

はじめてみよう！

# ルール駆動開発

全開発者必見！

レガシーモダナイズを成功させる開発手法



Red Hat



## 登場人物

### Y社



**佐藤**  
IT部門の部長 香澄の上司

### 香澄

IT部門に所属  
入社5年目 本作の主人公



### 朋子

営業部門に所属 香澄の同期



### 荒木

人事総務部門のスペシャリスト



### HIKARI



Y社のレガシーな人事総務システム

### レッドハット社

### 高橋

ベテランアーキテクト



## ストーリー

「HIKARI」は約20年前からあるY社の人事総務システム。ホストで動いていたが、10年前にオープン化。

その後も機能拡張やDBリプレースなど続けてきたが、グループ会社の統廃合により、システム間連携の必要性や、保守費の高騰等の理由で、5年前に全面刷新の計画が立てられた。

IT部門の精鋭と既存ベンダーとでチームが作られ、複雑化した「HIKARI」の全体を分析するところから始まったが、1年たっても現行解析に終わりが見えず、新システムへの要求も膨れ上がり、刷新の見積もりすら出来ずにいた。現行システムの改修もストップしたままで、何一つ成果を出せず、2年目にリプレース計画の中止が決定された。ストレスfulな環境下で体調を崩す者や退社する精鋭も多かった。

