HITACHI

レガシー脱却のリアル：簡易言語からCOBOLへの変換とデータ調査の舞台裏

Prepared by
株式会社 日立製作所 AI&ソフトウェアサービスビジネスユニット
アプリケーションサービス事業部 アプリケーション・モダナイゼーション本部
伊丹 恵美

Date
2025年12月18日

Agenda

1. はじめに
2. マイグレーション事例概要
3. 簡易言語からCOBOLへの変換
4. 実データからのレイアウト調査
5. おわりに

はじめに

1

HITACHI

本日の講演

本セッションでは、メインフレームからWindows環境への移行プロジェクトで得られた実践的な知見をご紹介します。

- **独自の簡易言語をCOBOLへ変換する際に工夫したポイント**
- **ファイルレイアウト仕様書がない中でのデータ移行における、データ構造の調査方法と品質確保の取り組み**

レガシーシステムの移行に関わる方に向けて、現場での課題とその解決策をコンパクトにお伝えします。

マイグレーション事例概要

2

HITACHI

2.1　プロジェクト方針

■プロジェクトの目的

本システムは、日立メインフレーム（VOS3）で稼働しており、利用期限が2030年に迫っている。システム有識者の減少も間近に迫っており、システムの継続性を維持するため、オープンシステム化を実施する。

■プロジェクト方針

- ✓ 業務アプリケーションにおけるロジック変更は行わない（ストレートコンバージョンを実施）
- ✓ 現行プログラムは、COBOL言語へ統一する（資産移行については原則移行ツールを適用する）
  ⇒テーマ①：独自の簡易言語をCOBOLへ変換する際に工夫したポイント
- ✓ 現行データは、同様の結果が次期システムにて得られるようにデータ変換を実施する
  ⇒テーマ②：データ構造の調査方法と品質確保の取り組み
- ✓ 新システムのオープン環境を踏まえ保守性を考慮した運用設計とする

