Langages et BD GCOS7

Jacques PRINTZ
Professeur au CNAM

FEDERATION DES EQUIPES BULL

Le contexte des langages dans les années 70s

- Émergence du concept de langage de « haut niveau »
  - FORTRAN, COBOL
  - ALGOL (langage de spécification des algorithmes), puis PL1 – Naissance de la programmation structurée chez IBM
  - Beaucoup de SSII développent des compilateurs
- La majorité des programmeurs sont dans une culture assembleur
  - Macro-assembleur comme METASYMBOL, initialement développé chez SDS puis repris à la CII
- **Le pari du MIT** : développer un OS (i.e. MULTICS) en PL1 (avec le PL1 MULTICS « bootstrapé »)
- Très grosse activité de recherche sur les langages et les compilateurs (analyse syntaxique, optimisation, etc.)

Les langages de la NPL-L64 de Honeywell-Bull

- Langages d'implémentation de l'OS
  - NAL Assembleur strictement **réservé aux équipes Bull**
  - HPL Sous ensemble strict de PL1
  - MACPROC Macro-générateur initialement développé sur MULTICS, puis migré sur 4A Backup et GCOS64
  - MLP Macro-assembleur pour développer l'émulateur G100
- Langages commerciaux
  - COBOL Le **langage principal** de la ligne de produit, strictement conforme au standard 74
  - RPG Langage très populaire pour les petits systèmes (NB : le langage principal du IBM S38)
  - FORTRAN et PL1 ; BASIC et APL (avec une micro-machine « d'accélération » en firmware qui ne sera jamais livrée)

Stratégie d'implémentation des compilateurs

- **Standardisation** des éléments essentiels de l'architecture des compilateurs :
  - ***CU format*** standard, **identique** qqs le langage
  - ***Stack & Parameter Passing*** standard
    - Format de pile identique qqs le langage + exploitation optimale de l'« *Interior Decor* » (segments, registres, etc.)
- HPL et COBOL (+ le tri-fusion) initialement développés à Boston
  - Rapatriés à Paris en 76-77
- NAL, MLP, FORTRAN, RPG développés à Paris
- Exploiter au mieux le jeu d'instructions très riche de la machine

FEDERATION DES EQUIPES BULL

Technologie de développement des compilateurs

- 1ère étape : chaque équipe se débrouille
  - Outillage minimal avec un outil d'analyse syntaxique RLS (basé sur Floyd-Evans)
  - Un générateur de code par compilateur
- 2ème étape : tous les compilateurs à Paris
  - Développement d'une « *Compiler Factory* » avec 2 langages : SDL et META et une méthodologie de test fondée sur les graphes de « couvertures »
  - Développement d'une architecture commune permettant de factoriser les générateurs de code et l'optimiseur : OIL et CCG
- Transfert de technologie réussi Recherche (Surtout MIT, un peu INRIA) → Industrie

FEDERATION DES EQUIPES BULL

Des fichiers séquentiels aux bases de données

- Au début des années 70s, le monde des fichiers est encore fondamentalement séquentiel (BFAS et ISAM)
  - Rubans et cartes perforées pour les mises à jour, bandes magnétiques pour le stockage de masse
  - Mémoire « rapide » sur tambour (*les 1er MULTICS ont leur mémoire virtuelle sur tambour*), puis avènement des disques (*à secteurs variables, puis à secteurs fixes*)
- Les disques vont permettre le développement des fichiers indexés (UFAS alias VSAM d'IBM, MLDS d'origine L62 et GE58), puis des bases de données → 2 index définissent une relation
  - Aide hardware DPS7 : instruction HASH (*construit un nombre aléatoire sur 32 bits à partir d'une chaîne de caractères de longueur quelconque*)
  - IDS2 est fondamentalement une gestion de pointeurs et d'index (*i.e. les DB-Key*), à partir desquels le programmeur peut « naviguer » dans la base vue comme un ensemble de pages

FEDERATION DES EQUIPES BULL

Bases de données et langages BD

- Une personnalité de classe internationale, Charles BACHMAN (ACM **Turing award**), créateur du modèle navigationnel CODASYL/IDS et contributeur majeur du modèle ANSI/SPARC
  - Les 3 niveaux de schémas
  - Standardisation ANSI, ISO, + relais AFNOR
  - À l'AFNOR, 3 modèles en compétition
    - CODASYL (Honeywell-Bull)
    - SOCRATE (INRIA-CII)
    - Relationnel (via IBM France)
- Démarrage des travaux IDS2 à Boston, puis rapatriement à Paris en 76-77
  - Avec un architecte de première force, R.Fayot

FEDERATION DES EQUIPES BULL

Info-centre, L4G et relationnel

- DML implémenté comme une extension standard du COBOL 74
  - Conforme à la norme ISO NDL
  - Interfaçage intégré avec le moniteur TP TDS via les ordres COBOL ACCEPT et DISPLAY
- Concept Info-centre très à la mode + forte demande client
  - Choix du langage IQS (sur la base des travaux faits à la CII autour du SGBD SOCRATE qui deviendra CLIO) sous la direction de J.J.Laclaverie
    - Interfacé avec UFAS, IDS2 et le moniteur TP TDS
- 1er essai du relationnel en 78-79
  - Une implémentation sur MULTICS (MRDS)
- Standardisation simultanée des deux standards NDL et SQL
  - Début de la vague de fond des SGBD relationnels

