Algorithme et programmation

Mathématiques 2nde
Module No 24

Démarche de réalisation: Introduction à l'algorithmique, Langages et bases de pseudo-code

Programme

- **Mod #4 Démarche de réalisation: Introduction à l'algorithmique et bases de pseudo-code**
  - L'ordinateur, une machine programmable
    - Concept de langage : langage assertionnel vs langage procédural
    - Approche synthétique vs approche algorithmique
    - Programmation et programmatique
    - Générations de langage
    - Passer du langage symbolique au langage binaire : assemblage et édition de liens
    - Passer du langage évolué au langage binaire : compilation et interprétation
    - Panorama des langages
  - Les étapes de conception d'un logiciel
  - Un exemple : un programme de paie très simplifié
    - Analyse du problème
    - Solution globale : Processus et données
    - Formalisation du pseudo-code
    - Ecriture du programme
    - Test et validation
  - Programmatique

Quelques questions

Une machine programmable

- En plus de ses possibilités de Collecte, de Stockage, de Traitement et de Communication, l'ordinateur est une **machine programmable.**

- Ses possibilités de **Programmation**, c'est à dire d'adaptation à un problème nouveau, permettent de développer des **logiciels** spécifiques capables de satisfaire aux besoins exprimés par les utilisateurs.

- Ils permettent aussi, lorsque le marché est suffisamment vaste pour justifier un tel investissement, à des sociétés spécialisées de développer des solutions standards : les **progiciels**.

Langage procédural et langage assertionnel

- Un Langage est un ensemble de signes formant un système, destiné à l'expression et à la communication.
- Notion de **langage procédural** :
  - Prenez la première à droite,
  - Tournez dans la seconde rue à gauche,
  - Merci d'accélérer.
- Notion de **langage assertionnel** :
  - Déposez-moi à l'aviation avant 17 h.
- Un langage de programmation (ou langage informatique) est donc un ensemble de signes permettant à celui qui veut écrire un programme, d'exprimer et de formaliser son besoin pour le communiquer aux circuits de l'ordinateur de manière compréhensible par ceux-ci.
- Les langages informatiques sont le plus souvent de type procédural.

Approche synthétique vs démarche algorithmique

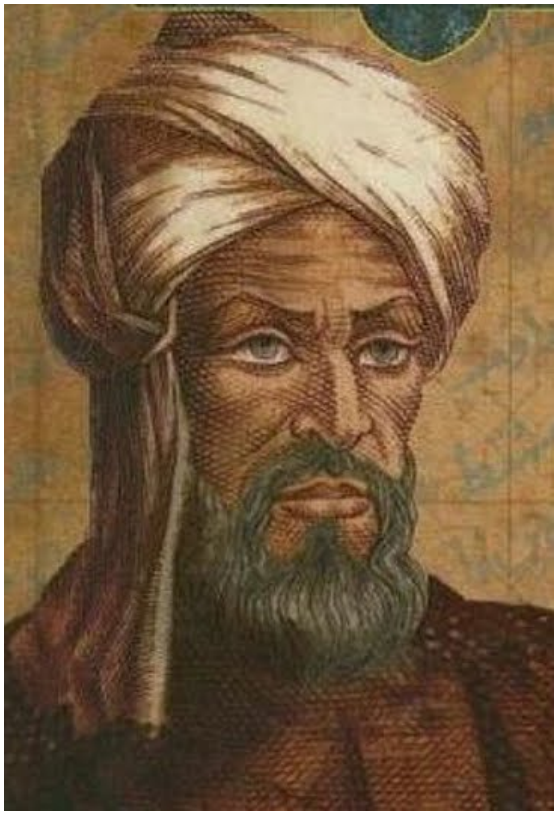
- Soit 5 noms

ALPHA	Albert
ZOULOU	Zoe
BRAVO	Brigitte
MIKE	Michel
FOX	Fernand

- Vous devez les trier par ordre alphabétique
- C'est quasi immédiat en raison du faible nombre d'individus : ALPHA, BRAVO, FOX, MIKE, ZOULOU (approche synthétique)
- C'est beaucoup plus difficile avec 100 noms
- Il faut alors disposer d'une démarche systématique : l'**algorithmique**

Algorithmique

- Un **algorithme** est une suite finie et non-ambiguë d'opérations ou d'instructions permettant de résoudre un problème.
- Le mot algorithme vient du nom latinisé du mathématicien perse Al-Khawarizmi, surnommé « le père de l'algèbre ».
- Le domaine qui étudie les algorithmes est appelé l'algorithmique.
- **L'algorithmique** est à la base de la conception des logiciels lorsqu'on utilise un langage procédural puisqu'elle permet de formaliser le processus logique (les étapes du raisonnement) permettant la résolution du problème posé.



Un algorithme pour résoudre une équation du second degré

- Soit l'équation $ax^2 + bx + c = 0$ dans l'ensemble des réels

- Si a= 0 alors l'équation est du premier degré et on obtient la solution directe : x=-c/b (avec $\neq 0$) .
- Si a $\neq$ 0 :
  - Calculer le discriminant : delta=b*b-4*a*c.
  - Si le discriminant est strictement positif, calculer deux solutions :
    - x1=(-b+racine (delta))/2*a
    - x2=(-b-racine (delta))/2*a
  - S'il est nul, calcul d'une solution double :
    - x=-b/2a
  - S'il est strictement négatif, pas de solution dans l'ensemble des réels.

Programmation et programmatique

- **Programmation**
  - Traduction d'un problème dans un langage compréhensible par l'ordinateur.
- **Programmatique**
  - Méthode pour concevoir un bon programme
- La plupart des langages informatiques sont **procéduraux**. Ils imposent de décomposer la solution au problème posé sous forme d'un **algorithme**.
- Seuls quelques langages de 4ème génération, intégrant les technologies de l'intelligence artificielle, sont **assertionnels**.
- SQL, langage d'interrogation des bases de données est assertionnel.

Générations de langage

PREMIERE GENERATION

- **Langage de Base** : Le langage binaire que les circuits de l'ordinateur peut décoder

SECONDE GENERATION

- **Langage Symbolique** : S'affranchir des contraintes du codage en binaire

TROISIEME GENERATION

- **Langage Evolué** : S'affranchir des contraintes de la spécificité du répertoire d'instructions

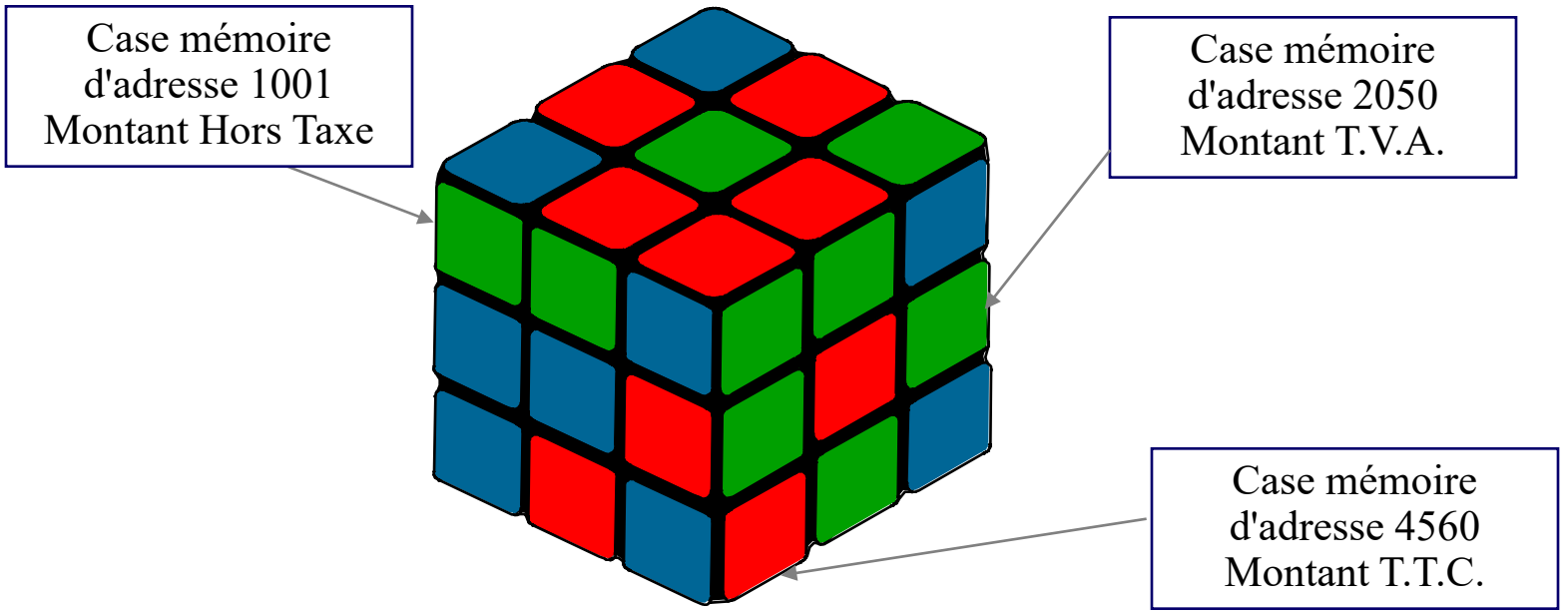
QUATRIEME GENERATION

- **Langage "de quatrième génération":** Se rapprocher du langage naturel

Une typologie commode, mais qui ne signifie pas que seuls les langages de 4 ème génération sont utilisés aujourd'hui !

