2003-6-24
COBOLコンソーシアム セミナー

COBOLの2002年規格と今後の展望

ISO/IEC JTC 1/SC 22/WG 4 (COBOL) **日本代表委員**
米国INCITS J4 (COBOL) **日本代表委員**
情報処理学会情報規格調査会
SC 22/COBOL WG **幹事**
プログラム言語COBOL JIS**改正原案作成委員会 委員**

㈱ 日立製作所 ソフトウェア事業部 言語設計部
主任技師 高木 渉

目次

1. COBOLの現状
2. 2002年規格COBOL制定
3. 2002年規格COBOLの主な機能
4. 最先端テクノロジーへの対応
5. 次のCOBOL規格へ

1. COBOLの現状

プログラム言語利用状況

世界中で一番よく使われているのはCOBOL！

	COBOL	Java	C／C++
世界の プログラマ数	200万人 (~300万人)	100万人	100万人
年間の プログラム開発量	50%＊	35%	－

＊：ミッションクリティカルなビジネスアプリケーションの
70%以上がCOBOLで記述

出典：University of LIMERICK, Department of Computer Science and Information SystemsのHP （アイルランドの大学） 2002年4月更新

COBOLの位置付け

米国企業のプログラム言語採用の実態：

・COBOLは，63%の企業で採用されている
・COBOLは，Javaとも併用されている

出典：The Gartner Programming Language Survey 1 Oct 2001

「COBOLは死んだ！」は大きな間違い

「ジャーナリズムはどうしても新しいテクノロジに注目してしまうが，COBOL資産を有効に活用することはとても重要である。」

出典：米国IDCのResearch Director、Dirk Coburn氏
（専門はJava and XML for eBusiness）談

これからのオープン・ソース時代におけるCOBOLの位置付け

Common Business Oriented Language (COBOL) は、1950 年代後半から使用されて いる言語で，ビジネス・アプリケーションにおいて主要な言語になっています。ビジネス・アプリケーションの新規開発で使用される言語を調べてみると，マイクロコンピューターによる開発では、現在 COBOL と Visual Basic が約 35 パーセントを占めています。

出典：http://www-6.ibm.com/jp/developerworks/linux/000929/j_cobol.html
T.W.Burger 2000年5月

COBOLの価値と将来性

**COBOLは，先端IT環境に対応し進化を続けている。
近未来，COBOLに代わるビジネスアプリケーション用言語の
出現は期待できない。**

- ミッションクリティカルなビジネスアプリケーションの70%は，今もCOBOLで記述されている。
- 2004年までの開発･拡張プランの80%では，レガシーシステム（既存資産）の利用･エンハンスが含まれる。
- COBOL誕生以来，1000以上もの様々なプログラム言語が出現したが，現在の使用頻度は少ない。 PL/I ，Pascal，Basic，Ada･･･

出典：The new face of COBOL ACUCORP Seminar Series 2002

ご参考：アメリカの学会誌でも，COBOLの過去･現在･未来を特集
◇IEEE Software，Vol.17，No.2，pp.16-72．2000．

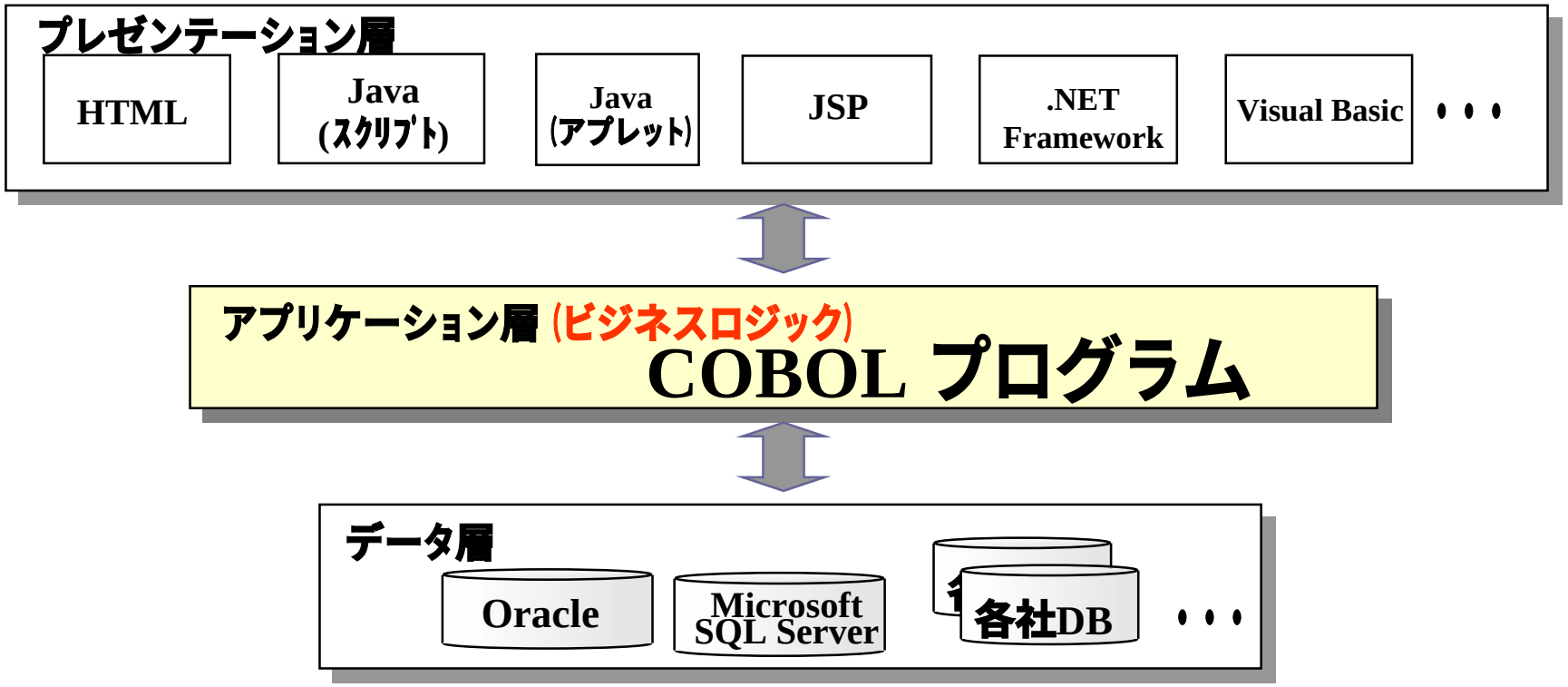
企業ｼｽﾃﾑで「COBOL」が選ばれる理由

- **高い生産性と保守性**
  - 誰が書いても，均質な品質と性能
  - 誰が書いても，他人に読みやすい
- **安定した品質と高性能のプログラムを提供**
  - 長年使われてきた実績のある安定した言語処理系
  - C言語より速い10進演算
- **メインフレーム時代から蓄積されたシステム開発技法の活用**
  - 30年以上に渡るビジネスシステム開発ノウハウを有効活用
  - COBOLプログラマの経験･スキルを活用
- **標準国際規格があり将来性も安心**
  - 30年以上もの間，国際規格が上位互換性を堅持