FEDERATION DES EQUIPES BULL

IDS2 vs modèle relationnel pour les OLTP « lourds »

- Débuts industriels **laborieux** du relationnel
  - Scepticisme sur la capacité du modèle à supporter des charges transactionnelles lourdes + Culture d'entreprise
    - Stratégie de portage d'une souche SQL + contact avec les californiens ORACLE et INGRES dès 1981
      - Développement d'un compilateur C pour permettre le portage ORACLE + Interfaces avec GCOS7 (≈ 30 KLS)
    - **ORACLE est un modèle d'architecture conçue portable,** qui rentre sans difficulté dans l'architecure GCOS7
- Très **grand succès académique**, à mettre au crédit de Ted Codd (un autre ACM **Turing award**), d'IBM
  - Recherches intenses sur les caches et le transactionnel (surtout aux US) dans l'équipe du labo IBM San José, ainsi que l'optimisation des requêtes SQL (langage QUEL sur INGRES, à Berkeley)
  - Tous les étudiants apprennent les bases de données avec SQL à l'université

FEDERATION DES EQUIPES BULL

Passé, présent et futur

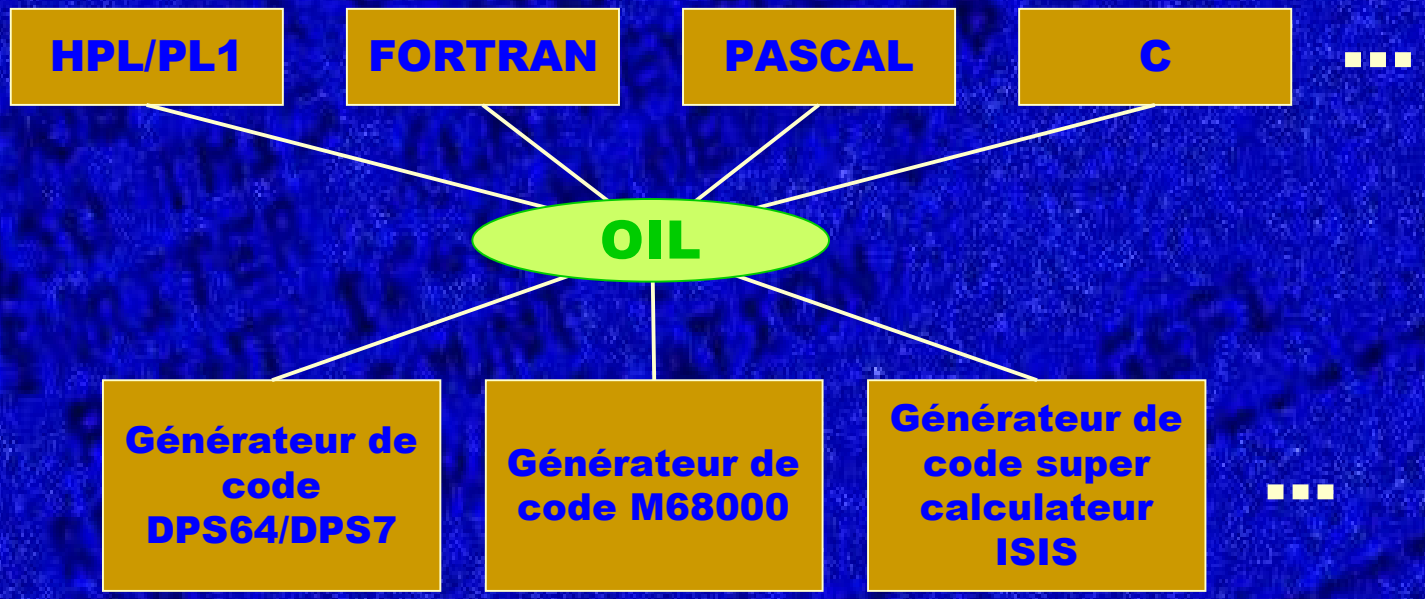
- Tous les compilateurs sont aujourd'hui dans des architectures du type OIL-CCG
  - Cf. la « *compiler factory* » de R.Stallman de GNU
  - Test grandeur nature de la solidité de l'architecture OIL-CCG pour le FORTRAN ISIS
  - Le destin de Ada (*1ère implémentation sur DPS7, pour le bootstrap*) aurait-il été différent si une stratégie OPEN SOURCE avait été jouée ?
    - Cf. le compilateur Ada GNAT
- La rigueur logique du relationnel a permis de mettre en place une architecture de cache très performante tant au niveau des I/O que des requêtes, mais les très grands objets restent un problème
- Avec les modèles objets + BDO et les IHM, le concept de navigation est redevenu à la mode
- Construction des applications clients/serveurs par transformation de modèles (cf. la notion de « *pattern* », en fait très ancienne !!!)
  - Concepts de « *factory* » analogues à ceux de la naissance de GCOS et de la 1ère Factory sur MULTICS

FEDERATION DES EQUIPES BULL

Annexes

FEDERATION DES EQUIPES BULL

Architecture OIL-CCG



Architecture du CCG