メインフレーム・マイグレーションによるCOBOL資産活用事例

ACOS2（メインフレーム）→NX7700（HP-UX）マイグレーション事例報告

1. 会社紹介
2. 事業内容
3. 基幹システム構成図
4. 当時システムの課題
5. マイグレーションを選択した理由
6. マイグレーション方式
7. 移行資産
8. 移行スケジュール
9. 移行におけるポイント
10. マイグレーションの成果
11. 今後の課題
12. まとめ
13. 変換事例紹介

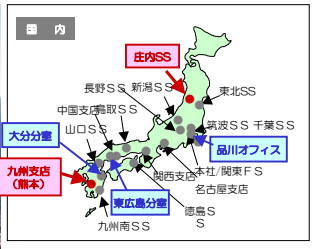
キヤノンアネルバ株式会社
情報システム本部　第二情報システムグループ
マネージャー
佐藤義昭

1. 会社紹介

社　　名	：キヤノンアネルバ株式会社
本社／工場	：神奈川県川崎市麻生区栗木2-5-1
富士工場	：山梨県南都留郡鳴沢村8532-28
支　　店	：9支店（国内6支店、海外3支店）
サービス拠点	：18拠点（国内14拠点、海外4拠点）
従 業 員	：連結1,479名（2007年12月末）
売 上 高	：連結623億円（2007年12月期実績）

（本社）

（富士工場）



2. 事業内容

エレクトロンデバイス装置事業 | パネルデバイス装置事業 | R&D装置

保守サービス

＜半導体／ストレージ＞
- スパッタリング装置
- ドライエッチング装置
- CVD装置

＜PDP／液晶＞
- 蒸着装置
- スパッタリング装置
- プラズマCVD装置
- 液晶注入装置

＜小型／実験用＞
- スパッタリング装置
- ドライエッチング装置
- 蒸着装置
- CVD装置

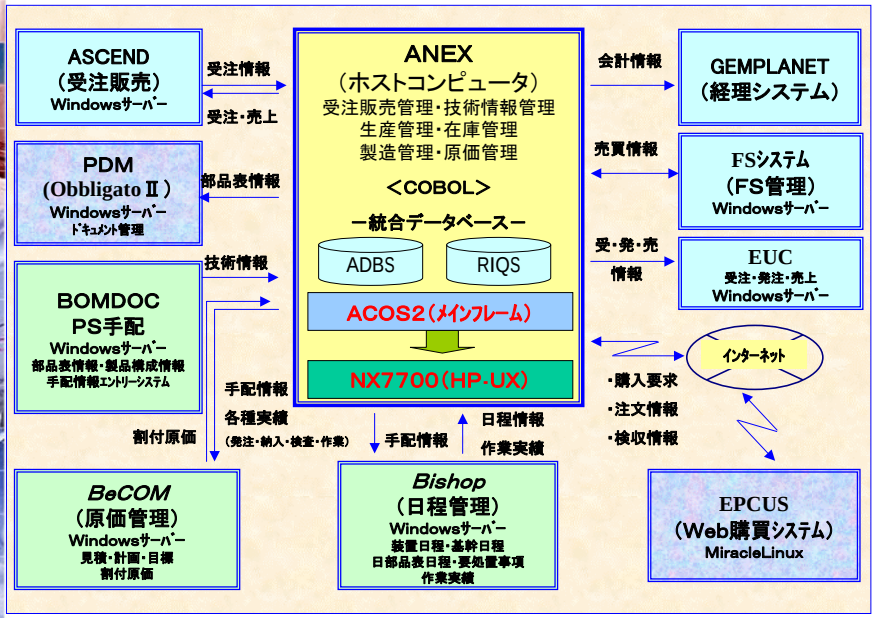
＜保守サービス＞
- オーバーホール
- 改造
- クリーニング
- リモート管理
- ユーザトレーニング

C-7100 TMR Head & MRAMスパッタリング装置

LCDスパッタリング装置

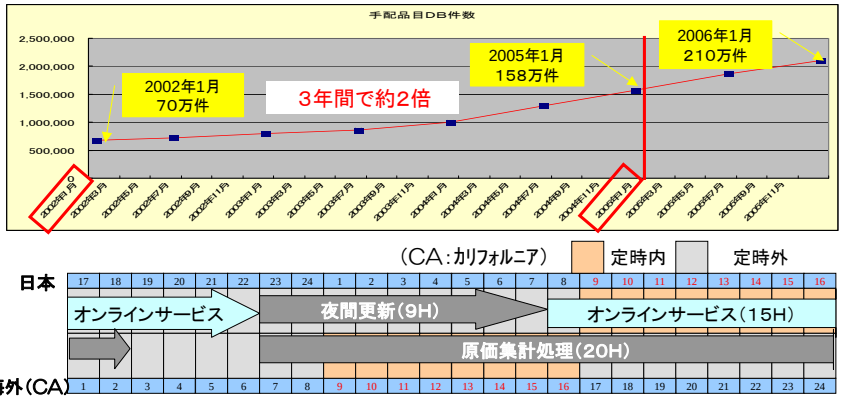
プラズマCVD装置

3. 基幹システム構成図



4. 当時システムの課題(1/2)

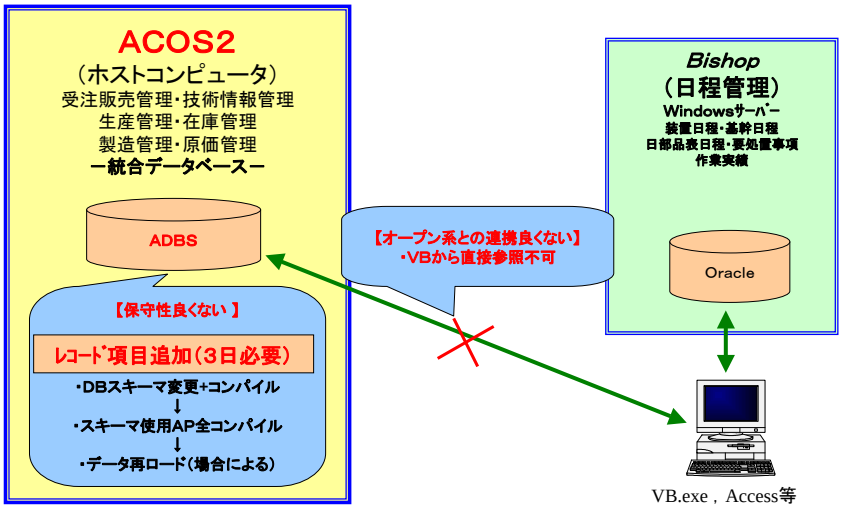
- ACOS2リース期限近づく：2006年3月（約20年間）
- データ量増加に伴い処理能力の限界、夜間更新が夜間の内に終了しない
  （夜間更新：9H，原価集計処理：20H）。。。10倍以上改善したい
- 海外拠点（CA）を考慮した24時間に近いオンラインサービスを提供したい



5

4. 当時システムの課題(2/2)

◎階層型DB（ADBS）の保守性及びオープン系との連携良くない



6

5. マイグレーションを選択した理由

当時システムの課題

- ACOS2リース期限
  - ・2006年3月リース満了
- ACOS2処理能力限界に近づく
  - ・データ量増加・処理能力に限界
- ACOS2を維持する問題点
  - ・ADBSの維持管理大変
  - ・オープン系連携良くない
  - ・保守性良くない