HITACHI
Inspire the Next

ビジネスロジック構築はCOBOLで

- ユーザインタフェース(プレゼンテーション層)は必要に応じて使い分け
  ビジネスロジックは資産・知識・経験が活かせるCOBOLで構築



ビジネスの根幹である業務ロジックは，急激には変化しない
言語選択のポイントは，長期の保守性／信頼性／互換性

➔COBOLを推奨！

2. 2002年規格COBOL制定

COBOLの歴史： COBOLの誕生

■ 最初のアイデア

- ●英語でのプログラム作成
- ●共通の言語仕様

■ 最初のCOBOL仕様書

- ●CODASYL
  (the Conference on Data Systems Languages)
- ●1960/4 **仕様書発行**(COBOL-60)

COBOL規格の歴史：規格化

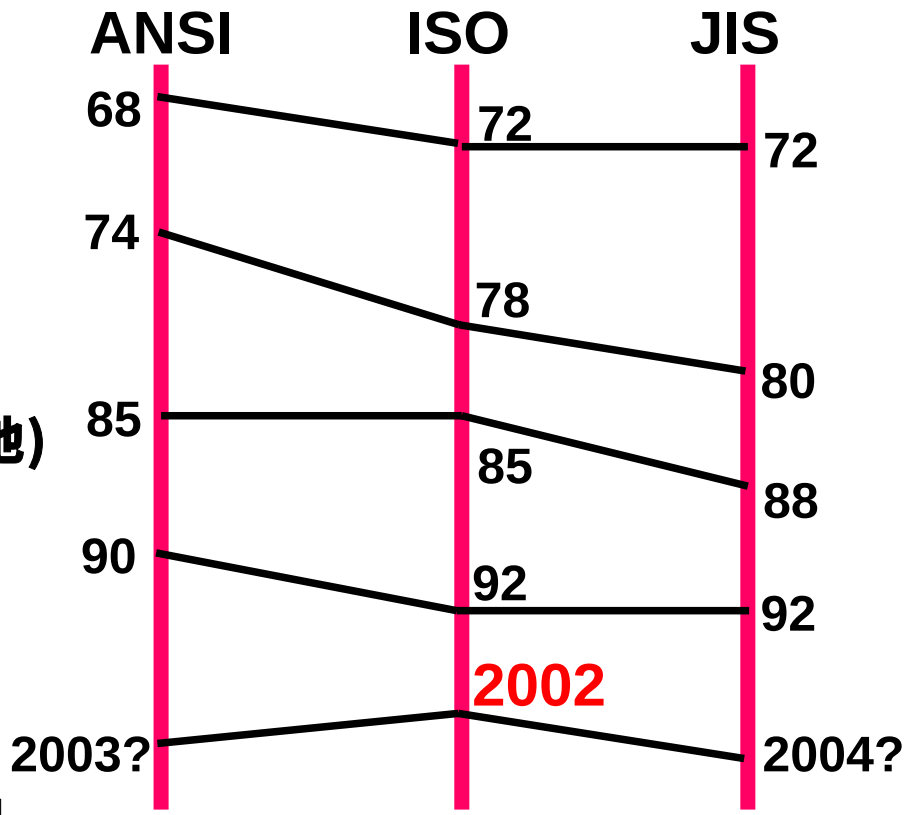
第１次

第２次
大改訂(相対・索引ファイル他)

第３次
大改訂(構造化プログラミング他)

第３.1次
小改訂(組込み関数)

第４次
大改訂
・オブジェクト指向，例外処理，
マルチバイトなど，100項目もの
大小改訂



規格委員会

ISO/IEC JTC 1/SC 22/WG 4 (COBOL) 1回/年

- 国際規格原案の作成・審議
  SC22では他に FORTRAN，C，C++，Ada，LISP，PROLOG，POSIX，言語共通，国際化など
- メンバー：アメリカ，カナダ，日本(ITSCJ)，イギリス，ドイツ，オランダ

(ANSI) INCITS J4 (旧 X3J4) 6回/年

- 国際規格原案の作成（ISOからの委託）
- メンバー：Microfocus(議長), IBM, Unisys, HP, EDS, ITSCJ(日本), 他

CODASYL 4回/年

- 言語仕様の開発と保守
- 1992年X3J4に吸収・合併 (1959〜1992)

情報処理学会 情報規格調査会 IPSJ/ITSCJ SC 22/COBOL WG

- 国際規格の審査
- ISO/ANSI のCOBOL WGへの参加
- メンバー：日立(主査)，奈良先端科学技術大学院大学，沖電気工業，三菱電気，日本IBM，東芝，NHK， NEC， 東京農工大学，NTT，富士通，日本ユニシス，日立ソフトウェアエンジニアリング，

* メンバー欄はそれぞれ順不同

3. 2002年規格COBOLの主な機能

2002年規格COBOLの主な機能

2002年規格COBOLは，先端テクノロジーを吸収しての大改訂

プログラム開発の生産性を向上

1. オブジェクト指向機能
2. 例外処理機能
3. 翻訳指令機能(条件翻訳等)
4. 利用者定義のデータ型機能
5. 利用者定義の関数機能
6. プログラムの再帰呼出し
7. 自由形式の正書法
8. 多オクテット文字(漢字等)処理機能
9. ビット操作機能
10. 浮動小数点データ操作機能
11. ファイルの共用と排他制御の機能
12. データの妥当性検査機能
13. ポインタ項目とアドレス付け機能
14. 数値項目の31桁への拡張

etc

オブジェクト指向機能：特徴

ソフトウェアのモジュール化の新しい手段
既存資産と融合し，無理なく拡張

- オブジェクト指向言語の特徴を実現
  - カプセル化（クラス定義/インタフェース定義）
  - 継承（多重継承を含む）
  - ポリモーフィズム
  - 適合（コンパイル時/実行時における引数の整合性チェックの仕掛け）
  - ガーベジ・コレクション（自動メモリ管理機構）

