

第4回今治市DX勉強会 サイボウズkintoneのご紹介

2025年11月27日
サイボウズ株式会社
地域DXディレクター
久保 正明



自己紹介

くぼ まさあき
久保 正明

1968年 : 愛媛県生まれ

1990年 : 地元鉄道会社系の人材サービス会社 入社
プログラマー、システム営業、派遣営業を経て
情報システム部門事業部長などを歴任

2008年 : サイボウズ株式会社入社
サイボウズカスタマーセンター センター長、
ローカルプランディング部長を経て、現在は
地域DXディレクターとして地域のネットワークと
サイボウズの情報共有ノウハウで地域課題解決の事例づくりに取り組む
・四国情報通信懇談会運営委員
・愛媛デジタルデータソリューション協会事務局長
・愛媛オレンジバイキングス取締役

趣味：地域活動（元PTA会長、防災士、お祭り、花園町ブライダル、松山ティクアウト部、
たてヨコ愛媛、松山・野球拳オンラインまつり2020）



会社概要

名 称	サイボウズ株式会社 東京証券取引所 プライム市場（コード4776）
事 業 内 容	グループウェアおよび チームワーク強化メソッドの開発・販売・運用
創 業	1997年8月（愛媛県松山市にて3名で創業）
所 在 地	東京都中央区日本橋2-7-1 東京日本橋タワー
拠 点	東京, 大阪, 松山, 名古屋, 福岡, 仙台, 札幌, 那覇, 上海, 深圳, 台北, ホーチミン, サンフランシスコ, シドニー(合弁)など
業 績	2024年12月期連結：売上 29,675百万円、経常利益 5,335百万円 2024年12月期個別：売上 28,743百万円、経常利益 6,347百万円
従 業 員 数	1,321名（2024年12月末 連結） 1,030名（2024年12月末 単体） ※役員監査役除く無期雇用（正社員）の社員数。執行役員は含みます。

企業理念

存在意義
Purpose

チームワークあふれる社会を創る

文化
Culture

1. 理想への共感

共通の理想を作り、理想に共感して行動する

2. 多様な個性を重視

多様な個性を重視し、互いに活かし合う

3. 公明正大

オープンな信頼関係の
基盤を作る



4. 自主自律

一人ひとりが個人としての主体性を持ち、
よりよいチーム作りに関わっていく

5. 対話と議論

お互いの考えの前提を理解し、論じ合って意思決定をする

主力製品群

中小企業グループウェア



82,000社以上

業務アプリ作成プラットフォーム



38,000社以上

大企業向けグループウェア



8,100社以上

メール共有システム

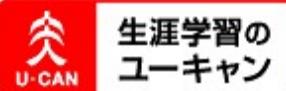


15,000社以上

顧客企業



1,450万人 180,000社以上で、
愛用されています。



グローバル展開中



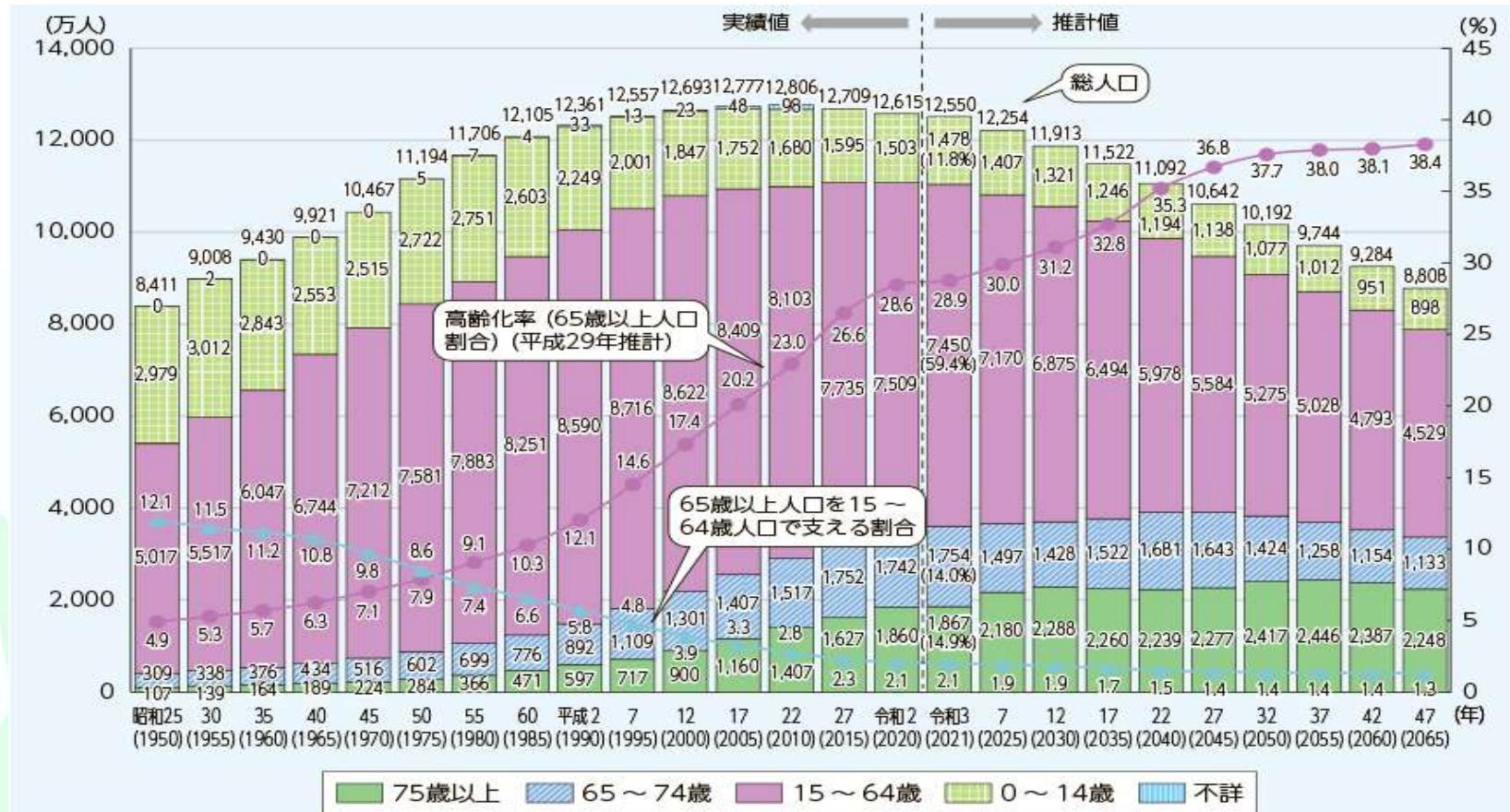
コロナ以降、私たちのライフスタイル・ワークスタイルが大きく変化



世の中にインパクトを与える様々な新技術



高齢化の推移と将来推計（総務省）



DXによる変革が必要！

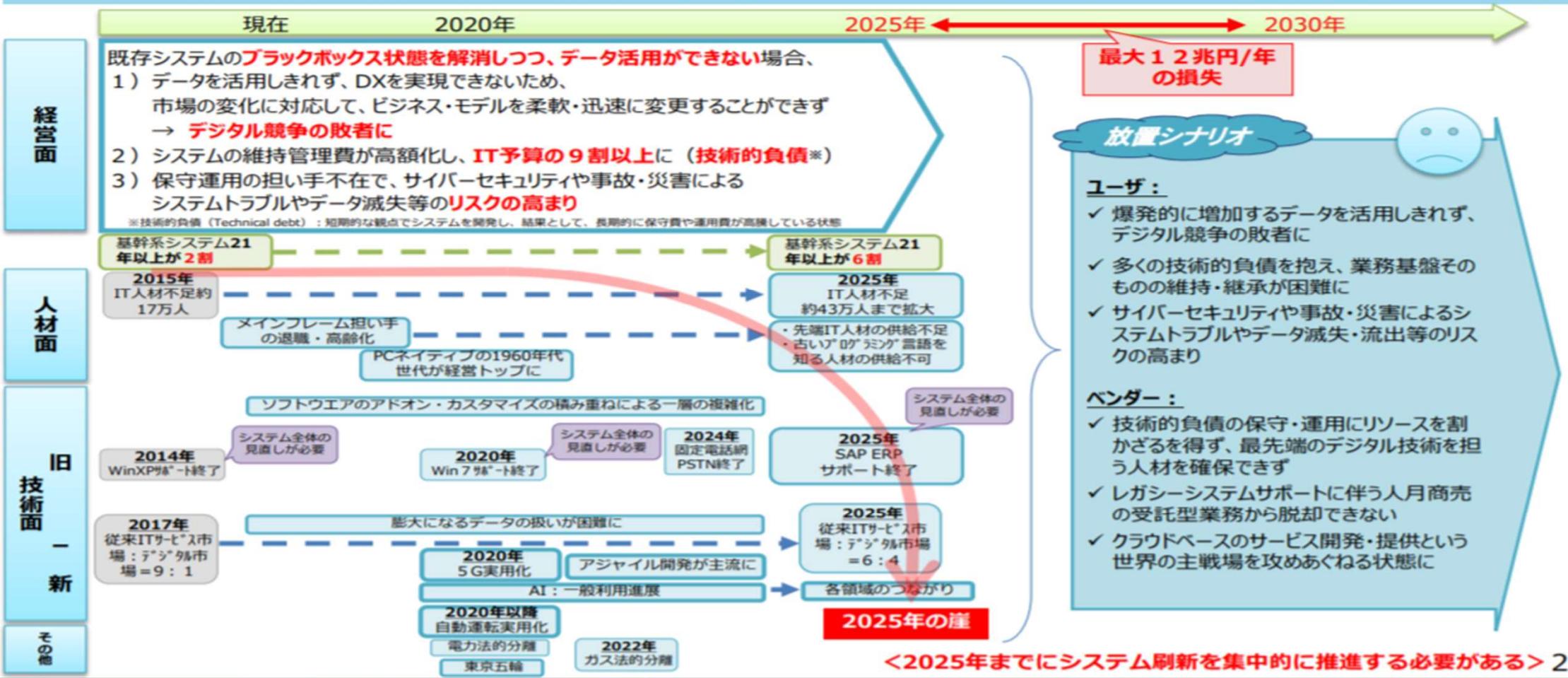


2025年の崖



多くの経営者が、将来の成長、競争力強化のために、新たなデジタル技術を活用して新たなビジネス・モデルを創出・柔軟に改変するデジタル・トランスフォーメーション（＝DX）の必要性について理解しているが…

- 既存システムが、事業部門ごとに構築されて、全社横断的なデータ活用ができなかったり、過剰なカスタマイズがなされているなどにより、複雑化・ブラックボックス化
 - 経営者がDXを望んでも、データ活用のために上記のような既存システムの問題を解決し、そのためには業務自体の見直しも求められる中（＝経営改革そのもの）、現場サイドの抵抗も大きく、いかにこれを実行するかが課題となっている
- この課題を克服できない場合、DXが実現できないのみでなく、**2025年以降、最大12兆円／年（現在の約3倍）の経済損失が生じる可能性（2025年の崖）。**



多くの経営者が、将来の成長、競争力強化のために、新たなデジタル技術を活用して新たなビジネスモデルを創出・柔軟に改変する**デジタルトランフォーメーション**の重要性について理解しているが・・・

- 既存システムが、**事業部門ごとに構築**されて、全社横断的なデータ活用ができなかったり、**過剰なカスタマイズ**がなされているなどにより**複雑化・ブラックボックス化**
- 経営者がDXを望んでも、データ活用のために上記のような**既存システムの問題を解決し**、そのためには**業務自体の見直しを求められる中**(=経営改革そのもの)、現場サイドの抵抗も大きく、**いかにこれを実行するかが課題**となっている

→この課題を克服できない場合、**DXが実現できない**のみでなく、
2025年以降、**最大12兆円/年の経済損失**が生じる可能性

DX推進の課題

- IT技術者不足 2025年の崖 なんと43万人（現時点ですでに17万人）
- 2025年基幹システム21年以上稼働が6割
- レガシーを引きずる 維持で予算の8割以上 データ活用出来ない
- 保守運用の**担い手不足**によりサイバーセキュリティの問題
- 高度IT人材の高騰
- とてもではないがDX実現が出来る環境ではない

ノーコードを活用してDX人材不足をカバー

「ノーコード」とは「ノンプログラミング」のことで、プログラムを書かずにアプリやWebサイトを作ったりシステム連携したりすること

ノーコード・カオスマップ 2022年8月版

・本カオスマップは、当協会が独自に作成しております。サービスの信頼性や正確性を完全に担保するものではありません。
・商標及びロゴマークに関する権利は、各自の権利の所有者に帰属します。
・掲載に問題がある場合や、出回り謝罪に明謝を希望する場合は、下記までご連絡ください。
連絡先: info@no-coders-japan.org



NoCoders JAPAN
Association

ビジネスアプリ

salesforce AppSheet N Google Sheets
 Airtable Power Apps Forguncy
 ZOHO Works AppMaster.io
 DronaHQ Betty Blocks UI Bakery
 Querier Retool
 Pleasanter EasyCSV backenless
 Claris JUST.DB coda スタートダービー^{SmartDB}
 SPIRAL outsystems UnitBase
 SCALE CLOUD Genexus PEGA CELF
 mendix ZeroOne WebPerformer
 モバイルアプリメイン
 UNIFINITY INC. Plato glide.
 thunkable AppGyver Click

EC / D2C

shopify BASE MakeShop stripe
 ARCADIER gumroad CX ORDER COLOR ME
 MemberSpace SQUARESPACE TéP

Webサイト

WORDPRESS ferret One ペライチ Ownd
 SHIRAHAMA WiX STUDIO strikingly
 Stacker Carrd Wraptas webflow
 PORY KARTE bx BuilderX

モバイルアプリ

Adalo yappli CRAYON Joint Apps
 Monaca Bravo Magic Instructions appsole Appify

顧客管理

チャットボット Chat Plus hachidori GENE CHAT hitabo anybot COTOHA kuzen SYNALIO KARAKURI Chat Dealer AI

フォーム

formrun Google Forms Formailer フォームメーラー formzu SECURE FORM mailchimp Typeform JotForm

業務自動化

make formerly Integromat zapier ActRecipe Reckoner warp workato Astend KAMINASHI PARASOLA DataSpider Cloud sheet2api IFTTT

テスト自動化

Autify SmartQA Magic Pod Techtouch

学習

NoCoders JAPAN Association NoCodeCamp idemo NOCODE SCHOOL LikePay Academy .dev ゼロイチ

IoT / AI

Gravio ifLink Open Community MENOU lobe obviously.ai MatrixFlow Neural Network Console

VR/AR/3D

ROBLOX BUILDBOX MINSAR palanAR

受託開発 / 人材マッチング

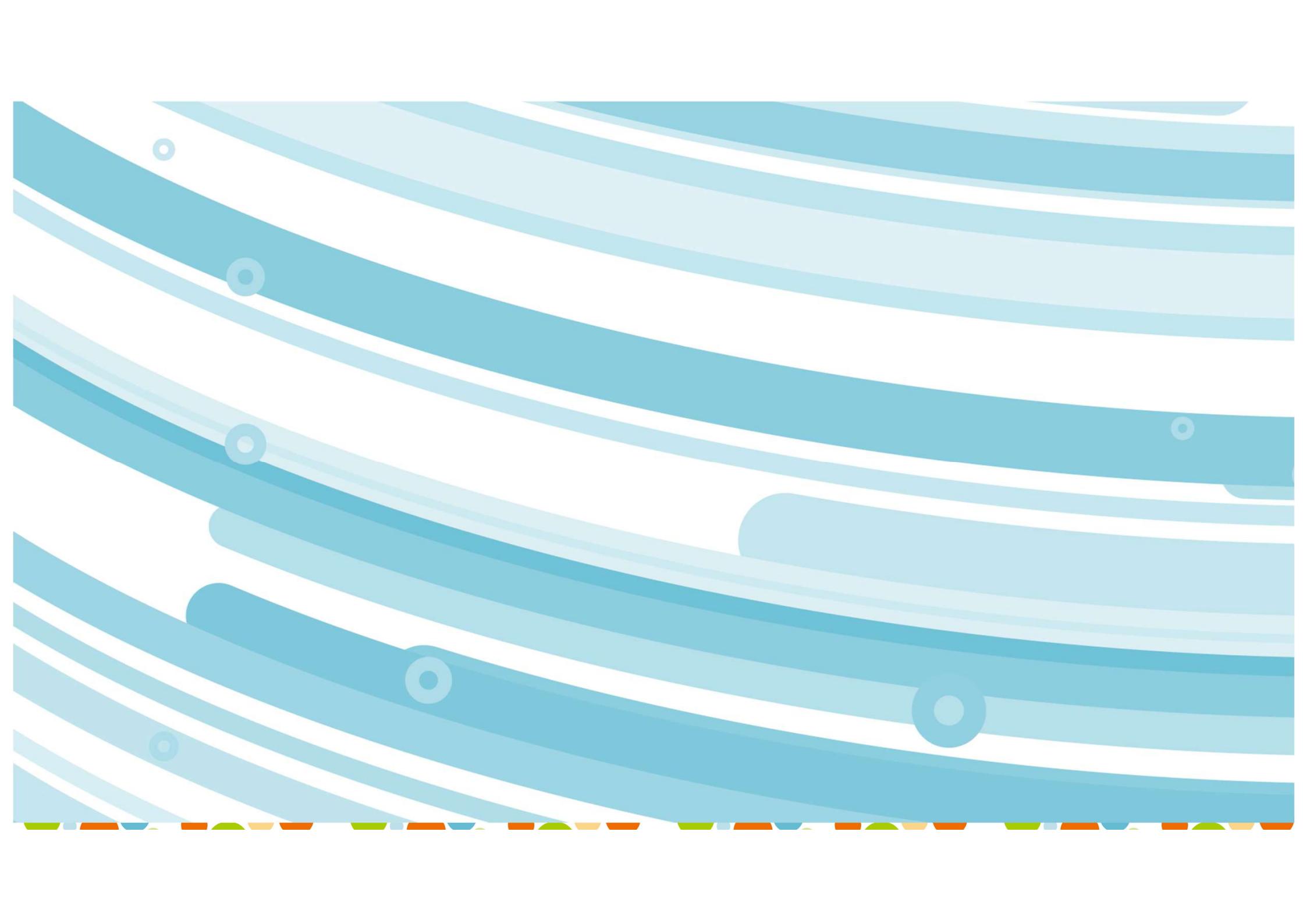
NOCODO NOCODOBiz reve Swoooo FastAPP Sketchto APP Sales Renovation sowacana BOLT Walkers npakeji i-enter corporation ノーコードラボ

