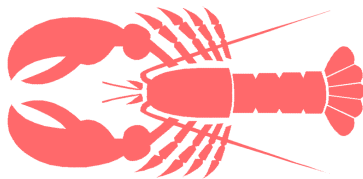


**UNE COOPERATION REUSSIE ENTRE  
PROGRAMMATION LINEAIRE EN  
NOMBRES ENTIERS ET PROGRAMMATION  
PAR CONTRAINTES POUR LES PROBLEMES  
D'EMPLOI DU TEMPS**

*Stage de DRT réalisé par : David GRAVOT  
Responsable de stage au LIA : Philippe MICHELON*

*Année universitaire : 1999/2000*



*reviewed by ROSTUDEL, November 2006*

# REMERCIEMENTS

En premier lieu je tiens à remercier Mrs Philippe MICHELON et Henri BERINGER, le premier ayant proposé ma candidature au second, membre fondateur de la société TEMPOSOFT et précédemment directeur des consultants d'ILOG. Au sein de l'équipe d'optimisation, j'ai également rencontré des collègues aussi sympathiques que compétents : Malika et Nabil GUERINIK, docteurs en optimisation, Henri DANCY, responsable éclectique de l'équipe et, récemment parmi nous, Guillaume BEUCHARD et Alex FLEISCHER. Mes remerciements vont aussi à l'équipe des consultants ; souvent à l'origine de nouvelles fonctionnalités dans l'optimisation, et plus largement encore, à l'ensemble du personnel de TempoSoft, où la bonne humeur générale contribue grandement aux résultats de cette start-up.

<b>Introduction.....</b>	<b>4</b>
<b>I. L'entreprise TempoSoft.....</b>	<b>4</b>
A) Présentation.....	4
B) Organigramme .....	5
C) Les progiciels TempoSoft.....	6
<b>II. Présentation du travail de DRT.....</b>	<b>7</b>
<b>Chapitre 1 : Paramétrage de TempoPlanner.....</b>	<b>8</b>
<b>I. Schéma général .....</b>	<b>8</b>
<b>II. activités – Les charges de travail .....</b>	<b>9</b>
A) L'organisation générale de l'entreprise.....	9
B) Les activités et les propriétés des activités .....	10
C) Les fonctions, les métiers, les postes de travail .....	11
D) Les charges de travail.....	12
<b>III. Les Ressources Humaines.....</b>	<b>15</b>
A) Description des ressources humaines.....	15
B) Les préférences des salariés.....	17
C) Gestion prévisionnelle des indisponibilités .....	17
<b>IV. Les contrats et la réglementation.....</b>	<b>18</b>
A) Organisation des règles en contrat .....	18
B) Le langage de paramétrage .....	18
C) Les compteurs .....	20
D) Les modèles (briques de planification) .....	21
<b>Chapitre 2 : L'Optimisation dans TempoPlanner.....</b>	<b>24</b>
<b>I. Principes généraux de l'optimisation .....</b>	<b>24</b>
A) Génération de planning .....	24
B) L'Optimisation : Arbitrage entre objectifs contradictoires .....	25
C) Généricité.....	25
D) Rappel sur les techniques d'optimisation combinatoire.....	25
E) La Programmation par contraintes à l'aide des outils ILOG.....	26
F) Options d'une optimisation .....	29
<b>II. Grandes lignes de l'optimisation .....</b>	<b>30</b>
<b>III. L'Optimisation « Globale » , une recherche arborescente originale.....</b>	<b>32</b>
A) Construction du modèle mathématique.....	32
A.1 Décomposition temporelle et en groupes d'activités .....	32
A.2 Durées fondamentales et contraintes de charge .....	33
A.3 La création des modèles individuels .....	33
A.4 La création des compteurs, contraintes et objectifs .....	34
B) La programmation « sous-hypothèse » .....	34
B.1 Principe général.....	34
B.2 Implémentation .....	35
B.3 Contrainte sous hypothèses et relaxation linéaire .....	36
C) Heuristique de l'optimisation globale .....	36
D) Résultats .....	39

D.1	Deux exemples numériques .....	39
D.2	Gérer la combinatoire .....	40
<b>IV.</b>	<b>L' Optimisation « Détaillée », une approche locale .....</b>	<b>41</b>
A)	Rappel des objectifs .....	41
B)	Heuristique.....	41
B.1	Schéma général .....	41
B.2	Sélection gloutonne d'une fenêtre.....	42
B.3	Amélioration locale au sein de la fenêtre sélectionnée .....	43
C)	Résultats .....	44
<b>V.</b>	<b>Conclusion.....</b>	<b>45</b>

# Introduction

## I. L'entreprise TempoSoft

### A) Présentation

TempoSoft est une société française qui a été créée en février 1998 sous l'impulsion de la société ILOG (société française cotée sur le marché boursier américain, et leader mondial dans l'édition de composants C++), avec le concours financier de deux grands investisseurs spécialisés dans le financement de sociétés « High Tech », ATLAS Venture et CDC Innovation (filiale de la Caisse des Dépôts et Consignations). Ce montage permet à ce jour de capitaliser la structure TempoSoft à plus de 100 millions de francs.

Le métier central de TempoSoft est d'être un éditeur de progiciel dans le domaine de l'organisation et de l'optimisation du temps de travail. L'ensemble des ressources de TempoSoft est focalisé sur le développement de produits autour de ce pôle fonctionnel. Une telle approche permet à TempoSoft d'assurer à un moindre coût pour ses clients, une évolution continue, tant sur le plan technique que sur le plan fonctionnel ou réglementaire, de l'ensemble de ses produits.

Par la qualité des algorithmes d'optimisation mis en œuvre, les logiciels de TempoSoft permettent d'améliorer de façon conséquente l'emploi des ressources humaines conformément aux objectifs de l'entreprise (masse salariale, qualité du service au client, motivation des salariés...). Ils offrent donc un retour sur investissement rapide en termes tant quantitatifs (réduction des coûts ou augmentation du chiffre d'affaire) que qualitatifs.

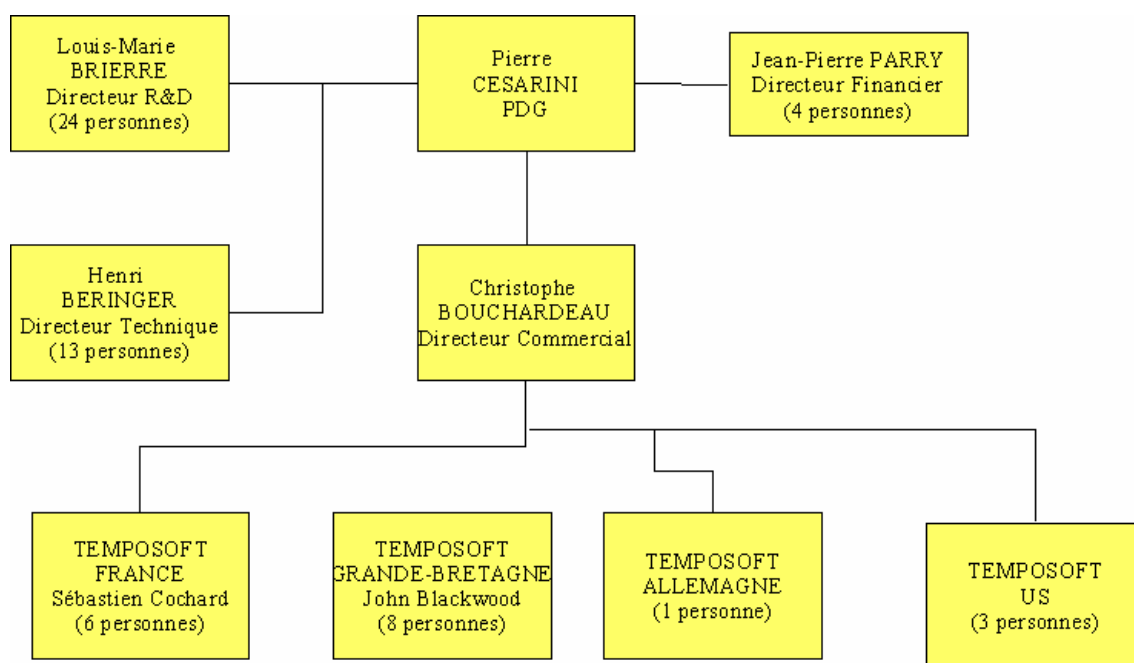
A travers une gamme de logiciels intégrés, TempoSoft fournit des outils permettant aux entreprises de gérer et d'optimiser leur organisation du temps de travail dans toute sa complexité. Pour cela, TempoSoft articule son offre autour des modules *TempoPlanner* et *TempoScheduler* dont nous reparlerons un peu plus loin.

Parallèlement, TempoSoft garantit à ses clients, si nécessaire, la prise en considération de l'existant en terme de logiciels de paie et de gestion des Ressources Humaines.

Ces outils puissants et simples à manipuler s'adressent aux opérationnels et aux cadres de la fonction personnel. Ils leur permettent de répondre à un enjeu majeur de leur métier en particulier dans le contexte du passage aux 35 heures : la résolution optimisée des problématiques associées à l'aménagement et la réorganisation du temps de travail.

La force de TempoSoft est de bâtir un avantage compétitif déterminant basé sur la combinaison de compétences de haut niveau, tant dans la maîtrise technologique du développement de progiciels intégrant la composante optimisation, que dans la connaissance fonctionnelle approfondie de la planification des horaires.

## **B) Organigramme**



Mon stage DRT s'effectue au sein du service Optimisation de TempoSoft, secteur dirigé par M. Henry DANCY, également mon maître de stage. Ce service est inclus dans le département Recherche et Développement dirigé M Louis-Marie BRIERRE.

## C) Les progiciels TempoSoft

TempoSoft édite deux logiciels, TempoPlanner et TempoScheduler, logiciels basés sur des architectures logicielles comparables mais destinés à des publics différents : TempoPlanner est un outil de simulation, il gère donc des ressources virtuelles, alors que TempoScheduler est un outil opérationnel prévu pour planifier des ressources réelles.

### ➤ **TempoPlanner**

TempoPlanner, progiciel de paramétrage, de simulation, de planification et d'optimisation du temps de travail permet de modéliser les données de l'entreprise mais aussi de choisir l'organisation du temps de travail la mieux adaptée pour chaque salarié ou groupe de salariés en tenant compte des ressources, des charges et des contraintes de l'entreprise.

Ainsi, les utilisateurs peuvent simuler et analyser toute mesure de réorganisation du temps de travail (annualisation, élargissement des horaires...) et mesurer ses impacts en fonction des objectifs de l'entreprise.

TempoPlanner s'adresse aux responsables fonctionnels (DRH, Responsables des organisations...) souhaitant sélectionner les formes d'aménagement du temps de travail les mieux adaptées au contexte de leur entreprise.

### ➤ **TempoScheduler**

TempoScheduler, progiciel de planification et de suivi des horaires et des activités permet d'optimiser les horaires quotidiens de chaque salarié.

TempoScheduler tient compte des charges exceptionnelles, des contraintes sur les activités, des compétences et des préférences de chaque salarié ainsi que des congés et des absences prévues. Il permet de gérer les mises à jour des plannings prévisionnels directement ou en s'interfacant avec les différents logiciels de GTA (Gestion des Temps et des Activités), des logiciels de paie et plus généralement avec des logiciels de gestion des ressources humaines existants.

C'est aussi un outil d'aide à la décision qui permet de construire les tableaux de bord nécessaires pour suivre l'évolution des ressources et des activités sur le court, moyen et long terme. TempoScheduler s'adresse aux responsables opérationnels (responsables d'agence, de service, de magasin ou d'atelier...) souhaitant mettre en œuvre et gérer facilement toute nouvelle organisation du temps de travail au sein de leur équipe.

## II. Présentation du travail de DRT

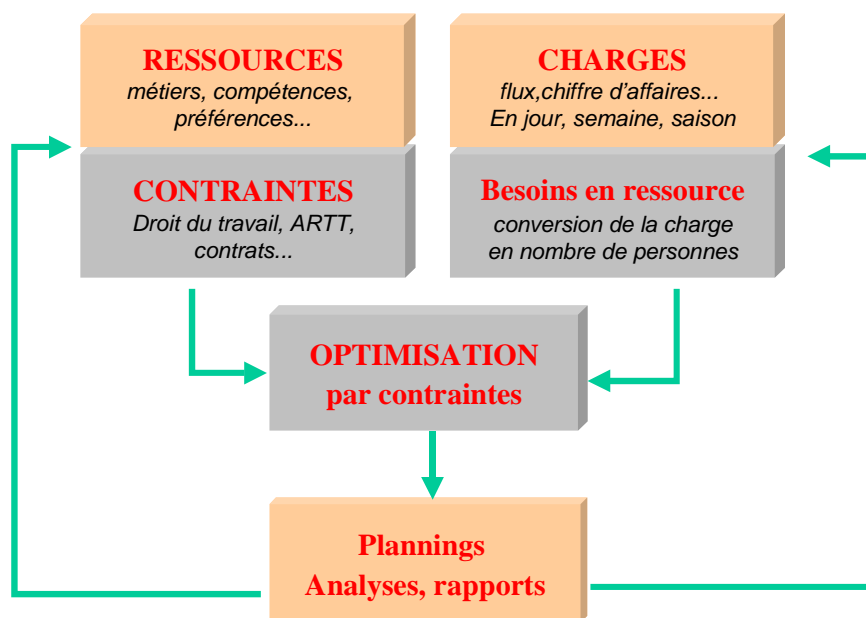
Je suis arrivé dans l'équipe à un moment où les premières fonctionnalités d'optimisation commençaient à être développées dans une optique réellement opérationnelle. Jusqu'alors, un certain nombre d'études avaient été réalisées pour des clients cherchant à simuler des réglementations ARTT\*, mais le logiciel n'était pas utilisable dans un environnement de travail réel, tant en terme d'IHM que de souplesse dans la modélisation, et a fortiori dans la sémantique d'optimisation.