手続き型とオブジェクト指向の混在（相互に呼び出し）が可能！

オブジェクト指向機能：参照方法

- オブジェクト参照データの定義

```
01 OR-1 USAGE OBJECT REFERENCE クラス名.
```

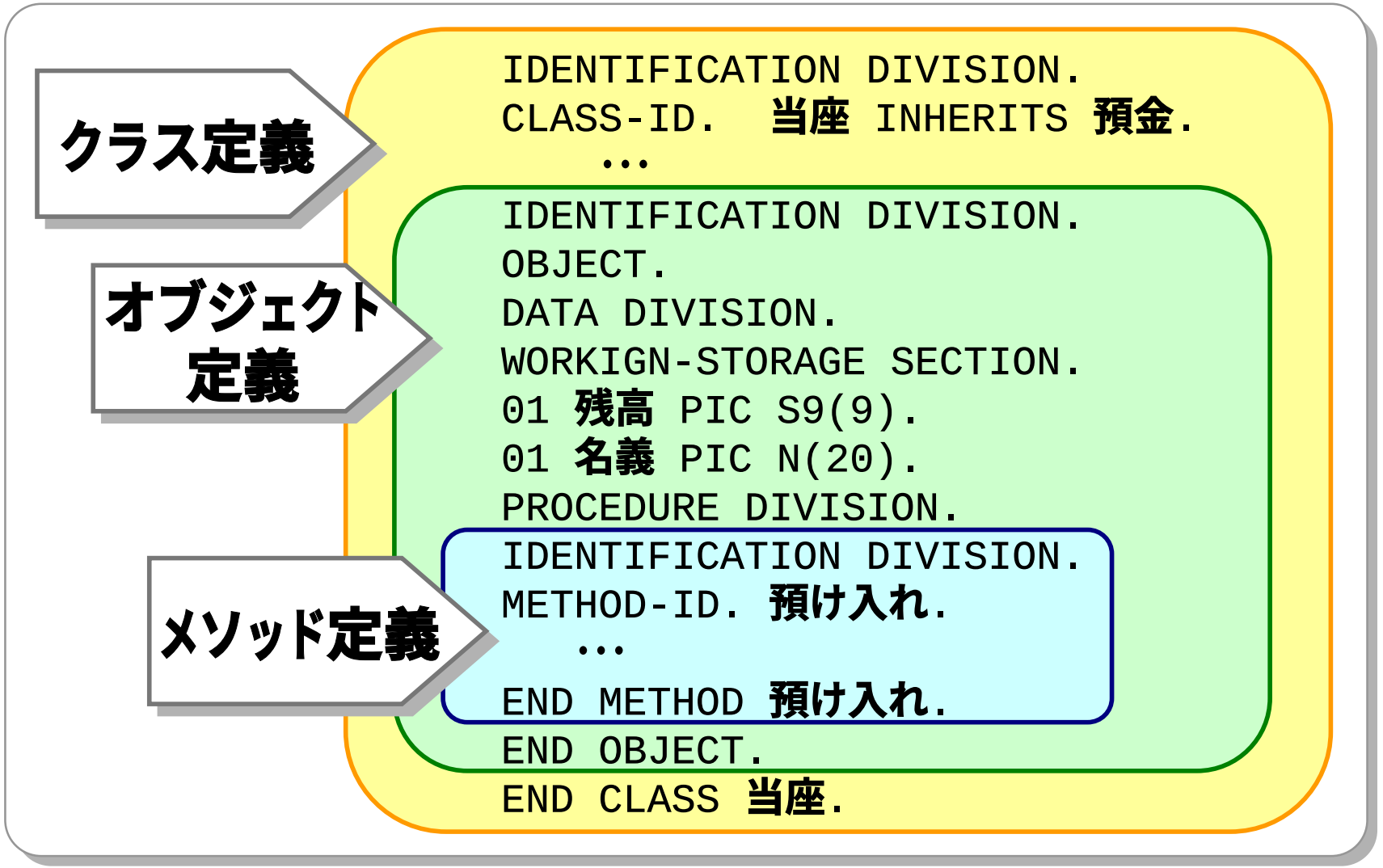
- オブジェクト指向のメソッドを呼ぶ

```
INVOKE OR-1 "メソッド名"
```

- 既存のプログラムを呼ぶ

```
CALL "既存のプログラム名"
```

オブジェクト指向機能：定義方法



例外処理機能

さまざまなエラー発生に対して，手続き部の先頭で記述したエラー処理手続きに制御を移す。

例外処理は，COBOLプログラムでもかなりの部分を占める。
実行時エラーに対する処理をアプリケーションの本質的なロジックから分離することで，保守性（可読性）の高いプログラムが作成できる。

```
01  A  PIC 99 VALUE 50.
01  B  PIC 99 VALUE 60.

PROCEDURE DIVISION.
DECLARATIVES.
 ERROR-HANDLER SECTION.
 USE AFTER EXCEPTION  CONDITION EC-SIZE-TRUNCATION
...
END DECLARATIVES.
>>TURN EC-SIZE-TRUNCATION CHECKING ON

  ADD  A  TO  B        *> システムが例外発生を自動検出
  ...
  IF A  >  60          *>プログラムによって例外を発生させる
     RAISE EXCEPTION  EC-SIZE-TRUNCATION
```

翻訳指令機能: 条件翻訳

利用目的等の違いによるプログラムの差分を一つのソースプログラムで一元管理

```
>>DEFINE TYPE1 AS 1.
IDENTIFICATION  DIVISION.
PROGRAM-ID. PROG-1.
DATA DIVISION.
   ....
PROCEDURE DIVISION.
>>IF TYPE1 IS DEFINED
       *> こちらが取り込まれる
 DISPLAY "DEFINED".
>>ELSE
       *>こちらはコンパイルの対象から除かれる
 DISPLAY "NOT DEFINED".
>>END-IF.
```