ノーコード全盛時代がやってくる

IDCジャパンが国内の400～500社・団体に調査したところ、ノーコードの導入率は**2020年に8.5%**だったが
2021年には37.7%まで増えた。

IDCジャパンは「**23年には新規開発される**

アプリケーションの6割がノーコード
などによるものになる」と予測する。



ノーコードツールのkintoneとは？

従来の業務システム
プログラミング & 専門知識

```
General [Declarations]
Rows("1:20").Select
Selection.EntireRow.Hidden = False
ActiveSheet.Shapes.Range(Array("TRADOS1", "TRADOS2", "TRADOS3", "SwitchnonTRADOS")).Select
Selection.Delete

Rows("17:18").Select
Selection.EntireRow.Hidden = True
Rows("15").Select
Selection.EntireRow.Hidden = True

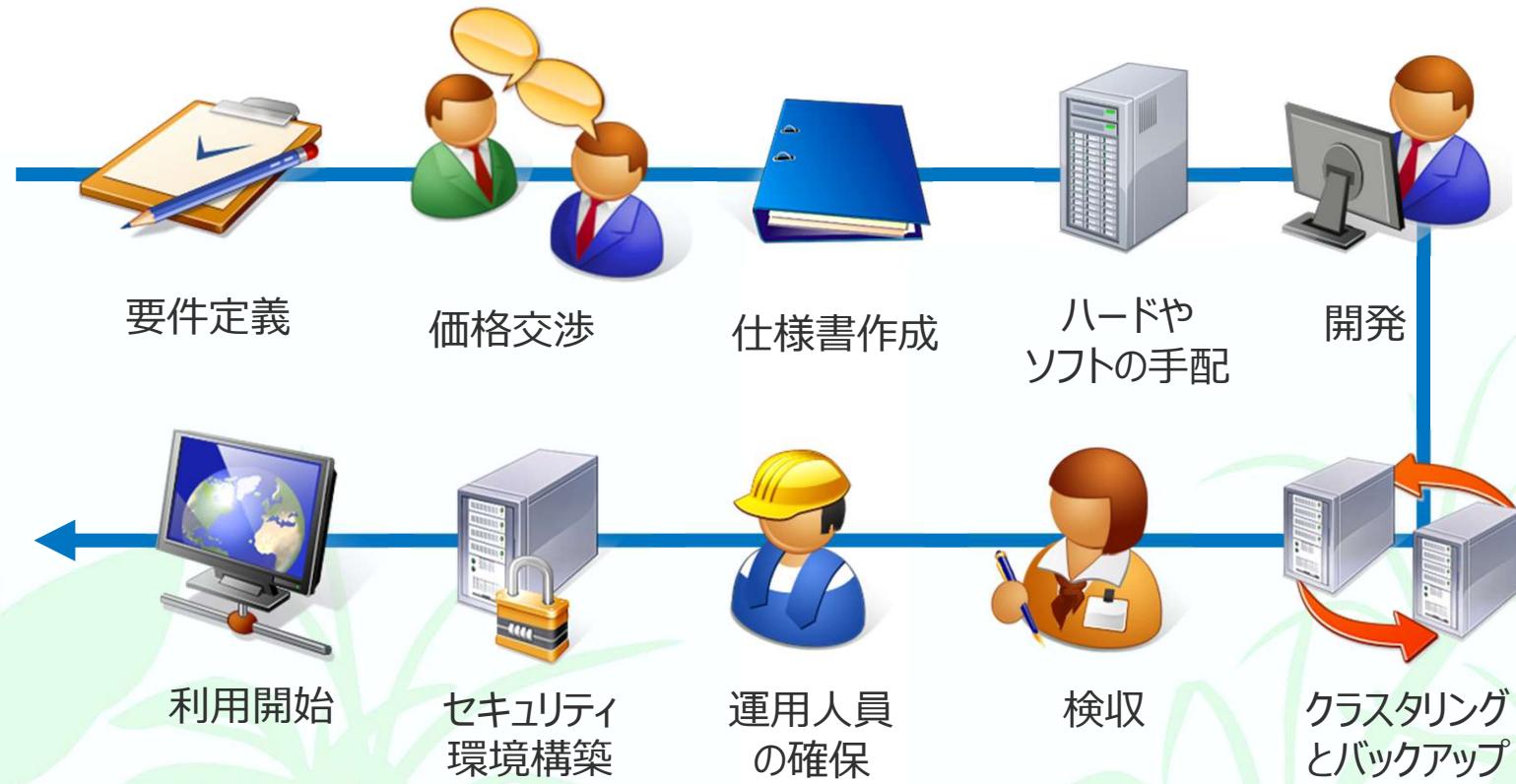
With Range("B2")
    With ActiveSheet.Buttons.Add(421, 278, 106, 64)
        .Caption = ">>> Trados Version"
        .Name = "SwitchTRADOS"
    End With
    ActiveSheet.Shapes.Range(Array("SwitchTRADOS")).Select
    Selection.Characters.Text = ">>> Trados Version"
    With Selection.Characters(Start:=1, Length:=20).Font
        .Name = "Meiryo UI"
        .FontStyle = "Bold"
        .Size = 12
        .ColorIndex = 16
    End With
    Selection.OnAction = "TRADOS"
End With

With Range("F14")
    With ActiveSheet.Buttons.Add(421, 38, 95, 100)
        .Caption = "Output PO in English"
        .Name = "nonTRADOS"
    End With
    ActiveSheet.Shapes.Range(Array("nonTRADOS")).Select
    Selection.Characters.Text = "Output PO in English"
    With Selection.Characters(Start:=1, Length:=20).Font
        .Name = "Meiryo UI"
        .FontStyle = "Bold"
        .Size = 12
        .ColorIndex = 16
    End With
    Selection.OnAction = "SavePOnonTRADOS_EN"
End With

With Range("F14")
    With ActiveSheet.Buttons.Add(521, 38, 95, 100)
        .Caption = "Output PO in English"
        .Name = "nonTRADOS"
    End With
    ActiveSheet.Shapes.Range(Array("nonTRADOS")).Select
    Selection.Characters.Text = "Output PO in English"
    With Selection.Characters(Start:=1, Length:=20).Font
        .Name = "Meiryo UI"
        .FontStyle = "Bold"
        .Size = 12
        .ColorIndex = 16
    End With
    Selection.OnAction = "SavePOnonTRADOS_EN"
End With
```



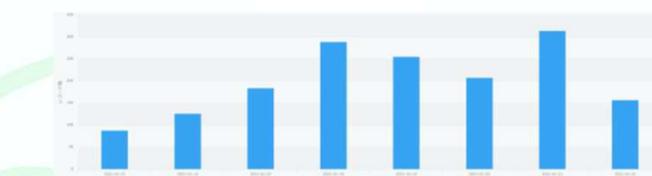
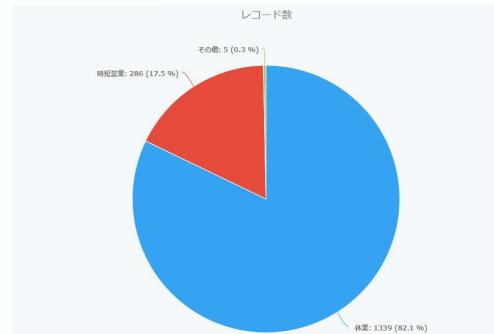
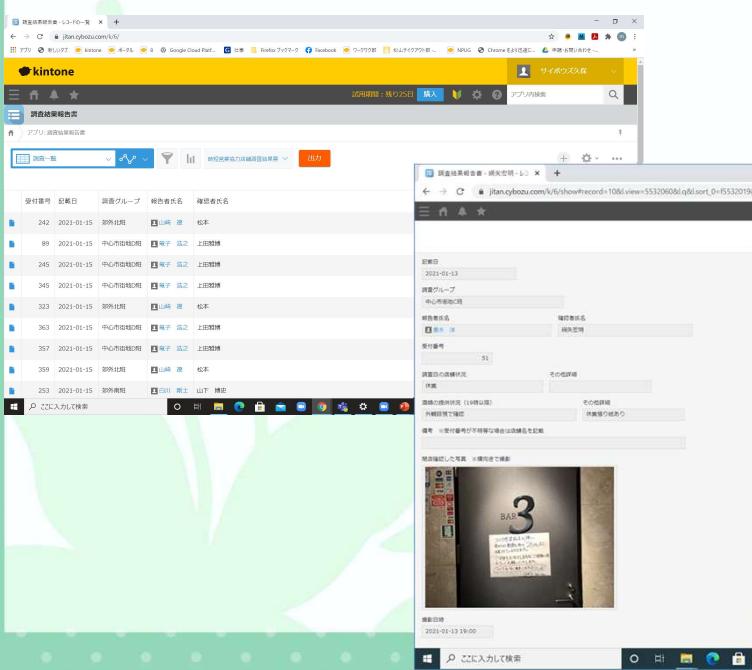
kintoneによる新しいSI



松山市時短協力金巡回報告アプリ

わずか2時間でサービス運用開始！

時短協力金支給にあたり、申請飲食店の時短営業状況巡回確認報告書作成工数を大幅に削減！



1/13より時短協力金受付開始



1/13 16時頃市役所に往訪し
作業状況を確認



その場でkintoneの試用申込
を行い対面でアプリ開発



18時に職員を集め、スマートフォンアプリ
の説明を行い当日より運用開始

Kintoneデモ

kintoneログイン情報（今日から30日間利用可能）



<https://imabari-dx4.cybozu.com/>



ID/初期PWは user01～user40 です

30日間無料お試しも可能

kintone お試し

で検索

RPAサービス「BizRobo!」 及び各種AIツールのご紹介

第4回 今治市DX勉強会
デジタル未来コンソーシアム

1. 会社紹介

2. RPAとは

3. 「BizRobo!」のご紹介

4. 生成AIとは

5. 各種製品のご紹介

6. 質疑応答



永吉 慶伍
Keigo Nagayoshi

会社

株式会社デジタルテクノロジー四国

所属

LX推進部

出身

高知県土佐市



社名

株式会社デジタルテクノロジー四国

住所

愛媛県松山市三番町四丁目9番地5

設立

2021年4月1日

代表

代表取締役会長 清水 一郎

代表取締役社長 元屋地 裕之

資本金

1億円

従業員数

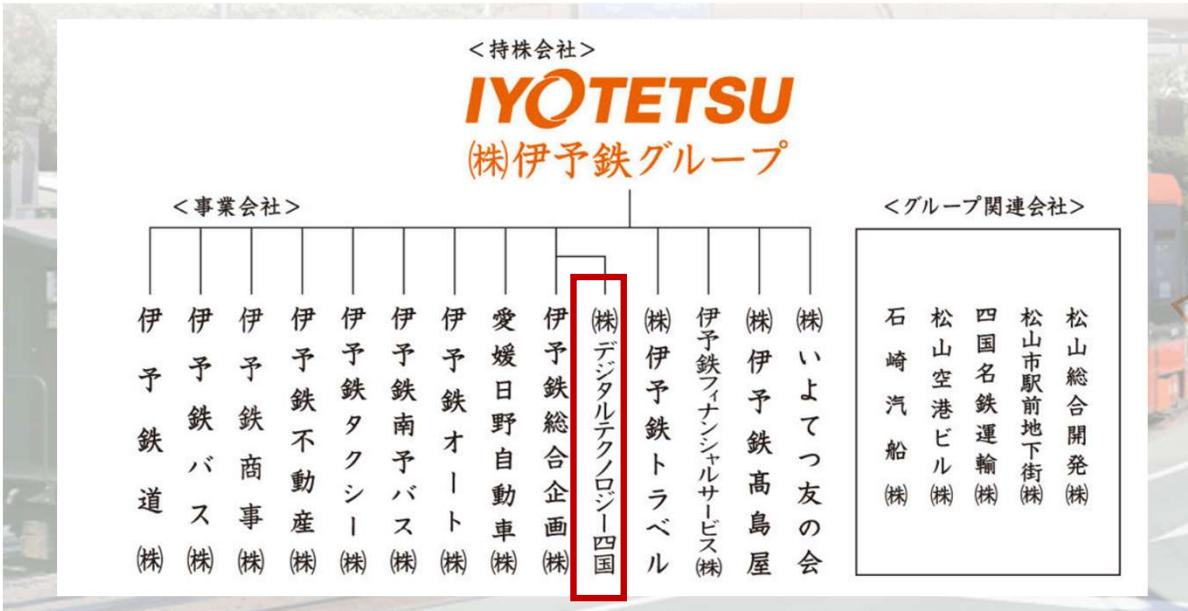
17名

出資元

伊予鉄総合企画株式会社

オープングループ株式会社

〈伊予鉄グループ 概要〉



 いよてつ総合企画
IYOTETSU PLANNING INC.

伊予鉄グループ内で唯一IT部門を持つ

広告事業

人材ビジネス事業

情報システム事業

指定管理運営事業

保育運営事業

DX推進事業



株式会社 デジタルテクノロジー 四国

デジタル戦略を加速させ、
より専門性に特化した人材を育成するため
2021年4月1日に設立



会社名	伊予鉄総合企画株式会社	
設立	1986年4月	
所在地	愛媛県松山市三番町4丁目9-5	
代表者	代表取締役会長 元屋地裕之 代表取締役社長 定松伸一	
事業内容	広告事業 / 人材ビジネス事業 / 情報システム事業 指定管理運営事業 / 保育運営事業 等	
持ち株会社	株式会社伊予鉄グループ	
グループ会社	伊予鉄道株式会社 伊予鉄商事株式会社 伊予鉄タクシー株式会社 伊予鉄オート株式会社 株式会社伊予鉄トラベル 伊予鉄フィナンシャルサービス株式会社 株式会社伊予鉄高島屋	伊予鉄バス株式会社 伊予鉄不動産株式会社 伊予鉄南予バス株式会社 愛媛日野自動車株式会社
グループ関連会社	石崎汽船株式会社 四国名鉄運輸株式会社 松山総合開発株式会社	松山空港ビル株式会社 松山市駅前地下街株式会社

会社名	オープングループ株式会社	
設立	2000年4月	
所在地	東京都港区西新橋三丁目3番1号KDX西新橋ビル3階	
代表者	代表取締役社長 高橋和道	
事業内容	純粹持ち株会社	
グループ会社	オープン株式会社 リーグル株式会社 ホスピタリティパートナーズ株式会社 ご近所ワーク株式会社 オートロ株式会社	

◆DX支援事業

- ・RPAをはじめ、各種DXツールの取り扱い
- ・業務コンサルティング

◆キャッシュレス決済事業

- ・「みきゃんアプリ」
- ・アプリ基盤提供

◆BPOサービス

- ・業務請負
- ・システム開発/テスト代行

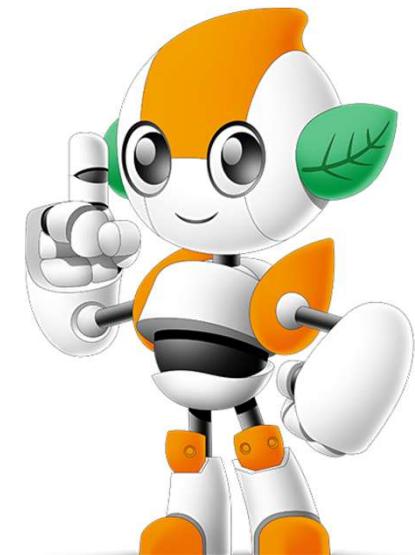
◆WEBマーケティングサービス(WEB広告)

◆地域起業家支援

◆各種IoT製品の取り扱い



みきゃんアプリ
許諾番号: 410053

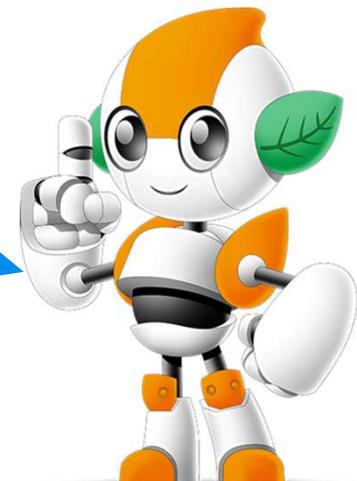


地場企業として、四国に根差したサポートで伴奏支援します！

- ・官公庁/自治体
- ・製造業
- ・建築、建設
- ・食品
- ・小売
- ・サービス
- ・教育機関
- ・病院

業種・規模を問わず、
幅広くご利用いただ
いております！

四国での支援企業
200社以上！



《提供パッケージ》

RPA

BizRobo!