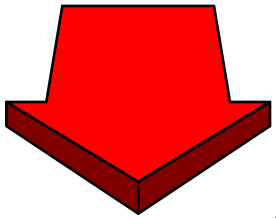
Principes de la programmation

- Nous nous proposons de rappeler les principes des langages de programmation en tentant de résoudre un problème très simple
- Effectuons la somme d'un opérande stocké à l'adresse 1001 (le montant Hors Taxe) avec un opérande stocké à l'adresse 2050 (le montant de la T.V.A.).
- Le résultat calculé (le montant T.T.C.) doit être stocké à l'adresse 4560.



Première génération (1G)

- Chaque ligne du programme correspond à une instruction du **répertoire des instructions de la machine** et est codée directement dans le langage binaire interprétable par l'ordinateur.



Valeur binaire du code opération Charger

Valeur binaire du No de registre 2

Valeur binaire de l'adresse 1001

			Que Faire ? Code Opération	Avec quoi le faire ? Opérande No 1	Avec quoi le faire ? Opérande No 2
0 0 1 0	0 1 0	0001111101001	Charger	le Registre No 2 de l'U.C.	avec le mot d'adresse 1001
0 0 1 0	0 1 1	0100000000010	Charger	le Registre No 3 de l'U.C.	avec le mot d'adresse 2050
1 0 1 1	0 1 0	011	Additionner	le Registre No 2 de l'U.C.	avec le Registre No 3 de l'U.C.
0 0 1 1	0 1 0	1000111010000	Ranger	le Registre No 2 de l'U.C.	dans le mot d'adresse 4560

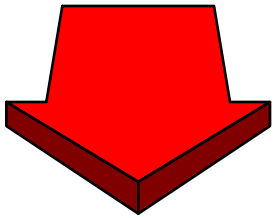
1G = langage machine

- Le programme 1G
  - 00100100001111101001
  - 00100110100000000010
  - 10110100000000000011
  - 00110101000111010000

- Seul l'ordinateur le comprend
- Il peut ne comprendre que ce langage
- Il se programmait avec les clefs du panneau de commande
- Il se lisait avec les voyants du panneau de commande

Deuxième génération

- A chaque élément de l'instruction est associée un équivalent symbolique, ce qui en facilite l'écriture, mais à chaque ligne du programme correspond toujours une et une seule instruction du répertoire.



Code symbolique de l' opération Charger

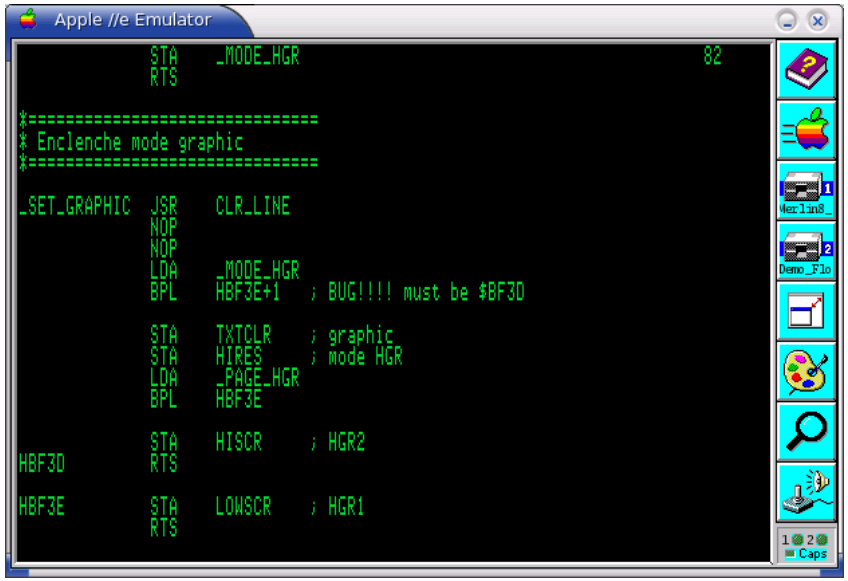
Code symbolique du registre 2

Etiquette symbolique associée à l'adresse 1001

				Que Faire ? Code Opération	Avec quoi le faire ? Opérande No 1	Avec quoi le faire ? Opérande No 2
LOAD	R2	HorsTaxe	←	Charger	le Registre No 2 de l'U.C.	avec le mot d'adresse 1001
LOAD	R3	TVA	←	Charger	le Registre No 3 de l'U.C.	avec le mot d'adresse 2050
ADD	R2	R3	←	Additionner	le Registre No 2 de l'U.C.	avec le Registre No 3 de l'U.C.
STORE	R2	TTC	←	Ranger	le Registre No 2 de l'U.C.	dans le mot d'adresse 4560

2G = Langage assembleur

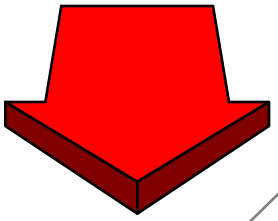
- Le programme 2G
  - LOAD R2, HorsTaxe
  - LOAD R3, TVA
  - ADD R2, R3
  - STORE R2, TTC



- Comme l'ordinateur ne comprend que le langage 1G, ce programme doit être traduit.
- Ce traducteur est un programme **Assembleur**
- Un langage 2G Assembleur est spécifique d'une machine donnée.

Troisième génération (3G)

- Le principe des étiquettes symboliques est conservé, et chaque ligne du programme engendre plusieurs instructions du répertoire.



Code symbolique d'une instruction regroupant l'ensemble des fonctions de chargement, de calcul et de stockage du résultat obtenu

ADD HorsTaxe **TO** TVA **GIVING** TTC

Etiquette symbolique associée à l'adresse 4560

Etiquette symbolique associée à l'adresse 2050

Etiquette symbolique associée à l'adresse 1001

Que Faire ? Code Opération	Avec quoi le faire ? Opérande No 1	Avec quoi le faire ? Opérande No 2
Charger	le Registre No 2 de l'U.C.	avec le mot d'adresse 1001
Charger	le Registre No 3 de l'U.C.	avec le mot d'adresse 2050
Additionner	le Registre No 2 de l'U.C.	avec le Registre No 3 de l'U.C.
Ranger	le Registre No 2 de l'U.C.	dans le mot d'adresse 4560

3G = Langage symbolique

- Le programme 3G
  - ADD HorsTaxe TO TVA
    GIVING TTC

```
SCOMSTO.M214.LIBRARY(MSGTSTCO) - 01.07          Columns 00001
> ____________________________________________ Scroll ===>
  do-the-work.
      move in-part-number        to reprt-part-number
      move in-description        to reprt-description
      move in-quantity-on-hand   to reprt-quantity-on-hand
      move in-quantity-on-ord    to reprt-quantity-on-ord
      move in-unit-price         to reprt-unit-price
      move in-reorder-level      to reprt-reorder-level
      write reprt-rec from reprt-record
      perform read-a-record.


  print-table.
      move parts-no(part-index)
                to reprt-part-number
      move parts-desc(part-index)
                to reprt-description
      move parts-on-hand(part-index)
                to reprt-quantity-on-hand
      move parts-on-ord(part-index)
                to reprt-quantity-on-ord
```

- Comme l'ordinateur ne comprend que le langage 1G, ce programme doit être traduit.
- Ce traducteur est un programme **Compilateur**
- Un langage 3G est indépendant de la machine mais un compilateur est dépendant de la machine.