L'équipe d'optimisation a de fait mené de front plusieurs travaux : traduction, modélisation et heuristique, tout en maintenant un objectif de genericité, chacun des clients bénéficiant des mêmes possibilités offertes par le produit. Nous examinerons dans ce rapport plusieurs fonctionnalités introduites dans les « versions » successives du produit et leur impact sur la modélisation mathématique.

# Chapitre 1 : Paramétrage de TempoPlanner

## I. Schéma général

TempoPlanner permet de modéliser les différents éléments de l'entreprise qui entrent en ligne de compte dans les procédures d'organisation du temps de travail. Ainsi, on retrouve schématiquement :



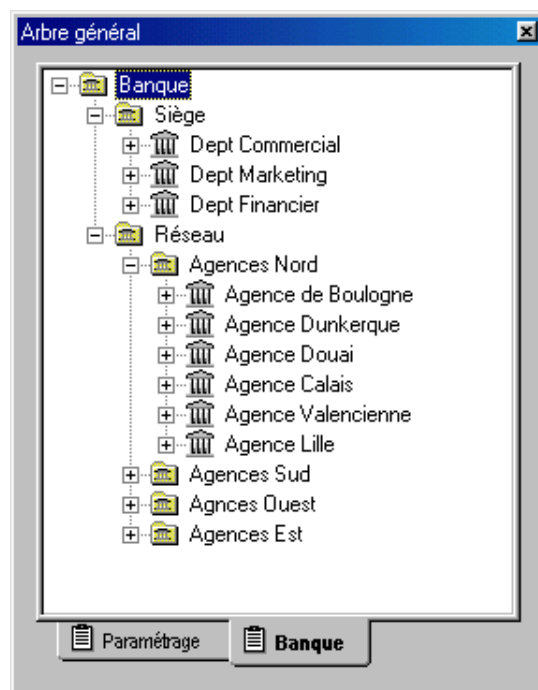
## II. activités – Les charges de travail

### A) L'organisation générale de l'entreprise

Il est possible de décrire la hiérarchie de l'ensemble des sites géographiques et domaines fonctionnels (régions, départements, groupements, agences). Les feuilles de cette hiérarchie sont les domaines de planification auxquels sont rattachés les prévisions de charges et les équipes planifiées.

Dans l'interface utilisateur, cette hiérarchie prend naturellement la forme d'un arbre dans lequel il est possible de déplacer, ajouter ou supprimer des éléments à l'aide de la souris. Il est possible de créer des raccourcis pour accéder directement aux éléments fréquemment utilisés.

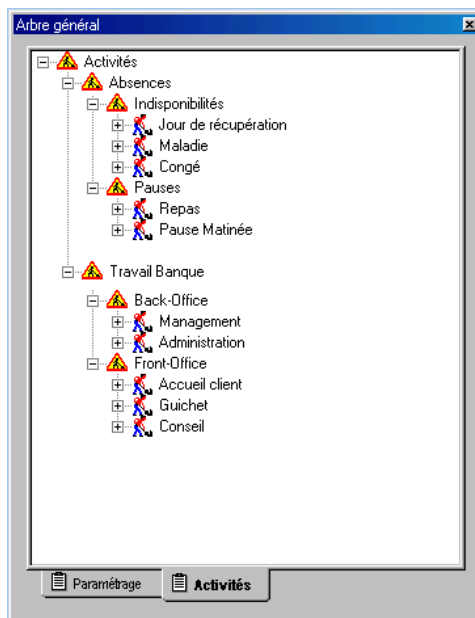
A chaque niveau de cette hiérarchie, il est possible d'associer un calendrier précisant les décompositions de l'année en sous-périodes (premier jour de la semaine, périodes de paie...), les dates de certains types de jour particuliers (week-end, jours fériés, Noël, jours chômés par l'entreprise...) et les bornes temporelles (début des heures nocturnes) qui sont nécessaires au calcul de certains compteurs.



## **B) Les activités et les propriétés des activités**

Les activités :

Les types d'activité que chaque employé peut avoir durant sa journée sont décrits sous la forme d'une hiérarchie. On pourra décrire aussi bien les différentes activités faisant partie de la journée de travail que les activités annexes comme le repas ou la pause.



### *Description des différents types d'activités d'une agence bancaire*

Suivant le besoin de l'entreprise, les types d'activité peuvent être décrits de façon plus ou moins fine. Dans certains cas, il suffira de décrire les différents postes sans détailler les activités au sein de ces postes.

Contraintes de succession des activités :

Il est possible de spécifier les types d'activité pouvant précéder ou suivre une activité donnée ainsi que les activités pouvant être en début et en fin de journée de travail. Ceci permet de décrire un grand nombre de contraintes liées au fonctionnement de l'entreprise ou imposées par les outils de production utilisés.

### Propriétés des activités :

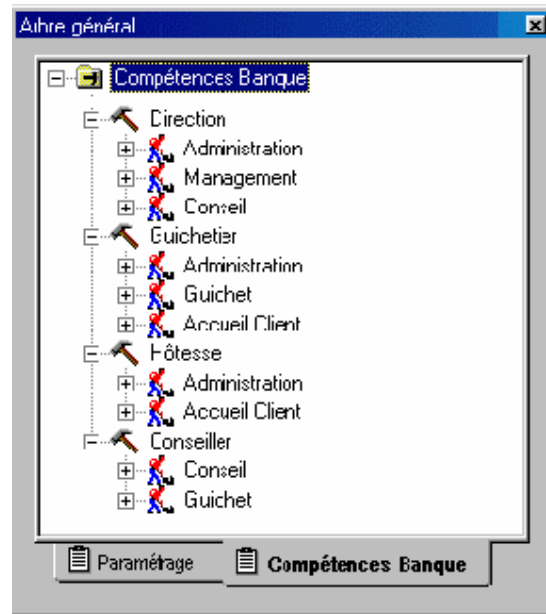
Parmi les informations relatives au site, outre l'équipe et la charge (voir ci-dessous), l'utilisateur peut préciser des propriétés pour chaque type d'activité. Ces propriétés doivent avoir une valeur par défaut indépendante du site, mais ces valeurs peuvent être modifiées au niveau de chaque site. Nous en faisons ici une liste non exhaustive :

- Durée minimale, maximale ou exacte d'une plage de temps continue consacrée à un type d'activité.  
Cette contrainte permet de forcer la durée maximale d'une plage sans pause repas, la durée minimum et maximum de la pause repas, la durée minimum d'affectation d'un agent à une activité de type inventaire...
- Nombre maximum de plages non contiguës sur un type d'activité dans un jour.  
Il est ainsi possible de limiter le nombre de repas ou de plage affectée à telle activité.
- Nombre maximal de personnes affectées simultanément à un type d'activité donné. Ceci permettra de tenir compte de certaines limitations en ressources secondaires. Par exemple, si une activité nécessite l'accès à un poste de travail particulier unique, on précisera qu'au plus une personne peut faire cette activité à un moment donné.
- Durée minimale entre deux plages affectées au même type d'activité.
- Pourcentage minimum de personnes affectées à une activité donnée ayant une compétence donnée. Cette contrainte permettra par exemple d'imposer que pour l'activité de vente au moins 30% des vendeurs aient la compétence pour conseiller les clients sur certains produits complexes (par exemple la micro-informatique).

### **C) Les fonctions, les métiers, les postes de travail**

Outre la notion clé d'activité, il est possible de définir et d'utiliser d'autres notions comme celles de fonctions, de métiers ou de postes de travail. Il s'agit généralement d'un ensemble d'activités pouvant être réalisées par la personne titulaire de ce métier ou de ce poste de travail, sachant que l'on peut aussi faire une distinction entre les activités qui sont faites de façon habituelle, et celles qui sont faites exceptionnellement. On peut aussi

spécifier que telle personne qui occupe telle fonction occupe tant de son temps à faire telle ou telle activité.



*Les différents métiers ou postes de travail*

## **D) Les charges de travail**

### ➤ **Différents types d'activités :**

Les types d'activité diffèrent en ce qui concerne la nature de la charge associée. Sur ce point, le logiciel peut gérer cinq natures différentes d'activités :

- **Sans Charge** : correspond aux activités pour lesquelles il n'existe pas de charge (comme les repas ou les pauses).
- **Fixes** : correspond aux activités qui ne sont pas mobiles dans le temps comme l'accueil des clients. La charge est exprimée en nombre de personnes devant être affectées à l'activité à un moment donné ou en nombre d'unités d'œuvre traitées pendant le pas de temps choisi.
- **Mobiles** : pour les activités qui peuvent être déplacées au sein de plages horaires, d'un jour ou même au sein d'une semaine comme les tâches d'encadrement. Par exemple, « il me faut 10 hommes-heures entre 5h et 7h du matin, et c'est au logiciel de donner la répartition optimale sur chaque tranche horaire ».
- **Insécables** : pour les activités qui doivent être faites par la même personne pendant toute leur durée. Typiquement pour les tournées...

- **Composées et insécables** : il est possible enfin que les activités soient en fait composées d'une succession d'activités insécables qui doivent être faites dans un ordre donné et par la même personne. Typiquement, une tournée peut se décomposer en plusieurs étapes : la préparation de la tournée, la tournée en elle-même (qui peut là encore se découpler en livraison, collecte...) et le retour de tournée

Cet échantillon de type de charge permet de modéliser un grand nombre d'activités rencontrées dans les entreprises.

### ➤ **Description des charges globalement ou par activités :**

Il est possible de décrire une charge globale exprimée dans une unité quelconque (chiffre d'affaire, unité d'œuvre...).

A cette charge globale correspondent des règles de calcul qui permettent de déterminer les effectifs requis pour couvrir cette charge. Ainsi, par exemple, pour un niveau de chiffre d'affaire, la direction de l'entreprise sait qu'il faut un certain type d'organisation, correspondant à une répartition des effectifs : tant de vendeurs, tant de personnes à l'accueil, tant de personnes aux caisses...

Il est aussi possible de décrire plus finement la charge par activités (description locale de la charge). Pour un domaine donné (par exemple un magasin) et une activité donnée (par exemple accueil ou service client ou réapprovisionnement des rayons), l'utilisateur peut maintenir une courbe de charge pour chaque jour (ou chaque semaine) de l'horizon de planification.

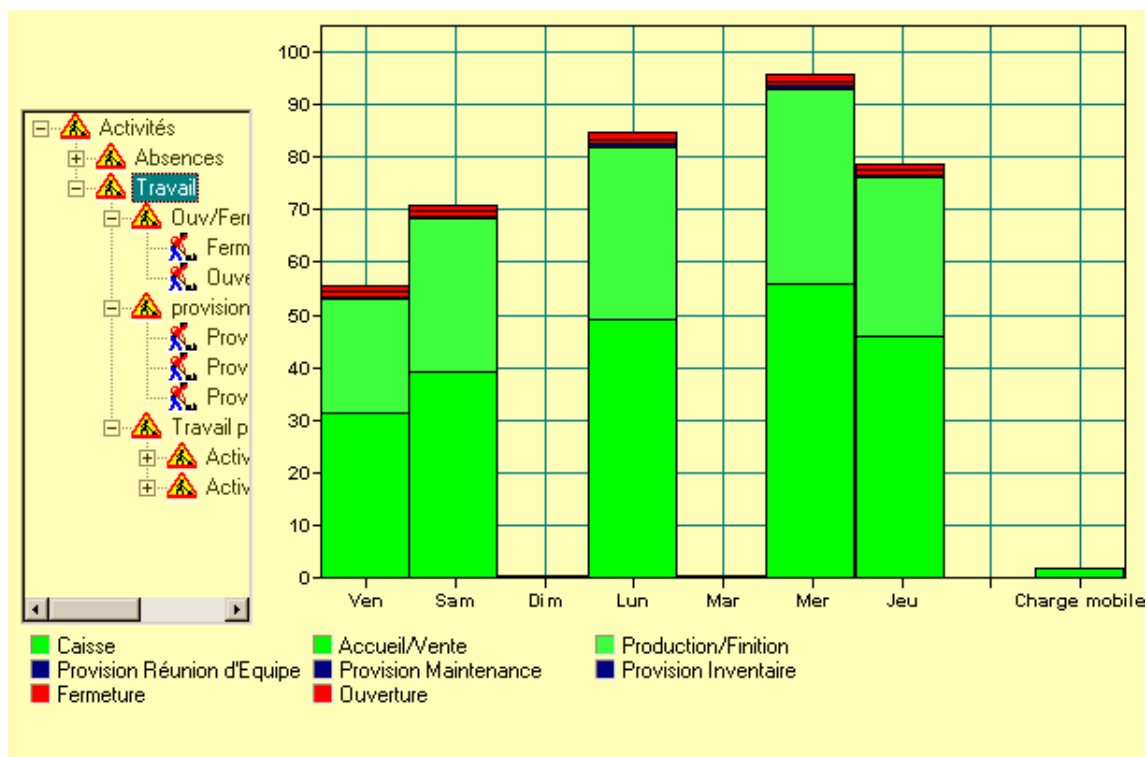
Une courbe de charge est composée d'une valeur par tranche de temps, la tranche de temps étant fixée librement (5 minutes, quart d'heure, demi-heure, heure, jour ou semaine ou encore plages au sein d'une journée définie par l'utilisateur).

