現れ、プラスドライバーで外すことができます。側面はガラス板とプラスチックの境界(図6.7の矢印)にマイナスドライバーを差し込んで開けることができます。スマートマットライト1はESP-WROOM-02(ESP8266チップ)基板のWi-Fiアンテナを使用していますが、スマートマットライト2はWi-Fiアンテナ(図6.9の矢印)が外しとなり、ネットワークの接続性が向上しています。

図6.6: スマートマットライトの裏側



図6.7: スマートマットライトの側面

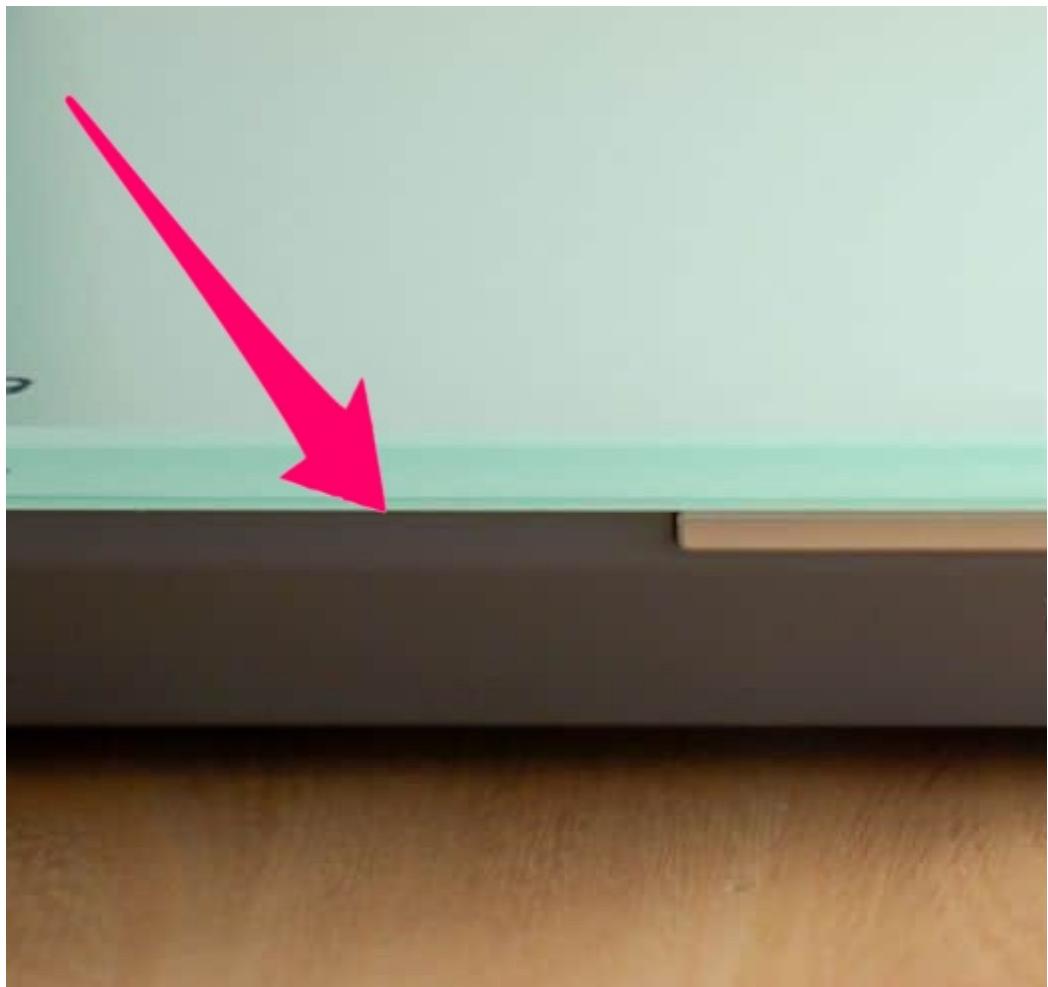


図6.8: 分解した中身(バージョン1)

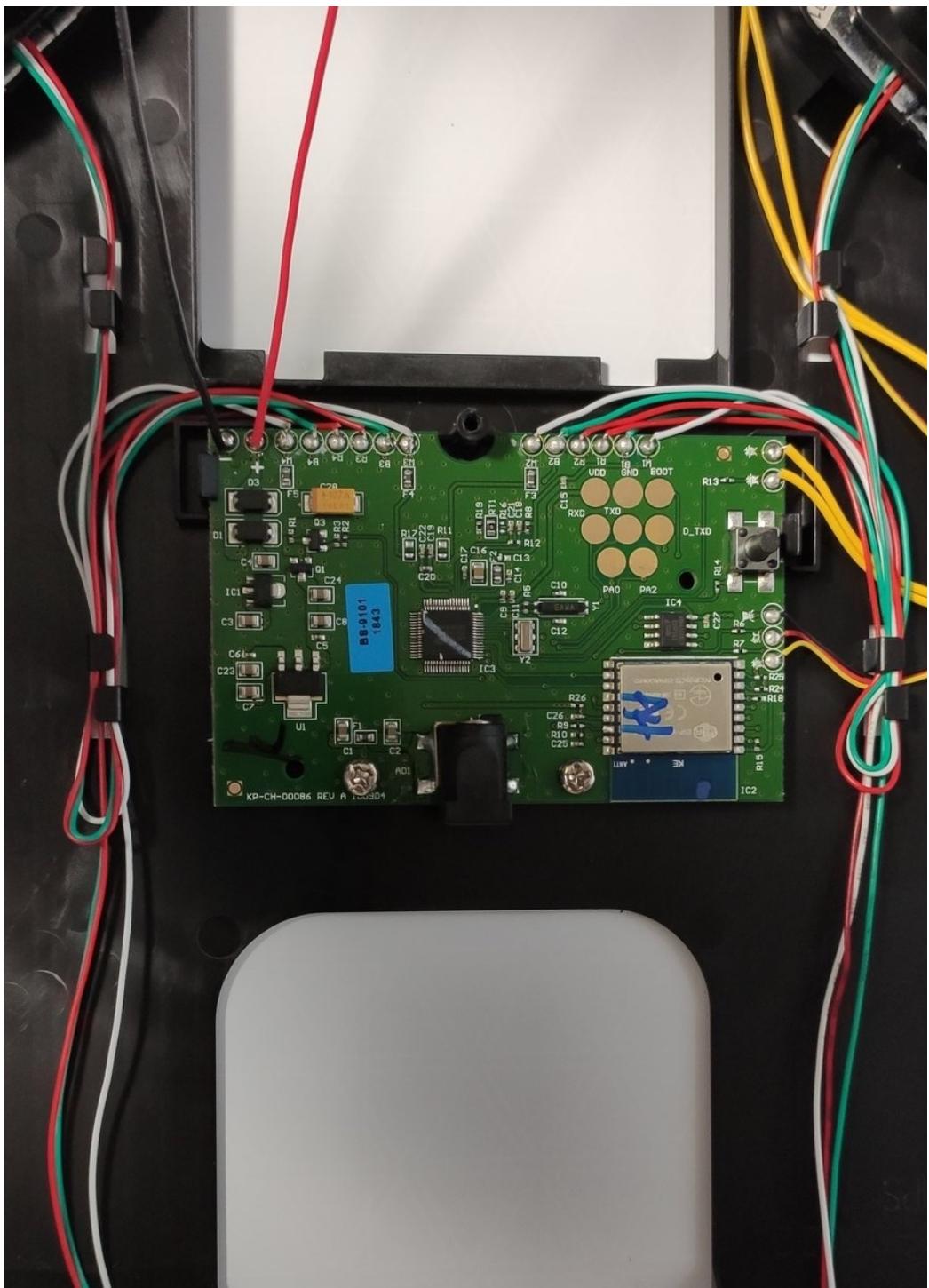
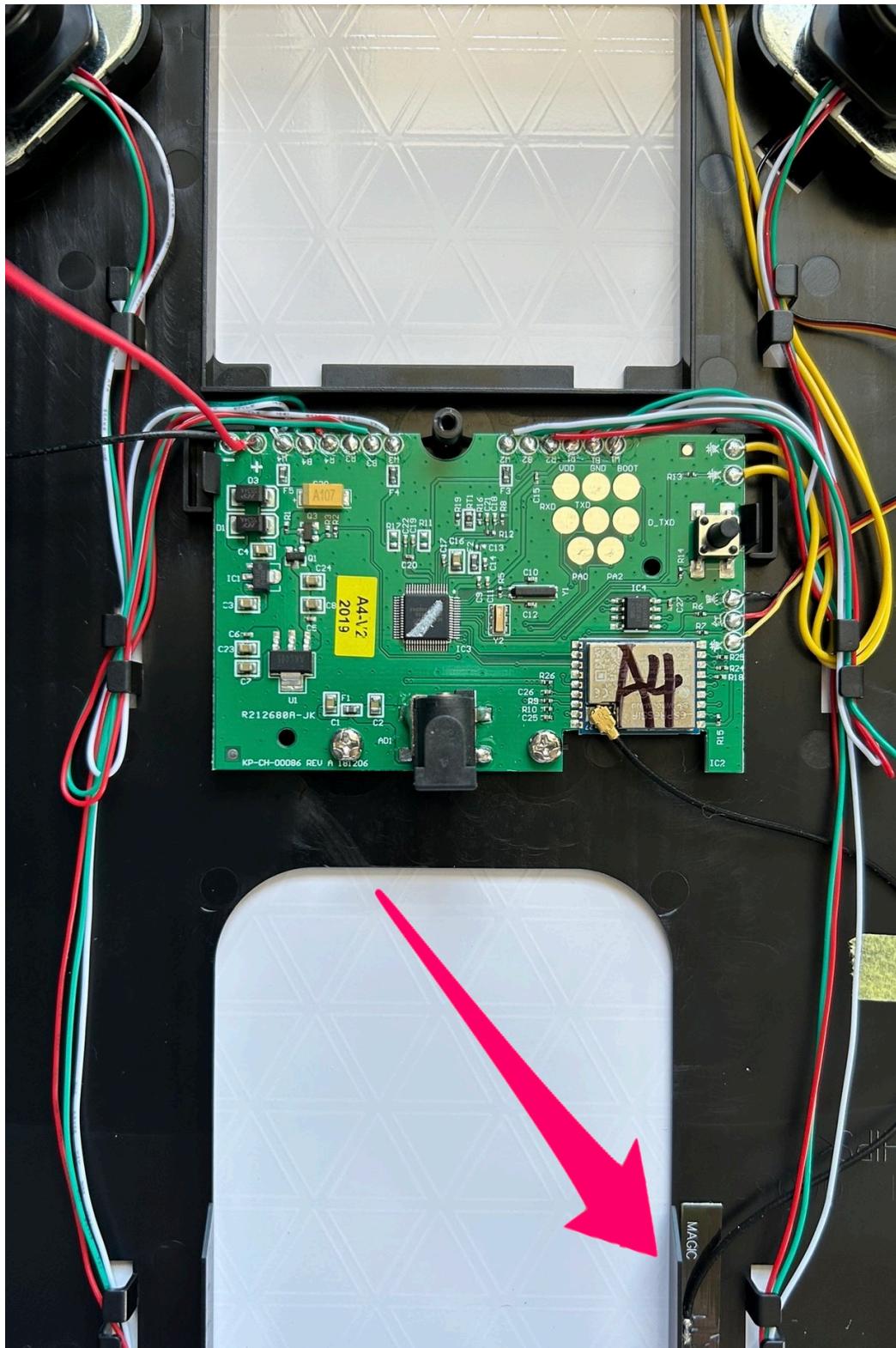


図6.9: 分解した中身(バージョン2)



6.4 スマートマットライトをハックする

スマートマットライトをそのまま使っても、Amazonアカウントと連携して自動注文してくれたり、EメールやLINE(ID連携が必要)で通知されるので非常に便利です。が、注文先を変更したり、消費データだけを利用(分析)したり、他の機能と連携させたい場合があります。これらのニーズを実現するため、スマートマットクラウドはAPI連携⁶が可能ですが、スマートマットライトはAPIが提供されていません。そこでこの記事では、Google SpreadsheetとGAS(Google Apps Script)を使用して管理画面の消費レポートからデータを取得し、他の機能と連携させる方法を説明します。

6.5 パケットキャプチャで通信の内容を確認

まず、macOSのインターネット共有の機能を利用して、スマートマットライトの通信内容をWireshark⁷でパケットキャプチャします。

図6.10: 接続構成



すると、以下の処理をしていることが確認できます。

1. DNSでネームを検索
2. HTTP POSTで測定データを送信
3. HTTP GETでデータを取得

6.<https://smartshopping.my.site.com/help/s/article/000001143>

7.<https://www.wireshark.org/>

図6.11: パケットキャプチャ

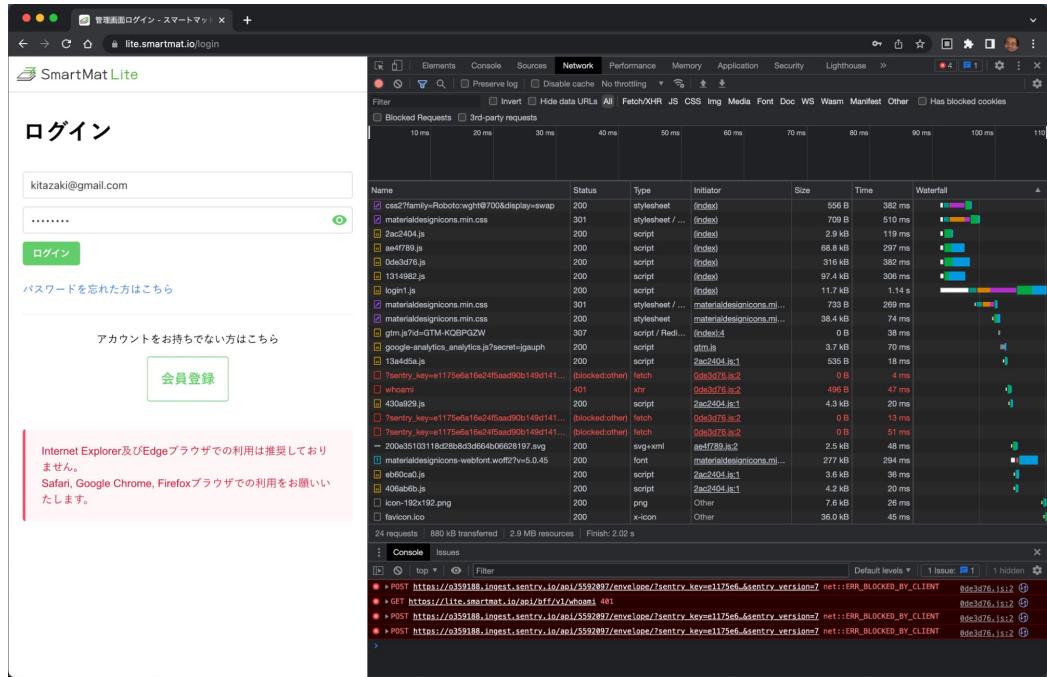
No.	Time	Source	Destination	Protocol	Length	Info
285	439.653946	Espresso_68:3d:3d	broadcast	ARP	68	ARP Announcement For 192.168.2.6
286	439.654016	Espresso_68:3d:3d	broadcast	ARP	68	ARP for host 192.168.2.11 Tell 192.168.2.6
287	439.876912	1a:3e:ref:fb:92:64	Espresso_68:3d:3d	ARP	42	192.168.2.1 is at 1a:3e:ref:fb:92:64
288	439.876913	192.168.2.1	DNS	84	Standard query 0x0000 A measure.lite.smarmat.io	
289	439.973548	192.168.2.1	DNS	281	Standard query response 0x0000 A measure.lite.smarmat.io CNAME k8s-smplrdmeasure-514e25ea6-873684100.ap-northeast-1.elb.amazonaws.com A 52.	
290	439.973549	192.168.2.1	TCP	68	HTTP 1.1 200 OK [SN] Seq=1 Ack=1 Win=542 Len=1420	
291	439.963167	192.168.2.6	IGMPv2	68	Membership Report group 224.0.0.1	
292	439.994545	192.168.2.6	TCP	58	HTTP 1.1 200 OK [SN] Seq=1 Ack=1 Win=542 Len=1420	
293	431.000511	192.168.2.6	HTTP/-	279	POST /v1/devices/1/version/2/_l1/l1.js , JavaScript Object Notation (application/json)	
294	431.000512	192.168.2.6	HTTP/-	279	POST /v1/devices/1/_l1/l1.js , JavaScript Object Notation (application/json)	
295	431.116264	52.198.167.198	HTTP/-	252	HTTP/1.1 200 OK [SN] Seq=1 Ack=226 Win=27872 Len=198	
296	431.341562	52.198.167.198	TCP	52	[TCP Retransmission] 68 -> 49238 (PSH, ACK) Seq=226 Ack=199 Win=542 Len=8	
297	431.358324	192.168.2.6	TCP	68	49238 -> 8 (ACK) Seq=226 Ack=199 Win=542 Len=8	
298	431.358325	192.168.2.6	TCP	68	[TCP] Seq=226 Ack=199 Win=542 Len=8	
299	431.358326	192.168.2.6	HTTP	162	HTTP/1.1 200 OK [SN] Seq=226 Ack=199 Win=542 Len=26	
300	431.929173	52.198.167.198	TCP	54	88 -> 49238 (SYN, ACK) Seq=134 Ack=27872 Len=8	
301	431.923232	52.198.167.198	TCP/-	288	HTTP/1.1 200 OK , JavaScript Object Notation (application/json)	
302	432.005518	52.198.167.198	TCP	52	[TCP Retransmission] 68 -> 49238 (PSH, ACK) Seq=199 Ack=338 Win=27872 Len=234	
303	432.005519	192.168.2.6	TCP	68	49238 -> 8 (ACK) Seq=199 Ack=338 Win=27872 Len=8	
304	434.124179	192.168.2.1	IGMP	99	Standard query 0x0000 PTR 0xa0a07000..sub._apple-nodev2._tcp.local, "QM" question	
305	434.124284	fe00::189:effff:fe..ff02::fb	MDNS	119	Standard query 0x0000 PTR 0xa0a07000..sub._apple-nodev2._tcp.local, "QM" question	
306	434.124284	169.254.184.228	IGMP	99	Standard query 0x0000 PTR 0xa0a07000..sub._apple-nodev2._tcp.local, "QM" question	
307	436.446841	192.168.2.6	DHCP	144	DHCP Discover - Transaction ID 0x73e977c2	
308	436.446841	0.0.0.0	DHCP	342	DHCP Discover - Transaction ID 0x73e977c2	
309	444.571535	0.0.0.0	DHCP	342	DHCP Discover - Transaction ID 0x73e977c2	
> Frame 221: 479 bytes on wire (3832 bits), 479 bytes captured (3832 bits) on interface en7, id 0						
> Ethernet II, Src: Realtek_7e:8f:7d (00:0e:4c:7e:8f:7d), Dst: IPv6cast_fb (33:33:00:00:00:fb)						
> Internet Protocol Version 6, Src: fe00::1cfa:9a:fe:8f, Dst: ff02::fb						
> User Datagram Protocol, Src Port: 5353, Dst Port: 5353						
> Multicast Domain Name System (query)						
00000	33.33.00.00.00.00	70.00.00.4c.78.07	04.06.00.00.00.00	33Lx.....	
00010	84.80.01.93.11.17	ff.0e.4c.7e.8f.7d	00.00.00.00.00.00	01.0c.00.00.00.00@.....	
00020	ba fe 85 f9 49 9a	ff 02	00 00 00 00 00 00	00 00 00 00 00 00@.....	
00030	80 00 00 00 00 00	ff 02	00 00 00 00 00 00	00 00 00 00 00 00@.....	
00040	80 00 00 00 00 00	ff 02	00 00 00 00 00 00	00 00 00 00 00 00@.....	
00050	63 73 2d 73 64 84	ff 02	00 00 00 00 00 00	00 00 00 00 00 00lb_d.....	
00060	63 73 2d 73 64 84	ff 02	00 00 00 00 00 00	00 00 00 00 00 00ns-sd_u.....	
00070	63 73 2d 73 64 84	ff 02	00 00 00 00 00 00	00 00 00 00 00 00dp_local.....	
00080	74.63.70.c9.1c.00	01.07.5f.72.64.6c	00.09.6e.tcp.....	01.07.5f.72.64.6c.09.6e	tcp.....@rdn	
00090	6b c9 30 00 8c 00	01.07.5f.72.64.6c	00.09.6e.tcp.....	01.07.5f.72.64.6c.09.6e	tcp.....@rdn	
000a0	6b c9 30 00 8c 00	01.07.5f.72.64.6c	00.09.6e.tcp.....	01.07.5f.72.64.6c.09.6e	tcp.....@rdn	
000b0	6c 80 01 84 5f 59	78.78	c8.30.00.8c.00	01.04.5f	_ipp 0.....	
000c0	78.74.78.c8.30.00	8c.00.01.07.5f.72.64.6c	00.09.6e.tcp.....	01.07.5f.72.64.6c.09.6e	tcp.....@rdn	
000d0	5f 69.78.78.75.73.62	c9.30.00.8c.00.00.01.08.5f.78	_ippsub 0....._p	5f 69.78.78.75.73.62.c9.30.00.8c.00.00.01.08.5f.78	_ippsub 0....._p	
000e0	72.69.78.74.85.5c	30.00.8c.00.00.01.07.5f.75.78	Rinter 0....._us	72.69.78.74.85.5c.30.00.8c.00.00.01.07.5f.75.78.73	Rinter 0....._us	
000f0	63.41.4c.73.48.80	ac.00.01.07.5f.69.67.5f.69.67	canc 0....._lps	63.41.4c.73.48.80.ac.00.01.07.5f.69.67.5f.69.67.73	canc 0....._lps	
000g0						

6.6 ブラウザの開発者ツールで通信の内容を確認

ブラウザの開発者ツールを使用して、スマートマットライトの管理画面にログインします。すると、以下の処理をしていることが確認できます。

1. サインインのURL(/api/bff/v1/signin)でID(email)とパスワード(password)を送信
2. クッキー(cookie)でセッションID(sid)を受信

図 6.12: ブラウザの開発者ツール



6.7 APIの仕様を推察

管理画面にログインした後、スマートマットのデータを確認します。詳細情報を確認すると、スマートマット毎にURLに管理番号(subscriptionId)が付与されています。

図6.13: スマートマットの詳細情報

The screenshot shows a web browser window with the URL <https://lite.smartmat.io/detail/8594>. The page title is "SmartMat Lite". On the left sidebar, there are links for "商品一覧", "対応商品ラインナップ", "ヘルプセンター", and "お問い合わせ". A green button labeled "スマートマットを購入する" is also present. The main content area is titled "詳細情報" and contains a "商品情報" section with an image of a pack of hand towels and the text "100%時の商品重量".

HTTP GETでデータ取得用のURL(/api/bff/v1/subscription/search_detail)にアクセスし、以下のJSONデータを取得しています。

```
{  
  "deviceSerialNumber": "WXXXXXXXXXXXXX",  
  "isFirstConnected": false,  
  "isConnected": true,  
  "battery": 66,  
  "remainingPercent": 0,  
  "triggerRemainingPercent": 20,  
  "measuredAt": "2023-04-08T06:06:03+09:00",  
  "current": 0,  
  "frequency": 8,  
  "subscriptionId": 8594,  
  "title": "Ar ハンドタオル 200枚入 5個パック",  
  "full": 2070.21,  
  "productUrl": "https://www.amazon.co.jp/dp/B00K0Z0ZFU",  
  "imageUrl": "https://m.media-amazon.com/images/I/414WdyYdVuL._SL500_.jpg",  
  "outputType": 4,  
  "orderable": false,
```

```

    "dailyAverageConsumption": 6.5,
    "dailyConsumption": 0,
    "weeklyConsumption": 0,
    "totalOrderCount": 4
}

```

これで、スマートマットクラウドのAPIから取得できるデータとフォーマットが一部共通であることを確認できました。

表6.1: データフォーマット

戻り値	説明	具体例
deviceSerialNumber	シリアル番号	W42190800XXX
isFirstConnected	(不明)	
isConnected	Wi-Fi 接続状況	true
battery	電池残量	73
remainingPercent	最新計測時刻の残量 % 表示の場合のみ	57
triggerRemainingPercent	閾値(%表示の場合)	20
measuredAt	最新計測時刻	
current	最新計測値(単位はグラム)	11330
frequency	計測頻度(1日 n回)	1
subscriptionId	(不明)	
title	商品名	お米
full	満タン重量 % 表示の場合のみ	20000
productUrl	(不明)	
imageUrl	商品画像 URL	
outputType	発注通知方法	メール通知
orderable	(不明)	
dailyAverageConsumption	(不明)	
dailyConsumption	(不明)	
weeklyConsumption	(不明)	
totalOrderCount	(不明)	

6.8 GASの実装

通信の確認とAPIの解析結果をもとに、GAS(Google Apps Script)を作成します。作成するスクリプトは大きくふたつの処理から成ります。

1. スマートマットライトの管理画面へログインし、スマートマットの測定データを取得する
2. 取得したデータをスプレッドシートに記録する

GASの中の変数に必要な情報を設定します。

- ・「email」 →管理画面のログインID
- ・「password」 →管理画面のパスワード
- ・「XXXX」 →スマートマットの管理番号
- ・「sheet_id」 →スプレッドシートID
- ・「sheet_name」 →スプレッドシートのシート名

GitHubにサンプルコード⁸を載せましたので、参考にしてみてください。

```
function fetchJSONData() {  
  
  const postdata = {  
    'email': '', // ID  
    'password': '' // Password  
  }  
  
  const loginUrl = 'https://lite.smartmat.io/api/bff/v1/signin'; // ログインURL  
  const dataUrl = 'https://lite.smartmat.io/api/bff/v1/subscription/  
search_detail?subscriptionId=XXXX'; // データURL  
  const sheet_id = ''; // SpreadSheetID  
  const sheet_name = ''; // SpreadSheetName  
  
  const options = {  
    'method': 'post',  
    'Content-Type': 'application/json',  
    'payload': JSON.stringify(postdata),  
    'followRedirects': false  
  }  
  
  // ログインページにPOSTリクエストを送信して、Cookieを取得する  
  const response = UrlFetchApp.fetch(loginUrl, options);  
  const headers = response.getHeaders();  
  const cookie = headers['Set-Cookie'];
```

8.<https://github.com/kitazaki/smartmat/blob/main/smartmat.gs>

```

// Cookieを使用して、JSONデータを取得する
const jsonData = UrlFetchApp.fetch(dataUrl, {
  'headers': {
    'Cookie': cookie
  }
}).getContentText();

// JSONデータを解析する
const parsedData = JSON.parse(jsonData);

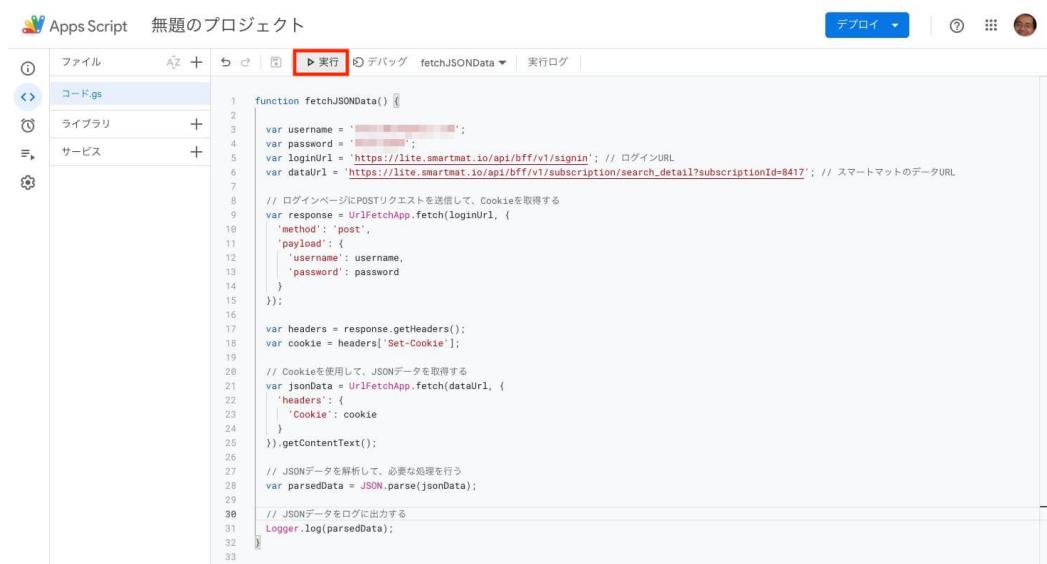
// スマートマットがWi-Fiに接続されている場合: true → 1、切断されている場合: false → 0
let connect = 0;
if (parsedData['isConnected']) {
  connect = 1;
}

// JSONデータをSpreadSheetに出力する
const MySheet = SpreadsheetApp.openById(sheet_id);
MySheet.getSheetByName(sheet_name).appendRow(
  [new Date(), connect, parsedData['battery'], parsedData['remainingPercent']]
);
}

```

GASを保存したら「実行」を押します。

図6.14: GASの実行



初めて実行する場合、承認が必要ですので、「権限を確認」を押します。

図 6.15: 権限を確認



次に、アカウントを選択します。

図6.16: アカウントの選択



「詳細」を押したあと、「(安全ではないページ) に移動」を選択します。

図6.17: 詳細の選択



このアプリは Google で確認されていません

アプリが、Google アカウントのプライベートな情報へのアクセスを求めてています。デベロッパー (kitazaki@gmail.com) と Google によって確認されるまで、このアプリを使用しないでください。

詳細

安全なページに戻る

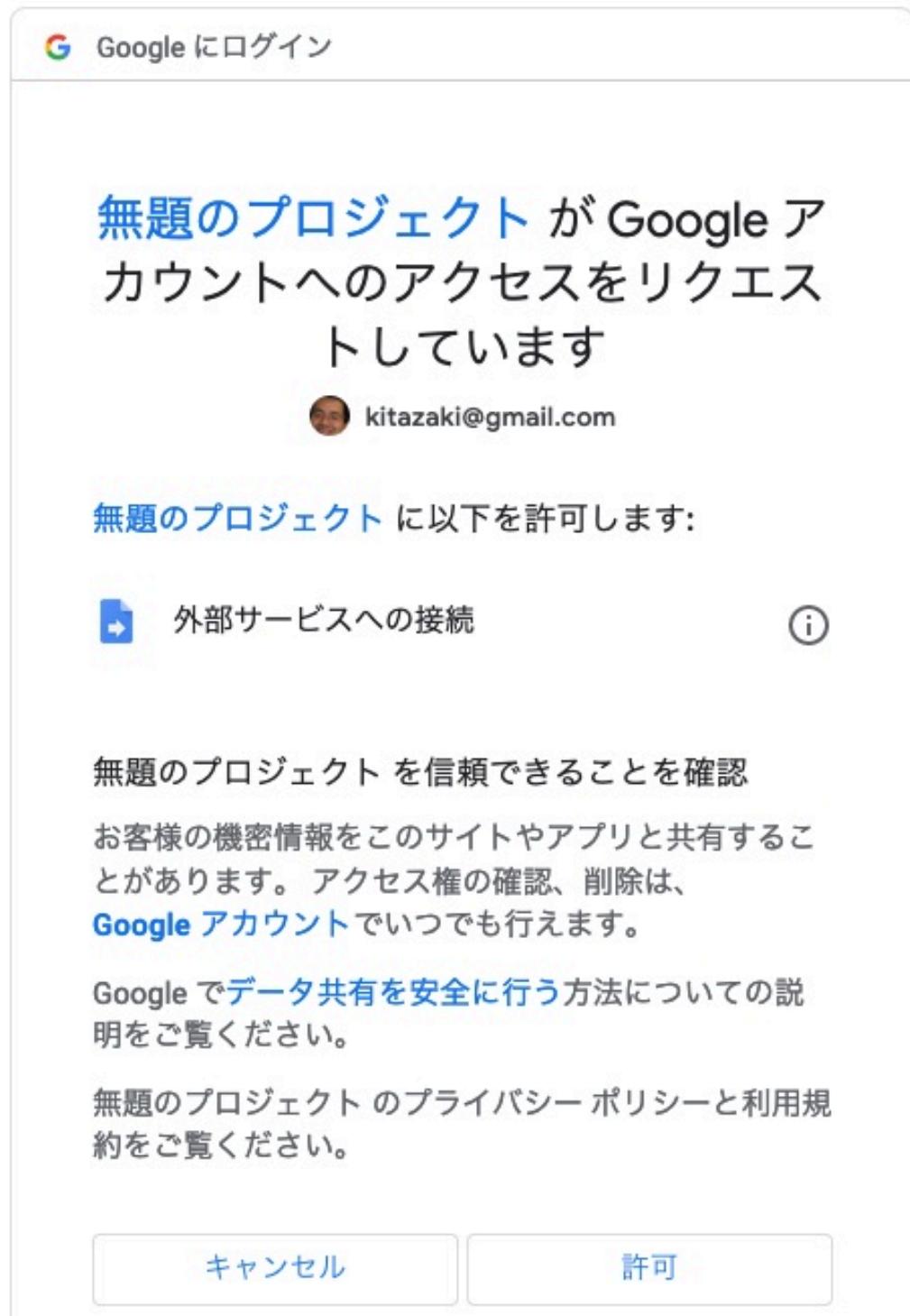
図6.18: (安全ではないページ) に移動の選択

リスクを理解し、デベロッパー (kitazaki@gmail.com) を信頼できる場合のみ、続行してください。

[無題のプロジェクト \(安全ではないページ\) に移動](#)

外部サービスへの接続で「許可」を押します。

図 6.19: 外部サービスへの接続

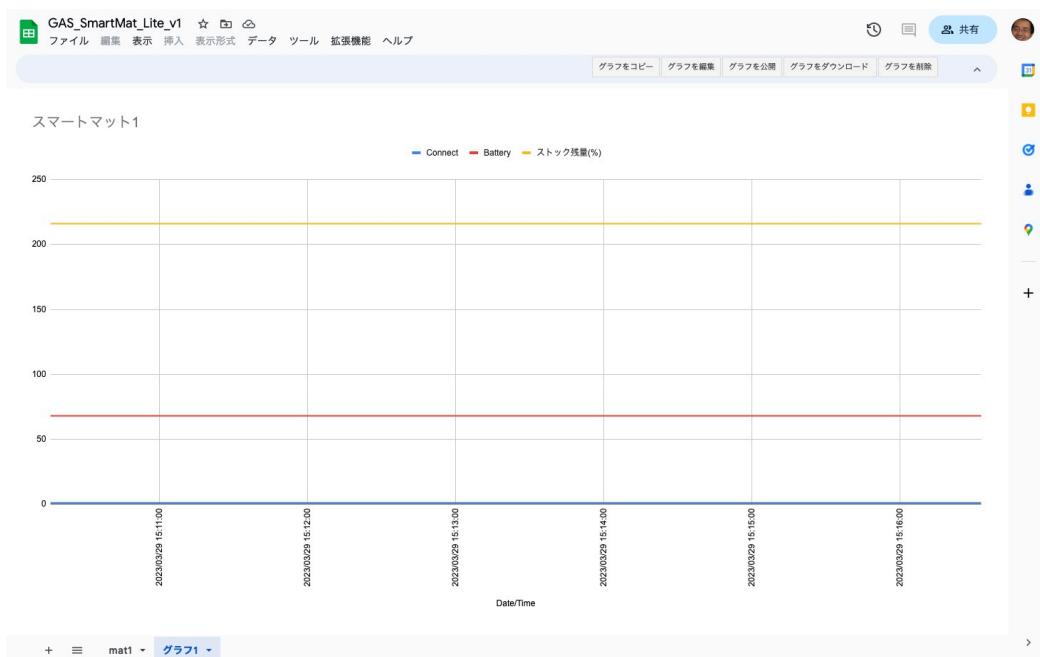


GASの実行が正常に完了すると、スプレッドシートにスマートマットライトのデータが記録されるので、記録されたデータを用いてグラフを作成します。

図6.20: スプレッドシート

	A	B	C	D	E
1	Date/Time	Connect	Battery	ストック残量(%)	
2	2023/03/29 15:10:16	1	68	216	
3	2023/03/29 15:16:33	1	68	216	
4					
5					