翻訳指令機能: その他の機能

旧規格仕様警告，ソースリスト出力指示，例外の伝播，ソース形式(自由/固定書式)，発生する例外の指示，etc

```
>> FLAG85   DIVIDE OFF     (旧規格仕様警告)
>> LISTING   ON           (ソースリスト出力指示)
>> PAGE   次ページの先頭に印字されるコメント
>> PROPAGATE   ON         (例外の伝播)
>> SOURCE FORMAT IS   FIXED
                          (ソース形式  自由書式／固定書式)
>> TURN   EC-RANGE-PERFORM-VARYING   CHECKING ON
                          (発生する例外の指示)
>> IMP   ＜コンパイラオプション等＞
```

利用者定義のデータ型機能

利用者が自由にデータ型を定義･参照することで，同じ構造のデータ項目が簡単に記述可能

```
*>「日付」という利用者定義のデータ型を宣言する
01   日付   IS TYPEDEF.
  05   年      PIC 9(4).
  05   月      PIC 9(2).
  05   日      PIC 9(2).
01   FILLER.
 *>型名「日付」を使用し、「受注日」を宣言する
  05   受注日     TYPE 日付.
 *>型名「日付」を使用し、「納入日」を宣言する
  05   納入日     TYPE 日付.
```

データの抽象度を上げてデータオリエンテッドな開発を可能に！
型チェックにより，型の不整合による不良を早期に発見！

利用者定義の関数機能

**利用者が定義した関数を式中に利用することで，
簡潔で可読性の高いプログラムが作成可能**

＜利用者定義関数＞

戻り値を伴う処理を
関数として
定義可能

```
IDENTIFICATION DIVISION.
FUNCTION-ID.  最大公約数.
DATA DIVISION.
LINKAGE SECTION.
 01  A  PIC S9(9).
 01  B  PIC S9(9).
 01  R  PIC S9(9).
PROCEDURE DIVISION
 USING A B RETURNING R.
...
END FUNCTION  最大公約数.
```

＜関数の参照＞

```
*> 別のプログラムの中で参照
COMPUTE 結果 = FUNCTION 最大公約数(X Y)
```

モジュール化／コンポーネント化を促進！

自由形式の正書法

カラム位置を意識せずに自由な形式でコーディングが可能

- **自由形式正書法**

  [>>SOURCE FORMAT IS FREE]

```
L+++++++++1+++++++++2+++++++++3+++++  +++R
口座入金 SECTION.
  ADD  財布  TO  口座.  *> 貯金が増える
```

- **固定形式正書法** (A/B**領域がなくなる**)

  [>>SOURCE FORMAT IS FIXED]

```
L+++++CP+1+++++++++2+++++++++3+++++  +++R
000130     小遣い計算 SECTION.
          :
000240  MOVE  ZERO  TO  財布.
```

多オクテット文字(漢字等)処理機能

- 変数名とデータに日本語文字が使用可能

```
01   日本語    PIC N(3)  VALUE N"日本語".
```

- 多オクテット文字処理機能関連の組込み関数

```
MOVE FUNCTION NATIONAL-OF(X) TO 結果.
```

ビット操作機能

ビット列とブール演算を用いたプログラミングが可能

- ビット･ブールを扱える機能

```
01  ビット1  PIC 1(5)  VALUE  B"10101".
01  ビット2  PIC 1(5)  VALUE  B"01010".
01  ビット3  PIC 1(5).

COMPUTE ビット3  =  ビット1  B-OR  ビット2
```

- 組込み関数

```
COMPUTE 結果 = FUNCTION INTGER-OF-BOOLEAN(ビット1).
```

浮動小数点データ操作機能

浮動小数点表現を使用した数字データ項目が追加

- USAGE句で3種類の浮動小数点形式を追加

```
01  A  USAGE  IS  FLOAT-SHORT.
01  B  USAGE  IS  FLOAT-LONG.
01  C  USAGE  IS  FLOAT-EXTENDED.
```

- 組込み関数（文字列から浮動小数点値を返す）

```
COMPUTE 結果 = FUNCTION NUMVAL-F(X).
```

ファイルの共用と排他制御の機能

複数プログラムから同一ファイルへの アクセスが可能（データの保全性が向上）

- ファイル共用とレコードロックの指定

```
FILE-CONTROL.  *>共用モードで、自動的にレコードロック
  SELECT  USER-FILE  ASSIGN  TO SYS001
     ORGANIZATION  IS  SEQUENTIAL
     SHARING  WITH  ALL  OTHER
     LOCK  MODE  IS  AUTOMATIC
     WITH  LOCK  ON  RECORD.
```

- アクセス拒絶時の再試行回数指定

```
READ  USER-FILE  RETRY FOREVER
```

データの妥当性検査機能

VALIDATE命令で，データオリエンテッドなアプローチによる生産性･可読性･保守性の向上

```
01 入力日付.
  05  年-1 PIC 9(4)    DESTINATION IS  年-2.
    88   VALID  VALUE 1990 THRU 1999.
  05  月-1 PIC 9(2)    DESTINATION IS  月-2.
    88   VALID  VALUE    1 THRU 12.
  05  日-1 PIC 9(2)    DESTINATION IS  日-2.
    88   VALID  VALUE    1 THRU 31.
01 出力日付.
  05 年-2  PIC  9(4).
  05 月-2  PIC  9(2).
  05 日-2  PIC  9(2).

  MOVE  FUNCTION CURRENT-DATE  TO  入力日付.
  VALIDATE  入力日付.
  DISPLAY 出力日付.
```

＜VALIDATE命令の処理イメージ＞

IF 年-1 >= 1990 And <= 1999
then MOVE 年-1 TO 年-2
else INITIALIZE 年-2

ポインタ項目とアドレス付け機能

ポインタという新しいデータ･クラスを導入

- データアドレスとデータポインタ機能

```
01 P USAGE POINTER.
01 MY-REC BASED .
  02 NAME PIC X(30).
  02 ADDR PIC X(30).
      ....
SET ADDRESS OF MY-REC TO P.
```