 **WinActor[®]**

 **AUTORO**

ローコードツール

 **kintone**

CRM

 **PARTNER**

グループウェア

 **サイボウズ
Garoon**

サイボウズ
Office

AI-OCR

カルクペーパー

 **デジパス**

AIサービス

カルクワーカス

ShareMind

AI搭載スライド自動生成サービス

 **イルシル**

チャットボット

カルクチャット

Tebot

 **AI Concierge**

その他

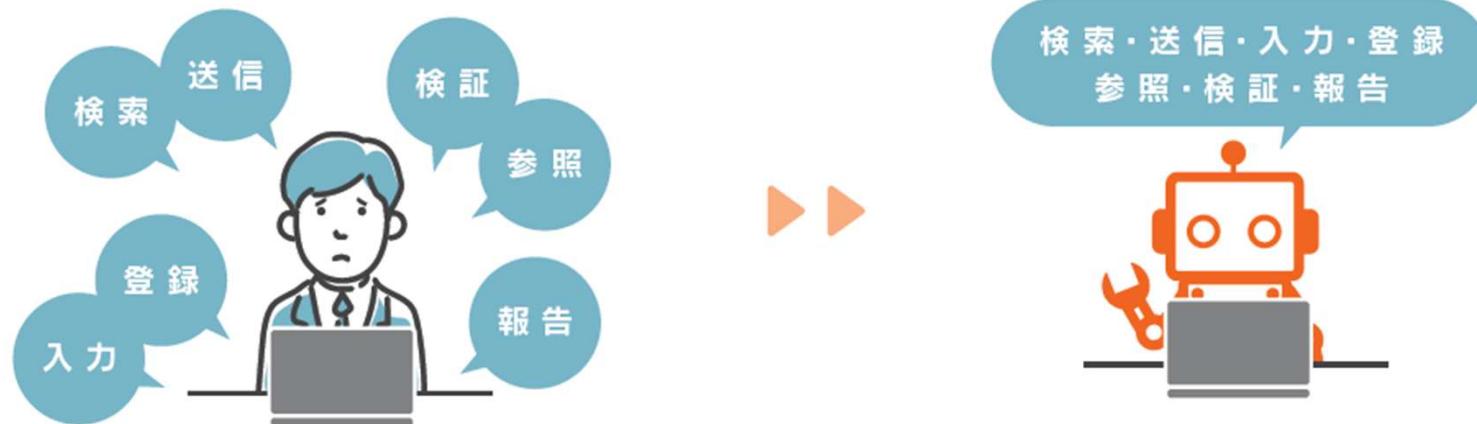
 **MySherpa**

RPAとは

RPAとは・・・ロボティック・プロセス・オートメーション

(Robotic Process Automation)

RPAは主にオフィスのデスクワークなどで発生する定型作業を、
パソコンの中にあるソフトウェア型のロボットが代行・自動化する技術



Before:PCを使った単純作業

After:ロボットがプロセスを代行 (RPA)

まずは実際の動きを
ご覧ください。



請求データ登録

支払依頼書_20190701.xlsx	2019/08/01 13:41	Microsoft Excel ワ...
支払依頼書_20190712.xlsx	2019/08/01 13:42	Microsoft Excel ワ...
支払依頼書_20190716.xlsx	2019/08/01 13:42	Microsoft Excel ワ...
支払依頼書_20190719.xlsx	2019/08/01 13:42	Microsoft Excel ワ...
支払依頼書_20190720.xlsx	2019/08/01 13:43	Microsoft Excel ワ...
支払依頼書_20190722.xlsx	2019/08/01 13:44	Microsoft Excel ワ...

請求データが記載された同じフォーマットの複数のExcelファイル

請求データ

支 払 依 頼 書

提出日	2019年7月1日		
取引日	2019年7月1日		
部 門	300 企画部		
業 目	201 交際費		
仕 入 先 コード	102		
仕 入 先 名	口 口 株 式 会 社		
仕 入 内 容	金 額	税 額	摘要
スマートフォン	70,000 円	5,600 円	1台
	円	円	
	円	円	

請求データ

RPA Demo

支払依頼入力

取引日	2019年10月 3日
仕入先	
費目	
部門	
仕入内容	
金額	
税額	
合計額	
摘要	
登録	
閉じる	

会計システムに請求データを登録

株式会社 デジタルテクノロジー四国 × **BizRobo!**

請求情報を システムに登録する

RPA導入前



働き方改革推進キャラクター
IyoRobo!

株式会社 デジタルテクノロジー四国 × **BizRobo!**



請求情報を
システムに登録する

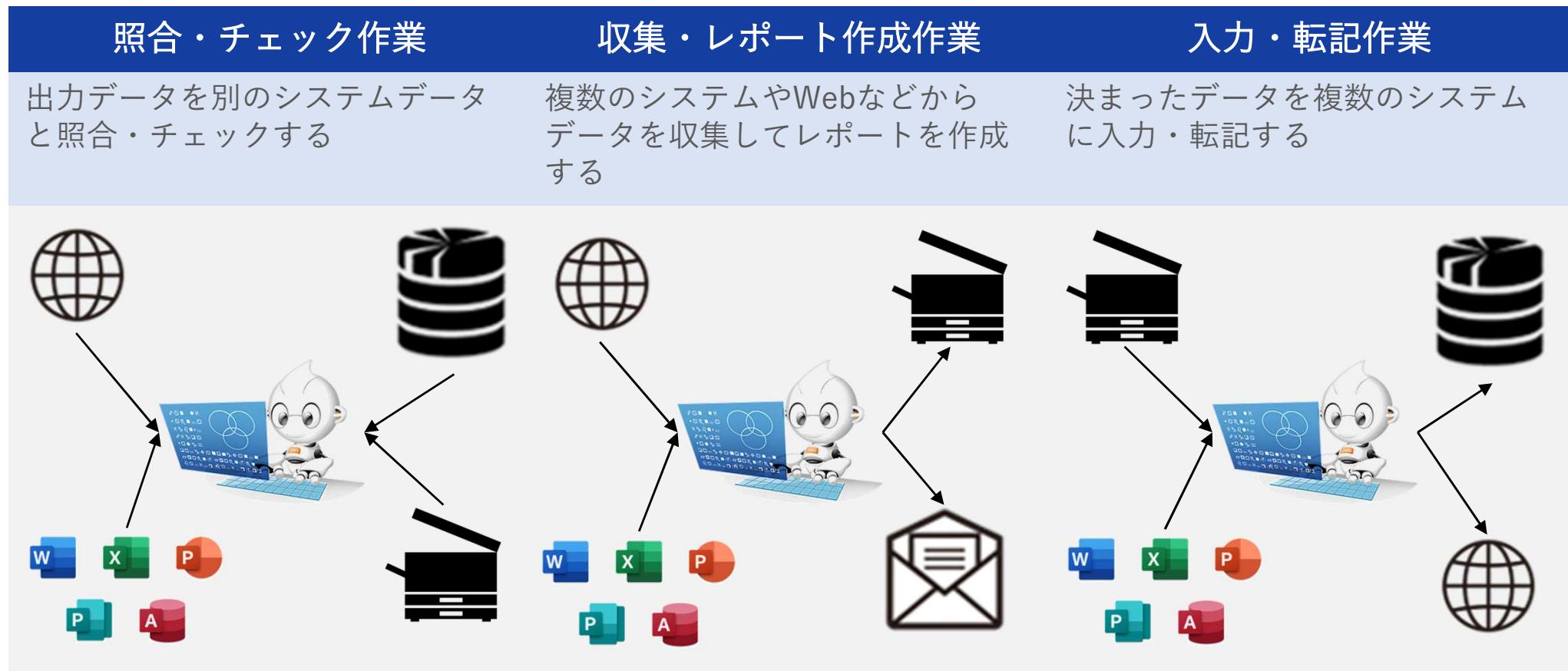
RPA導入後



働き方改革推進キャラクター
IyoRobo!

RPAは、人間が行っている作業の内、**条件が明確な判断や定型的なPC操作**を記録させることで、人間の代わりに作業を行ってくれます。

「システム間や作成データの照合」、「データ収集・各種ソフトでのレポート作成」、「システムへの転記・入力」などに適しています。「データのダウンロード、アップロード」といった単純な操作での切り口も有効です。



1

自分でカスタマイズ

- ◆ ロボットの作成・修正は自分で行う
- ◆ IT技術者でなくとも作成可能
- ◆ 業務の変更や臨時の作業にも即座に対応

2

圧倒的な処理能力

- ◆ 人間の行う作業の数倍～数十倍のスピードで処理
- ◆ 24時間365日稼働

1

業務の効率化

- ◆ 作業時間の削減
- ◆ ヒューマンエラーの解消
- ◆ 導入にあたっての業務整理やフローの見直しが可能

2

生産性の向上

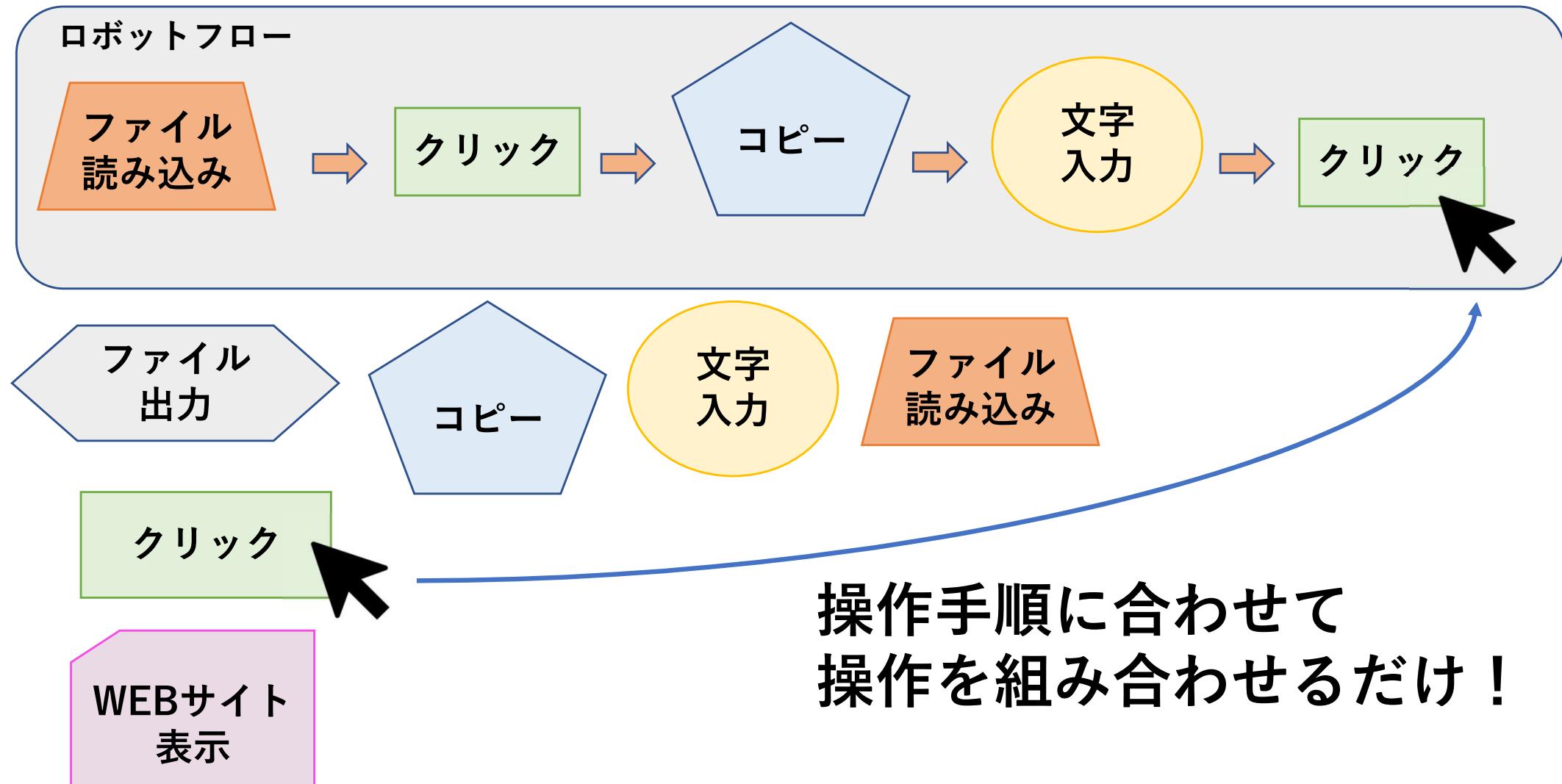
- ◆ 単純作業から解放され、人間にしかできない仕事へシフト
- ◆ 情報収集や確認など、人間でやるとキリがない作業を実現可能
- ◆ 1人しかできない仕事が何人でもできるように

「BizRobo!」のご紹介

```
75
76  var _line = __webpack_require__(1);
77
78  var _line2 = _interopRequireDefault(_line);
79
80  function _interopRequireDefault(obj) { return obj && obj.__esModule ? obj : { default: obj };
81
82  window.Line = _line2.default;
83
```