HITACHI

2.2　システム構成

■システムの特徴

・オンライン処理はなく、バッチ処理のみ

・データ管理はファイルのみ

■移行対象資産

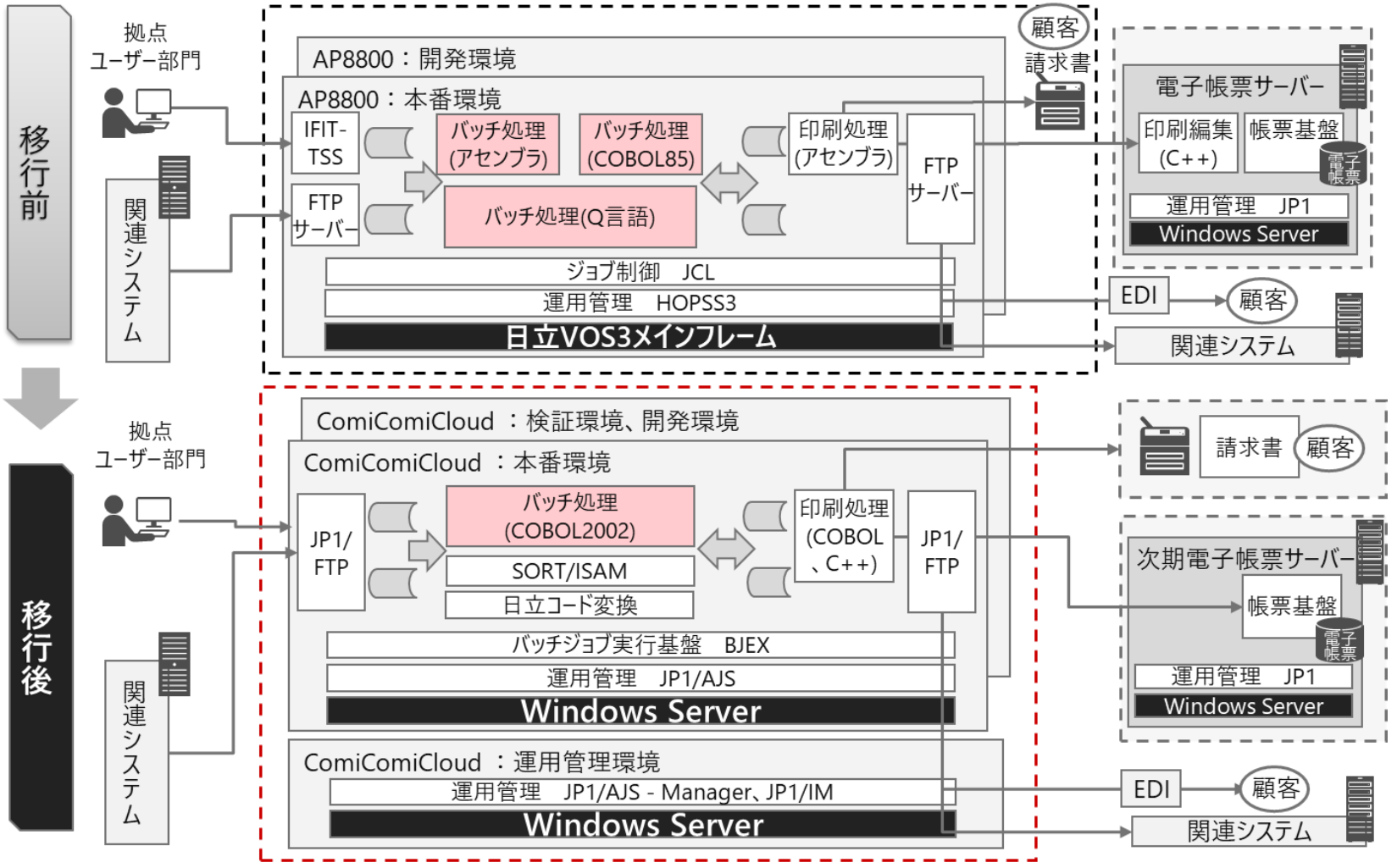
COBOL　　：320本

Q言語　　：4,300本

アセンブラ　：300本

JCL/カタプロ：4,000本

対象資産規模：約1.5Mstep



HITACHI

2.3　移行パス

No.	機能	現行システム	新システム
1	OS	VOS3/FS、LS	Windows Server
2	アプリケーション	COBOL85：　320本（COPY句含む） Q言語　：4,300本 アセンブラ：　300本 ✓COBOL技術者は問題なく確保できている ✓特にQ言語は技術者の調達が困難	**COBOL2002**　：7,920本 言語の統一 ✓新システムはCOBOLだけでアプリ保守可能 ✓言語それぞれのコーディング基準不要 ✓新言語スキルが不要のため生産性確保
3	バッチ処理	JCL（ジョブ制御言語）：4,000本 ※JCL中に記載されたQ言語：約3,000本	uCosminexus BJEX
4	帳票（電子、紙）	・現行外部ライター機能→電子帳票サーバー ・LCDS形式→日立システムズのプリンティングサービス	・次期外部ライター機能→次期電子帳票サーバー ・STDOUT形式→他社のプリンティングサービス
5	ファイル転送	XNF、TIOP3	JP1/FTP
6	ジョブ管理	HOPSS3、AOMPLUS、JSS3	JP1/AJS3
7	OSファイル	VSAM、SAM	ISAM、SAM
8	文字コード	EBCDIK/KEIS	JIS8/Shift-JIS

HITACHI

2.4　スケジュール

① 資産棚卸

●移行対象資産の調査
●資産・データ関連の把握

② 移行性分析

●移行方式設計項目の抽出
●変更観点と修正規模の把握

③移行方式設計

●移行方式項目別の変換方法の確定
●パイロットテストによる移行方式設計の評価

④ 資産移行

●ツール開発（カスタマイズ）やパラメーターの作成
●PGM資産のツール変換および手修正
●先行テストによる大量の比較検証に耐えられる品質の評価

⑤データ移行設計・データ移行

●ファイルレイアウト調査
●データ変換ツール設計・開発・テスト
●テスト／本番切替向けデータ移行作業

⑥ テスト

●現新比較テストの実施
●システムテストの実施
●業務運用テストの実施

簡易言語からCOBOLへの変換

3

HITACHI

3.1　独自簡易言語（Q言語）とは

本章では、独自簡易言語Q言語についてと、その特徴を踏まえ、膨大なQ言語プログラムを効率的かつ正確にCOBOLへ変換した取り組みについてご紹介します。

Q言語の概要

- 事務計算処理における簡易言語として、日立グループにて開発
- 日立メインフレームVOS3にて動作する
- 本プロジェクトでは、プログラム数は約7,800本にのぼる

COBOL変換のポイント1

＜Q言語の特徴1＞
- Q言語は「セクション」単位で基本処理を管理
- ユーザーは各セクションをコーディングするだけでよい
- 全体の流れ制御はQシステム(*1)が自動で行う

▶

＜COBOL変換への課題＞
Qシステムの制御処理を実装する必要があり、COBOL変換時に記述がばらつく可能性がある。

▶

テンプレートを用いてコード変換を効率化し、記述の統一を徹底

COBOL変換のポイント2

＜Q言語の特徴2＞
- ファイルのレコード構造は定義しない
- 実行時に参照したいデータ位置を指定可能（部分参照可能）

▶

＜COBOL変換への課題＞
COBOLでは読み書きするファイルのデータ項目定義は必須。無いものをどう生み出すのか。

▶

Q言語にはないCOBOLのデータ項目定義の作り方を工夫

※１：Qシステムとは、Q言語プログラムをコンパイルおよび実行するためのシステムのこと

HITACHI

3.1　Q言語からCOBOL変換の課題と対策（その1）

課題① Qシステム制御処理	• Qシステムにより実現される処理（プログラムには表れない処理）を実装する必要あり • 上記処理をCOBOL変換するにあたり、対象本数が多く、記述がばらつく可能性あり