Quatrième génération (4G)

- Les Langages dits "**de Quatrième Génération**" (L4G) ont pour vocation de s'affanchir du caractère hermétique des langages informatiques classiques.
- Cependant, le développement d'applications s'est avéré un domaine trop complexe pour autoriser l'utilisation d'un langage très proche du langage naturel, par nature trop imprécis et trop ambigu.
- Certes, les L4G de ce domaine ont apporté des gains de lisibilité et de productivité importants, mais ne sont que de super langages évolués.
- Par contre, les langages spécialisés dans le domaine de l'interrogation des Bases de Données, domaine plus simple parce que bien délimité, ont tenté avec plus de succès de se rapprocher du langage naturel.

Passer du langage symbolique au langage binaire

- Nécessité d'un **traducteur** pour passer d'un Langage Symbolique ou Evolué au Langage de Base, seul compréhensible par les circuits de l'ordinateur.
- Notion de **Programme Source** (avant traduction) et de Programme Objet (après traduction).
- Pour passer du Langage Evolué au Langage de Base en créant et en conservant un programme objet : le **Compilateur**.
- Pour passer du Langage Evolué au Langage de Base pour une exécution immédiate, sans conserver la programme objet : **l'Interpréteur**.
- Pour passer du Langage Symbolique au Langage de Base : le programme **Assembleur**.

Les langages

- Des milliers de langages ! Parmi les plus importants :
- Les Langages Symboliques, dits aussi Langages "Assembleur", spécifiques à chaque type d'Ordinateur ou à chaque type de Microprocesseur.
- Les Langages Evolués de Troisième Génération (avant le développement des concepts de la programmation structurée)
  - Langages pour la formation : **Basic**, **Logo**
  - Langages scientifiques : **Fortran**, **APL**
  - Langages de gestion : **Cobol**, **RPG** (ou **GAP**)
  - Tentative d'un langage à vocation universelle : **PL/1**
- Les Langages Evolués de Troisième Génération (après le développement des concepts de la programmation structurée)
  - Langage pour la formation : **Pascal**, nouveaux **Basic**
  - Langages pour le développement : nouveaux **BASIC**, **PASCAL** et **C**
- Les Langages de Quatrième Génération
  - Les langages de requête (**SQL**, **SQL+**, **Prolog**, outils décisionnels)
  - Les langages de développement, liés aux SGBD (**Oracle Forms**, **SAP ABAP**, **SAGE ERP X3**)
- Les langages pour les applications Temps Réel
  - **LTR/3**, **ADA**
- Les Langages de l' "Intelligence Artificielle"
  - **PROLOG**, **LISP**
- Les Langages "Orientés Objet"
  - **Smalltalk, Simula, Python**
  - Extensions « orientées objet » des langages classiques (C => **C++**,**Objective C**; BASIC => **Visual Basic**)
- Les langages de l'Internet
  - **Java** (compilé) et l'écosystème JEE
  - Les langages de script : **Javascript**, **Vbscript**, **Perl** (client), **ASP** et **PHP** (serveur)

Orienté Objet

- L'informatique classique : séparer les données des traitements qui s'appliquent sur ces données.
- Cette séparation était-elle justifiée ? Concevoir la facture d'un système de facturation c'est à la fois définir chacune des rubriques qui la composent et le processus d'établissement de la facture.
- L'*objet* "Facture" : un ensemble de données et leurs attributs (*Propriétés*) ET un ensemble de procédures (*Méthodes*).
- Notion d'*héritage*. Je définis un nouvel objet "Facture_export" que je définis d'abord comme étant un objet de la *classe* "Facture". Il hérite donc de toutes les propriétés (dont la propriété "TVA") et méthodes définies en amont. Concept de *réutilisabilité*.
- Les *Langages Orientés Objet*
  - Facture_export.TVA = 0 (affectation d'une valeur à une propriété)
  - Imprimer Facture_export (lancement d'une méthode)
- Bibliothèques d'objets. Objets techniques et objets "métier".

Autres langages

- A côté des **langages de programmation** dont nous venons de parler et qui implantent physiquement les modèles de données et de traitements traduits en algorithmes, il y a d'autres langages :
- des **langages de définition de données** qui rendent concrets les modèles conceptuels de données ;
- des **langages de requête**, comme SQL, qui permettent de réaliser des traitements (création, interrrogation, mise à jour, suppression) sur les données définies;
- des **langages de decription de pages**, propres au monde du web, comme HTML ou XML, qui utilisent des balises insérées dans le texte pour le structurer et lui donner une forme particulière.

Cas d'une application web

- Une application web peut combiner plusieurs de ces langages :
  - **HTML** (description) pour définir le contenu et la forme des pages avec diverses balises;
  - **Javascript** (programmation) pour animer la page en exécutant au niveau du poste de l'utilisateur (client) du code inséré dans la page entre des balises <SCRIPT> et </SCRIPT>;
  - **PHP** (programmation) pour animer la page en exécutant au niveau du serveur du code inséré dans la page entre des balises <?php et ?>;
  - **SQL** (requête) qui sera appelé par le code PHP pour interroger la base de données.

Qu'est-ce-qu'un framework

- En informatique, un ***framework*** (cadre logiciel) est un ensemble de bibliothèques, d'outils et de conventions permettant le développement d'applications.
- Il a pour objectif de fournir suffisamment de briques logicielles et d'imposes suffisamment de rigueur pour pouvoir produire une application aboutie et dont la maintenance est aisée.
- Ces composants sont organisés pour être utilisés en interaction les uns avec les autres, parfois dans le cadre d'un schéma d'urbanisation.
- Un *framework* est habituellement implémenté à l'aide d'un langage à objets, bien que cela ne soit pas strictement nécessaire.

Qu'est-ce-qu'un API ?

- Une interface de programmation (***Application Programming Interface* ou *API***) est un ensemble de fonctions, procédures ou classes mises à disposition des programmes informatiques par une bibliothèque logicielle, un système d'exploitation ou un service.
- Ces éléments facilitent l'écriture des programmes en fournissant des procédures toutes faites pour gérer des ressources particulières (affichage, pilotage de périphériques…).
- La connaissance des API est indispensable à l'interopérabilité entre les composants logiciels.
- Dans le cas typique d'une bibliothèque, il s'agit généralement de fonctions considérées comme utiles pour d'autres composants.
- Les *API* peuvent donc être une partie d'un *framework*.

Qu'est-ce-qu'un SDK ?

- Un **SDK (*Software development kit*)** regroupe les utilitaires pour développer dans un *framework*, ou pour une plate-forme, ou en complément d'un progiciel, …
- Il peut se réduire à une simple API, sous la forme d'une bibliothèque de modules de code, ou aller jusqu'à inclure des dispositifs matériels pour communiquer avec un système donné.
- Ils incluent généralement des outils d'aide à la mise au point et des utilitaires variés, allant jusqu'à constituer parfois un véritable environnement de développement intégré (*EDI*)
- Les *SDK* incluent fréquemment des exemples de code et des notices de support technique ou toute autre documentation permettant d'éclaircir nombre de points concernant le système de base concerné.

Qu'est-ce-qu'un EDI ?

- Un **Environnement de Développement Intégré** (EDI ou *IDE* pour *Integrated Development Environment*) est un programme regroupant un ensemble d'outils pour le développement de logiciels.
- Autrefois, les EDI étaient souvent dédiés à un seul langage de programmation (par exemple *Visual C++* pour C++, *Visual Basic* pour Basic, *Eclipse* pour Java).
- Ce n'est plus le cas (*Visual studio* pour Basic, C++, C#, F#, JScript; *Eclipse* pour Java, C++, PHP)
- On peut également trouver dans un EDI un système de gestion de versions et différents outils pour faciliter la création de l'interface graphique (*GUI* pour *Graphical User Interface*).

Exemples

- Un **framework** : Microsoft .Net framework 3.5
- Un **API** : *Java Message Service* (*JMS*) : *API* de communication asynchrone par message dans le framework JEE
- Un **SDK** : Le SDK Android fournit les bibliothèques d'API et les outils de développement nécessaires pour construire, tester et deboguer des applications pour les mobiles exploités sous Android.
- Un **EDI** : Eclipse (Open source) vs Visual Studio (propriétaire Microsoft)

Quelles réponses à nos questions ?

- Qu'est ce qu'un algorithme ?
- Un algorithme est une suite finie et non-ambiguë ...?? permettant de résoudre un problème, en particulier de ...??

Quelles réponses à nos questions ?

- Qu'est ce qu'un algorithme ?
- Un algorithme est une suite finie et non-ambiguë d'opérations ou d'instructions permettant de résoudre un problème, en particulier de formaliser le processus logique de déroulement d'un logiciel.

Quelles réponses à nos questions ?

- Qu'est-ce que la programmation ?
- Traduction d'un problème ...??.
- Qu'est-ce que la programmatique ?
- Méthode pour ...??

Quelles réponses à nos questions ?