C言語との連携を意識した機能

数値項目の31桁への拡張

数字項目の桁数不足を解消

```
01   VERY-HUGE-NUMBER   PIC 9(31).
```

■ 経済の拡大により取扱い桁数の増加
　　⇒ 19 桁以上必要

■ SQL3 との連携
　　(SQL3は，既に31桁数字項目をサポート)

4. 最先端テクノロジーへの対応

規格を越えてテクノロジーへ対応

各COBOLベンダーが最新テクノロジーに対応

＝＞膨大な既存COBOL資産を有効活用

Webテクノロジー対応の紹介 (日立COBOL2002の場合)

☆ Java-COBOL連携

- Webアプリケーションの中核，ビジネスロジックをCOBOLで作成。
- Java™アプリケーションから，COBOLプログラムを JavaBeans あるいはEJBとして呼出す。

☆ COBOL-XML連携

- XMLデータをレコードと同様に扱う。
- データ交換用XMLデータを扱うアプリケーションもCOBOLで作成。

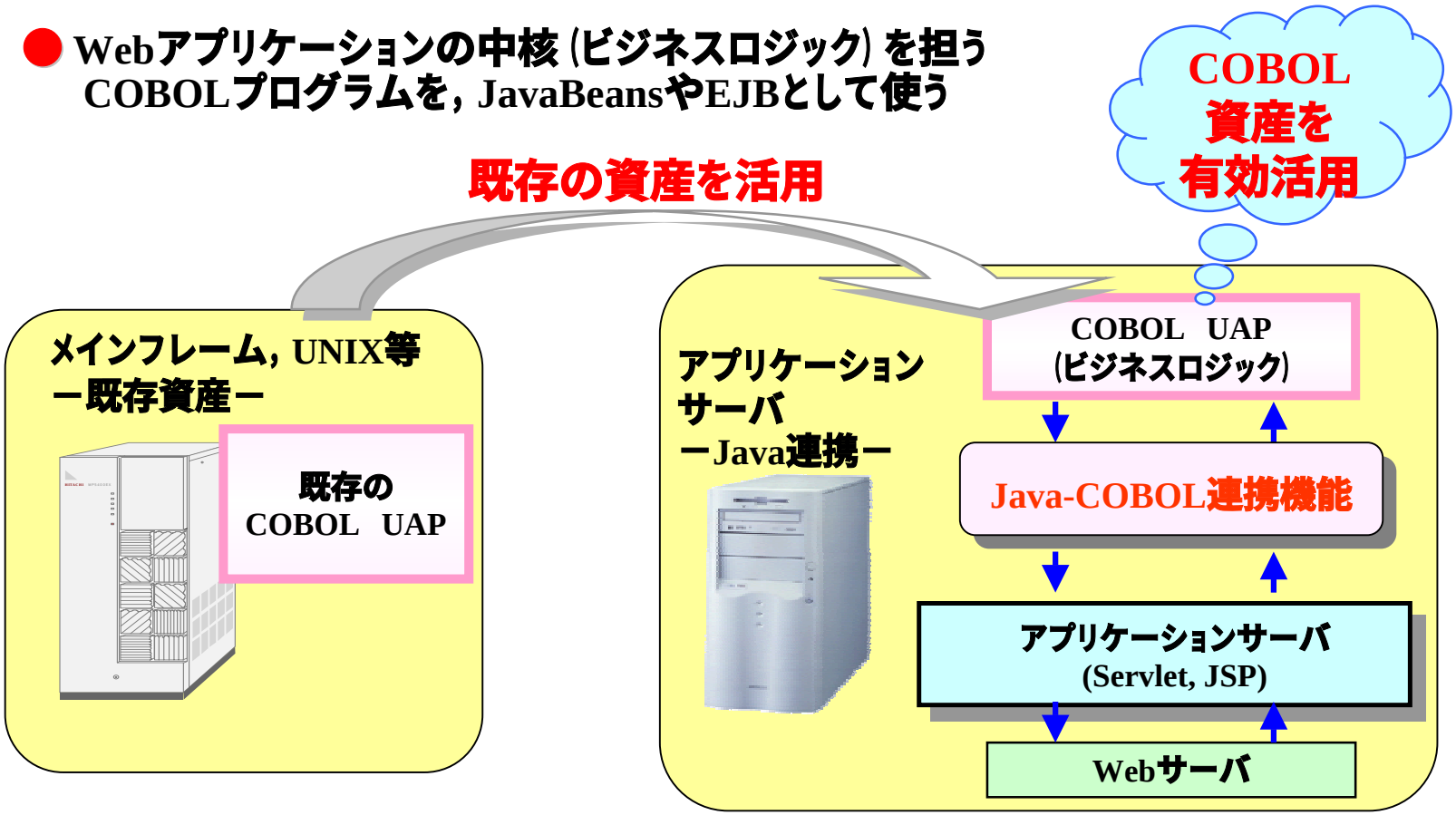
☆ COBOL-SOAP連携

- SOAPメッセージを利用した「Webサービス」のインタフェース

Java-COBOL連携機能: ねらい

Javaベースのアプリケーションサーバ環境で Javaアプリケーション（Servlet，JSP）から COBOLプログラムを呼び出すための開発／実行環境

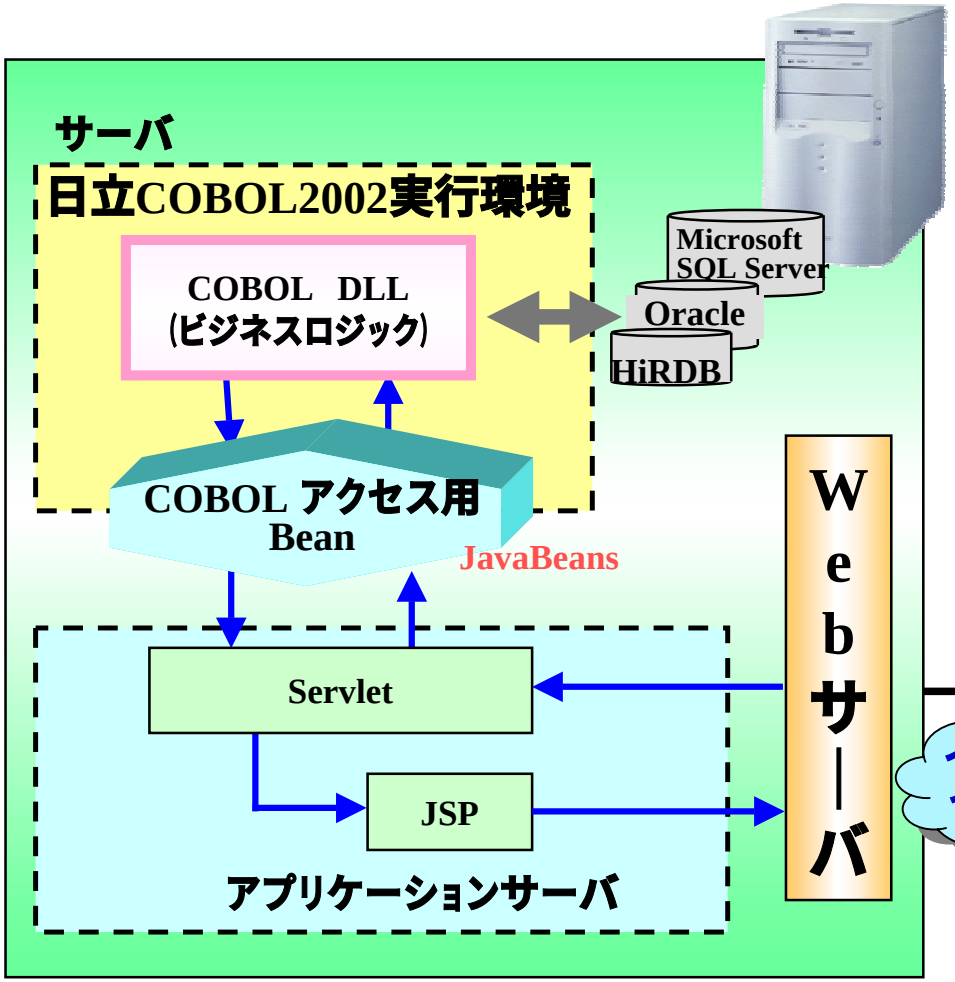