既存ベンダーにすでに支払った額は大きかったが、この失敗を受けて、無償でサポート延長をする契約が結ばれ、

レガシーなシステムはその後Y社の人事総務を担い続けることに。

しかし、このたび「HIKARI」の再びの刷新プロジェクトとして、「NOZOMI」プロジェクトが開始されることになった…！





うん、「HIKARI」はいろいろ問題が多くてね。

それにしても、浮かない顔してるね。

うん、ありがとう…

「NOZOMI\*」のリーダーアーキテクトを務めることになったんだってね。すごい抜擢だね、おめでとう！

香澄

\*HIKARIのリプレースプロジェクト



改修が追いついていなくて、現場が手作業でやってることも多いよ。私の給料も正しく計算されているか、怪しいくらいだわ。

それに、最近は働き方改革とかで、規則や規約がたびたび変わるでしょ、

はあ…

何しろ古くからあるシステムだから、中身がぐちゃぐちゃで、全体を把握している人がいないの。

朋子



それにね、前回の失敗は、既存ベンダーに丸投げしたことにあるというのが上層部の結論みたい。

今度はさすがに失敗出来ないでしょ。

でも、予算は前回よりもずっと少なくなるみたいなの。

それに「HIKARI」のリプレースが失敗したこと、知ってるでしょ？

ああ、あの黒歴史ね…

カッ





うん。  
それで悩んでるところ。

既存ベンダーに  
対する社長の  
不信は  
わかるけどね。

だから  
既存ベンダーは  
外される方向なの。

私達でやらなきゃ  
ならないのよ。

でも既存ベンダーを  
入れないとなると、  
これはやり方  
大きく変えないと  
難しそうだね。  
現行システムの中を  
知らないわけだし。



そう。  
たしか高橋さん、  
「ルール駆動開発」  
とか言ってたわ。

レッドハット？  
確か少し前に  
営業部門の新システム  
入れるときに  
協力してくれた  
会社だよな。  
私はよく  
知らないんだけど。

そっか。  
高橋さんって  
知ってる？  
レッドハットの。

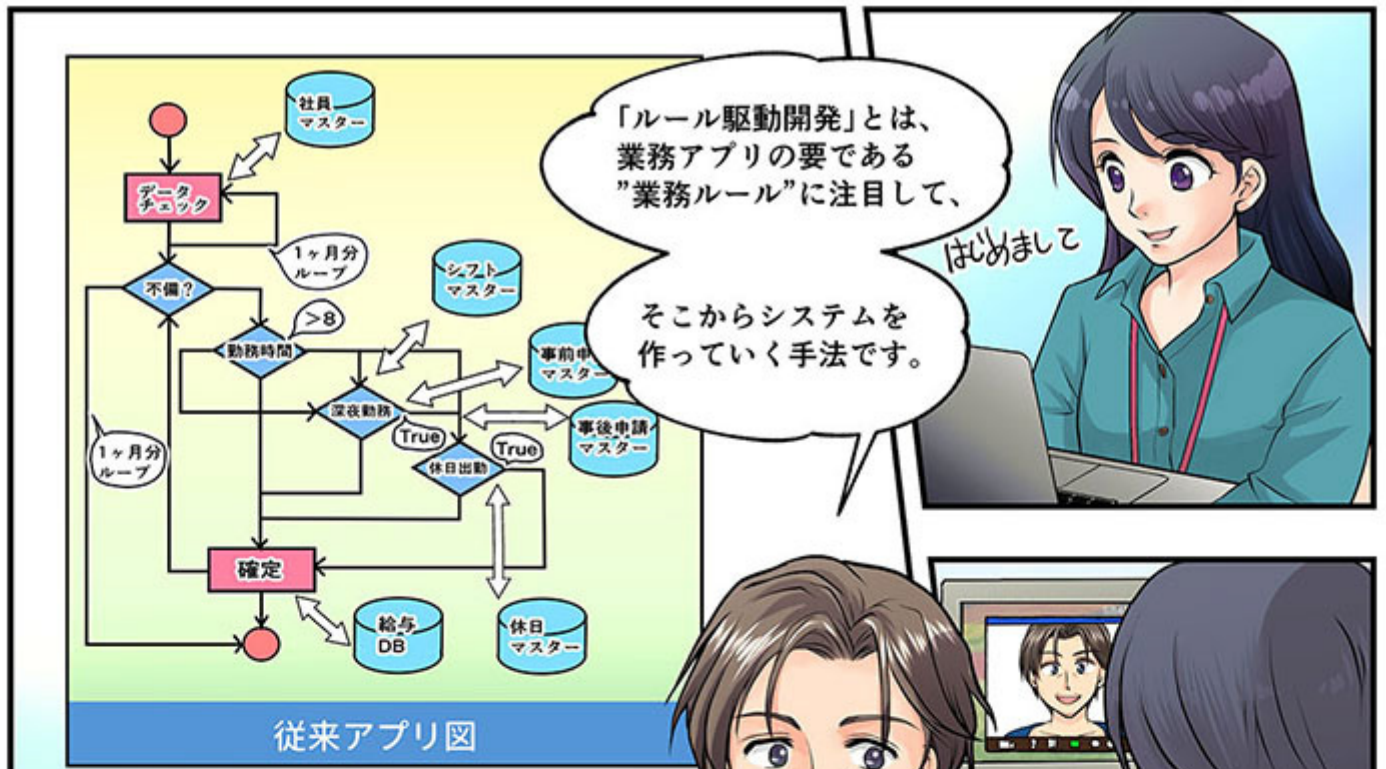


ルール駆動開発…？

「これはレガシーシステムを  
リプレースする時にも  
使える開発手法だ」って。

今度 香澄に  
高橋さんを  
紹介してあげるよ。

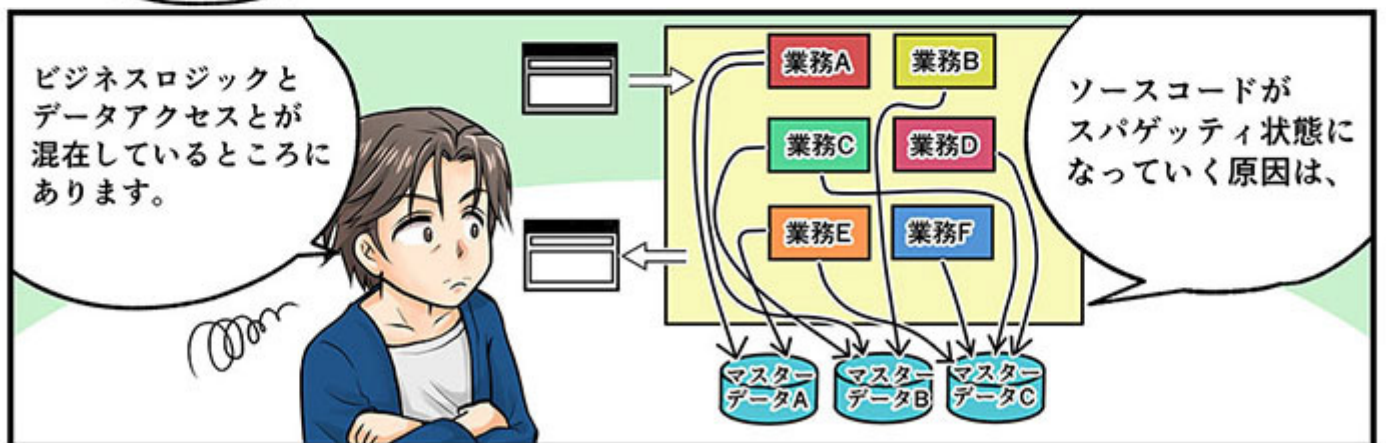




業務ルールはアプリの中で条件文として実装していますが、  
変更の繰り返しの中で、  
このように何度もデータアクセスを繰り返す入り組んだ構造になってしまいます。