図6.21: グラフ



GASを定期実行する場合、トリガーを設定します。時計マークを押します。

図6.22: トリガーの設定



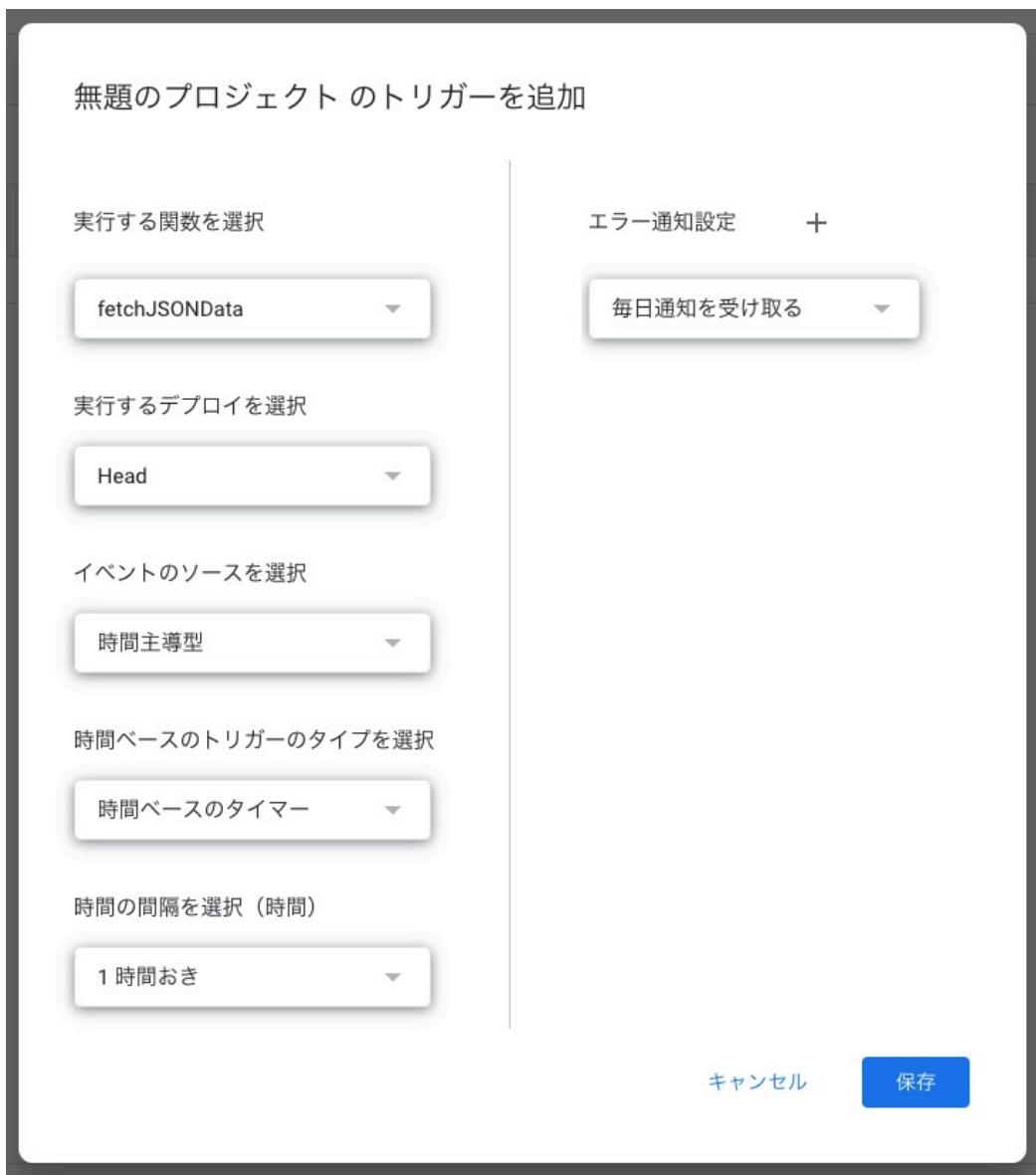
「トリガーを追加」を押します。

図6.23: トリガーを追加

+ トリガーを追加

- ・「イベントのソースを選択」 → 時間主導型
 - ・「時間ベースのトリガーのタイプを選択」 → 時間ベースのタイマー
 - ・「時間の間隔を選択 (時間)」 → 1時間おき
- を選択し、「保存」を押します。

図6.24: トリガーを保存

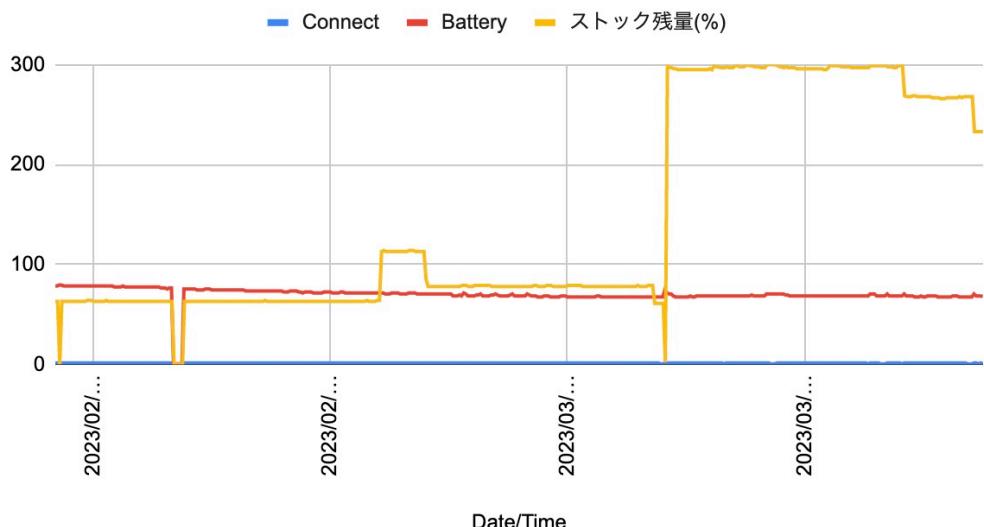


6.9 結果表示

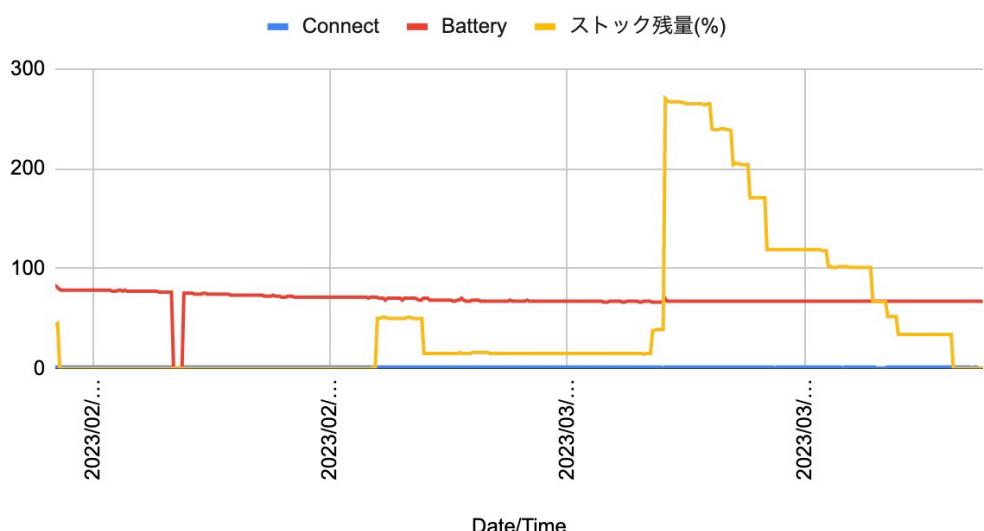
在庫の消費傾向に加えて、Wi-Fi接続状況と電池の消費傾向も把握できるようにしました。これで在庫がなくなる時期を予測したり、使い過ぎを警告することもできそうです。

図 6.25: Wi-Fi 接続状況・バッテリー容量・ストック残量のグラフ

マット1



マット2



6.10 さいごに

スプレッドシートと GAS を使用して、スマートマットライトをハックしてみました。みなさんもぜひ、いろんな IoT デバイスをハックして自動化を楽しんでみましょう！

第7章 RPAによるリアルタイム処理

鎌田 誠

7.1 RPAのメリット

RPAの導入目的として「生産性向上」と「品質向上」があります。24時間365日、文句も言わず動いてくれるRPAは、まさに生産性向上のために産まれてきた申し子のようなものですよね。品質向上についても、ヒューマンエラーをしっかりと排除してくれて、常に一定の品質を確保できます。

本章では、その一方であまり注視されていないRPAによる「リアルタイム処理」について語りたいと思います。

リアルタイム処理を実装することで、これまで実施できなかった業務ができるようになるなど、大きなメリットもありますので、ぜひ最後までお読みいただければ嬉しく感じます。また本章ではPower Automate for desktopで作成したサンプルフローも掲載致します。その他のRPA製品でも同様のフローは作れますので、よかつたら参考にしてください。

7.2 RPAのリアルタイム処理とは？

7.2.1 リアルタイム処理を知る

製造業でのRPA導入事例として「作業日報の入力作業自動化」など、よく上がる事例です。

図7.1: 作業日報

作業日報							
部署名	工程2	承認	審査	作成			
作業者名	Y						
日付	2023/4/18						
項目	加工指示番号	機種名	オーダー数	開始時間	終了時間	実数	備考
1	WO-202304170010	X1	100	8:00	12:00	100	
2	WO-202304170011	X2	200	13:00	16:00	200	
3	WO-202304170012	X3	100	16:00	17:00	35	残数65は、明日対応
4							
5							
6							
7							
8							

工場では、工程1→工程2→工程3→完成品のように、いくつかの工程を通して製品が生産されま

す。たとえば工程2のYさんが機種X1を100台、機種X2を200台、機種X3を35台の作業をしたとしたら、その実績を作業日報に記入します。

各工程ごとにその作業日報を取りまとめて（たとえば、工程2ではYさん含めて10名の方が作業をしていれば、10名分の作業日報）、基幹システムに投入します。作業日報ができるのは作業が終つてからですので、投入できるのは夕方か夜になりますよね。

そうすると、基幹システム上、正しい在庫が見えるのは全ての投入が終つてからになります。つまり、翌朝にならないと基幹システム上では正しい在庫が見れないのです。

これは正しい姿でしょうか？

常に最新の「在庫状況」が基幹システム上で見えることで、最適な判断ができます。最新の在庫が基幹システム上で見えない以上、現場に赴き在庫状況を確認しなければならず、非常に無駄な作業となります。

しかし、現実はそこに人手を割いてでも最新の情報にするのが難しく、結果、翌朝にならないと在庫が見えない状況になっている工場も多いです。

こういったケースにおいて、RPAによるリアルタイム処理が活きてきます。必要なときにシナリオを動かさのではなく、リアルタイム処理で常にシナリオが動き続け、必要な条件が揃った際に必要な処理をさせることができます。

先ほどの例では、Yさんが各機種の作業を完了した際にシナリオが自動的に動き始める仕組みです。これにより、常に最新の在庫状況が見れるようになります。

図7.2: 工程移動フロー(改善前)

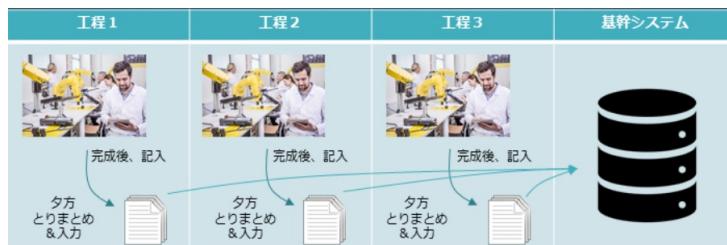


図7.3: 工程移動フロー(改善後)



7.2.2 リアルタイム処理のメリット・デメリット

このように、本来実施したかったにも関わらず、人が実施するには現実的ではないような作業でも、RPAなら実現可能なのです。ただし、RPAでリアルタイム処理を実現させるためには、いくつ

かのメリット/デメリットがあります。

メリット

1. 時常動いているため、都度都度シナリオを実行する手間がない。
2. 必要なタイミングで自動で処理を実行してくれる
3. 処理A,処理B等複数の処理を埋め込むことで、1ライセンスでも複数処理が可能

デメリット

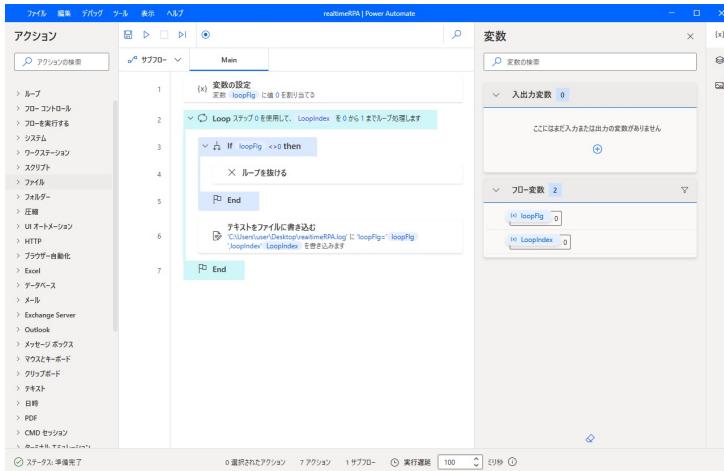
1. トリガーとなる電子的な仕組みが必要
2. シナリオが複雑化しやすく、エラー時の原因がわかりにくい
3. RPAでPCを占有してしまうため、RPA実行専用のPCが必要となる

このようにみると、いいことばかりではありません。特にシナリオが複雑化しやすく、トラブル時の復旧時間が長くなりがちです。その分、現場には負担を強いることになるので、可能な限りエラー原因が追跡できるように情報を残すようにし、復旧時間の短縮を意図した作りとしておきましょう。

7.3 リアルタイム処理の実装

リアルタイム処理を実装するには、処理が実行されるための条件が満たされたらどうかを常に監視する必要があります。そのため、無限ループを組んで、その中で発動条件を常に監視します。

図7.4: Power Automate for desktopによる無限ループシナリオ



まず、loopFlgを初期化(0を代入)します。次に、loopFlgが0以外の値になったらループを抜けるようにLoop処理を追加します。Loopの中では、loopFlgの値を確認し、0以外のときにループを抜けるようにしています。また、Loop内では、ログファイルにloopFlgの値と、Loop回数(loopIndex)を出力しています。

このLoop処理の中で、特定の条件が満たされた際に、所定の処理を実行するシナリオを作成すればいいわけですが、特定の条件とは、どのようなものがあるでしょうか？たとえば……

- ・10時、12時、15時のような特定の時間を迎えたとき

- ・5分毎、10分間毎のような特定の周期を迎えたとき

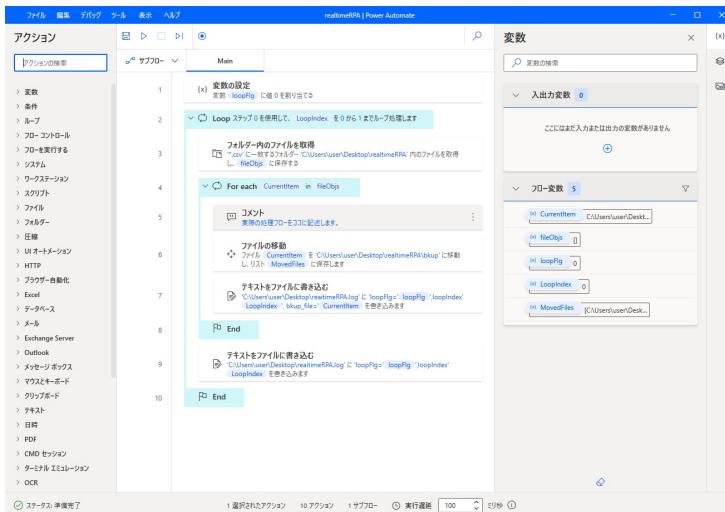
- ・あるファイルが生成されたとき

ぱっと思いつくのはこれぐらいでしょうか？

ここでは、「ファイルが生成されたとき」をトリガーとして処理するシナリオを考えていきましょう。

先ほどの作業日報の例で行くと、トリガーとしてはYさんの作業を完了させたタイミングになります。機種X1 100台分の作業が完了した際に、基幹システムにX1 100台と投入することで、在庫がリアルタイムで見えるようになります。そのためには、実績値を電子化する必要があります。筆者の例だと、データを入力するだけの簡単なフォームを作り、CSVファイルに出力するようにしました。今なら、Microsoft 365のFormsを使ったり、チャットツールから入力し、Power AutomateでCSVファイルを出力する、なんてやり方もありそうですね。

図 7.5: CSVファイルが生成された際のロジック



ここでは、指定したフォルダー内にCSVファイルがある場合に、コメント「実際の処理フローをココに記述します。」の箇所に処理フローを登録していきます。たとえば、これまでの例のように作業日報の登録であれば、CSVファイルを読み取り、基幹システムを操作して転記と登録を行います。併せて、処理済みのファイルはbkup フォルダーに移動させます。これは同じファイルで二重登録を避ける目的と、実際に登録したデータのエビデンスを残す意味があります。また、ログファイルも出力しています。ログファイルについては後ほど説明致します。

このようなフローを組むことで、基幹システムへの登録作業をRPAが人知れずリアルタイムに行ってくれるようになり、これまでどうしても実現できなかったリアルタイムでの正しい在庫管理ができるようになりました。

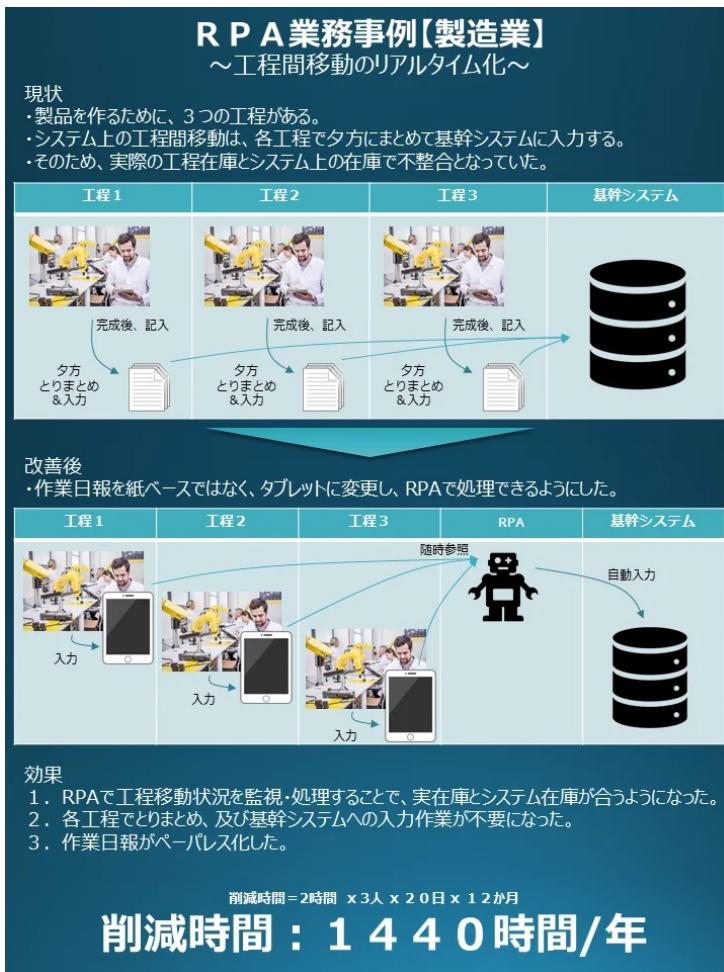
7.4 リアルタイム処理の適用事例

RPAにおけるリアルタイム処理の事例を見てみましょう。

7.4.1 製造業における工程間在庫移動

これまでの事例として上げてきたものが、製造業における在庫移動の事例になります。実際に筆者が情シス時代に構築したシナリオで、数値もかなりリアルな値になっています。では改めて処理フローを見てみましょう。

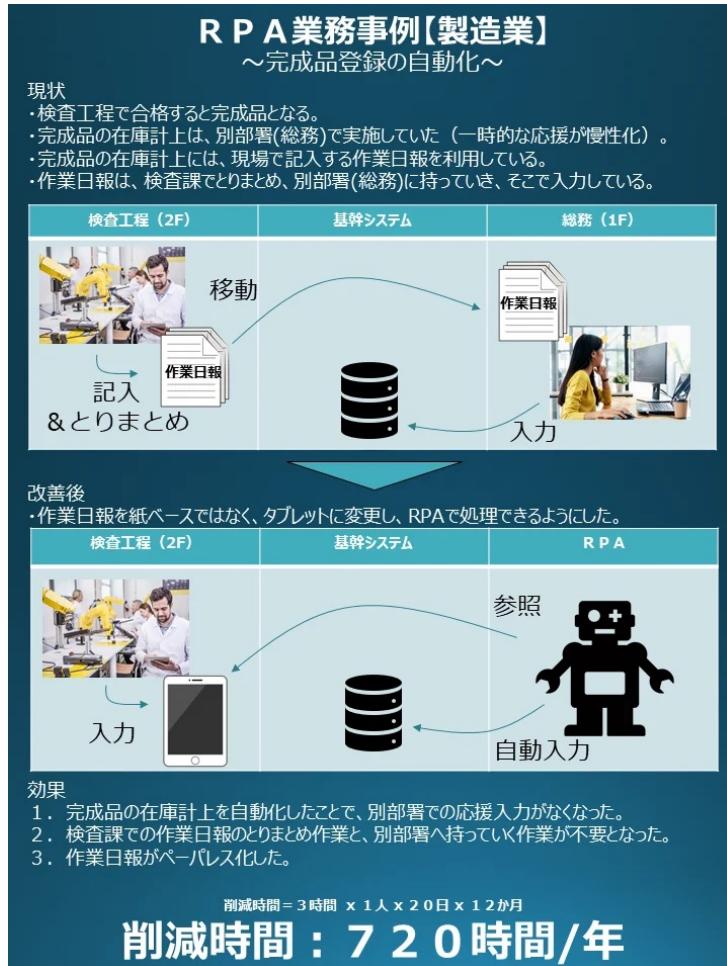
図 7.6: 工程間移動の実例



7.4.2 製造業における完成品入力

前項は在庫間移動でしたが、ここでは完成品入力になります。製造業においては、ひとつの機種を生産するのに、加工指示書と呼ばれる書類が発行されます。各工程は、この加工指示書に基づいて実際の生産を実施します。こちらも、私が情シス時代に実際に構築したシナリオです。完成品入力は、現場応援のため総務部門で入力することになっており、加工指示書と作業日報の物理的な移動（現場→総務）も発生していました。

図7.7: 完成品登録の事例



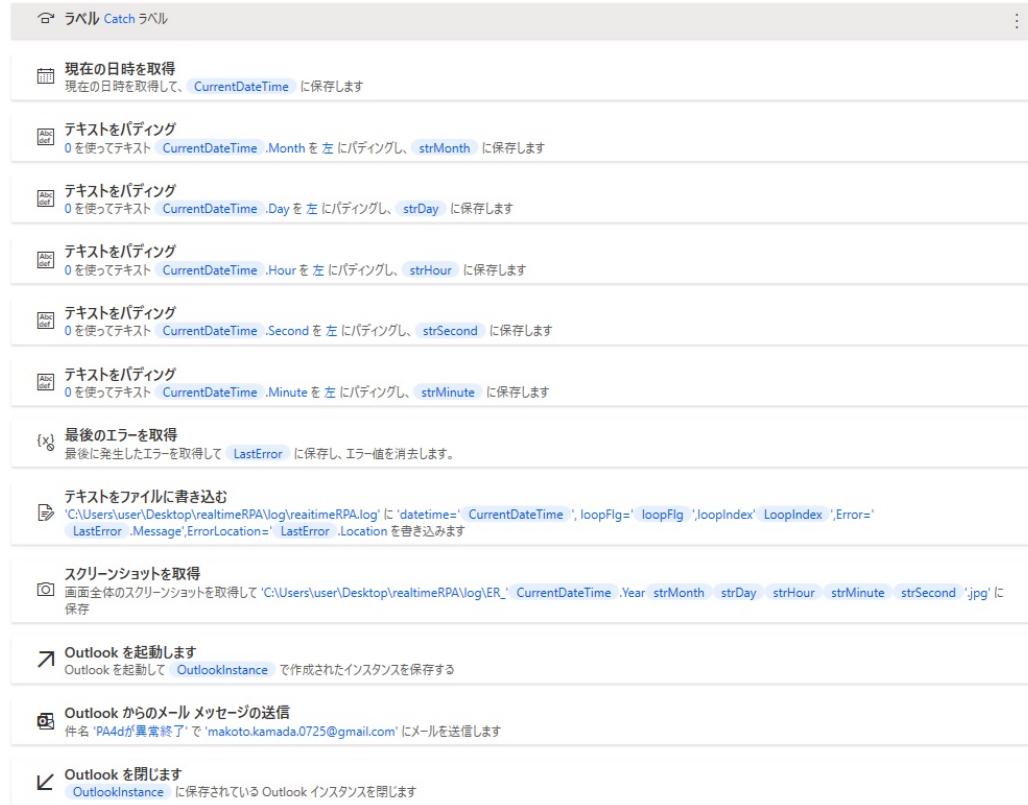
7.5 リアルタイム処理の課題と解決策

リアルタイム処理は、パソコンを占有してしまうため、常に人が見続ける（エラーなく処理が終わるか監視する）ことができません。そのため、エラーが発生した際にいかに早く気づくことができるか、またなぜエラーが発生したのかを知ることが大切です。

7.5.1 エラーをすばやく知るには

リアルタイム処理で動くシナリオは、通常、人が監視をしていないところで動きます。よって、異常終了等でシナリオが止まっていても知ることができず、現場の方からの指摘で知ることになってしまいます。そうならないためには、エラー時はしっかりとエラーをトラップし、管理者へ通知することが大切です。また、通知も一人だけではなく、関係者複数人に送るようにしましょう。

図 7.8: エラー時の処理例



こちらのフローは、エラー発生時に処理されるフローになります。ログファイルへの書き込みと、画面キャプチャを取得し、Outlookを使ってメール送信をおこなっています。今回はメールとしましたが、Teams等のチャットツールに送る方が、より早く気づいてもらえるメリットがあります。

7.5.2 エラーの原因を知るには

エラーでシナリオが止まってしまった場合を想定して、シナリオのどこまで進んだのか、その際の変数の状況はどうだったのかをファイルに保存しておくようにしましょう。また可能であれば、エラー時の画面キャプチャも保存しておくと、ログだけでは分からず状況が見え、解決のためのヒントとなる場合も多いのでオススメです。

実際にエラーで止まってしまった場合は、取得したログやキャプチャ画面等をヒントに原因を特定し対処していくことになります。そのために、ログファイルに記録すべき情報を検討します。一般的には、下記のような項目になります。

- ・ログの種類
- シナリオ名
- 発生日時
- 直前の処理名（行番号など）

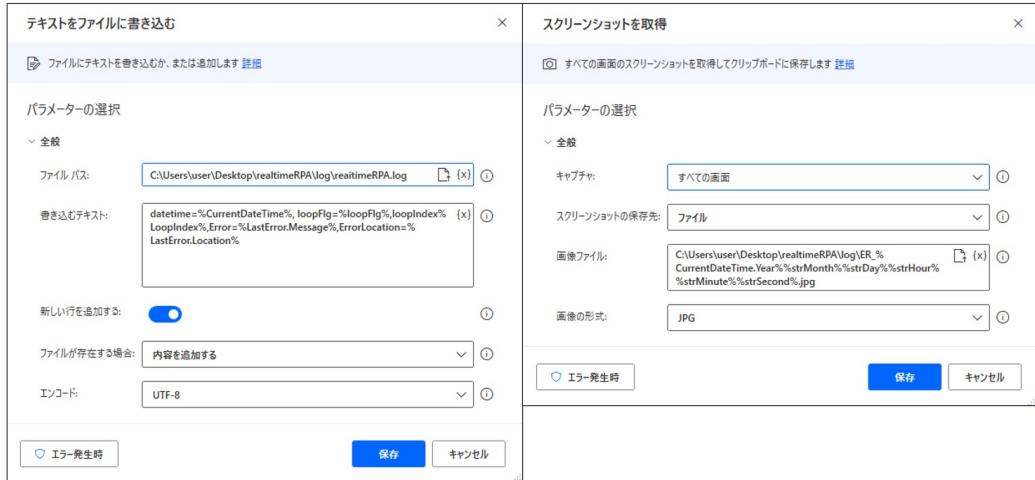
—エラーメッセージ

—変数の値

—(などなど)

図7.9の左側は、図7.8で示したフローにあるログ情報をログファイルに保存するノードの画面です。一方、右側はエラー発生時のスクリーンショットの記録です。

図7.9: エラー時のログ取得



シナリオをリリースする前に、トラブル対応に必要な情報（ログ）が十分に出力されているか確認しておきましょう。

また、登録が目的のフローであれば、登録がどこまで済んだのかの確認も必要となります。実際の登録状況と、監視フォルダーやbkup フォルダー、またログファイルの内容を見比べ、どこから処理を再開させればいいのかを判断する必要があります。

7.6 リアルタイム処理の今後の展望

RPAにおいては、生産性向上、品質向上の2点に注目が集められていましたが、このようにRPAを活用することで、人手ではできなかったことが実現でき、あるべき姿に近づくことができます。全てのシステムがクラウド化されれば、クラウドフローで事足りることかもしれません、まだまだオンプレミスの仕組みも多いですし、クラウドからオンプレミス回帰の流れもできつつあります。また、AIによる自動化等もRPAには追い風だと思っています。

7.7 最後に

今回記載した内容は、一部note記事にしてあったり、過去のRPACommunityのライトニングトーク大会でも発表した内容になりますので、併せてそちらも参照頂けると幸いです。

note - RPA 事例集について

<https://note.com/kerdy/n/nd2f20dad64f2>

YouTube - RPA でリアルタイム処理

<https://youtu.be/MMfycoq9tsc?t=789>

第8章 Raspberry PiとGCPを使って、SMSのE2E監視を実装してみた!

瀧川 大樹

8.1 はじめに

こんにちは！私は普段、某通信事業者のショートメッセージングサービス(以下 SMS と記載)を提供するシステムの保守業務をしています。本章では Raspberry Pi を使って SMS の送受信テストを自動化し、GCP の GAE と Cloud SQL で WebAPI を構築して、社内の監視システムに連携させた事例について紹介します。

8.2 E2E監視の重要性と構築した経緯

なじみがない方のために、SMSについて少し紹介させて頂きます。SMSは電話番号を送信先情報に指定し、テキストメッセージをやり取りするメッセージングサービスです。ユーザーコミュニケーションの用途としての利用は減っていますが、Web サービスの2段階認証の際のワンタイムパスワードをユーザーへ送付するような、システムからユーザーに送られるケースについての利用は増えています。社内では SMS 関連のシステムは重要インフラとして定義され、障害が発生した場合については、迅速な状況確認と復旧が要求されます。私たち運用担当は障害対応の際、まず、お客様が実際にサービスを使っているかどうかを最初に確認します。ここで、お客様が実際にサービスを使っているのか確認するために、End to End の監視(以後、E2E 監視と記載)は非常に重要です。

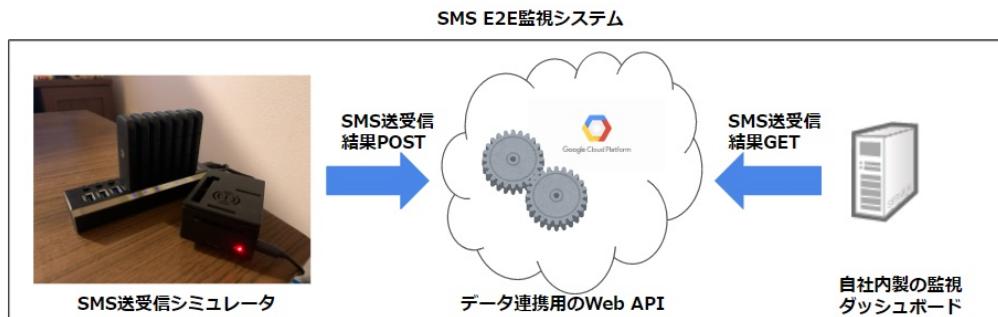
E2E 監視とは、システム外部からユーザーと同様の方法でアクセスして、そのシステムが正常に稼働しているか確認するための監視です。障害が発生した際に手作業で端末を使って試験をすると障害検知・復旧が遅くなり、お客様と約束している品質を守ることが難しくなります。そのため、私たちの部署では、システムの導入の際には、E2E 監視の実装をすることを必須の条件としています。E2E 監視の実装の内容としては、サンプルとしてお客様のリクエストをエンドポイントに実行し、その応答結果や応答時間を監視しています。

以前まで、SMS の E2E 監視は、SMS の E2E 監視専用のサーバーを構築して実現していました。SMS が利用するプロトコルは非常にニッチなこともありますし、SMS の E2E 監視専用のサーバーはベンダーが構築したものを利用していました。最近、この SMS の E2E 監視サーバーのサポート期間が終了することになり、構築をしたベンダーからリプレイスの提案を頂きましたが、その提案頂いた費用が非常に高額であったため、自社内製で SMS の E2E 監視を実装する検討を開始しました。

8.3 監視の構成

E2E監視システムは、SMSの送受信を行うSMS送受信シミュレータ、監視結果を表示する監視ダッシュボード、シミュレータと監視ダッシュボードを連携させるWeb APIの3つで構成されています。以下、それぞれについて少し詳しく説明します。

図 8.1: E2E 監視システムの構成



8.3.1 SMS 送受信シミュレータ

SMSのE2E監視を実現するために、SMSの送信・受信をシミュレートする方法を検討しネットで調べたところ、Raspberry Pi¹とUSB接続のLTE ドングルを使ってSMSを送受信を試しているネット記事を見つけました。Raspberry Piとは、教育目的に製造された小型コンピューターで、現在普及が進んでおり教育分野以外も様々な分野で利用されています。

また、USB接続のLTE ドングルは、外出した際にWifi環境がない場合に利用するUSB型のモデムのことを指します。Raspberry PiからATコマンドと呼ばれるモデムを操作するコマンドをシリアル接続でドングルに流し込むことにより、通話やインターネット接続やSMSの送受信などの携帯端末の操作を実行することができます。今回は、ATコマンドを使ったSMSの送受信の処理をスクриプトとして実装することによって、SMS送受信シミュレータを実現しました。詳しくは実装パートで説明します。

¹https://ja.wikipedia.org/wiki/Raspberry_Pi

図8.2: SMS送受信シミュレータ



8.3.2 監視ダッシュボード

私の属している部署で運用・保守しているサービスは、自社内製の監視ダッシュボードでE2Eの監視の状態を確認できるようになっています。監視の方式は統一されていることが望ましいと考えたため、SMS送受信シミュレータの送受信結果は監視ダッシュボード上に表示させることにしました。ただ、ダッシュボードは社内の商用ネットワークに属しており、SMS送受信シミュレータとダッシュボードを直接連携させるのはセキュリティー上避けたいと考えたため、連携については、後述するデータ連携用のWeb APIを利用して連携するような方式としました。

8.3.3 データ連携用のWeb API

シミュレータでのSMSの送受信結果をダッシュボードに表示させるために、部署で契約しているGoogle Cloud Platform(GCP)上にWeb APIを構築しました。社内でのクラウド利用については、ふたつの要件を満たせば利用が可能となります。Cloud環境を社内で使ってみたいと考えている方は、実装の前に、会社の利用ルールや部門の予算の状況を確認することをおすすめします。

8.3.4 予算の確保

クラウド利用に当たって、部門でクラウド利用のための予算を取っているため、利用に当たってどのくらいお金が要るのか、部門内で説明が必要でした。

8.3.5 社内のセキュリティーガイドラインとセキュリティーポリシーを満たすこと

実装に当たっては、社内のセキュリティー部門が提供するセキュリティーガイドラインを満たすように実装する必要があります。また、実装後に、社内のセキュリティーポリシー的に運用前には

必ず脆弱性診断を受け、診断結果が問題ないことを確認して初めて、実装したものと運用開始することができます。また、運用を開始してからも年次で脆弱性診断を実施する必要と、月次でセキュリティーパッチの適用調査をする必要があります。

8.4 SMS送受信シミュレータの実装

SMS送受信シミュレータの要件は大きくふたつあります。ひとつ目は、SMSの送受信を実行できること、ふたつ目はインターネットに接続し、SMSの送受信結果をWeb APIを使って登録できることです。

8.5 SMS送受信スクリプトの処理概要

ひとつ目の要件を満たすために、SMS送受信スクリプトを実装しました。本項では、SMS送受信スクリプトの処理の概要について説明します。

8.5.1 ATコマンド投入用のUSBデバイスファイルの認識

このドングルはUSBハブを介し複数本ドングルを挿して運用する前提条件があり、ドングルをRaspberry Piに接続するとUSBデバイスファイルが複数作成されます。これらのデバイスファイルがどのドングルと紐づくかの指定がデフォルトでは存在せず、スクリプトからシリアル接続する際にライブラリーでエラーとなってしまいます。このエラーを回避するため、デバイスファイル名の固定化と、スクリプト内でwvdial²用とATコマンド登録用でデバイスファイルのパスを定義しています。デバイスファイル名の固定化は、Raspberry Piのudevルールに定義を追加します。USBの接続ポートやドングルのシリアル情報を指定することができますが、利用しているドングルはシリアル情報がなかったため、USB接続ポートを指定してデバイスファイル名を固定化しています。

8.5.2 SMSの送受信

SMSを送受信する前に、SMSの送受信できる状態であるかどうか、ネットワーク状態(以下NW状態と記載)の判定を実施しています。NW状態の判定には、ATコマンドで電波強度確認、在圏状態の確認を実施しています。また、SMSの送受信を可能にするために、SMSモード設定コマンド等を実施しています。ドングルでSMSを送受信する際に注意が必要な点はふたつあり、ひとつ目はSMSの文字コードで、正しく設定しないと文字化けをします。実装したシミュレータではGSMに設定しています。ふたつ目はSMSの読み込み先、書き込み先の設定です。読み書きでそれぞれ端末かSIMカードで保存するかを設定することができますが、書き込み先と読み込み先を一致させる必要があります。今回の場合ドングル上かSIMカード上になりますが、受信したら一度削除するシナリオになっているので、SIM上を保存先として指定しています。

2.<https://wiki.archlinux.jp/index.php/Wvdial>

8.5.3 Web APIで送受信結果を Post

SMSの送受信の試行を実施後、Web APIのユーザーとパスワードで認証トークンを取得し、認証トークンでSMSの送受信結果をPOSTします。工夫したところは、WebAPIのユーザーとパスワードの秘匿性を上げるために、パスワード管理モジュールを使って、スクリプト内に認証情報を直接埋め込まないようにしています。スクリプト内にパスワード等の情報を埋め込んでしまうと、スクリプトが誤って外部に公開された場合、認証が危険化するリスクがあると考えたため、実装はやや煩雑になりましたが対策を実施しました。

8.6 インターネット接続処理の概要

ふたつ目の要件を満たすために、wvdialと呼ばれるPoint-to-Point Protocolダイアラをインストールしてインターネットへの接続を可能としています。インターネットへの接続には、wvdialの設定ファイルの編集とATコマンドを使ってドングルへの設定投入が必要でした。設定ファイルには、ドングルのデバイスファイルのパス、ドングルに搭載したSIMのAPNの情報を設定します。次に、ドングルへの設定についてですが、ドングルにATコマンドを投入して通信事業者のネットワークに接続します。設定に必要なATコマンドを「8.7.1 APNの設定」に記載しています。シミュレータはQuectel製のLTEチップが実装された、SORACOM Onyx LTE USB ドングル³を利用しておらず、「8.7.2 APNのユーザーとパスワードの設定」と「8.7.3 Packet Data Protocol(PDP)有効化」については、Quectel製のオリジナルのコマンドとなります。

8.7 ドングルに設定必要なATコマンド

8.7.1 APNの設定

AT+CGDCOUNT=1,"IP","利用するAPN名"

8.7.2 APNのユーザーとパスワードの設定

AT+QICSGP=1,1,"APN名","ユーザー","パスワード",0

8.7.3 Packet Data Protocol(PDP)有効化

AT+QIACT=1

8.8 Web APIの実装

Web APIはGCP上のGoogle App Engine(GAE)⁴とCloud SQL⁵を使って実装しました。また、運用の負荷をかけずに、実装は柔軟にしたいという考えでPaaSを利用して実装しており、GAEは認

3.<https://soracom.jp/store/7326/>

4.<https://cloud.google.com/appengine?hl=ja>

5.<https://cloud.google.com/sql?hl=ja>

証 API と登録 API と結果取得 API を提供しています。また、Cloud SQL にはシミュレータで実行した SMS の配信結果情報が登録 API で保存されています。なお、この Web API はインターネット上に公開しているため、以下のようなセキュリティーの対策を行っています。

8.8.1 ファイアウォールで接続 IP を制限

GAE はファイアウォールの機能を提供しており、GAE 上にアクセスする IP を制限することができます。今回接続するシミュレータの IP とダッシュボードの IP に接続を制限しています。接続するシミュレータの IP は事業者毎によって振られるものとなるため、広いレンジで制限をかけています。

図 8.3: ファイアウォールルール画面

The screenshot shows the 'Firewall Rules' section of the App Engine dashboard. On the left sidebar, 'Firewall Rules' is selected under the 'App Engine' heading. The main area displays a table of firewall rules:

優先度 ↑	アクション	IP 範囲	説明
500	許可		⋮
800	許可		⋮
900	許可		⋮
1000	許可		⋮
1200	許可		⋮
デフォルト	拒否		⋮

At the top of the main area, there are buttons for 'ルールを作成' (Create Rule), 'IP アドレスをテスト' (Test IP Address), '編集' (Edit), and '削除' (Delete). A note above the table says: 'App Engine のファイアウォール ルールを設定して、アプリケーションの発信トラフィックの安全性を高めることができます。詳細'.