- Qu'est-ce que la programmation ?
- Traduction d'un problème dans un langage compréhensible par l'ordinateur.
- Qu'est-ce que la programmatique ?
- Méthode pour concevoir un bon programme

Quelles réponses à nos questions ?

- Qu'est ce qu'un langage informatique ?
- Un langage informatique est un ensemble de ...??? permettant à celui qui veut écrire un programme d'exprimer et de formaliser son besoin pour ...?? de manière compréhensible par ceux-ci.

Quelles réponses à nos questions ?

- Qu'est ce qu'un langage informatique ?
- Un langage informatique est un ensemble de signes permettant à celui qui veut écrire un programme, d'exprimer et de formaliser son besoin pour le communiquer aux circuits de l'ordinateur de manière compréhensible par ceux-ci.

Quelles réponses à nos questions ?

- Quelles typologies pour les langages informatiques ?
- Plusieurs typologies envisageables : langages assertionnels et langages procéduraux, ...???; langages compilés, pseudo-compilés et interprétés; langages orientés-objet ou non.

Quelles réponses à nos questions ?

- Quelles typologies pour les langages informatiques ?
- Plusieurs typologies envisageables : langages assertionnels et langages procéduraux, L2G, L3G et L4G; langages compilés, pseudo-compilés et interprétés; langages orientés-objet ou non.

Quelques questions

Poursuite de la démarche

Nous poursuivons la démarche initiée dans les modules précédents : **Identifier un besoin (Cahier des charges)** et **Savoir poser et analyser un problème (Démarche de conception)**

- LES ETAPES DE CONCEPTION D'UN LOGICIEL

1. Identifier un besoin
2. Analyser le contexte du besoin : processus et données
3. Elaborer une solution globale pour répondre au besoin
4. Choisir un niveau de langage (Symbolique, évolué L3G, L4G)
5. Décomposer la solution en opérations élémentaires jusqu'au niveau où il devient possible d'associer à l'opération une instruction du niveau de langage choisi
6. Rédiger et tester le pseudo-code
7. Choisir le langage
8. Rédiger les programmes
9. Compiler ou interpréter (traduire le langage choisi dans le langage de base de l'ordinateur)
10. Vérifier que les programmes satisfassent bien le besoin exprimé à l'aide d'un jeu d'essai
11. Documenter le programme
12. Livrer le logiciel à l'utilisateur (Formation, Assistance)

Niveau conception

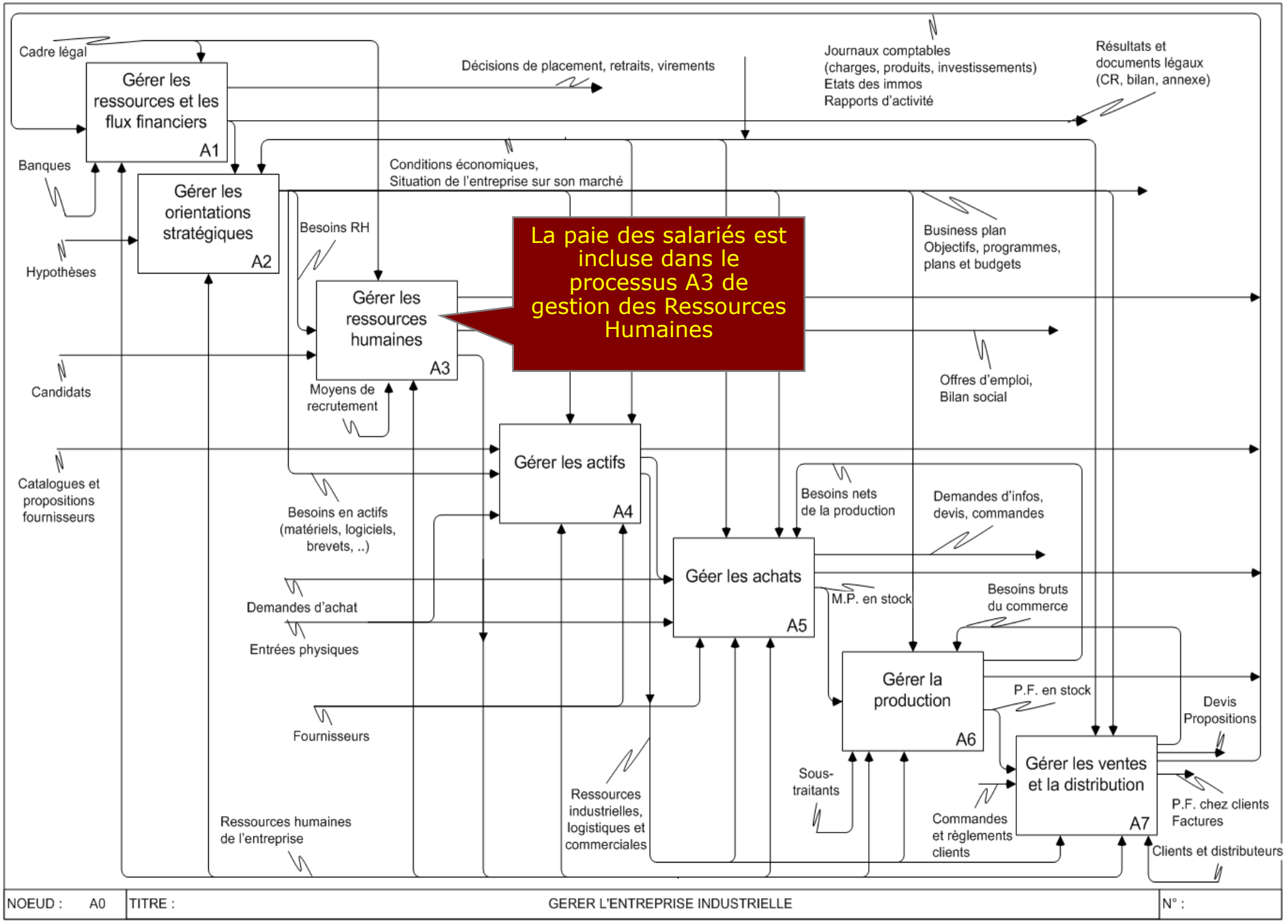
Niveau réalisation (auquel est consacré ce chapitre)