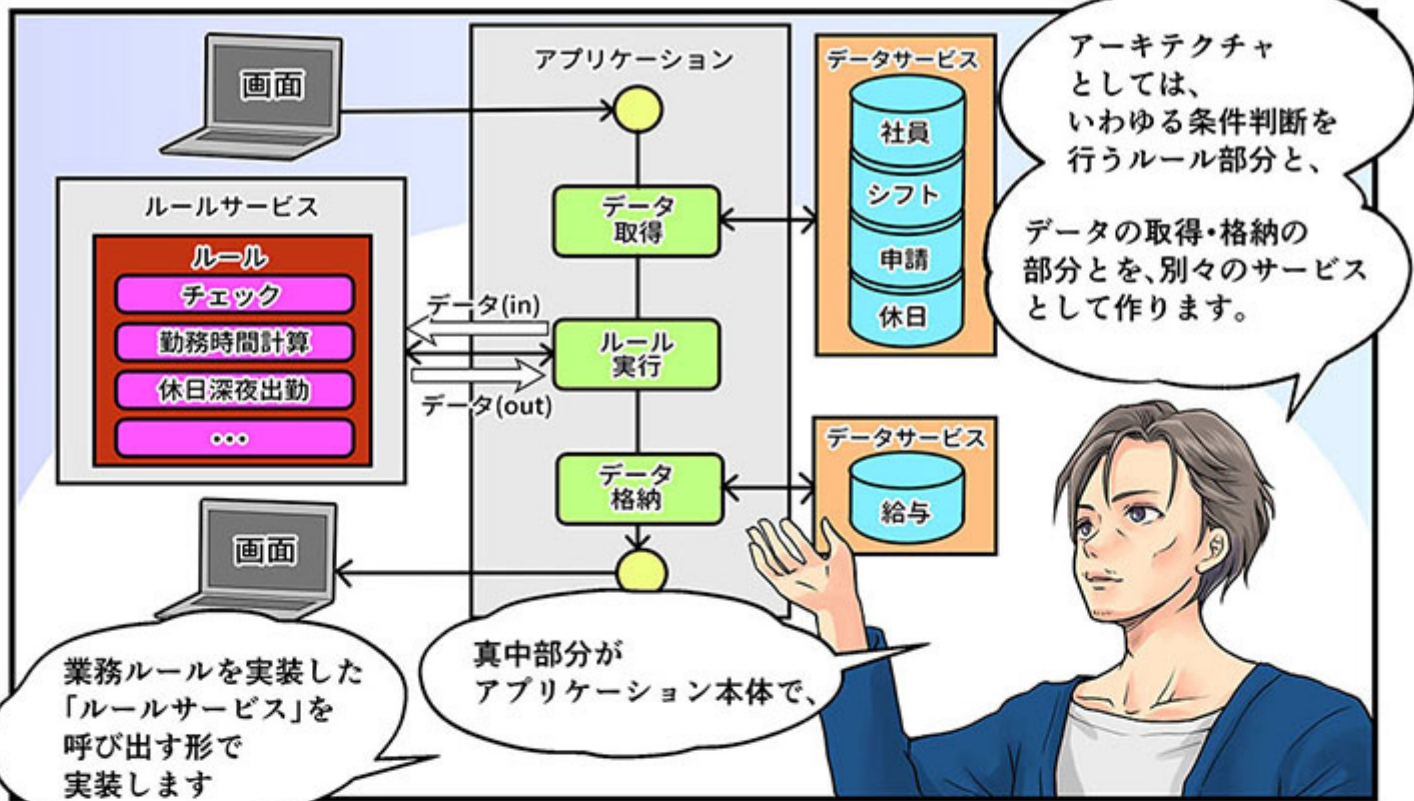
高橋

重要なのは、アーキテクチャ。

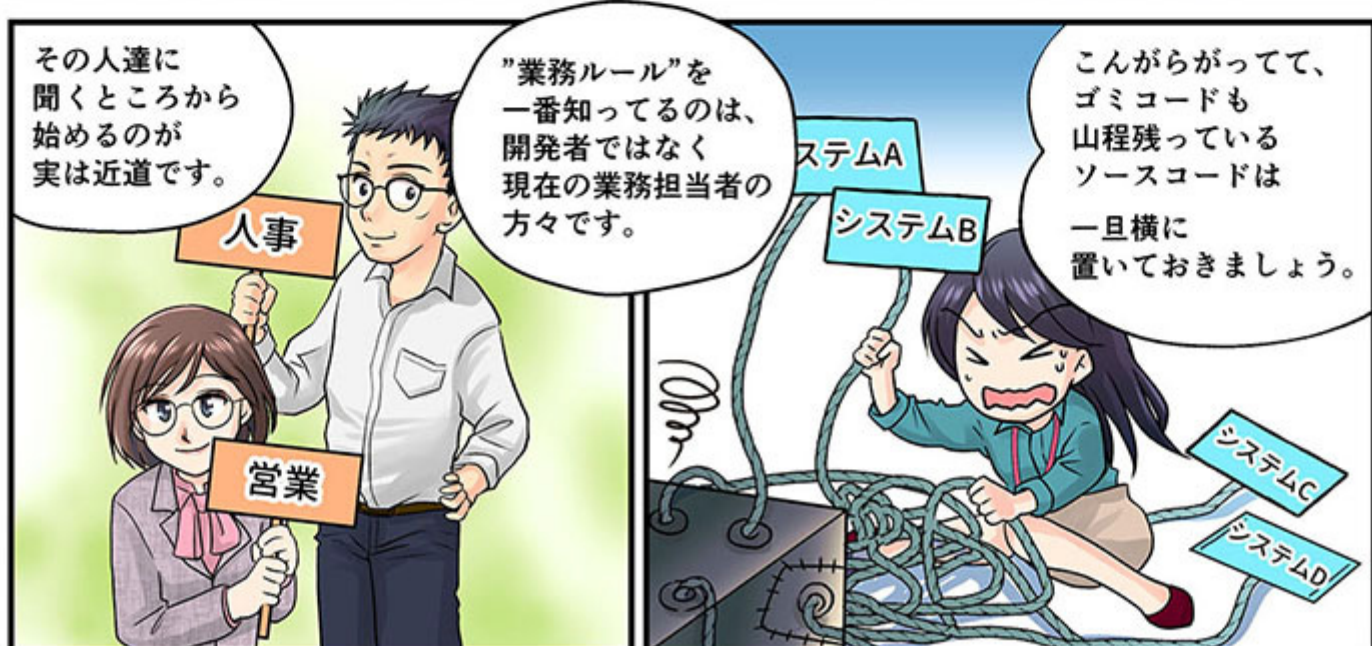
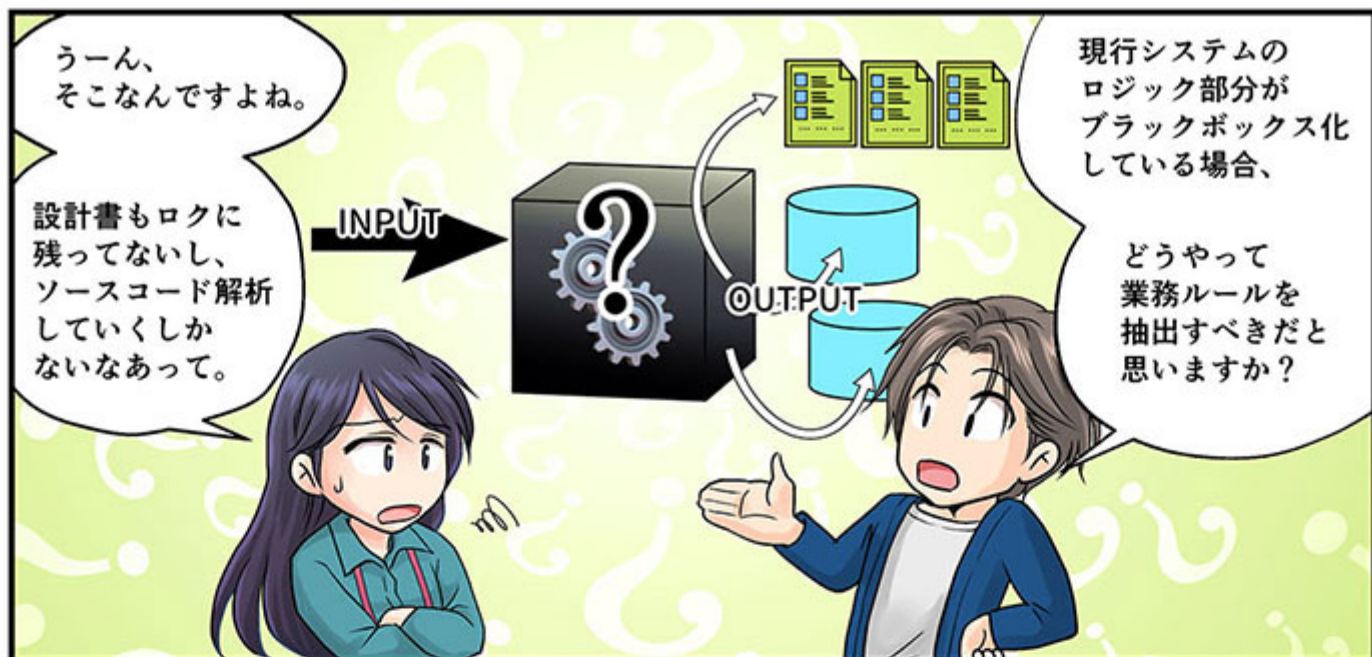


まさに、HIKARIはこれだわ。  
解読不能で  
ごちゃごちゃ…











似たような業務でも、担当が違う場合はルールが違うことがあります。

違ってよいルールであれば別ものとして実装すべきですし、

発注

仕入原価

販管費

本当は同じであるべきならば、統一すべきです。

人事情報DB

聞き出す際に重要なことは、

業務単位に、そして業務用語を使って整理していくということです。

申請

賞与

手当

退職金

異動履歴

社会保険

口座変更届

年末調整

考課

教育管理

休暇届

でも、それだと一から要件定義し直すのと同じくらい時間がかかるのではないですか？

ルール駆動開発のメリットとして、

本来あるべき業務ルールとは、というところに立ち返って整理するところから始められるという点があります。

購買フロー

売上フロー

そのためには、ルールエンジンを使って、

イテレーション開発で進めていくのが重要です。

ゴミ溜めのソースコードから時間をかけて必要なルールを発掘するより、

ずっと短い期間で効率よく終わりますよ。

ルールエンジンを使った、イテレーション開発…





# Red Hat Decision Manager

条件  
(○○ならば)

アクション  
(▲▲する)

RuleTable キャンペーンポイント率ルール					
キャンペーン名	CONDITION				ACTION
	対象日		購入金額		ポイント
文化してる?	11/3	11/23		¥4,999	0%
			¥5,000	¥9,999	10%
			¥10,000	¥39,999	20%
Black Friday	11/24	11/30		¥3,999	0%
			¥4,000	¥20,000	20%