8.8.2 認証キーの提供と有効期限の短時間化

結果取得 API、認証 API を実行するためには、認証 API で取得した認証キーを必要とするように実装しています。また、そのキーの有効期限については非常に短い期間に設定しています。これは有効期限を長くし、もしキーが第三者に漏洩した場合、情報漏洩や不正アクセスのリスクが高まるためです。

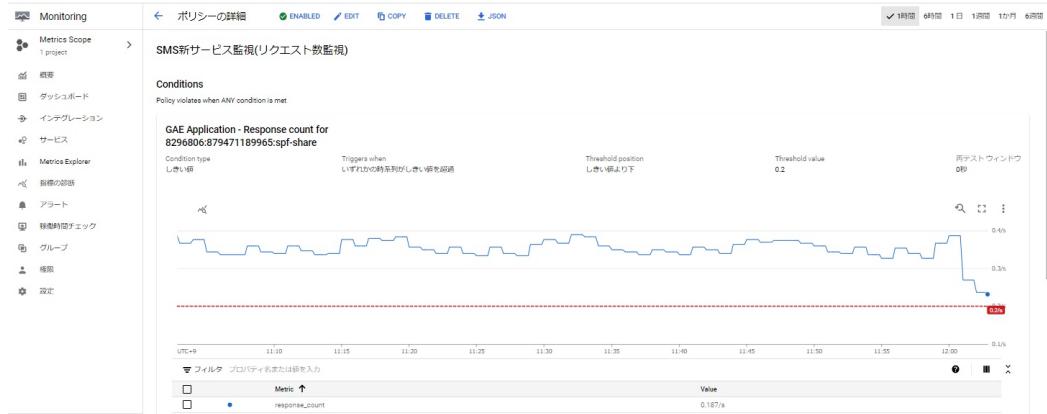
8.8.3 入力値のバリデーション機能とプレースホルダーの利用

スクリプトではバリデーションとプレースホルダーを実装して、SQL インジェクション対策として想定外の SQL が実行されることを防いでいます。公開されているオープンソースの Python ライブラリーを利用しておらず、バリデーションは「Cerberus」、プレースホルダーは「pymysql」を使用しています。バリデーションは API の入力値に対してルールを設定し、ルールに違反する入力値であればエラー応答を返します。プレースホルダーは入力値内に特殊文字があればエスケープ処理を実施します。

8.8.4 リクエスト数とリクエストエラーの監視

GCPが提供するCloud Monitoring⁶で、リクエストエラーとリクエスト数を監視しています。Cloud MonitoringではAPIの実行ログやメトリクスを確認できます。また、Cloud Monitoringが提供するアラート機能によって、メトリクス等に閾値を設定し、リクエストのエラーやリクエスト数が設定した閾値を超過した場合は、メール等で異常を知ることができます。

図 8.4: アラート機能画面



8.9 自社内製の監視ダッシュボード側の実装

シミュレータとWebAPIについては自身で実装をしましたが、自社内製の監視ダッシュボードの実装については、実装をする専門の部隊がいたため、実装を依頼しました。監視ダッシュボードに実装が必要な機能は、定期的に結果取得APIでSMSの配信結果を取得し、ダッシュボード上にOK or NGを表示させるというものです。実装を専門部隊にお願いするにあたって、E2E監視の構成やAPIの仕様書を起こして、要件を実装部隊に伝える必要がありました。E2E監視の構成やAPIの入出力の情報や期待する処理フロー等をパワポで作成して要件を伝えて、無事に期限内に実装がされました。今回、実装を依頼する際にAPIの仕様をパワポにわざわざ起こして実装を依頼したのですが、Google先生に聞いてみるとSwagger⁷と呼ばれる、Web API開発環境があることを知りました。Swaggerを使えばWeb APIの設計、ドキュメントの自動生成、テスト等が効率よくできそうなので、今後Web APIを構築する機会があれば使ってみたいと思います。

8.10 初期費用とランニング費用

構築と運用にかかる費用としては、以下の初期費用とランニングコストが発生します。基本料とSMS送信料金のランニングコストについては、月6万円程かかってます。現状、ランニングコスト

6.<https://cloud.google.com/monitoring/?hl=ja>

7.<https://swagger.io/>

に月10万円程かかっていますが、ランニングコストの累計額がベンダーから提示されたリプレイスの見積もり額に達するのは何百年先になるので、費用対効果については十分あったと考えています。

- ・初期費用
 - シミュレータの実装費用
 - SIM 初期契約料
- ・ランニング費用
 - GAE と Cloud SQL の費用
 - 通信事業者の基本料 + データ使用料
 - SMS の送信費用

以下、それぞれの詳細について紹介いたします。

8.11 初期費用について

シミュレータはRaspberry Piとドングルの費用で、1台大体2万円ぐらいかかります。半導体不足の影響でRaspberry Piの値段が上がっているようなので、今はもう少しかかるかもしれません。SIMの初期契約料については、MVNOのソラコムさんが提供している特定地域向けIoT SIMを契約していて、初期費用はDCMのSIMで3,300円、AUのSIMで1,650円でした。

8.12 ランニング費用について

8.12.1 GAE と Cloud SQL の費用

GAEについては、安定性と不要に料金追加されることを防ぐためにインスタンスについてはオートスケールではなく、固定化していて、念のためインスタンス数はふたつにして冗長化しています。

Cloud SQLに関しては処理も多くなく、ストレージの容量も必要ないので一番低いスペックとしました。クラウド側のランニングコストに関しては、1ヶ月にGAEが2万円、Cloud SQLが8千円程となっています。GAEとCloud SQLのスペックについては以下の通りです。データ処理量としては約0.5rec/sを処理しています。

- ・ GAE
 - 環境:Standard
 - リージョン:asia-northeast1
 - インスタンス数:2
 - インスタンスクラス:B1
- ・ Cloud SQL
 - リージョン: asia-northeast1
 - DB: MySQL
 - マシンタイプ: 標準
 - vCPU: 1
 - メモリー: 3.75

—ストレージ：10G

8.12.2 通信事業者の基本料+データ使用料

通信事業者の基本料とデータ使用料については、月300MB分のデータ使用量が基本料に含まれ、超過した際は追加でデータ料を支払う料金体系となっていて、契約したSIMにつき数百円/月の料金が発生します。

8.12.3 SMSの送信費用

SMSの送信費用については、自事業者同士のSMS送信に関しては無料ですが、異なる事業者同士のSMS送信については、費用が発生してしまいます。SMSの送信については1通3円かかり、サービス仕様上の1日に送信できる通数の制限が200通のため、上限にかかるないように送信をしています。

8.13 運用について

SMS E2E監視の運用について説明します。監視システムに関わるコンポーネントがシミュレータ、GCP、監視ダッシュボードと多いことから誤検知が多くなり、従来より監視品質が低下することが想定されました。そのため、以下の3つの対処を行うことによって、誤検知か障害かを切り分けています。

8.13.1 対処1: 複数の監視シナリオで監視する。

シミュレータに異常が発生するケースに対する対処です。2台のシミュレータで監視シナリオをふたつ用意しておき、ふたつの監視シナリオがNGとなった際は障害と扱うように運用フローとして立てつけました。この対処により、シミュレータ故障による障害の切り分けが可能となります。

8.13.2 対処2: 複数の監視手段を用意する

Web API側に異常が発生するケースに対する対処です。シミュレータ⇒Web API⇒監視ダッシュボードとは別の監視手段として、シミュレータからのSMSの送信・受信のログを監視しシステムのアラームとして発報させるようにしています。監視ダッシュボード上NGかつ上記のログ監視のアラームが発報した場合は、障害と扱うような運用フローとしています。

8.13.3 対処3: 複数のネットワークからAPIを叩く

監視ダッシュボードに異常が発生するケースに対する対処です。監視ダッシュボードとは別に誤検知確認ツールをGoogle Apps Scriptで実装しています。誤検知確認ツールは監視ダッシュボードと仕様は同じで、SMSの配信結果取得APIを定期的に叩いて、その結果をスプレッドシート上に表示させるようにしています。また、監視ダッシュボードと誤検知確認ツール双方がNGの場合、障害と扱うようにしています。

運用は少し煩雑となりますが、複数のパターンを用意しておき、組み合わせによって監視の確度を上げる対策をしています。E2E監視を入れていない場合は、システム監視担当が実端末を使って手動でSMS送受信のテストを実施していましたが、E2E監視を導入することによって、手動でのSMS送受信テストを実施せずによくなりましたが、現場の工数削減にも寄与できたと考えています。

8.14 まとめ

今回、Raspberry Piを利用したSMSサービスのE2E監視を実装した経緯、システム構成、システムの実装、運用についてご紹介させて頂きました。シミュレータの実装、GCPの機能、構築の費用、監視・運用等の雑多な内容になってしましましたが、何かひとつでも困りごとを解決する糸口になれば幸いです。

第9章 現場で使えるPython自動化入門

青木 敬樹

9.1 はじめに

これを手に取られた方は、実務で使用する自動化に興味がある方でしょうか？この章はPython自動化入門としています。しかし、一般書でも様々な自動化の手法やライブラリーの紹介が豊富にされています。そのため本章では、Pythonを実務で使用した際の私の経験に基づくTipsを詰め込んだいと思います。Python限定の話もありますが、言語に特定されず別のもので読み変えていただくこともできると思います。本章は

1. 頭をすっきりテスト駆動開発
2. みんなで読もう Sphinx
3. 最後の救いlog取得

の3つで構成されています。

9.2 頭をすっきりテスト駆動開発

9.2.1 はじめに

皆さんはテスト駆動開発という言葉をご存知でしょうか？アジャイル開発で有名なこちらのメソッド、提唱したのはKent Beckです。

まず初めにテスト駆動開発を知る前のこのメソッドに対するイメージは「面倒くさそう」というものでした。これは個人開発程度であればテストという行為そのものが必要ではなかったためであり、また成果物の品質を担保するためのテストとテスト駆動開発を混同していたためです。そのため、開発後の工程を取り入れる必要性がわからませんでした。しかし、今ではその恩恵にあづかっており、そのイメージは「多少手間はかかるが便利で有効」と変わっています。この説では、テスト駆動のメリットとその手法の紹介を行いたいと思います。

9.2.2 テスト駆動とは？

まずテスト駆動開発自体の紹介を行う前に、初めに私がなぜテスト駆動開発を必要としたのかの背景を説明したいと思います。皆さんの中で近しい状況があればこの節を読み進めるモチベーションになりますし、逆に読み飛ばす判断材料にもなると思います（もちろん、ぜひ読んでいただきたいところですが……笑）。

事の始まりは業務自動化用のスクリプトを書いていた所からでした。次のような、出力された文字列情報を解析して異常がないか確認するためのスクリプトです。

図9.1: 解析対象例

ファイルシステム	サイズ	使用量	残り	...
○○○	80G	5G	74.3G	
○○○○	42M	20M	21.5M	
○○○○○○	112M	5M	96.4M	
○○○○	31M	6M	24M	
○○○○	2.4G	2.1G	300M	

さらっと書けてしまいそうですが、上記の図のような物を判定するとき、いくつも考えなければならぬことがあります。たとえば、

- ・入力が想定外のものだったら？
- ・予期せぬ内容だったら？
- ・条件分岐はある？（組み合わせに問題はない？）
- ・型変換時ミスはしていない？

とくにPythonのように型を明示的に指定しない言語では、比較的潜在する想定外のバグが増えやすい傾向にあると思います。もちろんひとつずつ抜け漏れなく考えることができれば問題はありません。しかし気をつけることが多いため、実装している間は複雑なライラ棒をしているような気持ちになりました。さらに弊害としてひとつの機能を実装するだけでも疲れてしまい、なかなか思うように開発も進まずフラストレーションのたまる日々でした。

ここまで背景をお話しましたが、この悩みを要約すると

- ・沢山注意しながら開発をするのが大変
- となると思います。この悩みを解決するのがテスト駆動開発でした。

テスト駆動開発のステップ¹はKent Beckによると

1. まずテストをひとつ書く
2. すべてのテストを走らせ、すべて成功することを確認する
3. 小さな変更を行う
4. すべてのテストを走らせ、全て成功することを確認する
5. リファクタリングを行って重複を除去する

とされています。

ここでの大きなポイントはまずテストを書くということです。小さく機能を開発しテストを実施するというステップを繰り返せば、ミスをした瞬間にテストが指摘してくれます。初めはテストに引っかかるたびに小言を言われているような気がして億劫でしたが、機能の実装を完了してみたかったよりも時間がかかっておらず、疲れも少なかったことを今でも覚えています。これは頭のリソースを機能開発にだけ使うことができ、バグへの懸念はテストに指摘されたタイミングで修正するだけでいいというのが理由だと考えています。もちろん小さなスクリプトであればテストを最初に書くため、書かない場合より時間がかかることは否めません。しかしそのコードが長くなればなるほど複雑さが増すため、頭のリソースを機能開発にだけ向けることができるテスト駆動開発の方

¹Kent Beck (2017)「テスト駆動開発」和田 卓人訳 オーム社

が、効率が上がり開発時間が短くなる可能性があります。テスト駆動開発はいわば長距離走向けの手法と言えるでしょう。

あくまで私の体験談で1サンプルのお話でしたが、ここまでテスト駆動開発が広まっていることを考えると試す価値はあると思います。この説ではテスト駆動開発の紹介のみにとどまりますが、興味を持たれた方は、Kent Beckの「テスト駆動開発」もしくはRobert C. Martin「Clean Code」²の第5章を参考にしてみてください。

9.2.3 誰がためのテスト駆動？

さて、ここまで素晴らしい手法としてテスト駆動開発として説明してきましたが、私はテスト駆動開発は常に使うべきとは考えていません。理由としては、プロジェクトで開発する以外の場面ではテストを成果としづらいと考えているためです。もちろん、チーム内でテストを残す文化があれば別ですが、チームで成果と見なされないものに工数を掛ける行為はなかなか継続しないものです。

そもそもの話となりますが、テスト駆動開発で作成するテストと品質のためのテストでは目的が異なります。テスト駆動開発で書くテストは開発者の注意点を網羅していればよく、カバレッジをそこまで意識しなくとも問題はないと思われます。品質担保のための成果物としてテストを残すためにはカバレッジを上げる必要がありますが、そのカバレッジ向上と成果物の品質の向上は対数曲線的な関係にあると考えており、幾ら細かいバグをテストしていてもスクリプトの機能自体は向上しません。

趣味ではなく現場で活用することを考えると、工数も切り離せないファクターのひとつです。成果に対して労力を必要以上にかけることは無駄であると考え、割り切りが必要ではないか？と考えています。逆に言えば、ある程度品質が求められるのであればそれなりの工数が必要となるため、テストする工程を別で考えた方がいいと思います。

以上のことから、私は求められる品質を考慮したうえで開発のためのメソッドのひとつとして、効率を上げるためにテスト駆動開発を利用することを心掛けています。

9.3 みんなで読もうSphinx

9.3.1 はじめに

いきなりで恐縮ですが、ツールを作成することの恩恵は自分で使う場合のみにとどまらないと考えています。そしてむしろその多くは他の方に使ってもらうことにより得られるものだと思っています。もちろんツールの種別によるとは思いますが、適切に抽象化されたメソッドは他の人に利用、継承してもらうことで複利的にその真価を發揮し続けます。

しかし、このような文化を形成していくためには、誰かにその仕様を適切に伝える必要があります。そのためにはドキュメントが必要となるでしょう。皆さんはツールを作成するときにどのようなドキュメントを作成していますか？ReadME.mdを作成する、エクセルで作成する、テキストで作成するなど様々だと思います。

2.Robert C. Martin (2017) 「Clean Code アジャイルソフトウェア達人の技」花井 志生訳、KADOKAWA

本節ではドキュメントを作成するために、Sphinx³というドキュメントジェネレータをご紹介したいと思います。

ドキュメントジェネレータとはその名の通りドキュメントを作成するためのツールです。百聞は一見にしかずということで、まずはSphinxの活用事例を見てみましょう。以下はOpen3Dという点群処理用のライブラリのドキュメント⁴です。

見ていただくとわかる通り、html形式のwebサイトのようなドキュメントです。

図9.2: Open3D ドキュメント

The screenshot shows the Open3D documentation website. The top navigation bar includes the logo, version (0.17.0), and search bar. The main navigation menu on the left has sections for 'GETTING STARTED' (Introduction, Getting started, Build from source, Link Open3D in C++ projects, Build documentation, Open3D-ML, ARM support, Docker) and 'TUTORIAL' (Point cloud, Visualize point cloud, Voxel downsampling, Vertex normal estimation, Access estimated vertex normal, Crop point cloud, Paint point cloud, Point cloud distance, Bounding volumes, Convex hull, DBSCAN clustering, Plane segmentation, Planar patch detection, Hidden point removal, Mesh). The current page is 'Point cloud'. Below the menu, there's a sub-navigation for 'Point cloud' with options like 'Visualize point cloud', 'Voxel downsampling', etc. The main content area displays a 'Point cloud' tutorial. It starts with a brief description: 'This tutorial demonstrates basic usage of a point cloud.' Then it moves to 'Visualize point cloud', which includes a code snippet and an image of a 3D point cloud visualization.

```
[2]: print("Load a ply point cloud, print it, and render it")
ply_point_cloud = o3d.data.PLYPointCloud()
pcd = o3d.io.read_point_cloud(ply_point_cloud.path)
print(pcd)
print(np.asarray(pcd.points))
o3d.visualization.draw_geometries([pcd])
Load a ply point cloud, print it, and render it
[Open3D INFO] Downloading https://github.com/isl-org/open3d_downloads/releases/download/20220201-[Open3D INFO] Downloaded to /home/runner/open3d_data/PLYPointCloud/fragment.ply
PointCloud with 196133 points.
[[0.65234375 0.84686458 2.37898625]
 [0.65234375 0.83984375 2.38408572]
 [0.65234375 0.83984375 2.37898625]
 ...
 [2.00839925 2.39453125 1.88671875]
 [2.00390625 2.39408506 1.88671875]
 [2.00390625 2.39453125 1.88793314]]
```

一言で言ってしまえば、Sphinxはこのhtmlドキュメントを作成するためのツールです（もちろん、htmlに限らずLatexやePub, Texinfo, manual pages, plain texといったフォーマットにも対応しています⁵）。類似のツールとしては、Doxygenなどが有名です。

Sphinxのようなドキュメント生成ツールで、ドキュメントを作成した場合のメリットをいくつか挙げてみると、

1. ドキュメントがきれい
2. コメントが残る
3. 検索ができる
4. レイアウトを統一しやすい

3.<https://www.sphinx-doc.org/>

4.<http://www.open3d.org/docs/release/tutorial/geometry/pointcloud.html>

5.<https://www.sphinx-doc.org/ja/master/>

といったところでしょうか。

まずドキュメントがきれいであることは明白です。ただのテキストよりはこちらの方が読みやすいです。次にコメントが残るという点について、詳しくは後述しますが、実はこのSphinxで生成するためのドキュメントの情報はプログラムにコメントとして残すのです。またプログラムにコメントとして残されているために、テキストとして残ることもメリットとなると思います。エンジニアとしては、データをなるべく環境に依存しない方法で記載をしたいものです。この点においてプログラムのコメントとして残っていれば安心です。

さらに、このSphinxでは検索機能もついてきます。ユーザーフレンドリーなUIがあれば、他の方にも使ってもらいやすいです。最後に、レイアウトを統一しやすいという点も上げられます。いろんな方がテキストでドキュメントを作成するとそのレイアウトは多種多様になり、その読みやすさも書き手によってまちまちになります。

最終的にドキュメントの質はその内容に依存しますが、レイアウトが整えられていることで読み手はコンテキストを把握しやすくなるため、レイアウトを統一しておく方がベターだと思います。

9.3.2 Sphinx入門

Sphinxのドキュメント生成方法の基本的なステップは、以下のとおりです。

1. コメントを記載する
2. rstファイルを生成する
3. ドキュメントを生成する

ここからは、実際にSphinxでドキュメントを生成する流れをご紹介します。

Sphinxはまずコメントを書きます。今回はGoogleのDocstring形式⁶で記述します。このほか、numpy形式などもあるので、好みのスタイルを使いましょう。

<test.py(Google Docstring形式コメント入り)>

```
class test:  
    """  
    testメソッド  
    """  
    def __init__(self, n):  
        self.number = n  
  
    def add(self, a):  
        """  
        内部変数nにaを足す関数  
  
        Args:  
            a(int): 内部変数nにaを足す
```

6.<https://google.github.io/styleguide/pyguide.html> 3.8.2 Modules

```

Returns:
    int : 足し算後の内部変数
    """
self.number += a
return self.number

def out(self):
    """
内部変数を出力するだけの関数

Returns:
    None
    """
print(self.number)

```

次に、コメントからreStructuredText（rstファイル）を作成後、ドキュメントを生成します。環境依存のあるconfigの記述など詳細部分は省きましたが、ほぼこれだけで以下のようなドキュメントが生成されます。

図9.3: ドキュメント例

test 0.1 documentation » test module

test module

`class test.test(n)`
Bases: `object`

`testメソッド`

`add(a)`
内部変数nにaを足す関数
Parameters: a (`int`) – 内部変数nにaを足す
Returns: 足し算後の内部変数
Return type: `int`

`out()`
内部変数を出力するだけの関数
Returns: `None`

test 0.1 documentation » test module

previous | modules | index

© Copyright 2023, Nishino. Created using Sphinx 5.0.0.

また、Sphinxではhtmlテーマがいくつもあるため、自分好みに様々なカスタマイズを行うことができます。以下でいくつかのテーマを紹介いたします（個人的にはOpen3Dでも使用されているサードパーティー製のRead the Docsが好みです）。

図 9.4: classic



図 9.5: Alabaster

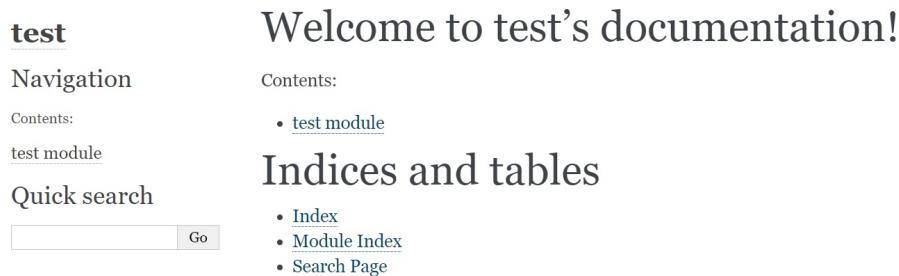
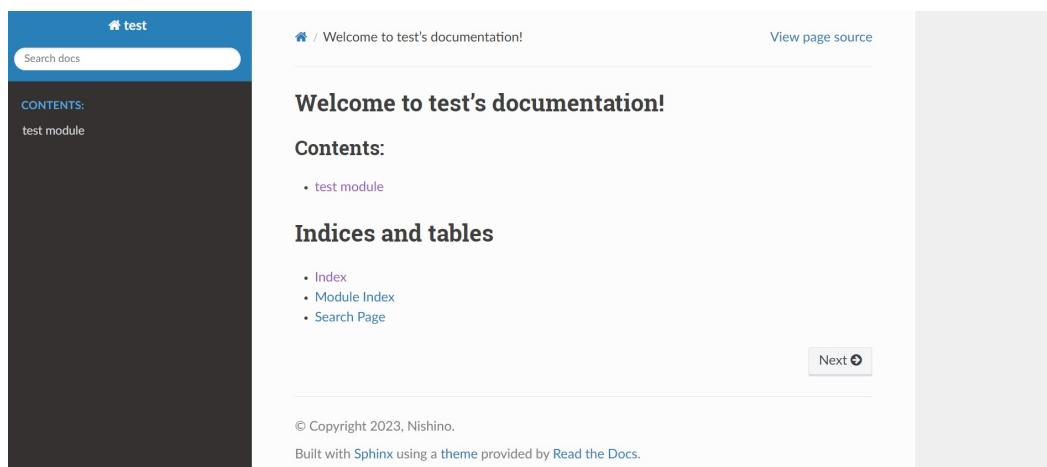


図 9.6: Read the Docs



9.3.3 型アノテーションのすすめ

さて少し Sphinx とは離れますぐ、コードに仕様を記載するという点で、型アノテーションについてご紹介したいと思います。型アノテーションは Python3.5 から追加された機能で、Python の引数の型と返り値の型を記述する方法です。あくまでもアノテーションのため、Python 自体の挙動には

関係がありません。以下にアノテーションの例を示します。

たとえば、add メソッドには引数の隣に int 型と記載しており、返り値の型は”->”で記述されています。詳しくは Python.org のドキュメントを参照ください。⁷

```
< test.py >
```

```
class test:
    def __init__(self, n:int):
        self.number = n

    def add(self, a:int) -> int:
        self.number += a
        return self.number

    def out(self) -> None:
        print(self.number)
```

型アノテーションは Python で定義されているため、通常のコメントとは異なり、IDEなどを使うと自動で読み取ってくれる場合もあります。たとえば Visual Studio Code で add 関数を呼び出すと、以下のように型ヒントを自動で示してくれます。

図 9.7: Visual Studio Code による型ヒント

```
1  from test import test
2
3  def main():
4      n=10
5      test_instance = te← / ←
6      test_instance.out((a: int) -> int
7      test_instance.add()
8      test_instance.out()
9  if __name__ == '__main__':
10     main()
```

その他サードパーティーライブラリーの mypy⁸などを使用すると、型ヒントから型チェックを行うことができます。たとえば以下のように、int で定義した変数に対して文字列を代入するようなコードを書いたとします。ここで実行すると、後から代入した文字列が表示されます。

しかし、mypy を実行すると、以下のようにエラーが出力されます。エラー文を読むと、2 行目で定義されている型と一致しないということのようです。C 言語等のコンパイルチェックのようですね。これにより事前の型チェックを行うことができるようになり、型誤りによるバグの混入を未然

7. <https://docs.python.org/ja/3/library/typing.html>

8. <https://github.com/python/mypy>

に防ぐことができます。

<mypy_usecase.py>

```
def main():
    a: int = 3
    a = "bug"
    print(a)

if __name__ == '__main__':
    main()
```

<実行結果>

```
bug
```

<mypy 実行結果>

```
(huga)hoge@hogehoge $ mypy mypy_usecase.py
mypy_usecase.py:2: error: Incompatible types in assignment (expression has type
"str", variable has type "int")  [assignment]
Found 1 error in 1 file (checked 1 source file)
```

9.3.4 現場での活用方法

現場で使用する場合には、生成されたドキュメント群を共有ディレクトリーに置いています。エクスプローラーでパスがたどれるのであれば、ドキュメント用に新規でサーバーを建てずともブラウザから確認することが可能です。

9.4 最後の救いlog取得

9.4.1 はじめに

みなさんはlogを残しているでしょうか？「9.3.1 はじめに」で述べたように、私はなるべく多くの方に使ってもらうことでツールは真価を発揮し続けるという考えをもっています。沢山の方が使えば使うほど想定外の問題が起きる可能性は上がります。仕様による動作から想定しないバグまでできることであれば全て網羅したいところですが、ツール作成時には必要以上の仕様を入れ込むことはコストパフォーマンスの低下を招きます。また、必要でない機能は仕様を決めるときの解像度が低いため、いい仕様にすることは難しいものです。

そこで、ツール使用者に適切な情報を与えるためにlog機能が必要になってきます。本節では、ツールの使用者に適切な情報を与えるためのlogについてお話ししたいと思います。

9.4.2 logを設定しよう

まず、前提としてどのようなlogが必要となるでしょうか？これは目的によって様々となります。たとえばレポートログを作成するツールであれば、形式は人が読みやすい書式がいいでしょう。今回はツールのデバッグ用のログとして考えてみましょう。

ログにはロギングレベルというものがあります。たとえばPythonのloggingモジュールでは以下の6つが定義されています。⁹

- CRITICAL
- ERROR
- WARNING
- INFO
- DEBUG
- NOTSET

なぜこんなにも分かれているかというと、常に出してほしい情報と必要なときに出してほしい情報のレベルは異なるためです。

たとえば、セキュリティーソフトをイメージしてみましょう。デバッグ用に必要な情報はなるべく多く出したいのですが、常に今どのファイルを検査していて、どのようなアップデートファイルをインストールしていて、バックグラウンドでどのシステムが動いているといった通知が常に出ていると、必要な情報が埋もれてしまい逆にわかりづらくなってしまいます。

一方で、攻撃を受けた場合は即座に通知をしてほしいものです。このように、その必要性に応じて適切にロギングレベルを適切に管理する必要があります。これだけの種類が用意されているとはいえ、私が現場で使う際はCRITICAL、WARNING、DEBUG程度でしか使い分けをしていません。理由としては、Pythonで作成するツールの多くはそこまで規模が大きくならないためです。粒度は必要に応じて定めてよいと思います。

Pythonであれば、loggingモジュールで必要なレベルを設定しておけば、import先でloggingモジュールを使っていればlogに情報が吐き出されます。適切に設定しておくことで、ユーザーフレンドリーなツールとなるでしょう。また、使用者側で確認ができるばツールから手を離すことができます。現場ではツール開発者がメンテナンスを行うことが多いと思います。ログを残してなるべく使用者側の手で解決してもらえるようにしておくといいと思います。

9.4.3 DEBUGはDEBUG

DEBUGログはバグ調査の最後の手段としてあるべきだと考えています。そのため、なるべくDEBUGでは様々な情報があったほうがいいのですが、それでも必要な情報は精査すべきです。

これは、私自身の体験なのですが、何度も呼び出されるモジュールに対して細かくDEBUGを吐き出すように設定していました。何度かテストを行なってログを確認しようとすると、テキストエディターでは開けなくなってしまいました。これは、あまりにもログを細かく設定しそうたために、

⁹<https://docs.python.org/ja/3/library/logging.html?highlight=log>

一度の呼び出しで沢山のログが吐き出され（一度で約500MB程度）いつの間にかログが肥大化していました。DEBUGの設定は、デバッグのために必要な情報を精査すべきでした。

この例は、あくまでも私の場合であり状況に応じてログを残すべきですが、サーバー上に置くものであればログがディスクの圧迫を招く可能性があるということも考慮に入れるべきでしょう。また、事前にログの量をある程度見積もることも重要なと思います。

9.5 さいごに

本章の裏テーマは「開発者の負担を減らす」というものでした。Pythonでは様々なモジュールがでており、また書籍やテックブログも潤沢であることで、ツールを開発しやすくなっていると思います。

一方で、現場ではその多くが開発者自身で運用やメンテナンスにも時間を割く必要があり、ツールを開発すればするだけメンテナンスコストが高まるという負のサイクルに課題を感じています。そのため、本章の2/3がツール利用者に自力で解決してもらうための導線を引くためのTipsとなっています。ご紹介した内容が少しでも有益なものとなれば幸いです。本章をお読みいただきありがとうございました。

第10章 GASとAWSを使ってWeb操作をハックしてみた!