La nature précise de la courbe de charge varie suivant que le type d'activité est fixe, mobile, insécable... Ainsi, pour les activités fixes, l'utilisateur précise le nombre de personnes qui doivent être affectées à l'activité durant une tranche de temps, alors que pour les activités mobiles, il donne un nombre d'unités de l'activité qui doivent être effectuées durant la tranche de temps.

### ➤ **Profils de charges :**

Enfin, l'utilisateur décrit des profils de charge correspondant à autant de jours types qu'il souhaite simuler ou planifier. Ces profils de charges sont construits soit directement soit par combinaison de courbes de charge diverses (somme, multiplication) et de périodicités différentes. Par exemple, il est possible de définir une charge comme étant le résultat de la combinaison d'une courbe de variation sur l'année (pas de temps semaine) avec une charge type sur la semaine (pas de temps du quart d'heure). Enfin, une courbe de charge peut être déduite d'une autre (par exemple le chiffre d'affaire escompté d'un magasin) à l'aide de règles donnant le nombre de personnes nécessaires sur l'activité pour un intervalle donné de la charge initiale. Ceci permet de tenir compte des économies d'échelle éventuelles.

Une commande permet d'importer des courbes de charges venant d'applications différentes. Ceci permet par exemple d'importer les charges gérées par d'éventuels outils développés en interne par les clients de TempoSoft.



*Profil hebdomadaire de charge des activités du groupe travail*

### III. Les Ressources Humaines

#### A) Description des ressources humaines

Chaque salarié est affecté à un groupe de salariés lui-même rattachée à un domaine. Cette affectation peut changer dans le temps. L'identité de chaque salarié peut être décrit à l'aide d'informations variées.



*Arborescence des ressources d'une banque*

Il faut préciser pour chacun d'eux :

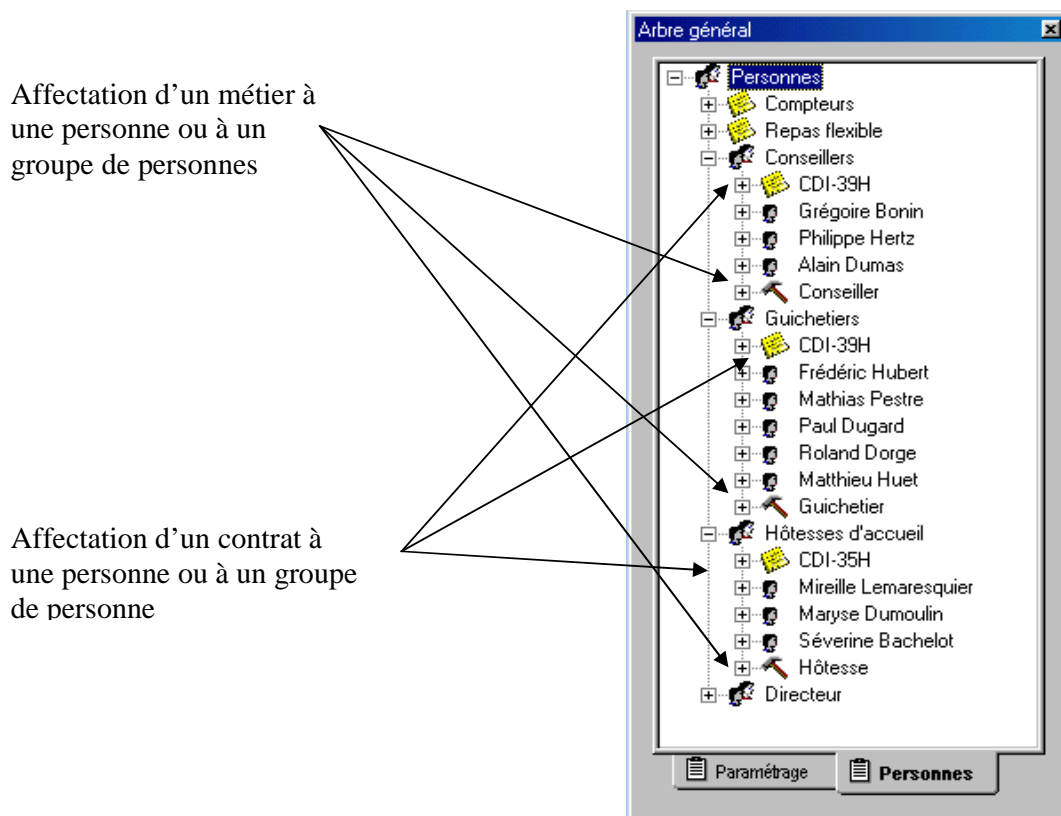
le type de contrat de travail qui le concerne, c'est à dire l'ensemble des contraintes qui doivent être vérifiées lors de la constitution de son planning ainsi que les règles d'évaluation des différents compteurs comme les heures supplémentaires ou le solde de modulation (cf. ci-dessous la gestion des contrats)

la « fonction » du salarié, c'est à dire l'ensemble des activités que, d'après son métier, il peut effectuer de façon régulière ou exceptionnelle (on précise pour chaque activité que le salarié peut effectuer, la priorité d'affectation de cette activité au salarié par

un nombre entre 1 et 100 qui peut être interprété comme le pourcentage de son temps de travail normalement consacré à cette activité), optionnellement, les compétences ou aptitudes du salarié.

Les salariés peuvent être groupés en équipe. Les informations communes à tous les salariés d'une équipe peuvent être alors saisies au niveau de l'équipe. Il peut y avoir des contraintes de planification au niveau d'une équipe. On peut par exemple imposer que tous les membres d'une équipe ait le même planning.

Toutes ces opérations s'effectuent par des 'glisser-déplacer' depuis l'endroit où sont « stockés » les contrats et les métiers possibles et l'arbre décrivant les salariés et les équipes.



## B) Les préférences des salariés

Chaque employé peut définir sur la semaine type un ensemble de plages horaires sur lesquelles il ne souhaite vraiment pas travailler, ainsi qu'un ensemble de plages où il préfère ne pas travailler. Ces plages sont saisies graphiquement (rouge, orange, vert). Ces préférences sont alors prises en compte sur l'ensemble des semaines planifiées.

Un employé peut également exprimer des préférences plus complexes comme le souhait d'avoir le plus souvent possible « n » jours de repos d'affilés ou d'avoir des semaines de travail toutes identiques (longueur maximum du cycle), ou encore d'avoir tel modèle de jour, un jour de semaine donné.

A chaque préférence est associé un niveau de priorité. Un compteur spécial de respect des préférences est conservé par le système afin de permettre un équilibrage entre les employés en ce qui concerne la satisfaction de leur préférence sur de longues périodes (objectif d'équité).

## C) Gestion prévisionnelle des indisponibilités

Les jours de congés, de récupération ou de maladie connus à l'avance peuvent être saisis de façon graphique dans l'écran d'édition du planning. De même, les activités non productives comme les formations pourront être placées dans le temps. Ces données seront prises en compte lors de la construction du planning.

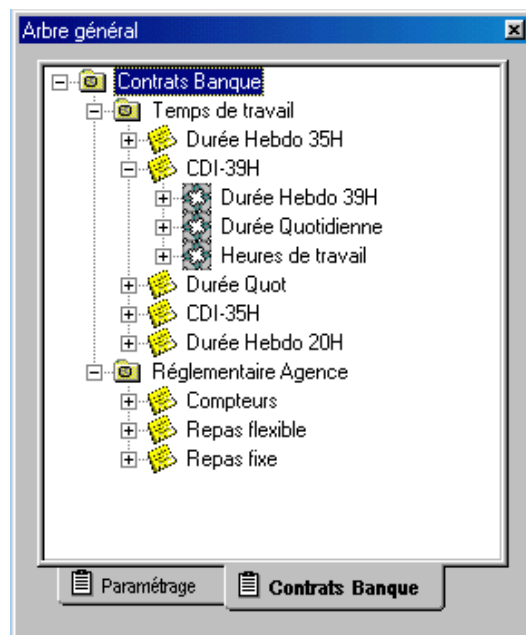
Personnes	Déc 14, '98			Déc 21, '98					Déc 28, '98						
	V	S	D	L	M	M	J	V	S	D	L	M	M	J	
Simon				Récupération											
Louis															
Isabelle															
Stéphanie					Formation										
Elise															
Pierre								Vacance							
Sylvie															
Carole	Maladie														
Chantal															
Jean-Marc															

## IV. Les contrats et la réglementation

### A) Organisation des règles en contrat

L'ensemble des informations réglementaires et contractuelles concernant un employé est regroupé dans un "contrat". Un même contrat peut être partagé par de nombreux employés. Un contrat peut être défini comme la modification d'un autre contrat. Le système gère donc une hiérarchie de contrats pouvant être partagés à travers de nombreux sites ou spécifiques à certains sites ou même à certaines personnes.

L'ensemble complet des informations contenues dans un contrat est défini de façon précise lors du paramétrage du logiciel pour chaque organisation. Cette souplesse permet de s'adapter facilement aux éventuelles évolutions du mode d'organisation du travail.



*Différents contrats établis pour une agence bancaire*

### B) Le langage de paramétrage

Dans un usage quotidien, l'utilisateur peut facilement fixer l'ensemble des contraintes réglementaires devant être considérées. Il lui suffit d'affecter un contrat à un salarié et de définir de nouveaux contrats par simple héritage d'un contrat prédéfini et modification de certaines constantes associées.

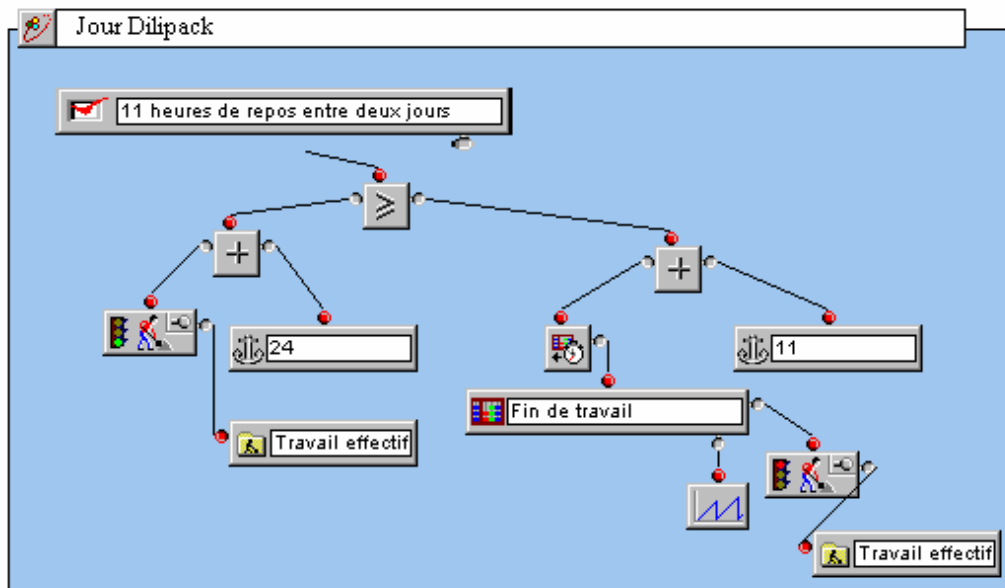
En revanche, lors de la première installation ou à la suite de modification structurelle des règles sur l'organisation du temps de travail, il est nécessaire d'ajuster le paramétrage.

On peut alors définir de nouveaux compteurs et/ou de nouvelles constantes ainsi que les contraintes associées. Par exemple, si l'entreprise introduit un système de compte épargne temps, il pourra être géré avec TempoPlanner après avoir défini le compteur associé.

Les contraintes et les compteurs sont définis à l'aide d'un langage puissant saisi sous forme graphique. Un éditeur spécialisé assiste l'utilisateur lors de la construction des contraintes et expressions en lui proposant à chaque niveau l'ensemble des possibilités offertes. Ce langage fait référence à des constantes et à des expressions définies dans le contrat qui peuvent être modifiées lors de l'héritage entre contrats. Il peut également faire référence à des constantes temporelles ou à des périodes définies dans le calendrier du domaine. Ceci permet d'écrire des contraintes valables pour un large ensemble de personnes, le détail de celles-ci étant fixé en fonction du contrat précis de l'employé et du domaine auquel il est rattaché.