このようなイメージを抱きませんか？

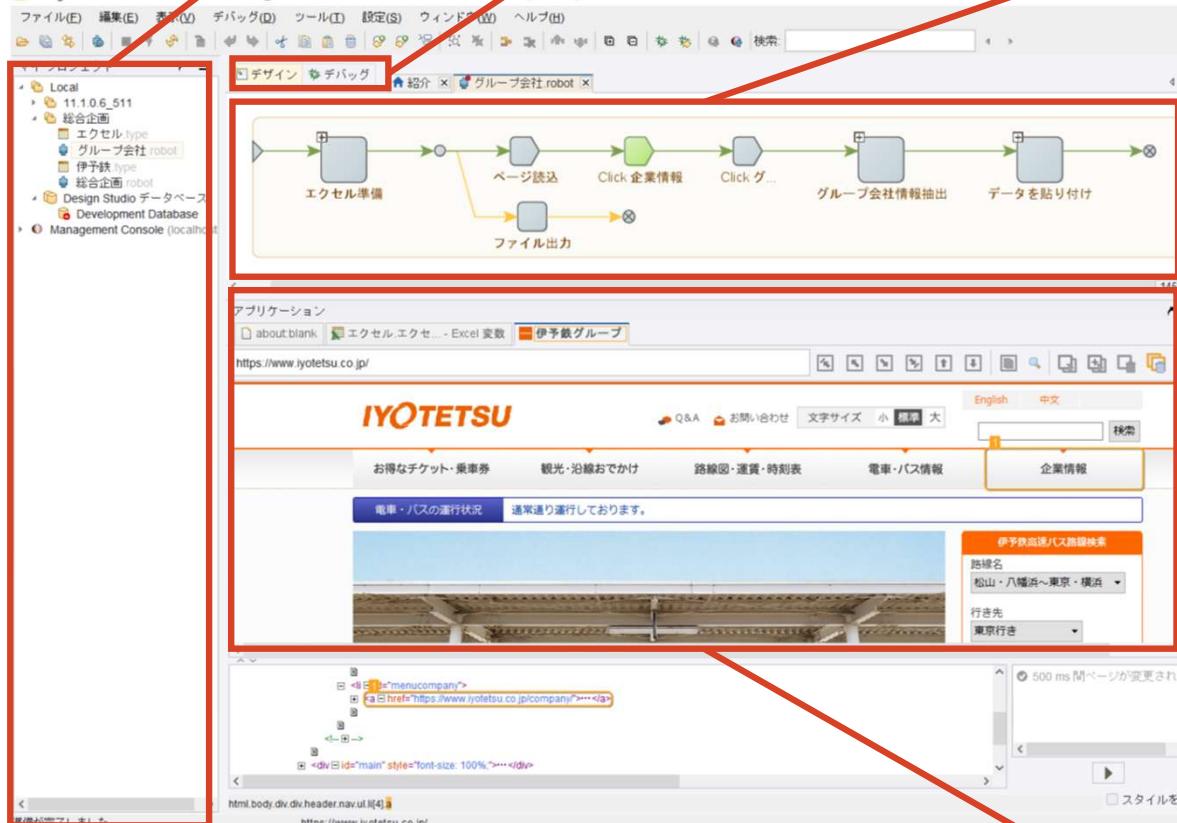
```
88 "use strict";
89
90 Object.defineProperty(exports, "__esModule", {
91   value: true
92 });
93 var _createClass = function () { function defineProperties(target, props) { for (var i = 0, l = props.length; i < l; i++) {
94     Object.defineProperty(target, props[i].name, props[i]);
95   }
96   function _classCallCheck(instance, Constructor) { if (!(instance instanceof Constructor)) {
97     throw new TypeError("Cannot call a class as a function");
98   }
99   _createClass(Constructor.prototype, [
100     {
101       key: "constructor",
102       value: function () {
103         _classCallCheck(this, Constructor);
104       }
105     }
106   ]);
107   return Constructor;
108 }
109 _createClass(Constructor.prototype, [
110   {
111     key: "constructor",
112     value: function () {
113       _classCallCheck(this, Constructor);
114     }
115   }
116 ]);
117 }
118 _createClass(Constructor.prototype, [
119   {
120     key: "constructor",
121     value: function () {
122       _classCallCheck(this, Constructor);
123     }
124   }
125 ]);
126
127 _createClass(Constructor.prototype, [
128   {
129     key: "constructor",
130     value: function () {
131       _classCallCheck(this, Constructor);
132     }
133   }
134 ]);
135
136 _createClass(Constructor.prototype, [
137   {
138     key: "constructor",
139     value: function () {
140       _classCallCheck(this, Constructor);
141     }
142   }
143 ]);
144
145 _createClass(Constructor.prototype, [
146   {
147     key: "constructor",
148     value: function () {
149       _classCallCheck(this, Constructor);
150     }
151   }
152 ]);
153
154 _createClass(Constructor.prototype, [
155   {
156     key: "constructor",
157     value: function () {
158       _classCallCheck(this, Constructor);
159     }
160   }
161 ]);
162
163 _createClass(Constructor.prototype, [
164   {
165     key: "constructor",
166     value: function () {
167       _classCallCheck(this, Constructor);
168     }
169   }
170 ]);
171
172 _createClass(Constructor.prototype, [
173   {
174     key: "constructor",
175     value: function () {
176       _classCallCheck(this, Constructor);
177     }
178   }
179 ]);
180
181 _createClass(Constructor.prototype, [
182   {
183     key: "constructor",
184     value: function () {
185       _classCallCheck(this, Constructor);
186     }
187   }
188 ]);
189
190 _createClass(Constructor.prototype, [
191   {
192     key: "constructor",
193     value: function () {
194       _classCallCheck(this, Constructor);
195     }
196   }
197 ]);
198
199 _createClass(Constructor.prototype, [
200   {
201     key: "constructor",
202     value: function () {
203       _classCallCheck(this, Constructor);
204     }
205   }
206 ]);
207
208 _createClass(Constructor.prototype, [
209   {
210     key: "constructor",
211     value: function () {
212       _classCallCheck(this, Constructor);
213     }
214   }
215 ]);
216
217 _createClass(Constructor.prototype, [
218   {
219     key: "constructor",
220     value: function () {
221       _classCallCheck(this, Constructor);
222     }
223   }
224 ]);
225
226 _createClass(Constructor.prototype, [
227   {
228     key: "constructor",
229     value: function () {
230       _classCallCheck(this, Constructor);
231     }
232   }
233 ]);
234
235 _createClass(Constructor.prototype, [
236   {
237     key: "constructor",
238     value: function () {
239       _classCallCheck(this, Constructor);
240     }
241   }
242 ]);
243
244 _createClass(Constructor.prototype, [
245   {
246     key: "constructor",
247     value: function () {
248       _classCallCheck(this, Constructor);
249     }
250   }
251 ]);
252
253 _createClass(Constructor.prototype, [
254   {
255     key: "constructor",
256     value: function () {
257       _classCallCheck(this, Constructor);
258     }
259   }
260 ]);
261
262 _createClass(Constructor.prototype, [
263   {
264     key: "constructor",
265     value: function () {
266       _classCallCheck(this, Constructor);
267     }
268   }
269 ]);
270
271 _createClass(Constructor.prototype, [
272   {
273     key: "constructor",
274     value: function () {
275       _classCallCheck(this, Constructor);
276     }
277   }
278 ]);
279
280 _createClass(Constructor.prototype, [
281   {
282     key: "constructor",
283     value: function () {
284       _classCallCheck(this, Constructor);
285     }
286   }
287 ]);
288
289 _createClass(Constructor.prototype, [
290   {
291     key: "constructor",
292     value: function () {
293       _classCallCheck(this, Constructor);
294     }
295   }
296 ]);
297
298 _createClass(Constructor.prototype, [
299   {
300     key: "constructor",
301     value: function () {
302       _classCallCheck(this, Constructor);
303     }
304   }
305 ]);
306
307 _createClass(Constructor.prototype, [
308   {
309     key: "constructor",
310     value: function () {
311       _classCallCheck(this, Constructor);
312     }
313   }
314 ]);
315
316 _createClass(Constructor.prototype, [
317   {
318     key: "constructor",
319     value: function () {
320       _classCallCheck(this, Constructor);
321     }
322   }
323 ]);
324
325 _createClass(Constructor.prototype, [
326   {
327     key: "constructor",
328     value: function () {
329       _classCallCheck(this, Constructor);
330     }
331   }
332 ]);
333
334 _createClass(Constructor.prototype, [
335   {
336     key: "constructor",
337     value: function () {
338       _classCallCheck(this, Constructor);
339     }
340   }
341 ]);
342
343 _createClass(Constructor.prototype, [
344   {
345     key: "constructor",
346     value: function () {
347       _classCallCheck(this, Constructor);
348     }
349   }
350 ]);
351
352 _createClass(Constructor.prototype, [
353   {
354     key: "constructor",
355     value: function () {
356       _classCallCheck(this, Constructor);
357     }
358   }
359 ]);
360
361 _createClass(Constructor.prototype, [
362   {
363     key: "constructor",
364     value: function () {
365       _classCallCheck(this, Constructor);
366     }
367   }
368 ]);
369
370 _createClass(Constructor.prototype, [
371   {
372     key: "constructor",
373     value: function () {
374       _classCallCheck(this, Constructor);
375     }
376   }
377 ]);
378
379 _createClass(Constructor.prototype, [
380   {
381     key: "constructor",
382     value: function () {
383       _classCallCheck(this, Constructor);
384     }
385   }
386 ]);
387
388 _createClass(Constructor.prototype, [
389   {
390     key: "constructor",
391     value: function () {
392       _classCallCheck(this, Constructor);
393     }
394   }
395 ]);
396
397 _createClass(Constructor.prototype, [
398   {
399     key: "constructor",
400     value: function () {
401       _classCallCheck(this, Constructor);
402     }
403   }
404 ]);
405
406 _createClass(Constructor.prototype, [
407   {
408     key: "constructor",
409     value: function () {
410       _classCallCheck(this, Constructor);
411     }
412   }
413 ]);
414
415 _createClass(Constructor.prototype, [
416   {
417     key: "constructor",
418     value: function () {
419       _classCallCheck(this, Constructor);
420     }
421   }
422 ]);
423
424 _createClass(Constructor.prototype, [
425   {
426     key: "constructor",
427     value: function () {
428       _classCallCheck(this, Constructor);
429     }
430   }
431 ]);
432
433 _createClass(Constructor.prototype, [
434   {
435     key: "constructor",
436     value: function () {
437       _classCallCheck(this, Constructor);
438     }
439   }
440 ]);
441
442 _createClass(Constructor.prototype, [
443   {
444     key: "constructor",
445     value: function () {
446       _classCallCheck(this, Constructor);
447     }
448   }
449 ]);
450
451 _createClass(Constructor.prototype, [
452   {
453     key: "constructor",
454     value: function () {
455       _classCallCheck(this, Constructor);
456     }
457   }
458 ]);
459
460 _createClass(Constructor.prototype, [
461   {
462     key: "constructor",
463     value: function () {
464       _classCallCheck(this, Constructor);
465     }
466   }
467 ]);
468
469 _createClass(Constructor.prototype, [
470   {
471     key: "constructor",
472     value: function () {
473       _classCallCheck(this, Constructor);
474     }
475   }
476 ]);
477
478 _createClass(Constructor.prototype, [
479   {
480     key: "constructor",
481     value: function () {
482       _classCallCheck(this, Constructor);
483     }
484   }
485 ]);
486
487 _createClass(Constructor.prototype, [
488   {
489     key: "constructor",
490     value: function () {
491       _classCallCheck(this, Constructor);
492     }
493   }
494 ]);
495
496 _createClass(Constructor.prototype, [
497   {
498     key: "constructor",
499     value: function () {
500       _classCallCheck(this, Constructor);
501     }
502   }
503 ]);
504
505 _createClass(Constructor.prototype, [
506   {
507     key: "constructor",
508     value: function () {
509       _classCallCheck(this, Constructor);
510     }
511   }
512 ]);
513
514 _createClass(Constructor.prototype, [
515   {
516     key: "constructor",
517     value: function () {
518       _classCallCheck(this, Constructor);
519     }
520   }
521 ]);
522
523 _createClass(Constructor.prototype, [
524   {
525     key: "constructor",
526     value: function () {
527       _classCallCheck(this, Constructor);
528     }
529   }
530 ]);
531
532 _createClass(Constructor.prototype, [
533   {
534     key: "constructor",
535     value: function () {
536       _classCallCheck(this, Constructor);
537     }
538   }
539 ]);
540
541 _createClass(Constructor.prototype, [
542   {
543     key: "constructor",
544     value: function () {
545       _classCallCheck(this, Constructor);
546     }
547   }
548 ]);
549
550 _createClass(Constructor.prototype, [
551   {
552     key: "constructor",
553     value: function () {
554       _classCallCheck(this, Constructor);
555     }
556   }
557 ]);
558
559 _createClass(Constructor.prototype, [
560   {
561     key: "constructor",
562     value: function () {
563       _classCallCheck(this, Constructor);
564     }
565   }
566 ]);
567
568 _createClass(Constructor.prototype, [
569   {
570     key: "constructor",
571     value: function () {
572       _classCallCheck(this, Constructor);
573     }
574   }
575 ]);
576
577 _createClass(Constructor.prototype, [
578   {
579     key: "constructor",
580     value: function () {
581       _classCallCheck(this, Constructor);
582     }
583   }
584 ]);
585
586 _createClass(Constructor.prototype, [
587   {
588     key: "constructor",
589     value: function () {
590       _classCallCheck(this, Constructor);
591     }
592   }
593 ]);
594
595 _createClass(Constructor.prototype, [
596   {
597     key: "constructor",
598     value: function () {
599       _classCallCheck(this, Constructor);
600     }
601   }
602 ]);
603
604 _createClass(Constructor.prototype, [
605   {
606     key: "constructor",
607     value: function () {
608       _classCallCheck(this, Constructor);
609     }
610   }
611 ]);
612
613 _createClass(Constructor.prototype, [
614   {
615     key: "constructor",
616     value: function () {
617       _classCallCheck(this, Constructor);
618     }
619   }
620 ]);
621
622 _createClass(Constructor.prototype, [
623   {
624     key: "constructor",
625     value: function () {
626       _classCallCheck(this, Constructor);
627     }
628   }
629 ]);
630
631 _createClass(Constructor.prototype, [
632   {
633     key: "constructor",
634     value: function () {
635       _classCallCheck(this, Constructor);
636     }
637   }
638 ]);
639
640 _createClass(Constructor.prototype, [
641   {
642     key: "constructor",
643     value: function () {
644       _classCallCheck(this, Constructor);
645     }
646   }
647 ]);
648
649 _createClass(Constructor.prototype, [
650   {
651     key: "constructor",
652     value: function () {
653       _classCallCheck(this, Constructor);
654     }
655   }
656 ]);
657
658 _createClass(Constructor.prototype, [
659   {
660     key: "constructor",
661     value: function () {
662       _classCallCheck(this, Constructor);
663     }
664   }
665 ]);
666
667 _createClass(Constructor.prototype, [
668   {
669     key: "constructor",
670     value: function () {
671       _classCallCheck(this, Constructor);
672     }
673   }
674 ]);
675
676 _createClass(Constructor.prototype, [
677   {
678     key: "constructor",
679     value: function () {
680       _classCallCheck(this, Constructor);
681     }
682   }
683 ]);
684
685 _createClass(Constructor.prototype, [
686   {
687     key: "constructor",
688     value: function () {
689       _classCallCheck(this, Constructor);
690     }
691   }
692 ]);
693
694 _createClass(Constructor.prototype, [
695   {
696     key: "constructor",
697     value: function () {
698       _classCallCheck(this, Constructor);
699     }
700   }
701 ]);
702
703 _createClass(Constructor.prototype, [
704   {
705     key: "constructor",
706     value: function () {
707       _classCallCheck(this, Constructor);
708     }
709   }
710 ]);
711
712 _createClass(Constructor.prototype, [
713   {
714     key: "constructor",
715     value: function () {
716       _classCallCheck(this, Constructor);
717     }
718   }
719 ]);
720
721 _createClass(Constructor.prototype, [
722   {
723     key: "constructor",
724     value: function () {
725       _classCallCheck(this, Constructor);
726     }
727   }
728 ]);
729
730 _createClass(Constructor.prototype, [
731   {
732     key: "constructor",
733     value: function () {
734       _classCallCheck(this, Constructor);
735     }
736   }
737 ]);
738
739 _createClass(Constructor.prototype, [
740   {
741     key: "constructor",
742     value: function () {
743       _classCallCheck(this, Constructor);
744     }
745   }
746 ]);
747
748 _createClass(Constructor.prototype, [
749   {
750     key: "constructor",
751     value: function () {
752       _classCallCheck(this, Constructor);
753     }
754   }
755 ]);
756
757 _createClass(Constructor.prototype, [
758   {
759     key: "constructor",
760     value: function () {
761       _classCallCheck(this, Constructor);
762     }
763   }
764 ]);
765
766 _createClass(Constructor.prototype, [
767   {
768     key: "constructor",
769     value: function () {
770       _classCallCheck(this, Constructor);
771     }
772   }
773 ]);
774
775 _createClass(Constructor.prototype, [
776   {
777     key: "constructor",
778     value: function () {
779       _classCallCheck(this, Constructor);
780     }
781   }
782 ]);
783
784 _createClass(Constructor.prototype, [
785   {
786     key: "constructor",
787     value: function () {
788       _classCallCheck(this, Constructor);
789     }
790   }
791 ]);
792
793 _createClass(Constructor.prototype, [
794   {
795     key: "constructor",
796     value: function () {
797       _classCallCheck(this, Constructor);
798     }
799   }
800 ]);
801
802 _createClass(Constructor.prototype, [
803   {
804     key: "constructor",
805     value: function () {
806       _classCallCheck(this, Constructor);
807     }
808   }
809 ]);
810
811 _createClass(Constructor.prototype, [
812   {
813     key: "constructor",
814     value: function () {
815       _classCallCheck(this, Constructor);
816     }
817   }
818 ]);
819
820 _createClass(Constructor.prototype, [
821   {
822     key: "constructor",
823     value: function () {
824       _classCallCheck(this, Constructor);
825     }
826   }
827 ]);
828
829 _createClass(Constructor.prototype, [
830   {
831     key: "constructor",
832     value: function () {
833       _classCallCheck(this, Constructor);
834     }
835   }
836 ]);
837
838 _createClass(Constructor.prototype, [
839   {
840     key: "constructor",
841     value: function () {
842       _classCallCheck(this, Constructor);
843     }
844   }
845 ]);
846
847 _createClass(Constructor.prototype, [
848   {
849     key: "constructor",
850     value: function () {
851       _classCallCheck(this, Constructor);
852     }
853   }
854 ]);
855
856 _createClass(Constructor.prototype, [
857   {
858     key: "constructor",
859     value: function () {
860       _classCallCheck(this, Constructor);
861     }
862   }
863 ]);
864
865 _createClass(Constructor.prototype, [
866   {
867     key: "constructor",
868     value: function () {
869       _classCallCheck(this, Constructor);
870     }
871   }
872 ]);
873
874 _createClass(Constructor.prototype, [
875   {
876     key: "constructor",
877     value: function () {
878       _classCallCheck(this, Constructor);
879     }
880   }
881 ]);
882
883 _createClass(Constructor.prototype, [
884   {
885     key: "constructor",
886     value: function () {
887       _classCallCheck(this, Constructor);
888     }
889   }
890 ]);
891
892 _createClass(Constructor.prototype, [
893   {
894     key: "constructor",
895     value: function () {
896       _classCallCheck(this, Constructor);
897     }
898   }
899 ]);
900
901 _createClass(Constructor.prototype, [
902   {
903     key: "constructor",
904     value: function () {
905       _classCallCheck(this, Constructor);
906     }
907   }
908 ]);
909
910 _createClass(Constructor.prototype, [
911   {
912     key: "constructor",
913     value: function () {
914       _classCallCheck(this, Constructor);
915     }
916   }
917 ]);
918
919 _createClass(Constructor.prototype, [
920   {
921     key: "constructor",
922     value: function () {
923       _classCallCheck(this, Constructor);
924     }
925   }
926 ]);
927
928 _createClass(Constructor.prototype, [
929   {
930     key: "constructor",
931     value: function () {
932       _classCallCheck(this, Constructor);
933     }
934   }
935 ]);
936
937 _createClass(Constructor.prototype, [
938   {
939     key: "constructor",
940     value: function () {
941       _classCallCheck(this, Constructor);
942     }
943   }
944 ]);
945
946 _createClass(Constructor.prototype, [
947   {
948     key: "constructor",
949     value: function () {
950       _classCallCheck(this, Constructor);
951     }
952   }
953 ]);
954
955 _createClass(Constructor.prototype, [
956   {
957     key: "constructor",
958     value: function () {
959       _classCallCheck(this, Constructor);
960     }
961   }
962 ]);
963
964 _createClass(Constructor.prototype, [
965   {
966     key: "constructor",
967     value: function () {
968       _classCallCheck(this, Constructor);
969     }
970   }
971 ]);
972
973 _createClass(Constructor.prototype, [
974   {
975     key: "constructor",
976     value: function () {
977       _classCallCheck(this, Constructor);
978     }
979   }
980 ]);
981
982 _createClass(Constructor.prototype, [
983   {
984     key: "constructor",
985     value: function () {
986       _classCallCheck(this, Constructor);
987     }
988   }
989 ]);
990
991 _createClass(Constructor.prototype, [
992   {
993     key: "constructor",
994     value: function () {
995       _classCallCheck(this, Constructor);
996     }
997   }
998 ]);
999
1000 _createClass(Constructor.prototype, [
1001   {
1002     key: "constructor",
1003     value: function () {
1004       _classCallCheck(this, Constructor);
1005     }
1006   }
1007 ]);
```