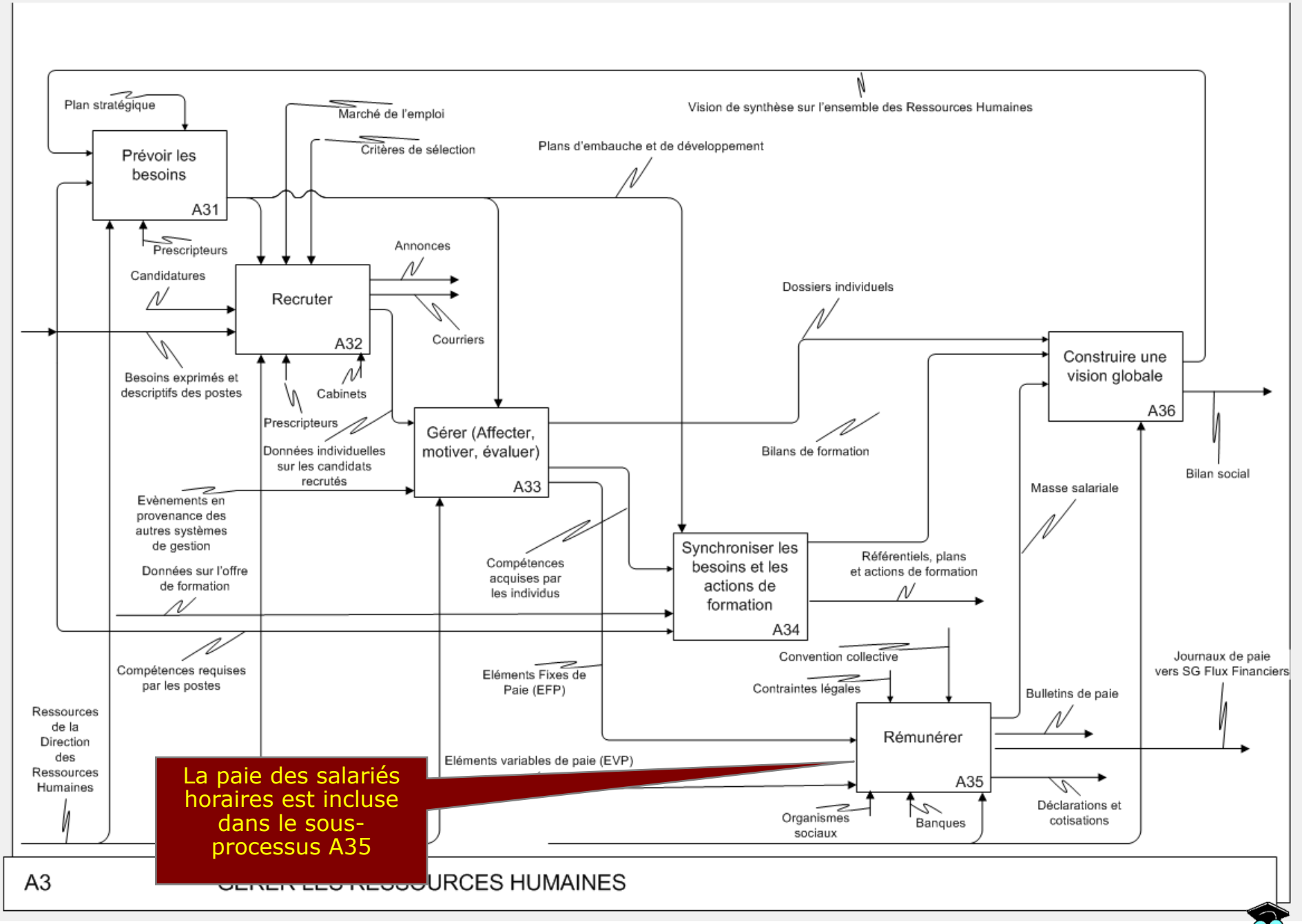
1. Identifier un besoin

- Rémunérer le personnel horaire
  - Saisir les éléments variables de la paie (EVP)
  - Calculer les éléments du bulletin de paie

2. Analyser le contexte du besoin

- Décrire le processus « Gérer l'entreprise »
- Décrire le sous-processus « Gérer les ressources humaines » que l'opération précédente a mis en valeur
- Décrire les données décrivant le salarié


Cadre légal
Gérer les ressources et les flux financiers
A1
Décisions de placement, retraits, virements
Journaux comptables
(charges, produits, investissements)
Etats des immos
Rapports d'activité
Résultats et documents légaux
(CR, bilan, annexe)
Banques
Gérer les orientations stratégiques
A2
Conditions économiques,
Situation de l'entreprise sur son marché
Hypothèses
Besoins RH
La paie des salariés est incluse dans le processus A3 de gestion des Ressources Humaines
Business plan
Objectifs, programmes,
plans et budgets
Gérer les ressources humaines
A3
Candidats
Moyens de recrutement
Offres d'emploi,
Bilan social
Gérer les actifs
A4
Catalogues et propositions fournisseurs
Besoins en actifs
(matériels, logiciels,
brevets, ..)
Besoins nets de la production
Demandes d'infos,
devis, commandes
Géer les achats
A5
Demandes d'achat
M.P. en stock
Besoins bruts du commerce
Entrées physiques
Gérer la production
A6
P.F. en stock
Devis
Propositions
Fournisseurs
Sous-traitants
Gérer les ventes et la distribution
A7
P.F. chez clients
Factures
Ressources industrielles, logistiques et commerciales
Commandes et règlements clients
Ressources humaines de l'entreprise
Clients et distributeurs
NOEUD : A0
TITRE : GERER L'ENTREPRISE INDUSTRIELLE
N° :


Plan stratégique
Marché de l'emploi
Vision de synthèse sur l'ensemble des Ressources Humaines
Prévoir les besoins
A31
Critères de sélection
Plans d'embauche et de développement
Prescripteurs
Annonces
Candidatures
Recruter
A32
Courriers
Dossiers individuels
Construire une vision globale
A36
Besoins exprimés et descriptifs des postes
Cabinets
Prescripteurs
Données individuelles sur les candidats recrutés
Gérer (Affecter, motiver, évaluer)
A33
Bilans de formation
Bilan social
Masse salariale
Evènements en provenance des autres systèmes de gestion
Compétences acquises par les individus
Synchroniser les besoins et les actions de formation
A34
Référentiels, plans et actions de formation
Données sur l'offre de formation
Convention collective
Journaux de paie vers SG Flux Financiers
Compétences requises par les postes
Eléments Fixes de Paie (EFP)
Contraintes légales
Bulletins de paie
Ressources de la Direction des Ressources Humaines
Rémunérer
A35
Eléments variables de paie (EVP)
Organismes sociaux
Banques
Déclarations et cotisations
La paie des salariés horaires est incluse dans le sous-processus A35
A3
URCES HUMAINES

Le modèle des données d'un salarié est complexe.

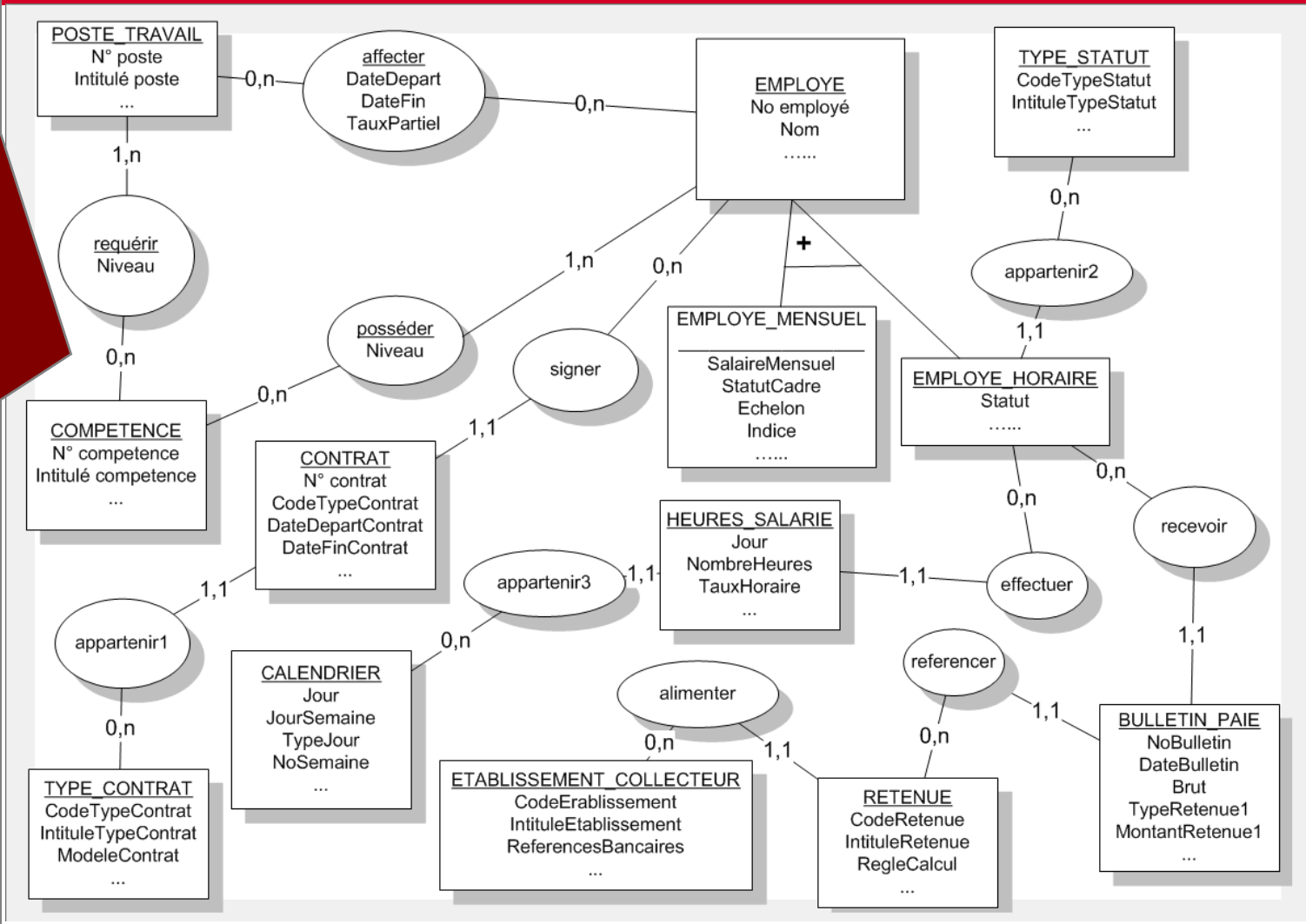
Ce schéma n'en représente qu 'une partie.

Un employé peut être mensualisé ou payé au pro-rata des heures effectuées.

Un Employé horaire effectue des heures, référencées par rapport à un calendrier qui définira heures ouvrables, jours de la semaine, jours fériés, chômés, etc.