大野 泰歩

10.1 はじめに

はじめて！筆者は普段通信事業者として商用設備の運用保守業務を担当しております。運用保守業務を実施されたことのある方であれば経験があるかと思いますが、保守しているシステムの不具合や故障(これらを以下、インシデントと記載)の対処を日々実施していく必要があります。この業務を滞りなく遂行していくには、情報の扱いができる限り自動にしていくことが望ましいです。

私はGoogle Apps Script¹(GAS)やAmazon Web Services²(AWS)を用いてWebページの操作を自動化の部分を実施しましたが、こちらをご説明する前に、既存で作成されていた情報収集の自動化部分からも記載していこうかと思います。

GASやAWSなどの技術については独学で学んだモノになりますので、拙い部分も多々あるかと思いますが、あくまで自動化の一例として温かい目で拝読頂けると幸いです。

10.2 情報収集の自動化(GAS)

「はじめに」で少し触れたように、私は運用保守業務を実施するうえで様々なインシデントを管理しており、こちらを管理するのにあたりスプレッドシートを用いております。毎度手動で入力していくのは大変なため、Gmailで通知されるインシデントから必要な情報を抜き出し、スプレッドシートに転記されるようにGASを用いて自動化しています。

自動化の方法としては、通知されるインシデントメールのテンプレートを定義することで、GASを用いて必要項目を抜き出しへスプレッドシートに転記するというものになります。

メールの例文を2種類程紹介します。

- ・メールサンプル1

■ID
1111111
■ホストグループ名
service_ABC

1.<https://developers.google.com/apps-script?hl=ja>

2.https://docs.aws.amazon.com/ja_jp/

■ホスト名
hostname_ABC

■アラームテキスト
・Error message

- ・メールサンプル2

```
[Linuxtime] 1111111  
[Hostname] hostname_ABC  
[Serial] abc123
```

ご紹介したようにテンプレートとしてメール本文の内容を決めることができた場合は、GASを用いてメール本文から情報を抜き出し、スプレッドシートに転記することができます。

今回は正規表現を用いて情報を抜き出す際の例を記載いたします。

- ・メール本文の抜き出し(GAS)

```
var myThreads = GmailApp.search('メール件名', 0, 50);  
var myMsgs = GmailApp.getMessagesForThreads(myThreads);  
  
for ( var threadIndex = 0 ; threadIndex < myThreads.length ; threadIndex++ ) {  
    var mailBody = myMsgs[threadIndex][0].getPlainBody();
```

- ・メールサンプル1_抜き出し方法例(GAS)

```
var myRegexp = new RegExp('■ID' + '[\s\S]*?' + '■');  
if (mailBody.match(myRegexp)) {  
    var incidentID = mailBody.match(myRegexp)[0].split('\n')[1];  
}  
else {  
    continue;  
}
```

- ・メールサンプル2_抜き出し方法例(GAS)

```
var myRegexp = new RegExp('\\[Linuxtime\\] ' + '.*?' + '\r');  
if (mailBody.match(myRegexp)) {  
    var incidentID = mailBody.match(myRegexp)[0].replace('[Linuxtime]', '');  
}  
else {
```

```
    incidentID = "Linuxtime未記載";  
}
```

このようにメール本文から情報を変数として取得することができましたので、後はスプレッドシートの中で転記したい位置に取得した情報を記載するようにすれば、情報の取得について自動化することができました。

その他に補足情報が必要な場合は、他のスプレッドシートに記載している情報を同じようにGASを用いたり関数を使用することで、更に多くの情報を集めることもできます。

10.3 Web操作の自動化(AWS)

情報の取得について自動にすることができたので、集めた情報を元にどのようにWeb操作を自動にしていくかを記載いたします。

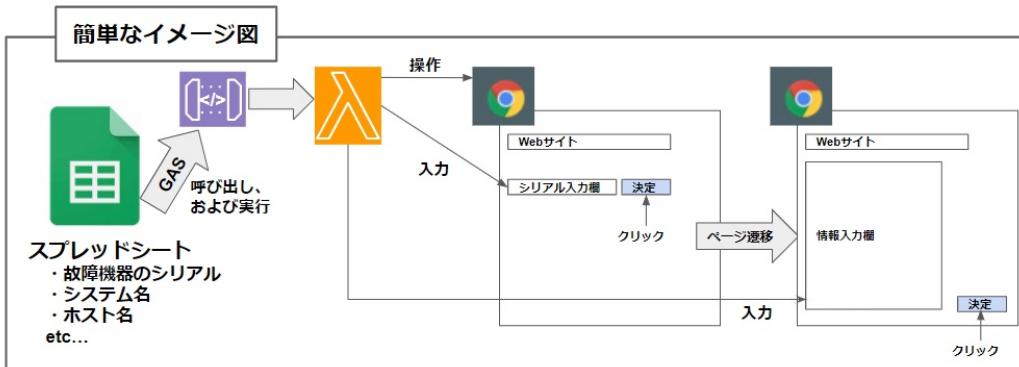
元々はGASを用いてWeb操作を実施しようと思っておりましたが、GASのみでWeb操作を可能とするドキュメントを見つけることができなかったため、AWSを用いて自動化していくことにしました。

また、この後説明していく処理の流れがイメージしやすいように、概要を添付します。

図 10.1: 処理概要

概要図

GASで必要な情報を「API Gateway」に渡し、
その後「Lambda」で処理の実行（pythonでselenium）



添付した「処理概要」の通りになりますが、下記の流れで処理を行っていきます。

1. スプレッドシートに情報収集
2. GAS から API Gateway にLambda の呼び出し処理
3. Lambda にて Web 画面の自動操作
4. 処理結果を GAS に返信

まずはLambdaでの処理を記載していきます。今回LambdaではPythonを使用していきます。Pythonを選択した理由としては、Web画面の操作を実施するのに有名なSeleniumというモジュールを使用するためとなります。

自動化するにあたって、このタイミングで1度躊躇しました。躊躇した理由としては、Lambdaの中には標準的な関数しか搭載されておらず、Seleniumなどの拡張モジュールを使用することができなかったからです。そのため、拡張モジュールを使用するためにLambdaの中のレイヤーという機能として使用したいモジュールの追加を行い、ツールを実行できるようにしました。

今回追加したモジュールは4つになります。

- ・Chromedriver: Seleniumの処理を行う土台のサイトとして使用するため
- ・headless-chromium: Web画面を表示しない状態で使用するため
- ・Selenium: Web画面操作の自動化を行うため
- ・headless-selenium: ヘッドレスモードでWeb画面操作を行うため

図10.2: レイヤー情報

レイヤー (4)					最終取得済 26 分前	C	レイヤーの作成
検索: レイヤーをフィルタリング					< 1 >	①	
名前	▼	バージョン	互換性のあるランタイム	互換性のあるアーキテクチャ			
chromedriver		2	python3.7	-			
headless		1	python3.7	-			
headless-chromium		2	python3.7	-			
selenium		2	python3.7	-			

ここまで準備できたら、後はpythonを用いてSeleniumの処理を記載することでWeb画面の操作ができるようになります。

- ・import内容(Lambda)

```
import json
import time
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.select import Select
```

- ・API Gatewayと連携するための関数(Lambda)

```
def lambda_handler(event, context):
    # TODO implement
    body = json.loads(event['body'])
```

```
sample_value = sample_function(body)

return {
    'statusCode': 200,
    'body': json.dumps(sample_value)
}
```

- ・Chrome ドライバの設定(Lambda)

```
def headless_chrome():
    ## Seleniumが使用するWebDriver作成時のオプションを作る
    options = webdriver.ChromeOptions()
    ## オプションのバイナリロケーションにLayerで用意したheadless-chromiumのパスを指定
    options.binary_location = '/opt/headless/bin/headless-chromium'
    ## オプションにヘッドレスモードで実行させる記述を書く
    options.add_argument("--headless")
    options.add_argument("--no-sandbox")
    options.add_argument("--single-process")
    options.add_argument("--disable-gpu")
    options.add_argument("--homedir=/tmp")

    driver = webdriver.Chrome(
        ## Layerで用意したchromedriverのパスを指定
        executable_path="/opt/headless/bin/chromedriver",
        options = options
    )
    return driver
```

- ・Selenium 使用例(Lambda)

```
def sample_function(body):
    driver = headless_chrome()
    driver.delete_all_cookies()
    driver.get('https://www.google.co.jp/')
    search_box = driver.find_element_by_xpath('//*[@id="APjFqb"]')
    search_box.send_keys(body['sample'])
```

この4つの内容を記載することで、プログラムの基本的な概要は完成しております。後は、実際に使用する場合に合わせて必要なオプションを追加したり、Selenium の処理を追記していくことで実行したい処理を完成させていくことができます。

Seleniumの使い方や関数の詳細については色々と書籍やWebページでの説明があるので、今回の自動化をする際によく使用していた関数を代表として紹介します。

- find_element_by_xpath: Webページのxpathを指定して要素を取得する
- send_keys: テキストボックスなどに文字列を入力する
- select_by_value: ドロップダウンなどのセレクト要素を選択する
- click: 選択している要素をクリックする

これにて、Web画面の操作を自動にする処理をLambdaを用いて作成することができました。そのため、後はAPI Gatewayを作成し、今回作ったLambdaと連携させることでGASから呼び出せるようになります。

10.4 GASからLambdaの呼び出し(GAS)

GASからLambdaに連携するにあたって、API Gatewayで生成した連携用のURLと収集した情報をjsonの形で整えて使用する必要があります。その整えた情報をUrlFetchApp.fetchを使用することで、Lambdaに渡すことができます。

- GASからLambdaへの連携(GAS)

```
const aws_url = 'API GatewayのURL';
const params = {
  'method' : 'post', //get or post
  'contentType': 'application/json',
  'payload' : JSON.stringify(
    {"sample_value1" : sample_value1,
     "sample_value2" : sample_value2,
     "sample_value3" : sample_value3})
};

const req = UrlFetchApp.fetch(aws_url, params);
```

これにてGASとLambdaの連携が完了しましたので、後は処理が完了した旨のポップアップを表示させたりメールを送るなど、お好みで設定する形となります。

10.5 さいごに

本章においては、定常的な業務の効率化を行うための方法について記載させて頂きました。AWSの使い方やGASの使用方法について他にも最適な方法はあるかと思いますが、この章をお読みいただいたの方の業務が少しでも楽になったり、効率化への閃きに寄与することがあれば幸いです。最後になりますが、本章をお読みいただきましてありがとうございました。

第11章 フローで思い通りの結果を得るために大切なこと

高井 美佑

11.1 はじめに

みなさま、ごきげんよう。突然ですが、Power Automateでフローを組んで、テストして、無事に成功したとき、安心してそのままウィンドウを閉じていませんか？

「フローが成功しました」という表示は「エラーなしで正しく実行されました」という意味で、「作成者であるあなたの想定通りに実行されました」という意味ではありませんよ？

さて、あなたが今閉じてしまったそのフロー、本当に「想定通り」の動きをしているでしょうか？不安になっていませんか？本章では、フローが「想定通り」に動いているかどうかの確認方法を実際のユースケースを見ながら、解説していきます。ぜひ最後まで一読頂き、想定通りの処理が行われているかどうかの確認方法を身に着けて頂きたいと思います。フローの処理に不備があったがために、偉い人に「手作業憐みの令」を出されないように。

11.2 テストは、2段階に分けられる

ITを本職とされている人よりも知識が薄い市民開発者が忘れがちなのは、テストは次の2段階に分けられるということです。

1. テスト機能を使用したフローの動作テスト
2. 作成者が期待する「正常な動作」が行われているかの確認

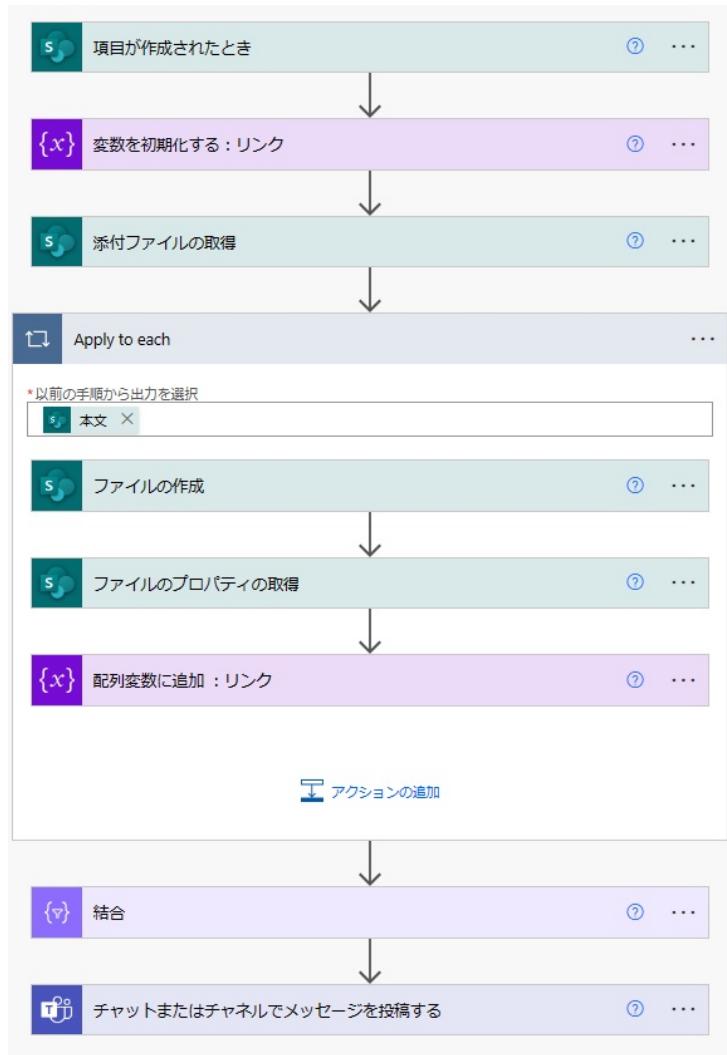
1.は、ほとんどの人が当たり前のことを行っていることだと思います。問題は2。作成者が期待する「正常な動作」が行われているかの確認です。これは、フローのテスト結果の中身を確認したり、フローの外に出力された結果（EX：フローで投稿したTeamsのメッセージ）を確認したりすることが必要です。

フローが成功していても、作成者が期待する「正常な動作」をしていないこともあります。市民開発者は、Power Platformでの開発が初めての開発経験という方も多く、1.のテスト機能を使用したフローの動作テストが成功すると安心してテストをやめてしまうこともあるのです（かつての私がそうでした）。そのため、実際のユースケースを通して、2.の確認方法を学んでいきたいと思います。

11.3 Case アップロードしたはずのファイルが開かない

11.3.1 どんなフローか

図 11.1: どんなフローか



このフローは、指定された SharePoint リストにレコードが作成されたときにそのレコードについている添付ファイルを取得し、その添付ファイルを指定されたドキュメントライブラリーにアップロードし、Teams にドキュメントライブラリーへのリンク付きメッセージを投稿するという動作をします。

11.3.2 このフローで期待される正常な動作はどういうものか

- 新しいレコードが作成されたときにフローが実行されること

図11.2: 新しいレコードが作成されたときにフローが実行される



- Teamsにファイルのリンク付きメッセージが投稿されること

図11.3: Teamsにファイルのリンク付きメッセージが投稿



- メッセージのリンクをクリックしたら、ファイルが表示されること
- 以上3点が、作成者である私が期待する正常な動作です。

11.3.3 テスト機能を使って、フローの動作テストを行う

フローを作成したら、まずはテスト機能を使って、動作テストを行います。動作テストは、画面右上の「テスト」ボタンをクリックすると行うことができます。

図 11.4: テスト機能 1

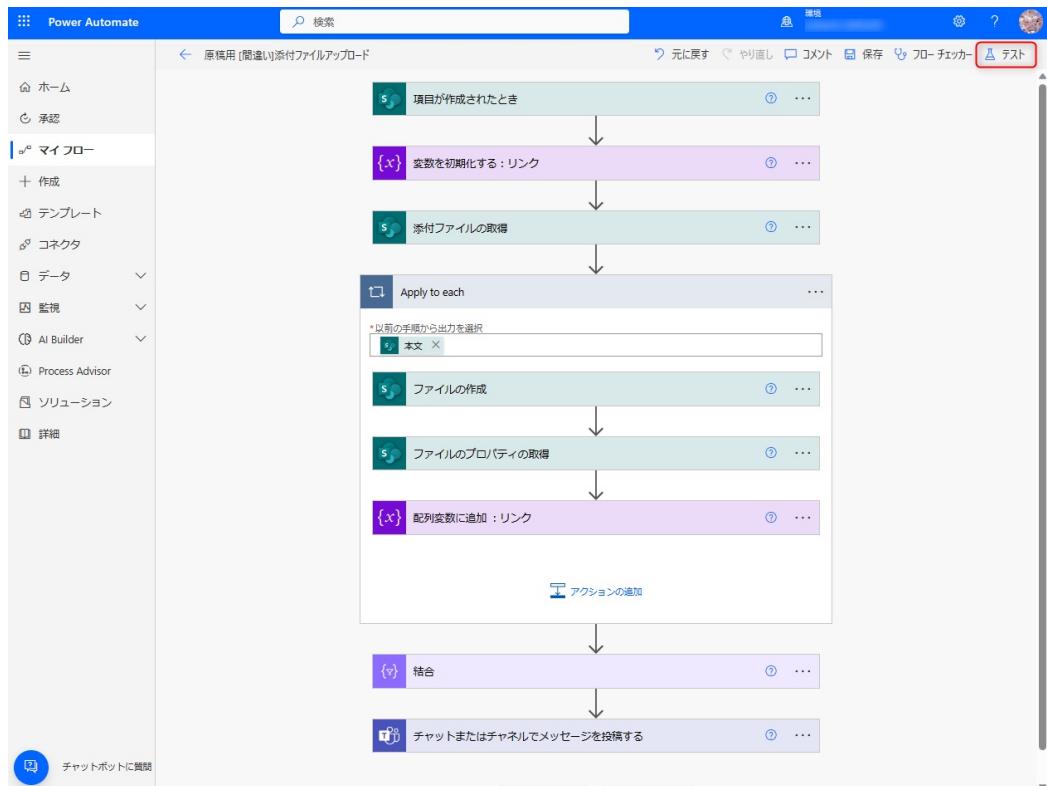
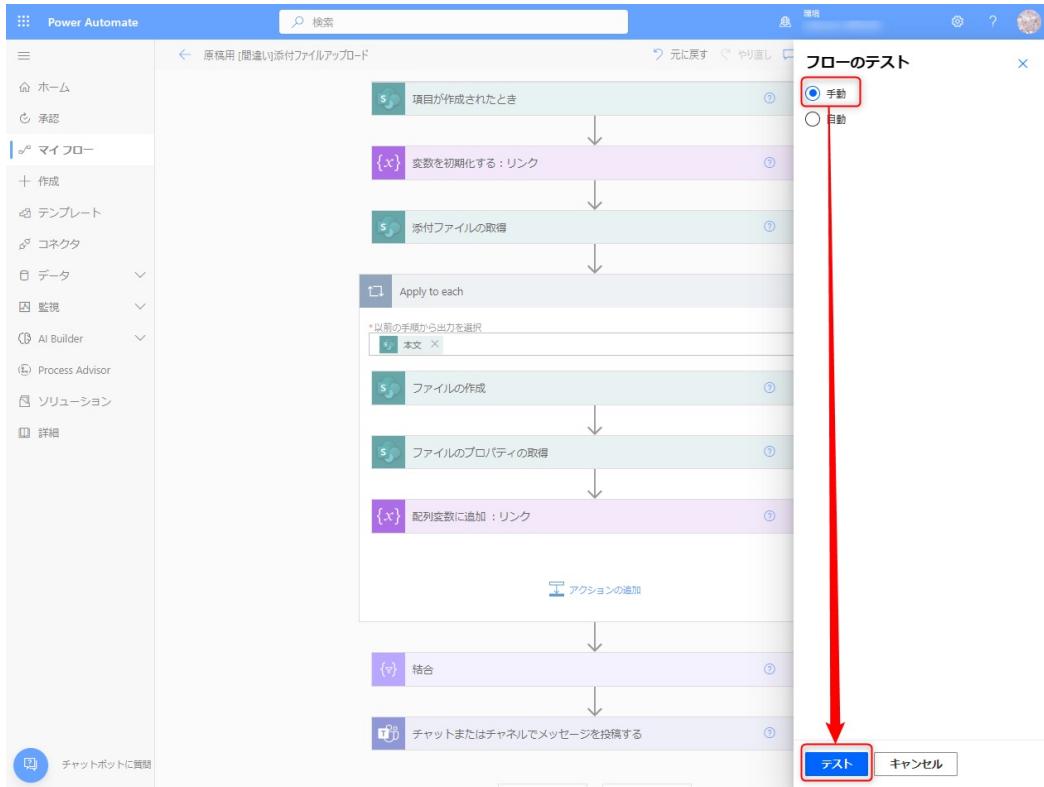
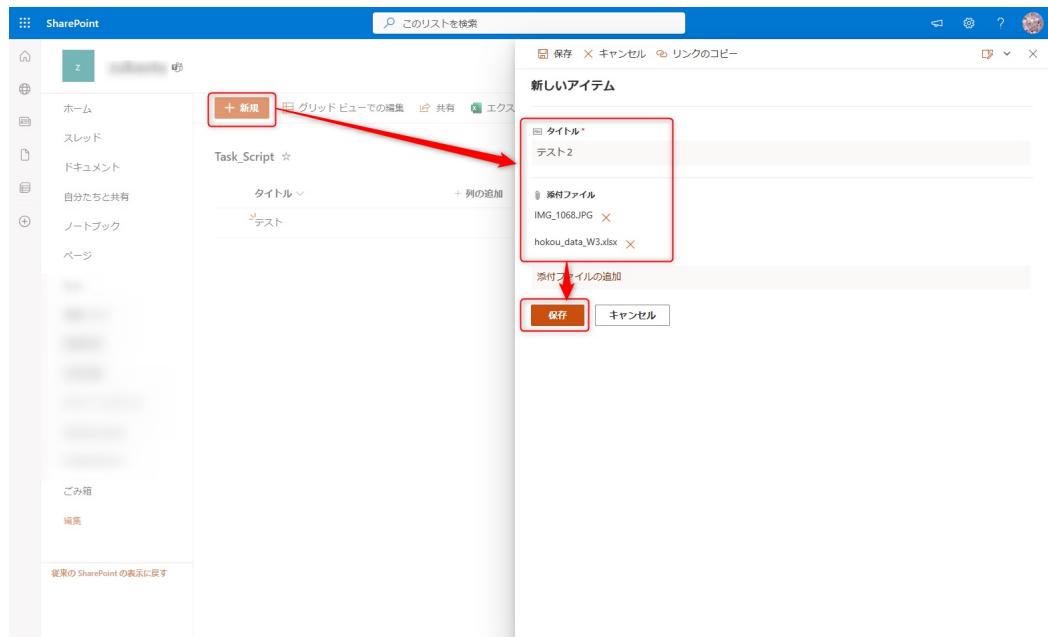


図11.5: テスト機能2



最初のテストは「手動」から行います。今回のトリガーは、SharePointリストのレコードの新規作成なので、テストボタンを押した後にレコードの新規作成を行ってください。

図 11.6: SharePoint のレコードの新規作成



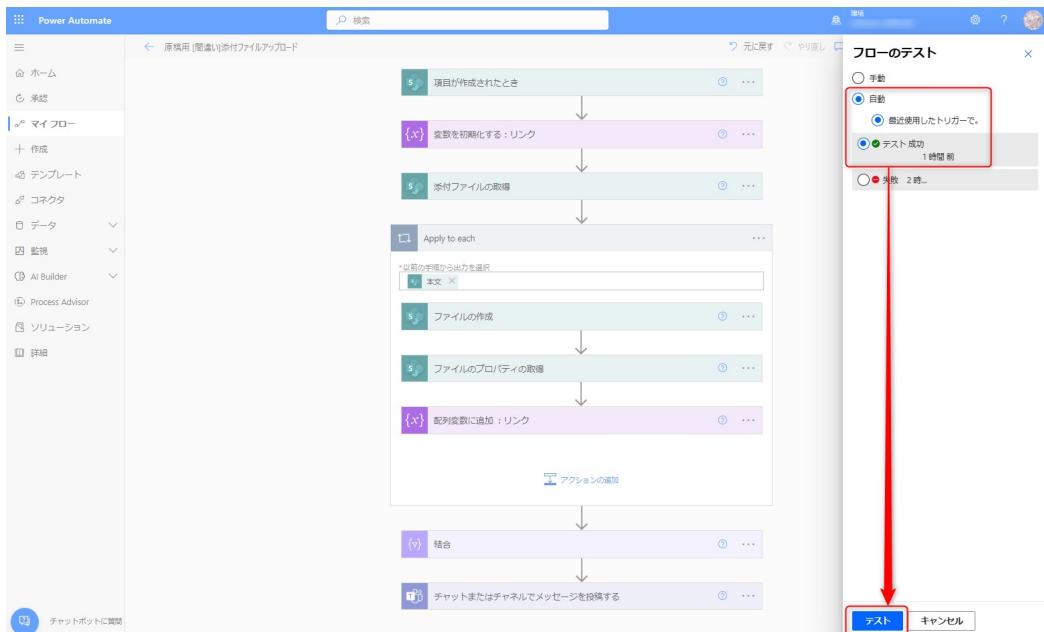
そうすることでフローが実行されます。テストが成功すると、緑色で「ご利用のフローが正常に実行されました」という通知が出てきます。

図 11.7: フローの実行



ちなみに2回目以降は、1回目のデータを利用して、「自動」からテストを行うことも可能です。

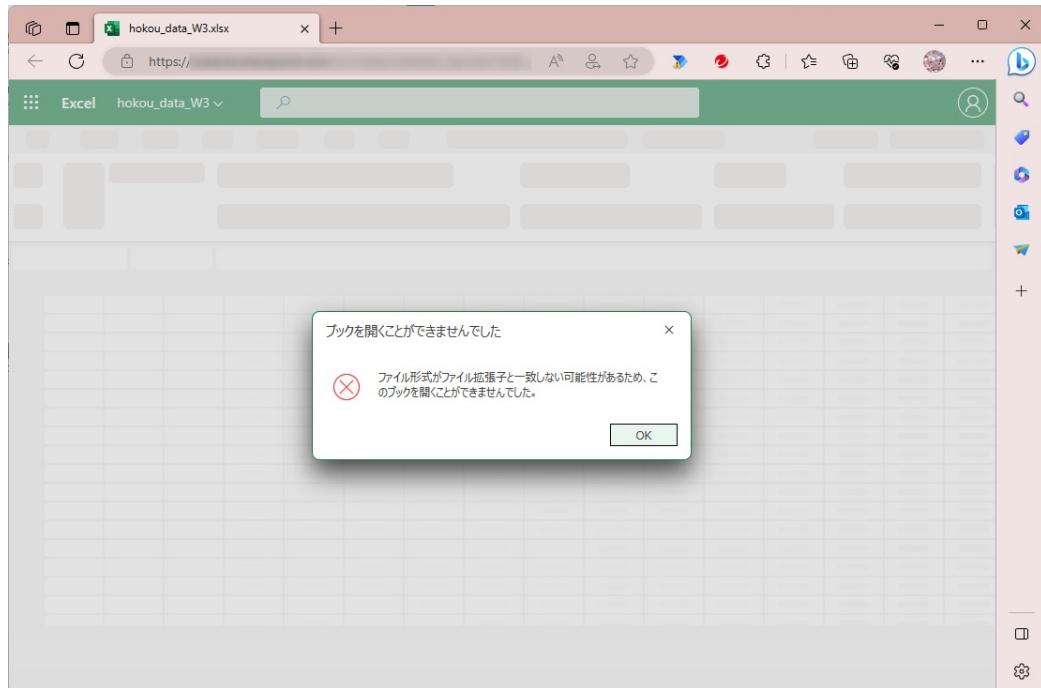
図 11.8: 自動からテストを実行



11.3.4 出力結果が正常な動作をするか確認する

フローが成功し、無事Teamsにメッセージが投稿されたので、リンクをクリックして、ファイルを確認してみます。

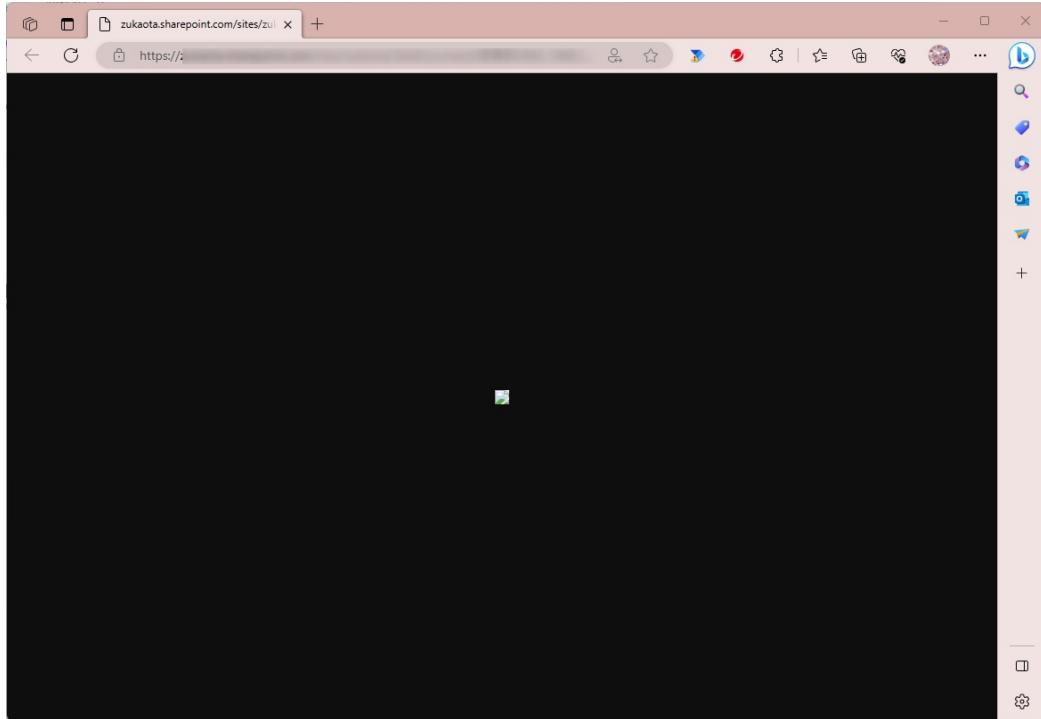
図 11.9: 出力結果



すると、なんということでしょう！！

「ブックを開くことができませんでした」というメッセージが表示されています。これはExcel固有の現象なのか、それとも画像など他のファイルでも同じようになるのか調べるために、画像のリンクもクリックしてみます。

図 11.10: 画像の表示



すると、画像も開くことができませんでした。ものの見事に真っ暗です。つまり、「なんらかの事情でメッセージにあるリンクが開けない」というバグが見つかりました。

11.3.5 「心当たり」をすべて確認し、問題を切り分ける

バグの修正は、無くしたものを探すことと似ている気がします。たとえば財布を落としたとき、どうするでしょう？最後にどこで使ったかを考え、最後に使った場所から現在地までどのように行動したかを思い出して、財布を落としたかも？と心当たりのある場所を1か所ずつ訪れて、探していませんか？バグの修正も同じです。「心当たり」を1か所ずつ確認していきます。今回のケースだと私は、ふたつの「心当たり」があります。

- ・実はフローが失敗している（テスト結果は見間違いだった）
- ・添付ファイルがドキュメントライブラリーにアップロードされる過程で壊れた

11.3.6 実行履歴を確認してみる

フローのテストを実行し、成功通知を確認しているとはいえ、バグが出ている以上は本当に成功しているのかの確認は必要です。実行履歴を確認してみましょう。フローの編集画面左上の「←」ボタンをクリックして、ひとつ前の画面に戻ります。

図 11.11: ひとつ前の画面に戻る



「28日間の実行履歴」から先ほどのテストの日時をクリックしましょう。

図 11.12: 28日間の実行履歴

開始	時間	状況
6月18日 15:21 (10分前)	00:00:02	テストに成功しました
6月18日 12:58 (2時間前)	00:00:02	テストに成功しました
6月18日 12:58 (2時間前)	303 ミリ秒	失敗

すべてのアクションに緑のチェックマークがついていて、画面上部に成功通知も出ています。間違いなく成功していることが確認できました。

図11.13: 正常に終了



11.3.7 ファイルサイズを確認してみる

フローが正しく動作していることがわかりました。それならば、なんらかの原因でファイルが壊れている説が有力になってきますね。添付ファイルと同じものがドキュメントライブラリーに正しく保存されていれば、ファイルの大きさも全く同じなはず……。なので、元データとドキュメントライブラリー保存のファイルサイズを比較してみることにします。

図 11.14: 元データ

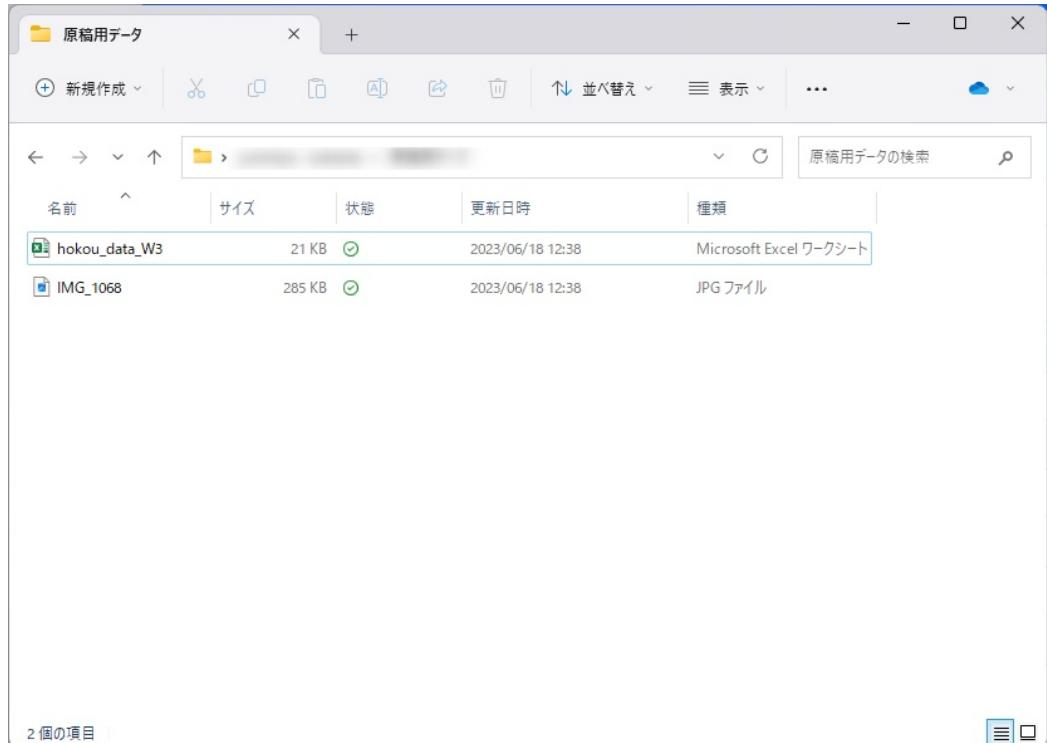


図 11.15: ドキュメントライブラリーのデータ

TaskDocument > 営業部

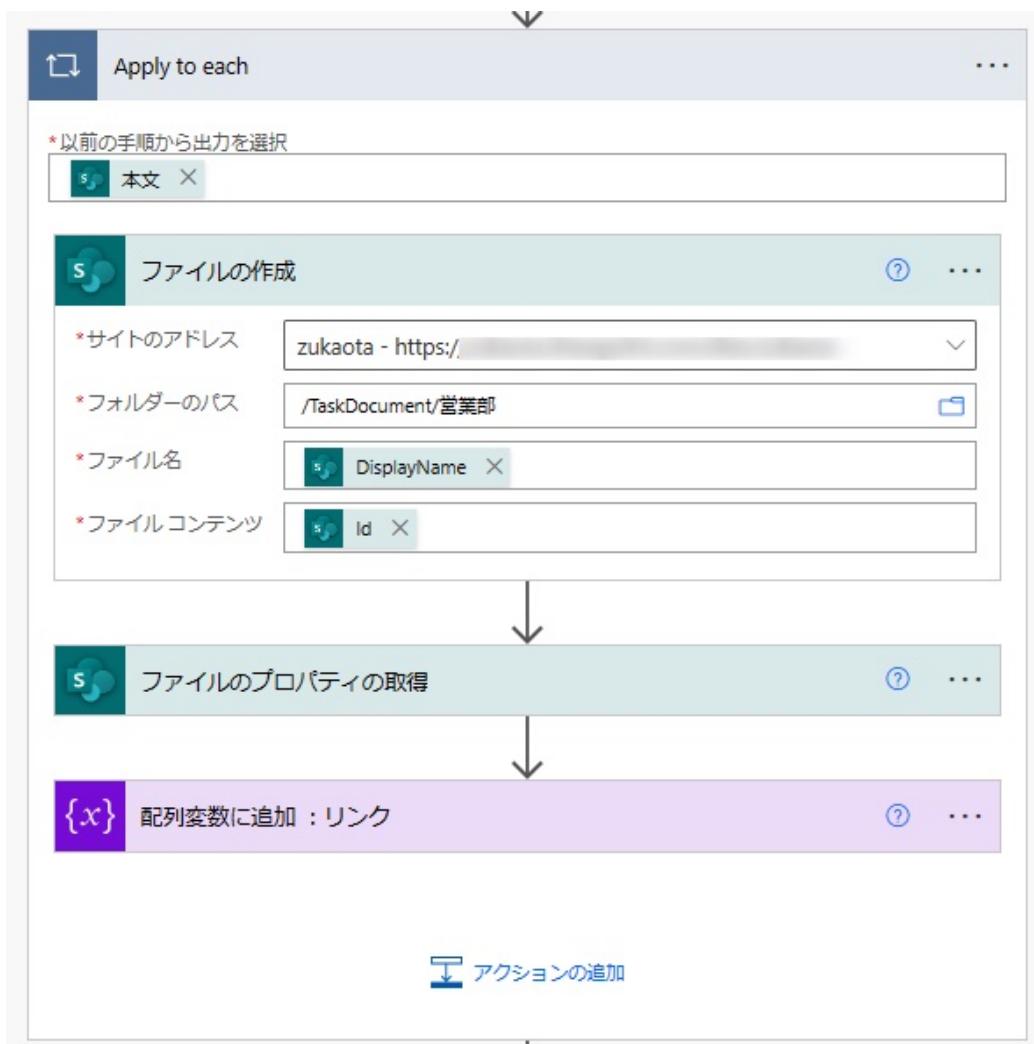
名前	ファイル サイズ
hokou_data_W3.xlsx	71 バイト
IMG_1068.JPG	65 バイト

比較してみると、明らかにサイズが異なります。元データは単位が「KB (キロバイト)」なのに対し、ドキュメントライブラリー側は「B (バイト)」です。元データよりもファイルが小さすぎることがわかりました。ここで思い出してほしいのは、ドキュメントライブラリーのファイルはPower Automateのフローで作成しているということです。つまり、フローの動作テストは成功しているが、フローに何か不具合があるためにファイルが正しく作成されておらず、ファイルサイズが元データと異なる、という状態であることがわかります。

11.3.8 ファイルの作成アクションを確認してみる

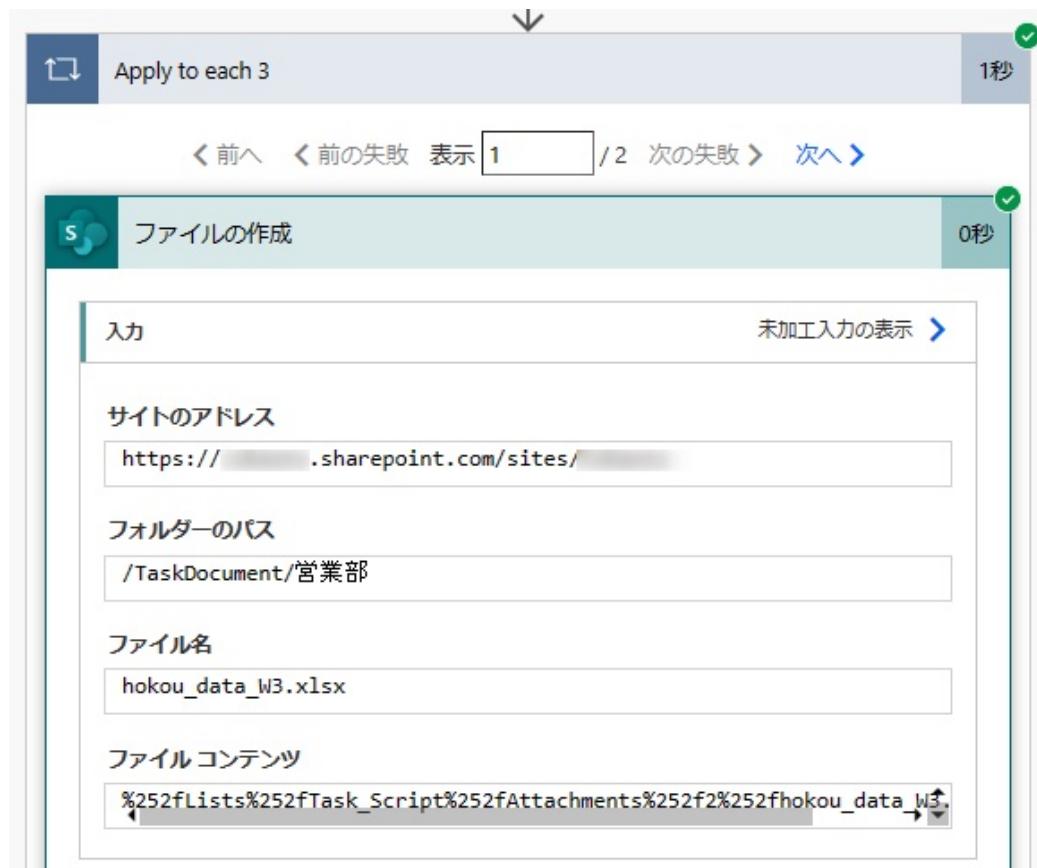
フローの中でドキュメントライブラリーに保存するファイルを作成しているのは、「ファイルの作成」アクションです。このアクションの設定が正しいかを確認していきます。編集画面から見ると、ファイルの作成アクションの設定は以下のようになっています。

図11.16: ファイルの作成1



実行履歴からテスト時に入力された値を確認していきます。

図11.17: ファイルの作成2



サイトのアドレス、フォルダーのパス、ファイル名は想定通りの値が入力されていることが確認できます。この3つは問題ないでしょう。でもファイルコンテンツは、不思議な並びの文字列で、この値が正しいものかどうか、現時点ではわかりません。改めて、ファイルの作成アクションとは何かを調べてみる必要がありそうです。

言葉の意味を調べるときは、公式リファレンスをあたるのが1番です。今回の場合は、Power Automateのフローなので、Microsoft Learn¹を読み解いていきます。

1.<https://learn.microsoft.com/ja-jp/connectors/sharepointonline/>

図11.18: ファイルの作成 (出典:Microsoft Learn)

ファイルの作成

操作 ID: CreateFile

SharePoint サイトにファイルをアップロードします。必ず既存のライブラリを選択してください。

パラメーター

Name	キー	必須	型	説明
サイトのアドレス	dataset	True	string	例: https://contoso.sharepoint.com/sites/sitename 。
フォルダのパス	folderPath	True	string	既存のライブラリから開始する必要があります。必要に応じてフォルダを追加します。
ファイル名	name	True	string	ファイルの名前。
ファイルコンテンツ	body	True	binary	ファイルのコンテンツ。

Microsoft Learn には、ファイルのコンテンツ欄には「ファイルのコンテンツ」を入れると説明されています。では、先ほど「ファイルのコンテンツ」欄に入っていた「Id」はファイルのコンテンツなのでしょうか?