●感覚的に開発できるUI

【プロジェクトエリア】

ロボットプロジェクトをツリー構造で展開/収束して表示します。

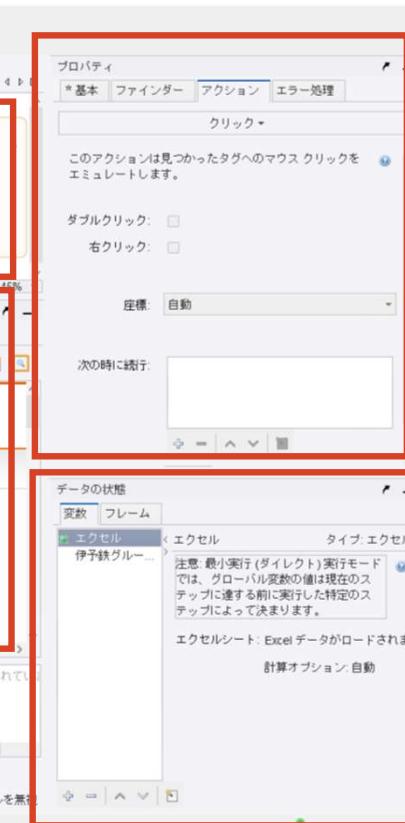


【モード切替】

デザイン：作成時利用
デバッグ：確認時利用

【ロボットビュー】

ロボットを構成するステップとフローが表示されます。このビューでステップの削除/移動、または接続などを行い、ロボットの構造を設定します。



【ステップビュー】

ステップに設定されたアクションやエラー処理の内容が表示されます。

【変数ビュー】

ロボットで使用する変数のリストが表示されます。ロボットを実行した際、カレントステップの変数値が表示されます。

【ウインドウズビュー】 現在のロボット状態におけるウィンドウのページが同期されて表示されます。

●安定した動作

「システムの作りを解析」「端末に依存しない動作」

⇒ロボットの共有や端末の変化に強い

位置が変わると操作不可

他社製品

画像認識・座標認識



- ・端末毎に修正が必要
 - ・ロボットの共有利用が困難
- ↓
限定的な利用
スケール△



位置が変わっても認識可能

BizRobo!

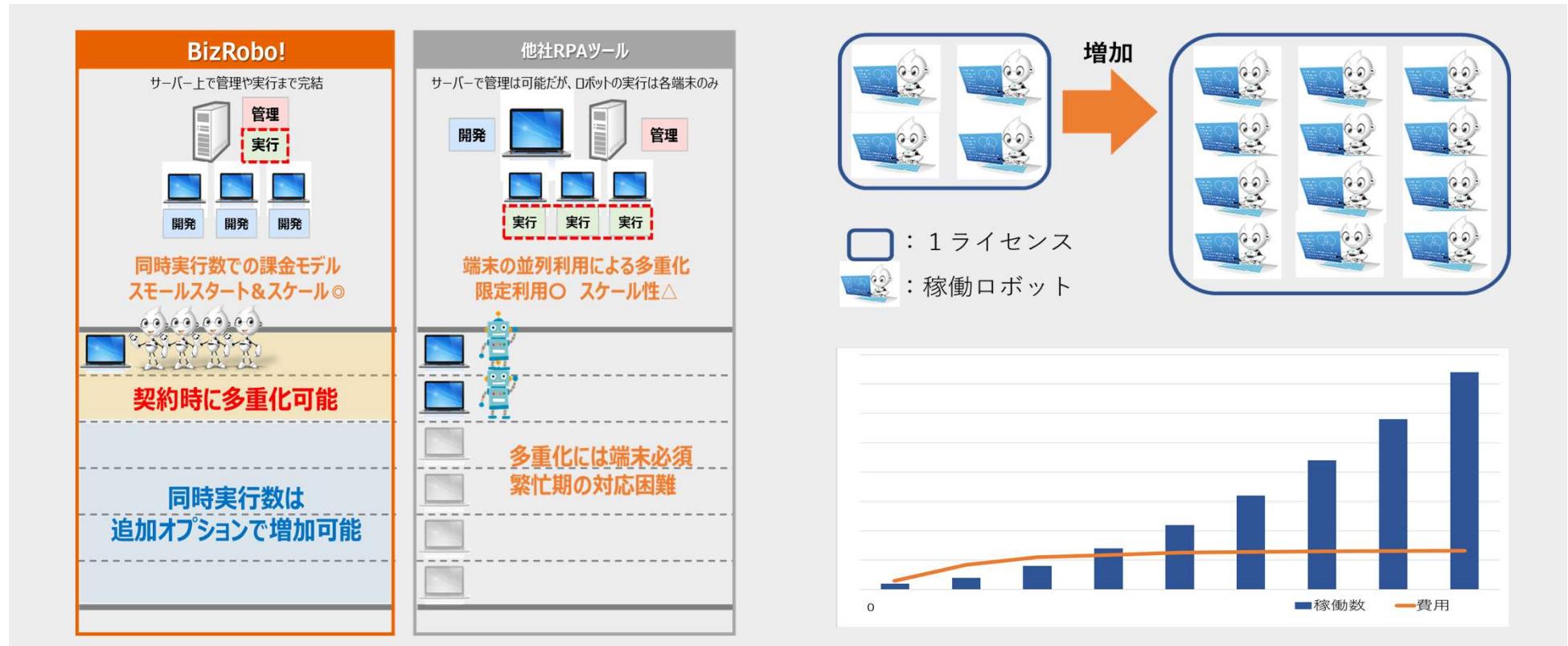
項目の存在を認識
(オブジェクト)



- ・端末毎の修正が不要
 - ・ロボットの複製が簡単
- ↓
どんな環境でも安定
スケール◎

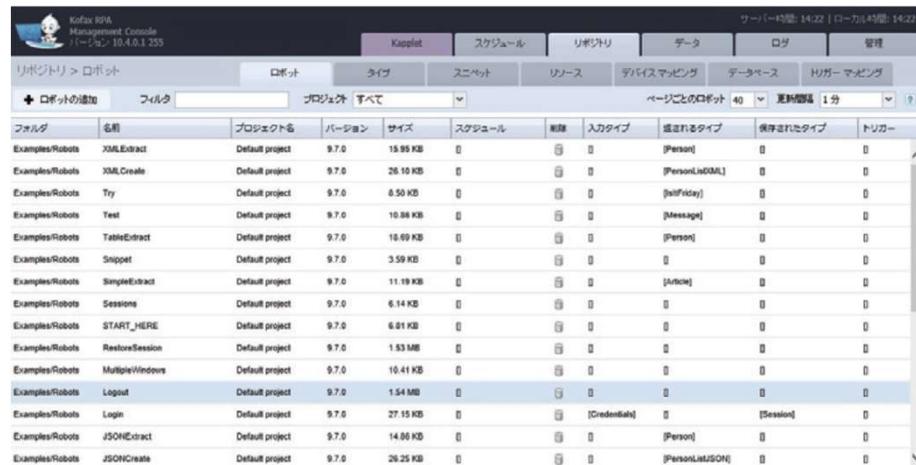
●フローティングライセンス

- 開発端末数や実行端末数が増加してもロボットを何体作成してもライセンス料は変わりません。
- より多くの方に触れて頂くことに適したライセンス体系です。



●管理機能搭載

自動実行機能やユーザー管理が可能です。ブラウザからの実行をすることもできます。



Kofax RPA Management Console リポジトリ > ロボット Kapplet タイプ スケジュール リポジトリ データ ログ 管理 ロボットの追加 フィルタ プロジェクト:すべて ページごとのロボット:40 更新間隔:1分

フォルダ	名前	プロジェクト名	バージョン	サイズ	スケジュール	実行	入力タイプ	返却されたタイプ	トリガー
Examples/Robots	XMLExtract	Default project	9.7.0	15.99 KB			[Person]		
Examples/Robots	XMLCreate	Default project	9.7.0	26.10 KB			[PersonListXML]		
Examples/Robots	Try	Default project	9.7.0	8.50 KB			[HalfFriday]		
Examples/Robots	Test	Default project	9.7.0	10.88 KB			[Message]		
Examples/Robots	TableExtract	Default project	9.7.0	15.00 KB			[Person]		
Examples/Robots	Snippet	Default project	9.7.0	3.59 KB					
Examples/Robots	SimpleExtract	Default project	9.7.0	11.19 KB			[Article]		
Examples/Robots	Sessions	Default project	9.7.0	6.14 KB					
Examples/Robots	START_HERE	Default project	9.7.0	6.01 KB					
Examples/Robots	RestoreSession	Default project	9.7.0	1.53 MB					
Examples/Robots	MultipleWindows	Default project	9.7.0	10.41 KB					
Examples/Robots	Logout	Default project	9.7.0	1.54 MB					
Examples/Robots	Login	Default project	9.7.0	27.15 KB			[Credentials]	[Session]	
Examples/Robots	JSONExtract	Default project	9.7.0	14.00 KB			[Person]		
Examples/Robots	JSONCreate	Default project	9.7.0	26.25 KB			[PersonListJSON]		



新しいユーザーを作成

ユーザー名	user01
パスワード	*****
再入力	*****
フルネーム	ユーザー
電子メールアドレス	user01@test.com
所属するグループ	選択可能なグループ
グループ	

キャンセル 保存



KAPPZONE NEWLY ADDED UPDATED A-Z Search ADD NEW KAPPLER ☰

KAPPLETS

ファイル監視ロボ INSTALL	株価取得ロボ OPEN	電卓で計算ロボ OPEN
ファイルの有無を監視する	株価を収集する	電卓を操作して計算を行うロボット

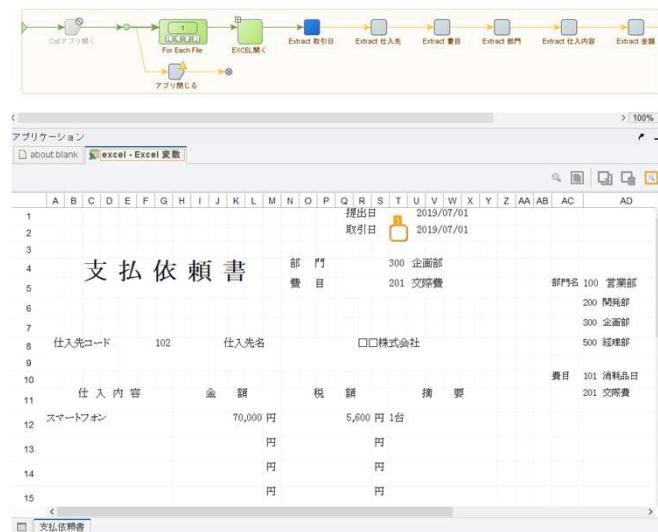
BizRobo!

●バックグラウンド処理

他社製品では、ロボット稼働中に、人が意図せずにキーボードやマウスに触れると、ロボットが止まってしまう可能性があります。



BizRobo!が実現する「バックグラウンド処理」では、ソースコードを直接読み取り、処理を実施していくため、高速かつ安定的にロボットが稼働します。



じつは裏で働いてます！



PC上の単純作業を
人間の代わりに作業する
次世代の事務処理ロボット

人間の作業や判断を
自分でロボットに教えて
作業を自動化

ロボットの数は無制限

1ライセンスで

何体でも稼働可能

企業が抱える問題（①人口減少、②長時間労働、③低労働生産性）に対するソリューションのひとつになり得る
→DXへの第一歩へ

代行

人間のルーチン作業を代行できる
人間の貴重なエネルギーをルーチン作業から解放

能力

人間と比べて、圧倒的な作業スピードと正確性
24時間365日働き続け、辞めることもない、複製も可能

変化

システムでは即応できない、事業・業務の変化のスピードに柔軟かつスピーディーな対応が可能



＼こんな企業様にオススメ／

- ✓ システム間の連携に課題を感じている
- ✓ 業務システムの改修を検討している
- ✓ 同じような内容を何度も入力・出力している
- ✓ まずは業務の棚卸しをしたい
- ✓ 従業員のITリテラシーを向上させたい

生成AIとは

■生成AIとは？ さまざまなコンテンツを新たに生み出す人工知能（AI）

生成 AI の種類

テキスト生成

テキストで質問や指示を入力すると、生成AIが内容に応じた回答を自動生成する。



画像生成

テキストで指示入力すると、生成AIがイメージに近いオリジナルの画像を生成する。



動画生成

画像やテキストを入力すると、生成AIが指示や内容に応じた短い動画を生成する。



音声生成

音声・テキストを入力すると、生成AIが特徴を学習して新たな音声データを生成する。



☑ 業務効率化

- ・集計や分析をAIに任せることで、作業時間を大幅に短縮
- ・人間の主觀による偏りを排除し、客観的なデータ分析が可能
- ・自動要約によるレポート作成支援
- ・議事録作成

☑ コミュニケーション・意思決定支援

- ・セミナーやイベントの資料生成支援
- ・顧客向けのメールやFAQの自動生成
- ・アンケートから読み取れる課題や傾向の抽出
- ・データ分析結果に基づくアクションプランの提案