A l'aide de ce langage de paramétrage, l'utilisateur est capable de modéliser tout type de contrainte liée au temps de travail. Citons par exemple :

- amplitude maximum de la journée de travail
- nombre maximum d'heures travaillées dans la journée
- temps de repos minimum entre deux journées de travail
- nombre maximum d'heures travaillées dans une semaine civile
- durée hebdomadaire de travail contractuel
- nombre minimum de jours de repos dans une semaine civile
- nombre maximum de samedis travaillés sur un nombre donné de semaines
- nombre minimum de suite de « p » repos (contenant un dimanche) sur une suite de « n » semaines
- durée maximale (ou proportion de temps maximale) affectée à un type d'activité dans une semaine ou dans un jour...



*règle définie à l'aide du langage de paramétrage*

La règle précédente traduit la contrainte classique en droit français selon laquelle un temps de repos minimal de 11 heures consécutives par périodes de 24 heures doit être assuré à tout salarié.

La souplesse et la puissance de ce langage de paramétrage permet donc de modéliser toutes les contraintes classiques de la réglementation du travail, mais en plus de tenir compte de n'importe quel type de contrainte locale, liée à l'entreprise, à l'établissement. L'utilisateur n'est absolument pas limité dans la définition des contraintes.

### **C) Les compteurs**

#### **➤ Définition :**

De nombreuses contraintes prises en compte lors de la construction du planning peuvent être exprimées à l'aide de compteurs à travers lesquels il est possible d'imposer pour chaque personne un nombre minimum et un nombre maximum d'occurrences des éléments comptés sur l'ensemble de l'année comme sur des sous périodes de l'année.

L'ensemble des compteurs utilisés par l'application est librement défini lors du paramétrage. Voici par exemple un ensemble de compteurs possibles :

- nombre de jours de congés (pris, à prendre...)
- nombre d'heures travaillées
- nombre de jours travaillés (un jour de semaine j)
- nombre de nocturnes (journées qui sont travaillées après une heure donnée)
- nombre de semaines ayant  $n$  jours travaillés
- nombre de séquences de plus de  $n$  jours travaillés consécutivement
- nombre de séquences de deux jours de repos consécutifs (contenant le dimanche)

Ces compteurs permettent en particulier d'exprimer la spécificité des contrats passés avec chaque personne. En effet ceux-ci correspondent à des bornes sur la valeur de ces compteurs pour une année entière ainsi qu'éventuellement sur des sous-périodes de l'année.

Les valeurs prises par ces compteurs en début de l'horizon de planification constituent l'une des données essentielles de la planification.

L'utilisateur peut de plus préciser la valeur idéale d'un compteur pour une personne à une date donnée. Un des objectifs du système est alors de se rapprocher autant que possible de cette valeur idéale. De plus, l'utilisateur peut imposer des bornes sur la valeur de ces compteurs à une date donnée.

➤ **Les valeurs - cibles pour les compteurs :**

Un des objectifs de la planification peut être de limiter le nombre d'heures supplémentaire inférieur à un seuil donné ou encore d'amener le compteur de modulation annuelle à zéro en fin d'année ou à une valeur donnée à une certaine date. De tels objectifs peuvent être exprimés par l'utilisateur qui pourra saisir pour un salarié ou un groupe de salariés, un compteur et une date donnée soit la valeur souhaitée de ce compteur à cette date, soit la valeur minimale ou maximale souhaitée à cette date. Une pondération (entre 1 et 100) de l'importance de ces cibles les unes par rapport aux autres peut être précisée.

Lors de la construction du planning, le logiciel essaiera de respecter ces cibles (le compromis avec les autres objectifs de la planification qui peuvent entrer en conflit avec ces cibles étant ajuster par l'utilisateur lors du lancement du calcul).

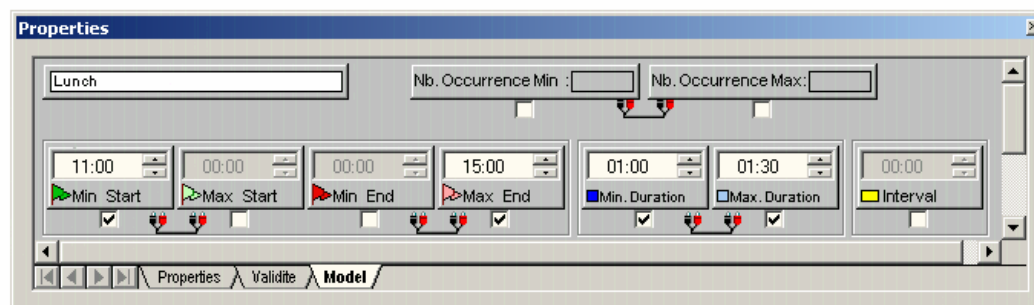
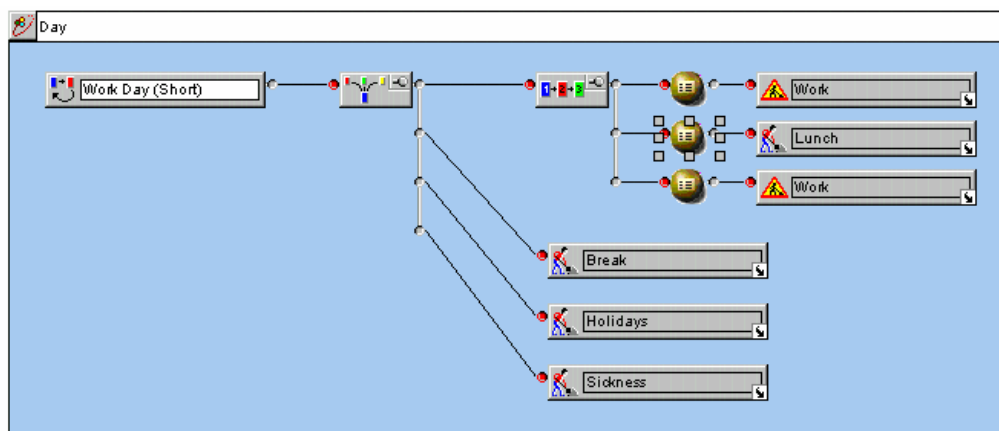
**D) Les modèles (briques de planification)**

➤ **Modèles de jour :**

La brique élémentaire de construction des plannings est souvent le modèle de jour. Un modèle de jour est une séquence de plages pour lesquelles on précise :

- l'heure de début au plus tôt et au plus tard
- l'heure de fin au plus tôt et au plus tard
- la durée minimale et maximale
- le groupe des activités pouvant être effectuées sur cette plage

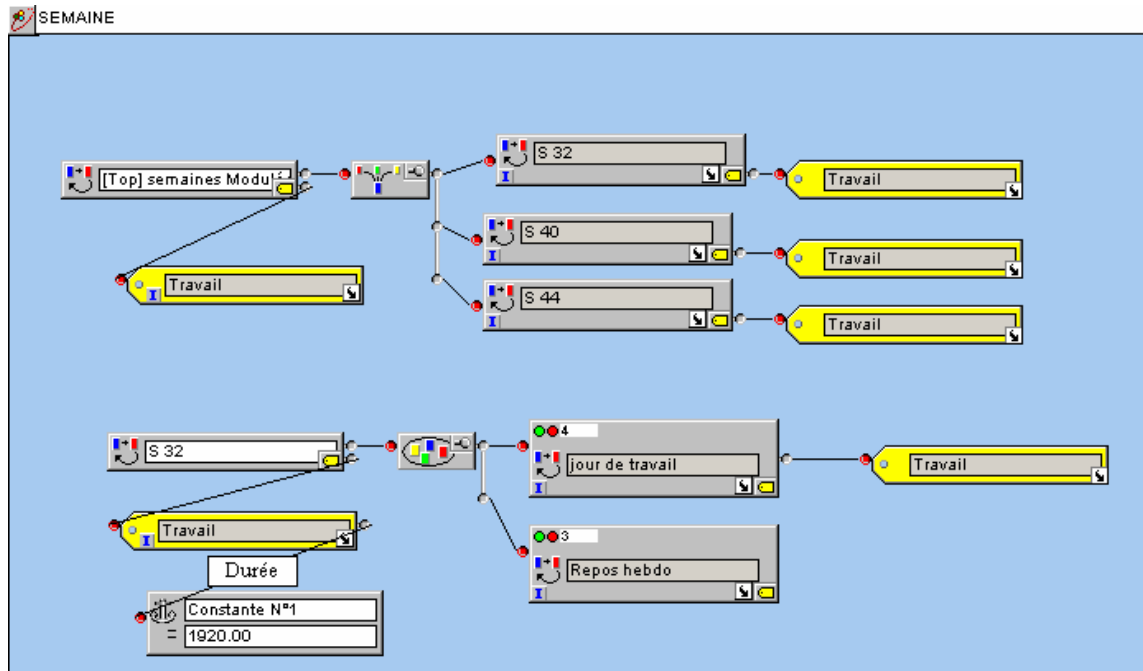
L'utilisateur peut décrire de nombreux modèles de jour qui pourront servir soit pour construire des modèles de semaine soit directement pour préciser les horaires que chaque employé peut effectuer. La journée de repos est un modèle de jour particulier.



*Exemple de modèle de jour*

➤ **Modèles de semaine et de cycle :**

Pour certaines personnes qui souhaitent une certaine stabilité de leur planning ou un enchaînement très spécifique de jours de repos, il est souhaitable d'imposer au logiciel de composer le planning à partir de modèles de semaine ou de cycle prédéfinis plutôt qu'à partir de seulement des modèles de jour.



*exemple de modèles de semaine*

Dans le schéma ci-dessus, le modèle de semaine S 32 est un ensemble non ordonné d'exactly 4 modèles « jour de travail » et 3 modèles « Repos Hebdo ». Les **étiquettes** Travail (jaunes) indiquent que dans chacun des modèles ainsi étiquetés, il existe un compteur de la durée d'activités d'un groupe d'activités. Ce groupe est défini par les « activités-feuilles » des chemins . Cette notion permet de poser des contraintes d'amplitude sur certains modèles en spécifiant le groupe d'activités visé. Ainsi, dans le schéma ci-dessous, on a indiqué que dans le modèle S 32, la durée du groupe travail (excluant par exemple les pauses repas) devaient être de 1920 minutes (soit 32 heures).

# Chapitre 2 : L'Optimisation dans TempoPlanner

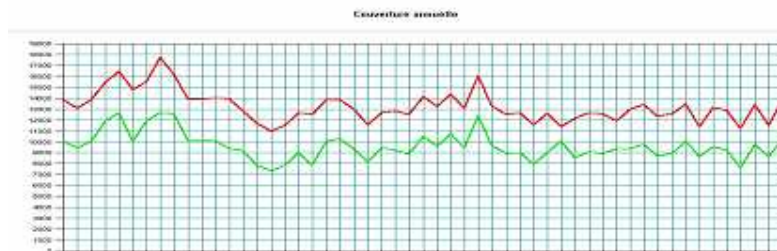
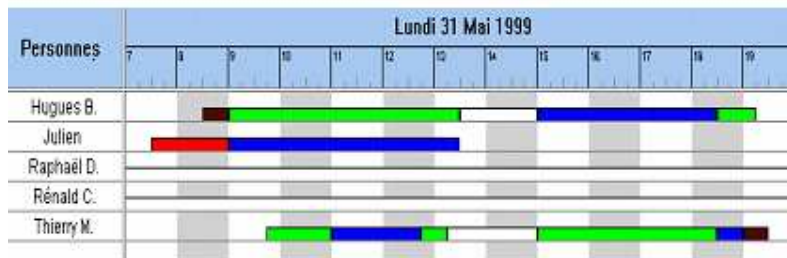
## I. Principes généraux de l'optimisation

### A) Génération de planning

TempoPlanner construit automatiquement un planning « optimal » sur l'horizon de temps (ou *horizon de planification*) choisi par l'utilisateur. En cela, il veille à :

- couvrir la charge au plus juste
- respecter les modèles périodiques
- respecter les contraintes réglementaires
- atteindre les objectifs sur la valeur des compteurs à certaines dates
- respecter les préférences des salariés
- assurer une certaine équité en lissant entre les employés la valeur d'un compteur défini par l'utilisateur

	travail							
Hugues B.	D	JP PR	JP PR	R	JP PR	JP PR	JP PR	R
Julien	D	AM	JP PR	AM	JP PR	JP PR	JP PR	R
Raphaël D.	D	R	JP	PM	JP	PM	JP	R
Rénauld C.	D	R	R	R	R	R	R	R
Thierry M.	D	JP PR	PM	JP PR	R	JP PR	JP PR	R



*patterns de modèles, placement d'activités et couverture de charge*

## **B) L'Optimisation : Arbitrage entre objectifs contradictoires**

Avant de lancer le moteur d'optimisation, l'utilisateur a la possibilité de définir les priorités des contraintes et des objectifs :

- priorité des différentes charges entre elles : par exemple, la priorité de la couverture de la charge d'une activité de vente peut être jugée plus importante que celle de l'activité administration
- priorité des contraintes réglementaires, entre elles et avec le respect des charges de travail
- priorités des préférences personnelles

## **C) Généricité**

La multiplicité d'objectifs contradictoires ne permet pas de définir *a priori* une « sémantique » du problème d'optimisation. En effet, même si on peut distinguer des grandes classes de contraintes/objectifs, leur nombre, portée, pondérations et les objets sur lesquels ils portent (par exemple les courbes de charge d'activités) seront connus à la lecture du scénario. Le poids respectif de ces classes est donc impossible à évaluer, à moins d'un traitement pré-optimisation coûteux et flou. Les algorithmes mis en œuvre par TEMPOSOFT s'adaptent à cette généricité imposée, en exploitant, indépendamment ou en coopération, plusieurs méthodes d'optimisation.