図 11.19: ファイルのコンテンツ欄

The screenshot shows a search interface titled '動的な値' (Dynamic Value) with a sub-tab 'Expression'. A search bar contains the text '動的な値を検索'. Below it, a section titled '添付ファイルの取得' (Attachment File Get) is expanded. A red box highlights the first result, 'Id' (File Identifier), which is described as 'ファイル識別子' (File Identifier). Other results listed are 'AbsoluteUri' (添付ファイルへのリンク) and 'DisplayName' (名前).

名前	説明
Id	ファイル識別子
AbsoluteUri	添付ファイルへのリンク
DisplayName	名前

動的な値から改めて「Id」を確認したところ、「ファイルの識別子」であり、「ファイルのコンテンツ」ではないことがわかりました。つまり、ファイルの作成アクションの「ファイル コンテンツ」の設定を間違っていたため、ドキュメントライブラリーに正しくファイルが作成されていないということがわかりました。

11.3.9 ファイル コンテンツ は存在するのか？

ファイルの作成アクションよりも前のアクションで「ファイルのコンテンツ」が取得できていれば、ファイルの作成アクションの「ファイル コンテンツ」に「ファイルのコンテンツ」を入れることができます。では、今のフローで「ファイルのコンテンツ」を取得できているのでしょうか？動的な値を検索して、確認してみましょう。

図 11.20: 動的な値 1

The screenshot shows the 'Dynamic Value 1' configuration page. At the top, there are tabs for '動的な値' (Dynamic Value) and 'Expression'. On the right, there are three icons: a question mark, a smiley face, and a close button. Below these, a search bar contains the text 'コンテンツ'. A large section titled '項目が作成されたとき' (When item is created) is expanded, showing two items:

- Abc コンテンツの承認状態**
コンテンツの承認状態です。下書き、保留中、承認済み、または拒否された次のいずれか...
- Abc このリストアイテムのコンテンツの承認に関連付けられているコメント**
リストアイテムのモデレーションに関連付けられているコメント

図 11.21: 動的な値2

The screenshot shows the 'Dynamic Value 2' configuration interface. At the top, there is a search bar with the placeholder 'ファイル' (File) and a help/smileys/x icon area. Below the search bar, the results are displayed in two sections:

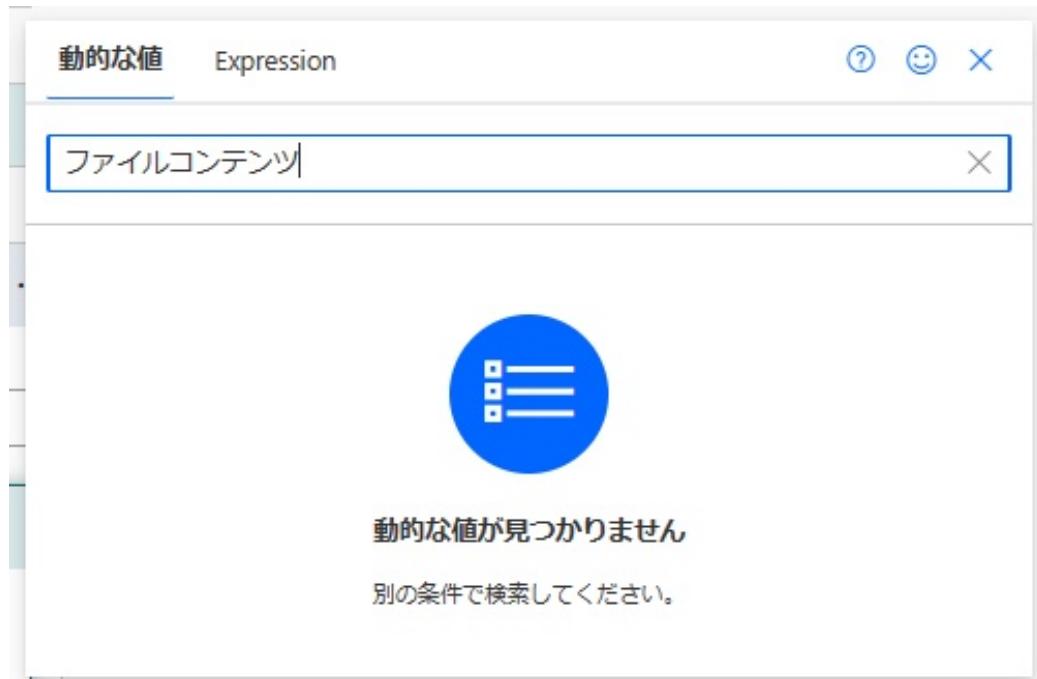
- 添付ファイルの取得** (Get Attached File):
 - Id**: ファイル識別子 (File Identifier)
 - AbsoluteUri**: 添付ファイルへのリンク (Link to attached file)
 - DisplayName**: 名前 (Name)
 - 項目** (Item):
 - SharePoint リストアイテムの添付ファイル** (Attached file of SharePoint list item)
 - body**- 項目が作成されたとき** (When item is created):
 - アイテムへのリンク**: ファイルまたはリストアイテムの取得に使用できるリンクです。そのアイテムへのアクションを実行するためのリンクです。
 - バージョン番号**: ファイルまたはリストアイテムのバージョン番号。

図 11.22: 動的な値 3

The screenshot shows the 'Dynamic Value' dialog box from SharePoint. At the top, it says '動的な値' (Dynamic Value) and 'Expression'. Below that is a search bar with the placeholder 'ファイル' (File). The search results are listed under 'SharePoint リストアイテムの添付ファイル' (SharePoint List Item Attached File). The results are as follows:

- body**
- 項目が作成されたとき**
- アイテムへのリンク**
ファイルまたはリストアイテムの取得に使用できるリンクです。そのアイテムへのアク...
- バージョン番号**
ファイルまたはリストアイテムのバージョン番号。
- 添付ファイルあり**
アイテムに添付ファイルが存在することを示します。
- 拡張子付きのファイル名**
ライブラリの場合は、拡張子を含むファイル名を返します。リストの場合は、タイトル...
- 識別子**
選択したファイルに対するファイル関連操作で使用できる値です。
- 完全パス**
アイテムまたはフォルダーまたはファイルの完全パス
- 名前**
ドキュメントライブラリ内のアイテムのファイル名、リスト内のアイテムの表示名です...

図 11.23: 動的な値 4



「ファイルの作成」アクションの「ファイル コンテンツ」欄にカーソルを当て、「コンテンツ」「ファイル」「ファイルコンテンツ」の3つで検索しましたが、「ファイルのコンテンツ」らしき動的な値は見つかりませんでした。つまり、新しくアクションを追加して、「ファイルのコンテンツ」を取得しなければならないということがわかりました。

11.3.10 どのアクションで「ファイルのコンテンツ」を取得するか

取得したいファイルのコンテンツは、レコードの添付ファイルのコンテンツです。ゆえにレコードの添付ファイルのコンテンツを取得するには、どうすればいいかを考える必要があります。わからないことがあるときは、公式リファレンスに立ち返るのが1番です。今回も Microsoft Learn を読み解いていきたいと思います。

SharePointコネクタのアクション一覧を見ていくと、「添付ファイルのコンテンツを取得」というアクションがあることがわかりました。

図 11.24: 添付ファイルのコンテンツを取得 (出典:Microsoft Learn)

新しいドキュメント セットを作成する	新しいドキュメント セットリスト品目を作成します。
新しいフォルダーの作成	新しいフォルダまたはフォルダ パスを作成します。
添付ファイルのコンテンツを取得	ファイル識別子を使用してファイルの内容を返します。 内容は別の場所にコピーすることも、添付ファイルとして使用することもできます。
添付ファイルの削除	指定された添付ファイルを削除します。
添付ファイルの追加	指定したリスト アイテムに新しい添付ファイルを追加します。
添付ファイルを取得する	指定されたリストアイテムの添付ファイルのリストを返します。「添付ファイルの内容の取得」の手順を追加し、このアクションによって返される「ファイル識別子」プロパティを使用して、ファイルの内容を取得できます。

これを使えば、添付ファイルのコンテンツが取得できそうです。Microsoft Learn のアクション名をクリックして、詳細ページに飛んでみましょう。

11.3.11 「添付ファイルのコンテンツを取得」アクションを理解する

初めて見るアクションは、Microsoft Learn で確認。ここまで読み進めた皆さんには、きっと覚えられたと思います。

図 11.25: 添付ファイルのコンテンツを取得 (出典:Microsoft Learn)

添付ファイルのコンテンツを取得

操作 ID: GetAttachmentContent

ファイル識別子を使用してファイルの内容を返します。 内容は別の場所にコピーすることも、添付ファイルとして使用することもできます。

パラメーター

Name	キー	必須	型	説明
サイトのアドレス	dataset	True	string	例: https://contoso.sharepoint.com/sites/sitename 。 ↗
リスト名	table	True	string	SharePoint リスト名。
ID	itemId	True	integer	ファイルが添付されたリスト ID。
ファイル識別子	attachmentId	True	string	添付ファイルのファイル識別子。

戻り値

添付ファイルのコンテンツ。

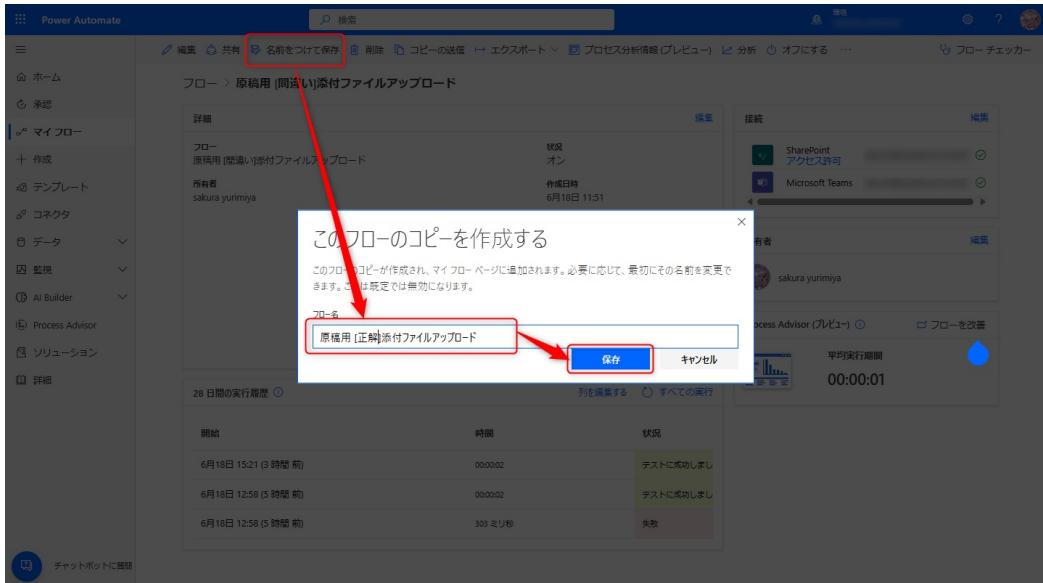
添付ファイルのコンテンツ binary

Microsoft Learn を読むと、ファイルが添付されたレコードの ID と添付ファイルのファイル識別子がわかれば、添付ファイルのコンテンツを取得できるということが理解できます。

11.3.12 フローをコピーし、「添付ファイルのコンテンツを取得」アクションを追加する

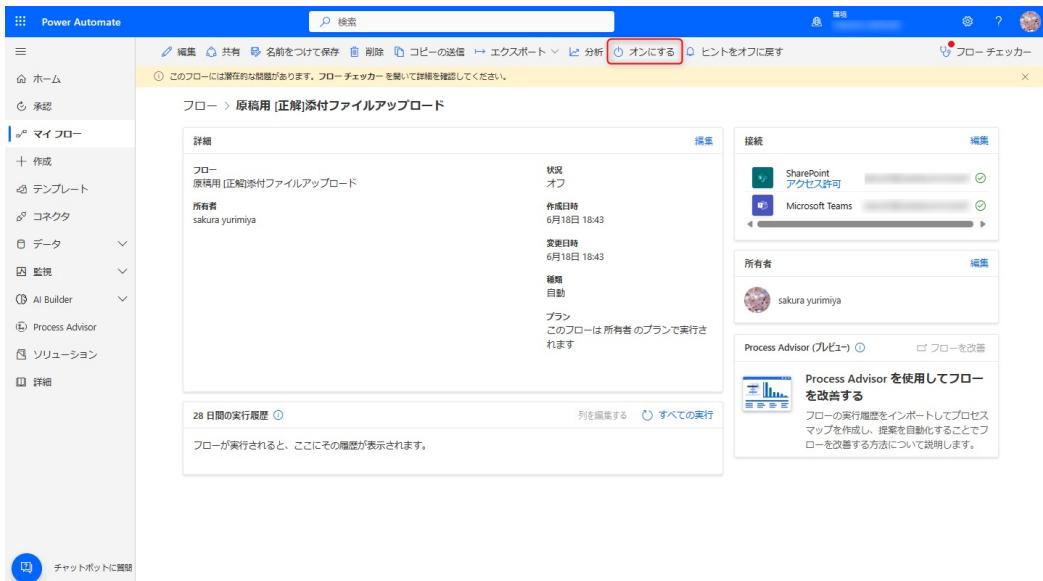
いよいよ「添付ファイルのコンテンツを取得」アクションを追加して、「ファイル コンテンツ」を取得したいところですが、まずはフローをコピーしておきましょう。フローをコピーすることで現在のフローをそのまま残すことが可能になり、差分比較が行いやすくなります。フローは「名前を付けて保存」でコピーできます。

図 11.26: フローのコピー 1



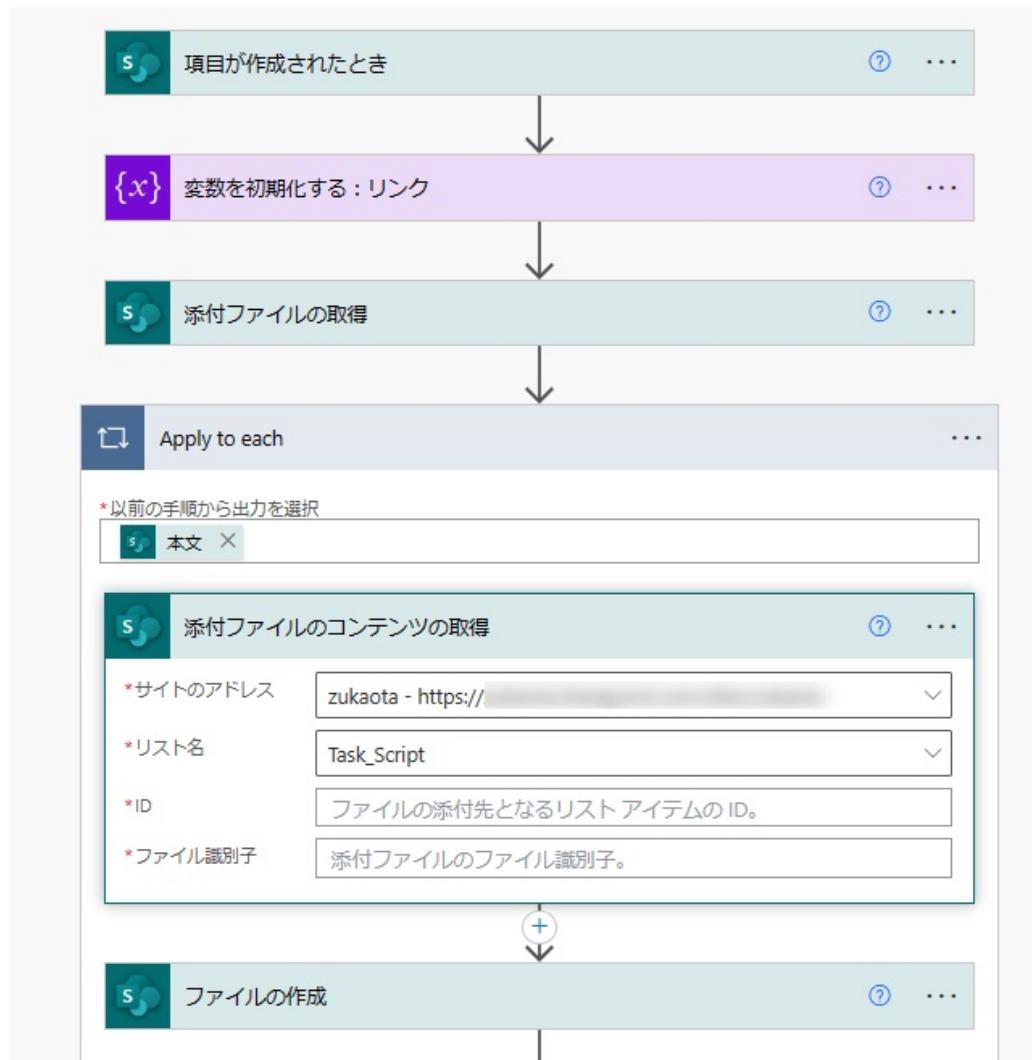
コピーしたフローはデフォルトでオフになっていますので、オンにしてから編集しましょう。

図 11.27: フローのコピー 2



「添付ファイルのコンテンツの取得」アクションは、添付ファイルをひとつずつ取得しなければならないので、Apply to each アクションの先頭に入れます。

図 11.28: 添付ファイルのコンテンツの取得



11.3.13 ID とファイル識別子が取得できるか確認する

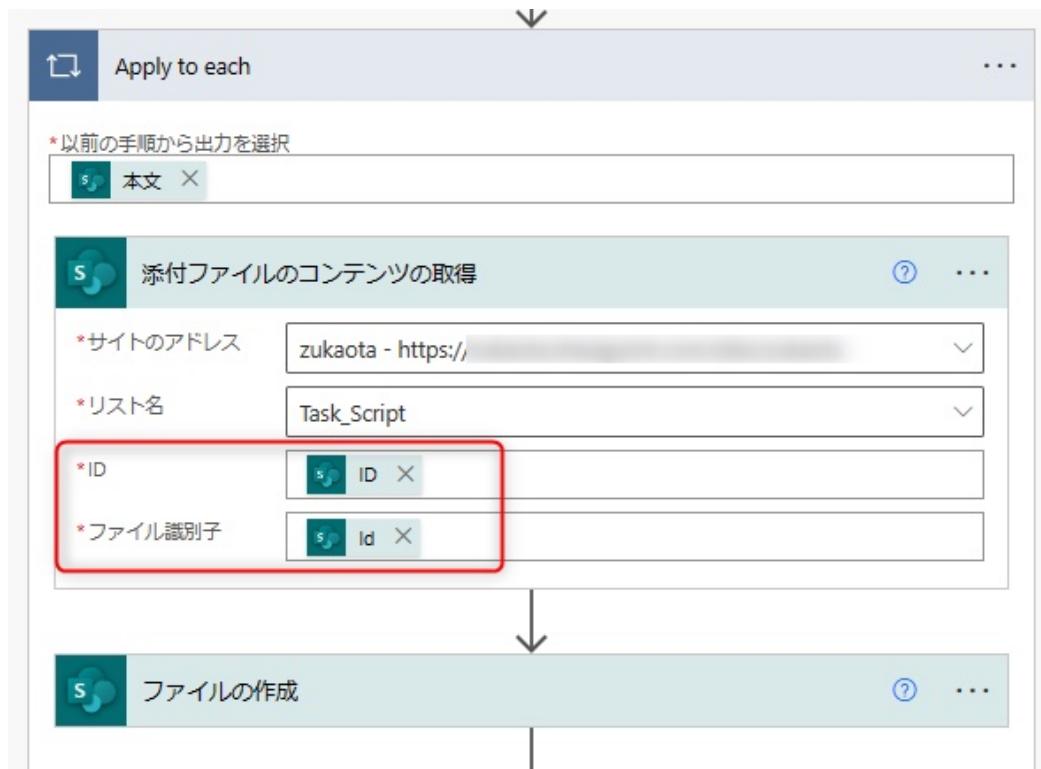
では、先ほどのファイルのコンテンツと同様に、ID とファイル識別子が現在のフローで取得できているのかを確認しましょう。動的な値で「id」を検索した結果は画像の通りです。

図 11.29: 動的な値



項目名の下の説明で、「項目が作成されたとき」の「ID」が「ID」に、「添付ファイルの取得」の「Id」が「ファイル識別子」になりそうだということがわかりました。実際にアクションに動的な値を入れ込んでみると、こんな感じです。

図 11.30: 添付ファイルのコンテンツの取得



11.3.14 「ファイルの作成」アクションの設定し直す

「添付ファイルのコンテンツの取得」アクションの設定が完了しました。これで「ファイルのコンテンツ」が取得できるようになるはずです。では改めて、「ファイルの作成」アクションの「ファイルコンテンツ」にカーソルをあてて、動的な値を検索してみましょう。

図11.31: 動的な値

The screenshot shows the 'Dynamic Value' Expression builder interface. At the top, there's a search bar with the placeholder 'コンテンツ'. Below it, a section titled '添付ファイルのコンテンツの取得' contains a single item: '添付ファイルのコンテンツ' with the description '添付ファイルのコンテンツ。'. This item is highlighted with a red border. Below this, another section titled '項目が作成されたとき' contains two items: 'コンテンツの承認状態' with the description 'コンテンツの承認状態です。下書き、保留中、承認済み、または拒否された次のいずれか...' and 'このリストアイテムのコンテンツの承認に関連付けられているコメント' with the description 'リストアイテムのモデレーションに関連付けられているコメント'.

無事に「添付ファイルのコンテンツの取得」の「添付ファイルのコンテンツ」が出てきました。こちらを「ファイル コンテンツ」に入れていきましょう。

図11.32: ファイルの作成

The screenshot shows the 'File Creation' dialog. It has four input fields: 'サイトのアドレス' containing 'zukaota - https://...', 'フォルダーのパス' containing '/TaskDocument/営業部', 'ファイル名' containing 'DisplayName', and 'ファイルコンテンツ' containing '添付ファイ...'. The 'DisplayName' field is highlighted with a red border.

これで「ファイルが破損している可能性」を潰したことになります。改めてテストを行っていきましょう。

11.3.15 フローをテストし直す

テスト機能を使用して、再びフローの動作テストを行います。最初のテストと同一の環境で行うために、1回目のテストでドキュメントライブラリーにアップロードしたファイルは、すべて削除

しておきましょう。

図 11.33: ドキュメントライブラリー

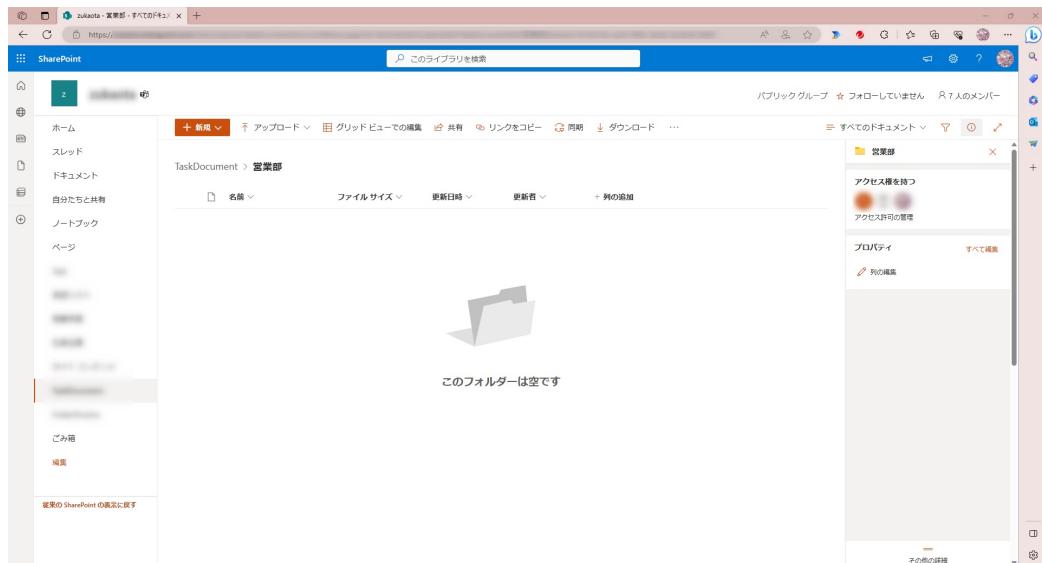
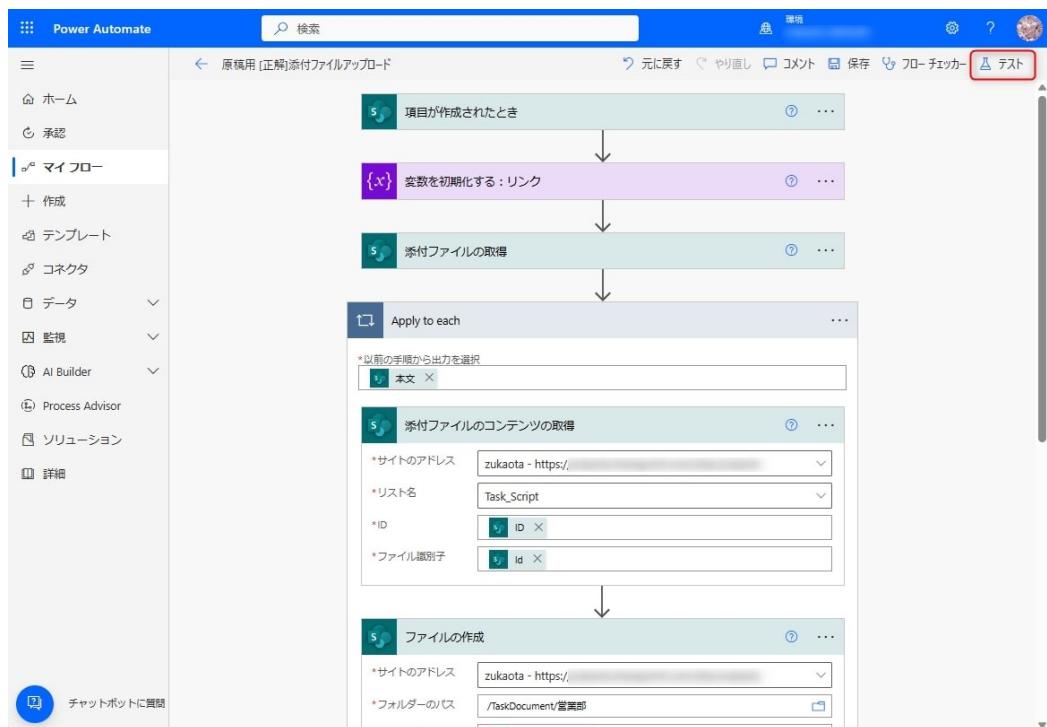


図 11.34: テストの実行



改めてテストすると、フローの成功が確認できました。

図11.35: フローの実行



前回同様、想定する「正常な動作」の確認をしていきます。まずは、Teamsにメッセージが投稿されているかを確認します。

図11.36: メッセージの投稿



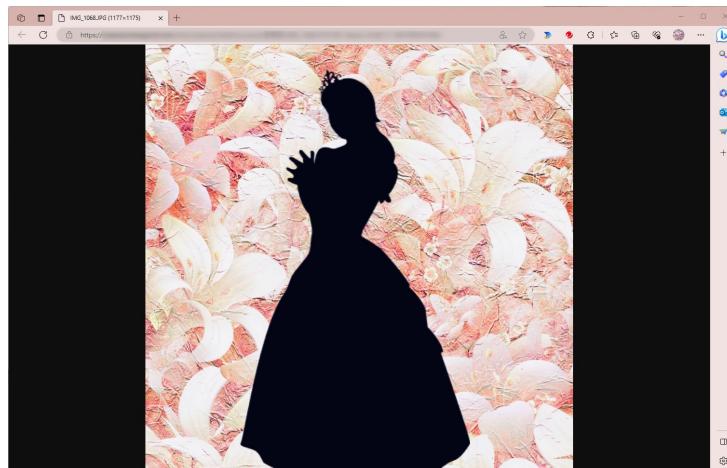
きちんと投稿されていました。次に、メッセージのリンクが開くかどうかを確認します。ひとつのリンク「hokou_data_W3」をクリックします。

図 11.37: Excel ファイル「hokou_data_W3」の表示

家族構成	売上月	売上
ファミリー	4月	¥15,000
ファミリー	4月	¥20,000
単身	4月	¥12,000
ファミリー	4月	¥18,000
単身	4月	¥9,000
単身	4月	¥5,000
単身	4月	¥10,000
ファミリー	4月	¥17,000
単身	4月	¥15,000
ファミリー	5月	¥12,000
単身	5月	¥15,000
ファミリー	5月	¥13,000
ファミリー	5月	¥12,000
単身	5月	¥16,000

Excel ファイルが問題なく開きました。ふたつ目のリンク「MG_1068」もクリックしてみます。

図 11.38: 画像ファイルの表示



こちらも問題なく開きました。ドキュメントライブラリー上のファイルサイズも確認しておきましょう。

図 11.39: 元データ

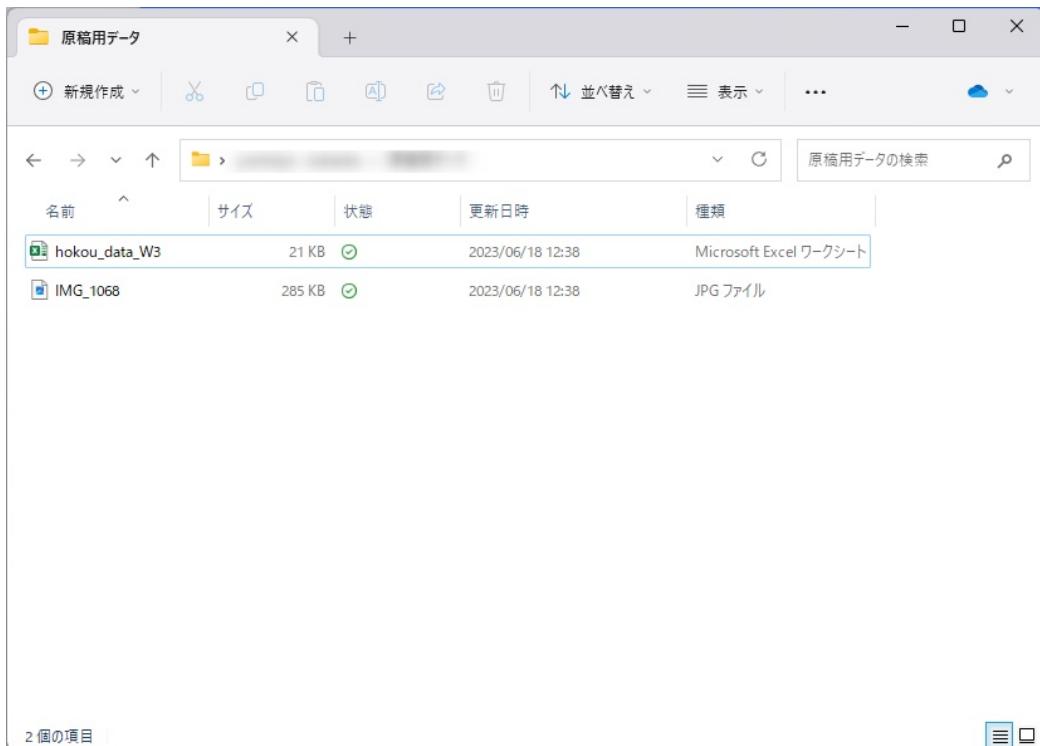


図 11.40: ドキュメントライブラリー上のデータ

TaskDocument > 営業部

名前	ファイル サイズ
hokou_data_W3.xlsx	20.8 KB
IMG_1068.JPG	285 KB

元データとほぼ同じ大きさで、ファイルが作成されていることがわかりました。これでデバッグ完了です。作成者の想定通りに正しく動くフローができました。

11.4 バグのない正しいフローを作るために

バグのない正しいフローを作成するためには、テストが非常に重要です。アクション数が少ない小さなフローなら、頭の中でテスト項目を思い描いてテストすることも可能ですが、アクション数が多いフローやPower Appsなど他のツールと連携するフローは頭の中でテスト項目を思い描いて

も、どうしても抜け漏れが出ることも多いです。そのため、バグを確実に潰した正しいフローを完成させるにはテスト項目を洗い出し、ひとつずつ確認していくことが必要です。最後の効率よく抜け漏れのないテストを行うためのコツを書いていきます。

11.4.1 テスト項目ってどうやって洗い出すの？

Power Automateの場合は、想定通りの「正常な動作」が何かを考え、それを書き出して確認していくのがいいです。今回のフローの場合は、以下の通りです。

- ・SharePointリストにレコードが作成されたときにフローが実行される。
- ・Teamsにリンク付きメッセージが投稿される。
- ・メッセージの中にあるリンクを開くことができる。

11.4.2 テスト方法と想定結果、実際の結果を一覧表にして確認する

テストを漏れなくきっちり行うためには、一覧表にするとわかりやすいです。次の表は、最初のテスト結果をまとめてみました。

図 11.41: テストの一覧表

テスト方法	想定結果	実際の結果
フローの実行 SharePoint リストにレコードを新規作成する	フローが実行される	想定通り
	Teams にメッセージが投稿される	想定通り
メッセージのリンク確認 メッセージ内のリンクをクリックし、問題なくリンクが開くか確認する	リンクが開き、ファイルが表示される	リンクが開かない

テスト方法の太字部分はテスト方法のタイトル、下の文章はテストの方法を説明しています。想定結果がいくつかある場合は、テスト方法のセルを結合させて、ひとまとめに見えるようにしています。この方法のいいところは、4つあります。

- ・Excelなどの表が作成できるソフトを使うと、不足分の付け足しが簡単。
- ・方法と想定結果/実際の結果が1行にまとまって見やすい。
- ・確認事項が記録されるので、抜け漏れがなくなる。
- ・チームでの確認が行いやすい。