重要度 低 × 優先度 高 = AIの使い道

フル活用して、効率化や質の向上を図りたい部署は、アイデア豊富で手が早い新入社員が入ったつもりで、どんどん協業をしましょう。

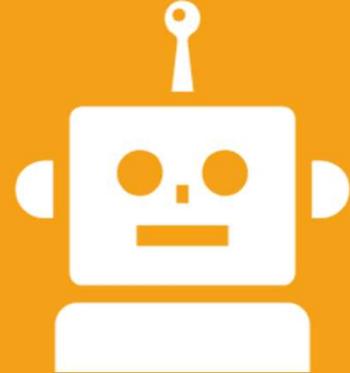
役割	生成AIの活用方法
総務	<ul style="list-style-type: none"> ・社内ポリシーの整備 ・社内イベントの企画
経理	<ul style="list-style-type: none"> ・資料作成のサポート（社内説明資料、報告書、マニュアルなど）
法務	<ul style="list-style-type: none"> ・契約書のドラフト作成とレビュー ・法的リスクの分析
人事	<ul style="list-style-type: none"> ・求人票の作成 ・社員研修プログラムの作成サポート ・人事制度策定のサポート
営業	<ul style="list-style-type: none"> ・企業情報調査/収集 ・仕様書案作成 ・商談議事録作成 <p>⇒ 内容に基づいたアクション設定・メール送信</p>

経営課題の発見および解決に際して、戦略コンサルを用いるようなフレームワークを作成することが可能になります。

外部の専門家に依頼するしか無かった業務も、内製化しクイックに行うことができ、自社の経営レベルを一段階向上できます。

役割	生成AIの活用方法
経営者・役員	<ul style="list-style-type: none"> ・意思決定サポート（メリデメ洗出し等） ・戦略立案/メンター役 ・ビジネスアイデア出し ・スピーチやプレゼン素案作成
経営企画	<ul style="list-style-type: none"> ・事業計画ドラフト策定 ・企画書/申請書ドラフト作成 ・経営分析 ・市場調査サポート
マネージャー	<ul style="list-style-type: none"> ・プロジェクト管理サポート (タスク分解、ネクストアクション、スケジュール、担当割) ・1on1やチーム課題への助言 ・会議のアジェンダ出し

RPA



「人の手足」のように、
作業を自動化するもの

人間が条件やルールを
設定します

決められたルールや条件に
よって処理を実行します

AI



「人の脳」のように、
思考支援をするもの

膨大なデータを元に
自ら学習します

膨大なデータを元に
予測・分析・認識をします

■RPA×AIで製造ラインを最適化

AIで製造ラインの機械トラブルやミスを自動で検知
検知内容をRPAがアラートや警報として流すことで、トラブルの防止が可能

■RPA×AIで環境分析を最適化

RPAで日々の環境データ（温度・湿度・天気など）を記録
記録した内容をAIが分析し日々の予測を最適化できます。

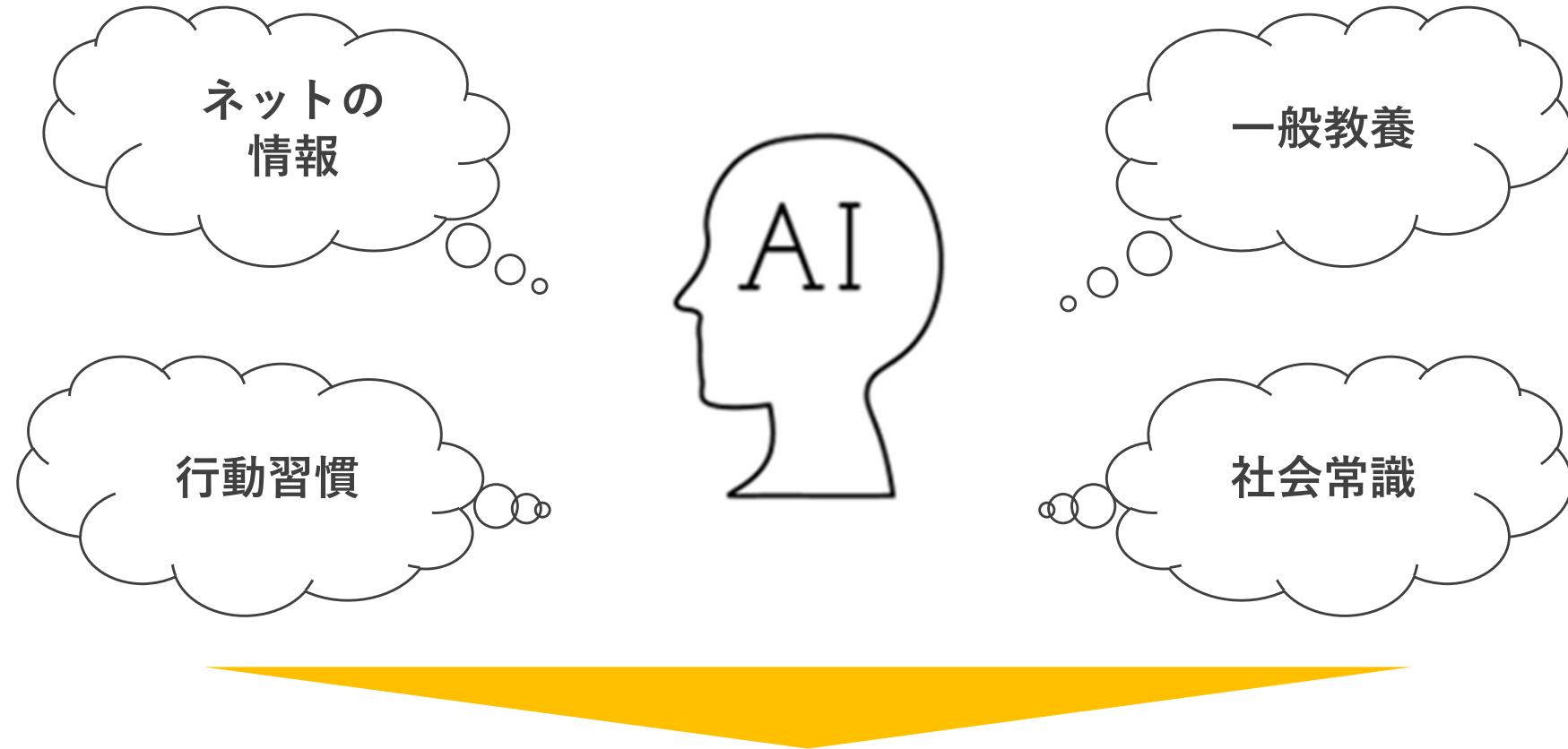
■RPA×AIで営業を最適化

RPAで顧客データ（業種や業界ごとの売上など）を収集
AIで最適な人員配置を行い、顧客の属性を踏まえたうえで、最適な営業戦略を提案

■RPA×AIで事務作業を最適化

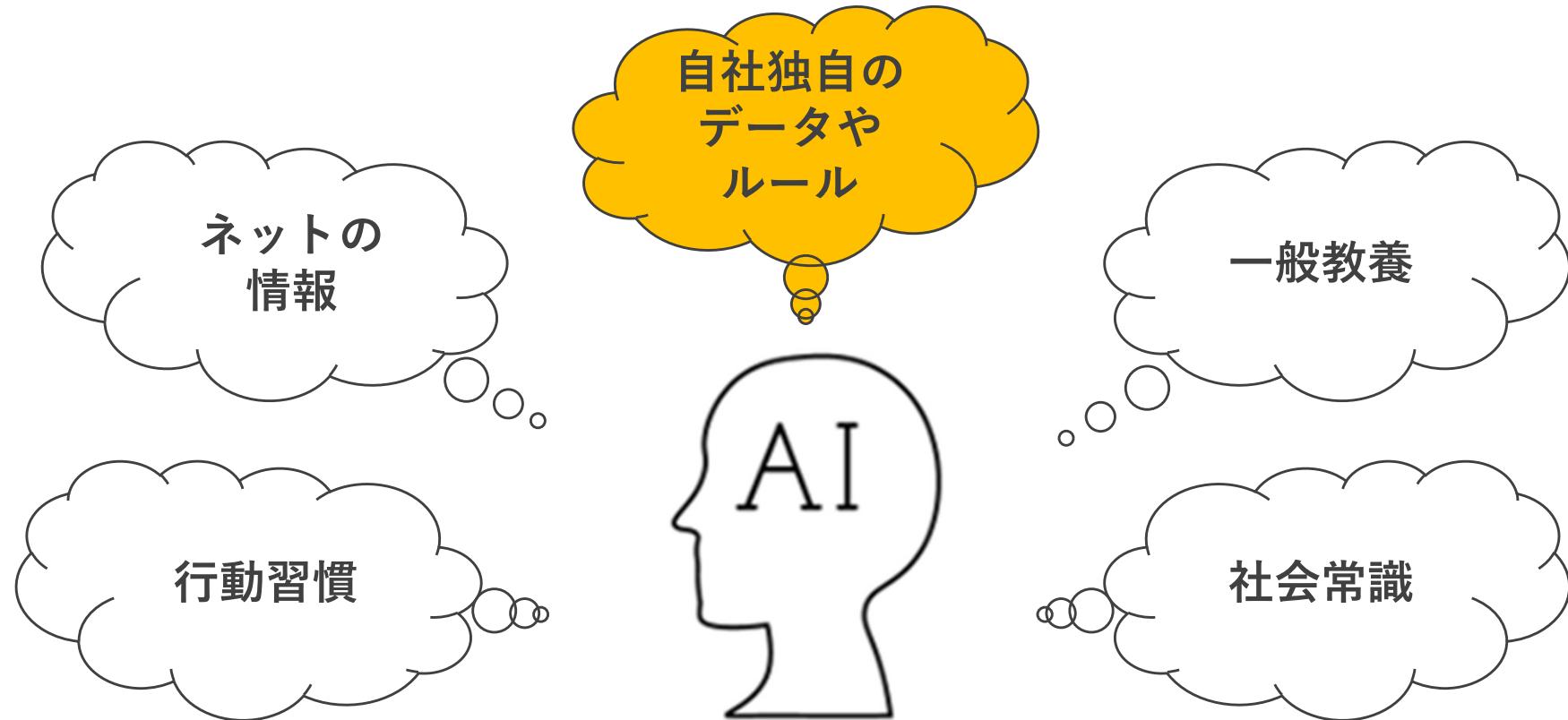
AIで紙の帳票を読み取（定型帳票及び非定型帳票）
読み取った情報を人が確認して、RPAでシステムに自動入力

広い情報源や学習データからコンテンツの生成が可能



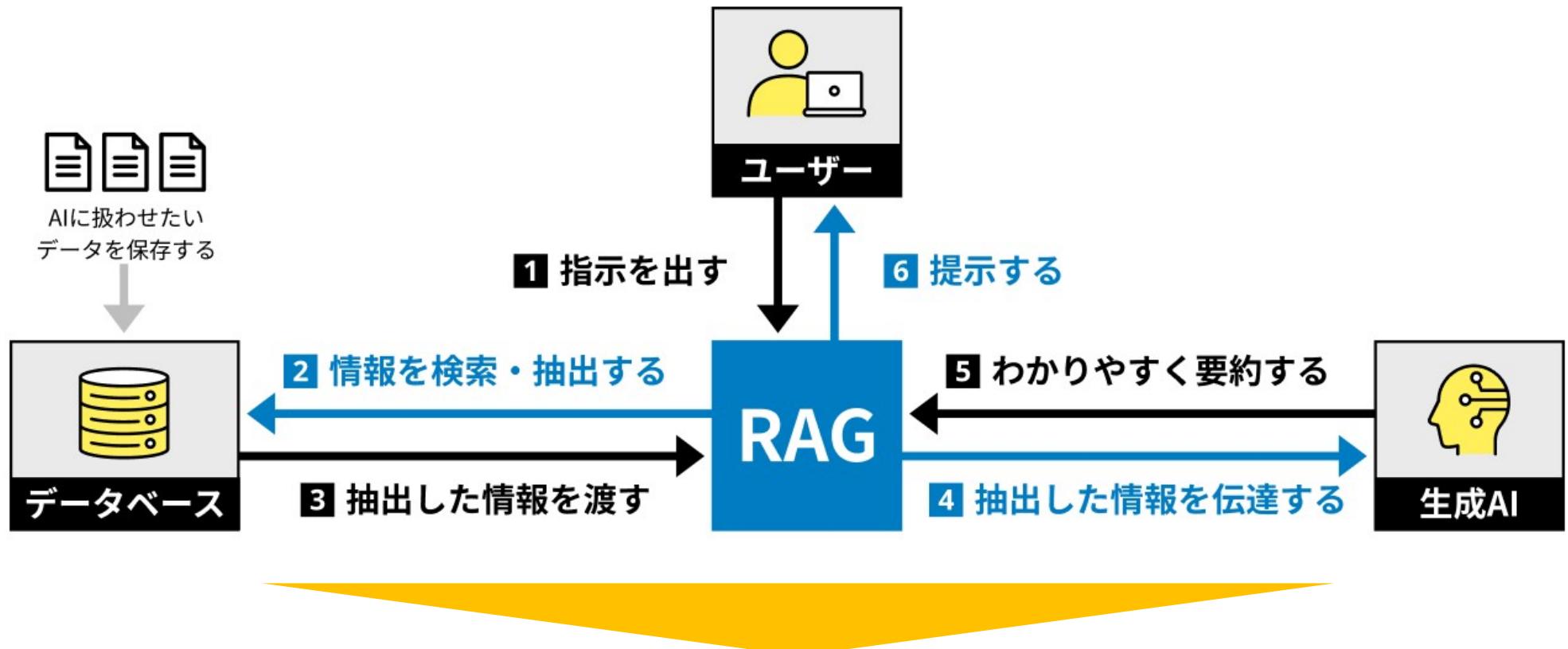
利用難易度が低く、且つ汎用性が非常に高い一方で、
クラウドサービスの為、内部データを利用することが出来ないので
業務利用の範囲が限定的となってしまう

RAGとは...ロボティック・プロセス・オートメーション (Retrieval-Augmented Generation)



従来のAIの頭脳・知識に加え、オリジナルの学習データを元に
コンテンツを作成する

RAGと生成AIを組み合わせて構築することで 自社専用のAIアシスタントが誕生



自社のデータを活用することで、
常に最新かつ正確な情報のため、生成結果の信憑性・確実性が高まり、
パーソナライズされた情報を取得することが可能に

生成AIとRAGの違いについて

	メリット	デメリット
生成AIのみ	<ul style="list-style-type: none">・非エンジニアでも実行可能・コストが低い	<ul style="list-style-type: none">・自社独自の情報や専門領域での学習が不足する
生成AI×RAG	<ul style="list-style-type: none">・膨大なデータ量の学習が可能・自社の最新データに基づく回答が可能・業界やタスクに特化可能	<ul style="list-style-type: none">・学習データの準備が必要

RAGを使ってできること

■社内問い合わせ対応

社内規定をデータベースに登録することで、AIが社内問い合わせに対応できるようになります。

担当者が専門的な知識やスキルを身につける必要なし。

問い合わせを受ける社員は、他の業務に専念でき、問い合わせる社員は、情報収集の効率化になります。

■コンテンツ作成

ChatGPTなどでも作成できるが、自社の形式沿ったコンテンツを生成させるには難しい。

RAGなら、参考資料をデータベースに登録し、その資料にもとづいた資料を作成可能。

■分析

顧客データや市場調査レポート、アンケート調査結果、財務諸表などをデータベースに登録すると、そのデータを使った分析を任せられる。

1. カルカワーズ
2. Tebot
3. ShareMind

カルクワーカス

生成AIを手軽で安価に！
誰でも使える中小企業向け
生成AIツール



【特徴】

- ・実務に即したアプリで誰でも簡単にAIの効果を体感！
- ・プロンプト不要で思った文章が完成！
- ・10年以上の支援ノウハウ AI活用のスタートを後押し！

【効果】

- ・議事録作成AIで、文字起こしから議事録作成。
- ・メール返信AIで、内容に応じたメール内容を生成。
- ・情報抽出AIで、長い文章から要点の切り出しが可能。

【費用】

**基本料：50,000円+ユーザー利用料：980円～（月額）
 （別途初期費用+API利用料）**

基本的なChat GPTをベースとして、 実務に適した19のアプリケーションを用意

Chat GPTの
基本機能



社内Chat GPT

書類作成

日常業務

思考支援

議事録

らくらく申請書作成
AI

タスクサポートAI

アイデア出しAI

らくらく文書作成AI

メール返信作成AI

タスク分解AI

SNS投稿作成AI

文章校正チェックAI

課題解決サポートAI

スライド原稿作成AI

適切表現・炎上
チェックAI

意思決定サポートAI

求人票作成AI

マインドマップ作成
AI

ディープ検索AI

NEW

議事録AI

メンタルヘルス

お悩み解決サポート
AI



誰でも数分で導入可能！
圧倒的に使いやすく低価格な
AIチャットボット



【特徴】

- ・直感的な操作/専門知識不要で誰でも簡単に運用可能
- ・必要な機能を厳選し、優れたコストパフォーマンス
- ・多様な対応方法と自動学習機能を搭載

【効果】

- ・窓口での問い合わせを自動化
- ・チャットボットを通じたユーザー行動や要望のデータ分析で、サービス改善や新たなインサイトの獲得が可能
- ・社内の問い合わせを自動化