## **D) Rappel sur les techniques d'optimisation combinatoire**

L'optimisation des horaires du personnel est un problème complexe dont la résolution nécessite de maîtriser un grand nombre de règles et préférences sur les emplois du temps de chacun. Il ne peut être résolu de façon réellement satisfaisante sans mettre en œuvre des techniques d'optimisation performantes. Les méthodes d'optimisation habituellement utilisées sur ce type de problème sont les suivantes :

### ➤ **Méthodes heuristiques :**

Ces méthodes construisent une solution en prenant une décision après l'autre sans les remettre en cause. Le défaut de telles méthodes est que, quelle que soit la sophistication des règles pilotant les prises de décision, des mauvais choix sont effectués de temps à autres, ce qui dégrade sensiblement la qualité du planning construit. De plus, ces méthodes sont fragiles et difficiles à adapter à chaque cas particulier. Leur avantage est de fournir une solution dans des temps calculs limités.

### ➤ **Méthodes d'amélioration locale :**

Ces méthodes partent d'une solution et cherchent à la modifier ponctuellement afin de l'améliorer. Le plus souvent développées sans outil spécialisé, ces méthodes sont difficiles à maintenir. De plus, elles se heurtent vite à l'existence de minima locaux dans lesquels la recherche se trouve bloquée. Ce défaut est souvent contourné par des méta-heuristiques d'échappement des minima locaux comme le recuit-simulé ou la méthode tabou (ou encore des approches de type algorithme génétique). Toutefois, celles-ci ont un coût élevé en temps calcul et ne garantissent pas la découverte de l'optimum.

➤ **Programmation linéaire mixte :**

Cette méthode est complète dans le sens qu'elle garantit la découverte de la solution optimale si la recherche n'est pas interrompue. En pratique toutefois, le temps nécessaire à la détection de la solution optimale devient vite rédhibitoire (même pour des problèmes de petite taille). La programmation linéaire est néanmoins intéressante car capable de trouver la solution optimale rapidement sur certains sous-problèmes simples comme l'affectation des vacances à une équipe sur une journée isolée.

➤ **Programmation par contrainte (PPC) :**

Issue des travaux en programmation logique, les premières implémentations ont été réalisées dans les années 1970 (A. Colmerauer), avec le langage PROLOG. Les éditeurs les plus connus sont Cosytech (langage Chip), Prologia (PrologIII) et ILOG, dont les premiers moteurs conçus en LISP ont évolué vers une suite de composants C++ dédiés à l'optimisation combinatoire. En PPC, le programmeur définit un ensemble de variables liées entre elles par des contraintes, et c'est le rôle du système de satisfaction de contraintes de trouver la ou les solutions correspondantes.

La PPC prend toute sa valeur dans des problèmes d'optimisation complexe lorsqu'elle dépasse l'aspect « énumération intelligente » pour s'allier à des classes d'algorithmes de la Recherche Opérationnelle : recherche arborescente du type Branch and Bound, Programmation linéaire, recherche locale. On citera pour l'exemple la contrainte « tous différents », qui peut se représenter comme un couplage maximum du graphe biparti où sont représentées d'un côté les variables et de l'autre l'union des domaines de ces variables (un arc reliant une variable à l'un des éléments de son domaine). Dans ce cas, les mécanismes de propagation de la contrainte « Tous différents » sont directement influencés par les informations relatives au couplage.

## **E) La Programmation par contraintes à l'aide des outils ILOG**

La complexité et généralité des problèmes d'affectation auxquelles s'attaque TEMPOSOFT a incité dès sa création à faire le choix d'une coopération « dense » PPC/RO. Les outils ILOG ont semblé les plus à même d'y répondre et de s'articuler facilement avec l'architecture logicielle du produit, notamment **ILOG Solver** (moteur de propagation de contraintes) et **ILOG Planner** (code de Programmation Linéaire partageant – si besoin – ses variables avec celles de Solver). Ces outils ont été justement conçus pour faciliter les approches hybrides en fournissant des concepts de programmation de haut niveau, et des algorithmes de propagation de contraintes et de programmation linéaire efficaces. Ainsi, des approches hybrides ont été déjà développées avec succès avec les outils d'ILOG dans de nombreux domaines allant de l'optimisation de portefeuille jusqu'à la planification de production en passant par la planification de personnel. Depuis peu (été 1997), cette technologie est arrivée à sa maturité avec l'intégration de l'outil **CPLEX** dans la gamme des outils de programmation par contraintes d'ILOG.

TEMPOSOFT utilise également à une moindre échelle la bibliothèque **ILOG SCHEDULER**, un ensemble de composants dédiés aux problèmes d'ordonnancement/scheduling/affectation.

Ci-dessous sont décrits les concepts fondamentaux de l'implémentation de problèmes de PPC avec les outils ILOG :

- Classe IlcManager : une instance de la classe IlcManager, appelée Manager, gère l'ensemble des variables et contraintes de Solver, les entrées et sorties, les allocations mémoire du programme. Il est possible de créer autant d'instance de IlcManager que l'on souhaite pour un programme Solver.
- Classe LinOpt : une instance de cette classe gère le Programme Linéaire, i.e le définit (variables et contraintes), lance un algorithme (simplex primal/dual ou méthode de point intérieur). En cas de coopération avec Solver, le linOpt maintient aussi une solution optimale lorsqu'une variable apparaissant dans le programme linéaire est modifiée par une propagation externe
- Classe IlcGoal : une instance de la classe IlcGoal, appelée Goal, gère la propagation et le postage d'un ensemble déterminé de contraintes et restitue les données au Manager du programme Solver.
- Postage d'une contrainte : une contrainte est considérée comme postée à Solver (respectivement Planner) quand elle est ajoutée à un Manager dans le programme (respectivement au LinOpt).
- Propagation d'une contrainte: lorsqu'une contrainte est postée elle est immédiatement propagée, c'est à dire que le domaine de définition des variables qui constituent cette contrainte est réduit à l'ensemble des valeurs pouvant satisfaire la contraintes et la solution du problème.
- BackTrack : le postage et la propagation d'une contrainte ne sont pas irréversibles, c'est à dire que l'on peut récupérer les données initiales de la

contrainte par le phénomène de Backtrack. A noter que lorsqu'un goal échoue, toutes les contraintes ayant été propagées à l'intérieur de ce goal sont automatiquement 'backtrackées'.

## F) Options d'une optimisation

Chaque optimisation est définie à sa création par un ensemble de priorités pouvant être modifiées par la suite. La fenêtre ci-dessous montre la fenêtre de propriétés « Date et Type » de l'optimisation *Année*. Dans cette fenêtre d'options sont saisies comme on peut le voir la date de début, le type de l'optimisation, la portée externe (l'horizon de planification) et la portée interne, appelée également la *maille*, soit la précision du calcul.

The image shows a software dialog box titled "Optimisation" with a close button (X) in the top right corner. It has four tabs: "Date et Type", "Plans", "Options avancées", and "Script". The "Date et Type" tab is active. Inside the dialog, there is a text field for "Nom" containing the word "Année". Below this, there are two main sections. The first section is "Date de début" and contains three radio buttons: "Absolue" (which is selected), "Aujourd'hui:", and "A la date du calendrier". Next to "Absolue" is a date field showing "03/01/00". Below "Aujourd'hui:" is a numeric field with "0" and a "jour(s)" label. The second section is "Type" and contains four radio buttons: "Globale" (selected), "Détaillée", "Mixte", and "Compteurs". Below these sections are two "Portée" (range) fields. The "Portée externe" field has a numeric spinner set to "52" and a dropdown menu set to "SEMAINE". The "Portée interne" field has a dropdown menu set to "SEMAINE". At the bottom of the dialog are three buttons: "OK", "Annuler", and "Aide".

Fenêtre de propriétés de l'optimisation *Année*

L'onglet plans définit la liste des **plans en entrée** de l'optimisation et le **plan de sortie** de celle-ci. Ces notions renvoient à des définitions de compteurs différentes et qui peuvent être stockées dans des champs différents dans la base de données. Le principe est de lire pour un compteur sa valeur si elle existe dans l'ordre des plans d'entrée, sinon on utilisera l'expression du compteur définie par défaut ou dans son plan de sortie.

L'onglet 'Script' contient quant à lui un script Visual Basic qui s'exécute au début du lancement de chaque optimisation. Ce script permet de lancer par exemple des enchaînements d'une même optimisation mais répétée sur 52 semaines.

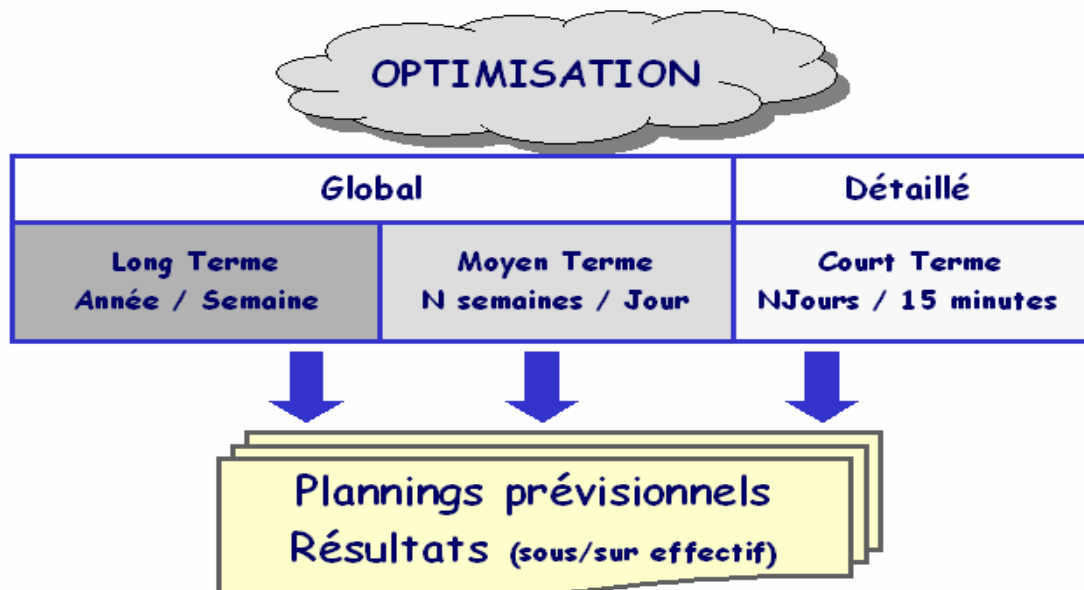
## II. Grandes lignes de l'optimisation

Les deux principales méthodes accessibles à partir de l'objet d'optimisation vu ci-dessus sont « GLOBALE » et « DÉTAILLÉE ». L'optimisation globale calcule des durées et choisit des modèles tandis que l'optimisation détaillée place les activités au sein de la journée. Ceci correspond à des problématiques différentes, définies au cas par cas avec les clients de TEMPOSOFT.

### ➤ Différences d'objectifs

Ainsi, la génération et la visualisation de plannings automatisés au quart d'heure, éditables et utilisables dans un mode opérationnel par un chef d'équipe, un responsable de rayon, ou un employé correspondent typiquement au scénario d'une optimisation détaillée.

En revanche, simuler un passage aux 35 heures dans un cadre d'annualisation du temps de travail peut être vu comme un problème d'optimisation à une maille beaucoup plus large : il faut générer des durées de travail hebdomadaires individuelles (ou dans le cadre de la modulation des types de modèles de semaine), respectant les contraintes de maille semaine (minimum et maximum horaire hebdomadaire) ou supérieures (nombre de semaines de congés payés sur l'année – contrainte glissante de succession d'au moins deux semaines de congés pendant l'été).