- Webアプリケーションの中核（ビジネスロジック）を担う COBOLプログラムを，JavaBeansやEJBとして使う



JSP：Java Srver Pages　EJB：Enterprise JavaBeans

Java-COBOL連携機能: 仕掛け

- Javaベースのアプリケーションサーバの採用と既存COBOL ソースの流用
  → メインフレーム集中型の検索システム等をWeb環境(イントラネット) へ迅速に移行



- Javaアプリケーションは，COBOLアクセス用Beanを呼び出すだけでCOBOLアプリを実行。
- COBOLアプリケーションは，ただのサブルーチン。
- COBOLアクセス用Beanは，ウィザードを使って自動的に作成。

COBOL・XML連携機能

- XMLデータをCOBOLのレコードとして入出力。
- 既存COBOL資産を活かして，e-ビジネス向けのデータ交換用XMLデータを扱うアプリケーションを作成。

XMLデータ

```
<注文伝票>
  <商品>
    <商品名>洗顔石鹸</商品名>
    <注文数> 10</注文数>
  </商品>
  <商品>
    <商品名>シャンプー</商品名>
    <注文数>  5</注文数>
  </商品>
  <商品>
    <商品名>リンス</商品名>
    <注文数>  5</注文数>
  </商品>
</注文伝票>
```

COBOLアプリケーション

COBOLレコード：注文伝票

商品	
商品名	注文数
洗顔石鹸	10
シャンプー	5
リンス	5

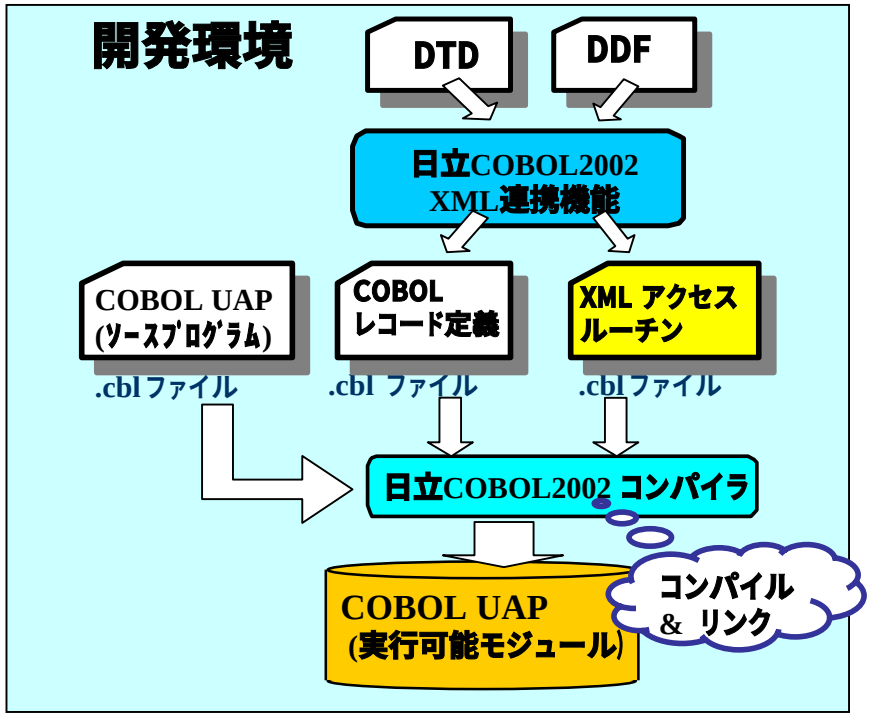
DTD(文書定義ファイル)