解決策①

【ACOS2のバージョンアップ】

（ipx7300M500→ipx7300M600V）

・短期移行可能

・階層型DB（ADBS）改善できず

・要求処理能力未達

解決策②

【ERP導入】

（IFS,Baan,SAP他）

・DBオープン化実現

・BPRにより業務プロセス変り

ユーザーの反発が予想される

解決策③

【ACOS→UNIXマイグレーション】

（NX7700i／3020M-8）

・DBオープン化実現

・プログラム資産移行→業務プロセス変更なし

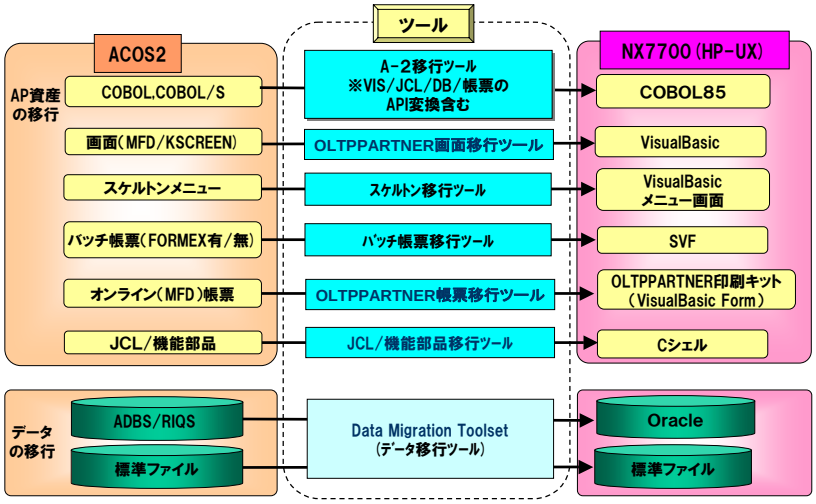
・iStorage使用で要求処理能力達成

・ACOS2→NX7700実績なし

7

6. マイグレーション方式

◆ 資産と移行ツール（方式）



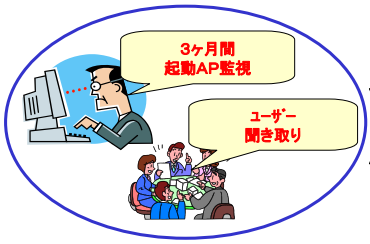
8

7. 移行資産

ACOS2

移行資産棚卸しを実施

【COBOLﾌﾟﾛｸﾞﾗﾑ】
7，069本



【移行ﾌﾟﾛｸﾞﾗﾑ】
3，328本
（47%）
－911JOB－

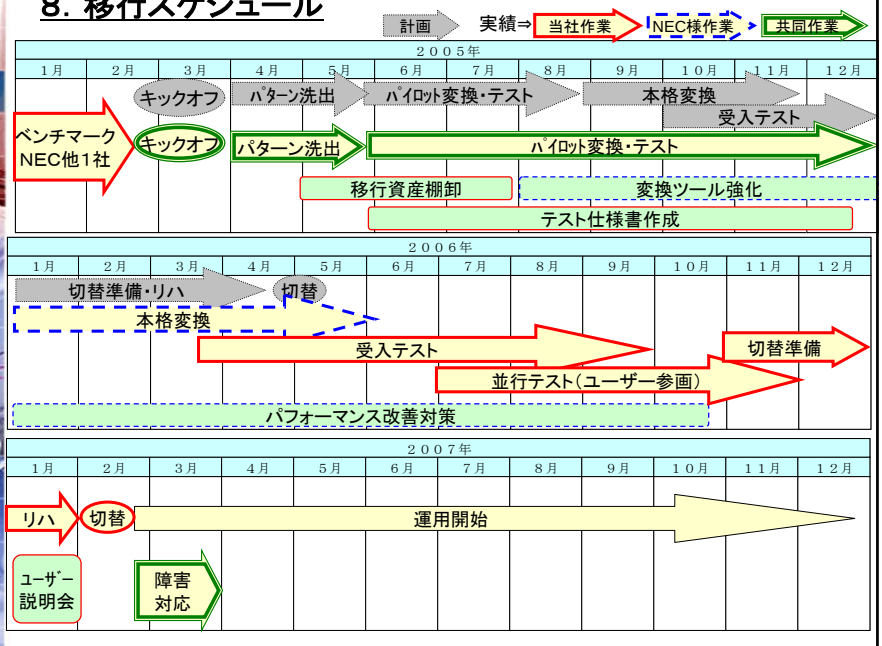
【不要ﾌﾟﾛｸﾞﾗﾑ】
3，741本
（53%）

半分以上もの不良資産を整理出来た

システム	プログラム			移行JOB	移行帳票	移行画面
	総本数	移行本数	移行率			
受注管理	1,018	383	38%	146	164	107
技術管理	523	81	15%	102	1	19
生産管理	2,683	936	35%	358	269	344
在庫管理	914	398	44%	183	145	137
製造管理	439	156	36%	112	57	55
FS管理	165	47	28%	10	9	0
マクロ	1,327	1,327	100%	－	－	－
合計	7,069	3,328	47%	911	645	662

9

8. 移行スケジュール



10

9. 移行におけるポイント(1/10)

【ベンチマーク】

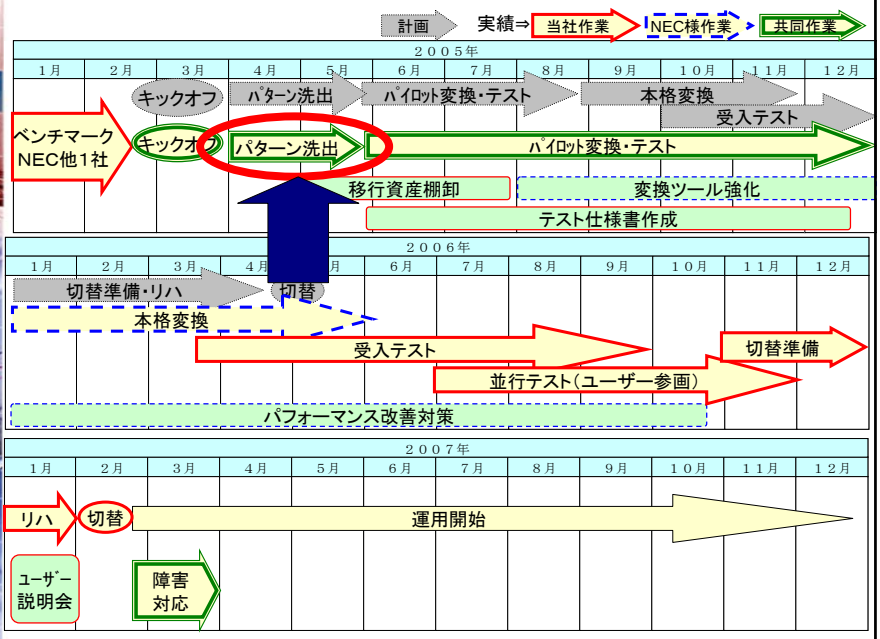
（目的） コンピューターの処理能力を比較・評価する

当社は当時問題となっていた原価集計処理（バッチ：処理時間20H）の評価を実施
ADBS移行／AP移行・処理結果／処理パフォーマンス ： 20H → 1H（20倍）

「パターン洗い出し」「パイロット変換」時に初めて解った事が沢山あった

◎可能な限り移行後の全体イメージが掴める様な評価実施を推奨

11



12

9. 移行におけるポイント(2/10)