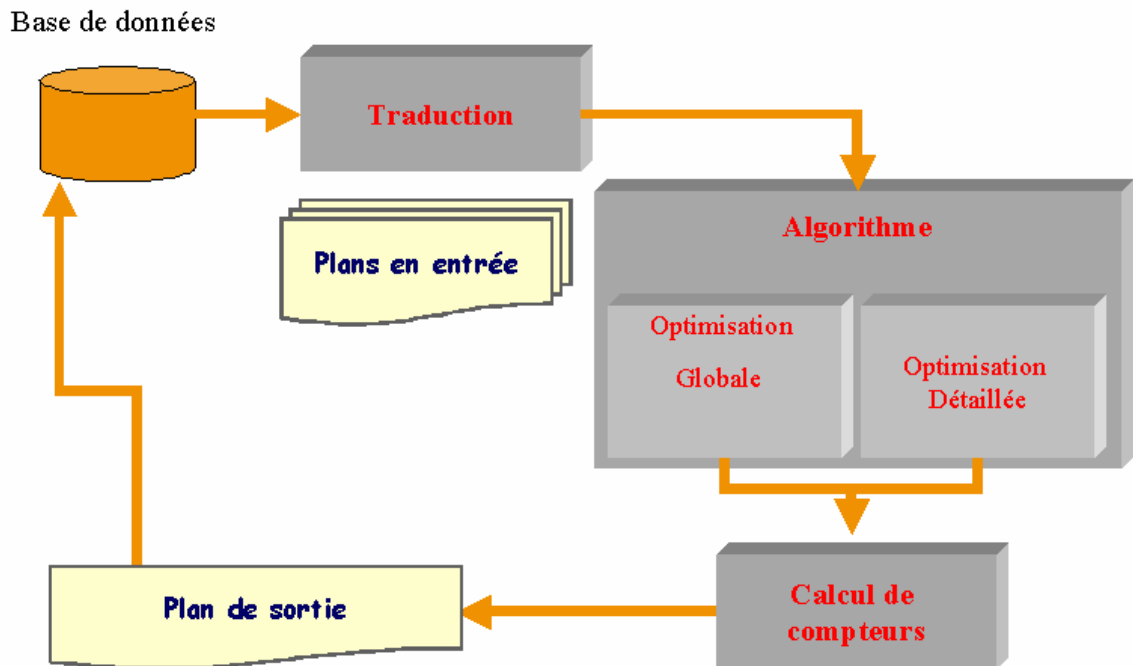
*Notion de plans d'écriture pour différentes optimisation. (horizon/portée interne)*

➤ **communication entre optimisations :**

La notion de plans lecture/écriture permet de communiquer les résultats d'une optimisation en entrée d'une suivante. L'enchaînement d'optimisation de types différents, typiquement globale suivie de détaillée est ainsi rendue aisée, par un partage des données en entrée sortie.

➤ **Le partage de la modélisation :**

Si les algorithmes diffèrent, nous avons outre la communication externe de certaines données entre deux scripts d'optimisation—défini par l'utilisateur— un partage en interne d'un grand nombre de structures : celles de *traduction*, i.e les objets qui permettent de générer le modèle mathématique (variables, contraintes) à partir du *règlementaire* (personnes, activités, contraintes IHM) , les modèles (de semaine ou de jour), traduits de manière similaire, mais dont on exploite les propriétés de manière différente suivant les types d'optimisation.



*schéma du flot entrées/traitement/sorties*

### III. L'Optimisation « Globale », une recherche arborescente originale

L'optimisation dite « Globale » est basée sur un algorithme de recherche arborescente guidée par la programmation linéaire et la propagation de contraintes dites *sous-hypothèses*. Cette notion importante sera examinée par la suite.

#### **A) Construction du modèle mathématique**

La création des variables et des contraintes se fait pour partie dans la phase de traduction du *modèle instancié* - une interface logicielle basée sur l'architecture COM (Component Object Model)- qui permet une lecture orientée objet des objets de la base de données : personnes, activités, charges, contraintes... Plus précisément, on distingue en mode global 4 étapes clés pour la construction du modèle mathématique :

La phase de décomposition en temps et en activités

La création des durées fondamentales et des contraintes de charge

La création des modèles individuels

La création des contraintes et objectifs

##### *A.1 Décomposition temporelle et en groupes d'activités*

Comme nous l'avons indiqué plus haut, l'un des objectifs majeurs de la génération de plannings est de répartir la charge de travail entre les personnes, si possible équitablement, de manière à couvrir au mieux cette charge. L'exigence de précision dans le résultat n'est en revanche pas la même selon que l'on souhaite obtenir des informations sur un horizon d'un an ou sur une journée.

L'utilisateur peut donc définir la granularité temporelle de l'optimisation en fonction de l'horizon de celle-ci. Typiquement, on travaillera à la maille semaine lorsqu'on lance un scénario d'annualisation mais on « descendra » à la maille jour lorsque l'horizon concerne un nombre raisonnable de semaines. En pratique, la phase de traduction repèrera parmi les compteurs de durées d'activité celles indiquées à calculer par l'optimisation pour indiquer que la portée de ces compteurs participera à la décomposition temporelle.

Note : Ce mécanisme permet d'envisager en théorie une décomposition du temps non uniforme.

La décomposition en groupe d'activités consiste à partitionner en groupes maximaux (au sens de l'inclusion) l'ensemble des activités du scénario. Ainsi, si l'ensemble des activités correspondant à du travail n'apparaissent dans le réglementaire (contraintes et compteurs) qu'avec ce groupe, il n'est pas nécessaire de définir une variable durée pour chaque activité : pour chaque période et personne, on aura une seule

variable. La compétence des personnes peut être prise en compte par paramètre, c'est à dire qu'on ne trouvera pas dans un même groupe d'activités deux activités et une personne à la fois compétente pour l'une et incompétente pour l'autre.

Note : le principe de décomposition s'apparente à une réduction du nombre de variables dans un programme linéaire.

### A.2 Durées fondamentales et contraintes de charge

Soit T une instance de la décomposition temporelle, G un groupe d'activités de charge C (définie comme la somme des charges des activités constituant ce groupe). La contrainte de charge correspondante sera :

$$\sum_p X_{G,T,p} + s - s' = C$$

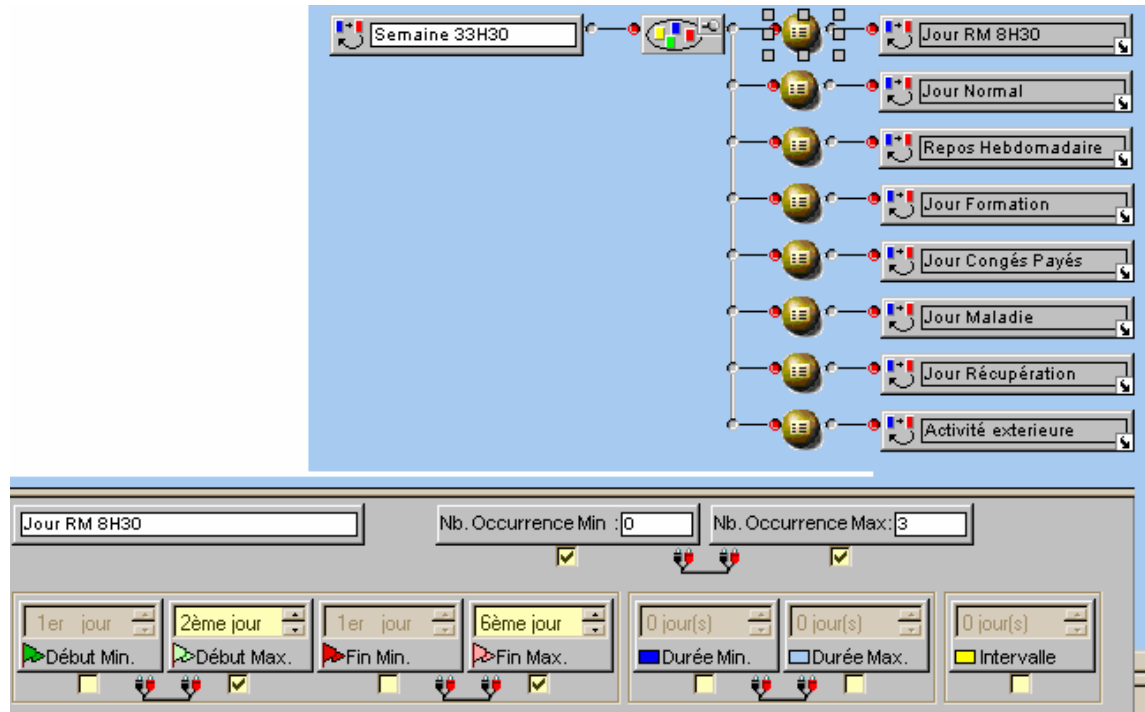
s et s' sont les variables d'écart – positives – indiquant respectivement le sous-effectif et le sureffectif (pour le groupe G sur la portée T). Elles seront directement rajoutées à la fonction de coût.

### A.3 La création des modèles individuels

La traduction de l'arbre des modèles permet d'attacher à chaque modèle rencontré, c'est à dire pour chaque portée et chaque personne :

- sa variable active, un booléen, qui indique si le modèle est actif ou non.
- des variables entières - début, fin nombre d'occurrences d'un modèle au sein d'un modèle de portée supérieure.
- Les variables d'étiquettes (pour les étiquettes cochées)

La filiation et la typologie des modèles (choix, ensemble, suite, tâche) est également renseignée lors de la création des modèles.



début, fin et occurrences variables lors d'un changement de portée

#### A.4 La création des compteurs, contraintes et objectifs

Chacune des nouvelles variables associées est une combinaison des durées fondamentales et des variables actives de modèles. Parmi les contraintes, on distinguera celles définies par l'utilisateur de celles définies en interne, notamment les contraintes de modèles. ces contraintes relient les variables actives de modèles entre elles mais aussi les variables de durées stockées sur chacun des nœuds de modèles. Elles sont implémentées sous la forme de contraintes spécifiques, que nous examinons plus en détail ci-dessous.

### **B) La programmation « sous-hypothèse »**

#### B.1 Principe général

Les modèles ne sont pas implicitement liés à une durée, à part les feuilles de l'arbre des modèles, sur lesquels on a éventuellement des propriétés de durée. « Remonter » les informations de durée est possible sur une branche ; ainsi soit  $X_m$  la durée d'un groupe G de la décomposition sur le modèle m. Si m a pour fils m1 suivi de m2, on pourra écrire :

$$X_m = X_{m1} + X_{m2} \quad (1)$$

Si en revanche, m1 et m2 sont les fils d'un choix, on aura :

$$X_m = X_{m1} \text{ ou } X_{m2} \quad (2)$$

Autant (1) permet telle quelle de propager les bornes (elle est même linéaire donc directement intégrable dans une relaxation linéaire du problème), autant (2) est typiquement non linéaire. Elle peut d'ailleurs se réécrire en intégrant les booléens B1 et B2 des deux modèles fils :

$$X_m = B1.X_{m1} + B2.X_{m2} \quad (2')$$

Lorsque m est potentiellement actif, la seule relaxation linéaire de cette contrainte est de minorer (respectivement majorer)  $X_m$  par la somme des bornes inférieures (respectivement supérieures) des variables  $X_{m1}$  et  $X_{m2}$ . Cette relaxation est bien sûr insuffisante dès lors qu'on aura choisi entre  $m1$  et  $m2$ , où on pourra réécrire (2) en :  $X_m = X_{m1}$  (si l'on choisit  $m1$ ) ; la contrainte redevient plus « forte ».

Le principe des contraintes sous-hypothèse est d'implémenter cette relation entre variables, et propager le maximum d'informations sur des variables liées à des modèles, tant que ceux-ci sont encore potentiellement actifs. Nous avons ainsi défini les HypoVar – couple <variable booléenne, variable (entière ou flottante)> et redéfini les contraintes arithmétiques telles que :

l'égalité entre HypoVar, la somme d'HypoVar, la contrainte « inférieur ou égal ». Ainsi, l'amplitude d'un modèle succession est égal à l' « Hypo somme » des amplitudes des fils.

D'autres contraintes telles que :

choix d'une hypoVar parmi n (ainsi une variable de durée sur un nœud de modèle choix).

Egalité à la première (respectivement dernière) hypoVar active parmi n (égalité de la variable début – resp. de la fin – d'un modèle à la variable début du premier – resp. dernier - fils actif)

« Une HypoVar » parmi n, qui implémente les relations père-fils entre des variables lorsque le père est un choix

## B.2 Implémentation

Nous avons utilisé la possibilité de surcharge des contraintes d'ILOG SOLVER pour implémenter les contraintes sous-hypothèse. Ceci autorise non seulement un héritage – au sens de la programmation objet – des classes de contraintes SOLVER, mais surtout une définition complète des mécanismes de propagation de la contrainte. Ainsi, on peut attacher à une variable utilisée dans la contrainte sous-hypothèse un « démon », objet en mémoire qui se « réveille » dans des conditions que nous déterminons et qui effectue alors une action là aussi prédéfinie.

Prenons par exemple le cas d'une contrainte d'égalité sous-hypothèse :  $(X1, B1) == (X2, B2)$ , B1 et B2 étant les booléens. Une égalité classique entre X1 et X2 implémenterait les deux démons suivants : Propagation du domaine de X1 sur X2 et réciproquement. Ainsi, nous aboutirons à un échec si  $X1 = [2..3]$  et  $X2 = [4..5]$ . Le démon

«  $X_i$  change le domaine de  $X_j$  » est « réveillé » dès qu'il y a une modification du domaine de  $X_i$ .

La contrainte sous-hypothèse implémentera ces deux démons avec une différence sensible dans le cas d'incompatibilité : l'échec se traduit cette-fois par une désactivation du booléen.