【費用】

月額基本料：9,800円～60,000円

シナリオ型

設定シナリオを順番に選択する

Tebotについてチャットでお答えします

Tebot

よくご質問いただく内容はこれらです。

- 無料トライアルがしたい
- サービス資料が欲しい
- Tebotとは？
- 利用料金は？

利用料金は？

Tebot

利用料金に関するご質問ですね。
チャットボット利用者からフリーワード（自由記入形式）で質問を受け付けますか？

- フリーワードで対応したい
- 選択肢形式のみで対応したい

入力してEnterで送信

送信

○：クリックだけの簡単操作

✗：問合わせ対応できる領域が狭い

Q&A型

設定したQ&Aとマッチングする

Tebotについてチャットでお答えします

Tebot

料金プランを教えて

Tebot

以下の候補より選択してください。
ない場合は別の言葉で再度ご質問ください。

- どのようなプランがありますか
- フリープランはありますか
- お支払い方法はどのような種類がありますか？

この中に該当するものが無い

Tebot

Q. どのようなプランがありますか

Tebotは必要な機能が全て使えて初期費用0円、月額費用4.5万円で提供しております。
また、AI機能は使わずシナリオ機能と有人チャット機能のみをご利用いただけるライトプラン（月額9,800円）。お手頃料金でお問い合わせ

料金プランを教えて

送信

○：対応領域が広く生の声が集まる

✗：Q&Aマッチしないと回答不可

生成AI型

設定した資料からAIが回答文生成

Tebotについてチャットでお答えします

Tebot

料金プランを教えて

Tebot

Tebotの料金プランは以下の通りです：

1. “通常プラン”：
- 初期費用：0円
- 月額費用：45,000円
- 必要な機能が全て利用可能
2. “ライトプラン”：
- 月額費用：9,800円
- AI機能を使わず、シナリオ機能と有人チャット機能のみ利用可能

違いは？

Tebot

###違い
- “費用”：通常プランは月額45,000円、ライトプランは月額9,800円。
- “機能”：通常プランは全ての機能が利用可能ですが、ライトプランはAI機能を使わず、シナリオ機能と有人チャット機能のみ利用可能。

料金プランを教えて

送信

○：Q&A準備不要で対応領域が広い

✗：回答文の事前チェック不可

Share Mind

自社専用のAI環境
日常業務を劇的に改善する
業務特化AIアシスタント



【特徴】

- ・アップロードするだけで、誰でも簡単に自社専用の生成AIアシスタントが作成可能
- ・自社リソースを元に正確な回答を返し、学習データの原文表示で信頼性を担保。
- ・占有のサーバーを利用していているため、安心して業務データを取り込める環境

【効果】

- ・社内ヘルプデスクでの対応を自動化
- ・社内文書や規程の検索を効率化
- ・自社データを元にした文書生成やアイデア出しを実現

【費用】

月額利用料：100,000円
(別途初期費用)

業務特化の生成AIアシスタントが作り放題 日常業務をサポートする生成AIアシスタント

STEP 01 学習リソースをアップロード



STEP 02 フォルダを選択して質問を入力



STEP 03 質問に対する答えを生成AIが回答



日常業務で使用している業務マニュアルなどの専門知識をアップロードするだけで、あなたとチームをサポートする業務特化の生成AIアシスタントが作成可能です。

	カルクワーカス	Tebot	ShareMind
製品性質	文書生成AIツール	AIチャットボット	RAG構築
特徴	手軽に始められるAIツール	簡単設定・高品質なチャットボット	自社データを用いたAI活用
価格	50,000円/月～	9,800円/月～60,000円/月	100,000円/月～
課金対象	AI利用料 +ユーザー数	Bot数 その他オプション	AI利用料
利用シーン	文章生成や情報収集 アイデア出し	社内外からの問い合わせ対応 データ分析	自社データを元にした 文章生成や情報収集 アイデア出し 社内の問い合わせ

《DX支援提供ソリューション》

1 コンサルティング

- ・業務量調査
- ・製品選定/提案

3 運用支援

- ・お問い合わせ対応
- ・開発代行
- ・環境構築

2 パッケージ販売・導入支援

- ・パッケージ販売
- ・初期設定、システム構築

4 研修・OJT

- ・DXツールの操作研修
- ・個別のOJT

1. 都心部と同等の技術・知識

- ベンダーと密に連携しているため、ハイレベルな技術・知識が提供可能

2. 地方ならではの素早く親密なサポート

- お客様と近い距離で現地・リモート問わず、幅広いサポートが可能

3. 豊富な導入実績

- 民間・自治体での導入実績多数
- 全国区での導入事例も紹介可能

質疑応答



**DIGITAL
TECHNOLOGY**
SHIKOKU



P E R S O L

悩める情シスの課題と解決策 (オフェルタITソムリエとは)

パーソルコミュニケーションサービス株式会社

プロフィール



パーソルコミュニケーションサービス株式会社

まじま まさる
真嶋 優

- ・好きなもの：ワンコ
- ・趣味：アジリティ、ボウリング

<職歴>

TSTソフトウェア（現：インフォコムテクノロジーズ）
ハイテクシステム（NECシステムテクノロジーへ派遣）
コーポレートソフトウェア
(現：パーソルコミュニケーションサービス)

Agenda

- 情シスのあるべき姿
- 悩める情報システム部門（現状や課題など）
- パーソルコミュニケーションサービスができること
- オフェルタ ITソムリエ メニュー
- オフェルタ ITソムリエ 提供イメージ
- 弊社が選ばれるポイント

解決施策の最適化ポイント

情報システム部門を取り巻く環境

情報システム部門は社内・外との関わりも多く、そのさまざまな環境からニーズや要望など多く寄せられ、スピード感のある適切な対応が求められます。そこには整理や優先順位付けなど適切な対応が必要になってきます。

しかしながら、限られた人材やリソースで多くのことに対応していくことは非常に難しいことです。

そこで弊社がそれらの課題や業務内容の整理をすることで、最適化実現に向けた的確な運用を提案します。



運用の最適化

根本課題の可視化が可能



業務の可視化

コアとノンコア業務の可視化と業務のすみ分けとノンコア部分のアウトソース可能



従業員体験価値（EX）向上

従業員の業務負荷軽減とコア業務への集中



経営層

- ・売上向上
- ・コスト削減
- ・効率化
- ・品質橋上
- ・デジタル化



ベンダー

- ・関係強化
- ・自社の業務に関する理解
- ・効率化
- ・コスト削減



責任者様



社内

- ・情報共有
- ・連携
- ・社員満足度向上
- ・ITリテラシーの向上



スタッフ

- ・効率化
- ・品質橋上
- ・モチベーションの維持・向上
- ・スキルアップ
- ・コア業務への集中

情シスの体制について

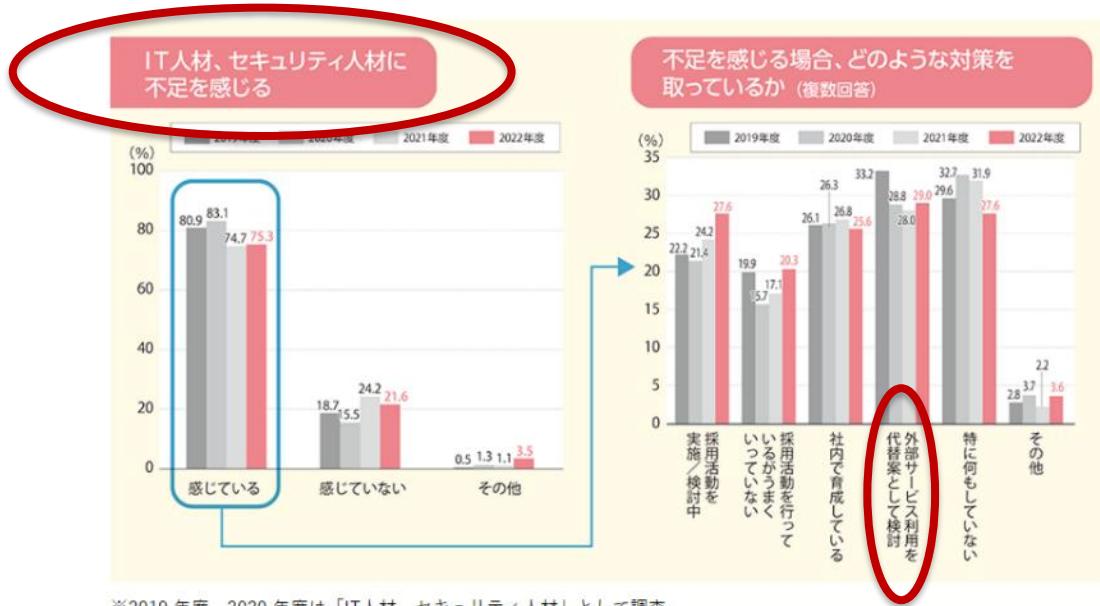
- 2~4名程度の情シス体制が多数派



出典：株式会社ソフトクリエイト 数字で見る“情シス”の実像2023
「情報システムの現状とIT活用実態アンケート2023」調査結果の概要より
<https://www.softcreate.co.jp>

情シスの人材について

- 4人中3人が情シスの不足を訴える
- 人員不足の対策としては「外部サービス利用を代替案として検討している」増加傾向

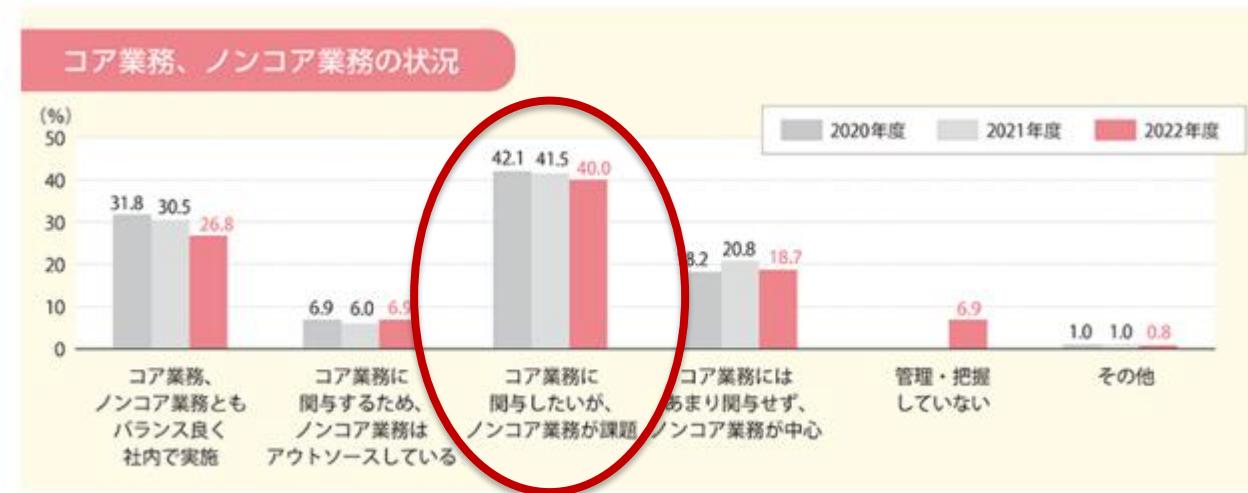


※2019年度、2020年度は「IT人材、セキュリティ人材」として調査。

出典：株式会社ソフトクリエイト 数字で見る“情シス”的実像2023
「情報システムの現状とIT活用実態アンケート2023」調査結果の概要より
<https://www.softcreate.co.jp>

情シスの業務について

- コア業務に関与したいが、ノンコア業務をどうすればよいのか？
- ノンコア業務に課題ある



出典：株式会社ソフトクリエイト 数字で見る“情シス”の実像2023
「情報システムの現状とIT活用実態アンケート2023」調査結果の概要より
<https://www.softcreate.co.jp>

情シスが今後注力すべきと考える業務について

- コア業務への転換や専念と回答する企業は多いが…
- クラウド化後のセキュリティ対策なども増加傾向
- 人材不足への対策についても増加傾向

情シスが今後、注力すべきと考えている(注力している最中の)活動(複数回答・年次推移)



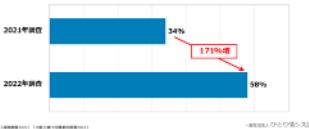
出典：株式会社ソフトクリエイト 数字で見る“情シス”の実像2023
「情報システムの現状とIT活用実態アンケート2023」調査結果の概要より
<https://www.softcreate.co.jp>

【ご参考】ひとり情シスの現状

一般社団法人ひとり情シス協会 2022年度調査速報値より

ひとり情シス実態調査2022・総合

ひとり情シス企業の旺盛な増員意向



ひとり情シス企業の旺盛な増員意向

従業員数100名から500名の「ひとり情シス」企業は32%と昨年より微増である。しかし、58%のひとり情シス企業で昨年を大きく上回る増員意向を持っている実態が判明した。コロナ禍でのテレワーク環境への準備、デジタル化への対応など業務量が増えていることと、働き方改革の浸透により実働時間減少などで情シsstaff増員の必要性を感じている。

ひとり情シス実態調査2022・総合

7割の企業で情シス職の採用が難航

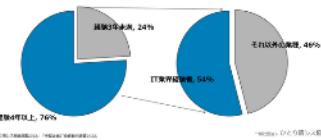


7割の企業で情シス職の採用が難航

情シスの採用に積極的になる反面、**極めて採用が難しいと感じる企業が36%**など、実際の採用活動で苦心している姿が報告された。**主な理由は給与面、必要スキルのアンマッチ。**

ひとり情シス実態調査2022・総合

ジュニアひとり情シスの増加



経験3年未満のひとり情シスの増加

ベテランひとり情シスの定年退職や転職などの後任として情シス経験の浅いひとり情シスが増加している。ひとり情シスの24%が3年未満の経験であることが判明。社内の管理部門や技術部門から異動して着任する場合と、IT業界の勤務経験者の転職が54%を占めた。

ひとり情シス実態調査2022・総合

パートナーを積極活用する計画増



半数以上の企業がパートナーを今後積極活用

ひとり情シスの業務量増大に伴い外部パートナーの積極活用の方針が強まっている。比較的軽度なPCクライアント管理やヘルプデスクなどのアウトソーシング活用よりも、プロジェクト管理やBCP環境構築などのプロフェッショナルワークの外部委託意向が鮮明になった。

社団法人ひとり情シス協会 <https://promit.gr.jp/>

ひとり情シスの実態調査、コンテンツ開発、情報交換、育成など
事務局 清水 博（書籍「ひとり情シス」の著者）

設立 2020年7月1日

悩める情報システム部門

調査から見えてくる課題

人材の課題

- ・ 少数精鋭の体制が増加している。
- ・ 人材を増やす予定だが、マッチした人材が採用できない。
- ・ 1人あたりの業務負荷が大きい。
- ・ 多忙すぎて人材育成まで手が回らない。

業務の課題

- ・ 情シスの業務範囲は幅広く多岐にわたる。
- ・ コア業務・ノンコア業務のすみ分けがはっきりしていない。
- ・ 担当者がコア業務に専念できない。
- ・ 業務が属人化している。

社内/外環境の課題

- ・ 経営からDXなどへの対応を急がされている。
- ・ 社内の部門でシャドーITが急増している。
- ・ 社内でなかなか情シスの取り組みを理解してもらえない。
- ・ ベンダーになかなか自社の状況を理解してもらえない。

セキュリティ/コンプライアンスの課題

- ・ 社内で不正行為が急増している。
- ・ 社内でセキュリティ事故が増加傾向にある。
- ・ 社内のITリテラシーを向上させたい。
- ・ もっとセキュリティに関する施策を検討したい。