【パターン洗い出し】

（目的） 実装されている“ACOS2+当社独自機能”を「明確」にする

◎ACOS2の世界では当たり前に使っていた機能がオープンの世界では異例

プログラム資産

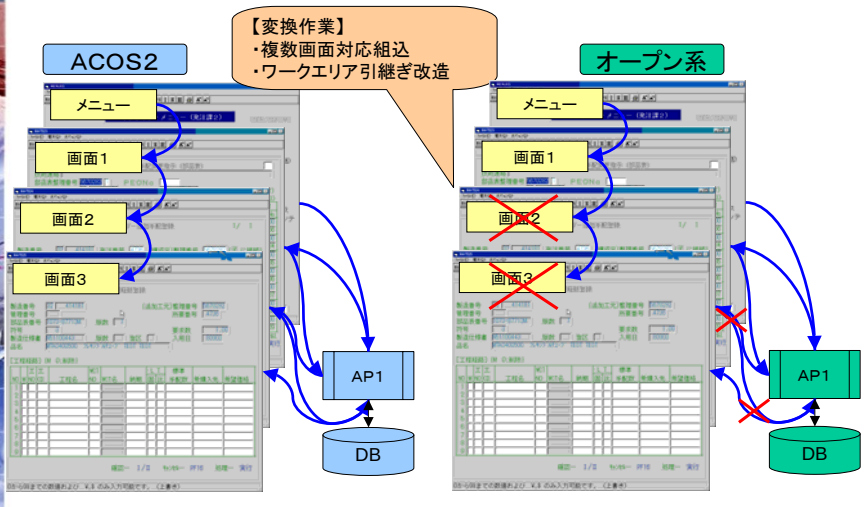
複数画面構成のオンラインプログラム	バッチ処理内ACCEPT命令を使用	オンラインプログラム￥VASSIGN文で制御
MFD画面機能標識でｶｰｿﾙ制御	処理メッセージ(ログ)管理(SYS@MSG)	バッチ+オンライン混合ジョブ有り
SEQﾌｧｲﾙの処理ﾊﾞｯﾃｨﾝｸﾞはOS側で自動ｳｴｲﾄ！	DBアクセス処理	帳票の出力方法対話・自動ﾗｲﾀ、VIS帳票
：	：	：

13

9. 移行におけるポイント(3/10)

【パターン洗い出し】(**複数画面構成のオンラインプログラム**)

・ACOS2：複数画面OK　／　オープン系：1画面1処理完結



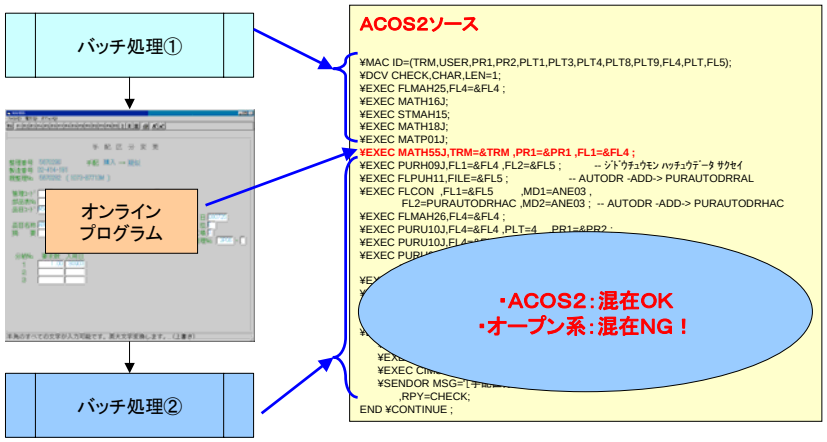
14

9. 移行におけるポイント(4/10)

【パターン洗い出し】(バッチ+オンラインの混合ジョブ)

・ACOS2　:バッチ処理+オンライン処理:JCLで指定するだけ

・オープン系 :バッチはバッチのみ　／　オンラインはオンラインのみ



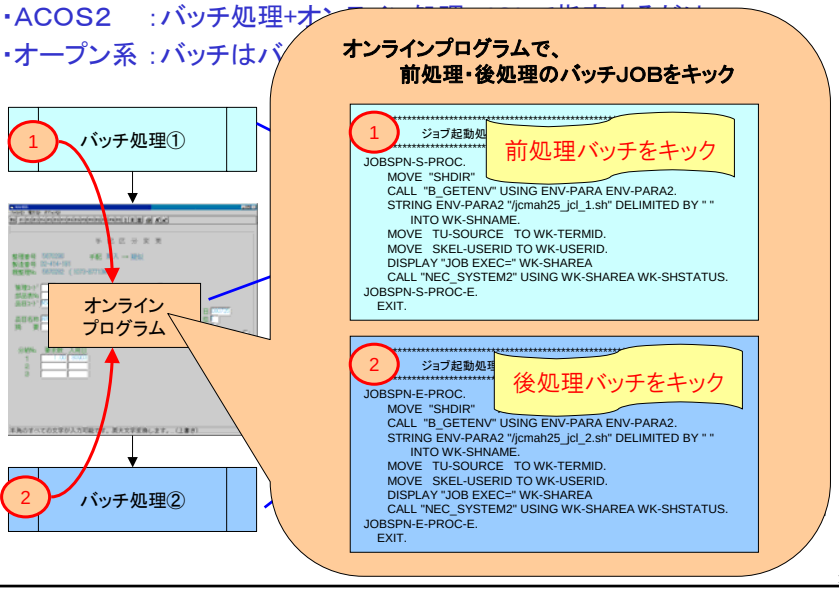
15

9. 移行におけるポイント(5/10)

【バッチ+オンラインの混合ジョブ】

・ACOS2　:バッチ処理+オ

・オープン系 :バッチはバ



16

9. 移行におけるポイント(6/10)

【パターン洗い出し】

（目的） 実装されている“ACOS2+当社独自機能”を「明確」にする

◎ACOS2の世界では当たり前に使っていた機能がオープンの世界では異例

プログラム資産

複数画面構成の

バッチ処理中

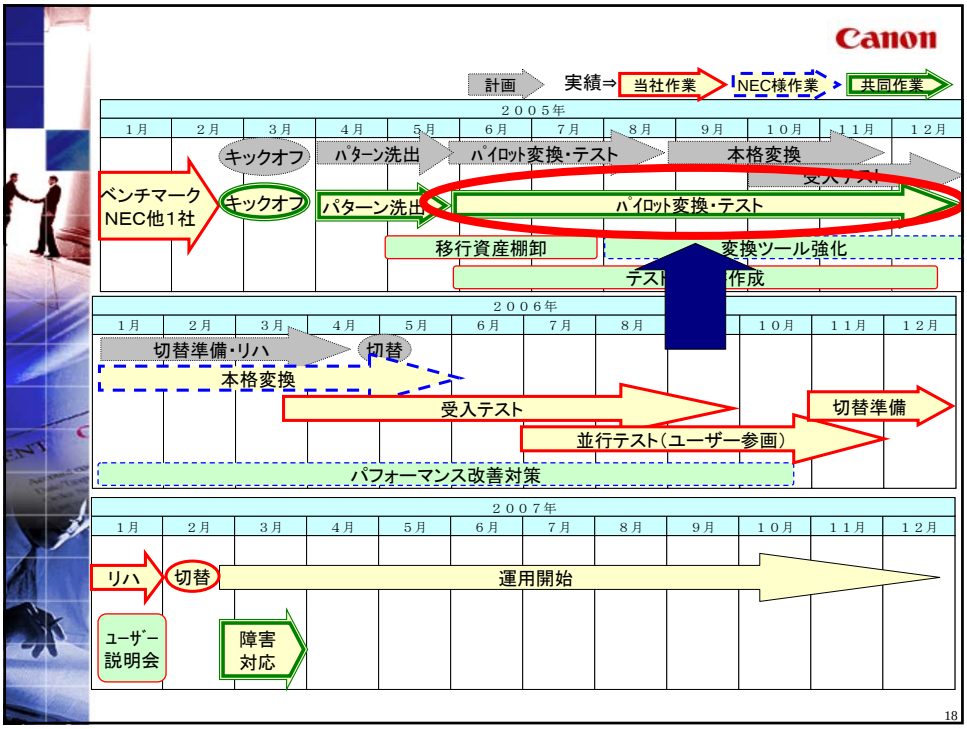
帳票の出力方法
対話・自動ﾗｲﾀ、VIS帳票

はOS側で自動ｳｴｲﾄ！

DBアクセス処理

ACOS2 × オープンの違いを明確にし
品質アップに繋げる！！

17



18

9. 移行におけるポイント(7/10)