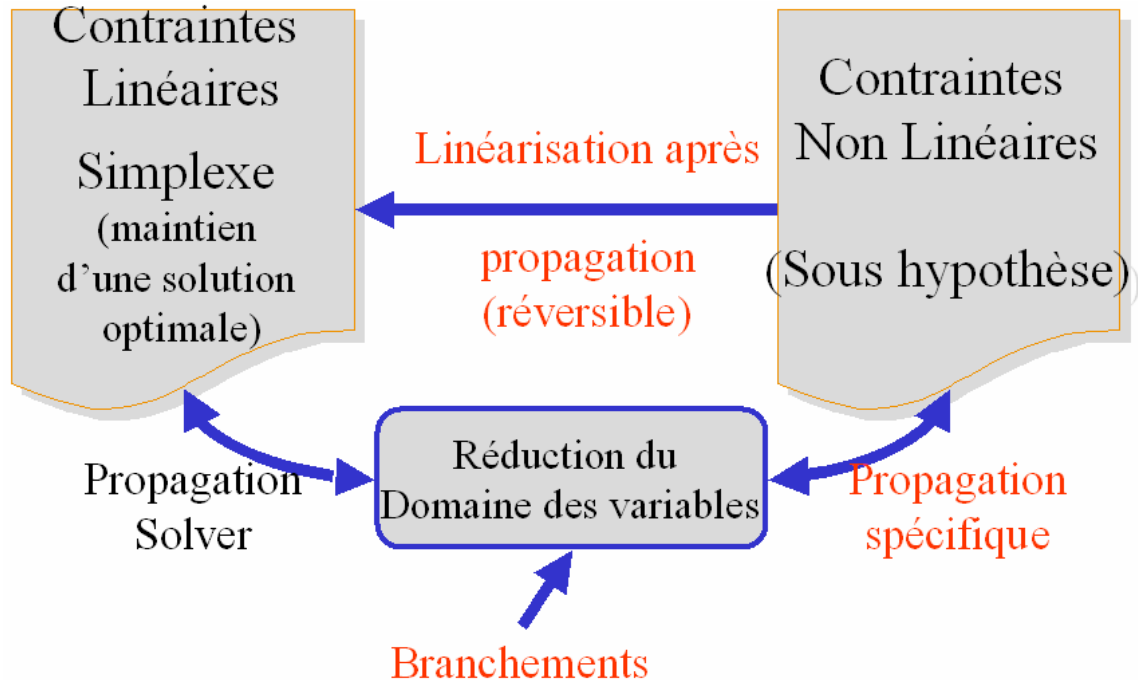
De plus, on introduit un démon supplémentaire par booléen afin d'impacter les modifications de domaine de ces variables. En particulier, si l'un des booléens devient nul, il faudra également annuler l'autre. Toutes les propagations sur  $X_1$  et  $X_2$  seront alors stoppées.

### B.3 Contrainte sous hypothèses et relaxation linéaire

Dans l'exemple étudié précédemment, que se passe-t-il si les deux booléens sont sûrs ? C'est le cas par exemple lorsqu'à l'issue de l'instanciation de variables de modèles (au sein d'un branchement – voir par la suite), un modèle père est sûr et que l'on propage ce choix de modèle sur les fils. Dans le cas d'une succession, les fils partagent la *variable active* (le booléen) du modèle. Dans ce cas, le mécanisme de contrainte spécifique implémente de manière la contrainte arithmétique classique  $X_1 = X_2$ . A ce moment, il peut être intéressant d'inclure cette contrainte devenue linéaire dans le programme linéaire (PL) courant. Là encore la coopération entre ILOG SOLVER et ILOG PLANNER permet de définir un postage au simplexe lorsqu'une contrainte sous hypothèse devient linéaire. Ce mécanisme est bien sûr réversible ; en effet, les conditions de linéarité dépendent de l'activation des deux booléens et le retour à la non linéarité doit se traduire par une « sortie » de la contrainte du PL. La notion de *booléen réversible* est également utilisé pour indiquer que la partie linéaire doit être postée une et une seule fois même si la propagation est appelée plusieurs fois (c'est le cas en général).

## C) Heuristique de l'optimisation globale

L'heuristique est basée sur une adéquation entre le « pattern » des modèles et les durées d'activités. Le paramétreur joue un rôle important dans cette correspondance lorsqu'il indique sur un modèle les propriétés qui relie celui-ci aux activités qu'il couvre. Plus le modèle est précis, plus son activation impacte les contraintes liées aux activités (notamment celles de charge). La partie des contraintes spécifiques conduit à « garder en mémoire » sous forme de contraintes – floues au départ et de plus en plus précises au fur et à mesure de l'instanciation des modèles – un ensemble de relations qui communiqueront et guideront les branchements ainsi que le problème linéaire.

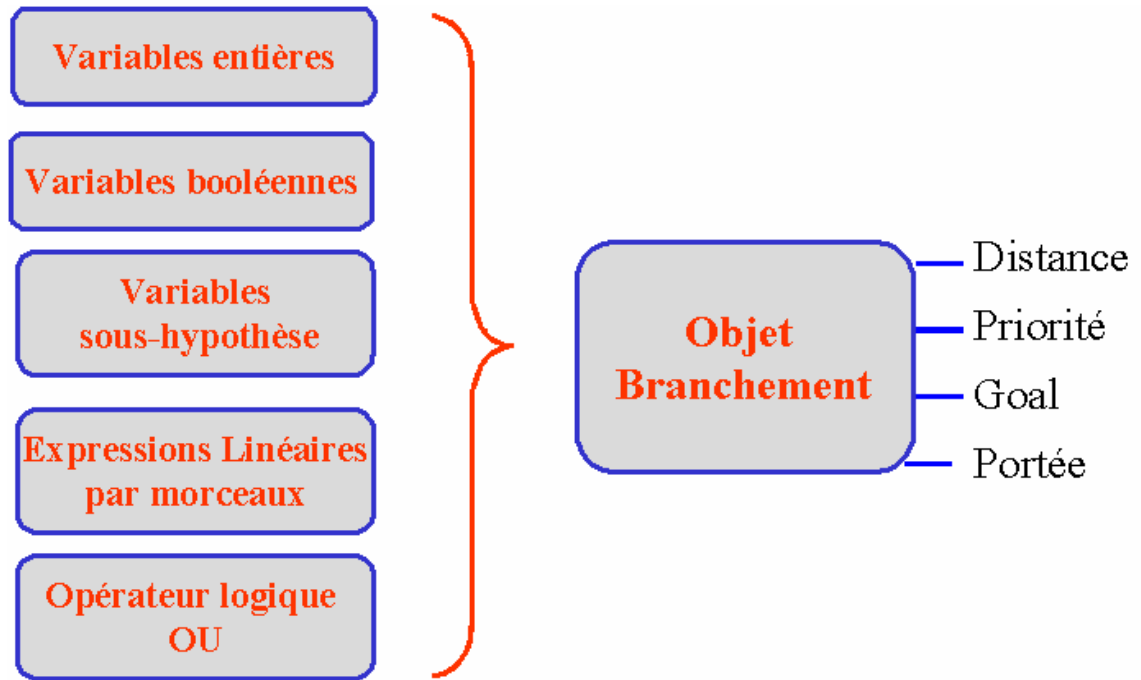


*coopération modèle linéaire/modèle non linéaire (sous hypothèse)*

Le programme linéaire est initialement construit avec les contraintes complètement linéaires liées aux variables d'activités, notamment celles de charge mais aussi le respect des préférences ou l'équité de la couverture de charge. Comme on l'a vu plus haut, les relations durées/modèles, implémentées sous forme de contraintes spécifiques sous hypothèse, renseignent de manière réversible le programme linéaire. A noter que dans cette coopération, **le simplexe maintient une solution optimale**. C'est à dire que toute modification de domaine d'une des variables ou ajout/retrait d'une contrainte est testée au regard de la précédente solution optimale. Si celle-ci n'est plus réalisable, une nouvelle solution est recalculée automatiquement par le simplexe.

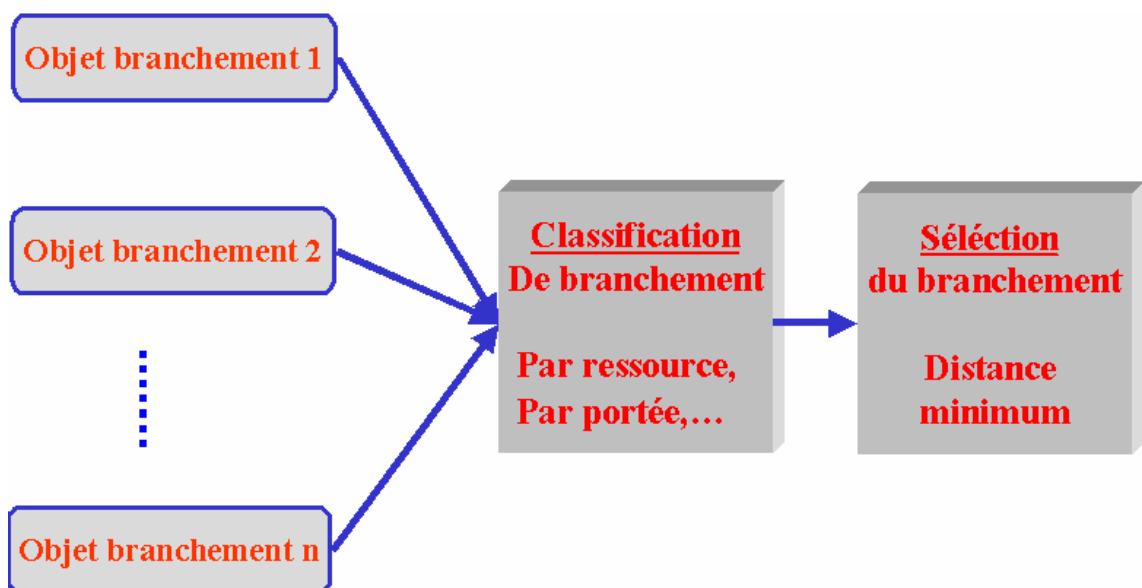
Cette coopération du modèle linéaire et du modèle sous-hypothèse participe pleinement à l'heuristique en ce sens qu'elle permet de représenter de manière incrémentale l'ensemble des contraintes du problème et toutes les relaxations linéaires induites par les modèles choisis. Le modèle linéaire et le modèle non linéaire s'enrichissent donc « en continu » du maximum d'informations disponibles.

Typologie des branchements : Outre les variables de modèles – booléennes et entières, d'autres expressions non linéaires peuvent être utilisés dans des compteurs et contraintes : expression d'une contrainte comme un choix de contraintes, expressions linéaires par morceaux. L'« objet branchement » a donc été introduit afin d'encapsuler les différentes typologies de branchement et de définir des stratégies dynamiques de branchement.



*typologie des branchements*

Stratégies de branchement : il est possible de combiner les différentes caractéristiques des branchements pour définir la stratégie générale. Désignons respectivement par « D », « I », « P » et « R » les indicateurs de distance, portée temporelle, « priorité » ET « ressource individuelle ». Ainsi, la stratégie « PI », sélectionnera entre deux branchements non instanciés à un instant de la recherche celui de priorité la plus forte, et si les deux objets avaient la même portée, celui de portée temporelle la plus étendue (typiquement un modèle de semaine avant un modèle de jour).



*stratégie de sélection d'un objet de type branchement*

Les différentes stratégies actuellement proposées aux paramétreurs sont :  
"PI" (stratégie par branchement), "R", "RPI", "RIP", "IP".

Le branchement par ressource est souvent utilisé. Il permet de calculer une solution correspondant à toutes les variables d'une seule personne avant de passer à la suivante.

## **D) Résultats**

La recherche arborescente ainsi implémentée donne des résultats satisfaisants dès la première solution instanciée pour des problèmes de taille relativement importantes : quelques dizaines de personnes, arbres de modèles relativement complexes (deux portées et quatre niveaux de profondeur), contraintes en lignes et en colonnes.

### *D.1 Deux exemples numériques*

Nous examinons deux résultats correspondant à une annualisation et à une optimisation hebdomadaire.

Pour un problème d'annualisation (52 semaines, la maille est la semaine) sur 7 personnes et 26 activités, la traduction génère un problème à :

9835 variables

2100 contraintes linéaires

3160 contraintes non linéaires

1586 branchements (essentiellement binaires)

La construction de ce modèle consomme 107 secondes et environ 13 Mo

La première solution entière est obtenue après 158 points de choix (instanciation de branchements) dont 0 échecs et 32 secondes.

Pour un problème de portée une semaine à la maille jour, le placement de modèle inclut des changements de portée donc des branchements sur des valeurs entières (par exemple quel jour dans la semaine commence le modèle Jour de Repos ?). L'exemple étudié incluait également des expressions linéaires par morceaux, implémenté comme branchements internes par ILOG SOLVER :

18 activités et 9 ressources

nombre de contraintes linéaires = 456

nombre de contraintes non linéaires = 506

7514 variables

Nombre de branchements entiers = 473

Nombre de branchements binaires  $\{0,1\} = 224$

Nombre d'expressions linéaires par morceaux = 88

La première solution entière est obtenue après 29 points de choix dont 0 échec.

## *D.2 Gérer la combinatoire*

L'« explosion combinatoire » lié à la taille du problème (approximativement nombre de contraintes \* nombre de ressources \*  $2^{\text{profondeur de l'arbre des modèles}}$ ) entraîne parfois des dépassements de capacité mémoire. Mais dans ces cas, la stratégie « branchement par ressource » permet de limiter le nombre de points de choix (branchements) même si cette stratégie est en théorie plus restrictive qu'un parcours de l'arbre où l'ordre des variables sur lesquelles brancher est libre.