各セクションごとに記載すべき内容は決まっており、ユーザーはセクションごとにコーディングを実施。
基本ファイルのオープン／クローズ、ファイル読み込み、全体的な流れの制御はQシステムが行っており、**COBOL変換時に実装が必要**となる。

＜Q言語プログラム＞

```
QTE7120  START
SYS100   INP   R50
SYS110   TRA   R50
SYS005   PRI   R160,F58
         MATF  ( M5Z3 = S5Z3 ) 1 : 1
*
RECORD   SECT
         IF    (M1Z3#'Q52') THEN
          SET   M1Z3        ,L012
   ・・・
RECORDS  SECT
         IF    (S1Z3#'ABC') THEN
   ・・・
         ENDIF
MONLY    SECT
   ・・・
EQUAL    SECT
   ・・・
PAGE     SECT
         SET   'ﾅｲﾖｳ       ',L001
   ・・・
END
```

定義部

処理部

＜プログラムの記述内容＞

・ファイル定義（マスタ、トランザクション、出力ファイルなど）
・作業エリア定義
・実行制御定義（マッチング、ソートキー定義）

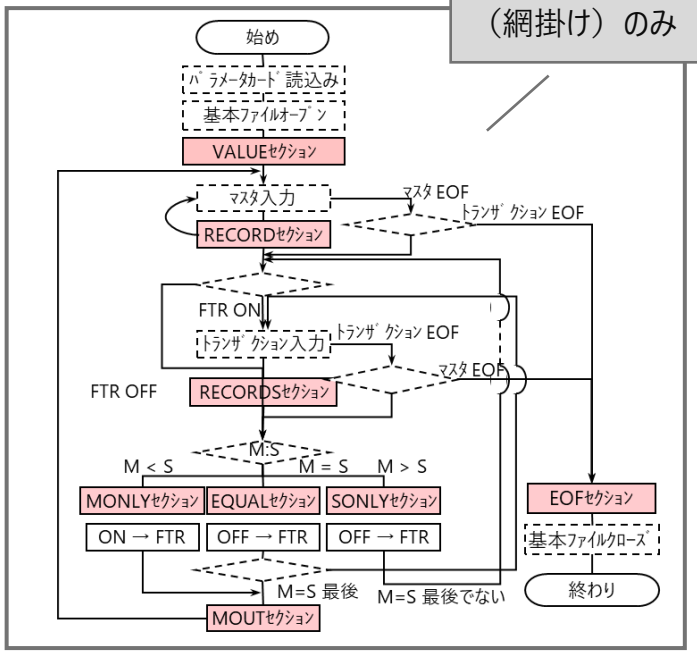
RECORD／RECORDSセクション
マスタレコード／トランザクションレコード入力後、各レコードに対して最初に実行される処理。

MONLYセクション／EQUALセクション
マスタとトランザクションのマッチング結果に該当するレコードに対する処理を記載する。

PAGEセクション
プリンターファイル出力時の、各ページのヘッダー処理に使用する。

＜セクションの実行順序例＞

コーディングは各セクション（網掛け）のみ



HITACHI

3.1　Q言語からCOBOL変換の課題と対策（その1）

対策

- 使用される命令の組み合わせにより、処理の実行順序・制御内容がパターン化されていることに着目
- 13種類のテンプレートを作成し、Qシステムにより実現されていた処理を実現

7つの命令句の組み合わせにより、13パターンのCOBOL構造体テンプレートを作成。
テンプレートにて、Qシステムにより実現されていた処理（ファイルオープン・クローズ、ファイル読み込み、実行順制御など、コードに明記されない処理）を実現。

＜命令の組み合わせと構造体テンプレート＞　　※○は、命令句の使用ありを表す

命令句	処理内容	構造体テンプレートの種類					
		①MAIN構造体	②INP構造体	・・・	⑧MATF構造体	・・・	⑬MATF+BREAK+SORS構造体
MAIN	下記命令句含まない	○	○	○	○	○	○
INP	マスタファイル処理		○	・・・	○	・・・	○
TRA	トランファイル処理			・・・	○	・・・	○
SORM	マスタファイルのソート処理			・・・		・・・	○
SORS	トランファイルのソート処理			・・・		・・・	○
BRE	ブレーク処理			・・・		・・・	○
MATF	マスタとトランのマッチング処理			・・・	○	・・・	○