【パイロット変換・テスト】

（目的） 実装されている“ACOS2+当社独自機能”の「変換方式確立」

◎仕上がり状態の見極め重要（品質、ソースコーディング他）

ACOS2

【移行ﾌﾟﾛｸﾞﾗﾑ】
3，328本
・ﾌﾟﾛｸﾞﾗﾑ：2,001本
・ﾏｸﾛ：1,327本
－911JOB－

パイロット対象（全体10%）
・ﾌﾟﾛｸﾞﾗﾑ：187本（9%）
・JOB　：71JOB（8%）

JOBパターン分類

ｼｽﾃﾑ	JCL－ID	処理名	スケルトン	画面遷移	画面前進後退	画面レスポンス	画面重ね合せ	画面部分送信	ACCEPT（SENDOR）	DATA検証	次TPP呼出	mail（TPP間通信）	WC起動OL	WC起動OL+BAT連携	バッチ（大）	VB連携（既存）
BMP	BMPH08	製造仕様書承認								○						○
BMP	BMPH00	通常手配指示書承認								○						○
ICS	FB0010J	品目マスタメンテナンスTR入力	○	○						○			○			
ICS	FB0030J	品目TR-構成TRマージ	○							○					○	
ICS	FB0050J	品目マスタ・構成マスタ	○							○					○	
ICS	FB0020J	構成マスタメンテナンスTR入力	○	○				○		○			○			
ICS	FB0040J	カードファイルチェック	○							○					○	
ICS	FD3600J	自動ﾒﾝﾃ更新								○	○					
ICS	FH4000J	入出庫状況問い合わせ(リアル)		○	○	○		○			○					
ICS	FJI200FS	持ち(出)処理	○							○						
ICS	FJ2000J	入庫処理									○					
SAL	JCSOC09T	受注月次統計処理	○						○						○	
SAL	SALA50J	受注販売問合せ		○	○	○	○	○				○				
SAL	JCSCC15	入金登録処理	○	○	○	○	○	○		○				○		
SAL	SODC29J	出荷伝票即時発行(OAﾌﾟﾘﾝﾀｰ)								○						
SAL	SODC34J	(所要登録データ)特急払い出し処理				○	○	○			○					
SAL	SODB02J	販売ファイルﾒﾝﾃﾅﾝｽ	○	○	○	○	○	○		○			○			
SAL	SODC02J	受注変更更新処理		○		○	○	○		○						
SFC	SFCC50J	部門別工数集計表	○				○	○								

19

9. 移行におけるポイント(8/10)

【パイロット変換・テスト】

計画：3ヶ月　→　実績：7ヶ月

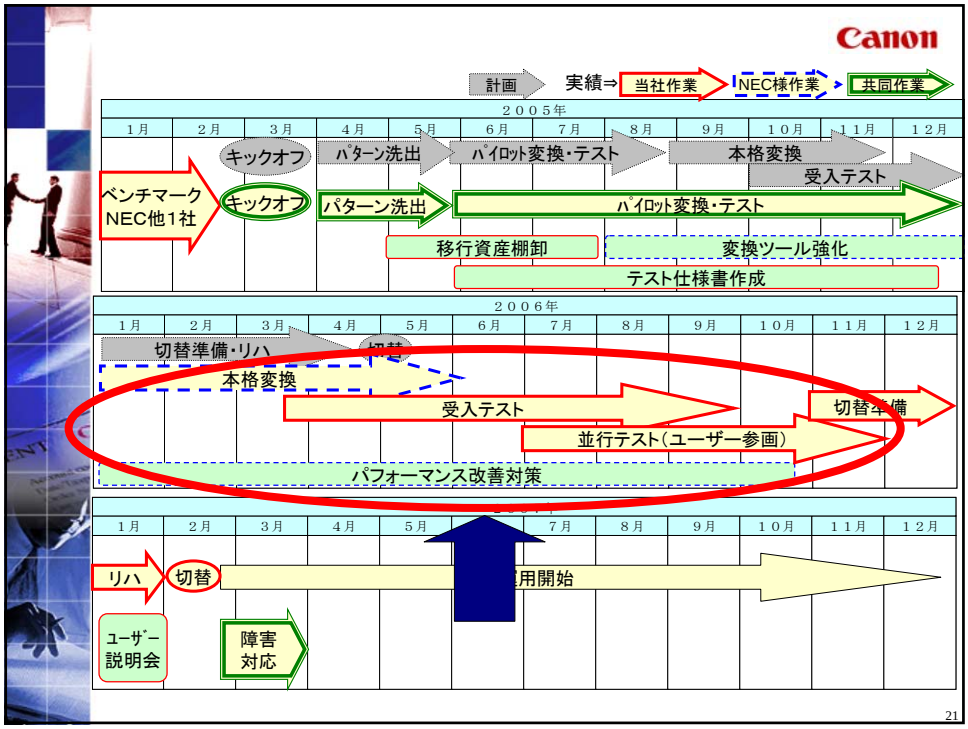
① パイロット変換・テスト：1回目
- ・バッチJOB　：異常終了が目立つ
- ・パフォーマンス：出ないケースがある（特にDB全件順処理JOB）
- ・オンラインAP　：不安定（複数画面構成AP）

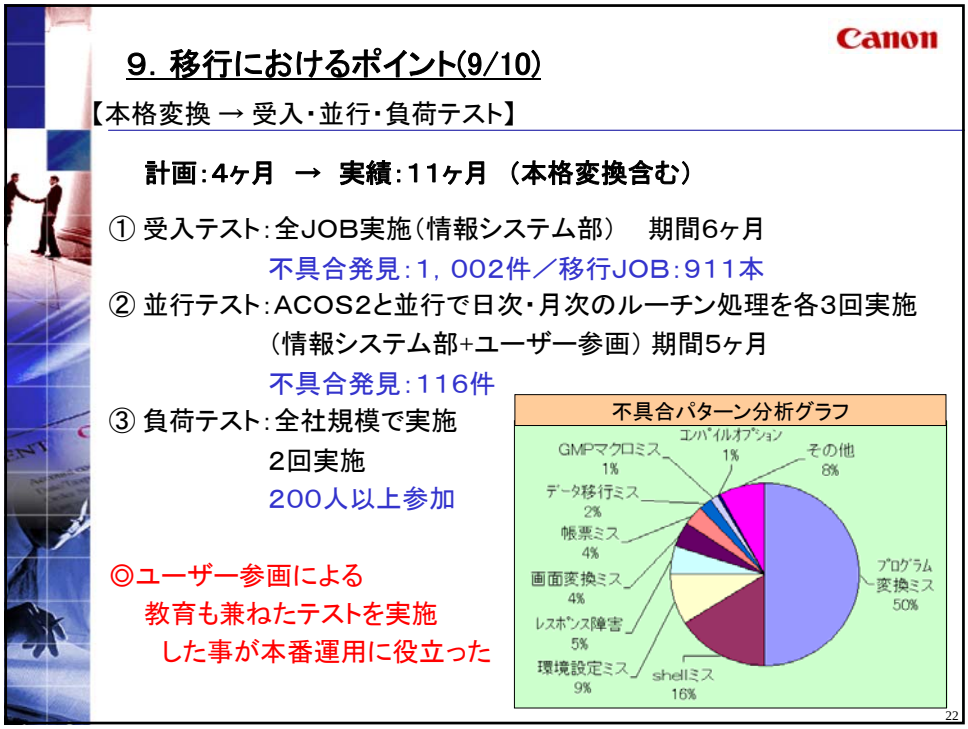
② 変換方式再検討（NEC様にて再検討）
- 変換ツール改善、DBアクセス部の部品（GMP）化、パフォーマンス対策
- オンラインAP変換方法検討、オンライン画面ACOSに近いキー操作実現、他