ルールエンジンは、  
○○ならば▲▲する、  
といったルールを  
直感的に実装して、  
しかもそれを  
効率よく動作させる  
仕組みです。

業務ルールを  
こういった形式で  
整理することで、  
認識齟齬がないかどうか  
常に確認しながら  
進められるし、



設計・実装

要件整理

テスト・解析



実装・テストを繰り返すことで  
品質や性能を確認しながら  
進められます。

これが、  
イテレーション開発です。

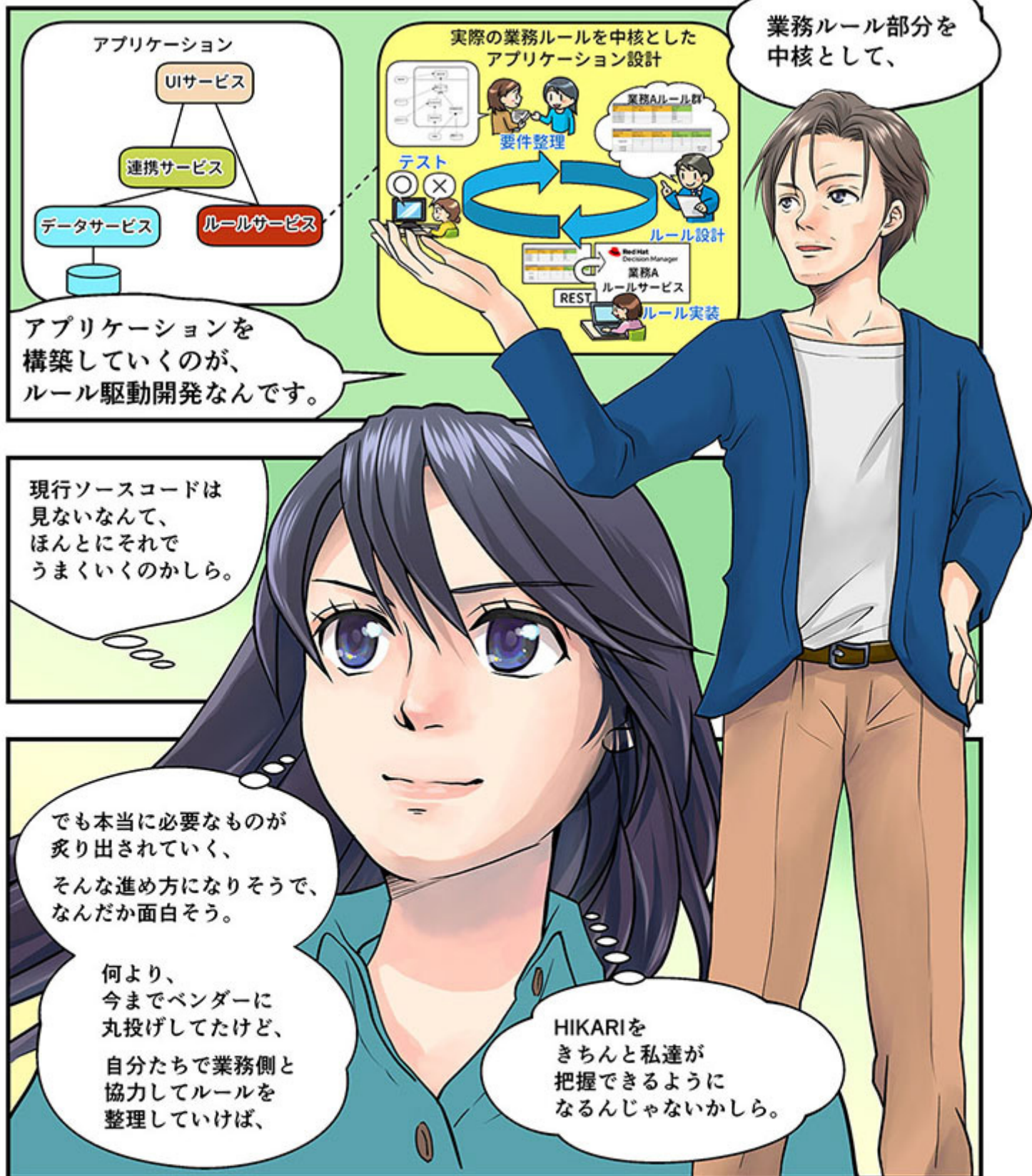
このように…

安全かつ  
スピーディーだわ

要件整理→実装→テスト  
を繰り返しながら  