INP、TRA、MATF命令を使用している場合、**MATF構造体**に分類

＜テンプレート構造例＞

Qシステムにより実現されていた処理を実現（赤字）

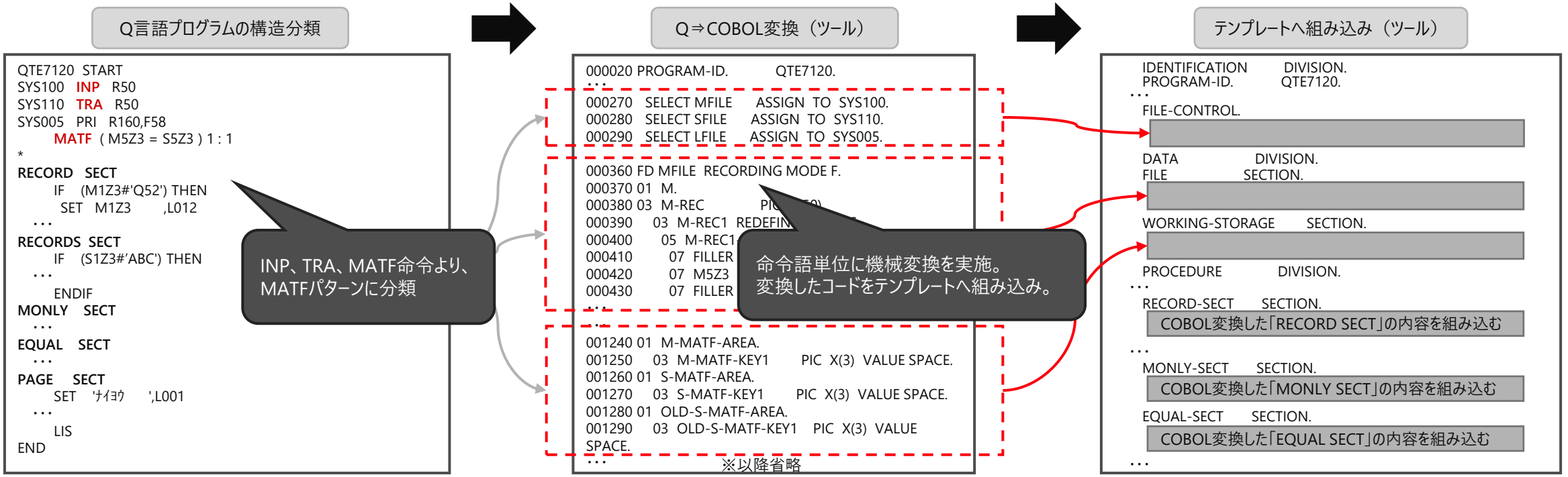
MATF構造体テンプレート	処理内容
メイン処理	**初期処理→業務処理→終了処理　実行**
初期処理	**・ファイルオープン** **・VALUE-SECT の実行**
その他初期処理	VALUE SECT
業務処理	**・RECORD SECT の実行** **・RECORDS SECTの実行** **・MATF SECT を条件付きで繰り返し実行**
マスタファイルレコード編集処理	RECORD　SECT
トランファイルコード編集処理	RECORDS　SECT
マッチング判断処理	**マッチング結果より、呼び出しセクションを制御する（EQUAL処理／MONLY処理／SONLY処理）**
EQUAL処理	EQUAL SECT
MONLY処理	MONLY SECT
SONLY処理	SONLY SECT
終了処理	**・ファイルクローズ** **・必要に応じて外部CALL（例：CBLABN）**

HITACHI

3.1　Q言語からCOBOL変換の課題と対策（その1）

結果	• テンプレートを適用し、ツールでのコード変換を行うことで、効率的な開発を実現 • コードの書き方や構造が統一され、可読性と保守性の向上を実現

Q言語のプログラムを、使用している命令の組み合わせによって、プログラム構造を分類し、テンプレートを決定。
ルールベースのQ⇒COBOLの機械変換を行いながら、テンプレートへの組み込みをツールにて実現。



HITACHI

3.2　Q言語からCOBOL変換の課題と対策（その2）

課題②
データ項目定義の違い

- Q言語とCOBOLでは、プログラムにおいてファイル利用時のデータ項目定義に違いがある
- COBOLへ変換した場合、コンパイルエラーおよび、実行時エラーが発生する可能性あり

例）担当者マスタファイル（1レコード80バイト）を利用する場合の差異

担当者番号	担当者名	所属部署コード	FILLER
9（5）	X（20）	9（3）	X（52）

Q言語の場合

```
QTE7120  START
SYS100   INP   R80          ※レコード長80バイトのマスタファイルを定義
SYS110   TRA   R80          ※レコード長80バイトのトランファイルを定義
・・・
                            ※ファイルのレコード構造定義はなし

MATF  ( M1Z5 = S1Z5 ) 1 : N   ※マスタとトランファイルのマッチング処理
・・・
```

参照したいデータ位置を指定し、プログラム実行可能

INP／TRAなどの各命令で、ファイルの属性（レコード長、ブロック長、固定長／可変長など）を定義する。
ファイルのレコード構造は定義せず、プログラム実行時に参照したいデータ位置を指定（部分参照）することが可能。