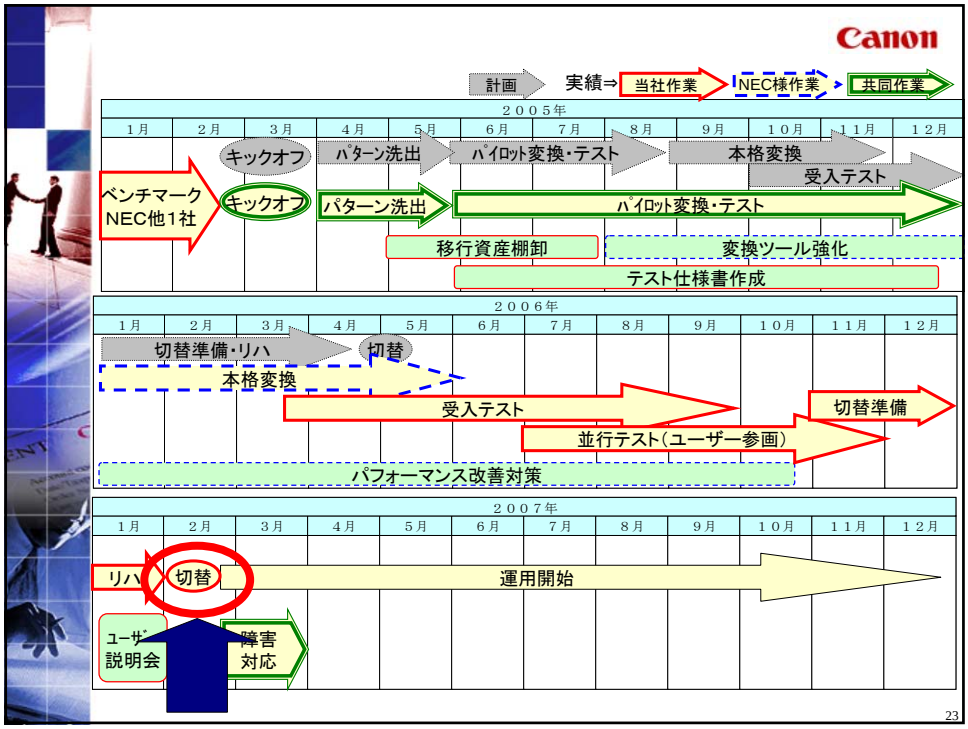
③パイロット変換・テスト：2回目
- ・バッチJOB　：正常に動作する様になった
- ・パフォーマンス：向上した（特にDB全件順処理JOB）
- ・オンラインAP　：動作安定（複数画面構成AP）

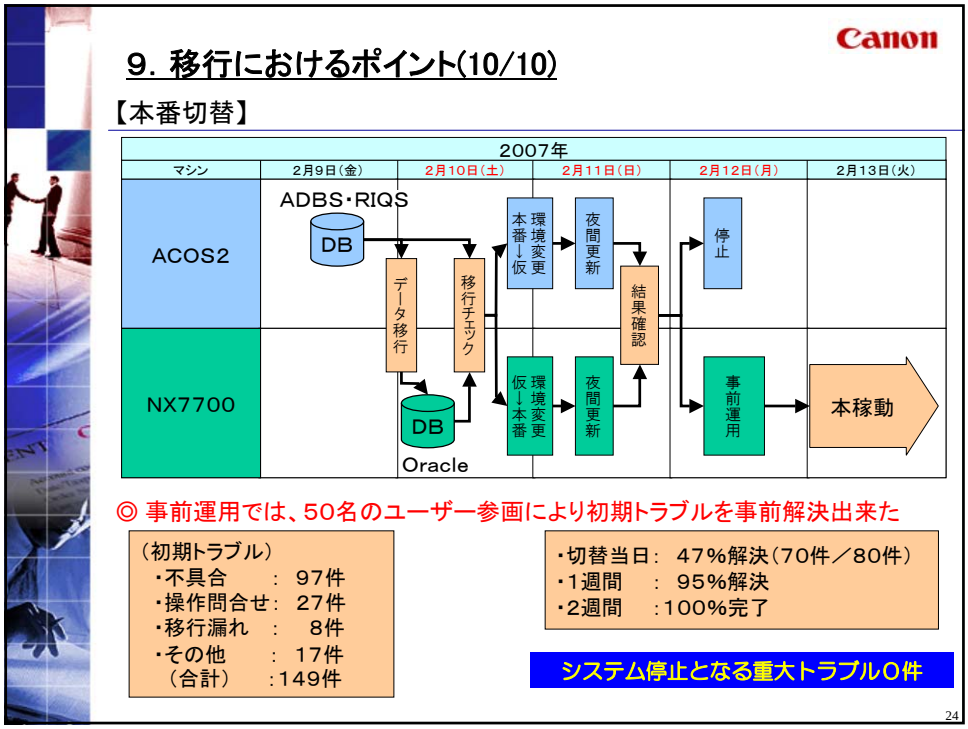
◎ACOS2の変換方式決定　→　本格変換を実施する事に決定

20


Canon
計画
実績⇒
当社作業
NEC様作業
共同作業
２００５年
１月 ２月 ３月 ４月 ５月 ６月 ７月 ８月 ９月 １０月 １１月 １２月
キックオフ
ﾊﾟﾀｰﾝ洗出
ﾊﾟｲﾛｯﾄ変換・テスト
本格変換
受入テスト
ベンチマーク
NEC他1社
キックオフ
パターン洗出
ﾊﾟｲﾛｯﾄ変換・テスト
移行資産棚卸
変換ツール強化
テスト仕様書作成
２００６年
１月 ２月 ３月 ４月 ５月 ６月 ７月 ８月 ９月 １０月 １１月 １２月
切替準備・リハ
本格変換
受入テスト
切替準備
並行テスト（ユーザー参画）
パフォーマンス改善対策
１月 ２月 ３月 ４月 ５月 ７月 ８月 ９月 １０月 １１月 １２月
リハ
切替
用開始
ユーザー
説明会
障害
対応
21


Canon
9．移行におけるポイント(9/10)
【本格変換 → 受入・並行・負荷テスト】
計画：4ヶ月 → 実績：11ヶ月 （本格変換含む）
① 受入テスト：全JOB実施（情報システム部） 期間6ヶ月
不具合発見：1，002件／移行JOB：911本
② 並行テスト：ACOS2と並行で日次・月次のルーチン処理を各3回実施
（情報システム部+ユーザー参画）期間5ヶ月
不具合発見：116件
③ 負荷テスト：全社規模で実施
2回実施
200人以上参加
◎ユーザー参画による
教育も兼ねたテストを実施
した事が本番運用に役立った
不具合パターン分析グラフ
ｺﾝﾊﾟｲﾙｵﾌﾟｼｮﾝ 1%
GMPマクロミス 1%
その他 8%
ﾃﾞｰﾀ移行ミス 2%
帳票ミス 4%
画面変換ミス 4%
ﾌﾟﾛｸﾞﾗﾑ変換ミス 50%
ﾚｽﾎﾟﾝｽ障害 5%
環境設定ミス 9%
shellミス 16%
22