進めていく？







業務ルール部分を  
中核として、

アプリケーションを  
構築していくのが、  
ルール駆動開発なんです。

現行ソースコードは  
見ないなんて、  
ほんとにそれで  
うまくいくのかしら。

でも本当に必要なものが  
炙り出されていく、  
そんな進め方になりそうで、  
なんだか面白そう。

何より、  
今までベンダーに  
丸投げしてたけど、  
自分たちで業務側と  
協力してルールを  
整理していけば、

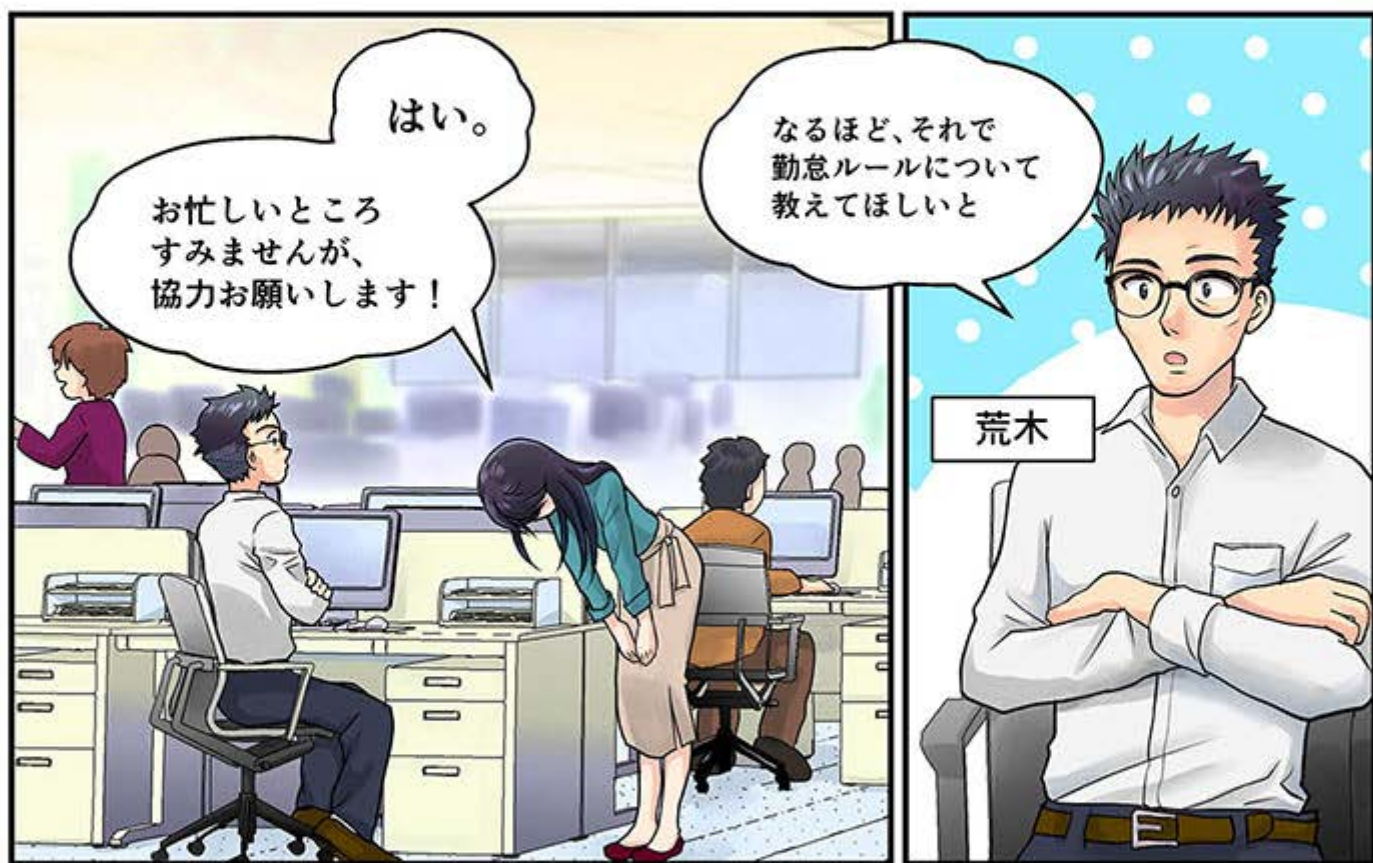
HIKARIを  
きちんと私達が  
把握できるように  
なるんじゃないかしら。