COBOLの場合

```
DATA DIVISION.
FILE SECTION.
FD 担当者マスタ-ファイル
    RECORD CONTAINS 80 CHARACTERS
    LABEL RECORDS ARE STANDARD.
01 担当者マスタ-レコード.
   05 担当者番号       PIC 9(5).
   05 担当者名         PIC X(20).
   05 所属部署コード   PIC 9(3).
   05 FILLER           PIC X(52).
```

DATA DIVISIONにて、
各ファイルのレコード構造（型、桁数）を定義する必要あり。
レコード構造のデータ項目定義がない場合、コンパイルエラーおよび実行時エラーが発生してしまう。

HITACHI

3.2　Q言語からCOBOL変換の課題と対策（その2）

対策

- ファイルのレコード構造全てを定義することは不可能と判断
- プログラムから、データとして使用している箇所に限定して抽出し、データ項目定義を実施

現行ドキュメントが整備されておらず、ファイルレイアウト定義がない。
（右図のようなカラム単位のレイアウトはなし）

担当者番号	担当者名	[illegible]	FILLER
S9（5）	X（20）	[illegible]	X（52）

ファイルレイアウト定義からのアプローチは不可

Q言語プログラムには、レコード構造定義はなし。
ただし、プログラム実行時に参照される箇所のデータ項目の **位置、属性、長さ** は抽出可能。

```
QTE7120  START
SYS100   INP   R80
SYS110   TRA   R80
・・・
MATF  ( M1Z5 = S1Z5 ) 1 : N
・・・
```

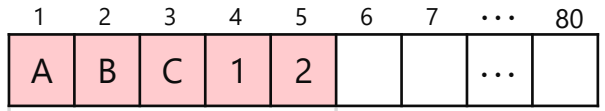
ファイル識別名
M：マスタファイル

属性と長さ
Z5：半角文字5バイト

M 1 Z 5

レコードのデータ位置
1：先頭1バイト目

例）レコード参照箇所イメージ
マスタファイル(M) の 1バイト目から半角文字で5バイト



データ使用箇所の記述よりデータ項目を抽出するアプローチを採用

HITACHI

3.2　Q言語からCOBOL変換の課題と対策（その2）

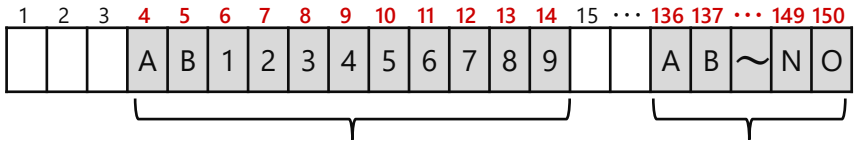
結果	・機械的に、プログラム実行時に参照するデータ項目の位置、属性、長さを抽出し、COBOLのデータ項目を定義 ・プログラムとして実行可能（コンパイルエラー／実行時エラーなし）なCOBOLに変換することができた

各命令のオペランドに定義されたデータの部分参照項目の抽出と、COBOLのデータ項目定義への変換を**ツールにて実現**。
Q言語にはなかった、ファイルのデータ項目定義をCOBOLに実装し、実行可能なプログラムとすることができた。

```
TST760   START
SYS100   INP   R150 , B27900 , N10
SYS110   TRA   R150 , B27900 , N10
・・・
MATF  ( M004Z011 = S004Z011 ) 1:1
・・・
SONLY    SECT
         MOV  ALL ( S , CO )
         MOV  M136Z015 , CO136Z015
         WRT  FCO
EOF      SECT
・・・
```

各命令のオペランドより、
プログラム実行時に参照する
データの位置と属性、長さを抽出。

マスタレコード（1レコード150バイト）



ツールにて抽出＆変換

レコード全体のデータ定義に、抽出した部分
参照データごとに再定義する。
参照箇所以外は、未使用項目（FILLER）
として定義し、固定長レコード形式を担保。

```
・・・
FD MFILE  RECORDING MODE F.
 01  M.
  03  M-REC              PIC  X(150).
  03 M-REC1  REDEFINES  M-REC.
   05 M-REC1-1.
    07  FILLER          PIC  X(3).
    07  M004Z011        PIC  X(011).
    07  FILLER          PIC  X(136).
  03 M-REC2  REDEFINES  M-REC.
   05 M-REC2-1.
    07  FILLER          PIC  X(135).
    07  M136Z015        PIC  X(015).
・・・
```

実データからのレイアウト調査

4

HITACHI

4.1　データ移行（文字コード変換）の課題に対する取り組み

背景

ホストとオープン環境では、システム内で扱う文字コードが EBCDIK/KEIS と JIS8/SJIS と異なる。
ホストから転送されたファイルをオープン環境で利用できる形式にデータ変換する必要あり。

データ移行（文字コード変換）の課題

正しく文字コード変換するためには、格納されているデータの属性と位置を特定する必要があるが、
現行保守ドキュメント（ファイルレイアウト定義）が存在しないため、文字コード変換（EBC/KEIS ⇒ JIS8/SJIS）が困難。