Canon
計画
実績⇒
当社作業
NEC様作業
共同作業
２００５年
１月 ２月 ３月 ４月 ５月 ６月 ７月 ８月 ９月 １０月 １１月 １２月
キックオフ
パターン洗出
パイロット変換・テスト
本格変換
受入テスト
ベンチマーク
NEC他1社
キックオフ
パターン洗出
パイロット変換・テスト
移行資産棚卸
変換ツール強化
テスト仕様書作成
２００６年
１月 ２月 ３月 ４月 ５月 ６月 ７月 ８月 ９月 １０月 １１月 １２月
切替準備・リハ
切替
本格変換
受入テスト
切替準備
並行テスト（ユーザー参画）
パフォーマンス改善対策
２００７年
１月 ２月 ３月 ４月 ５月 ６月 ７月 ８月 ９月 １０月 １１月 １２月
リハ
切替
運用開始
ユーザー説明会
障害対応
23


Canon
9．移行におけるポイント(10/10)
【本番切替】
2007年
マシン
2月9日(金)
2月10日(土)
2月11日(日)
2月12日(月)
2月13日(火)
ACOS2
NX7700
ADBS・RIQS
DB
データ移行
移行チェック
環境変更 本番→仮
夜間更新
結果確認
停止
環境変更 仮→本番
夜間更新
事前運用
本稼動
DB
Oracle
◎ 事前運用では、50名のユーザー参画により初期トラブルを事前解決出来た
（初期トラブル）
・不具合 ： 97件
・操作問合せ: 27件
・移行漏れ ： 8件
・その他 ： 17件
（合計） :149件
・切替当日: 47%解決(70件／80件)
・1週間 ： 95%解決
・2週間 :100%完了
システム停止となる重大トラブル0件
24

10. マイグレーション成果(1/4)

【指標値】

◎パフォーマンス改善には苦しんだが無事指標値クリアー

JOB区分	指標値（期待値）	評価判定	検証機能	レスポンス ACOS	レスポンス NX	倍率
夜間更新	ﾌｪｰｽﾞ①：10倍	○	7月14日：夜間更新	7時間31分	52分	5～12倍
			8月27日：夜間更新	4時間35分	54分	
			9月18日：夜間更新	8時間40分	46分	
	ﾌｪｰｽﾞ②：15倍	○	7月14日：夜間更新	4時間35分	37分	MAX16倍
			9月18日：夜間更新	8時間40分	32分	
原価集計	ﾌｪｰｽﾞ①：10倍～13倍	○	BOMDOC原価集計（BMPA78）	5時間7分	30分	10倍
	ﾌｪｰｽﾞ②：20倍	○	同上	20時間	1時間	20倍
オンライン（一覧）	ﾌｪｰｽﾞ①：同等	○	CIMA50（一覧）	1秒未満	1秒未満	同等～5倍
			CIMA55（一覧）	1秒未満	1秒未満	
			発注残一覧	1～2秒	1～2秒	
	ﾌｪｰｽﾞ②：5倍	○	所要発注状況問合せ	1秒未満	1秒未満	
			受注状況問合せ	1～5秒	1秒	
オンライン（一般）	ﾌｪｰｽﾞ①：同等	○	CIMA50（整理番号）	1秒未満	1秒未満	同等
			販売ファイル問合せ	一瞬	一瞬	
	ﾌｪｰｽﾞ②：同等	○	オーダー情報問合せ	1秒未満	1秒未満	
			取引先マスター問合せ	1秒未満	1秒未満	
バッチ一般	ﾌｪｰｽﾞ①：10倍	○	入金予定用ファイル作成	120秒	20秒	6倍
	ﾌｪｰｽﾞ②：10倍	○	概算特採未処理一覧	26分25秒	39秒	41倍

25

10. マイグレーション成果(2/4)

【パフォーマンス対策】

◎DBアクセス処理時間短縮対策

- SQL分析と最適化（不要I-Oの抑制、動的SQL化など）
- DBアクセス部品見直し（カーソル制御コンパイルオプションなど）
- インデックスの最適化など

◎オンラインAP負荷集中時のレスポンス対策

- シングルプロセス構造AP→マルチプロセス構造APに改造

. . . . . NEC様報告より

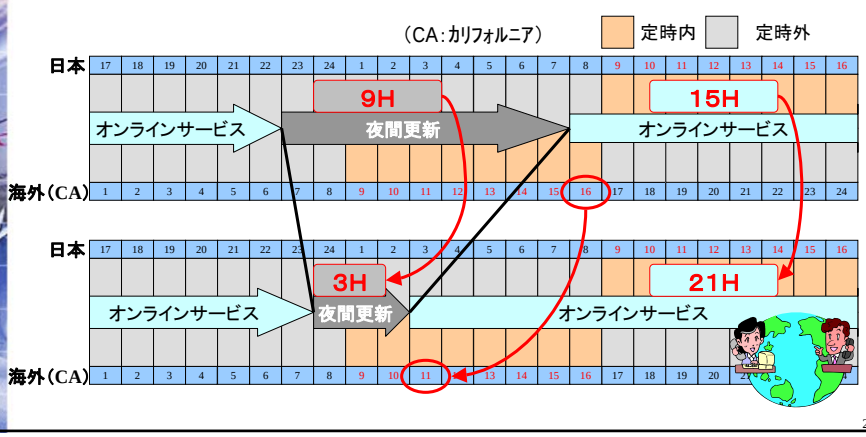
26

10. マイグレーション成果(3/4)

夜間更新（バックアップ等の前処理含む）を約1／3に短縮

◎オンラインサービス時間：15H→21H（1．4倍）に拡大

◎海外拠点（CA）のオンライン開始時間16時→11時に出来た



27

10. マイグレーション成果(4/4)

- プラットフォーム切替ても業務ノウハウを継承する事が出来た。
- DBをオープン化する事が出来た。
- システムの不良資産を削除出来た。
  （資産が半分以下となった）
- CSシステムとの連携強化がより可能となった
- 巨大プロジェクト成功の自信がついた。

28

11. 今後の課題

- NX7700の構成をフルに利用した処理改善
- UNIX上での開発能力スキルアップ必要
- UNIX言語知識のスキルアップ必要
- 既存のプログラムは1件毎の処理、SQL命令の利点である一括処理との融合
- :
- :
- :

29

12. まとめ

- UNIX（オープン）でもCOBOL資産を十分に活用する事が出来た。
- マイグレーションを完了後、1年半経過しているがNX7700での運用が定着出来ている。

30

13. 変換事例紹介

変換事例

「NECのマイグレーションサービスでの変換結果です。
最新の変換方式では変わっている可能性があります。」

31

ADBSアクセスの変換例

（旧）ソース

```
FIND AND GET    FIRST   STRUCT-R    WITHIN  PRNT-SET.
MSG-DML-PROC2   :=  "#020"
CALL: DML-CHK1-PROC
```

（新）ソース

```
＊＊FIND AND GET    FIRST   STRUCT-R    WITHIN  PRNT-SET. コメントアウト
¥ADBS_FIND_AND_GET TYP=4,LOC=FIRST,
REC=STRUCT-R,WITHIN=PRNT-SET,                ・・・・・・・・・ GMPマクロ追加
WITHINATTR=SET;
MOVE “#020”     TO MSG-DML-PROC2             ・・・・・・・・・・ 変換
PERFORM DML-CHK1-PROC THRU DML-CHK1-PROC-E    （MOVE,PERFORM）
```