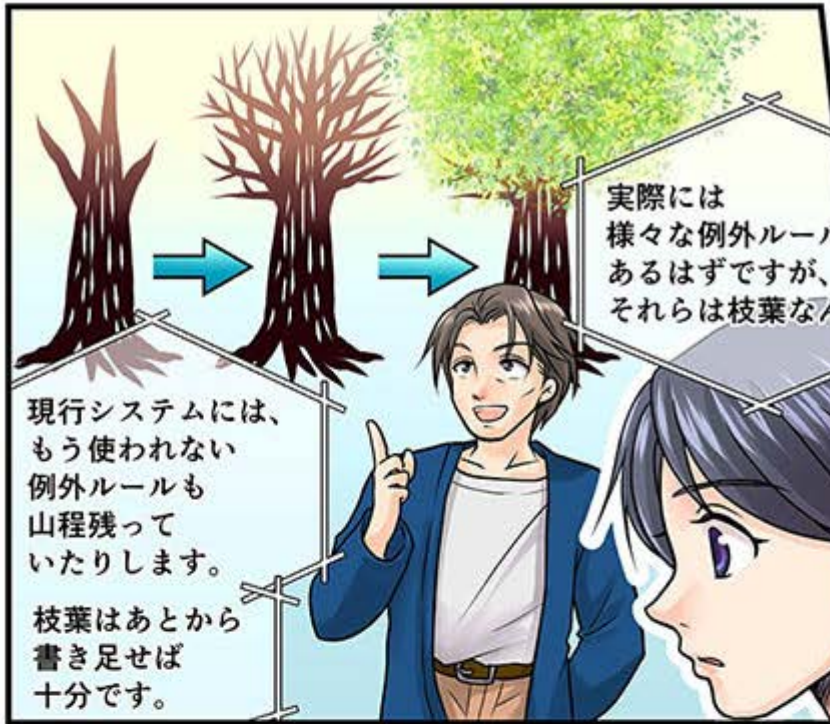
ルール駆動開発、  
やってみたいです、  
もっと詳しく教えて下さい！

高橋さん、  
NOZOMIプロジェクトに  
ぜひテクニカルアドバイザー  
として入って  
いただけませんか？









現行システムには、  
 もう使われない  
 例外ルールも  
 山程残って  
 いたりします。  
 枝葉はあとから  
 書き足せば  
 十分です。

実際には  
 様々な例外ルールも  
 あるはずですが、  
 それらは枝葉なんです。

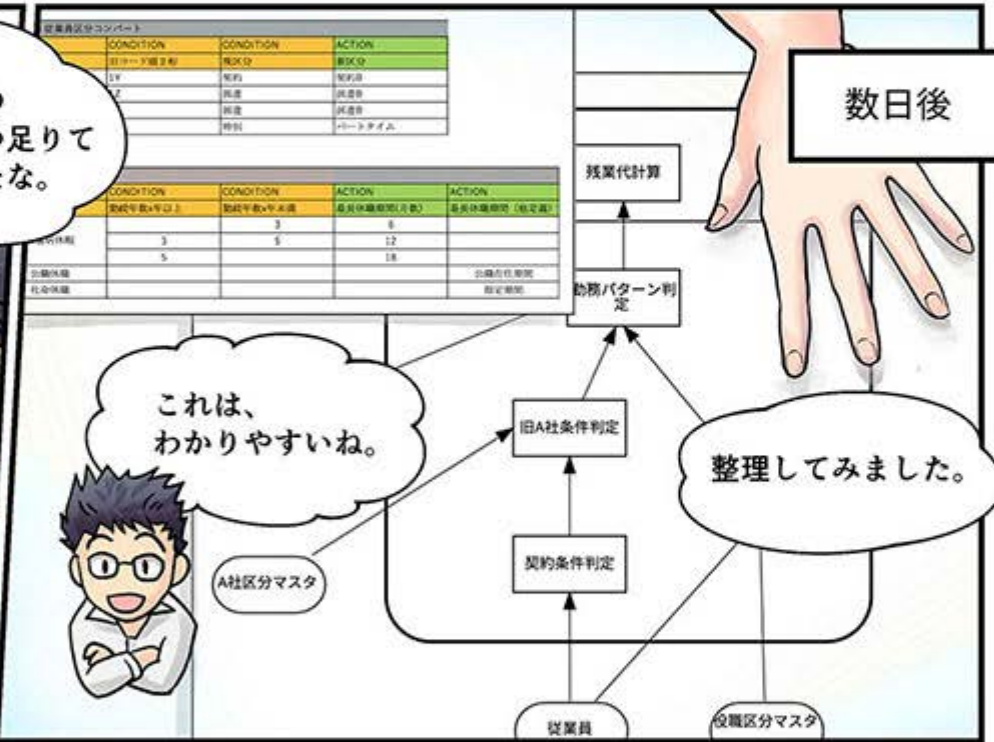
ルールを  
 抽出する時は、  
 まず幹となる  
 主要なルール、

誰でも知ってる  
 ルールから  
 書き出して  
 いくんです。



あれ、  
 時短勤務の  
 種類が一つ足りて  
 いなかったな。

ここですか？  
 では追加しましょう。



数日後

これは、  
わかりやすいね。

A社区分マスタ

整理してみました。

従業員

役職区分マスタ

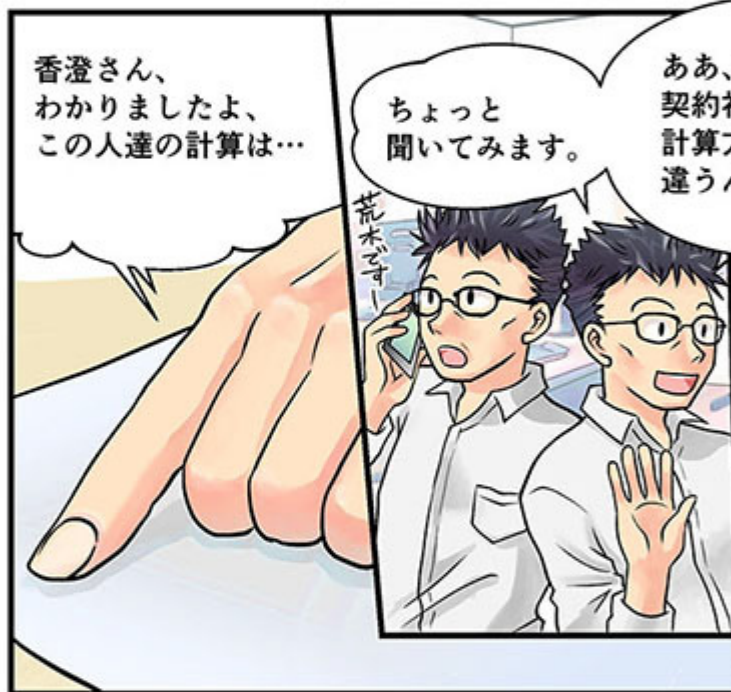


いわゆる  
 システム設計書として  
 書き起こすのではなく、  
 業務側に合わせた  
 記述で進めるから、ね。

これなら積極的に  
 協力してもらえそう。

要件定義の  
 認識合わせが  
 とても  
 やりやすいわ。









はい、  
ありがとうございます。  
総務の荒木さんに  
協力いただいた  
おかげです。

複雑で大規模に  
見えていましたが、  
整理して見ると  
実際に必要なところは  
半分ほど  
だったみたいです。

しかも、  
今までの計算式の  
間違いを  
発見出来ました。

テスト導入とはいえ、  
勤怠業務部分を  
三ヶ月で  
作り変えたってね。

会議室

総務部長から、  
「うちのIT部門だけで  
ここまでできるとは  
思わなかったよ、  
すごいね」  
とのコメントを  
もらってるよ。

佐藤



ルールが整理され、  
可視化されたので、  
誰が見ても  
わかりやす  
くなりました。

今後もこの形式で  
きちんと統制をかけて  
メンテナンスをしていけば、

中身がブラックボックス化  
することはないと思います。

「わかりやすく  
正確だから、  
現行システムでは  
出来ていない  
ルールの修正も、  
新しい方に入れて  
結果を見ている、」

それに  
現場からは、

というコメントを  
もらってるよ。

品質も  
良いものが  
できたようだね。



それでこそ、  
IT部門の  
存在意義がある  
というものだね。

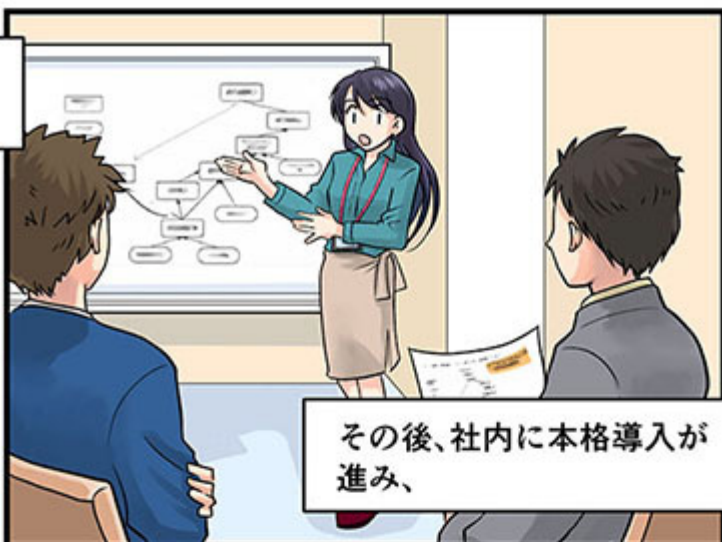
いいね。  
業務部門の  
要望を整理して、  
管理する、

内製化への  
舵を切ろう。





プロジェクトは社長賞を受賞し、  