Un employé horaire reçoit un bulletin de paie. Le brut est calculé sur la base nombres d'heures * taux horaire

Ce bulletin peut référencer plusieurs retenues (cotisations sociales, retraites, retenues à la source, etc.)



3. Elaborer une solution globale

Nous allons recenser les éléments permettant de concevoir un programme en prenant l'exemple **(très simplifié bien sûr !)** de l'élaboration d'un Bulletin de Paie dans une optique L3G :

- L'OBJECTIF
  - Calcul du Bulletin de Paie
- LES RESSOURCES
  - Un Micro-ordinateur disposant d'un interpréteur du langage BASIC
  - Un employé du service Paie
- LES CONTRAINTES
  - Les règles légales
  - La convention collective de l'entreprise
- LES ENTREES
  - Le nom du salarié
  - Le taux horaire
  - Le nombre d'heures travaillées
- LE TRAITEMENT
  - Calcul du brut (taux horaire * nombre heures)
  - Calcul des retenues (selon contraintes)
  - Calcul du net
- LES SORTIES
  - Le Bulletin de Paie (Brut, retenues et Net)

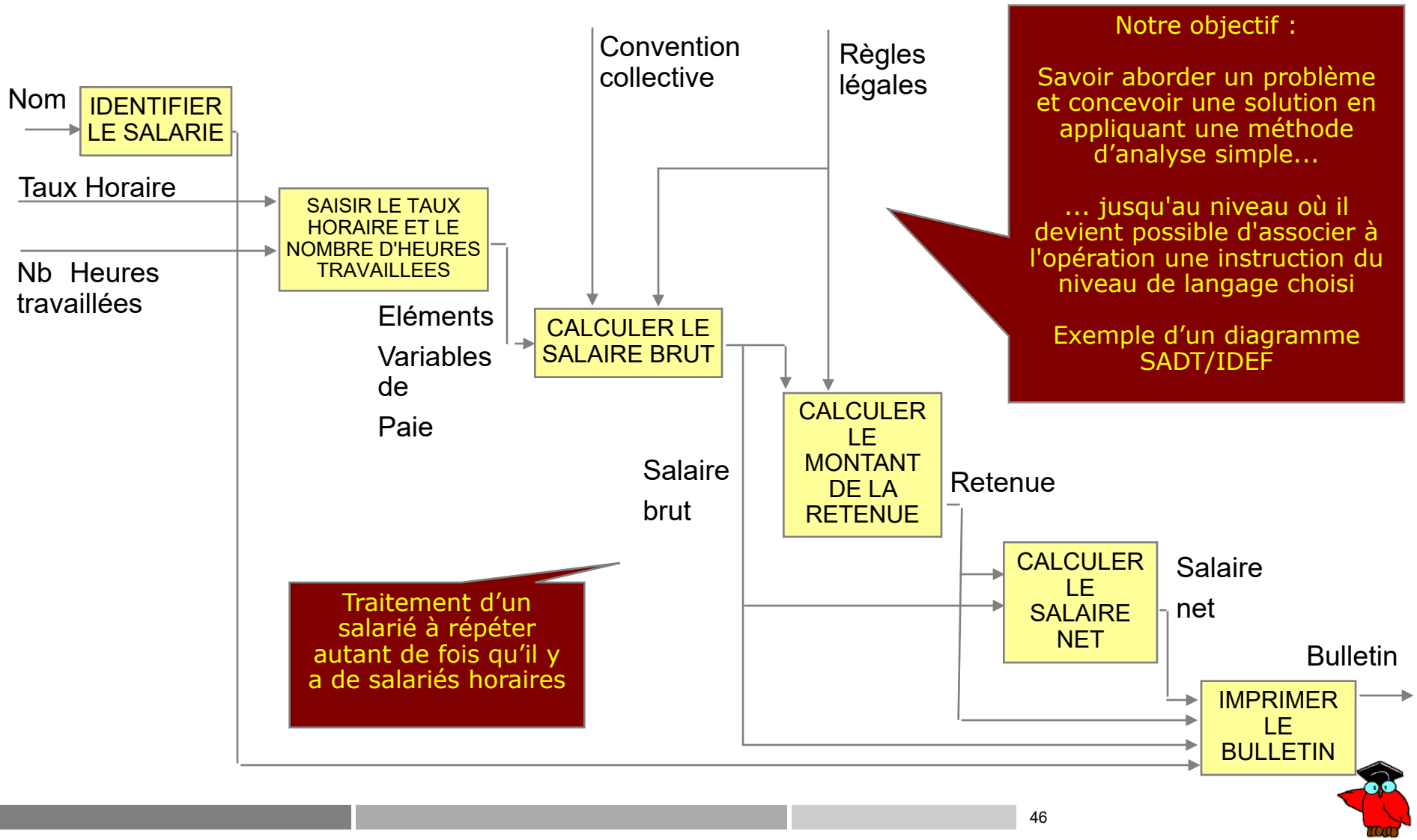
Pour les besoins de l'exemple, nous avons considérablement simplifié la problématique « données » par rapport à notre modèle précédent

4. Choisir un niveau de langage

- Choix L3G
- Autres choix
  - L2G Assembleur
  - L4G Langage 4ème génération
- Critères de choix
  - Complexité
  - Coût
  - Adéquation au problème posé

5. Décomposer la solution en opérations élémentaires

L'élaboration de la solution : le niveau 1 (La procédure du Bulletin individuel)



6. Rédiger et tester le pseudo-code

L'élaboration de la solution : le niveau 2 (Pseudo Code)

Notre objectif :

Maîtriser la démarche de réalisation en sachant analyser un problème simple, en le formalisant sous forme de **« pseudo-code »**

Mise en évidence des trois structures de base de l'algorithmique :
. Enchaînement (séquence)
. Répétitive (Tant que .... répéter ....)
. Alternative (Si ... alors ... sinon ...)

Contrainte (imaginaire) :
Taux de 2,8% pour la tranche inférieure à 100 000 FCfa, taux de 1,5% pour la tranche supérieure

```
$:    /* Calcul du bulletin de paie */
$1:  TANT QUE tous les salariés n'ont pas été traités
        $11:    Afficher "Nom du Salarié"
        Enregistrer sous l'étiquette N$ la valeur saisie
        Afficher "Taux horaire"
        Enregistrer sous l'étiquette T la valeur saisie
        Afficher "Nombre d'heures travaillées
        Enregistrer sous l'étiquette H la valeur saisie
        Imprimer "Fiche de Paie de Mr" N$
        Calculer S = H * T
        Imprimer "Salaire brut = " S       :11$
        $12:   SI S > 100000
                ALORS $121:   R = 100000 * 0,028 + (S - 30000) * 0,015
                   :121$
                SINON  $122:  R = S *0,028  :122$
        FIN SI :12$
        $13:  Imprimer "Retenue CNPS: " R
        Calculer S1 = S - R
        Imprimer "Salaire Net : " S1
        Passer au bulletin suivant (Saut de Page)  :31$
FIN TANT QUE     :1$
$2:  Imprimer « Fin de Traitement »  :2$
/* Fin programme */ :$
```

7. Choisir un langage

- Choix Basic
- Autres choix
  - C
  - C++
  - Java
  - Cobol ....etc.
- Critères de choix
  - Degré de maturité
  - Outils disponibles
  - Support technique

8. Ecriture du programme

L'élaboration de la solution : le niveau 3 (Le langage de Programmation - Exemple 3G –BASIC livré à l'origine avec le PC)