TempoSoft a également implémenté un mécanisme de « compilation » des modèles qui permet de traiter l'augmentation de la complexité due à des changements de portée. En résumé, on garde dans l'optimisation les modèles de jours apparaissant dans des modèles de semaine. Un « compilateur de contraintes » propage alors les relations entre ces modèles et leur descendance.

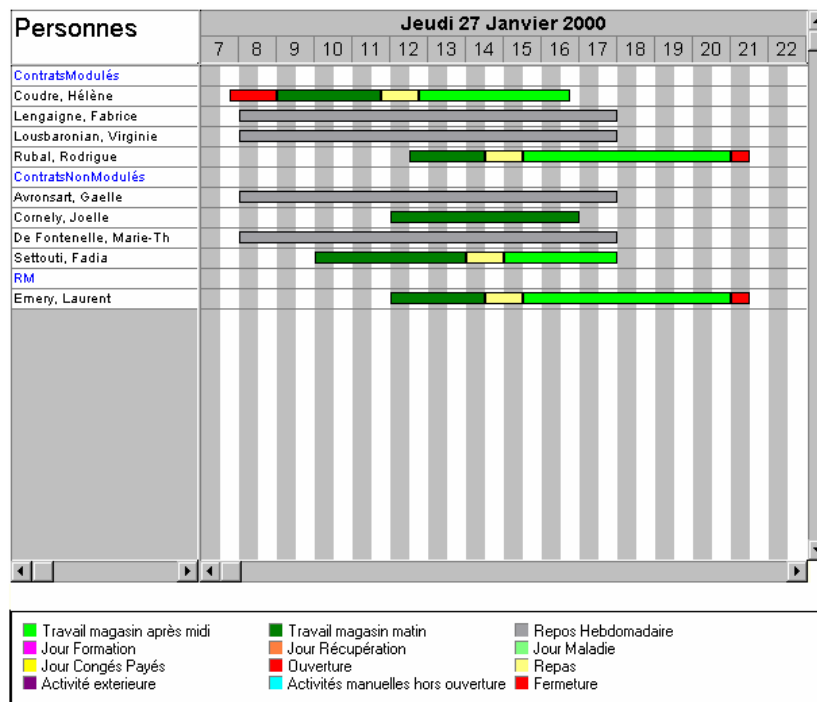
En quelque sorte, le branchement par ressource est très efficace pour trouver une première solution lorsque une stratégie plus vaste « piège » l'optimisation. D'autres mécanismes sont envisagés pour les prochaines versions du produit pour empiler-dépiler les « goals » de branchement d'une manière moins gourmande en mémoire.

N'oublions pas cependant que les clients de TempoSoft sont généralement plus sensibles à la rapidité d'obtention d'une solution - si tant est que celle-soit acceptable (couverture de charge, équité et contraintes inviolables) – qu'à l'optimalité de celle-ci. La modélisation de la fonction de coût par notamment des notions de priorité de charge sur les activités est en effet une représentation assez arbitraire – faute de mieux ! – des activités réalisées par les employés.

## IV. L'Optimisation « Détaillée », une approche locale

### A) Rappel des objectifs

L'optimisation dite « détaillée » est lancée sur un horizon plus réduit (par exemple la semaine) et descend jusqu'à une maille beaucoup plus fine (par exemple le quart d'heure). On s'intéresse alors au placement des activités d'une personne au sein même de la journée, en respectant l'intégralité des contraintes de modèles. Des contraintes spécifiques de choix et séquençage des activités au sein des « modèles feuilles » doivent ainsi être rajoutées aux contraintes de modèles et contraintes utilisateurs.



*Exemple de planning détaillé portant sur une journée et de maille 15 minutes*

### B) Heuristique

#### *B.1 Schéma général*

Le planning à optimiser peut être vu comme un damier de couple (jour, personne).

		Jours						
Personness								

- |   |  |
|---|--|
| <ul style="list-style-type: none"> <li>• Choix de fenêtre</li> <li>• Définition dynamique du sous-problème:             <ul style="list-style-type: none"> <li>• Charge différentielle</li> <li>• Sous-ensemble de contraintes</li> </ul> </li> </ul> | <ul style="list-style-type: none"> <li>• Commencer avec la solution vide</li> <li>• S'arrêter quand on ne peut plus améliorer</li> </ul> |
|---|--|

Chacun de ces couples est optimisé individuellement, mais en tenant compte des valeurs existantes dans les autres cases du damier. L'algorithme de l'heuristique se présente ainsi :

*Tant que l'on peut améliorer la solution, on choisit un couple (jour, personne)*

*On vide la liste des affectations existantes pour le couple sélectionné*

*On propage les contraintes existantes au sein du couple et déduites des valeurs des autres couples qui sont figés.*

*On affecte une activité à la personne pour chaque instant de la journée*

*Fin tant que*

Il est possible de rendre l'optimisation moins locale en remplaçant le couple (jour, personne) par un couple (groupe de jour, personne). On parle en fait de fenêtre d'optimisation, c'est à dire d'intervalle sur lequel on optimise réellement, tout ce qui est en dehors de cette fenêtre étant figé. La durée de cette fenêtre est paramétrable, l'unité étant un nombre de jours. Par exemple, si on optimise une semaine avec une fenêtre d'optimisation de 2 jours, on va optimiser réellement toutes les séquences de 2 jours consécutifs de la semaine (lundi - mardi, mardi - mercredi, etc..). Pour des problèmes dits 24/24 où le travail de nuit est autorisé, il est conseillé d'étendre la fenêtre d'optimisation à deux jours pour que la fenêtre d'optimisation ne « tronque » pas artificiellement des contraintes qui intersectent deux jours.

### *B.2 Sélection gloutonne d'une fenêtre*

La stratégie de recherche d'une fenêtre locale (un couple <jour, personne>) à améliorer est basée sur une évaluation du coût de chacune des fenêtres restantes. On choisit le couple <jour, personne> dont cette évaluation est la plus forte, c'est à dire celui

susceptible de la plus forte amélioration. L'évaluation est basée sur le coût des contraintes qui interceptent la fenêtre : contraintes individuelles « en ligne » et contraintes de charge différentielle en colonne. C'est une évaluation grossière puisqu'elle ne prend pas en compte le séquençement des activités au sein de la journée.

Lorsqu'on choisit un couple qui a déjà été choisi, le sous problème qui est généré n'est pas du tout le même que la première fois, car les valeurs figées correspondant aux couples voisins ne sont plus du tout les mêmes. Il s'agit d'une ré-optimisation locale, qui permet d'améliorer la solution.

### B.3 Amélioration locale au sein de la fenêtre sélectionnée

Une fois une fenêtre d'amélioration choisie, on définit un problème d'optimisation dont les variables de décision sont uniquement dans cette fenêtre. Toutes les contraintes croisant la fenêtre se réécrivent en fait naturellement comme :

Partie Constante (extérieure à la fenêtre) + Partie Variable (au sein de la fenêtre)

Ainsi, les contraintes de charge deviennent différentielles : on compare le temps effectué par la personne à améliorer au sein du jour choisi à *ce qui reste* de charge à couvrir.

La première étape est là encore une évaluation du « pattern » des modèles afin de choisir dans l'arbre des modèles un chemin qui semble le mieux approprié. , c'est à dire le choix d'un chemin dans l'arbre des modèles. Ce chemin instancie à 1 les booléens de modèle mais ne choisit pas les débuts et fin des modèles. Ceux-ci seront en effet directement renseignés par la pose directe des activités sur le planning, par le biais de contraintes spécifiques entre les feuilles des arbres (les modèles tâches sur lesquels on indique les groupes d'activités) et les activités du planning de la personne.

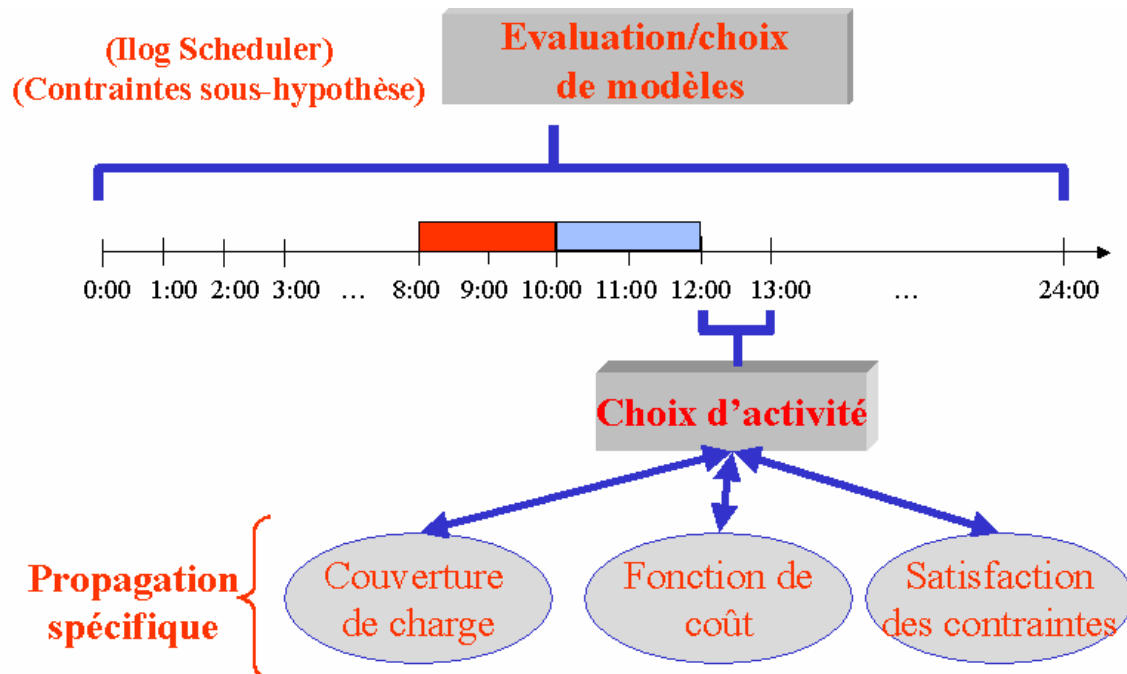


Schéma de principe de l'amélioration locale pour un couple <personne,jour>

La pose des activités est décidée de manière chronologique et consiste à placer « slot par slot » (le slot désignant la maille de l'optimisation, défini par rapport à un diviseur commun des durées minimales de placement des activités) une activité. On choisit alors l'activité qui améliore la mieux la couverture de charge.

### C) Résultats

L'heuristique utilisée actuellement dans l'optimisation détaillée donne de bons résultats « en moyenne », c'est à dire qu'on sait correctement améliorer la plupart des fenêtres d'amélioration locale.

Mais il arrive parfois que l'heuristique d'amélioration locale ne puisse remplir un planning d'une <personne,jour>. Ceci est dû principalement à un « piège » dans l'heuristique de choix des activités ; celle-ci étant chronologique et basée sur la couverture de charge, il arrive que les placements d'activité en fin de journée deviennent impossibles. Pour tenter d'autres solutions, on revient ainsi au slot incriminé pour prendre une autre décision, et si cela échoue, on effectue cette-fois un backtrack sur 1 slot précédent et ainsi de suite jusqu'à trouver une solution acceptable. Le nombre de backtrack nécessaires peut dans certains cas excéder les limites définies par l'utilisateur.

De manière analogue, les contraintes temporelles de portée supérieure à la maille jour sont mal prises en compte dans l'optimisation détaillée.

## V. Conclusion

La modélisation des problèmes d'affectation et de couverture de charge sous contraintes est rendue grandement générique par le module d'optimisation TEMPOTOOLS. Pour rendre compte de cette généricité dans nos algorithmes, nous avons construit un langage de traduction de contraintes utilisateurs et une bibliothèque commune de contraintes spécifiques – notamment les contraintes qui définissent le pattern des modèles. Ces objets peuvent être intégrés dans différents types d'algorithmes, voire encapsulés dans de nouveaux modules qui généreraient par exemple des contrôles de saisie (l'utilisateur peut-il fixer manuellement un modèle sans violer les seules contraintes liées à ce modèle).

L'analyse des résultats et la discussion avec nos équipes de test, de consultants et les clients ont permis d'identifier un manque de correspondance entre le placement des modèles et l'impact de ce placement sur la couverture de charge. En mode global comme en mode détaillé, l'évaluation des choix de modèles n'est donc pas assez précise.

L'équipe d'optimisation s'est donc attelée à la mise au point d'un outil de prévision et d'évaluation de « fonctions de charge » associées aux modèles. Cet outil permettrait soit de remonter finement des informations de couverture de charge sur un modèle en fonction des estimations dynamiques des variables début et fin du modèle et de sa descendance. En mode global comme en détaillé, nous espérons de meilleures solutions – tant en qualité qu'en robustesse.

A terme, l'utilisateur pourrait également définir lui-même des probabilités de couverture sur des modèles définis non plus à un niveau individuel mais sur des équipes et envisager ainsi de modifier les échelles de l'optimisation. Ainsi, on pourrait définir un modèle sur un groupe de 100 personnes et calculer des optimisations visant à affecter des volumes horaires à plusieurs groupes de ce type.

Le travail accompli jusqu'ici permet d'envisager avec confiance et intérêt les prochains enjeux qu'aura à faire face l'équipe d'optimisation de TEMPOSOFT.