私は昇進をすることとなった



その後、社内に本格導入が  
進み、



あのシステムにも  
このシステムにも  
適用できそうだな。

まだまだ  
忙しい日々が続くそう。



ルール駆動開発で  
シンプルかつ  
柔軟度の高い  
アーキテクチャに  
なるし、

業務ルールを  
整理して見直すから、  
業務効率と  
サービス品質の  
向上が大いに  
期待できそうね。



これは当分  
無理そうだな。

IT部門の増員を  
お願いしないと。



落ち着いたら、  
今度こそ  
美味しいもの  
食べに行こうね！



# モダナイゼーションで利用されるレッドハットの主要な製品群

モダナイズの成功に  
必要なルールエンジン

 **Red Hat**  
Decision Manager\*

 **Red Hat**  
Consulting

ルール駆動開発の  
支援をいたします!

 **Red Hat**  
Process Automation  
Manager\*

業務のタスク管理や  
進捗管理にどうぞ

マイクロサービス化の  
必須アイテム

 **Red Hat**  
AMQ

システム間連携と  
サービス連携の決定版

 **Red Hat**  
Fuse

新しいAPIビジネスは  
いかがですか?

 **Red Hat**  
3scale API  
Management

 **Red Hat**  
OpenShift  
Container Platform

コンテナ環境といったら  
コレでしょ!

 **Red Hat**  
Ansible Automation  
Platform

運用もよりインテリジェンスに  
モダナイズしましょ

 **Red Hat**  
Data Grid

超高速処理を可能にする  
In Memory DB

全ての挿絵は  @koyagi\_rm さん! Special Thanks!!

詳しい説明をご希望の方はレッドハット セールズオペレーションセンター  
[sales-jp@redhat.com](mailto:sales-jp@redhat.com) までお問い合わせください。

\* Process Automation Manager 及び Decision Manager 製品は Red Hat から IBM へ移管されました。Red Hat PAM/DM V7.13 が 2022年8月4日にリリースされ、Red Hat からの最終バージョンとなります。今後の製品バージョンは V8.0 から IBM よりリリースされます。IBM PAM/DM 8.0 は 2022年7月22日にリリースされました。  
<https://access.redhat.com/articles/6968468>  
<https://community.ibm.com/community/user/automation/viewdocument/faq-ibm-process-automation-manage>

© レッドハット株式会社