対応策

実データからデータ構造の調査（属性と位置の特定）を実施

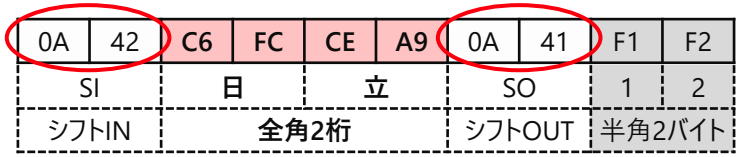
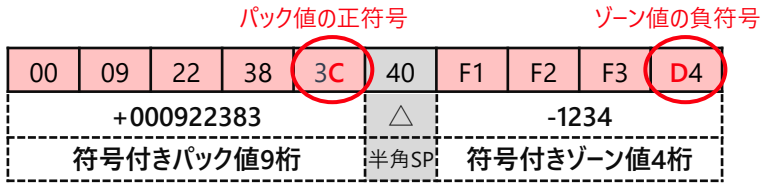
対象ファイル数は全10,000種類以上
・データ移行対象となる各JOBのINPUTファイル
・データ比較対象となる各JOBのOUTPUTファイル
・帳票ファイル
⇒ 調査が必要なデータ数は10,000を超えるため、効率的に正確に行えることがマイグレーション推進の鍵となった。

本章では、「実データからのレイアウト調査方法 と 品質確保の取り組み」についてご説明します。

HITACHI

4.2　実データからのレイアウト調査方法

実データからレイアウト（データ属性と長さ）の調査を実施。ツール化にて大量なデータも効率的に調査可能に。

No	調査手順	データ例（EBCDIK/KEIS）
全ファイル、全レコードを対象にNo.1～4の手順でレイアウト調査を実施。		
1	制御文字（シフトIN/OUT）の確認 ・x'0A42'-x'0A41'で囲まれている場合、全角文字と判断	
2	（シフトIN/OUTに囲まれていない全角データもあり） 2バイト毎に文字コードを判定し、KEIS全角文字範囲内の文字か判断 KEIS全角文字範囲外 9E91、9192、9240、40C6	
3	符号付き数値形式（ゾーン／パック）の確認 符号(C,D,F）付き数字かどうかを判断	
4	1バイトで文字コードを判定し、半角文字か判断	

HITACHI

4.2　実データからのレイアウト調査における課題と取り組み（その1）

課題①	全ファイルのレイアウト特定に向けて、対象の実データを現行環境から取得するが、 取得した日時によっては、現れないデータパターンが存在し、レイアウト精度が十分であるか分からない。

【対策】　受領する全ファイル、全データパターンを対象にレイアウト調査を実施。

一度調査完了したファイルに対しても、異なるデータとしてファイルを受領するたびに、レイアウト調査を実施し、
可能な限り調査できていないデータパターンの抜けをなくし、レイアウトを補正していった。

例）6バイトの項目

【1回目】

EBCDIK	…	40	40	40	40	40	40	…
変換定義	…	半角6バイト						…
字体	…	△	△	△	△	△	△	…

⇒全て半角スペース（x'40'）のため、**半角6バイト** とレイアウト判断

【2回目】

EBCDIK	…	F1	F2	F3	F4	F5	F6	…
変換定義	…	符号なしゾーン値6バイト						…
字体	…	1	2	3	4	5	6	…

⇒6桁数字項目のため、**符号なしゾーン値6バイト** とレイアウト修正

【3回目】

EBCDIK	…	F1	F2	F3	F4	F5	D6	…
変換定義	…	符号付きゾーン値6バイト						…
字体	…	1	2	3	4	5	-6	…

⇒最後の桁の上位4ビットの値により符号（ x'D6'）あり
と判断できたため、**符号付きゾーン値6バイト**とレイアウト修正

HITACHI

4.2　実データからのレイアウト調査における課題と取り組み（その2）

課題②	実データからはどうしても判断が難しいデータパターンが存在し、正しいレイアウトが特定できない

「**半角と符号付ゾーン値**」「**半角と符号付パック値**」のように、符号を表すコード（C,D,F）により、全ファイル、全データパターンを見ても判断が難しいパターンが存在する。

例）半角と符号付ゾーン値（ X型とS9型）

F7	F2	C1

半角文字　72A

符号付ゾーン値　+721

例）半角と符号付パック値（ X型とCOMP-3型）

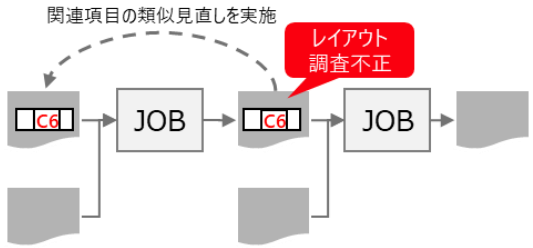
47	4C

半角文字　ｱ<

符号付パック値　+474

【対策】　組み合わせテスト以降～稼働直前まで、現新比較テストを実施し、文字コード変換の妥当性を確認。

総合テスト以降においても、業務視点での確認に加えて、現新比較を実施することで、
さまざまなデータバリエーションにおける文字コード変換結果の妥当性を確認した。



現新比較にて検出したレイアウト調査不正については、変換定義を修正し、再度実行、現新比較にて一致することを確認。
その上で、対象項目の関連箇所についてもレイアウトの再確認を実施して、品質の積み上げを実施。