あくまでも一例ですが、テストで迷ったときはぜひやってみてください。きっと以前よりも安心感のあるフローが作成できると思います。

11.5 終わりに

今回、Power Automateでのバグの潰し方とテスト方法を紹介いたしました。このふたつを本書で紹介しようと思ったのは、実はMicrosoft Buildがきっかけです。Microsoft Buildでは、WindowsやPower Platformをはじめとする数多くの製品へのAI搭載が発表されました。

Power Automateでも米国レビュー環境のみですが、Copilotが実装されており、フローの作成は自然言語のみで可能となりつつあります。しかし、Copilotはあくまでも「副」操縦士です。作成者のあなたの思考ややりたいことを完璧にトレースできるわけではありませんし、時には間違った解釈をしてしまうこともあります。簡単に動くものが作成できる時代だからこそ、フローが「想定通り」に「正常な動作」をするかどうかを確認するテスト作業は、今後ますます重要になってきます。

皆さんにはぜひ入念なチェックに時間を使っていただき、楽しくAIと共に存する世界の担い手となって頂きたいと思います。ここまでお読みいただき、ありがとうございました。

第12章 普通のOLなら業務に役立つロボットを必ず作れます。もちろんこれは体験談です。

新山 実奈子

図 12.1: For Your DX



これをお読みくださっているということは、あなた様は作業の自動化、ロボット作成にご興味があるか、または既に作成を始めた方とお見受けいたします。

ならば、輝く明日、円満退職、もしくは快適な老後のために、それらの作成や運営、保守作業の術をここでサッと身に着けていただき、今後の役立ててまいりましょう。毎日の仕事を通して、これからずっと使える知恵を身の内に蓄積できますので、全力でお仕事をがんばってまいりましょう。ええ、お仕事とはいよいもので。対価を頂戴しつつ、実践の厳しさの中で自らの知恵、スキルの類向上させてもいただける、素晴らしい舞台でございます。

お仕事とは、あなたを育み、磨く舞台、いずれ自らまばゆく輝くための稽古場でもあると、そうお心得下さいませ。ちなみに、こちらの記事は特に文系の初心者/初学者の皆様に向けて書かせていただきました。草の根系ボトムアップ式の内容となります。お読みくださる際には。その旨をご承知の上、御覧くださいませね。

今回は、何らかのロボット作成 / 自動化ツールを導入する際にお役に立てるお話をさせていただきます。何事も使いまわしのきく御道具は重宝いたします。さあさ、御用とお急ぎのない方は、どうぞごゆるりと、自動化のコツをご照覧あれ。

12.1 自動化の手順も手腕も、三者三様にござりまする。

代表的なローコードツールのRPA/自動化ツールを3種ご紹介します。いずれも新山がお仕事で使わせていただいている、非プログラマに優しい、扱いやすいツールです。

12.1.1 WinActor

図12.2: For WinActor



Windows画面上であなた役を演じるActorロボットを作成

公式サイト :

製造: NTT-AT (<https://winactor.biz/>)

販売: NTTdata (<https://winactor.com/>)

学習におすすめな無料動画: ヒューマンリソシア DX 公式 YouTube チャンネル (<https://www.youtube.com/@dxresocia>)

コピー＆ペーストならば、一晩中、踊りあかせる！

一言で申し上げるならば、「プログラミングを知らなくてもロボットが組める」アプリです。一番の利点は、「ロボットを作成するために覚えることが少ない」こと。要は、組み立てが単純で、役立つロボットを作れる。

次の優れた点は「専用サーバーが必要ないこと」。組んだロボットは、変数表を外に出して参照する形できっちり組んでおけば、ほぼそのまま管理ツールに乗せられますし、一晩中、働かせることもできます。

さらに管理ツールはオンプレミス版とクラウド版管理ツールがあり、このうちクラウド版はAWS上で動きます。そのため、専用サーバーを立ち上げて運用するための費用と労力が不要です。このあたりの導入の容易さも魅力的な、純国産RPAツールです。ただし、お値段が張りますので、個人で勉強するには敷居が高いのが玉に傷。ですが、条件つきでお試し系の数か月だけ無料で使える、あるいは少額で試せるキャンペーンもありますので、ぜひともご検討ください。

さて、こちらの看板に「変数に入れてしまえばこちらのもの」と書きましたが、変数について簡単に説明します。PC上のUIで人間が手動で行う作業は、大別すると次の3つのうちのいずれかです。

それは、

1. アプリ画面のどこかを押すこと
2. 記入欄に値を設定（入力）すること
3. 出力された値を取得すること

ロボットに代行してもらう上記作業のうち、②と③で変数を用います。②では、値をExcel等アプリから読み込む際に、一度、変数に入れます。そして、③では出力された値を取得する際に、変数に入れてから、Excel等のアプリに書き込みます。

変数は「器」のような存在です。たとえば、お財布や小銭入れに似ています。買い物の都度、必要な分の金額を用意し、必要に応じて支払う際に、それらを使用しますよね。品物によって、あるいは量によって金額が変わります。変わらぬ数なので変数、と覚えておくと楽かもしれません。

変数とは逆に決まった数値を入れておく、定数というものもあります。こちらは消費税とかにたとえるとイメージしやすいでしょうか。どの品物でも一律10%（ただし軽減税率の品物は除く）みたいな感じで、固定値として使われます。もし消費税率が上がれば、定数の数値を変更させることで、必要な修正箇所に対して一気に対応できますね。こちらの率ばかりは、上昇ではなく下降していただきたいものです。

12.1.2 UiPath

図 12.3: For UiPath



初心者と非エンジニアは StudioX から使い始めましょう

公式サイト:<https://www.uipath.com/ja>

公式E-learning:<https://www.uipath.com/ja/rpa/academy>

学習におすすめな無料動画: 自動化が誰でもデキる時代に！ゼロから始めるRPA 入門 UiPath StudioX 導入編

<https://www.udemy.com/course/ogushi-rpa-uipath-studiox-01/>

安心してください、日本語化されましたよ。

UiPathは、近年、初心者向けと上級者向けに組み立て用ツールが異なるようになりました。こち

らのロボットは、人がボタンを押してスタートさせる扱いやすいものと、完全自動で動くものに分かれます。どちらもExcelと親和性が高く、扱いやすくなっています。初心者向けの優しいバージョンは、StudioXです。優しいとはいえ、バックオフィスのお仕事自動化は、大抵これひとつで片付いてしまうほど高機能です。何よりうれしいのは、個人で使用する際には、無償バージョンを提供いただけます。制限付きですが、小規模事業者でも無償でお使いいただけます（詳細は公式ページでご確認くださいね）。

さらに嬉しいお知らせがございますよ。公式E-learningが提供されているとはいって、初学者はひとりで学習を進めるうちに様々な不安に駆られたり、わからないこともどう調べればよいか困ってしまうことがありますよね。しかし、UiPathはイベントやコミュニティ活動が、とても充実しています。先輩ユーザーの皆さんも穏やかで優しい方ばかりなので、安心してコミュニティに参加できます。学習をすすめる際に、力になっていただけます。でもコミュニティに甘えすぎは禁物。参加者全員が、楽しく学習を進めていけるよう、長くいいお付き合いができるオトナとして、ご自重の上、ご利用くださると嬉しいです。さあ、コミュニティで僕と握手だ！

12.1.3 Power Automate (デスクトップ版)

図 12.4: For Power Automate



VBEでマクロを組む勢の皆様、ぜひ、お試しくださいね

公式サイト:<https://powerautomate.microsoft.com/ja-jp/>

公式E-learning:<https://learn.microsoft.com/ja-jp/training/>

学習におすすめな無料コースや資格情報等:

<https://www.microsoft.com/ja-jp/events/top/training-days>

Microsoft 365、Office系のアプリに準じたUIが、使いやすいです。あるいは、なにかしらMicrosoft系のプログラミング学習を行った、DOSやらShellやらVisual Basic等の経験がある場合は、その経験を生かしてすばやく習熟が見込めます。Power Platformは単体で使うよりも、各々のアプリの足りない部分をMicrosoft 365系の他のアプリで補完して複数のアプリを組み合わせて、自動化を広げていきます。もちろん、Excelとも親和性が高く、Excelで培っていたスキルも生かせます。

特にExcelの関数使いの皆様には、Appsの命令文の作成は容易です。習得を決意された暁には、ぜひ開発者向け環境を申請ください。OKとなれば、こちらもMicrosoft 365 E5プランの環境が無料で使えます。学習が捗りますよ。

そして、Power AutomateのDesktop版は（無償版であっても）、なんと、部品ごとに「例外処理」が指定できます。例外処理用の部品は、どのRPAツールにも専用部品がそろっていますが、通常はグループ等まとめた部品の集合に対して、行えるものになります。ひとつひとつの部品に標準で例外処理が組めるというのは、初心者には特に嬉しい機能です。

Appsで入出力、Automateが制御、SharepointをDBに！+ PowerQueryとの親和性が高く自動化の強い味方！

もちろんVBAやJSONとの親和性は、とてもとても高くて本当に安心して使えます！お仕事でExcelマクロをVBE(Visual Basic Editor)で組んでいますよ、という皆様はこのツールを試さないと本当にもったいないです。

そしてそして、Excel2016で登場した「取得と変換」機能が、Power Platform版でパワーアップしてその名もPower Queryとなりました！普段のお仕事で「取得と変換」をお使いの皆様は、今後の御社の自動化展開に向けて、ぜひPower Automateをお試しください。

■ Power Queryの解説は、初代Excel MVP田中亨先生のYoutube版がおすすめですっ！

【機能】Power Queryとは何か？何ができるのか？を詳しく解説

<https://youtu.be/Z-08XG5wwRc>

口ボット作成に役立つ”正しい”Excelの解説は、実績と信頼のOffice TANAKAからどうぞ。

お仕事でExcelを使われる際には、誰しも一度はお世話になったハズ。「みんな大好き！VLOOKUP」の名言で有名な、田中先生のサイトはこちらです。

公式サイト:<http://www.officetanaka.net/index.stm>

公式YouTube:<https://www.youtube.com/@OfficeTANAKA>

図 12.5: いつも心にノートを持とう



12.2 ロボット作成時の注意事項を申し上げますね。

繰り返しの多い単純作業等の労力を軽減するために、何らかの自動化ツールを導入する、あるいはロボットを作成しよう、または保守や運用を、とお考えですか？ ならば、失敗と落胆を避けるために、ここで少々お耳を拝借いたします。OL31年生、ロボット作成6年生の新山から、おすすめの下準備についてお話をさせていただきます。

大丈夫、あなた様は、ご自身が担当するお仕事のプロです。どうすれば業務をこなせるかを身に着けてらっしゃいます。ということは、必要不可欠な要素の半分は、すでにお持ちです。ならば、身に着けたお仕事の手順や内容を、これまで後輩に引き継いできたように、ロボットに教えてあげればよいのです。ただ一点、心配な点があるとすれば、ロボットに教えるコツをまだご存じない、ということでしょうか。ご心配なく。私が教わったり編み出したりしたコツについてこれからお話ししてまいります。このお話の幾許かでも、自動化作成のお役に立てれば幸いです。

12.3 突然のツール変更も視野に入れた準備が必要です。

さて、「会社がRPA/自動化ツールを導入することになりました」となった場合には、ある種の覚悟が必要です。いずれのツールも値が張るものです。まずは、3か月なり半年なりの、無料のトライアル期間から入られるかと存じます。トライアル期間を経て、あるいは契約が切れるタイミングで、自動化作成ツールを突然変更されることも、実は、ままあります。そう珍しいことではありません。変更理由は、様々です。費用対効果であったり、客先や本社に合わせようであったり、他の施策や顧客との兼ね合いでということもあります。

そして、この変更は、作成者側に大きな打撃を与えます。たとえば、これが事務用品であれば、変更されたとしても、使い勝手が悪くなる程度で済みますし、慣れてしまえばそれほど困りません。あるいは、消耗品費の枠内でこっそり買いなおしてしまう、という手も使えますが、変更対象が"自動化作成ツール"となれば、なかなかそうは参りません。

なぜ作成ツールの変更が大打撃を与えるかと申しますと、まず、各社のRPAや自動化ツールの間には、互換性はほとんどありません。すなわち、ロボットはすべて作り直しとなります。作成画面も運用画面も用語さえも変わってしまいます。これまでRPA導入のために勉強してきたことが、ほぼ全部通用しなくなります。つまり、何か月も残業をしてまで得たスキルが、ある日突然、使い物にならなくなります。時間的にも費用から見てもメンタル的にも、大赤字確定です。

この悲劇を避けるためにはどうすればよいのでしょうか。それには、「ツールが変更されることもある」ことを念頭に置き、最初の段階から準備をしておけばよいのです。途中でRPAツールの変更があっても対応できる形で準備をすすめましょう。

特に、着目すべき点があります。すべてのRPAツールに有効な、最強のアプリの存在です。この最強のアプリを、あなたの味方につけるのです。どのRPAツールにも必ず専用部品を用意させるアプリ界の超実力者、その名は、皆様おなじみ、どこの職場でも大活躍、"みんな大好きVLOOKUP"のExcelです。「応用の利く基礎が詰まった"正しい"Excelブックを必ず作り出し、あるいは発掘し、探し出し、装備する」これが、RPA/自動化のハードルを下げ、早期導入化をかなえてくれる立役者になります。

Excelごときで、そう変わるわけではないと思われますか？皆様は職場で、業務引継ぎの折に、"これ仕事で使ってるから"と、先輩からExcel文書を渡されませんでしたか？他の表計算ソフトを熟知しているわけでもないため、代替の提案もできず、いわばなし崩し的に、ずるずると、そのExcelとお付き合いを続けておられませんか？

私もそうでした。しかしながら、OLを続けるうちに気が付きました。実は正しいExcelと、そうではないExcelが存在します。どうやって正しいExcelを見分け、味方につけるのかは、若干、長いお話になりますので、ここでは詳しく説明できませんが、自動化やRPAを扱ううえでExcelがあなたを助けてくれることを、この機会に覚えておいてくださいませね。

図 12.6: Excel を自動化に組み込む利点



Excelを自動化に組み込む利点

ブックを複製、カスタマイズし、複数のロボットを持たせることができる。
つまり、社内で横展開する際にも活用できる。
Excelの知識が乏しくともリーダーが作成したブック配布があれば皆で利用できる。
ExcelはTips提供者が多数おりチャットGPT等で速攻スキルアップできる。

構成の一部を簡素化→ロボットは制御に注力できる。

12.4 “最初から100%自動化を目指さない”が成功の鍵！

まずはスモールスタートから始めましょう。自動化を行う予定の作業については、開始時点では、ご自身とロボット（あるいは自動化ツール）との共同作業、協業からはじめることを検討ください。次に、徐々にロボットに任せる比率を大きくしていくことをお考えください。この考え方は、ロボットを作成することにも、スキルを効率よくあげていくにも、運用保守することに対しても有効です。これから、具体的にどのように段階的にロボット/自動化をすすめていくかについて私の経験から、おすすめの手順をお話していきます。

図 12.7: 最初から 100% の自動化は目指さない

最初から 100%の 自動化は 目指さない

1. 自動化の優先順位を決める
費用対効果を計算して、優先順位をつける。
2. 作業を塊に分ける
自動化する予定の作業を、箇条書きで並べましょう。
3. 作業を見直す
箇条書きに並べた各作業が最適であるかを検討します。
4. 分解して再構成する
箇条書きの作業を区切りの良いところで分けていきます。
5. 分担を決める
ロボットと協業する内容とスケジュールを決めます。



<作業手順の例>

- ・「自動化の優先順位を決める」 費用対効果を計算して、優先順位をつける。
 - ・「作業を塊に分ける」 自動化する予定の作業を、箇条書きで並べましょう。
 - ・「作業を見直す」 箇条書きに並べた各作業が最適であるかを検討します。
 - ・「分解して再構成する」 箇条書きの作業を区切りのいいところで分けていきます。
 - ・「分担を決める」 ロボットと協業する内容とスケジュールを決めます。
- それでは、上記それぞれについて、以下で詳細にご案内いたします。ですが、その前にひとつだけ。

12.5 安易に下請けを使うと、危険です。保守ができずに、すぐ錆びてあっけなく壊れます。

これまで、社員自らが作成したと喧伝されるロボットやDominoのような自動化がございます。実は、その多くは黒子が作成しています。黒子は自社社員の場合もありますが、契約社員や派遣であったり、下請け孫請けの方々です。これは、リーマンショック以降、正社員数の急激な減少の影響かと思われます。社内には本業を（残業込みで）ギリギリ回せる人数しか残っておらず、再教育するお金も手段もないことから、教育は外部委託、必要あれば稟議書を回して外注というのが、私が経験してきた2010年以降のスタイルです。

それまではセキュリティの観点から、外部講師を招いて自社内で社員が作成するというスタイルが主流でした。しかし、終身雇用制度が崩れて社員数が激減してからは、社員自らが作成する機会自体が、本当に少なくなりました。

ここ数年、リスクリングという名で、向上心の高い社員が自らスキルを身に着けようとする動きがあります。それも全社員が行えるかというと、これは現実的には余裕のない企業が多く厳しい状況です。200社の人事部担当者を対象に、学校法人産業能率大学総合研究所が出した「日本の企業・

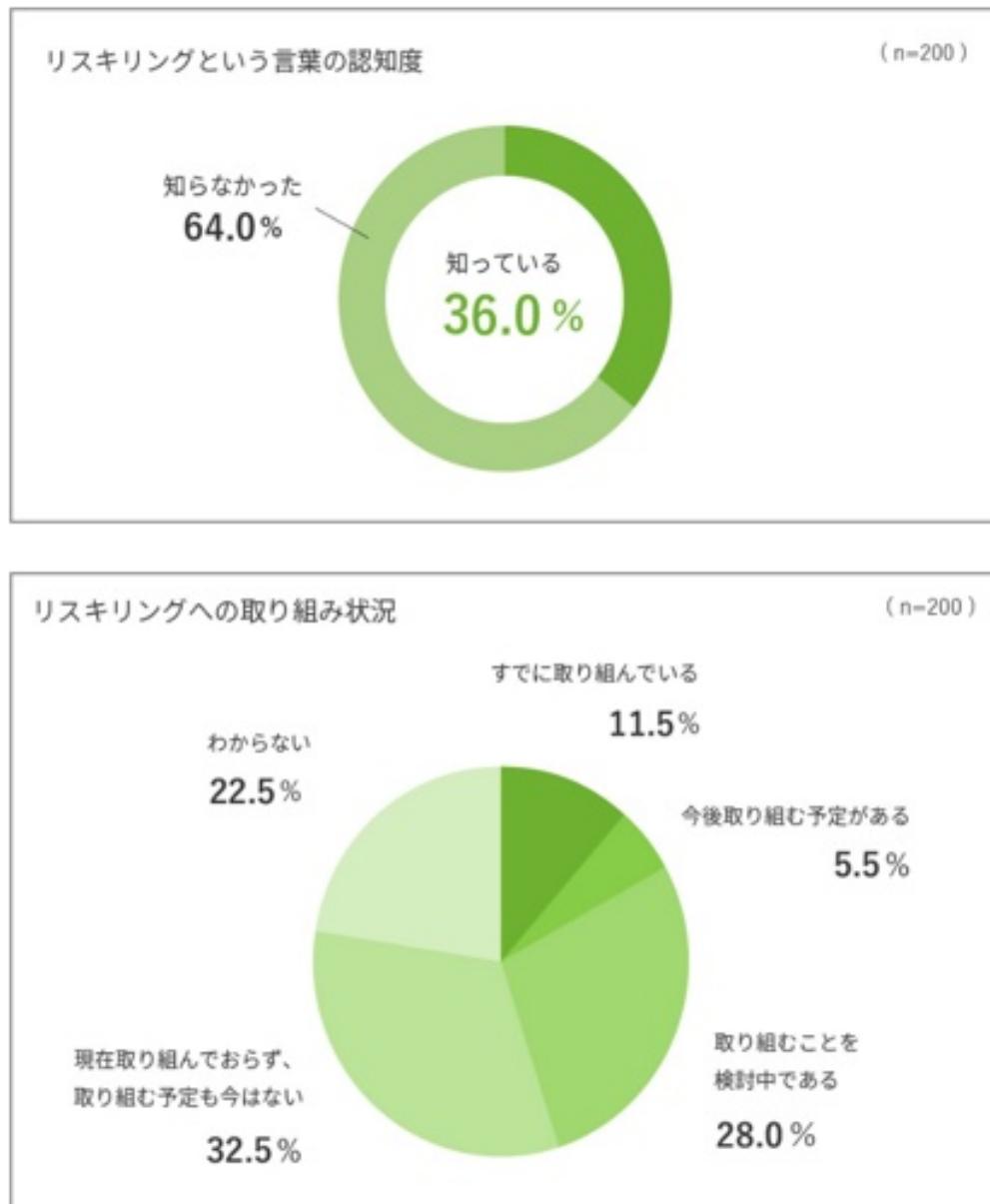
組織におけるリスクリソース実態調査 報告書2021」において、リスクリソースという言葉自体の認知度が36%、すでに取り組んでいるのは、11.5%でした。

<https://www.hj.sanno.ac.jp/cp/research-report/2021/09/30-01.html>

図12.8: リスクリソース1



図12.9: リスキリング2



出展 学校法人 産業能率大学 総合研究所

「日本の企業・組織におけるリスキリング実態調査 報告書」
<https://www.hj.sanno.ac.jp/cp/research-report/2021/09/30-01.html>

上記から、リスキリングが浸透しているとは言えない状況です。しかしこれを踏まえてもなお、ロボットに関しては、社員の方のリスキリングによる技術習得をお考え下さい。外部から講師を呼

んで、しっかりと習得して作成していただいたロボットは、外注した場合と本質が異なります。

自動化作成の現実を見てみましょう。下請けが作成したロボットを使用していて、変更が必要になった、あるいは壊れた場合のことをお考えください。RPA/自動化の歴史は浅く、いまだ標準化や均質化に至っていません。ロボット/自動化作成というのは、属人化の極みです。そうなりますと、作成したロボット/自動化は、ほぼ本人にしか保守できませんし、修復や改修は不可能に近くなる。これが現状です。

もし壊れた場合、派遣や下請けから作成者本人を呼び戻すことは、可能でしょうか？ 残念ながら、まず不可能です。彼らは、次の仕事に派遣されているか、移籍して他社で仕事を行っています。するとどうなるでしょうか？ 別の人が派遣されます。考えてみてください。もし属人化されたプログラムを、何ら引継ぎなく、資料もほとんどない状態で、修正しろと渡されたとしたら、彼はどうするでしょうか？

その場合の多くは、契約期間内に、「最初から作り直す」または「元通りには使えないが、いくつかの部分だけはなんとか動く、似て非なるモノを作成する」あるいは「動かせないし、直せない」とになります。

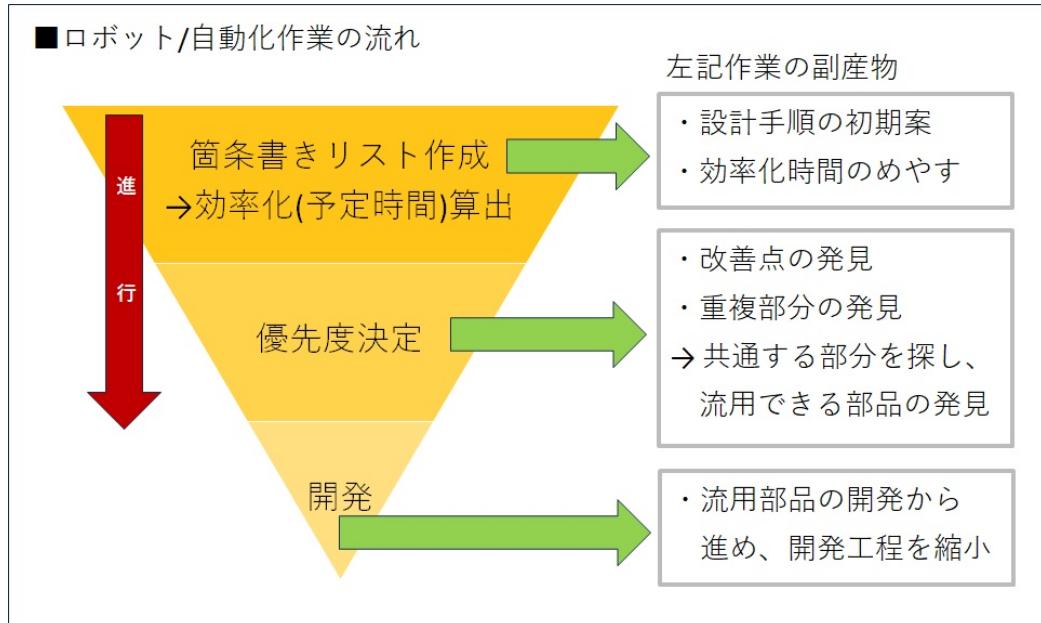
なお、下請けに丸投げしても、期間内に希望した品質のロボット/自動化ができ上がることは稀です。バブル期以降の「下請けに無茶振りすれば、ALL OK！」文化は、通用しなくなっています。下請けも、それをわかっていますから、契約時に「必ず完成品を引き渡す」との条項は入れずに、「完成品の引き渡しは努力目標」に留めることが多くなりました。

ここは、覚悟をお決めください。社員の皆様自らで作成し保守運用を行っていただくことは、結果としてコスト減にもつながります、ぜひとも社員の皆様に、スマールスタートから自動化を広げていただきたい。そして、自らと共に働く仲間たちの手で、輝く未来を、切り開いて参りましょう。

お待たせしました。それでは、作業手順の説明を始めてまいりましょう。

12.6 最初は、半分だけ手伝ってもらうつもりで進めましょう

図 12.10: ロボット/自動化作業の流れ

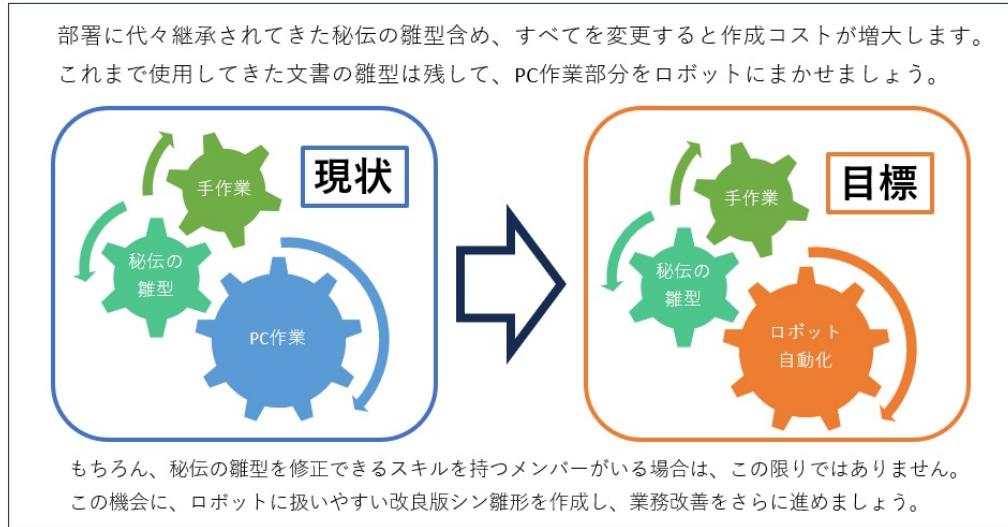


業務をいくつも抱えておられるかと思います。それらをひとつずつ、確実に、正確に自動化しようと、考えられておられませんでしょうか？これまで他のアプリで作成したものも、この機会にロボット/自動化に変更しようとされておられますか？

それは、危険です。難易度が跳ね上がります。初めてロボットを作るときに、難易度の高いものを目指してしまうと、挫折への近道になりますよ。100%確実にになると、設定が増えます。ロボットに任せる部分が大きいということは、設定することが多いということ。つまり、作成者側が覚えなければならないことが多くなりますし、確認やテストをしなければならない箇所も増えてまいります。比例してエラーの発生率も上がりますし、それに伴って修正箇所も増えると予想できますよね。

それを避けるためには、たとえば、「今まで他のアプリを使って作業できていた部分は変更せずに残す！」と決めます。これだけでも、かなりの労力が低減できます。そして、作業の流れを、すべて一度に置き換えるとせずに、段階的に置き換えること、小さくとも成功例を増やすことを優先しましょう。

図12.11: 現状と目標



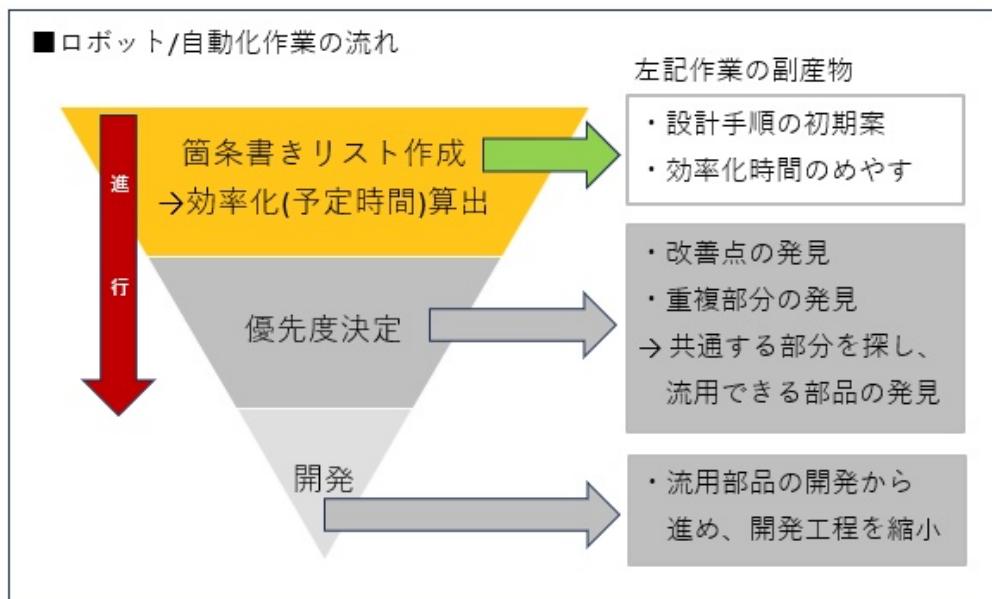
最初は単機能、次の段階では別の単機能をと、単純なものをいくつか作成し、小さな成功を積み重ねて、徐々にスキルを上げていきます。最後に、作成した小さなロボットを組み合わせて、完全自動化まで持っていく、この順番がおすすめです。この手順で100%の完全自動化ロボットをテストし、それが確実に動くとなったら、その次のステップに進みましょう。人間の手を離れ、完全自動で、会議中や終業後から翌朝まで、残業代も休日手当もなく働いてもらうことを目指してまいりましょう。

まずは、業務の中から単純な繰り返し作業の箇所だけを、ロボットに担当させます。次に、人間が手動あるいはExcelを用いて確認や突合の作業を挟んでください。ご存じの通り、技術の習得をする際は、最初のひとつを完成させることに最も時間がかかります。その次からは、同じ内容であっても、所要時間は短くなります。いくつか作成するうちに工夫が生まれ、よいものが作れるようになっていきます。ですから、簡単なロボットから始めましょう。いくつか簡単なロボットを作成することでスキルが上がり、それに伴って視野と発想が広がり、解決策にも幅が出てきます。

まずは、半分だけ、あるいは少しだけ手伝ってもらうために、簡単なごく小さなロボットをいくつか作成します。そして実際に動かしてみながら、ロボットと人間が協業することに慣れましょう。いくつか成功例を作り、それを横展開で社内に広げることが、自動化を無理なく成功させる近道です。その視点を手に入れることを目指してくださいね。まずは、あなた様から、仲間たちに、そして、社内の皆さんの順番で、よろしくお願ひいたします。

12.6.1 自動化の優先順位を決めましょう。

図 12.12: 自動化の優先順位を決める



**最初から
100%の
自動化は
目指さない**

- 1.自動化の優先順位を決める
費用対効果を計算して、
優先順位をつける。
- 2.作業を塊に分ける
自動化する予定の作業を、
箇条書きで並べましょう。



さて、上司を説得、あるいは納得させるためには、効率化できる裏付けになる数字が必要になります。どの業務から手を付けるかは大きな問題です。おすすめは費用対効果を計算して、優先順位をつける方法です。まず、担当業務リストを作ります。次に、そのリストから、ロボット/自動化部品を選定します。テーマや処理の流れ、手順が異なる上位3つの単純なつくりの、ただし行う作業にかぶりのないロボットを選んで作成しましょう。これらを雛型にして、横展開する際や、先々で

作成するであろう複雑なロボットの基礎になる"素体"を作成しておくのがおすすめです。

担当業務リストには、具体的な作業を箇条書きで並べていきます。そのあとで、必要に応じて適宜組み換え等を行い、あなたの作業に沿った形で最適化していきます。上司用の、説明用資料のメモもあわせて書き添えておきましょう。手順は次の通りです。

1. 任意の担当業務を洗い出し、それぞれの内容を単純な箇条書きとしリストを作成。
 2. 自動化を予定している担当業務の数だけ、すべて(1)を行う。
 3. およそその時間を算出し、ひと月あたり何人月かかっているかを割り出す。
- 割り出した時間数から、業務毎に効率化できる時間が決まります。他に配慮する項目がなければ、この時間が大きなものから作成します。なお、上記の1.から3.で割り出した時間は、のちのち上司に説明する資料になりますので、メモしておきましょう。

12.6.2 これから作成するメモのサンプル

さあ、次の例を参考に、あなたのメモを準備していきましょう。

■会議資料の作成

- ・毎週火曜日、午後一開催の会議で使う資料を作成する。
- ・金曜日の午後4時に課長にメールで送信、CCに主任を入れる。
- ・変更点あれば月曜日午前中に修正、なければそのまま会議フォルダーへ格納する。
- ・データ集めは吉田さんと田中さんと、あと経理の大泉先輩に手伝ってもらう。
- ・Net上にある収集用URLとIDとPW(パスワード)は変更ないかチェックしてから3人に配布。

上記のようなメモを、業務の数だけ作成してくださいね。最初の段階は、骨組みだけで十分です。

12.7 現状の状態と、どの程度効率化できるかを可視化する

それでは、時間を算出し、ひと月あたり何人月かかっているかを割り出しますよ。最終的に、どれだけの時間が削減できる見込みがあるかを算出していくます。

・計算の手順

- (1) 任意の作業に対して現状でどれだけ時間をかけているかを、算出します。
 - (2) その作業のうち、PCで行う作業の割合(率)を仮定します。
 - ・この部分が、後でロボット/自動化で効率化できる時間になります。
 - (3) (1)に(2)をかけて、効率化できる(予定の)時間を算出します。
- ・単純な計算の例: 会議資料を作成する際には、資料作成の時間の他に、作成のために情報収集の時間が必要ですし、承認者が存在する場合は承認者の時間も必要です。これらを併せて考えます。
- (1) 毎週末に行うのであれば4回、作業時間が2hあれば $4 \times 2 = 8H$ です。
 - (2) その作業の90%をPCで行う場合は、 $8H \times 0.9 = 7.2H$ になります。
 - (3) 忘れずに作業人数を(2)にかけて、作業ごとの合計時間を出してください。

図 12.13: 単純な計算の例

業務種類	業務内容	単位作業時間	人数	回数	PC作業率	効率化予想時間	効率化難易度	優先順位
会議資料作成	資料収集	60分	3	4	90%	13H	容易	高
	作成時間	120分	1	4	85%	14H	容易	
	承認時間	15分	1	4	100%	1H	容易	
経費申請	資料収集	15分	8	1	30%	1.3H	容易	低
	作成時間	20分	8	1	50%	1.3H	容易	
	承認時間	30分	1	1	100%	0.5H	容易	

ひとつの担当業務を行う際には、複数の作業があり、またその作業ごとに細かい作業が発生しますので、その作業毎に、上記の時間を算出します。

- ・担当業務リストに箇条書きに書き出した箇所のおおよその時間を青字で書き込みます。
- ・1か月ごとに算出します。複数回実行する作業はその回数分をかけてください。
- ・他の部門に手伝ってもらっている分、下請けに出している分も考慮します。
- ・おおよその時間を合計して、判断材料にします。

冒頭で申し上げた担当業務の各箇条書きリストに対して、費用対効果の表を作り、どの部分を自動化するのが効率的であるかを勘案して、上位のものからロボット/自動化するかの優先順位を決めていきましょう。

12.7.1 作業を塊に分けましょう。

仮の優先順位が決まりましたら、次に自動化する範囲を決めていきます。目印になるのが作業の塊です。作業を塊・まとまりごとに分けて、効率的に自動化できる範囲を見つめましょう。

- ・箇条書きの担当業務リストから引継ぎ準備メモを作る。
 - 各作業の引継ぎ用メモも、箇条書きにして並べます。
 - 余白は、十二分に取ってね。
 - 番号を付けたり、印をつけたりして、わかりやすくしましょう。
 - 「繰り返しの単純業務」があれば赤枠で囲むか、小さく赤字等で書き添える。
 - 感想もヒントになり得ます。最初のうちは、書き加えていきましょう。

12.7.2 これから作成するリストのサンプル

さあ、次の例を参考に、あなたの準備メモを修正していきましょう。

■会議資料の作成 その2(準備用)

- ・目的: 毎週火曜日、午後一開催の会議で使う資料を作成する。
 - 目的は、当面変更ないのでこのままでOK。
 - 資料集めのときに、フォーマット渡してあるけど、結構、バラバラな様式になる。これを自動で揃えたい。
 - Excelのマクロで、シートに貼り付けると、整理して表にコピペできるのを見た。よその部署だけど、譲ってくれないかな。頼めないけど、教えてもらえるかな。本當は、こっちを自動化したい。
- ・日程: 金曜日の午後4時に課長宛にメール。資料を添付して送信。CCに主任を入れる。
 - 人事異動の時に宛先変更あり(現状ではこの部分は自動化が難しそう)
 - 異動報がイントラサイトに掲載されるようになったら、自動化できる?
 - メーリングリストを作れば一斉同報できるから、これはOutLookでもOK。送信前に目視確認しよう。確認誤送信したら始末書だから、それは避けたい。
- 手順:
 - ・変更点あれば月曜日午前中に修正、なければそのまま会議フォルダーへUp。
 - フォルダーの名前とパス(どこにあるか)を調べておく。
 - 変更あるかないかで、作業が分かれ。分歧発生。
 - フォルダーも整理したほうがロボットにわかりやすいかな。
 - ・データ集めは吉田さんと田中さんと、あと経理の大泉先輩に手伝ってもらう。
 - ここが全自動化できたら、先輩たちの手が空くかな?
 - ・Net上にある収集用URLとIDとPWは変更ないかチェックしてから、お手伝い3人衆に配布。
 - メンテ等で変更もある。メールは私宛にくるから目視でチェックが必要。

メモにどんどん追加して具体化していきましょう。これを、業務の数だけ作成してくださいね。先々で役立ちますので、面倒ですががんばりましょう!

