Go forward with us

RPA TECHNOLOGIES

V1.0.0　2020.02.20

【入門】健康なロボットのための10か条

はじめに

RPAによるロボット開発も通常のプログラム開発と同様、事前に定義された手続きを自動化する取り組みです。

一方でプログラム開発において用途や特性に沿った開発手法やアプローチがあるように、RPAのロボット開発においてもプログラム開発とは異なる、ロボットに適したアプローチが存在します。

本書では、多くの方が長期間安定して利用できるいわゆる「健康なロボット」を作るための指針を10か条にまとめました。

ロボットが使用される状況や用途により10か条に沿わない場合も存在しますが、まずは10か条に沿った評価を行ったうえで、個別の判断をしていただければ幸いです。

健康なロボットのための 10か条

1. ロボットのサイズはコンパクトに保つこと
2. 単一の処理に集中すること
3. ロボットのフローに業務処理の骨格が明確に表れていること
4. 補助処理と本処理を明確に区別すること
5. 同一・類似の処理を複数存在させないこと
6. 用途や内容ごとにデータを整理整頓すること
7. 処理内容の見通しをよくするための案内・コメントを適所に設置すること
8. ロボット実行時の処理経路がトレースできるように適度にログ出力を設定すること
9. 例外処理はログと通知を重視し、自動回復は最低限とすること
10. 環境変数値はロボット内に組み込まず、外部情報の読み込みで切り替えること

健康なロボットのための 10か条（解説）

1. ロボットのサイズはコンパクトに保つこと

ロボット当たりのステップ数上限の目安は 100 ～ 200 程度にしましょう。
ステップ数の上限自体に厳格な数値的根拠はありませんが、ロボットを開いたときに全体が見渡せる範囲であることが、業務部門の非エンジニアを含め長期間複数人でロボットをメンテナンスしていく上でのちょうどいいサイズとなります。

2. 単一の処理に集中すること

人が作業をする場合には複数のことを同時にやってしまった方が（データをチェックしながら入力をするなど）省力化できることが多いですが、ロボットは繰り返し処理や個別処理を嫌がることはありません。
場合によっては人と役割分担したり、例外発生時に素早くリカバリーするためにもロボットへは単純に分解した作業を順次組み合わせて実行（データのチェックを集中して実施した後、正しいデータを集中して入力）させることにより、作成時の業務整理の機会を得るとともに、ロボット自体を簡潔に保ちエラー発生時の影響を低く抑えることができます。

健康なロボットのための 10か条（解説）

3. ロボットのフローに業務処理の骨格が明確に表れていること

ロボットのフローを参照すれば初見でもおおよその業務内容・処理内容が把握できるように全体の手順を設定しましょう。手順の明確でないロボットはステップの並びも雑然としやすく、ロボットの処理品質やメンテナンス性の低下にもつながります。長期間保守・運用する中で無駄を含んだ煩雑なロボットにならないよう、処理の塊ごとにグループ化し、何をやっているのか客観的にわかりづらいステップは整理しましょう。

4. 補助処理と本処理を明確に区別すること

ロボットで代替する業務処理を実行する「本処理」部分と、そのためのデータ変換や入力ファイルの有無確認といった「補助処理」部分がロボットのフローの中で混在せず、前処理・本処理のように配置するパートを区別することにより、ロボットに実装するステップの業務側面における手順が明確になるとともに、処理の前提となるデータの種類や条件の見通しが良くなるため、ロボットにエラーが発生した際にも、その原因がデータにあるのか、プロセスにあるのかの判別がしやすくなります。

健康なロボットのための 10か条（解説）

5. 同一・類似の処理を複数存在させないこと

業務手順をロボット化する過程では一部の処理や値を繰り返し実施・利用することがあります。複数回登場する一連の手続きについては共通部品としてSnippet化し、一か所にまとめましょう。また繰り返しロボット内で使用される数値や文字列についても変数として一か所で管理することで、意図せぬロボット品質の悪化（保守作業における変更漏れやロボットの肥大化の防止）を回避することができます。

6. 用途や内容ごとにデータを整理整頓すること

ロボットの行う業務処理は手順とデータで構成されています。データには入力データ・出力データ・一時データの3種類があり、個々のデータが構造を持つ（“アカウントというデータは、IDとPWの組み合わせから成り立っている“など）場合も多くあります。10か条の前半5つで手順部分に関して述べたのと同様に、データ部分についても用途・内容ごとにTypeを利用してデータを整理することによりロボットの品質を一定に保つことが可能になります。つまり、まず手順を作り始める前に必要なデータの棚卸をしておくことが重要になります。

健康なロボットのための 10か条（解説）

7. 処理内容の見通しをよくするための案内・コメントを適所に設置すること

ロボットは長い期間運用される中で担当者の引継ぎが発生したり、処理内容の変更や追加が行われます。業務の仕方が変わればロボットの見直しも発生します。誰にでもわかりやすいようにロボットをシンプルに保つことはもちろんですが、意味のある処理の塊をGroup化することにより手順の区切りを示したり、入力コードの値や、条件分岐の指定値、Group自体に業務内容を示すコメントを入れることにより、ロボットを通して業務の手順を学ぶ（簡易的な業務マニュアルとしても役立てる）ことも可能になります。

8. ロボット実行時の処理経路がトレースできるように適度にログ出力を設定すること

ロボットの実行結果を後で確認したり、サーバでバックグラウンド処理をしているロボットの処理中に進捗状況を確認することを目的に、Write Log でパンくず(*1)を残しましょう。
ロボットが都度処理するデータのキー情報などデータと手順を紐づける情報を適切にログへ残すことにより、エラーが発生した際の原因究明の効率化にも役立ちます。

(*1) グリム童話「ヘンゼルとグレーテル」の中に登場する。兄妹が森で迷子にならないように、帰り道を示す目印としてヘンゼルが道に落としたパンくずが由来。

健康なロボットのための 10か条（解説）

9. 例外処理はログと通知を重視し、自動回復は最低限とすること

RPAというソリューションの名称から、プロセスの完全自動化のためにエラー発生時の自動ロールバックや自動再実行の処理を実装しようとする場合がありますが、ロボットの運用においては例外を確実にとらえログ・通知をするまでに留め、ロボット内に自動回復のためのロジックは極力含めないことが推奨されます。
処理性能や確実性が高く求められる処理においてはロボットとは別にプログラムを作成し、ロボットと連携させるなど、業務要件を分析したうえで適宜役割分担をすることが、広くロボットを活用するうえで重要な要素です。

10. 環境変数値はロボット内に組み込まず、外部情報の読み込みで切り替えること

データ更新系の業務ロボットは通常、ロボットを実際に動かすのとは別の専用環境を用意して開発やテストを行います。（事故防止のため、本番環境での開発は推奨しません）
ロボットが稼働する環境が本番とは異なることに伴い、ロボットが読込むファイルパスや操作対象システムのURLおよびアカウントなどの環境変数が異なります。ロボットを開発環境から本番環境へ切り替える際、ロボット内部の定義情報を書き換えることは、事前のテストによる品質の担保自体を無効にしてしまいます。実行環境による動作の差異はロボット内の定義情報ではなく、外部情報（外部設定ファイルやデータベース）により管理するようにしましょう。