HITACHI

4.3　実データからのレイアウト調査における結果

結果	• 実データからのレイアウト調査では、**99.6%の精度**でデータ構造（属性と位置）を特定することができた。 • 識別困難なデータパターンに対しても、現新比較テストにて抽出し、確実に横展開を実施することで、レイアウト調査において品質確保することができた。

取り組み① 実績

全10,000種のファイルに対して、23,150本の実データを受領し、レイアウト調査を実施。
6割のファイルは1回のみ、4割のファイルは2回以上データを受領し、再レイアウト調査を行った。

取り組み② 実績

組み合わせテスト～稼働直前まで、現新比較テストにて抽出した「レイアウト調査不正」は40件
全10,000種のファイルに対して、40件（0.4%）の発生率　**⇒99.6%の精度でレイアウト調査できた**

本番稼働後、文字コード変換（レイアウト調査）に関するインシデントは発生していない状況

おわりに

5

おわりに

本セッションでは、メインフレームからWindows環境への移行プロジェクト事例を通じて、
以下のポイントをご紹介しました。

- 独自の簡易言語をCOBOLへ変換する際に工夫したポイント
- ファイルレイアウト仕様書がない中でのデータ移行における、データ構造の調査方法と品質確保の取り組み

レガシーシステムの脱却においては、単なるツール変換だけでなく、現場の実態に即した調査・設計と品質管理が不可欠です。本事例が、同様の課題に直面されている皆様の参考となれば幸いです。

ご清聴ありがとうございました。

お問い合わせ先

ご質問は、以下のお問い合わせ先にご連絡ください

株式会社 日立製作所
アプリケーションサービス事業部
アプリケーション・モダナイゼーション本部
お問い合わせフォーム：
https://www.hitachi.co.jp/products/it/appsvdiv/service/migration/

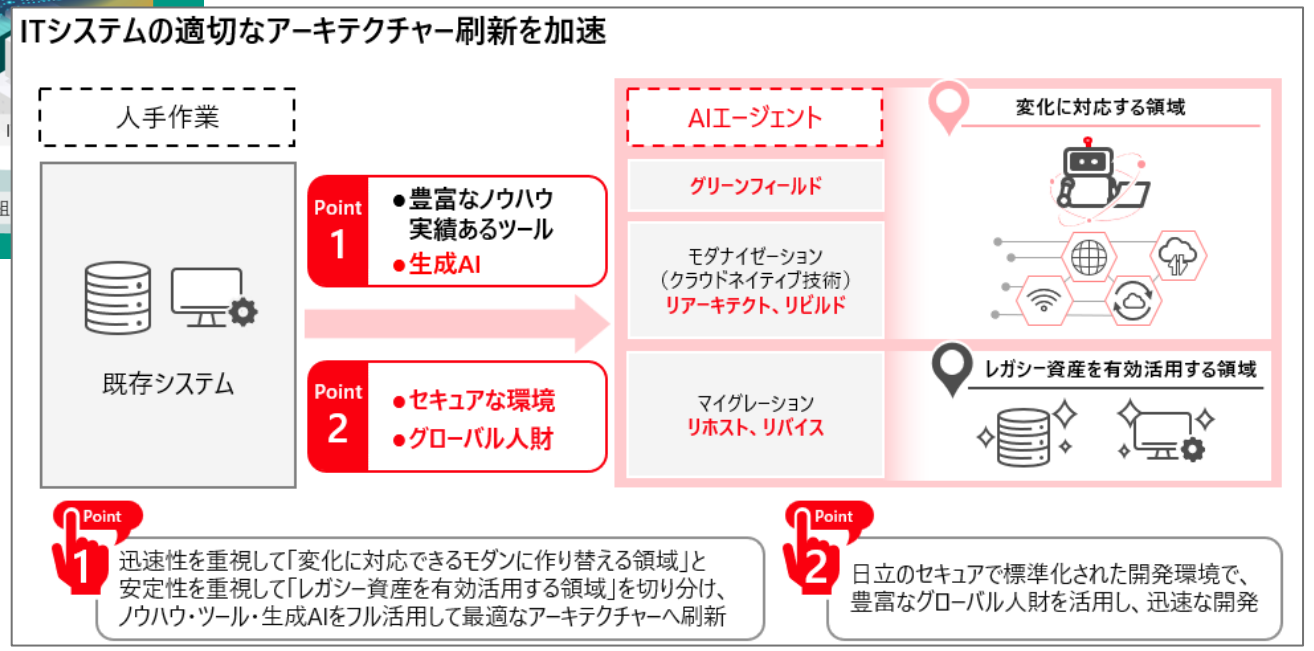
※上記フォームもしくは「マイグレーション サービス　日立」で検索いただき、
日立のマイグレーションサービスページからお問い合わせください