ここまででの作業で担当業務数だけのリストを作成いただきました。さらに、そのリストの中には業務に必要な作業を箇条書きにした内容を作成いただいたかと思います。何事も、アウトプットは大事です。時間はかかりますが、自動化に必要な数だけ、リストの作成を引き続きお願いします。次の作業では、これら作成したリストを、先々で自動化の設計図に流用できるよう直していきますよ。

12.7.3 リストの修正タイム

さあ、設計図に向けて整えていきましょう。下表はサンプルの修正例ですよ。

■会議資料の作成 その2(清書)

- ・目的: 会議で使う資料を作成する。
 - 自動化希望案1.「メール添付資料から、統合時の様式を統一する」
 - 手順
 - ・1. Outlook受信フォルダー内の該当メールを特定する。
 - ・2. 当該メールの添付資料を、PW入力して解凍する。
 - ・3. 解凍した資料を、メモ帳経由で任意のExcelに貼り付ける。
 - ・4. 任意のExcelには、あらかじめ書式を設定しておく。
 - ・代替案:「データ収集そのものを自動化」
 - 現行では、他部署含め3名に協力してデータ収集を行ってもらっているが、これをロボットに代行させることで、大きく省力化が見込める。
 - 手順

- ・1. インターネット上のファイルサーバーに接続。
 - ・2. ID/PWの入力(セキュリティーのためここは手作業)。
 - ・3. 任意のファイルを指定して、任意のフォルダーへダウンロードを行う。
 - ・4. 作業後にログアウトする。
 - ・5. 任意のフォルダーから、当該ファイルをPWを使って開く。
 - ・6. 資料収集用ファイルに開いたファイルから必要箇所をコピペする。
 - ・1.から6.の作業を必要なデータ分、繰り返す。
- 手動確認1. URL、IDおよびPWが変更されていないかを事前に確認する。
- 事前準備1. 作業が容易に自動化できるよう、フォルダー整理を行う。
- ・自動化希望2. 「資料作成時の労力を低減化する。」
- 検討: Excelの関数またはマクロ導入による自動化。
- ・他部署にて成功事例あり。
 - ・当該部署に取材と協力を希望する。
- ・自動化希望3. 「資料修正時の労力を低減化する」
- 検討: 上司から修正の指示があった場合の、対応見直しを行いたい。
- ・フォルダー移動作業の自動化を行いたい。
 - ・事前にフォルダーネ名、ファイル名のルール等の整備を行う必要あり。
- ・日程: 開催日時 毎週火曜日 13時～
- 準備期間 金曜日の午後4時に課長宛に資料添付(CC: 全主任)。
- 手動確認1. 「人事異動時に宛先変更の確認を行う。」
- ・異動報がインターネットサイトに掲載された場合は、こちらも自動化検討を希望。
- 手動確認2. 「事前承認用のメーリングリスト送信前には、必ず目視確認を行う」

効率化、実現へのヒント等を、思いついた順に、どんどん追加していきます。

では、上記を踏まえてリストの箇条書きを見直す作業に入りましょう。具体例をあげていきますので、参考になさってください。

・着目点

— “人間の判断が必要である” ものに取り消し線をつける(消さないでね)。取り消し線という目印で区切る理由があります。その前後で分けることで、部分的に自動化できる場合があるからです。

—繰り返しを見つけましょう。リスト内の業務を分解して考えて、お決まりの単純作業を探します。発見できたら、その個所をくるっと囲むか、星印等の目印をつけて、後で見分けられるようにしておきます。

一分岐を見つけましょう。斜線を入れて、赤字で「ここで分岐」と書き込みます。分岐先への矢印をつけるか、何か番号をつけてどこへ進むかを明らかにします。

一書き込むことが増えた場合は、切り貼りを行うのも手です。別の用紙を準備して、貼り付けていく、あるいは付箋で付け足す、いっそ全部付箋に置き換えもOK。

—WordやExcel、メモ帳を使うのも、もちろんアリです。やりやすいツールでGo!

—この作業も、ゆくゆくは効率化につながります。面倒ですが、がんばりましょう。

自動化するお仕事の数だけ繰り返し、塊リストを作成しましょう。次に、リストをコピーして見直す。今度は、自分が手伝ってもらえると助かる順に並べ替えて、先ほど作成したものと比較し、どうすればよくなるか勘案してみましょう。この段階で、作業時間効率化メモと同様に、上司説明用の文書用のメモを作つておく。この作業は、適宜付箋貼るくらいでもOKです。これらの作業を

行なって、本格的に優先順位が決まつたら、次の作業に進みます。

ポイントは、分岐箇所を発見することと、単純作業が繰り返される部分の特定です。

なお、PC画面でロボットが行う基本的な行動は、次の3点です。

1. アプリ画面のどこかを押すこと
2. 記入欄に値を設定(入力)すること
3. 出力された値を取得すること

この3点を、現実の作業はどう落とし込むかを考えてリストの内容を見直しましょう。

とはいって、どういった部品があるのかが不明のままだと、イメージがわかないですよね。部品名は異なりますが、おおよそ、細分化された約500程度の部品が提供されています。筆者としてはおおよそ数十個の部品を使えば、たいていの開発はカバーできると考えています。次に、RPAツール側で提供される部品を簡単にまとめましたので、目安として参考になさってください。

12.8 およそ大体のRPAアプリで準備されている部品の簡単なまとめ

- ・変数に値を入れたり出したりする部品
- ・フォルダーやファイル、URLを開いたり閉じたりする部品
- ・ExcelやWord等のOffice製品のための部品
- ・テキストを書き込んだり読み込んだりする部品
- ・数字や日時を計算する部品
- ・変数内の文字列や数式をあれやこれや変更したり追加や削除する部品
- ・繰り返しの回数や範囲を指定する部品
- ・エラー対策用の部品（エラーを見つけたり、エラーが起きた場合の処理を指定）
- ・画面の全体指定、範囲指定、キャプチャ等を行う部品
- ・各種ブラウザへの対応、操作を行う部品
- ・サブルーチン（よく使う部品の塊／必要な都度、呼び出して使用する）制御
- ・グルーピング用の部品（部品をまとめてグループ化し扱いやすくする）
- ・グループ毎、あるいは全体を制御用部品
- ・動作時に画面にRPAアプリを残す、残さないスイッチ、一時停止等を扱う部品

12.8.1 リストの修正タイム

さあ、さらに具体化に向けて、メモを修正していきましょう。

■会議資料の作成 その3

- ・目的: 会議で使う資料を作成する。
- ・自動化希望案1。「メール添付資料から、統合時の様式を統一する」
 - 小型ロボ1. 受信メールから添付資料を探し、解凍して指定のフォルダーに格納する。
 - ・ロボットは1の作業を、3回繰り返す。

- ・メール有無により分岐も作れるが、今回は複雑にせずまとめて処理。
 - ・手動：処理が空振りしないように、受信を確認してから動かす。
- 小型ロボ2. 指定のフォルダー内の各ファイルを開いて、必要部分をコピーする。コピーした内容を、資料ファイルへペーストする。
- ・ロボットは1の作業を、3回繰り返す。
 - ・ファイル有無により分岐も作れるが、今回は複雑にせずまとめて処理。
 - ・手動：処理が空振りしないように、ファイル数を確認してから動かす。
- 小型ロボ3. 資料ファイルにあらかじめ設定した、関数 / マクロを動かす。
- ・ロボットは1の作業を、3回繰り返す。
 - ・手動：処理が空振りしないように、ファイル数を確認してから動かす。
 - ・手動：要望通りに資料転記が行われたか、間違いないかをチェックする。
- ・手順：
- ①Outlook受信フォルダー内の該当メールを特定する。
 - ・1)Outlookは開いておく。
 - ・2)送信元の名前と、メールタイトルから対象を探す。
 - ②当該メールの添付資料を、PW入力して解凍する。
 - ・1)解凍用のフォルダーを作成しておく。
 - ・2)PWを確認しておく。外部から見られない形で保存しておく。
 - ・3)解凍後に、もとのファイルは保存しない。
 - ③解凍した資料を、メモ帳経由で任意のExcelに貼り付ける。
 - ④貼り付ける側のExcelには、あらかじめ書式を設定しておく。
 - ・1)メモ帳を開く。
 - ・2)転記元ファイルを開いて、必要箇所をコピーする。
 - ・3)書式を削除するため、いったんメモ帳にペーストする。
 - ・4)メモ帳から、必要箇所をコピーする。
 - ・5)転記先ファイルを開いて、必要箇所をペーストする。
 - ・代替案：「データ収集そのものを自動化」
 - 現行では、他部署含め3名に協力してデータ収集を行ってもらっているが、これをロボットに代行させることで、大きく省力化が見込める。
 - ・手順：
 - ①インターネット上のファイルサーバーに接続。
 - ・1)事前にURLが変更ないか確認する
 - ②ID/PWの入力（セキュリティーに関わるため、ここは手作業で対応します）。
 - ③任意のファイルを指定して、任意のフォルダへダウンロードを行う。
 - ・1)事前にファイル名、フォルダのパスを確認する。
 - ④作業後にログアウトする。
 - ⑤任意のフォルダから、当該ファイルをPWを使って開く。
 - ・1)事前にPWを確認する。
 - ⑥資料収集用ファイルを開いたファイルから必要箇所をコピペする。
 - ・小型ロボ2.を流用、一部修正して作業を行う。
 - ・上記の①から⑥の作業を必要なデータ分、繰り返す。
 - 手動確認1. URL、IDおよびPWが変更されていないかを事前に確認する。
 - 事前準備1. 作業が容易に自動化できるよう、フォルダ整理を行う。
 - ・自動化希望2.「資料作成時の労力を低減化する」
 - 検討：Excelの関数またはマクロ導入による自動化。
 - ・他部署にて成功事例あり。
 - ・当該部署に取材と協力を希望する。
 - ・小型ロボ3.を流用し、省力化を行いたい。
 - 手動確認1. 担当者確認→打合せを依頼、時間を確保する。
 - 事前準備1. 打合せ前に、依頼内容をプレゼンテーションファイル(PPTXファイル)に作成し、メール添付して送信する。

- ・自動化希望3.「資料修正時の労力を低減化する」
 - 検討: 上司から修正の指示があった場合の、対応見直しを行いたい。
 - ・小型ロボ2.を流用し、フォルダー移動作業の自動化を行いたい。
 - ・事前にフォルダーネ名、ファイル名のルール等の整備を行う必要あり。
- ・日程: 開催日時 毎週火曜日 13時～
- 準備期間 金曜日の午後4時に課長宛に資料添付 (CC: 全主任)。
- 手動確認1.「人事異動時に宛先変更の確認を行う」
 - ・異動報がイントラサイトに掲載された場合は、こちらも自動化検討を希望。
- 手動確認2.「事前承認用のマーリングリスト送信前には、必ず目視確認を行う」

最初のメモに比べると、かなり具体的になってきましたね。

この段階まで来たら、実行させるために、どういった動きをする部品が必要で、何を設定すればよいかを考えます。どこが重複していて、どのロボでも同じ動きをする箇所はどこかを探しましょう。最小公倍数を見つけるように、注意を払って最適解を見いだすコツをつかんでください。これはどの場面でも必ず行う作業ですので、手順を掴んでしまえば、非常に楽になります。まずは、ここをしっかりと押さえていくことが上達の早道です。

12.9 黄金パターンをご紹介します (Excel間のデータやり取り、ログイン等に使用)

1. 入力用のリストと、出力した文言や図や表を格納するフォーマットを準備します。
2. 入力用リストを開き、変数に入力する文字や値を代入します(変数についての説明は、冒頭のWinActorの紹介欄を参照ください。何らかの器、容器、ケースが変数。たとえば本章の最初の方では、お財布みたいなものが変数とお話ししました)。
3. ロボットに入力欄を押させる。
4. 変数に格納された文字や値の貼り付けを行う(※入力するだけならここで終了です)。
5. 出力された文字や値や図を、変数に格納する。
6. あらかじめ準備した出力フォーマットを開く。
7. 変数の中身を所定の場所に押して、貼り付ける。
8. フォーマットを閉じるか、別のファイルに貼り付ける等の作業を行う(ファイルの開閉等よく使う部品は、あらかじめRPAツール側で準備されています)。

さらに、確実に入力できたかどうかを確認するために、追加作業によりミスを避ける安心設計を目指しましょう。値の貼り付けを行った後に、変数に入ったままの文字や値と、入力欄に貼り付けた文字や値を比較する仕組みを作り、このふたつが一致していれば○、していなければ×等を別の欄を設けて記入します。ロボット作業後に人間がこの列を確認し、失敗あれば手でリカバリします。クラウド+PCは、専用線+専用システムより安定性が劣る環境なので、ロボットで自動化したときにミスはどうしても起こりやすくなります。ロボットと協働してさらに良いお仕事をするためには、常に人の目で確認し、エラーやミスが出た際には手動で修正することを心がけてください。

図 12.14: 確実に入力できたかどうかを確認

転記	業務種類	業務内容	単位 作業時間	人数	回数	PC作業率	効率化 予想時間
○	会議資料 作成	資料収集	60分	3	4	90%	13H
×		作成時間	120分	1	4	85%	14H
○		承認時間	15分	1	4	100%	1H

ここからは、文字ではなく画面を対象にロボットの動きを追っていきます。箇条書きリストを片手に、実際のPC画面をキャプチャする作業に入ります。省略せず、キャプチャ画像を全部集めていきます。Excelにどんどん貼り付けていきましょう。

可能であれば、全画面をコピーして紙に打ち出し、その束の各ページに、ロボットが「押す」等の具体的な操作を行う箇所を赤丸で囲んでください。入力する場所には、何を入力するかを青ペンで記入します。あるいはExcelにそれぞれの画面を貼り付けていき、赤丸を適宜配置、青文字で注意書きや入力内容を書き添える、こちらの方法でも結構です。実は、この画像集は、これだけでも簡単な設計図にもなりえます。紙は便利です。私たちは千年以上、紙と一緒に仕事をしてきました。これからも必要な場面で適切に使いましょう。

さて、集めた画像を並べてみてください。どこで分岐しているか、繰り返しているかが判別しやすくなるかと思います。最初に作るロボットは簡単なもの、分岐も繰り返しもなく、部品をただ順番に並べた部品集です。その部品集を使って、後々で分岐と繰り返しを加えて実務に耐えるものに仕上げていきましょう。ここでは、どの箇所が核になっているかを見つけ、部品の候補をいくつか仮決めしておき、担当業務の箇条書きリストの添付資料としましょう。

ものすごく忙しくて、設計図をお願いしても出してもらえない部署のロボットを、この方法で作成したことがあります。忙しい部署ほど自動化は必要なので、導入後に、とても感謝されました。手順と画面がわかれば、なんとかなります。本当に忙しい部署には、フロー図でなくても紙束でOK！として差し上げてくださいね。

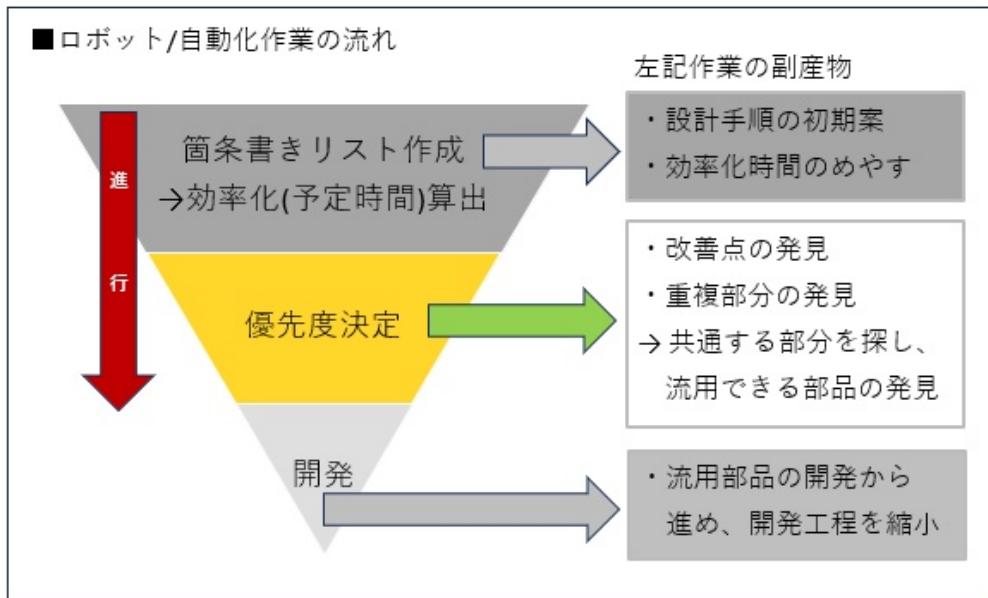
なお、原則として「ロボットには、判断ができない」という問題があります。ですので、判断が必要なものは自動化作業に向きません。思い出してください、皆様が業務を引き継がれるときも「これは、どうしますか？」と、ひっきりなしに聞いてくる後輩がおられたかと存じます。しかし、それは、長くは続きませんでしたよね？

その理由は、同じ作業を何回も繰り返すことによって、判断力が身についたからではないでしょうか。ロボットや自動化には、通常、繰り返し作業の実行で判断力が上がる機能はありません。ですから、判断が必要な作業については、人間が行う必要があることを意識なさってください。判断が必要な個所までをロボット、判断は人間、判断後の繰り返しはロボット、のように協働から始めましょう。完全に自動化するためには、ロボットが判定できるような工夫が必要になります。具体的には、判断の代わりとなる目安やものさし、確認表などを作成して、判断したと同様な結果が出せるかを繰り返しテストを行って確認する必要があります。まずは単純に、ゆくゆくは複雑にとお

考え方下さいね。

12.10 作業を見直します。

図12.15: 作業を見直す



**最初から
100%の
自動化は
目指さない**

3. 作業を見直す

箇条書きに並べた各作業が
最適であるかを検討します。

4. 分解して再構成する

箇条書きの作業を
区切りの良いところで分けていきます。

5. 分担を決める

ロボットと協業する内容と
スケジュールを決めます。

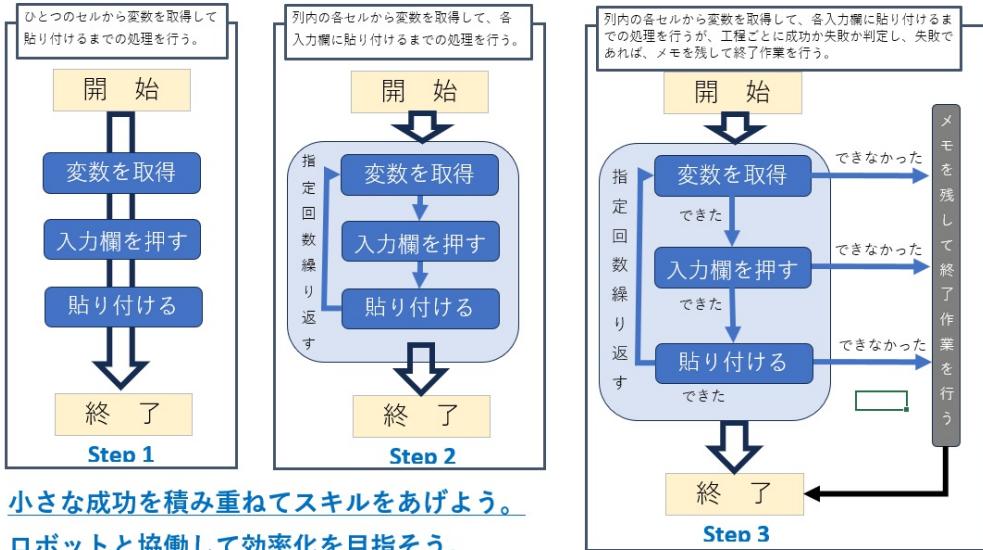


作業を見直す時に大切なことは、最初は分岐しないコースを想定し、まっすぐなレールを敷くことです（下図 Step1の状態）。最初は分岐も繰り返しもない、最短コースの直線的な、必ず成功する例をレールの上に作ることが大切です。まずはシンプルにしっかりと形を整えます。バリエーション

ンは後から付け足します。

かといって、さきほどリスト上で作った分岐や繰り返しの箇条書きが無駄になるわけではありません。まっすぐなレールで最短コースがうまく動くようになってから、分岐を加え、繰り返し作業を追加していきます。まずは、設計やテストでもこの短いまっすぐなレールのコースがしっかりと動くように、何回も練り直します。

図 12.16: 作業の流れ (Step.1~3)



それでは、ロボット/自動化ツールを3つ作る気持ちで作成しますよ。作成順も優先順位を気にしてつけてまいりましょう。ここで作成するウチの子ロボは、後で横展開をする場合の素体となりますので、種類の違う自動化を選んでくださいね。さきほど、費用対効果の順番と、手伝ってもらえると助かる順番をつけてくださったかと思います。今度は、それら上位の子たちにダブリがないかをチェックします。役割がダブってしまった子は、いったん外しましょう。必要であれば、今回作成する素体を流用して次の機会に作成してあげてください。

ここでは、複数のExcelをまたいだ処理、クラウドから資料をダウンロードするもの、備品やファイルの管理表など、どの作業でも必ずといっていいほどお世話になる機能の塊をリストと突き合わせて見つけていきます。機能のまとめ、塊で分けて、小さなロボットを作ります。ひとつの塊として分けることができる、役目ごとのちいさなロボットを作成していくことにいたしましょう。ちいさなロボット、略してちいロボさんですね。ちいロボさんは、素体用の、お役立ち&愛されロボを目指しますよ。

確認です。よろしいですか、ロボットは単純なことしかできません。複雑なことには向いていません。そうですね、あなたの仕事を引き継ぐ後輩が、アホな子だと仮定してください。ただしめっちゃスピード速くて正確。根がアホなんだけど、正確かつやることがとても速い、濃ゆいキャラの子です。そして笑顔がとても可愛い。そういう子には、どう指導してあげればいいですか？

ダメですよ、藁人形と白装束を準備しては。金槌とろうそくも買ってきました？それは災害用の備えにまわしましょう。いや、すごい行動力ですね。さすが、お仕事が速い。素晴らしい。いや本当に。ま、それはそれとして、よろしいですか？仮定のお話ですからね、イメージですからね、ホントにそういう後輩がいるわけじゃないんですよ。落ち着きましょう。はい、ここで深呼吸です。

それでは、もう一度、お尋ねしますよ。一度にたくさんのお仕事ができない、単純な作業なら速くて正確にできるタイプの子には、どのようにお仕事を指示していけばよい関係を築けますか？そうです、各作業ができるだけ細分化、単純化して、型にキッタリはめてあげればよいのですよね。では、それをちいロボさんにもあてはめて、作業を進めて参りましょう。

前作業として、それぞれのちいロボさんになる予定の塊に手を入れます。自動化する予定の箇条書きの作業を、今度はさらに細かく、部品化していきます。本当に単純なパーツに分解していきます。これをタブか字下げを使って、箇条書きで並べましょう。行間は多めにとってください。一番細かいと思ったパートでも、さらに細かくなることもあります。余白大切。

このときに大切なことは「現在の業務で使っているアプリ/ファイルは最大限残す」ことです、たとえRPA/自動化の作成ツール側に専用の代替部品が存在していたとしても、その機能に特化した専用アプリと代替品の作成ツール部品では、処理速度や性能が異なります。これまで専用アプリで行っていた作業に関しては、そのすべてを自動化ツールの中で置き換えるのではなく、その専用アプリが得意なことは専用アプリに任せることをおすすめします。

この考え方を軸にしていただき、ロボット/自動化ツールには制御を中心にして組み立て担当として働いてもらうことにして、作業の肝となる必要な各アプリの間を”繋ぐ”構成としましょう。こうしておけば、ロボット/自動化ツールが突然、変更になったとしても、最小限の置き換えで済みます。このあたりの備えも大切です。

図12.17: 自動化は単に手段であって目的ではない



12.11 そのまま乗せ換えることが正解とは限りませんよ？

というよりも、現行のシステムをそのまま乗せることは難しいです。安からう悪からうではないですが、安価であるということは、それなりの理由があるわけです。全部が今より新しくて素晴らしいそれでいて安いとすれば、それは夢の製品ですが、実現化には今よりも数多くの技術革新が必要となるでしょう。今は手が届かない、未来に発売される製品ですね。

我々は、上司が決めたツールでちいロボさんを作つてお仕事を助けてもらいましょう。ゆくゆくは、社内中で様々なかいロボさんたちの活躍がみられるはずです。そのためには、さきほど取り組んでいただいたように、「単純なことしかできないけれど、速くて正確」であることを生かして、現在ある作業を、どう変形・合体させたら効率化できるかを考えてみてはいかがでしょうか。

図 12.18: 我がロボに一点のヒューマンエラーなし!



これはナイショのお話ですが、私のお師匠は「"ロボットには、こうしないと業務が載せられないんです" という言い訳を生かせ。これまで上司先輩から邪魔されてきた、ずっと進めたかった効率化があるだろうから、それを今行え」と教えてくださいました。現場の人間が一番、業務に関して理解している。だから、これじゃなきゃ自動化できないんだというのを言い訳にして、この機会によくわからない癖に横やり入れたがる上司先輩を排除して、やりたくてもできなかった効率化に、こっそり挑んでしまえ、と、それは思い切り悪い笑顔で進めてくださったのですわ。

もちろん、上司先輩からの指摘が正しい場合はあてはまりません。ロボットにかこつけた修正の提案は、あくまで理不尽に妨害を受けて来た場合に限ります。それに、効率化の目途がきちんとたつていることも重要です。そうでなければ、自分の首を絞めるだけになりかねませんので、その点も理にかなったものであることは必須です。

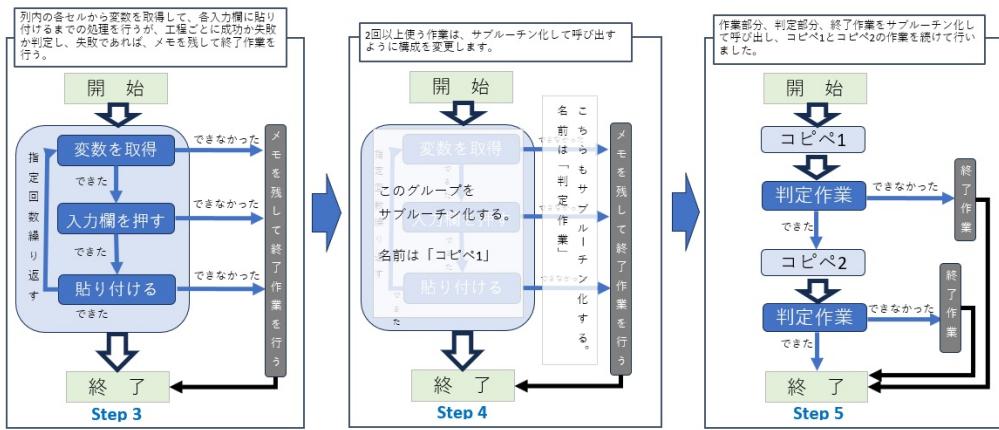
それでも私は、助言をくださったお師匠を尊敬しておりますよ。はい、心から。ちなみにこれは、くれぐれも誠に本当に絶対になにがなんでもナイショですけどね。

12.12 分解して再構成する。

各作業が最適であるかを検討してまいりますよ。プラレールを思い出してくださいね。開始から終了まで、まずはまっすぐのレールを引きます。次に分岐のふたまた等のパーツを準備して、カーブを組み合わせたり段差をつけたりして別のルートを作ります。電車はうまく走れそうですか？イ

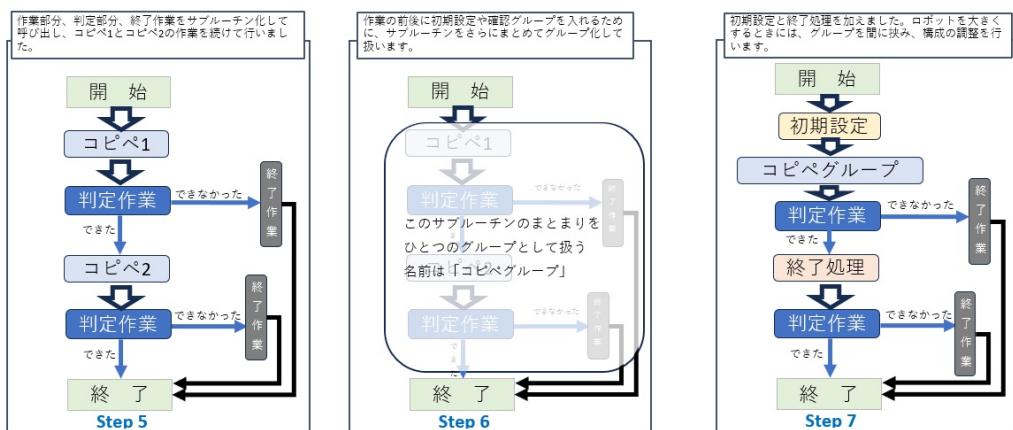
イメージの中では、ちいロボさんが運転手だったりすると楽しめます。それがうまく走ったら、今度は繰り返しの塊、ぐるぐる立体交差をつけたとして、全体のコースイメージを整えます。

図 12.19: 作業の流れ (Step.3~5)



お気づきになりましたか？ロボット/自動化は、つまりプログラム化するということの芯というか軸になる箇所は、ブレーラルのように、ブロック遊びのように、パーツをつないで遊ぶことと違う変わりません。その延長線にあるものです。パーツを選んで、組み合わせて、自分が望む形・動きを目指していくような、皆さんのが子供の頃に遊んだあの経験が生きてきますから、安心して自動化に進んでくださいね。

図 12.20: 作業の流れ (Step.5~7)



12.12.1 メモの追記・追加タイム

さあ、追加メモを作成してみましょう。以下の表は、サンプルですよ。

■会議資料の作成 その4→小型ロボ展開メモ

- ・目的: 会議で使う資料を作成する。→作成した小型ロボを他の業務の効率化に使用したい。
- ・作成した小型ロボについて:
 - ー小型ロボ1. 受信メールから添付資料を探し、解凍して指定のフォルダーに格納する。
 - ー小型ロボ2. 指定のフォルダー内の各ファイルを開いて、必要部分をコピーする。
 - ー小型ロボ3. 資料ファイルにあらかじめ設定した、関数 / マクロを動かす。
- ・小型ロボの追加作成を考える。
 - ー作成済み業務案件リストから、流用できる場面を見つける。
 - ー追加作成する小型ロボについて考える。
 - ー追加ロボも、作成した小型ロボ3種を流用して作る。

ちょっと先の自分が、ちょっと前の自分をもっと好きになれるように、ちゃんとメモを残してあげましょう。自分大切。自分大好き。褒められるの好き。役に立つの好き。このうちふたつは、ロボットも人間もきっと同じなのではと思うことがあります。

12.13 分担を決める。

ロボットと協業する内容とスケジュールを決めます。お時間のこともそうですが、内容も分けていきます。なにせ、ちいロボさんなので、100%全部、ひとりでこなせるわけではありません。作業の分担を決めていきましょう。特に、最初のうちは、人間の目で必ず確認作業を行ってあげてください。

ここでひとつ、覚えておいていただきたいことがあります。ロボットは必ず命じられたことを行います。命じられていないことは行えません。だから、ロボットを動作させて、その結果が希望通りではなかった、あるいはまちがっていたとしたら、あなたがどこかで間違った命令を出しているということです。

悪いのは、ロボットではありません。なぜならロボットは自分で判断できません。命じられたことを行っているだけです。だから、うまくいかなかつたときには、意地をはらずに、ロボットにごめんなさいをして、部品と設定や手順を見直しましょう。ご存じのように、努力は決してあなたを裏切りません。が、間違った努力は失敗につながります。もちろん失敗が次の成功への土台となることもあります。ロボットも、努力と同じように扱ってあげてください。

12.13.1 メモの追記・追加タイム

さあ、追加メモを作成してみましょう。以下の表は、サンプルですよ。

■会議資料の作成 その5→小型ロボとのタイムスケジュールを考える

！エラーは必ず発生すると考える。発生時は手動に切り替えて作業を完遂する。！

PCで一般回線を使用するロボット/自動化は、専用機で専用回線を使う従来のシステムに比較し、安定性に勝るとは言い難いことを意識する。

動作スケジュールを考えてみる。

- ・管理ツール導入前と導入後で使用できる時間帯が異なる

—導入前: エラー発生に備え、基本的に、定時内に行う。

- ・エラー発生時に、即時手動に切り替え、フォローを行うため。

- ・ロボット故障時には、手動操作に切り替えて遅滞なく作業を行なうことが大切。

—導入後: 深夜零時（サーバー停止の1時間前）まで使用可能とする。早朝6時からOK。

- ・エラー発生時のフォローもロボットが行えるよう調整する。

- ・エラー発生時には、ログを残して停止する。

- ・停止した際は、動作をエラー発生前までに戻し、動かす前の状態に戻す。

- ・通勤時: 離席の際（昼食時、会議時）に、ローカル上で動作させるのはOK（課長確認済）。

—ディスプレイを切っておく必要あり。

—セキュリティーとの兼ね合いがあるが、スクリーンは外しておく。

—マウスを少しづつ動かしてスクリーンが出ないよう専用小型ロボが必要。

処理しなければならない情報の量は、日々、増加していきます。来月にも、そのメモは、ちょっと先にいる自分への大事な宝物に変わります。小さなことでも書き加えていきましょう。ちょっとしたことを書くことや、書いた内容を恥ずかしいと思うのは、ずっと先、ずっと大人になったら、その頃にお願いします。

12.14 ツールを使い自動化/ロボット組み立てを行います。

自転車に乗れるようになるまでは、皆さんも何回も転ばれたかと思います。ロボットを動かせるようになるまでには、どのRPA/自動化ツールを選ばれたとしても、やはり、何回も転ばれるかと思います。私も、専門用語で「こけ」つ、ツールの組み立て方を覚えるのに「まろび」つ、悪戦苦闘しながら身に着けました。数十年ぶりに単語帳を使って用語を覚え、「いまさら受験生かよ」とのお褒めの言葉を周囲から頂戴し「人間、死ぬまで勉強です」と笑顔でお応えしたものですわ。

そういううちに、さらに設計でも派手に転び、組み立てですりむき、エラー対策で青あざをいくつもこさえました。ですが、それも最初のうちだけです。ちいさくとも、ひとつのロボットが動くようになる頃には、転びにくくなります。転んでも痛くなくなります。そういううちに、転ばなくなってきます。だから、きっと、だいじょうぶです。安心して、前進していきましょう。匍匐前進でもOKです。前へ前へ前へ。

図 12.21: 簡単にできる、と誰でもできるは違います



さて、組み立てのコツは、というよりも、お仕事のコツになりますかね？ それは得意パターンを持ち込んで、落とし込むことです。必勝パターンを編み出して、それで大枠を作つておいて、調整を繰り返して動きをよくしていく。これが私のロボット/自動化作成パターンです。

パターンを作つておくことで、RPAツール変更時の対応も楽になります。現実に組み立てる際には、それぞれのRPAツールにある程度習熟する必要がありますが、基本パターンに沿つて、必要な順番で学ぶことで、効率のいい覚え方ができます。

ただし、固有の部品や必要な構成というものは、各RPAツールで異なりますので注意が必要です。RPAツール変更時には、どの部品の組み合わせで自分の得意パターンが再現できるか、部品の性質や機能を見極めて、ツールにあわせた黄金パターン再組立てをお試しくださいね。

それでは次に、成功例、私が使つてきた具体的な構成の例を申し上げますね。

12.15 普及型ローコードRPAツールに有効な共通の構成

WinActor、UiPath、PowerAutomate/Desktop、それらを使って自動化を組み立てる際に、共通して有効な方法、考え方、構成があります。部品が多く使われる中規模以上向けになります。先々でお役立てくださいませ。もちろん初号機から使っていただくとさらによきかなでございます。というわけで、おすすめは次の通りです。