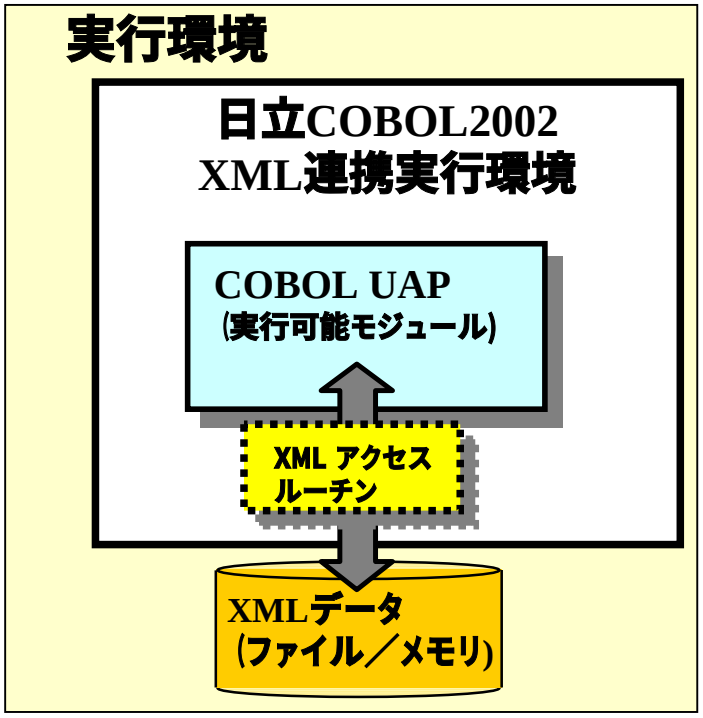
```
<!DOCTYPE 注文伝票 [
<!ELEMENT 注文伝票 (商品)*>
<!ELEMENT 商品 (商品名，注文数
) >
<!ELEMENT 商品名 (#PCDATA)>
<!ELEMENT 注文数 (#PCDATA)>
  ]>
```

COBOLレコード定義

```
01 注文伝票.
  05 商品 OCCURS 10.
    10 商品名 PIC X(20).
    10 注文数 PIC 9(6).
```

COBOL-XML連携の開発環境/実行環境





XMLデータ処理アプリケーションの開発手順

①XMLに対応づけるCOBOLレコードの構造やデータの型・サイズを指定するDDF(データ定義ファイル) を作成

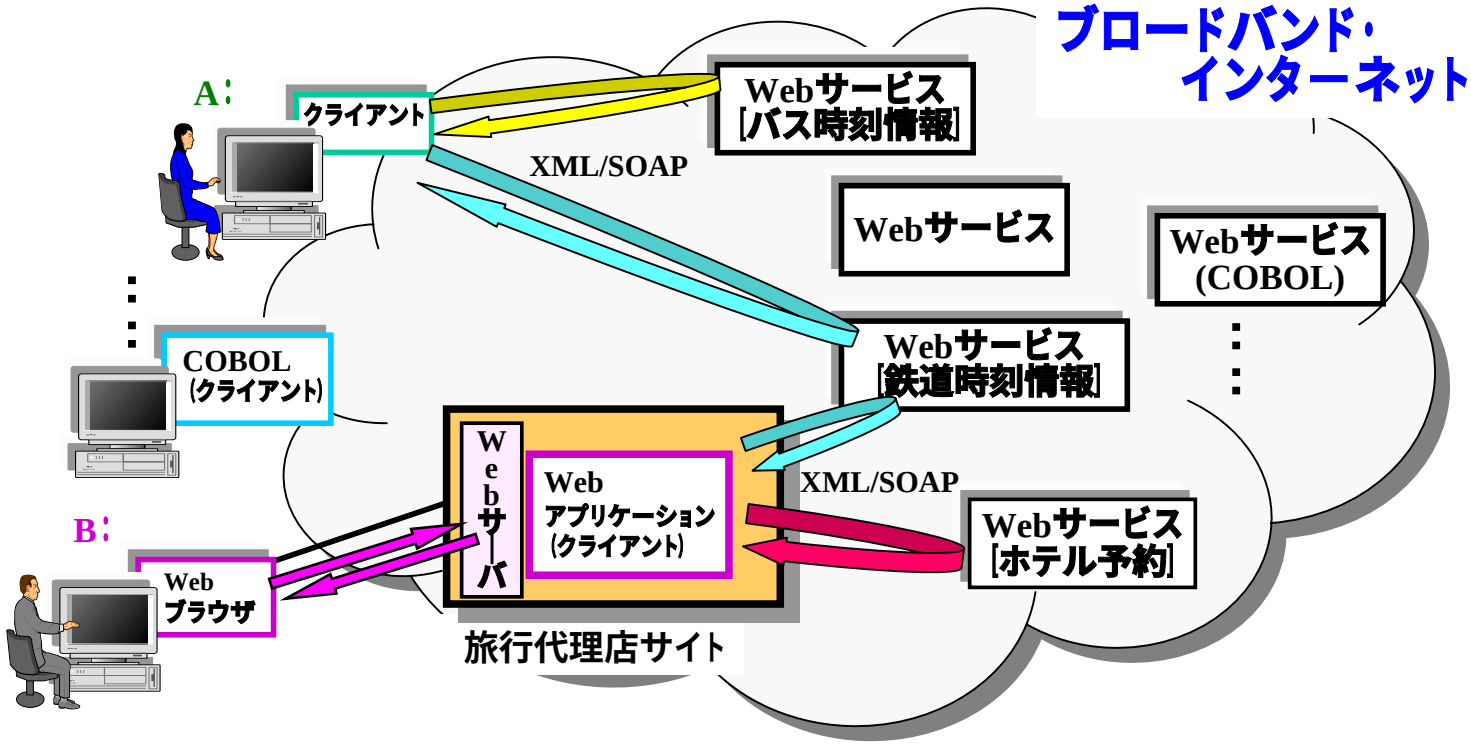
②COBOL2002 XML連携機能で，XMLアクセス用データ定義（COPY登録集原文）とXMLアクセスルーチン（COBOL副プログラム）を生成

③XMLアクセス用データ定義を取り込み，XMLアクセスルーチンを呼出してXMLデータを処理するCOBOLユーザソースプログラムを作成

④COBOLユーザソースプログラム，XMLアクセスルーチンをあわせてコンパイル／リンク

Webサービスについて

- Webサービスは，XMLベースのSOAPプロトコルで
  インターネット上のアプリケーション間の相互運用を実現
- 各アプリケーション（Webサービス）の統合サービスを提供



SOAPメッセージ

Webサービスのサーバとクライアントは，SOAP メッセージで通信
SOAPメッセージはXML形式のデータ

SOAPメッセージのフォーマット

SOAPメッセージの 具体例:

Webサービス「Sum_4num」へのリクエストメッセージ

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<soap:Header>・・・・</soap:Header>

<soap:Body>
<Sum_4num  xmlns="http://tempuri.org/">
<a>1</a><b>2</b><c>3</c><d>4</d>
</Sum_4num>
</soap:Body>

</soap:Envelope>
```

4個のデータをサーバに送る

Webサービス「Sum_4num」からのレスポンスメッセージ

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<Sum_4numResponse  xmlns="http://tempuri.org/">
<Sum_4numResult>10</Sum_4numResult>
</Sum_4numResponse>
</soap:Body>
</soap:Envelope>
```

結果データ‘10’をサーバから送る

HITACHI
Inspire the Next

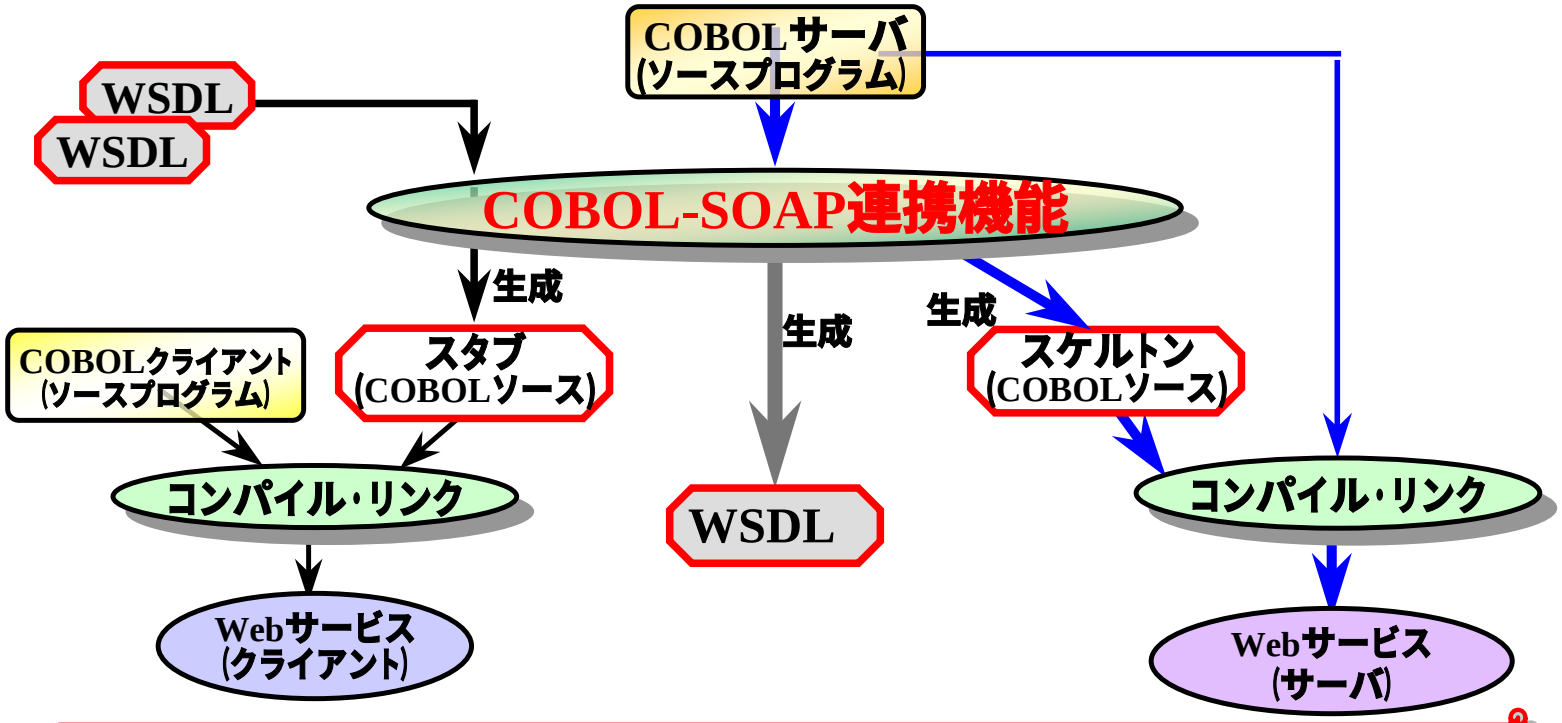
COBOL･SOAP連携の開発環境

■COBOLサーバプログラム（既存／新規）から，スケルトンおよびWSDLを自動生成

スケルトン：クライアントから渡されたSOAPメッセージを解析し，COBOLプログラムで処理可能なように引数のマーシャリング／アンマーシャリングを行なう。

■生成されたWSDLあるいは公開されているWebサービスのWSDLからスタブを生成

スタブ：Webサービスをアクセスするためにクライアント COBOLプログラムのCALL文の引数情報からSOAPメッセージを作成する。



●**WSDL**(Web Services Description Language)
XMLベースのインタフェース情報記述である。WSDLをUDDIディレクトリに登録することで，COBOLサーバをWebサービスとして公開できる。

●**UDDI**(Universal Description, Discovery and Integration)
WSDLデータのレジストリであり，Webサービスのディレクトリ／カタログに相当する情報サービスを提供するブローカである。

5. 次のCOBOL規格へ

規格委員会による機能募集

- COBOL Workshop **開催**
  - 2003年6月25日 ラスベガス
  - **事前に，次の規格に欲しい機能を募集**
  - **応募者が機能をプレゼンテーション**
- SC 22/WG 4 (COBOL) **開催**
  - 2003年6月26～28日 ラスベガス
  - COBOL Workshop**の聴衆の反応を得て，規格委員会で今後の規格化を判断**

提案されている機能

1. COBOL構文組込みのXML機能
2. 局所的例外処理
3. 構造化定数
4. 動的表(配列)
5. XMLクラス
6. コレクションクラスライブラリ
7. 非同期処理

今後のCOBOL

- Web関係テクノロジの規格への取り込み
  - XML, Webサービス等
- 引き続き，主たるビジネスロジック記述言語
  - 膨大な資産
  - 比肩できる，安定した言語の欠如
  - 最新技術の取り込み

→ COBOLは，COBOLらしく