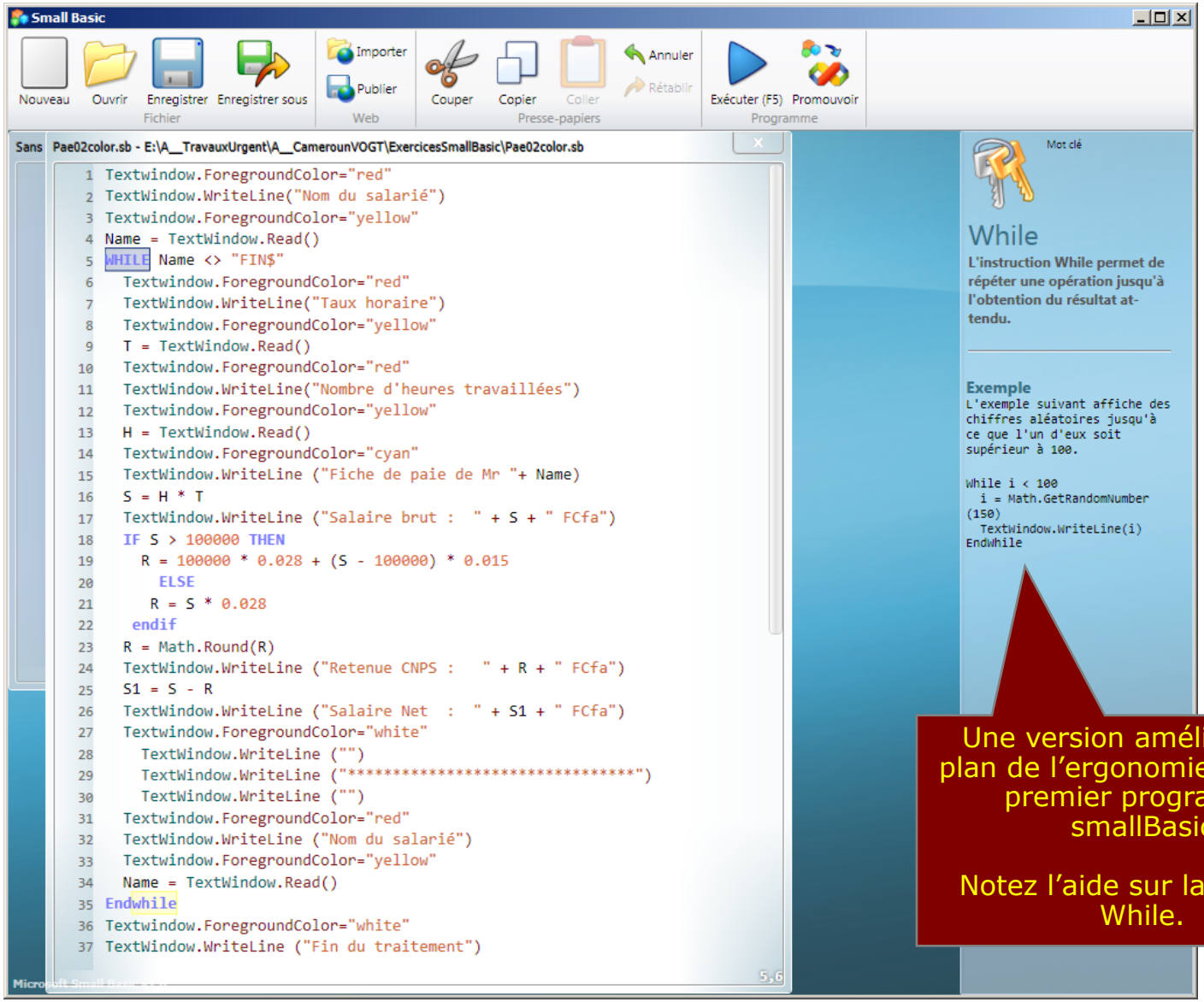
```
010     CLS
020     PRINT "Nom du salarié"
030     INPUT N$
040     WHILE N$ <> "FIN$"
050         PRINT "Taux horaire"
060         INPUT T
070         PRINT "Nombre d'heures travaillées"
080         INPUT H
090         LPRINT "Fiche de paie de Mr ";N$
100         S = H * T
110         LPRINT "Salaire brut = ";S
120         IF S > 100000
                THEN R = 100000 * 0,028 + (S - 300000) * 0,015
                ELSE R = S * 0,028
130         LPRINT "Retenue CNPS : ";R
140         S1 = S - R
150         LPRINT "Salaire Net : "; S1
160         LPRINT CHR£(12)
170         CLS
180         PRINT "Nom du salarié"
190         INPUT N$
200     WEND
210     PRINT "Fin du traitement"
```

8bis. Ecriture du programme

L'élaboration de la solution : le niveau 3 (Le langage de Programmation - Exemple 3G –Microsoft Small BASIC)

```
1       TextWindow.WriteLine("Nom du salarié")
2       Name = TextWindow.Read()
3       WHILE Name <> "FIN$"
4          TextWindow.WriteLine("Taux horaire")
5          T = TextWindow.Read()
6          TextWindow.WriteLine("Nombre d'heures travaillées")
7          H = TextWindow.Read()
8          TextWindow.WriteLine ("Fiche de paie de Mr "+ Name)
9          S = H * T
10         TextWindow.WriteLine ("Salaire brut = " + S)
11         IF S > 100000 THEN
12              R = 100000 * 0.028 + (S - 100000) * 0.015
13         ELSE
14              R = S * 0.028
15         endif
140        TextWindow.WriteLine ("Retenue CNPS : " + R)
150        S1 = S - R
160        TextWindow.WriteLine ("Salaire Net : " + S1)
170        TextWindow.WriteLine ("Nom du salarié")
180        Name = TextWindow.Read()
190     Endwhile
200     TextWindow.WriteLine ("Fin du traitement")
210
```

9. Compiler ou interpréter



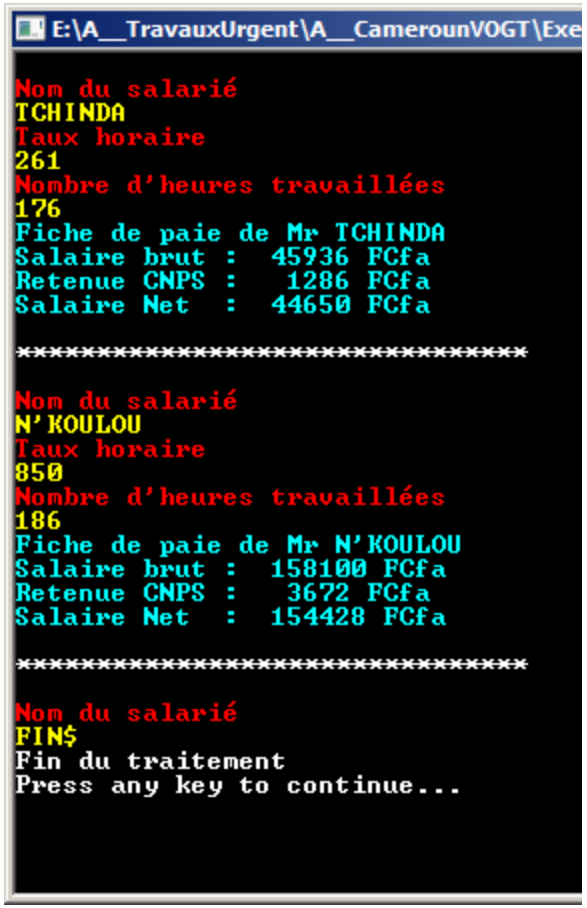
Une version améliorée, au plan de l'ergonomie, de notre premier programme smallBasic

Notez l'aide sur la fonction While.

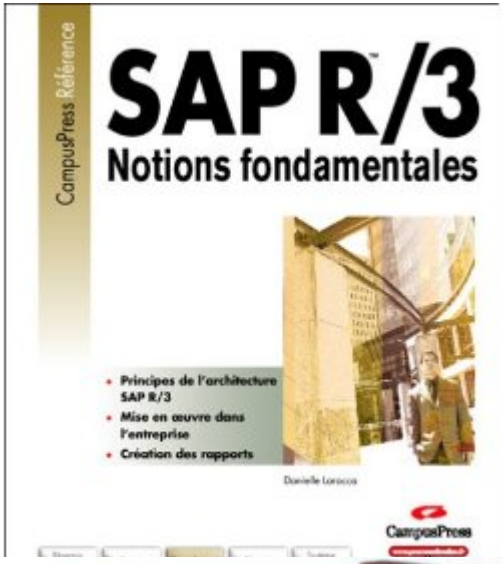
10. Vérifier que les programmes satisfassent bien le besoin exprimé

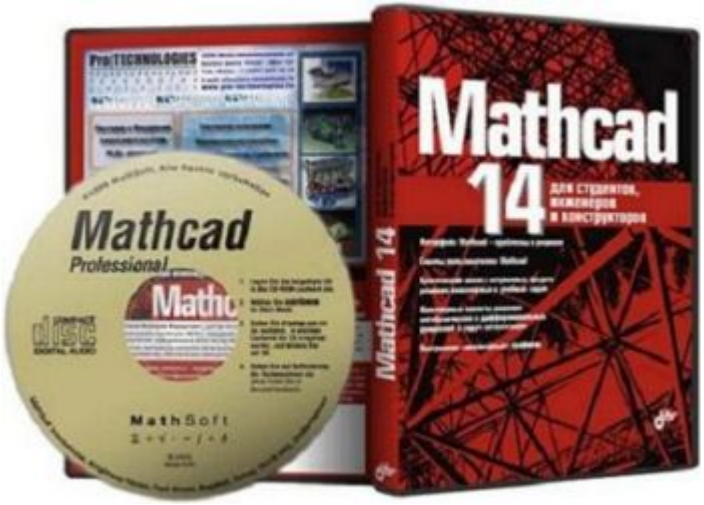
- **Constitution d'un jeu d'essai :**
- Salarié 1 : Benjamin TCHINDA
- Nombre d'heures : 176
- Taux horaire : 261
- Résultats attendus :
- Brut = 45 936, Retenue = 1 286, Net = 44 650