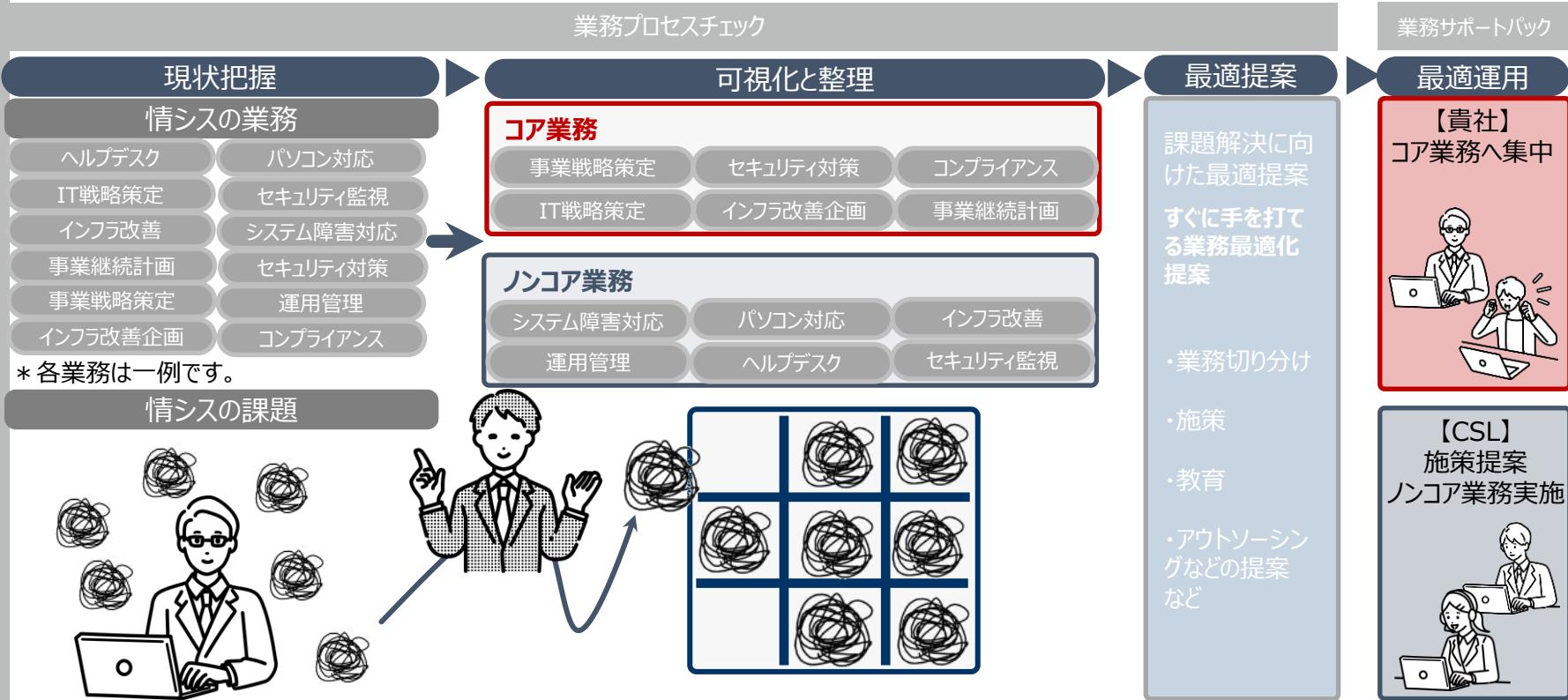
情報システム部門の“機能”的な一部として

企業の情シス部門に求められる役割は、既存システムの維持や管理だけではなく、ICTの効率的な利活用からDX推進による事業変革までさまざまな領域に拡大しています。そのような中で、人材不足問題は年々深刻化しています。

私たちパーソルコミュニケーションサービス（CSL）は、担当者の業務負荷の軽減によりコア業務への集中を実現し、さらに業務効率化やセキュリティ強化をはじめとした、情シスが抱えるあらゆる課題解決を通じ、お客様の情シス業務の標準化や、最適化などさまざまな課題解決を支援します。

お客様に最適なソリューションを提供すると共に、お客様が担うべき情シス業務への集中を全面的に支援します。

オフェルタ ITソムリエとは



ソリューションフロー

1

現状把握

業務の把握

- ・IT資産管理状況
- ・社内のIT活用状況
- ・IT業務の運用状況
- ・上記現状調査
- ・ドキュメント類などのご提示
(マニュアルなどの有無)
- ・ありたい姿の確認

2

課題・業務
整理

既存業務整理と 運用確認

- ・生産性集計
- ・工数集計
- ・業務プロセス整理
- ・社内連携の整理
- ・外部環境の整理
- ・ありたい姿の具現化に向けて
- ・業務エリア内ラウンド
- ・工数計測
- ・インタビュー

3

施策提案

業務振分・最適運用 のための提案

- 現状の報告
- ありたい姿の実現に向けた
提案
- 優先順位つけ
 - ・今すぐにアウトソースすべき業務など

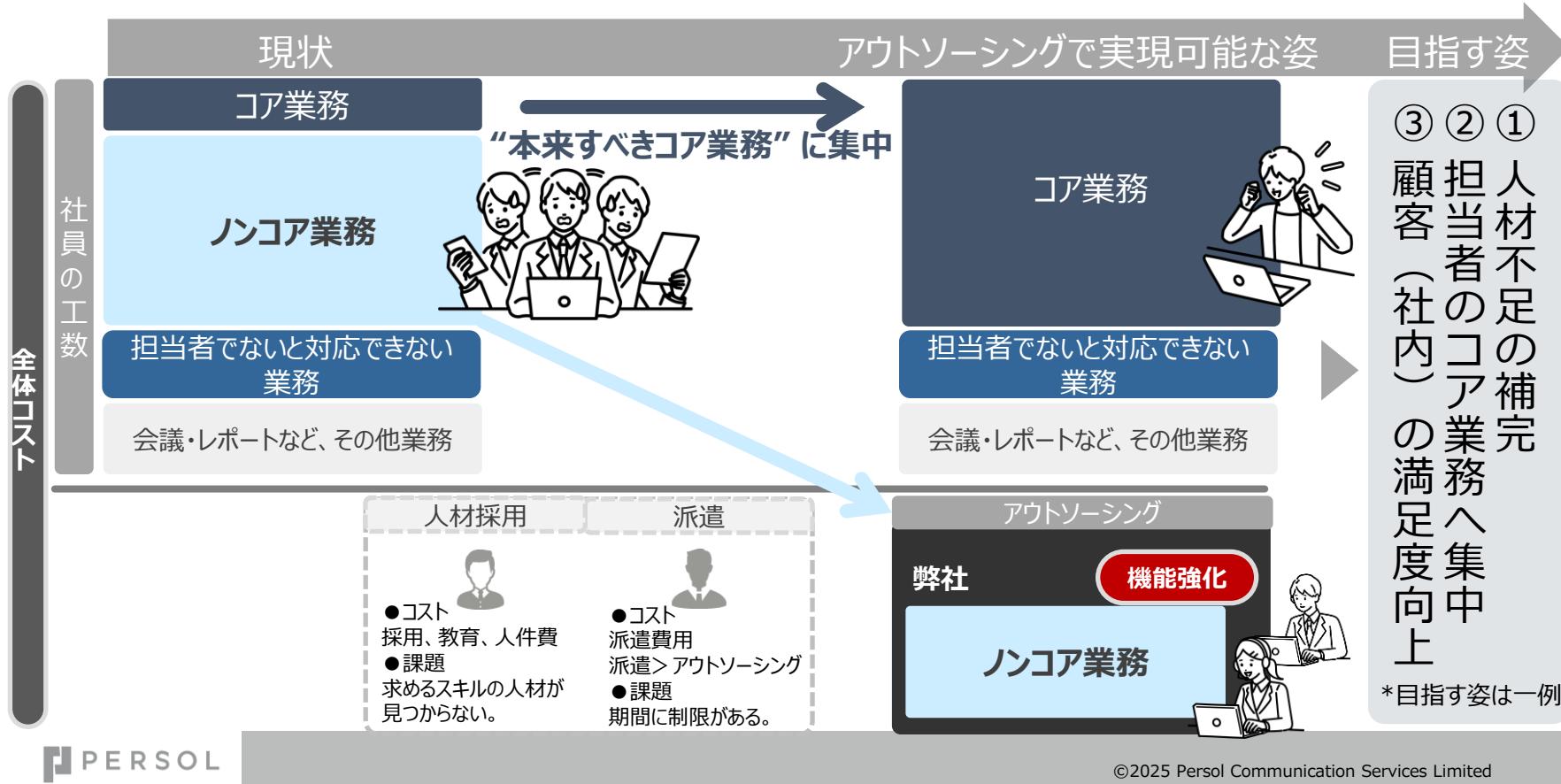
4

運用開始

運用の開始と 継続的な改善

- 段階的にアウトソースを
拡大可能
- 月次報告会を実施
 - ・改善提案
 - ・ドキュメント化すべき作業タスクの報告
- 継続的な改善活動

オフェルタ ITソムリエのポイント



業務サポートパック

必要なものだけ、選んでスマートスタートが可能です。



社員サポート

- ・有人ヘルプデスク（電話、メール、チャット）
- ・デジタルヘルプデスク（ボット、FAQ、tips）
- ・社員IT教育
(教育実施・マニュアル、コンテンツ作成)
- ・満足度調査
- ・VIP対応



IT資産マネジメント

- ・デバイス管理
- ・キッティング
 - ① 通常使用、予備機の管理
 - ② デバイス更改のイベント
- ・障害対応（1次受付・マルベン）
- ・ライセンス管理
- ・各種マニュアル作成



インフラ運用

- ・トラブル対応（1次受付）
- ・サーバ/ネットワーク監視
- ・リソース管理
- ・ユーザーからの声を収集
- ・変更管理、リスク管理
- ・最適化、自動化の支援



セキュリティ

- ・ユーザー向けアナウンス
- ・ハードウェア、ソフトウェアの利用状況/最新化の管理
- ・継続的改善モデル
- ・全社教育
(教育実施・対応マニュアル、啓発コンテンツ作成)



ユーザ管理

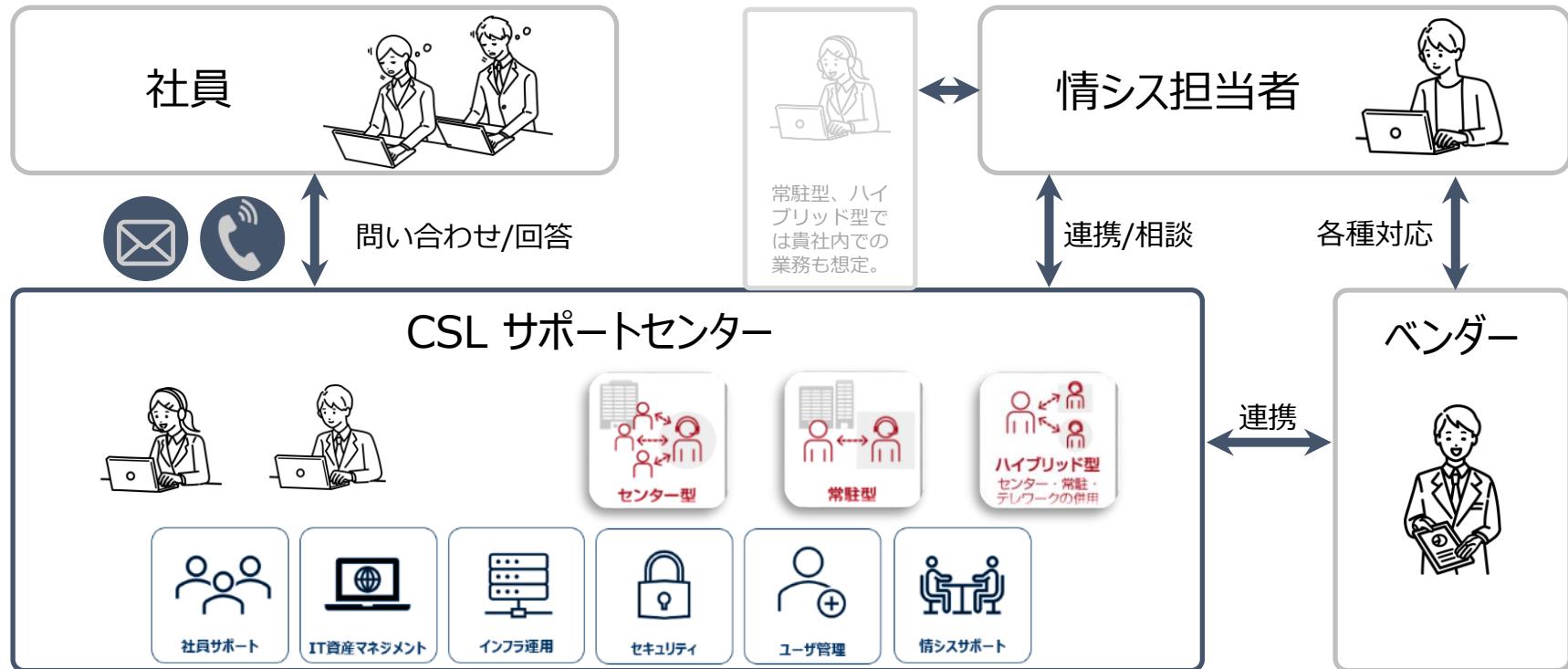
- ・ユーザー、組織の変更管理
- ・ユーザー向けアナウンス
- ・ハードウェア（PC、携帯、周辺機器）の所有
- ・ソフトウェアの所有
- ・サイト、フォルダのアクセス権限
- ・ユーザー体験価値の配信
- ・サービス、カタログ管理



情シスアドバイス

- ・問い合わせ傾向より、セルフ解決の促進
- ・対応時間の短縮
- ・デジタルヘルプデスクの利用促進
- ・ツール導入支援
- ・Microsoft 365 導入支援/活用アドバイス

オフェルタ ITソムリエ 提供イメージ



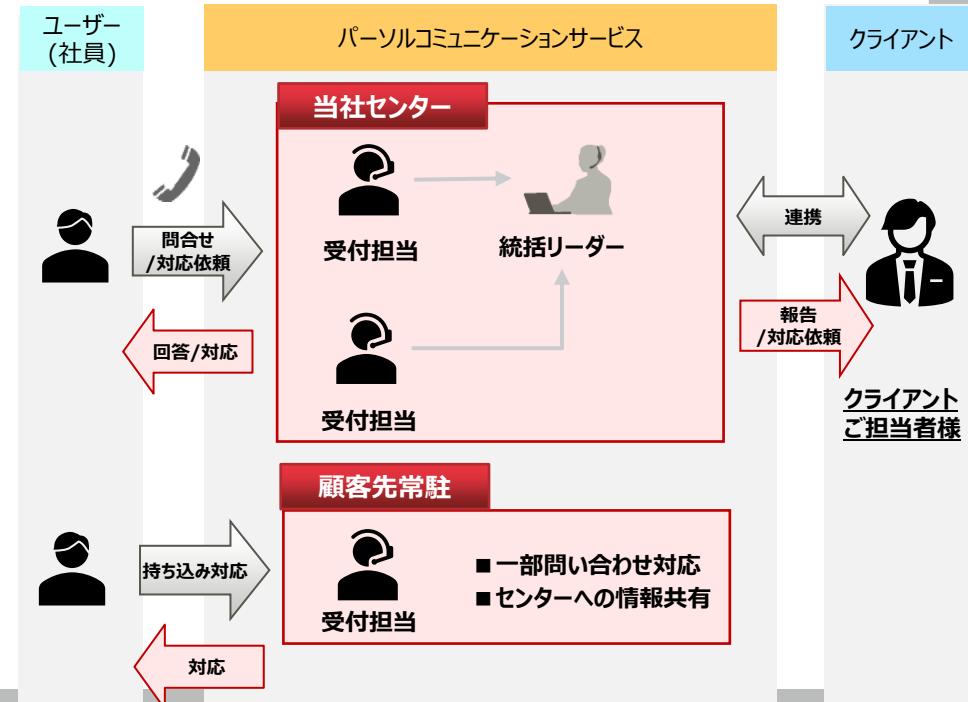
事例) 電機メーカー様_ヘルプデスク

- 当社センター・オンサイトの2拠点にてICT機器等の円滑な活用に向けた機器サポートを実施。

基本情報

業種	電機メーカー
センター	弊社センター/オンサイト
サービス時間	月曜日～金曜日 8:00～20:00 ※土日祝日休み
サポートメディア	電話・メール/持ち込み対応
月間件数	1,750件
サービス対象者	グループ含む従業員27,000名
サービス内容	<ul style="list-style-type: none">■ 業務アプリ/PC全般に対するトラブル対応■ 顧客先常駐ではセンターメンバーには貸与不可なライセンスを提供いただき2次窓口として機能および持ち込み対応を実施

サービスイメージ



私たちが選ばれる3つのポイント



実績

25年以上の経験と500社を超える
サービス提供実績

企業向けヘルプ、コールセンター運用のプロとして、これまで500社を超える企業へのサービス・ソリューション提供経験から得た、知識や情報を活かし、お客様のビジネススタイルに合わせたサービスを提供しています。



ノウハウ

企業向けヘルプデスクでのサポート業務で培った専門性と高いノウハウと人材

企業向けヘルプデスク専業会社として多様な業務をさまざまな業種業態へ提供することで培ってきた経験とノウハウ、そしてバラエティに富んだ人材が大きな強みです。



最先端サポート

パーソルグループの強みを活かした
最先端サポート

企業の課題解決や利便性向上、顧客接点の最適化を図るためのICTやAIといった最先端のテクノロジーを導入。人とICTの融合による高品質なサービスを提供しています。

ご清聴ありがとうございました。