マイグレーション サービス　日立　検索

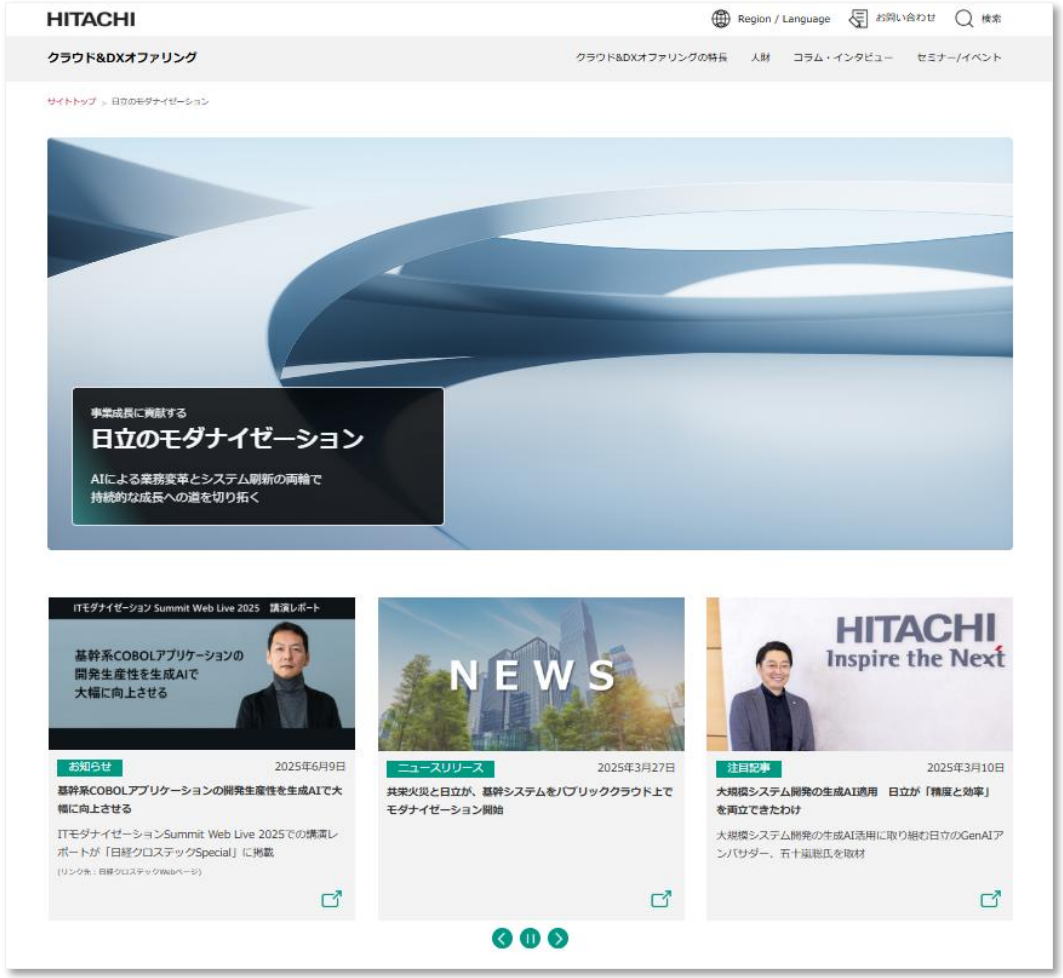
HITACHI

10/21 モダナイゼーション powered by Lumada を発表

AI ネイティブな基幹システムへ刷新する「モダナイゼーション powered by Lumada」を提供開始



日立のモダナイゼーションWebサイトのご案内



日立のモダナイゼーションの取り組みメッセージや、
最新ニュース、事例などを発信するサイトです。

ぜひご覧ください。

▶Webサイトを見る
https://www.hitachi.co.jp/products/it/CloudDX/modernization/index.html

他社商品名、商標などの引用、および略称に関する表示

【他社商品名、商標などの引用について】
Windows および Windows Server は、マイクロソフトグループの企業の商標です。

【英略称一覧】
LCDS：Line Conditioned Data Stream

【製品略称一覧】
- JP1/AJS ：JP1/Automatic Job Management System 3
- JP1/FTP：JP1/File Transmission Server/FTP
- JP1/IM：JP1/Integrated Management 2
- BJEX：uCosminexus Batch Job Execution Server
- XNF：Extended HNA based Communication Networking Facility
- AOMPLUS：Automatic Operation Monitor PLUS
- JSS3：Job Spooling Subsystem3


HITACHI