【メリット】
- 旧ソース行がコメントアウトで残っているので、変換後のソースを解読し易い

【デメリット】
- 旧ソースに比べて新ソースはコーディング行が増えてしまう

32

RIQSアクセスの変換例

（旧）ソース

```
SAL-CUONBR  :=  TD-C-CUONBR
READ  SALE-F  RECORD  KEY  IS  SAL-KEY   INVALID   MOVE  1  TO  INV-SW.
MSG-DML-PROC  :=  "SODA10-PROC  #010"
CALL:DML-CHK3-PROC
```

（新）ソース

```
MOVE TD-C-CUONBR TO SAL-CUONBR.
＊＊READ  SALE-F  RECORD  KEY  IS  SAL-KEY INVALID  MOV . . コメントアウト
¥SQLREAD_KEY SALSALES,KEY='SAL-KEY';     . . . . . GMPマクロ追加
IF SQLCODE > 0
   MOVE  1  TO  INV-SW                  . . . INVALID句をSQLステータス判定へ変換
END-IF                .
MOVE "SODA10-PROC  #010" TO MSG-DML-PROC.          . . . . 変換
PERFORM DML-CHK3-PROC THRU DML-CHK3-PROC-E.        （MOVE,PERFORM）
```

【メリット】
- 旧ソース行がコメントアウトで残っているので、変換後のソースを解読し易い

【デメリット】
- 旧ソースに比べて新ソースはコーディング行が増えてしまう
- READとINVALID句が2分割となってしまう

33

ストレートコンバージョンの課題

```
FIND  AND  GET  NEXT   MIDMND-R  WITHIN  PDDM-SET
¥ADBS_FIND_AND_GET TYP=4,LOC=NEXT,
REC=MIDMND-R,WITHIN=PDDM-SET,
WITHINATTR=SET;
 COMPUTE WK-MDM-CNT = WK-MDM-CNT +  1
 MOVE "S"        TO MDM-CNDSTS

 PERFORM BRANCH-SET-PROC THRU BRANC
 PERFORM BRCHST-SET-PROC THRU BRCHST
END-IF
MODIFY  MIDMND-R
¥ADBS_MODIFY REC=MIDMND-R;
MOVE "KOSEI-SET-P #030" TO MSG-DML-PR
PERFORM DML-CHK2-PROC THRU DML-CHK2-P
IF        (   DML-LOCK)
   GO TO KOSEI-SET-PROC-E
END-IF
```

1件毎の処理はそのまま！

```
select ＊＊＊＊＊
  from ＊＊＊＊＊
  where ＊＊＊＊＊
  order by ＊＊＊＊＊
```

【メリット】
- ロジックはそのままなのでファイルアクセスのタイミングは変わらず変換時のバグは少ない

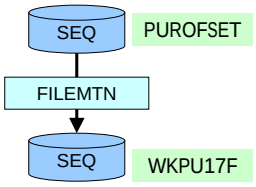
【デメリット】
- SQL分の利点を使いこなしていない。（1件毎の処理はそのまま！）パフォーマンス頭打ち

34

従来ファイルコピーの変換例

（旧）ソース

```
¥FILEMTN  ACT=COPY,
        INFILE=(FILE=PUROFSET,MEDIA=ANE03),
       OUTFILE=(FILE=WKPU17F,MEDIA=ANE03) ;
```



（新）ソース

```
#0004#¥FILEMTN  ACT=COPY,
#0005#         INFILE=(FILE=PUROFSET,MEDIA=ANE03),
#0006#        OUTFILE=(FILE=WKPU17FN,MEDIA=ANE03) ;
echo "`date +%x` `date +%X` [$$:${SHNM}] FILEMTN START : flpuk01n" | tee -a "${LOGFILE}"
cp -f ${TRNDIR}/ane03/purofset ${TRNDIR}/ane03/wkpu17fn |& tee -a "${LOGFILE}"
setenv SYSTEM_STATUS $status
echo "`date +%x` `date +%X` [$$:${SHNM}] FILEMTN FINISH(cp) :
                                    flpuk01n(STATUS:${SYSTEM_STATUS})" | tee -a "${LOGFILE}"
echo "`date +%x` `date +%X` [$$:${SHNM}] FILEMTN FINISH :
                                     flpuk01n(STATUS:${SYSTEM_STATUS})" | tee -a "${LOGFILE}"
if ($SYSTEM_STATUS != 0) then
   goto ELOGOUT
endif
```

. . . . . コメントアウト

. . . 変換

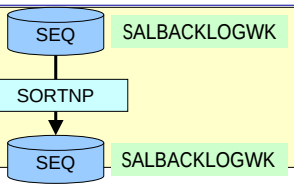
. . . . . 処理DISPLAY表示追加

35

従来ファイルソートの変換例

（旧）ソース

```
¥SORTNP
      INFILE=  (FILE=SALBACKLOGWK,MEDIA=ANE01 ,SKIPDL,BSZ=22528 ),
      OUTFILE= (FILE=SALBACKLOGWK,MEDIA=ANE01 ,BSZ=22528 ),
      WKFILE=  (FILE=SORTWK,AU=TRK,SZ=21,INCSZ=10,MEDIA=ANE06 ,TEMP);
¥OPTION  ;
      FUNCTION: BASEYEAR=75
      RECORD: KEY=(23 3) (149 10) (49 6 UDDATE)
      ENDSORT:
¥ENDOPTION ;;
```



（新）ソース

```
cat <<//EOD >${WRKDIR}/${JOBNM}_$$_01
/FUNCTION DUPREC=FIFO,ALTSEQ=JTEK,BASEYEAR=75
/RECORD RSZ=256,
KEY=22,3,148,10,48,2,YEAR,50,4
/END
//EOD
cat <<//EOD >${WRKDIR}/${JOBNM}_parfile_01
INFILE ${TRNDIR}/ane01/salbacklogwk,SEQ
OTFILE ${TRNDIR}/ane01/salbacklogwk,SEQ
WKDIRE ${WRKDIR}/${JOBNM}_$$_sortwk
REPOUT ${LOGFILE},"### SORT REPORT ###"
DIFILE ${WRKDIR}/${JOBNM}_$$_01
Q
//EOD
### Start sortkit ###
   echo "`date +%x` `date +%X` [$$:${SHNM}] SORT START : stsbc01" | tee -a "${LOGFILE}"
   nsort -p ${WRKDIR}/${JOBNM}_parfile_01 |& tee -a "${LOGFILE}"
   setenv SYSTEM_STATUS $status
   if(-e ${WRKDIR}/${JOBNM}_$$_01) then
      rm -f ${WRKDIR}/${JOBNM}_$$_01
   endif
   if(-e ${WRKDIR}/${JOBNM}_parfile_01) then
      rm -f ${WRKDIR}/${JOBNM}_parfile_01
   endif
   echo "`date +%x` `date +%X` [$$:${SHNM}] SORT FINISH : stsbc01(STATUS:${SYSTEM_STATUS})" | tee -a "${LOGFILE}"
   if (${SYSTEM_STATUS} != 0) then
      goto ELOGOUT
   endif
```