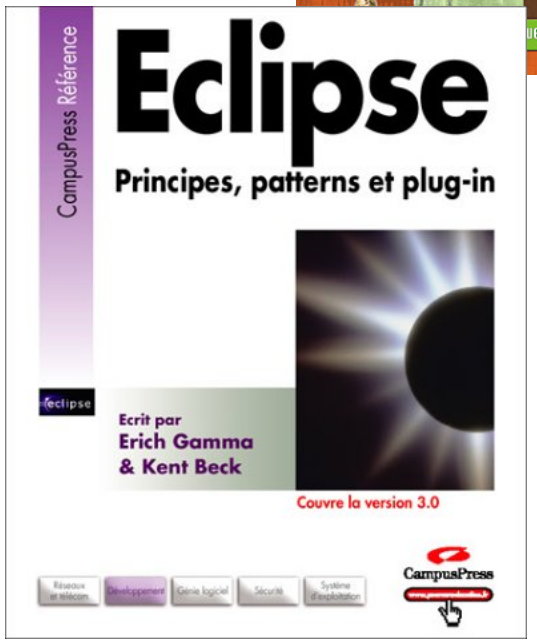
- Salarié 2 : Sébastien N'KOULOU
- Nombre d'heures : 186
- Taux horaire : 850
- Résultats attendus :
- Brut=158 100, Retenue=3 672, Net = 154 428



11. Documenter le programme







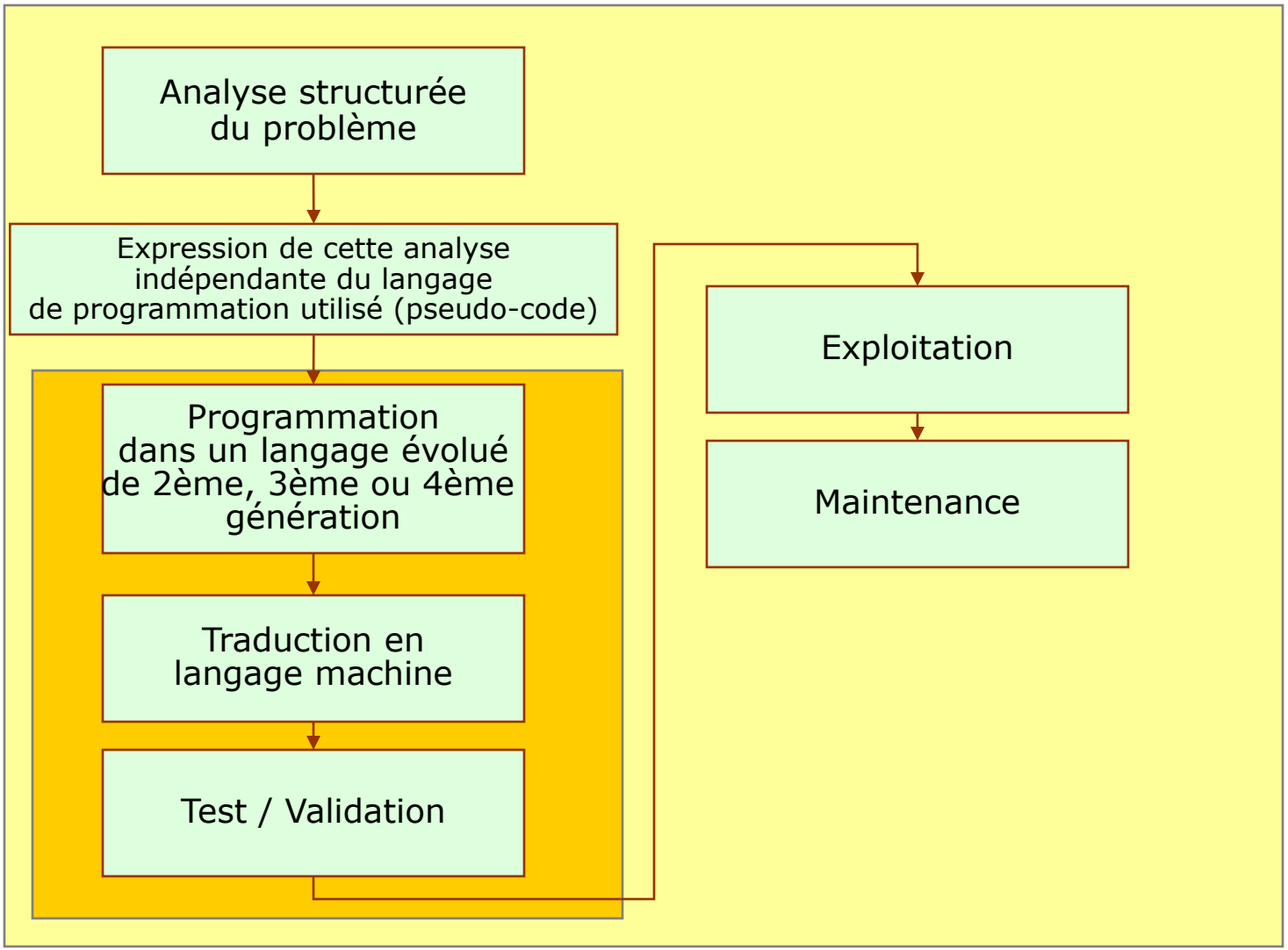
12. Livrer le logiciel à l'utilisateur

- Aider les utilisateurs à comprendre leur rôle et à prendre conscience de leurs responsabilités au sein de la nouvelle organisation;
- Aider les utilisateurs à acquérir le juste nécessaire en ce qui concerne les technologies de l'information;
- Aider les utilisateurs à maîtriser les nouveaux outils disponibles sur et accessibles depuis leur poste de travail;
- Aider les utilisateurs à reconsidérer leur poste de travail en fonction des possibilités et des contraintes de ces nouveaux outils;
- Aider les utilisateurs à faire bien;
- Aider les utilisateurs à faire simplement;
- Aider les utilisateur à agir en toute sécurité;
- Aider les utilisateurs à agir rapidement;
- Aider les utilisateurs à agir ensemble.

Programmatique

- Une analyse structurée du problème
- Un mode d'expression de cette analyse indépendante du langage de programmation utilisé
- Un référentiel pour la traduction en un langage compréhensible par un compilateur, interpréteur ou assembleur lui-même en mesure de fournir une traduction dans le langage binaire de la machine
- Un résultat exploitable pendant les phases ultérieures d'exploitation et de maintenance

Programmatique



Quelles réponses à nos questions ?

Quelles étapes pour passer de l'identification du besoin au déploiement de la solution ?

1. Analyser le contexte du besoin en modélisant processus et données selon une démarche structurée;
2. Elaborer une solution globale pour répondre au besoin;
3. Décomposer la solution en opérations élémentaires de manière à pouvoir formaliser l'algorithme d'exécution en pseudo-code;
4. Rédiger les programmes dans le langage choisi;
5. Compiler ou interpréter (Traduction en langage machine);
6. Valider le programme à l'aide d'un jeu d'essai;
7. Rédiger la documentation pour l'usage et pour la maintenance;
8. Livrer le logiciel à l'utilisateur (Formation, Assistance).

Quelles réponses à nos questions ?

- Qu'est ce que le pseudo-code ?
- Le pseudo-code est un langage simplifié, proche du langage naturel, qui décrire un algorithme sans référence à un langage de programmation particulier.
- Quel est son intérêt ??
- Plus lisible car débarrassé des contraintes syntaxiques d'un langage de programmation, il permet de se concentrer sur le formalisme de l'algorithme.

Quelles réponses à nos questions ?

- Comment choisir un langage de programmation ?
- Pas de réponse simple.
- Le meilleur langage est celui que vous connaissez !
- ... mais vous ne devez pas seulement prendre en compte votre expérience.
- Pour pouvoir correctement choisir quel est le meilleur langage de programmation vous devez réunir tous les différents critères (nature, portabilité, stabilité, pérennité, performances, etc.) qui sont présents dans le projet et ensuite considérer votre expérience qui doit en permanence évoluer.

Quelles réponses à nos questions ?

- Quelles sont les étapes du développement d'un programme ?
- S'inscrit dans les points 2 à 7 du processus général
  - Conception générale
  - Conception détaillée
  - Programmation
  - Test et validation
  - Documentation