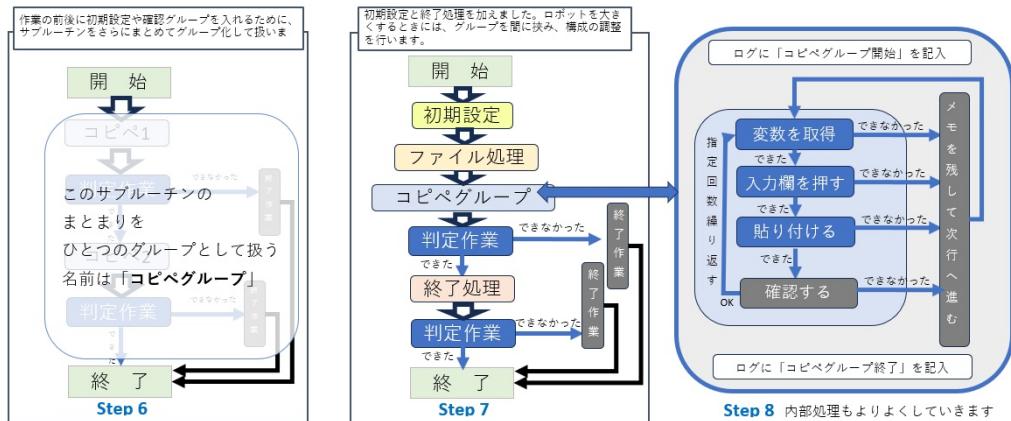
12.15.1 ロボット内に、雑形とするグループ構成および、その中にいる定型部品を格納

1. 初期設定のグループ 定数および変数読み込み、時刻設定等
2. ファイル処理のグループ 存在確認、読み出し、書き込み等
3. 処理本体（大きさによって変わる/入れ子構造で複数入る）

4. サブルーチン群

5. 終了処理のグループ

図 12.22: 作業の流れ (Step.6~8)



- ・据付のログだけに頼らずに、専用ログも準備する。

作業が進行中、グループやサブルーチンの入出時に一行メモ（グループ名と、入/出、あるいは開始/終了等）を書き込むためのログを別途、作成します。組み立てを行う際に、どこまで進んだがエラーが発生したかを確認できます（ロボットは内部速度が早いので、実際のエラー地点よりちょっと前に対応すべき問題が存在しますよ）。

- ・2回以上繰り返す作業は、サブルーチンとして切り出す。

切り出したサブルーチンは本体から都度、呼び出して使うようにします。とはいっても最初はどこからどこまでをサブルーチンにするかを判断するのは難しいです。そこで、全部作ってしまって、ここがダブっているな、という箇所を抜き出します。これをグループ化して、サブルーチンとして作成します。

サブルーチンとして切り出すことで、前後にある処理の重複が避けられます。全体のサイズ（部品数）も縮められますし、重複する箇所の分だけエラー発生も防げます（エラーは必ず出るものなので、あらかじめエラー対策の準備をしておくことはとても重要なことです。できるだけあちこちで手を打っていきましょう）。

12.15.2 ロボットとその資料を入れておく定型のファイル構成を決めておく

- ・① 資料置き場 設計図、マニュアル格納用フォルダー

—上司説明や引継ぎ時に使用するものを格納しておく

—そのロボット/自動化に関する情報はすべてここに収める

- ・② 作業用フォルダ一群

—仮置き用 稼働する際の作業時に一時的に格納しておく場所(作業内容により複雑化する場合は複数必要な場合あり)

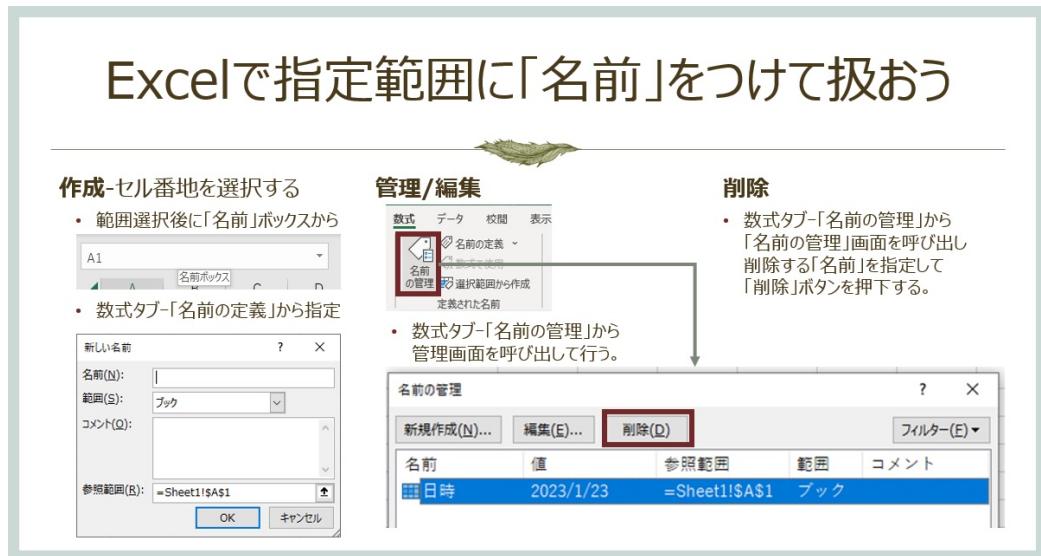
- 一転記元資料配置場所 所定の場所から情報をコピーし一時保管
- 一転記先資料配置場所 必要情報をペーストし所定の場所へ移動する
- 一なお、初期設定と狩猟処理で両方とも空にする、フォルダーに残さない
- ・③ テンプレート格納用 表の雛型置き場
 - 一必要情報をまとめるためにテンプレート（枠組み）を作成しておく
 - 一保存せず中間処理に使用する表の雛型を数推作成しておく
- ・④ 構築作業時の作業用 ロボットの部品置き場 / 終了時に削除する
 - 一ロボット組み立て時に途中作成の部品を預かっておく場所
 - 一素体組み立て時にも、流用できる部品が見つかったら仮に保存しておく
 - 一数種類の部品で比較実験するときなどに候補を待機させておく等に使用

12.16 Excelはお好きでしょう？ もう少し話しましょう。

組み立てる際に、Excelの助けを借りると自動化のハードルが下がります。Excelのブックに変数表や、よく使う文言、行事の日付、本支店リストと連絡先をまとめておくと便利です。詳細はまた別の機会にお話ししますが、ほんのさわりだけお伝えしますね。

Excelには、連続したセル範囲や表等の範囲を指定して「名前」をつけて扱うことができます。もちろん、ひとつのセルにも名前がつけられます。これを応用してロボット/自動化を楽に作成する、横展開に備える等を行うことができます。

図 12.23: Excel の名前つけ



Excelのブックをひとつ眺えていただき、そこに先ほどの「名前」の機能を用いて、変数の一覧、フォルダーのURL一覧、ファイルの名前一覧、よく使う社内行事の情報や、上期下期の異動、定期的な会議や社内行事、緊急連絡先リスト、よく使う文例、フォーマット、テンプレート等をまとめて

おき、ロボットから引用しやすいように整えましょう。さらに関数やマクロを組み合わせると、強力なお助けツールとなります。

図 12.24: Excel でお助け帳を

Excelであなただけの「お助け帳」を作ろう



A列	B列	C列	入力欄	出力値
・役割は名札： 視認性を高めるため にB列につけられた 名前を明記しておく 表札の役割を持た せる。	・役割は本体： 数式または値が入 る。セルには名前を 付ける。	・役割は備考： 説明文をつける。	数式で使われる値 が入る。「設定」欄。	値または関数の結 果が入る「取得」欄

たとえば、あるシートにCSVリストを貼り付けると、別のシートの指定欄に自動的に張り付ける、といった仕組みは比較的簡単な関数で行うことができます。貼り付けた指定欄に、さらに関数を埋め込んでおくことで、更に様々な働きが見込めますね。

このExcelブックは、部内で共有する、あるいはコピーして横展開することで、部内のExcelが得意でない方にも使っていただけます、ロボット作成者を増やすことへ向けてのハードルを低くすることができます。最終的には部内の複数名、欲を言えば誰でもが簡単なロボットを修正することができるようになることを当面の目標となさってください。簡単なロボットを作成し、必要に応じて変更し、壊れたら修理ができる、そのようなチームができれば、これまでできなかった仕事を目指せます。

図 12.25: 作りはじめましょう



Excelのみならず、WordやPowerPointでも同様に、横展開できるテンプレートやフォーム、お助けツールを作成することで業務の省力化を図れます。こちらもあわせてご検討ください。

次のシートは、お助けExcelツールのサンプルです。ロボット内には、専用部品が設けられていますが、それを使って作成するよりも、Excelの関数を使って作成するほうがハードルが低いです。もし、RPA作成ツールが変更になったとしても、このExcelは新しいツールからでも読み込みますので、一度作成しておけば、その点でも安心です。

図 12.26: Excel でお助け帳サンプル

A	B	C	D	E	F
名前	数式履歴セル書式（日付）	セル書式 (日付/0埋め)	B列の数式内容	説明	新山某 謹製 備考
2 今日の日付	2023/1/22	2023/01/22 =TODAY()	本日の日付。	基準日なのでB2セルは変更しない	
3 今日の日付_和暦	令和5年1月22日 -	=今日の日付_シリアル値	セル書式の日付-和暦から設定。		
4 今日の日付_シリアル値	44948	2023/01/22 =TODAY()	本日の日付のシリアル値。		
5 今年_西暦	2023	1905/07/15 =YEAR(今日の日付_シリアル値)	シリアル値から年を取り出す。		
6 今月	1	1900/01/01 =MONTH(今日の日付_シリアル値)	シリアル値から月を取り出す。	数式の設定上、B列のセル書式を標準に	
7 今日	22	1900/01/22 =DAY(今日の日付_シリアル値)	シリアル値から日を取り出す。		
8 来月の今日	2023/2/22	2023/02/22 =EDATE(今日の日付_シリアル値,1)	西暦_来月の本日のシリアル値。	セル書式を日付に戻しています。	
9 来月の今日_曜日	令和5年2月22日水曜日	2023/02/22 =EDATE(今日の日付_シリアル値,1)	和暦_来月の本日のシリアル値。	セル書式をユーザー定義で変更している。	
10 今月初日の日付	2023/1/1	2023/01/01 =EOMONTH(今日の日付,-1)+1	今月初日の日付	EMONTH関数は元の日付から求める日	
11 今月末日の日付	2023/1/31	2023/01/31 =EOMONTH(今日の日付,0)	今月末日の日付	に使用する。()の中は、まず基準となるセル番地または名前、次に当月0から	
12 先月初日の日付	2022/12/1	2022/12/01 =EOMONTH(今日の日付,-2)+1	先月初日の日付	1月を求めるのか、1や+1などの整数で	
13 先月末日の日付	2022/12/31	2022/12/31 =EOMONTH(今日の日付,-1)	先月末日の日付	EMONTH関数は元の日付から求める日	
14 今年度初日の日付	2022/4/1	2022/04/01 -	B列手入力	に使用する。()の中は、まず基準となるセル番地または名前、次に当月0から	
15 今年度末日の日付	2023/3/31	2023/03/31 -	B列手入力	1月を求めるのか、1や+1などの整数で	
16 今年度前期最終日の日付	2022/9/30	2022/09/30 -	B列手入力	よく使う日付を保存しておき、必要ある。	
17 今年度後期最終日の日付	2023/3/31	2023/03/31 -	B列手入力		
18 対象日の入力欄	1999/7/14	1999/07/14 -	B列手入力	取得する年月日を入力する。	
19 対象日のシリアル値	36355	1999/07/14 =対象日の入力欄	対象日のシリアル値。		
20 表示形式の変更	平成11年7月14日水曜日	1999/07/14 =対象日の入力欄	対象日の和暦表示		
21 和暦_年	1999	1905/06/21 =YEAR(対象日のシリアル値)	シリアル値から年を取り出す。	上記年月から求める形式で表示を行	
22 和暦_年	平成11年	1999/07/14 =対象日のシリアル値	シリアル値から年を取り出す。		
23 和暦のみ	平成	1999/07/14 =対象日のシリアル値	シリアル値から年を取り出す。		
24 月を取り出す	7	1900/01/07 =MONTH(対象日のシリアル値)	シリアル値から月を取り出す。		
25 日を取り出す	14	1900/01/14 =DAY(対象日のシリアル値)	シリアル値から日を取り出す。		

さて、後日、このあたりを含めてExcelをDX推進の心強い味方にする方法について、詳しくご説明いたしますので、どうぞお楽しみに。

図12.27: やっぱり心にノートを持とう



やっぱり心に ノートを持とう

実は不肖新山は、心に別のノートを秘めてもあります。こちらのノートはですね、何か素敵なことを見つけるたびに、書き込むためのノートです。誰ですか、ネタ帳とか言ったのは。正鶴を射抜きすぎて、おばちゃん泣いちゃいますよ？という冗談は置いておいて、こちらのノートの用途はですね、素敵な人と出会った可愛いものとかを書き込んでいくのです。例えば、駅前の手作りパン屋さんのうまパンのにおい、すれ違った手のひらに乗るような子犬の散歩、お婆ちゃんに席を譲るyanキーの兄ちゃん（そしてそのあと車両を変えてた）、受付の華道部の力作（えらいぞ、お当番）、育児休暇あけの先輩から香る赤ちゃんのにおい、お使いに出かけた先で雨がやんで見上げた空にかかった薄い虹、そんな日常の様々な小さな幸せを心のノートに書きこんでいきます。

そして寝る前に、心の中でノートを広げて、小さな幸せをくれた皆様に感謝とともに首に金メダルをかけて差し上げるのであります。

「どうか、みなさんのもとにも小さな幸せがたくさん集まりますように」と祈りながら眠ると、ぐっすり眠れますよ。

12.17 終わりに これからはロボットと二人三脚で

種明かしをしましょう。どうして、「普通のOLなら業務に役立つロボットを必ず作れます」とタイトルに書いたのか。デジタル化もRPAも、そしてCX、DXさえも昭和の御代から受け継がれ、使い古された「改善」のお色直です。それをきちんと立たせて動かして、一緒に働いていくためには何が一番必要ですか？

言うまでもなく現場の知恵ですね。今々に足りないものがあり、あるいは間違っているものがあり、または無駄なものがあることに気づくことが改善の始まりで、それは現場に居合わせなければ気が付かないもの。現場こそが、DX/RPAの鍵を握っています。実務、実作業を行っている皆様こそが、上司の誰よりも、本当のお仕事を、お仕事の現実を知っている。そのことは、自動化/ロボット作成作業の半身です。それがわかっていればこそ、改善ができ、自動化/ロボット作成作業が生きてきます。外注に出したすぐに錆付き壊れてしまうロボットではなく、改修しながらずっと生きて、仕事を助けてくれるもの、あなたの代わりに単純な繰り返し作業を行い、あなたに「ずっとやりたかった仕事に取り組む」時間を与えてくれるものです。

私が一番最初に作成したロボットは、2018年にコンサルさんに相談しながら作成したWinActorのロボットです。このロボットは、私がその現場を去った後に改修を重ね、さらにそのロボットをベースに横展開され、今でも働き続けています。面白いでしょう？自分が作ったロボットに兄弟や子供や孫ができる、その子たちが社会の一端を支えているのです。考えるだけでワクワクしませんか？

次は、あなた様の番でございますよ？ いけませんよ、他人事みたいな顔をされちゃあ。明日といわず、今日から、小さなロボットを、ちいロボ君を作り始めましょう。さあ、たったひとつのちいさなロボットから、大きな世界征服をはじめましょう（違）

See you Next Book!

DXはスマートスタートからはじめましょうね

図 12.28: DX とは

 **DXとは**

DXの定義*

企業がビジネス環境の激しい変化に対応し、データとデジタル技術を活用して、顧客や社会のニーズを基に、製品やサービス、ビジネスモデルを変革するとともに、業務そのものや、組織、プロセス、企業文化・風土を変革し、競争上の優位性を確立すること

ポイント

<u>データとデジタル技術を活用して</u>	… デジタルツールの導入 = DXではなく、データやデジタル技術はあくまで変革のための手段
<u>製品やサービス、ビジネスモデルを変革するとともに、業務そのものや、組織、プロセス、企業文化・風土を変革し</u>	… デジタルを使った製品やサービスを提供するだけでなく、データやデジタル技術を活用したプロセスの改善や、デジタルを活用しやすい組織づくりへの取り組みが必要
<u>ビジネス環境の激しい変化に対応し／競争上の優位性を確立する</u>	… 環境変化の中でも、企業が市場で淘汰されずに、成長し続けることが目的

*出所：経済産業省「デジタルトランスフォーメーションを推進するためのガイドライン（DX推進ガイドライン）Ver1.0」（平成30年12月）

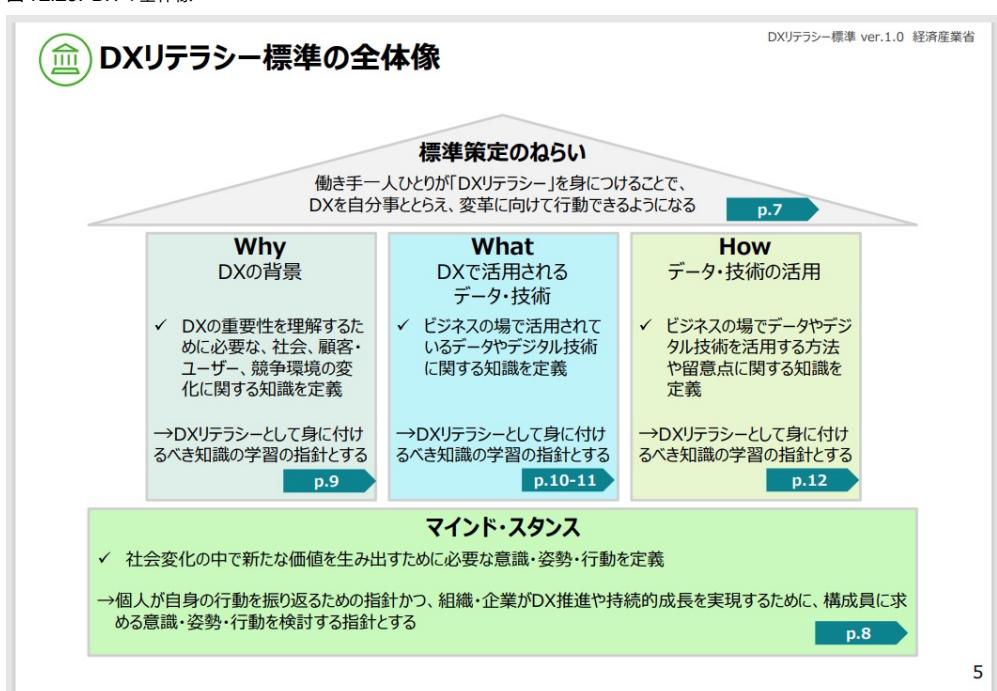
出典: DX リテラシー標準 ver.1.0 経済産業省

https://www.meti.go.jp/policy/it_policy/jinzai/skill_standard/DX_Literacy_standard_ver1.pdf

経済産業省のDXリテラシー標準によりますと、DXの定義とは「企業がビジネス環境の激しい変化に対応し、データとデジタル技術を活用して、顧客や社会のニーズを基に、製品やサービス、ビジネスモデルを変革するとともに、業務そのものや、組織、プロセス、企業文化・風土を変革し、競争上の優位性を確立すること」だそうです。勇ましいわ。なので、ロボットを作って効率化するだけではDXではないみたいです。それによってビジネスモデルを変革するところまで、もっていかなければならないのですね。

ウチのちいロボさんたちにできるかしら。ちょっと心配ですね。引き続きフォローと改良を続け、超合体ちいロボさんMAX ラックスの作成による大幅なコスト削減とビジネスモデル改革を目指して進んでまいりましょうか。まずはあなた様の最初の一歩、最初の小さなロボットから始めましょう。

図 12.29: DX の全体像



5

出典: デジタルスキル標準 経済産業省

https://www.meti.go.jp/policy/it_policy/jinzai/skill_standard/main.html

あらあら、先は長そうですね。ですが、何事も「千里の道も一歩より」なのですよ。私たちは、ちい口ボさんを作って作業を効率化し、その先にある何かを目指して手に手を取り合って協力して進んでまいりましょう。この一歩は小さいけれど、DXにとっては、きっと偉大な一歩ですわ。

うれしい無料講座もある「マナビ DX」

経済産業省では、DX人材育成に向けて、すべての講座が無料ではありませんが、一部無料の講座を提供しています。様々な講座が無料で学習できますので、どうぞご活用くださいませ(2023年7月24日の時点で、全講座404件、無料講座は97件ございます)。

まずはあなた様からご受講いただき、次に社内に展開する流れではいかがでしょうか。効率よく皆さんでスキルを取得できそうですね。この無料提供された講座郡から、有効な講座をいくつか指定して社内へ推薦＆共有、さらにスキル取得用のイベント開催でブーストをかけられそうではありませんか。

夏休みのプールやラジオ体操のカードのように、終了ごとにハンコをもらうシステムを取り入れてはどうでしょうか、皆勤賞や優秀賞は何がいいですかね。案として、私はおやつ券案を推しますが、たとえば、自社のイントラネット、クラウド上でのみ使えるような特別な称号やアバター等はいかがでしょうか。講座でスキルを獲得する際に、なにか嬉しいご褒美があると、社員の皆さんのスキル育成スピードが上昇するかもしれませんよ。スキル獲得 → ご褒美(社内認知) → さらにスキル獲得 → (以下省略) の無限コンボを現状へ叩き込むことを、ご検討くださいませね。

図12.30: マナビ DX

The left screenshot shows a search result for 'マナビDXでの学び方' (How to learn with Manabi DX) with a count of 97 items. It includes filters for 'スキル' (Skills), '講師' (Instructor), and '料金' (Price). Below the filters, there are several course cards with titles like 'AI-800 有料実践対策コース' (AI-800 Paid Practical Preparation Course), 'DXリテラシー基礎' (Basic DX Literacy), and 'はじめてPythonで実践自動化' (Practical Automation with Python).

The right screenshot shows a detailed view of a course titled 'DXリテラシー基礎講座' (Basic DX Literacy Course). It includes a brief description, a thumbnail image of a person, and a button labeled '登録する' (Register).

出典: マナビ DX

<https://manabi-dx.ipa.go.jp/courses>

図12.31: マナビ DX 無料講座の検索結果

The left screenshot shows a search result for 'DXリテラシー基礎' (Basic DX Literacy) with a count of 97 items. It includes filters for '料金' (Price) and '講師' (Instructor). Below the filters, there are several course cards with titles like 'AI-800 有料実践対策コース' (AI-800 Paid Practical Preparation Course), 'DXリテラシー基礎' (Basic DX Literacy), and 'はじめてPythonで実践自動化' (Practical Automation with Python).

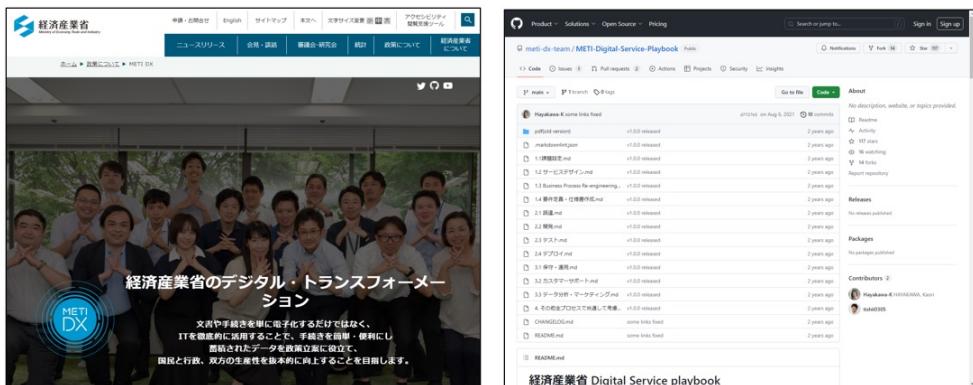
The right screenshot shows a detailed view of a course titled 'ゼロからわかる! DXリテラシー基礎講座' (Learn from scratch! Basic DX Literacy Course). It includes a brief description, a thumbnail image of a person, and a button labeled '登録する' (Register).

無料講座の中には、DXリテラシーの講座はもちろんありますし、その他、PythonやAzureの資格講座、データ分析や統計活用、Google Analytics、AWS等、様々な題材の無料講座で学べます。有料のものでも一部補助が出るものもあるようです。さらには、法人申し込み可能なものもございます。ぜひ、ご高覧の上、ご利用をお考えくださいませね。

経済産業省DX室から愛をこめて「METI DX」

経済産業省のDXポータルサイトです。取り組み、NoteやYoutub、Githubを使用したサービス提供等ございます。こちらは情報源です、将来に向けての、宝の山となるかも知れません。一度、訪れてくださいませね。あなた様のDXに使える部分があるかも。立っているものは親でも使え、ですわよ。輝く明日、円満退職、もしくは快適な老後のために、こちらもあわせて有効活用させていただきましょう！

図 12.32: METI DX



出典: METI DX

https://www.meti.go.jp/policy/digital_transformation/index.html

12.18 次に御目文字できる日を楽しみに励みますわ

それについても、気になっておりますのよ。社員の方全員で生産性向上を本気で考えるときに、見落とされている「言葉の問題」がございますよね。私たちは、母国語・日本語を用いて、象形文字である漢字の組み合わされた熟語から、類推することで意味をつかむことができます（明治の近代化の際に、先人の皆様がそのように組んでくださったからですね。感謝）。

本来は、PCにまつわる諸々も、カタカナではなく漢字（の組み合わせ）を用いるべきなのですが、この分野は翻訳が追いつかないのが現状です。英語を日本語に置き換えることが間に合わず、カタカナのまま置き換えるだけに留めてきました。その結果、文中のカタカナ率が非常に上昇してしまい、ある程度の年齢から上の層、主に経営陣、会社の舵取り層が、生産向上に必要なRPA,CX,DX等を理解しがたい状態にある、という状態がこの30年ほど続いています。私たちは、自分が知っていることは相手も知っているだろう、ということで、何事も省略しがちなのですが、これが裏目に出続けたこの30年ではありますのよ。これが続くと、会社も社会も傾きますわよね、どう考えをいたしましても。

そうですね、現状打開のためにChat GPT + CATツール + CX/DXを活用することで、このあたりの深刻な溝を短期間で強力に、埋め立てることができるのではないかと、最近、思う次第でありますよ。なお、CATツールと言っても、町内の猫さんたちの手をお借りするわけではなく、Computer Assisted Translationを略してCAT、Trados等の翻訳支援ツールです。

専門用語の難解さに加え、理系ならではの表現の難解さという問題も存在します。たとえば、ご本が大好きな文系に大学の数学の教科書を渡しても、意味を取って理解するのは至難の業ですので（実験）。しかしながら、表現に関しては、初心者むけには、平易な表現で専門用語を解説してくれる佐々木真さん提供の素敵なサイト[「分かりそう」で「分からぬ」でも「分かった」気になるIT用語辞典]から、お願いして表現をお借りするですか、あるいは公募するとかで文

系にもわかりやすい表現を集めるなど細かく工夫すれば、道が開かれるのではないかと夢見る次第です。※

かつて明治期の先人が、近代的な西洋の概念を和風の熟語へと移行し、近代化に貢献したように、この令和の御代でChat GPT + CAT ツール + CX/DX を用いて概念の再編成を行えると素敵ですね。ビジネス用語に入り込みすぎたカタカナ群を和風に戻し、高年齢層にも理解できるようにすることが行えるのではないかでしょうか。

もし、それができたならば、後期高齢者から小学生まで、共通の認識を持って未来に向かって建設的な会話をすることができますわね。世代を超えて新しい技術に関して話しあうこともできるようになるのではと思うと、わくわくしてきませんか？あらいやだ、お笑いにならないでくださいね。大人にだって未来も夢もありますのよ、OL31年生だって、将来を夢見てもよろしいでしょう？いつかこのあたりのお話の続きが、あなた様と楽しくできますように。

ではまた、近いうちにどこかで、きっとお会いいたしましょう。それまでどうぞお健やかに。あら、どなたが何をおっしゃろうとも、あなた様には笑顔が一番似合いますわよ。これから日々もご自愛第一に、どうぞ笑顔でお過ごしくださいませね。

図 12.33: さいごに



出典: 佐々木 真さんのサイト

<https://wa3.i-3-i.info/>

「「分かりそう」で「分からぬ」でも「分かった」気になれるIT用語辞典」

NextPublishing Sample

著者紹介

北崎 恵凡 (きたざき あやちか)

趣味でモノづくり活動やコミュニティー「野良ハック」や技術書の執筆や月刊I/O(工学社)、シェルスクリプトマガジン(USP出版)などへの寄稿を行う。共著書に「Jetson Nano超入門」(ソーテック社)、「現場で使える！Google Apps Scriptレシピ集」(インプレスR&D)、「図解と実践で現場で使えるGrafana」(インプレスR&D)、「はじめてのNode-RED MCU Edition」(工学社)がある。コミュニティーではオンライン・オフライン・ハイブリッドイベント企画・運営・配信・アーカイブ編集/管理支援を担当。

執筆者紹介

山田 雄一 (やまだ ゆういち)

Slerでの受諾開発SE、社内SE(内製型)を経て、現在はサポートエンジニア。技術同人誌を中心とした技術書の執筆や、たまに月刊I/O(工学社)のライターだったりもする。(主に近畿圏近郊のイベントレポートを担当) 共著書に「エンジニアのための見積もり実践入門」、「エンジニアのためのオンライン生活ガイドブック」、「エンジニアのための目標設定の技術」、「エンジニアのためのカジュアル面談のトリセツ」、「積み基板を作らないための電子工作入門」などがある。(全てditflame名義 インプレス刊)

鎌田 誠 (たきがわ だいき)

地元工場の情シスを20年経験した後、現在はその経験を活かして、名古屋を拠点にネットワーク系の技術営業に従事。お客様の課題解決等を行う。最近はコミュニティー活動に積極的に取り組んでおり、RPA Communityではライティングトーク支部の主催。Babylon.js勉強会では「Babylon.jsレシピ集vol.1」(インプレスR&D)及び「Babylon.jsレシピ集vol.2」の執筆も一部担当。IT知識は広く浅くをモットーに日々精進。

瀧川 大樹 (たきがわ だいき)

学生時代は社会情報学を専攻、推薦アルゴリズムについて研究。強調フィルタリングとコンテンツベースフィルタリングを組み合わせた独自のアルゴリズムを考案、LAMPで実装。また、研究の傍らデータセンターのアルバイトでサーバーの保守運用業務を経験。2014年より通信会社にてサーバー/アプリケーション保守運用業務に従事。保守運用業務に注力する傍ら、情報処理安全確保支援士の資格を取得し、システムのセキュリティー管理業務にも従事。

青木 敬樹 (あおき ひろき)

学生時代は機械工学を専攻。三次元画像の特微量抽出について研究。2021年に某通信会社に入社。入社後はサーバー・インフラの保守運用業務を担当。社内の「プロアクティブな運用」の推進チームの一員として活動し、運用可能な自動化を実現するために日々勉強中。

大野 泰歩 (おおの やすほ)

学生時代は機械学習の中でも強化学習という分野について研究。RoboCup 2D Soccerに触れておりました。2020年に通信会社に入社し、サーバー・インフラの運用保守業務に従事する傍ら、運用保守業務を自動化するためのシステム開発にも関わっておりました。

松岡 光隆 (まつおか みつたか)

株式会社コミュカル 代表取締役CEO。元ITエンジニア、複数のIT系Webメディアでライター・編集業も経験。エンジニア時代に、ユーザー側の立場として200回以上のコミュニティーイベントへの参加、50回以上の登壇を経験。その経験をベースとした独自のコミュニティー運営論を展開し2021年に株式会社コミュカルを起業。2023年現在では20以上のコミュニティーを同時に企画・運営する、コミュニティーのスペシャリスト。同時に約20社のコミュニティーを支援しながら、常に様々なコミュニティーへの参加を継続しており、年間約200回コミュニティーイベントに参加、年間約100回のコミュニティーイベントを主催運営、年間10回以上の登壇を続けている。

新山 実奈子 (にいやま みなこ)

OL 31年生、RPA 6年生。職歴はロボットアニメ背景作画に始まり、事務ロボット作成に至

る。主にWinActor、UiPath、Power Automate（Desktop）で業務自動化のお手伝いをしています。導入支援から開発、納入、教育までお役に立ちますよ。かくし芸としてhtmlを読んだり書いたりできる旧CIW Website Designerです。モットーは、"Que Sera, Sera"。絵本のようなマニュアル作成を企画中。

高井 美佑（たかい みゆう）

株式会社ソントレゾ所属。RPACommunity ライトニングトーク支部 / Power Automate 支部 主催。元・事務職のPower Platformエンジニアです。Power Platformの教育・開発支援・コミュニティー活動支援に従事しています。モットーは、「楽しく学んで、よりラクで明るい未来を作る」。

小崎 肇（こさき はじめ）

フリーランス。COBOLから始まった30年以上のエンジニア経験を経て、Excel-VBAのノウハウと共にUiPathを使った業務効率化プロジェクトに参画。RPACommunityでの登壇をきっかけに、運営側としても活動。定年退職後もUiPath開発者として日々精進。RPA界隈を脈やかしている中では、最高齢ではないかと咲いています。座右の銘は「人間万事塞翁が馬」。

瀧谷 匠（しぶや たくみ）

株式会社ASAHI Accounting Robot研究所 テクニカルエバンジェリスト。一般社団法人中小企業個人情報セキュリティ推進協会認定 DXアドバイザースペシャリスト。プログラミング知識ゼロの状態からRPA/AI-OCRの技術・知識及び概念、全社展開の導入スキーム等を独学で勉強。モットーは "Make everyone's work easier. (全ての人の仕事を楽にする)"。

◎本書スタッフ

アートディレクター/表丁：岡田章志 + GY
編集協力：山部沙織
ディレクター：栗原 翔
〈表紙イラスト〉
■ ■ ■ ■

技術の泉シリーズ・刊行によせて

技術者の知見のアウトプットである技術同人誌は、急速に認知度を高めています。インプレス NextPublishingは国内最大級の即売会「技術書典」(<https://techbookfest.org/>)で発行された技術同人誌を底本とした商業書籍を2016年より刊行し、これらを中心とした『技術書典シリーズ』を展開してきました。2019年4月、より幅広い技術同人誌を対象とし、最新の知見を発信するために『技術の泉シリーズ』へリニューアルしました。今後は「技術書典」をはじめとした各種即売会や、勉強会・LT会などで発行された技術同人誌を底本とした商業書籍を刊行し、技術同人誌の普及と発展に貢献することを目指します。エンジニアの“知の結晶”である技術同人誌の世界に、より多くの方が触れていただくきっかけになれば幸いです。

インプレス NextPublishing
技術の泉シリーズ 編集長 山城 敏

●お断り

掲載したURLは2023年8月1日現在のものです。サイトの都合で変更されることがあります。また、電子版ではURLにハイパーリンクを設定していますが、端末やビューアー、リンク先のファイルタイプによっては表示されないことがあります。あらかじめご了承ください。

●本書のご感想をぜひお寄せください

<https://book.impress.co.jp/books/xxxxxxxxxxxx>

アンケート回答者の中から、抽選で図書カード（1,000円分）などを毎月プレゼント。

当選者の発表は賞品の発送をもって代えさせていただきます。

*※プレゼントの賞品は変更になる場合があります。

●本書の内容についてのお問い合わせ先

株式会社インプレス

インプレス NextPublishing メール窓口

np-info@impress.co.jp

お問い合わせの際は、書名、ISBN、お名前、お電話番号、メールアドレスに加えて、「該当するページ」と「具体的な質問内容」「お使いの動作環境」を必ずご明記ください。なお、本書の範囲を超えるご質問にはお答えできないのでご了承ください。

電話やFAXでのご質問には対応しておりません。また、封書でのお問い合わせは回答までに日数をいただく場合があります。あらかじめご了承ください。

インプレスブックスの本書情報ページ <https://book.impress.co.jp/books/xxxxxxxxxxxx>では、本書のサポート情報や正誤表・訂正情報などを提供しています。あわせてご確認ください。
本書の奥付に記載されている初版発行日から3年が経過した場合、もしくは本書で紹介している製品やサービスについて提供会社によるサポートが終了した場合はご質問にお答えできない場合があります。

奥付サンプル

201X年X月X日 初版発行Ver.1.0 (PDF版)

著 者 × × × ×
編集人 × × × ×
発行人 × × × ×
発 行 株式会社インプレスR&D

〒101-0051
東京都千代田区神田神保町一丁目105番地
<http://nextpublishing.jp/>

●本書は著作権法上の保護を受けています。本書の一部あるいは全部について株式会社インプレスR&Dから文書による許諾を得ずに、いかなる方法においても無断で複写、複製することは禁じられています。

©201X, All rights reserved.

ISBN978-4-XXXX-XXXX-X



Next Publishing®

●インプレス Next Publishingは、株式会社インプレスR&Dが開発したデジタルファースト型の出版モデルを承継し、幅広い出版企画を電子書籍+オンデマンドによりスピーディで持続可能な形で実現しています。<https://nextpublishing.jp/>

NextPublishing Sample