コーディング量
かなり多くなる

36

RIQSファイルコピーの変換例

（旧）ソース

```
¥MAC      ID=(FL1,MDA=ANE15 );
¥SENDO MSG='P／S手配-MATMREQPS(RIQS) --> &FL1  - FLMAT06 -';
¥FILEMTN ACT=COPY ,
     INFILE=(FILE=MATMREQPP ,FILEFORM=RIQS),
     OUTFILE=(FILE=&FL1 ,MEDIA=&MDA ,INCSZ=100);;
```

RIQS　MATMREQPP
①FILEMTN
SEQ　＆FL1

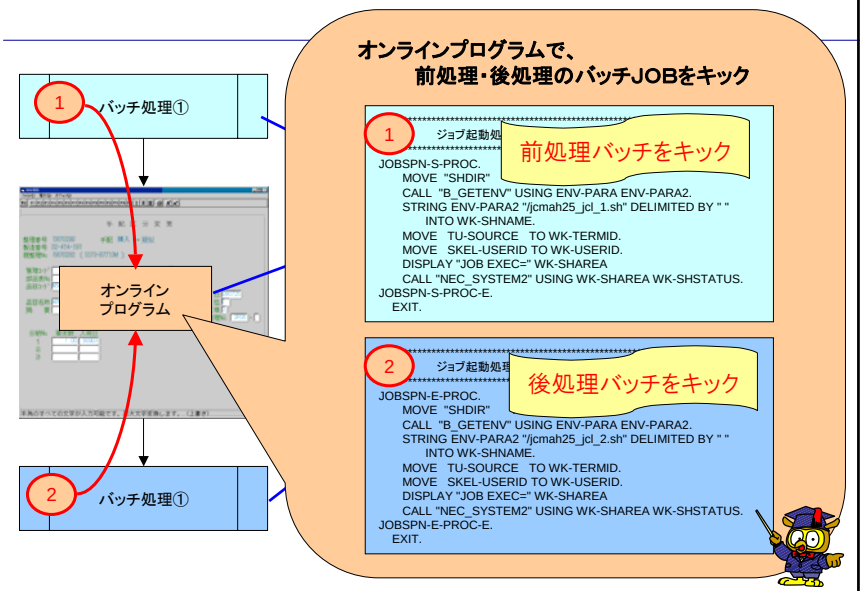
（新）ソース

```
${BINDIR}/MATMREQPP.sh tee -a "${LOGFILE}"
   setenv SYSTEM_STATUS $status
   echo "`date +%x` `date +%X` [$$] FILEMTN FINISH(MATMREQPP): flmat06(STATUS:${SYSTEM_STATUS})" | tee -a "${LOGFILE}"
   if ($SYSTEM_STATUS != 0) then
      goto FILEMTN_1
   endif
   mv -f ./MATMREQPP.dat ${T                    no ${p                    ti                    ${LOGFILE}"
   setenv SYSTEM_STATUS $s
   echo "`date +%x` `date +%X`                    TATUS:${SYSTEM_STATUS})" | tee -a "${LOGFILE}"
FILEMTN_1:
   echo "`date +%x` `date +%X`                    US:${SY                    -a "${LOGFILE}"
   if ($SYSTEM_STATUS != 0) t
      goto ELOGOUT
   endif
```

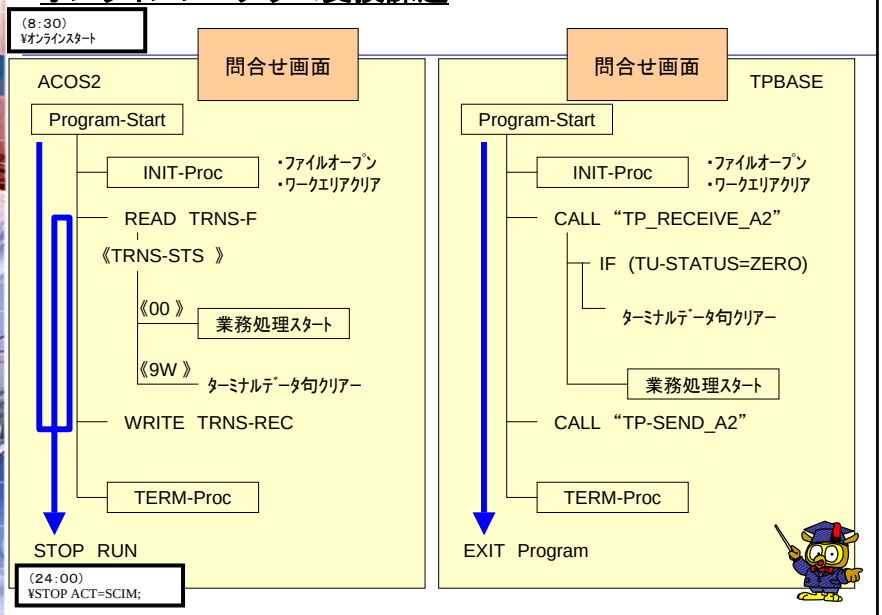
oracle　MATMREQPP
①COBOL
txt　＆FL1
②mv
txt　＆FL1

2段階の処理となる

バッチ処理+オンラインの混合ジョブ変換例



オンラインプログラム変換課題

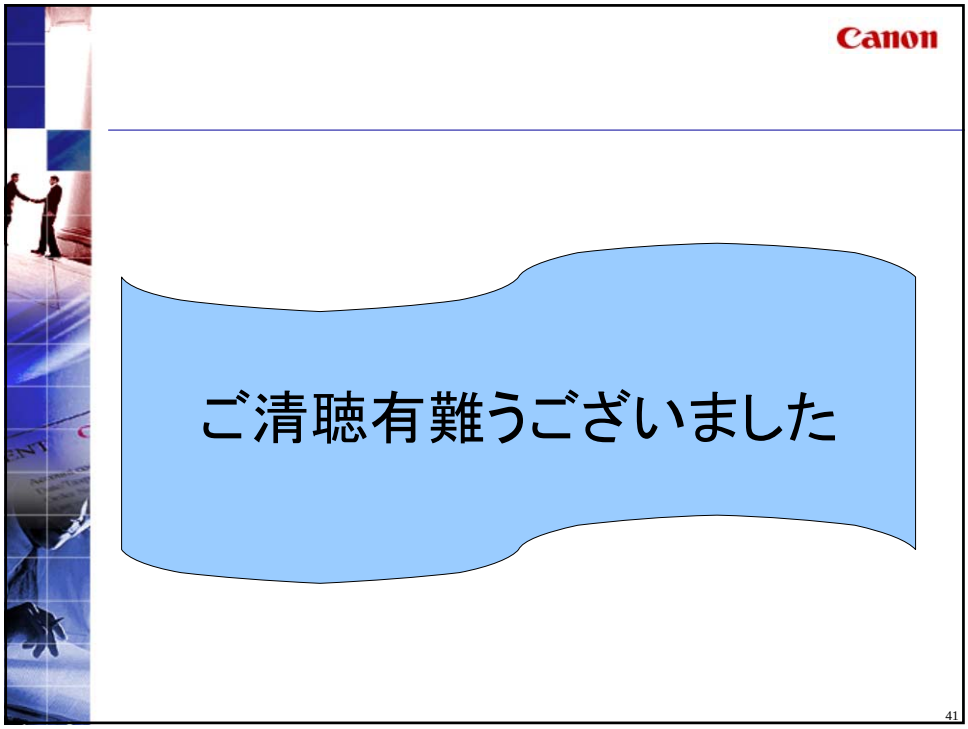


39

マイグレーションを御支援いただき、目標を達成できた事を、NECおよびNECソフト様に、この場を借りてお礼申し上げます。

. . . END

40


Canon
ご清聴有難うございました
41